UNIVERSITÀ CA' FOSCARI VENEZIA

DOTTORATO DI RICERCA IN INFORMATICA, 28° CICLO

(A.A. 2011/2012 – 2014/2015)

3D Acquisition and Analysis with Applications in Interaction and Contactless Measurement

SETTORE SCIENTIFICO DISCIPLINARE DI AFFERENZA: INF/01

TESI DI DOTTORATO DI LUCA COSMO
MATR. 807107

TUTORE DEL DOTTORANDO
prof. Andrea Torsello

COORDINATORE DEL DOTTORATO
prof. Riccardo Focardi

November 30, 2015

Author's Web Page: `http://www.dsi.unive.it/~cosmo`

Author's e-mail: luca.cosmo@unive.it

Author's address:

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
tel. +39 041 2348465
fax. +39 041 2348419
web: `http://www.dais.unive.it`

Abstract

Computer Vision has become a constant presence in everyday life in recent years. Thanks to the price reduction of apparatus and the increase of computational power, Computer Vision based techniques are the weapon of choice in various scenarios, from high accuracy 3D reconstruction and photogrammetry to the automatic analysis of video surveillance images and face recognition-based login systems.

Recent interest of the game industry in gesture-based interaction paradigms further fostered the presence of image sensing apparatus in our homes. This thesis is dedicated to the exploitation of this variety of low-cost devices to develop useful applications to be used in both the industrial field and home entertainment world. The accurate calibration of the image sensing devices is a fundamental step to allow further reasoning on the acquired scene.

The first part of the thesis by introducing some advanced camera calibration techniques effective in the calibration of low-end consumer cameras in which manufacturing imperfection makes them more likely to diverge from the common parametric models. Beside 2D images processing, the recent diffusion of depth-cameras, as the Mictosoft Kinect^R sensor, allows to exploit also three-dimensional data. This information is particularly useful when we need to recognize user's gesture, but its analysis introduces a new challenging problem: the Non-Rigid matching of shapes. In the second part of the thesis we tackle this problem with two method that try to retrieve respectively sparse and dense correspondence between two or more partial shapes.

In the last part we propose some applications exploiting all the notions and techniques introduced. We present three applications of Computer Vision techniques to the HCI world. Two tracking devices are developed to enable the definition of new interaction paradigms to be applied in different scenarios: an Interactive Table, an Interactive Witeboard and a Viewer Dependent display.

Sommario

Negli ultimi anni la Computer Vision è diventata una presenza costante nella vita quotidiana di ognuno di noi. Grazie alla riduzione del prezzo degli apparati e all'aumento della potenza di calcolo, le tecniche basate sulla Computer Vision sono diventate un ottimo strumento da applicare in diversi scenari, dall'acquisizione della struttura tridimensionale degli oggetti, alla fotogrammetria sino all'analisi automatica delle immagini di video sorveglianza e l'utilizzo in sistemi automatici di autenticazione basati sul riconoscimento facciale.

Il recente interesse del settore video-ludico in nuovi paradigmi di interazione basati su gesture recognition ha ulteriormente incrementato la presenza di apparati di acquisizione video (e non solo) nelle nostre case. Questa tesi è dedicata all'utilizzo di questa varietà di dispositivi a basso costo per lo sviluppo di applicazioni utili sia nel campo industriale che nel mondo dell'home-entertainment.

Il primo contributo di questa tesi riguarda lo sviluppo di alcune tecniche avanzate di calibrazione degli apparati di acquisizione, con particolare attenzione a dispositivi a basso costo che, a causa delle imperfezioni nella produzione, sono più propensi a divergere dai modelli parametrici comuni. Oltre ad immagini bidimensionali, la recente diffusione di camere di profondità, come per esempio il sensore Mictosoft Kinect^{R}, permette di acquisire insieme all'immagine anche infor-mazioni relative alla tridimensionalità della scena. Questi dati sono particolarmente utili quando dobbiamo riconoscere movimenti del corpo e gesti effettuati dagli utenti, tuttavia la loro analisi introduce un nuovo problema: la ricerca di corrispondenza tra forme "non rigide". Questo problema è affrontato nella seconda parte della tesi con due metodi che cercano di trovare delle corrispondenza sia sparse che dense tra due o più forme in presenza di parzialità.

Nell'ultima parte sono proposte alcune applicazioni che sfruttano le nozioni e le tecniche introdotte precedentemente per lo sviluppo di nuovi paradigmi di interazione mediante la progettazione di due dispositivi di tracking applicati in diversi scenari: un tavolo interattivo, una Witeboard interattiva e un Viewer Dependent display.

Published Papers

[1] Albarelli, A., Bergamasco, F., Celentano, A., Cosmo, L., and Torsello, A. Using multiple sensors for reliable markerless identification through supervised learning. *Mach. Vis. Appl. 24*, 7 (2013), 1539–1554.

[2] Albarelli, A., Celentano, A., and Cosmo, L. Evaluating stereo vision and user tracking in mixed reality tasks! In *2015 IEEE Symposium on 3D User Interfaces, 3DUI 2015, Arles, France, March 23-24, 2015* (2015), pp. 81–88.

[3] Albarelli, A., Celentano, A., Cosmo, L., and Marchi, R. On the interplay between data overlay and real-world context using see-through displays. In *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter, CHItaly 2015, Rome, Italy, September 28-30, 2015* (2015), pp. 58–65.

[4] Albarelli, A., Cosmo, L., and Bergamasco, F. Practical metrics for error assessment with interactive museum installations. *Handbook of Research on Interactive Information Quality in Expanding Social Network Communications* (2014), 70.

[5] Albarelli, A., Cosmo, L., Bergamasco, F., and Gasparetto, A. Objective and subjective metrics for 3d display perception evaluation. In *ICPRAM 2015 - Proceedings of the International Conference on Pattern Recognition Applications and Methods, Volume 2, Lisbon, Portugal, 10-12 January, 2015.* (2015), pp. 309–317.

[6] Albarelli, A., Cosmo, L., Bergamasco, F., and Sartoretto, F. Phase-based spatio-temporal interpolation for accurate 3d localization in camera networks. *Pattern Recognition Letters 63* (2015), 1 – 8.

[7] Albarelli, A., Cosmo, L., Bergamasco, F., Sartoretto, F., and Torsello, A. A 5 degrees of freedom multi-user pointing device for interactive whiteboards. *Pattern Analysis and Applications* (2015), 1–14.

[8] Albarelli, A., Cosmo, L., Bergamasco, F., and Torsello, A. High-coverage 3d scanning through online structured light calibration. In *22nd International Conference on Pattern Recognition, ICPR 2014, Stockholm, Sweden, August 24-28, 2014* (2014), pp. 4080–4085.

[9] Albarelli, A., Cosmo, L., and Celentano, A. Evaluating accuracy of perception in an adaptive stereo vision interface. In *International Working Conference on Advanced Visual Interfaces, AVI' 14, Como, Italy, May 27-29, 2014* (2014), pp. 333–334.

[10] Bergamasco, F., Albarelli, A., Cosmo, L., Torsello, A., Rodola, E., and Cremers, D. Adopting an unconstrained ray model in light-field cameras for 3d shape reconstruction.

[11] Bergamasco, F., Cosmo, L., Albarelli, A., and Torsello, A. A robust multi-camera 3d ellipse fitting for contactless measurements. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission, Zurich, Switzerland, October 13-15, 2012* (2012), pp. 168–175.

[12] Bergamasco, F., Cosmo, L., Albarelli, A., and Torsello, A. Camera calibration from coplanar circles. In *22nd International Conference on Pattern Recognition, ICPR 2014, Stockholm, Sweden, August 24-28, 2014* (2014), pp. 2137–2142.

[13] Cosmo, L., Albarelli, A., and Bergamasco, F. A low cost tracking system for position-dependent 3d visual interaction. In *International Working Conference on Advanced Visual Interfaces, AVI' 14, Como, Italy, May 27-29, 2014* (2014), pp. 351–352.

[14] Cosmo, L., Albarelli, A., Bergamasco, F., and Torsello, A. Design and evaluation of a viewer-dependent stereoscopic display. In *22nd International Conference on Pattern Recognition, ICPR 2014, Stockholm, Sweden, August 24-28, 2014* (2014), pp. 2861–2866.

[15] Cosmo, L., Rodola, E., Albarelli, A., Memoli, F., and Cremers, D. Consistent partial matching of shape collections via sparse modeling. *Minor revision @ CGF* (2015).

[16] Rodolà, E., Cosmo, L., Bronstein, M. M., Torsello, A., and Cremers, D. Partial functional correspondence. *Minor revision @ CGF abs/1506.05274* (2015).

Preface

Before starting I want to briefly introduce myself, my experience during these three years as PhD student and better specify my contribution in the works presented hereafter.

I started my PhD three years ago in the field of Computer Vision. I was lucky to become part of a very active and dynamic group. Since the first days of my PhD I was involved in their research activity. It has been - and it still is - a very stimulating relationship where intense activity periods was interleaved to hilarious moments and side lighthearted projects. This greatly contributed in improving my experience inside this splendid research group.

As the reader probably noticed from my Publications List, working inside a team means that everyone is involved on most of the group's projects and this results in a fairly high number of publications. Just to be correct with the reader and my teammates I want to briefly explain my contribution in the aforementioned publications.
There are projects in which I was involved on the way and others in which I play a more important role. My publications can be sub-dived into three categories: camera calibration, interaction, and non-rigid matching.

The first category was the main interest of the CV research group when I first joined. Although in some of these works I played a marginal role, I gave a relevant contribution in the projects regarding the *model free camera calibration*. Specifically, my contribution to the work presented in sections 3.1 [32] and 3.2 [11] concerns mainly the experimental section. I put them on this thesis since they was my first experience in the field of Computer Vision. In the next calibration methods, described in sections 4.1 and 4.2 [8, 10] about the *model free camera calibration*, I worked to adapt the model free calibration technique to the particular setups and on to elaborate an outlier detection procedure.

Besides the camera calibration topic, my research activity here in Venice was mainly focused on the HCI from a Computer Vision point of view, and it is indeed to this research area that most of my publications belong [1–7,9,13,14]. In all these works, some of which are described in chapter 8, my contribution concerns more the Computer Vision aspects of the user tracking than the study of the interaction paradigms adopted.

Finally, the two works about Non-Rigid matching [15,16] described in the second part of this thesis are the result of a collaboration with the TUMunich Computer Vision Group led by Prof. Dr. Daniel Cremers and they are two works of which I'm proud. I spent four exiting months in Munich working side by side with Emanuele Rodolá who has revealed himself - once again - to be an excellent mentor and an even more awesome beer drinker!

I would like to thank all of my team mates for their guidance and support during these three years and I'm glad to had Prof. Andrea Torsello as my supervisor for he has been always present when we needed him for advice and he always supported us in our ideas and scientific interests. It goes without saying that I am also grateful to all my friends - because study is not all in life - and my family for their moral and financial support during this long and fruitful period of my life lasted for almost 30 years.

Contents

1
Introduction

Computer Vision has become a constant presence in everyday life in recent years. Thanks to the price reduction of apparatus and the increase of computational power, Computer Vision based techniques are the weapon of choice in various scenarios, from high accuracy 3D reconstruction and photogrammetry to the automatic analysis of video surveillance images and face recognition-based login systems. Recent interest of the game industry in gesture-based interaction paradigms further fostered the presence of sensing apparatus in our homes. Nintendo was the first company to make of his famous wireless motion controller (Wii Remote) the battle horse of his gaming console. Sony and Microsoft immediately answered respectively with the *PlayStation Move*™ and the Microsoft Kinect®. Of particular success has been the later, probably thanks to the absence of a specific device to be held by the users. It is indeed based on a depth sensor from which the user's skeleton is retrieved allowing to recognize a wide range of user gestures. Recently, many others embedded device to track user's position and/or gestures have been introduced into the market (e.g. Leap Motion™, Creative Senz3D™, Xtion PRO™, etc.). The availability of such low-cost tracking systems together with the development of innovative visualization devices (the immersive steroscopic display Oculus Rift® to cite one over all) has started a completely new era of the consumer entertainment.

When we deal with systems based on the elaboration of data acquired by image sensing devices a mandatory step in almost all these setups is the recovery of all the parameter that determine the inner working of the imaging system. For instance, 3D reconstruction, photogrammetry and in general every process that involves geometrical reasoning on objects deeply rely on such knowledge. One of the most straightforward method to be able to accurately describe the acquisition process is to build a mathematical model able to approximately define the optical behavior of rays entering the camera through the optics and hitting the sensing image plane (CCD or CMOS in digital cameras). The Camera Calibration task consists in retrieving all the parameters of the adopted mathematical model to better reflect the characteristic of a specific setup. It is usually performed by showing to the camera (or cameras) a Fiducial Marker, that is a well known object with a particular geometrical structure that allows it to be easily identified inside the acquired scene. Exposing several times the Fiducial Marker to the cameras in different positions allows to collect enough information to be able to fully constraint the model parameters. The simplest and widely diffused camera model is the Pinhole Model. In the Pinhole

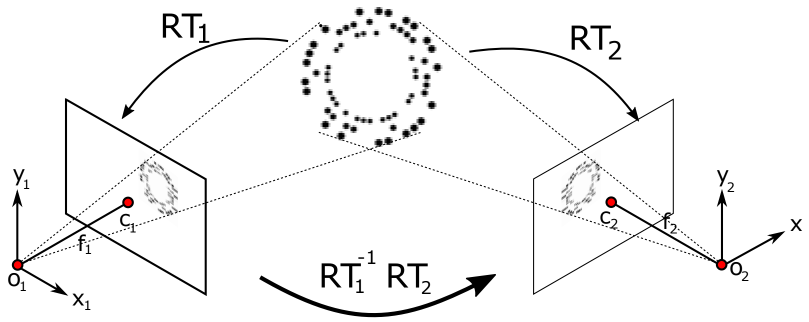
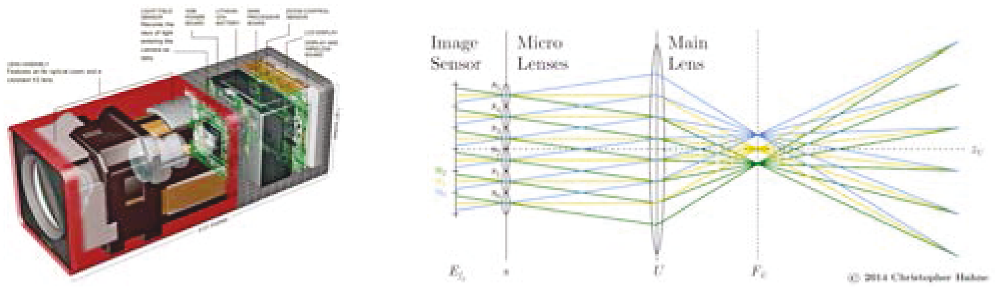


Figure 1.1: A simple camera network setup composed by two pinhole cameras. The reprojection of a planar RuneTAG Fiducial Marker onto the image plane of the two images is fully determined by the intrinsic parameters of the two cameras and their position with respect to the object.

Model (see figure 2.1) the light rays hitting the CCD are assumed to pass through the same physical point o, namely the optical center. The distance between this point and the *image plane*, where the image is projected, is the *focal length* f. With these few parameters we can describe the acquisition process of a single camera, moreover by introducing a roto-translation parameter we can describe more complex setups composed by two or more cameras (see Figure 1.1).

To this kind of approaches belongs the method described in section 3.1. We propose a novel Fiducial Marker based on circular features disposed in concentric rings. It is designed to allow to use different amounts of dots and, thanks to a cyclic code encoding schema, it is robust to the occlusion of a large amount of the marker surface. The combination of these two characteristic makes this marker suitable in different situations, from the Camera Calibration (in its denser version), to Augmented Reality applications.

Unfortunately, the mathematical model adopted is not always able to accurately describe the behaviour of the rays entering the camera. This is particularly true when we deal with low-cost hardware where the optics quality and the construction process are of low-quality. To tackle this problem in Chapter 4 we explore the potentiality of the unconstrained camera model introduced in [33]. We claim that [33] is indeed a powerful tool to describe an optical behavior diverging from the Pinhole model. In section 4.1 we apply the unconstrained model to provide an online calibration of the projector of a Structured Light 3D Scanner. Thanks to the robust outlier technique introduced we are able to increase both coverage and accuracy of the resulting acquisition. The recent commercialization of the new Lytro$^{\text{TM}}$ light-field camera brought also the plenoptic technology (see Figure 1.2)into the consumer market. In light-field cameras the central assumption is dropped, this makes them an ideal test case for a fully unconstrained model. In section 4.2 we investigate the ability of such a calibration technique to deal with this kind of cameras. Finally, in section 4.3 we step back to the centrality assumption and propose



(a) (b)

Figure 1.2: Left: A schematic rapresentation of the Lytrocamera. Right: The scene projection process on a plenoptic camera based on array of micro lenses.

a non-parametric distortion model for central cameras. Albeit being less accurate with respect to the fully unconstrained model, this calibration techniques allows us to exploit the complete set of mathematical tools derived from projective geometry (e.g. line and conic invariant, epipolar geometry, etc.).

Although it is very important to accurately calibrate the model describing the optical behavior of our camera network setup, it is only a preliminary step. The core of a Computer Vision application lies in the processing of the acquired data. It is a task strictly dependent on the application, it could span from the accurate estimation of an acquired object dimensions, as in a photogrammetry application (section 3.2), to the reconstruction of the three-dimensional structure of a object in order to acquire its model or to perform further reasoning on its shape. From the point of view of an interaction application some of the most important tasks concern the object tracking and the estimation of user's pose and gestures. Gesture tracking can be performed in different ways: by using specially crafted devices to be held by users, as we will see later, or by acquiring the body position of a part or the whole user's body and trying to reconstruct its movements. Microsoft Kinect® sensor, for instance, adopts this later approach by retrieving

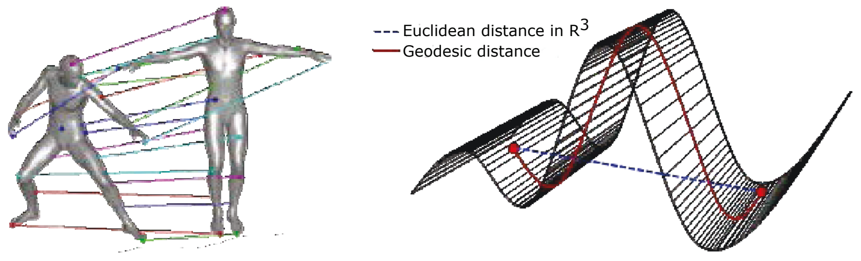


Figure 1.3: Left: an example of a Non-Rigid matching between two human bodies. Right: difference between the geodesic distance on the manifold and the euclidean distance in the embedding space $\mathbb{R}^3$.

the human body skeleton from the acquired depth-map. If skeleton extraction is one of the possible uses of the depth map, far more interesting is the retrieving of the complete body surface. However, conversely from a rigid transformation which is the typical case in three dimensional scanners, when we observe a human body moving we have to deal with its non rigidity, that is, the bending of human arts doesn't preserve the Euclidean distance between two points on the surface (see Figure 1.3). This makes finding matching points over the surface very difficult. In section 6.1 we propose a method to find sparse correspondences among a collection of shapes that, thanks to a L1 optimization cost, is very robust to outliers in the collection and allows for shape partiality. In section 6.2 we try to find a dense correspondence between a full shape and a partial one. This case is particularly important since it represents a possible scenario in the scanning of a human body. We cast the matching problem into the framework of Functional Maps [171] and we exploit some properties of Laplacian decomposition under the presence of partiality to better constrain the optimization of the functional map between the two shapes. At the end of the section we also investigate some ideas to extend this method to the partial vs. partial case which would lead to the ability to reconstruct a human body (or any other quasi-isometric deformable shape) from a series of depth-images without requiring the subject to stand still.

The ability to track user pose and gestures enabled the HCI community to develop new interaction paradigms to interact with computer systems in a more natural and intuitive way. One thriving field of application of such techniques is the so called *Interactive Tables* (Figure 1.4). They have proved to be a viable system to foster user participation and interest in many shared environments, among which educational and cultural environments such as museums and exhibitions have a leading role. They favor interaction among users and induce a sort of serendipitous discovery of knowledge by observing the information exploration performed by other users. Their use has been analyzed and evaluated in entertainment as well as in educational applications [25, 67, 113, 119].

In the educational perspective of HCI applications an important role is played by *Interactive Whiteboards*. The term *Interactive Whiteboard* (IWB) usually refers to a wide class of hardware and software bundles that share the common goal of serving as a techno-

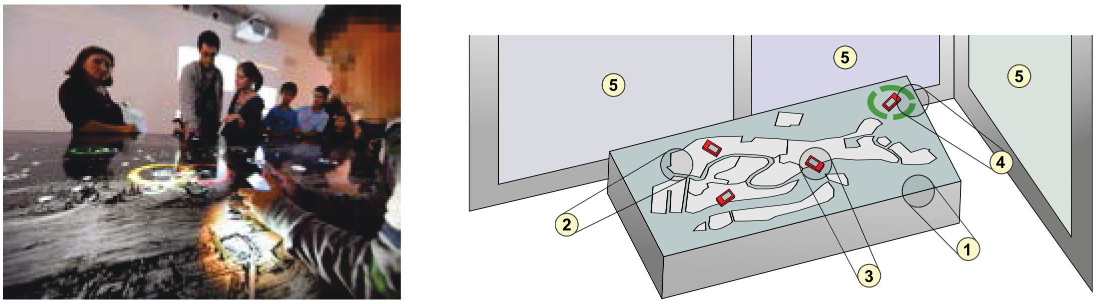
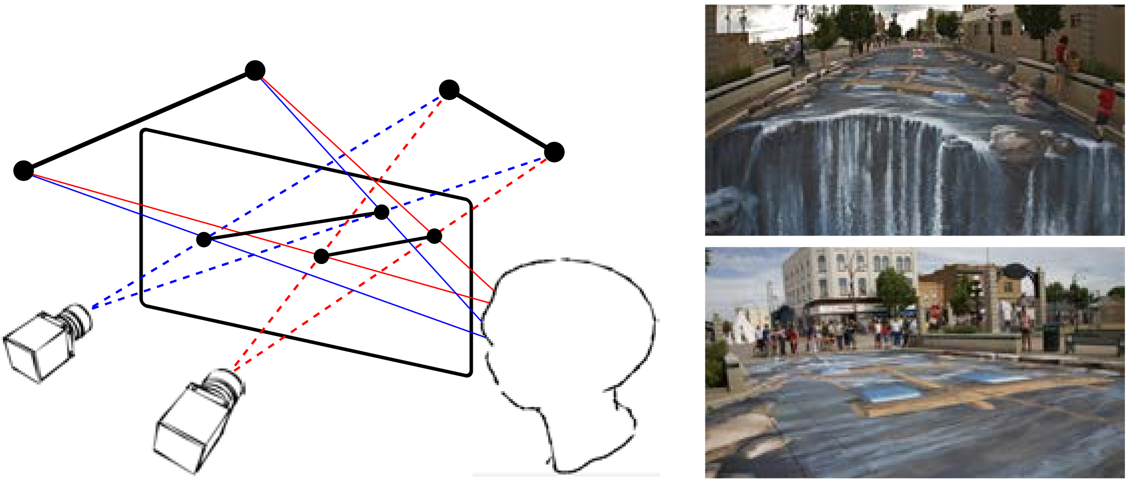


Figure 1.4: Left: A group of people operating on the map based multiuser art browser described. Right: Schematic representation of the components of the proposed multiuser interactive table.

logical substitute for traditional blackboards (or flipcharts) in both offices and educational settings. Some studies find them to be a key component to enhance the performance of both teachers and learners in the classroom [141]. Other surveys are more critical, as they expose the limits of current implementations [200, 201]. Anyhow, their increasing adoption spurs the interest of both researchers and practitioners in the design of innovative supporting technologies, interaction models and teaching practices [198].

It is interesting to analyze the advantages of being able to track the user's position also from a visualization point of view rather than limiting to gesture tracking for input purposes. Knowing the user's head position allows to provide a Viewer Dependent rendering of the scene. Viewer-dependent displays have been extensively proposed in recent scientific literature, since they offer many advantages. For starters, they are able to guarantee that the viewed objects exhibit a correct size within the Euclidean space where the user resides, thus allowing to interact with them naturally and to make meaningful comparisons between virtual and physical prototypes. The correct dimensional perception of virtual objects can be further increased by the adoption of a stereoscopic rendering. This implies knowing the position of each on the two eyes of the user in order to provide the correct rendering (see Figure 1.5).

In chapter 8 we apply the proposed techniques in the Human Computer Interaction field in order to track user's pose and gestures. Specifically, in section 8.1 we explain the setup of a big Interactive Table build for a museum exhibition. Thanks to both an



(a)

(b)

Figure 1.5: Left: The stereo inconsistency problem. Any stereo pair, when observed from a location different from the position of the capturing cameras, will result in impaired perception. Under these condition any observer will see an unpredictably distorted 3D object. Right: images from a work by Artist Edgard Muller. The simulated perspective only works if the observer stands in a very specific observation point (top).

image based tracking and sensor (accelerometers) analysis of an active device, we are able to handle multiple users in the navigation of some map-based content. Section 8.2 describes the implementation of a 5 degrees of freedom pointing device and its use in Interactive Whiteboard applications. The device is composed by two LEDs pulsating with a sinusoidal patterns. While the device position is tracked in a frame-by-frame basis by a camera network, the device identification and orientation disambiguation is entrusted to a time domain analysis of the LEDs intensity. This allows us to handle multiple users still maintaining the device geometry simple in order to be recognized in difficult visibility conditions (e.g. occlusions, distance from the camera). The device performance are further increased in section 8.3 where we introduce an interpolation schema to provide a better synchronization of the cameras involved in the network, and thus a higher accuracy of the triangulated point.

Finally, in section 8.4 we use the same tracking device, embedded in a pair of shutter glasses, to track the user head position and provide him with a viewer dependent stereoscopic rendering of the scene. This application is followed in section 8.5 by an interesting analysis of the human perception of stereoscopics. This study investigate the ability of humans of perceiving sizes in a virtual reality scenario where the virtual scene is mixed with the real background.

I

Camera Calibration

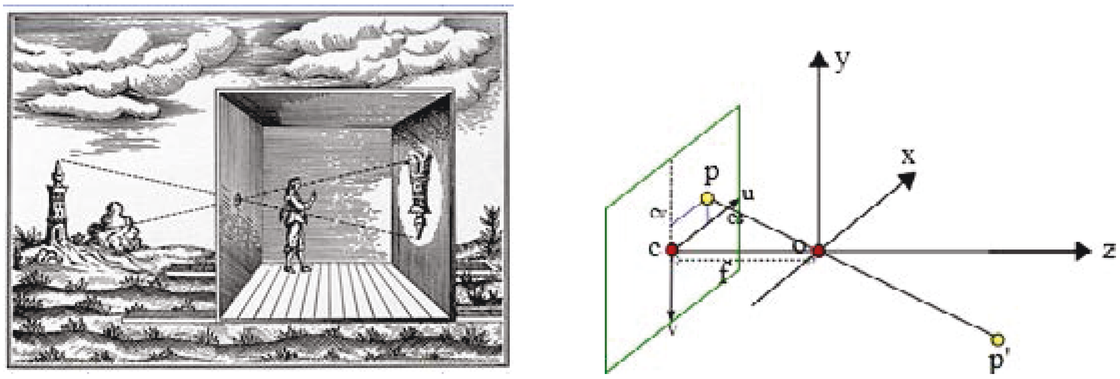
2
Related Work

The recovery of all parameters that determine the inner working of an imaging device is often a mandatory step to allow further analysis of a scene. For instance, 3D reconstruction, photogrammetry and in general every process that involves geometrical reasoning on objects deeply rely on such knowledge. The first step in describing the imaging system of a camera is to build a mathematical model trying to capture (and unavoidably simplify) the optical behavior of the rays entering the camera and hitting the optical sensor (CCD or CMOS). In the following we will review some most diffuse camera models, starting from the simpler Pinhole Camera model and the most known image undistortion techniques. We will see how we can exploit some *projective geometry* [106] invariants of the pinhole model to estimate with high accuracy the parameters of such models using Fiducial Markers.
After an overview of some particular camera models for which the Pinhole Model is not suitable, we will abandon any claim to build a camera specific imaging model and move the attention toward the unconstrained calibration technique in which each (discretized) camera ray is calibrated independently.

2.1 Pinhole model

For many applications, it is a good assumption to consider the camera to behave according to the *Pinhole Model*. The simple geometrical principles of this camera model are known since antiquity. Written references to the operation of the camera *obscura* can be found in the writings of Mozi (470 to 390 BC) and later in the works of the Greek philosophers and mathematicians Aristotele and Euclid [17]. In the Pinhole Model (see figure 2.1) the light rays hitting the light sensitive plane (CCD or CMOS in digital cameras) are assumed to pass through the same physical point o, namely the optical center. The distance between this point and the *image plane*, where the image is projected, is the *focal length* f. From these two simple entities, image plane and optical center, we can build two reference systems:

- the 3D reference system of the camera world, which origin corresponds to the optical center, the z axis, also known as *optical axis*, is perpendicular to the image plane and x and y axes are chose so to be aligned with the pixel grid.



(a) (b)

Figure 2.1: Left: A Camera Obscura illustration from the seventeenth century. Right: a schematic of the projective geometry quantities involved in the Pinhole camera Model.

- the 2D image plane reference system, with x and y axes aligned with the pixels grid (thus coherent with the camera word axis), and the center (c_x, c_y) set to the intersection between the plane and the optical axis.

The projection of a point placed at coordinates $p' = (x, y, z)$ in the 3D camera reference system onto the point $p = (u, v)$ of the image plane is described by the equation:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \qquad K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

where K is called the *intrinsic camera matrix* and λ is a term added to linearize the projection operation which is not linear by nature. Moreover the focal length can assume different values for the horizontal and vertical directions, respectively f_x and f_y, in order to allow for non squared pixels of the CCD.

More generally, if we have a 3D point with respect to the ”real” world, we need to bring him into the camera reference system by applying the proper rotation R and translation T. Given a 3D point (x, y, z) expressed in affine coordinates, its projection onto the image plane is thus governed by the liner equation:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.2}$$

2.1.1 Distortion

The Pinhole Model, albeit being very simple, is very widely adopted since it allows to leverage very powerful mathematical tools like the epipolar geometry and the projective

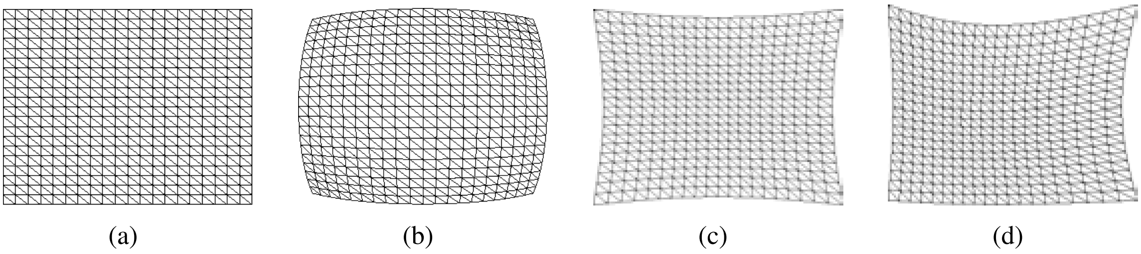

(a) (b) (c) (d)

Figure 2.2: Representation of the effect of the distortion on the projected point position. In the barrel distortion (b) point on the image are displaced outward with respect to the image center. This distortion effect is typical of wide angle optics. Conversely, with pincushion distortion (c) points are displaced toward the image center. (d) shows an example of tangential distortion.

invariance of lines and conics [106].
With cheap or wide optics, however, it is unable to accurately describe the image acquisition process of the camera since the image distortion phenomena comes into play.
The image distortion introduced by the lenses results in a displacement $\delta = (\delta_u, \delta_x)$ of the reprojected 3D point onto the image plane. This displacement can be expressed as a function of the ideal reprojection position itself: $(u_d, v_d) = (u, v) + \delta(u, v)$.

Main sources of distortion can be identified in both the lens manufacturing, and improper lens and camera components assembly. The main distortion effects can be subdivided in two categories:

Radial distortion : it is caused by an inaccurate radial curvature of lens. The effect is the displacement of the image points toward or away from its center (see Figure 2.2b 2.2c). The displacement amount is usually related to the distance of the point from the image center, as in the polynomial model proposed by Tsai [218].
Let $r_{u,v} = \sqrt{u^2 + v^2}$ be the radial distance of the point (u, v), Tsai approximates the displacement of the point by a polynomial with two non zero coefficients $(k1, k2)$ for the second and fourth degree term:

$$\delta_u = (k_1 * r_{u,v}^2 + k_2 * r_{u,v}^4)\frac{u}{r}, \quad \delta_v = (k_1 * r_{u,v}^2 + k_2 * r_{u,v}^4)\frac{v}{r} \tag{2.3}$$

Tangential distortion : albeit various work in literature [50, 218] demonstrated that accounting only for radial distortion is enough for many applications and optics setup, when high accuracy is sought other distortion sources need to be considered.
Misalignment of the optical centers of lens as well as slight tilt with respect to the sensing surface (CCD) can result in a mixture of radial and *tangential distortion*, see Figure 2.2d. This optical beahaviour was taken into account in the method proposed by Weng [234]. He proposed to enhance the model [218] introducing four coefficients accounting for lens misalignment and tilt.

More we move away from the pinhole model and the more the distortion assumes complex behaviors. Recently Claus and Fitzgibbon introduced theirs well-known rational model [68] which approximate projection displacement with a rational function and it's able to handle wide optics and in some extent also catadioptric cameras[1]. Other models have been proposed recently which differ in number and types of parameters [145, 229] or specifically designed to handle high distorted cameras [196, 210].

2.2 Calibration

Once the appropriate camera model has been chosen, we need to retrieve the parameters that better apply to the specific cameras and optics setup. If we aim at the best possible accuracy, it is universally considered a successful approach to rely on a known calibration target and exploit techniques that are able to recover camera parameters by exposing such object to the camera under different poses. A widely adopted technique is the one introduced in [251] based on a planar pattern composed by hundred of corner points for easy localization. Furthermore, Heikkilä [109] exploit a grid of circular points to accurately reverse the distortion model. An interesting variation on the theme is proposed in [20] with a technique that can simultaneously optimize camera and target geometry to overcome manufacturing imperfections of the target.

Fiducial Markers An artificial marker is a physical artifact consistent with a given model which can be used whenever a reliable pose estimation and identification is sought. This is the case, for instance, for many Computer Vision tasks, ranging from robot tracking [96] and pose recovery [102, 110] to intrinsic [77] and extrinsic camera calibration [211]. Within these scenarios, the adoption of an artificial marker is often preferred over less invasive alternatives, such as features that can be naturally found within the scene. In fact, while more convenient from a practical point of view, natural features are not guaranteed to be abundant in every scene or to exhibit an adequate level of detection and recognition reliability. Furthermore, since an artificial marker can be used to satisfy different needs, it is valuable to be able to create application-specific designs. For these reasons, fiducial tags are not only a widely used tool in practice, but they are also a lively research topic. Since a marker is usually created to be easily detected by a pinhole-modeled camera, most approaches are designed to exploit the projective invariance of basic geometrical entities. Specifically, most markers that can be found in literature are based on projective-invariant features that are both simple and easy to detect, such as points, lines, planes and conics. While it is difficult to track back to the earliest marker designs, it is sensible to believe that circular dots were among the first shapes used. In fact, circles appear as ellipses under projective transformations and the associated conic is invariant with respect to intrinsic or extrinsic parameters of the camera. This allows a

[1] Optical system that combines the use of lens and mirrors are usually referred as *catadioptric*. The introduction of mirrors in the optical system greatly complicates the distortion behavior allowing even to capture 360° images.

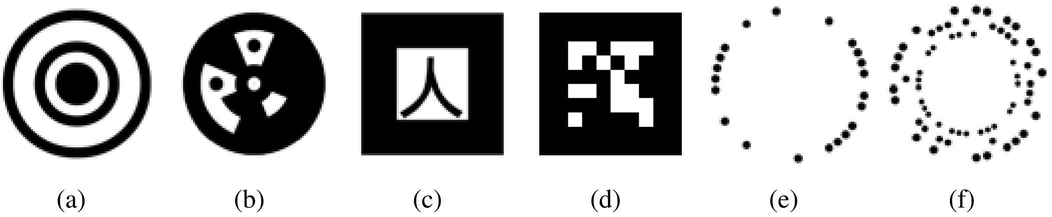
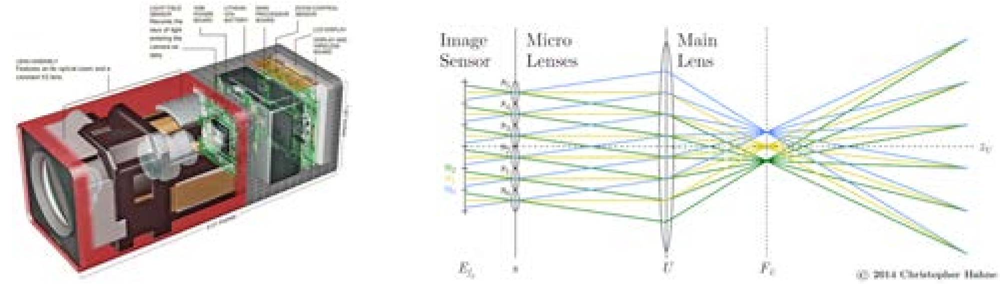


Figure 2.3: Some examples of fiducial markers that differ both for the detection technique and for the pattern used for recognition. In the first two, detection happens by finding ellipses and the coding is respectively held by the color of the rings in Concentric Circles (a) and by the appearance of the sectors in Intersense (b). While ARToolkit (c) uses image correlation to differentiate markers, ARTag (d) relies in error-correcting binary codes. Finally, in (e) and (f) we show two instances of RUNE-43 and RUNE-129 respectively.

fast and robust detection of the features directly on the image plane. Moreover, it is quite straightforward to find a proper homography which transforms back the found ellipses in circles, yielding an orthogonal view of the marker itself.

Such properties are exploited, for instance, in the earlier conception proposed by Gatrell [98], adopting a set of highly contrasted concentric circles which, after detection, are validated using the correspondences between the centroids of the ellipses found. In addition to the sheer feature localization, this design also allow to attach to each marker some additional information payload. This is obtained by alternating white and black circles according to some predefined pattern. This design has been slightly enhanced in [66] where the concentric circles are drawn using different colors and multiple scales, thus allowing to embed more information. Dedicated "data rings" are added to the fiducial design in [129] and [162]. A set of four circles located at the corner of a square is proposed in [69], where an additional pattern is placed between the four dots in order to distinguish between different targets. This ability to separate a set of different markers is crucial for complex scenes where more than a single fiducial is required to better handle occlusion or to track several objects at the same time. As an additional bonus, the availability of a coding scheme can be used to enable a validation step and to lower the number of false positives. For these reasons, a lot of effort has been dedicated to the design of effective coding schemas (see for instance [90, 157, 180]). A rather different but extensively used approach for marker recognition is to leverage on the geometrical properties of the cross ratio among detected feature points [142,215,219] or lines [224]. An interesting advantage of the cross ratio is that, being projective invariant, the recognition can be made without the need of any rectification of the image. Unfortunately, this comes at the price of a low overall number of distinctively recognizable patterns, and thus concurrently usable markers. In fact the cross ratio is a single scalar with a strongly non-uniform distribution [118] and this limits the number of well-spaced different values that can possibly be generated. Finally, also lines are a frequently used feature in the design of fiducial markers. Usually, they are exploited by detecting the border edges of a highly contrasted



(a) (b)

Figure 2.4: Left: A schematic rapresentation of the Lytrocamera. Right: The scene projection process on a plenoptic camera based on array of micro lenses. A standard camera lens focuses the scene onto the micro lenses array that act like multiple small pinhole cameras converging the captured rays to the device sensor. This way, the whole light field entering the main lens can be acquired

quadrilateral block. This is the case, for instance, for the ARToolkit [123] system which is often adopted as a reference baseline since it has a wide user base and its code is freely available in source form. Due to the easiness of detection and reasonable pose recovery accuracy that can be obtained with this kind of design [148], similar approaches are found in many recent proposal, such as ARTag [91] and ARToolkitPlus [227]. With these latter methods, the recognition technique of ARToolkit, which is based on image correlation between arbitrary images, is replaced by the reading of a binary coded pattern (see Fig. 2.3). The adoption of an error-correcting code makes both the marker detection and identification very robust, in fact we can deem these designs as the most successful from an applicative point of view. In section 3.1 we propose a new Fiducial Marker design , called *RUNE Tag*, based on concentric rings of dots and we explain the projective geometry principles that allows us to easily spot and identify the marker inside an image.

2.3 Unconstrained imaging model

The pinhole model augmented with a proper distortion correction is undoubtedly the most known and used method to approximate the optical acquisition process of most of the cameras. There exist, however, cameras which optical behavior can't be modeled by the Pinhole Camera Model.

Catadioptric cameras, for instance, albeit maintaining the central point assumption (all rays entering the camera cross the same point) are not suitable to be calibrated with generic pinhole plus distortion camera models. Many ad-hoc solutions have been proposed in literature to properly calibrate Catadioptric cameras [122, 191, 245].

Even more challenging are *Plenoptic* (light-filed) cameras (see figure 2.4). In Plenoptic cameras the single point of view requirement (and thus the central point assumption)

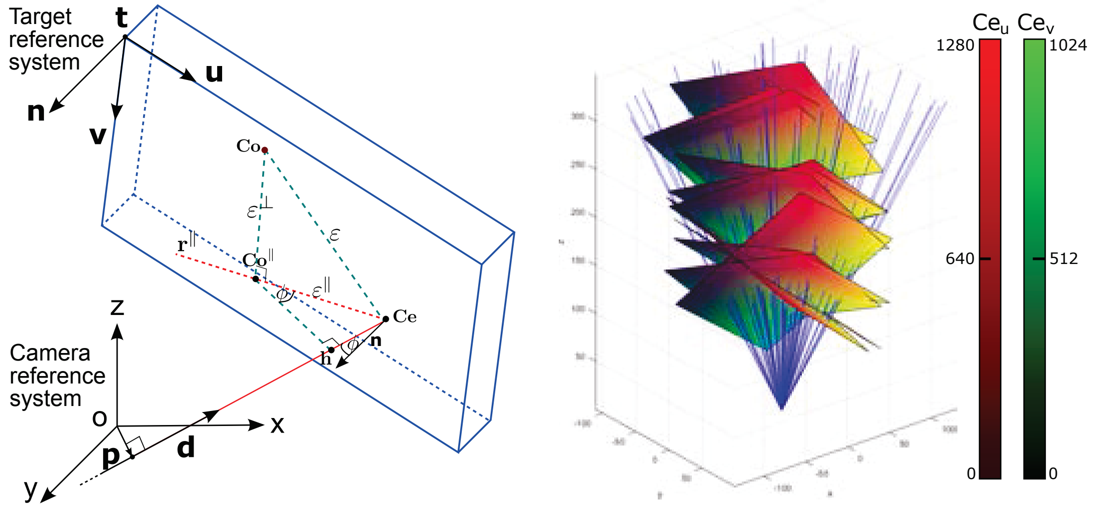


Figure 2.5: Left: Schema of the unconstrained camera model involving the optical center o, the ray direction, the observed and expected code and a target pose. Right: Rays and calibration target poses recovered by the optimization. Only a subset of the camera rays are plotted for visualization purposes.

is dropped.

In practice, most, if not all, the light-field devices ever built are made up of an array (explicit or implicit) of traditional cameras, each one contributing to capture a portion of the plenoptic function. The number, type and arrangement of such cameras, as well as their calibration, has been a very active topic in recent research. One of the main hurdles in plenoptic photography derives from the composite imaging formation process which limits the ability to exploit the well consolidated stack of calibration methods that are available for traditional cameras. While several efforts have been done to propose practical approaches [40, 65, 78, 120], most of them still rely on the quasi-pinhole behavior of the single microlens involved in the capturing process. This results in several drawbacks, ranging from the difficulties in feature detection, due to the reduced size of each microlens, to the need to adopt a model with a relatively small number of parameters.

In [33] Bergamsco et al. describe an approach to calibrate the camera which does not adopt any particular model. They, instead, calibrate independently each ray associated with a camera pixel. Although in the aforementioned paper this calibration approach is used to demonstrate its higher accuracy even in Pinhole-like cameras, it is in principle suitable for a wide variety of imaging models and even for imaging systems that can hardly be described by a parametric model.

In [33] each ray exiting from the camera and passing through the center of a CCD's pixel is modelled as a ray in the Euclidean space, totally independent from the behaviour of the other rays. The ray associated with the ith pixel is written as $r_i = (d_i, p_i)$ where $d_i, p_i \in \mathbb{R}^3$ are the ray direction and a point on the ray respectively (see Figure 2.5). In particular they are chosen so as $\|d_i\| = 1$ and $d_i^T p_i = 0$.

The resulting optimization problem has several millions of parameters (4 for each ray) and it is not tractable using the most commonly used calibration techniques described in the previous section. The authors adopts, instead, a dense fiducial marker exploiting the phase coding with the number-theoretical phase unwrapping approach presented by Lilienblum and Michaelis [143]. It is used to encode both the horizontal and vertical coordinate of each pixel of a LCD display in which a series of sinusoidal patterns are showed over the time. For each camera ray r_i and target pose $\Theta_s = (R_s, t_s)$ we have than an observed code Co_s^i and an expected code $C_e(r_i|\Theta_s)$ computed as the intersection between the ray r_i and the target plane Θ_i.
The optimization problem is than resolved for both the camera rays r and the target poses Θ casting it as generalized least squares problem minimizing the distances between the observed and expected codes:

$$(\hat{r}, \hat{\Theta}) = \arg\min_{r,\Theta} \sum_{i,j,s} (\epsilon_i^s)^T (\Sigma_i^s)^{-1} \epsilon_i^s \tag{2.4}$$

where $\epsilon_i^s = Co_i^s - Ce(r_i|\Theta_s)$ is the difference between the observed and expected code and Σ_i^s is the (conditional) error covariance matrix accounting for the heteroscedasticity of the error under the given pixel-pose combination.

In sections 4.2 we try to apply the unconstrained model to calibrate a Lytro light-field camera [165] and use the calibration to retrieve the 3D structure of the scene.
In section 4.1 we use the same calibration technique to online calibrate the projector in a stereo structured light scanner, increasing both the accuracy and coverage of acquired reconstruction.

The drawback of the described approach is that epipolar geometry can't be exploited even for simply distorted pinhole cameras. In the work described in section [33] we try to overcome this problem by constraining all the rays to pass through the same optical centers, this allows us to bring us back to the Pinhole Camera model, but still maintain a free distortion calibration.

3

Camera calibration exploiting image features

The estimation of camera intrinsic parameters plays a crucial role in all computer vision tasks for which the underlying model that drives the image formation process has to be known. As a consequence, a deluge set of different approaches has been proposed in literature over the last decades. Most of them optimize the model parameters minimizing some re-projection error measure of the inferred three-dimensional scene in various shots taken from different points of view .

A first distinction of the calibration techniques can be be done based on the kind of scene that is considered for calibration. Some approaches rely only on the identification of some distinctive (in the single image) and repeatable (among multiple images) features present in natural scenes. Under controlled conditions, when high accuracy is needed, it is generally a good idea to introduce into the scene artificial features. This allows to exploit previous knowledge of the scene to better localize the features in the images and take advantage of their spatial relation to filter outliers. Most of those lean on the observation of a known object (i.e. a calibration target) from different point of views, providing the necessary data to estimate the model through different optimization approaches.

In section 3.1 we will introduce a new Fiducial Markers designed as a set of planar circles placed as to be easily recognized. If we aim to obtain the higher possible calibration accuracy, it is very important indeed to localize with high precision the position of the corresponding features in both the image plane and 3D space. In the last section of this chapter a method for the refinement of 3D ellipses is proposed. The method consists in the optimization of the parameters of a 3D ellipse so that its re-projection onto the image planes is coherent with the images acquired by a camera network. Although the application context described in 3.2 could be misleading, it should be very straightforward to integrate the optimization problem in a calibration pipeline (at least for the calibration of camera extrinsic parameters) and we are working in this direction as a follow-up of this work.

3.1 An Accurate and Robust Artificial Marker based on Cyclic Codes

In this section we introduce a novel fiducial marker that combines several strengths of different approaches, resulting in an all-rounder that can be directly applied in many scenarios. The key idea underlying our design is to entrust the robustness of the detection process to a well-grounded and occlusion-resilient cyclic code rather than to the geometrical features themselves. The marker we are introducing is arranged to facilitate its localization thanks to a reading frame that can be fully constrained using only two marker dots. Furthermore, its design is flexible enough to allow to use different amounts of dots, granting a higher robustness or a greater working distance, depending on the specific scenario. Moreover, the large number of dots provided by our marker, beside boosting the pose estimation accuracy, can be exploited to enable applications that are usually considered beyond the domain of fiducial markers, such as camera calibration.

This section is organized as follows: in 3.1.1 we describe the general design and we introduce two localization methods, to be used respectively with calibrated and uncalibrated cameras. Afterwards, we introduce the adopted coding strategy and we suggest a technique to perform instant decoding, including proper recovery from errors due to occlusion and misdetection of the marker dots. Finally, in 3.1.2 we test the accuracy achieved when dealing with different real-world problems and we compare the obtained performance with some widely used fiducial markers.

3.1.1 Rings of UNconnected Ellipses

We design our tags as a set of circular high-contrast features (*dots*) spatially arranged into concentric *layers* (See Fig. 3.1). The tag internal area, delimited by the outer layer, is divided into several evenly distributed circular *sectors*. Each layer and sector intersection defines a *slot* that may or may not contain a dot.

In a tag composed by m layers and n sectors, we can encode a sequence of n symbols taken from an alphabet of 2^m elements. Each element of the alphabet is simply defined as the binary number encoded by the presence or absence of a dot. For example, if the 14^{th} sector of a $3-$layer tag contains a dot in the first and the last layer, we encode the 14^{th} symbol with the number $5_{10} = 101_2$. In this section we propose two instances of such design, namely *RUNE-43* and *RUNE-129*. The first is composed by a single layer divided into 43 sectors. Since the alphabet contains only 2 elements (1 bit given by the presence or absence of a dot), each *RUNE-43* encodes a sequence of 43 binary symbols. Conversely, the latter is composed by 3 layers divided into 43 sectors. 3 slots for each sector allow to encode a sequence of 43 symbols from an alphabet of $2^3 = 8$ elements. Not surprisingly, not all the possible codes can be used as valid codes for the tag. For instance, the tag composed by only empty slots does not make any sense. Therefore, we require the coding strategy to respect some properties to uniquely identify each dot regardless the projective transformation involved. We discuss this topic in detail in section 3.1.1.

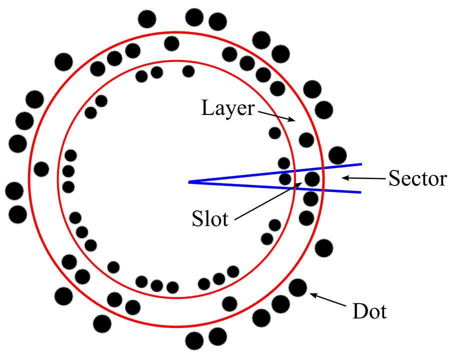


Figure 3.1: Our proposed design divided into its functional parts. An instance of a 3-layers RUNE-129 is displayed.

Finally, we set the dot radius equals to κ-times to the radius of the layer at which the dot is placed. We can take advantage of this property to dramatically speed up the detection as explained in section 3.1.1.

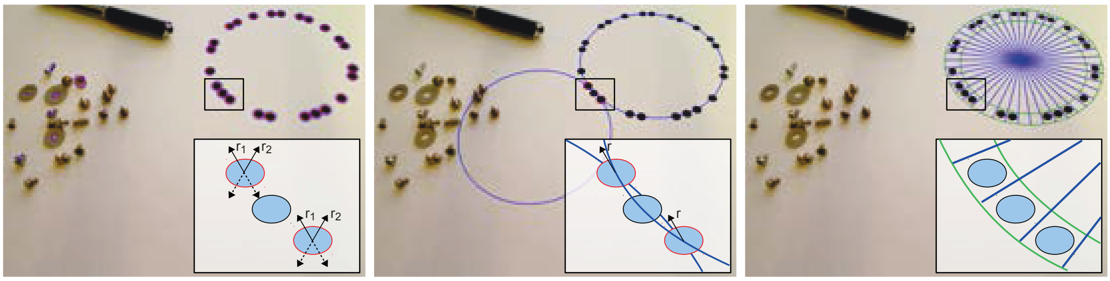
Candidate selection with a calibrated camera

One of the core features of every fiducial marker system is its ease of detection. Even if one of our principles is to promote the accuracy over the speed, we still need to setup an efficient way to identify each circular feature among the tags. Given an image, we start by extracting a set of candidate dots. To do this, we use a combination of image thresholding, contour extraction and ellipse fitting provided by the OpenCV library. Additionally, a subsequent naive filtering step based on dot eccentricity and area keeps only whose features respect a reasonable prior. Finally, each extracted ellipse can be further refined by using common sub-pixel fitting techniques such the one proposed in [170]. We give no additional details on the specific procedure we follow since is not important for all the subsequent operations. Any suitable technique to extract a set of circular dots from a generic scene would be fine.

At this point, we need a method to cluster all the candidate dots into different possible tags and discard all the erroneous ones that were originated by noise or clutter in the scene.

Total ellipses	10	50	100	500
Full (RANSAC)	252	2118760	75287520	$> 10^{10}$
Proposed method	45	1225	4950	124750

Table 3.1: Number of maximum possible RANSAC combinations required for ellipse testing.



(a) Feasible plane orientations estimation

(b) Candidate ring estimation

(c) Dot vote counting

Figure 3.2: Steps of the ring detection: in (a) the feasible view directions are evaluated for each ellipse (with complexity $O(n)$), in (b) for each compatible pair of ellipses the feasible rings are estimated (with complexity $O(n^2)$), in (c) the dot votes are counted, the code is recovered and the best candidate ring is accepted (figure best viewed in color).

Since we know that the dots are arranged in circular rings, we expect that dots belonging to the same layer would appear disposed around an elliptical shape once observed through a central projection. Therefore, dots in the same layer can be identified by fitting an ellipse through their 2D image coordinates and verifying the distance assuming this model.

A common approach would consist in the use of a RANSAC scheme that uses a set of 5 candidate dots to estimate the model (i.e. the ellipse) against which quantify the consensus of all the others. Unfortunately, since 5 points are needed to characterize an ellipse into the image plane, the use of RANSAC in a scenario dominated by false positives (even without clutter we expect the majority of dots to belong to different tag or even layer) would quickly lead to an intractable problem (See Table 3.1). A possible alternative could be the use of a specialized Hough Transform [247], but also this solution would not be effective since hindered by the relative low number of samples and the high dimensionality of the parameter space.

What makes possible the detection of our tags in reasonable time is the observation that there exist a relation between the shape of a dot and the shape of the ring in which is contained. Specifically, they both appear as two ellipses (since they originate from a projection of two circles) and the parameters of both curves depend on the relative angle between the camera and the plane in which they lie. Even if from a single conic is not possible to recover the full camera pose, there is still enough information to recover (up to a finite set of different choices) a rotation that transform that conic into a circle. This, combined with a known relation between the relative size of the dots and the rings, can give clues of the geometry of a layer and so ease the clustering process.

In this section, we give a detailed description on how the recovering of such rotation is done assuming a known camera matrix. In many situations, the requirement of a calibrated camera is not particularly limiting. For instance, if our tags would be used as a coarse registration method for a structured-light scanner solution (we give examples of this in 3.1.2), the camera would certainly be calibrated as implied by the reconstruction process. However, for the high accuracy exhibited in points localization, it would be interesting to use our tags as a calibration target instead of a classical chessboard. To deal with this situations, we propose a way to still use our tags in an uncalibrated scenario in section 3.1.1.

Given the set of initial dot candidates, we start by recovering the parameters describing their elliptical shape. Specifically, we translate the image reference frame so that the principal point coincides with the origin, and parametrize each conic as the locus of point such that:

$$\mathbf{x}^T\mathbf{Q}\mathbf{x} = \begin{pmatrix} u & v & 1 \end{pmatrix} \begin{pmatrix} A & B & -\frac{D}{f} \\ B & C & -\frac{E}{f} \\ -\frac{D}{f} & -\frac{E}{f} & -\frac{F}{f^2} \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = 0 \tag{3.1}$$

Where f is the camera focal length and u, v are pixel coordinates with respect to the optical center.

We follow [58] to estimate a rotation around the camera center that transforms the ellipse described by $\mathbf{Q}$ into a circle. Specifically we decompose $\mathbf{Q}$ via SVD

$$\mathbf{Q} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \text{ with } \mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$$

and compute the required rotation as:

$$\mathbf{R}_{\mathbf{Q}} = \mathbf{V} \begin{pmatrix} g\ \cos\alpha & s_1\ g\ \sin\alpha & s_2 h \\ \sin\alpha & -s_1 \cos\alpha & 0 \\ s_1 s_2 h \cos\alpha & s_2 h \sin\alpha & -s_1 g \end{pmatrix} \tag{3.2}$$

$$g = \sqrt{\frac{\lambda_2 - \lambda_3}{\lambda_1 - \lambda_3}}, \quad h = \sqrt{\frac{\lambda_1 - \lambda_2}{\lambda_1 - \lambda_3}}$$

where s_1 and s_2 are two free signs, leaving 4 possible matrices and α is any arbitrary rotation aroud the normal of the plane which remains constrained while observing just a single ellipse. At this point, if we fix $\alpha = 0$, each detected ellipse $\mathbf{Q}$ may spawn four different rotation matrices $\mathbf{R}^i_{\mathbf{Q}}, i = 1 \dots 4$ that transforms the conic into a circle (Fig. 3.3).

Since two of this four candidates imply a camera observing the back-side of the marker, we can safely discard all the $\mathbf{R}^i_{\mathbf{Q}}$ for which the plane normal $N^i_{\mathbf{Q}} = \mathbf{R}^i_{\mathbf{Q}} \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T$ is facing away from the camera (i.e. the last component is positive).

At this point, we search for whole markers by simultaneously observing the rotation matrices of a couple of ellipses. Specifically, for each pair $\mathbf{Q}_k$ and $\mathbf{Q}_w$, we produce the set of the four possible rotation pairs $\Re = \{(\mathbf{R}^i_{\mathbf{Q}_k}, \mathbf{R}^j_{\mathbf{Q}_w}); i, j = 1 \dots 2\}$. From this set, we remove the pairs for which the inner product of the relative plane normals is below a fixed threshold and average the remaining rotation pairs by means of quaternion mean. Finally, we keep the best rotation average by choosing the one that minimize the difference between the radii of $\mathbf{Q}_k$ and $\mathbf{Q}_w$ after being transformed by such rotation. The rationale is to avoid to choose ellipses with discordant orientations (as the marker is planar) and to use a compatibility score that takes advantage of the fact that ellipses on the same ring should be exactly the same size on the rectified plane.

Whenever a pair of dots $\mathbf{Q}_k$ and $\mathbf{Q}_w$ generate a good average rotation $\mathbf{R}_{(\mathbf{Q}_k,\mathbf{Q}_w)}$, two hypothesis on the ring geometry can be made (Fig. 3.2.b). Indeed, we expect the ring shape being such that the following two properties holds. First, it should pass trough the centers of $\mathbf{Q}_k$ and $\mathbf{Q}_w$. Second, the ratio between the ring radius and the radii of $\mathbf{Q}_k$ and $\mathbf{Q}_w$, after being transformed trough $\mathbf{R}_{(\mathbf{Q}_k,\mathbf{Q}_w)}$, should be exactly κ. Operatively, we first fit the two circles $\mathbf{C}_1$, $\mathbf{C}_2$ passing trough the centers of $\mathbf{R}^T_{(\mathbf{Q}_k,\mathbf{Q}_w)}\mathbf{Q}_w\mathbf{R}_{(\mathbf{Q}_k,\mathbf{Q}_w)}$ and $\mathbf{R}^T_{(\mathbf{Q}_k,\mathbf{Q}_w)}\mathbf{Q}_k\mathbf{R}_{(\mathbf{Q}_k,\mathbf{Q}_w)}$ and having radius $\kappa\ \hat{r}$ where $\hat{r}$ is the average radius of the two transformed dots. Then, we transform $\mathbf{C}_1$ and $\mathbf{C}_2$ back through the inverse of $\mathbf{R}_{(\mathbf{Q}_k,\mathbf{Q}_w)}$.

As soon as candidate rings are extracted, a circular grid made by sector and layers can be generated directly on the image (Fig. 3.2.c). Of course, if the tag is composed by more than one layer, we need to generate additional rings bot inward and outward. Then, for each slot the presence or absence of a dot can be observed to produce a binary sequence that will be analyzed in the decoding step to identify or discard the candidate marker.

To summarize, the detection step goal is to identify possible markers candidates by searching groups of dots belonging to the same ring, expecting them arranged in an elliptical shape. To do so, we avoid the direct estimation of ellipses in the image since it would require an unfeasible effort. Diversely, we take advantage of the geometrical properties of the dots and the known ratio κ to obtain two possible ring candidate for each pair of ellipses. As result, only $O(n^2)$ operations are required.

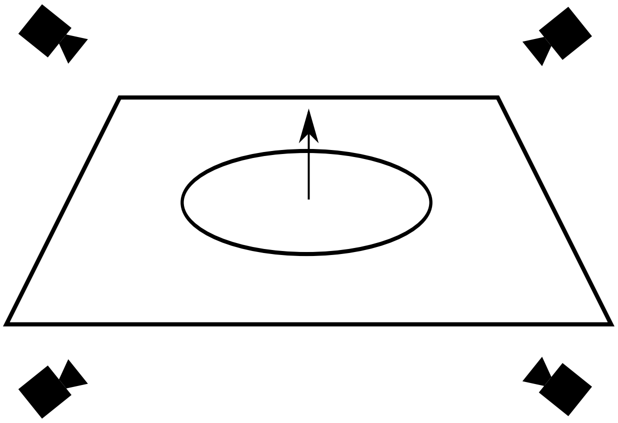


Figure 3.3: The four possible camera orientations that transform an observed ellipse into a circle

Dealing with the uncalibrated case

The approach described so far assumed a calibrated camera setup. Indeed, all the rotation matrices $\mathbf{R_Q}$ were designed to transform conics lying on the normalized image plane (hence requiring the focal length) around the camera optical center. It has to be noted, however, that camera intrinsics are not an implied requirement of the tag itself but just a powerful tool to dramatically speed-up the detection. As a consequence, we would be satisfied to just guess a raw estimation of the focal length and principal point good enough to still produce fair rotation matrices and sustain the detection procedure.

We decided to use the image center as our guess of the principal point. Even if it appears a bold assumption, we observed that this holds for most cameras. Diversely, the focal length is difficult to guess as it depends on the lens mounted. However, also in this case we can take advantage on the geometric properties involved when observing a set of coplanar circles.

In Section 3.1.1 we discussed how two feasible plane normals can be estimated from each conic. It's crucial to observe that, if we apply the same projective transformation to two circles lying on the same plane, only one plane normal estimated from the first circle will be parallel to a normal extracted from a second, whereas the other two will diverge [35]. Furthermore, this property holds only for the correct focal length and optical center and can be naturally expanded to multiple coplanar conics.

To better explain this behaviour, we extracted all orientations from a set of 3 coplanar

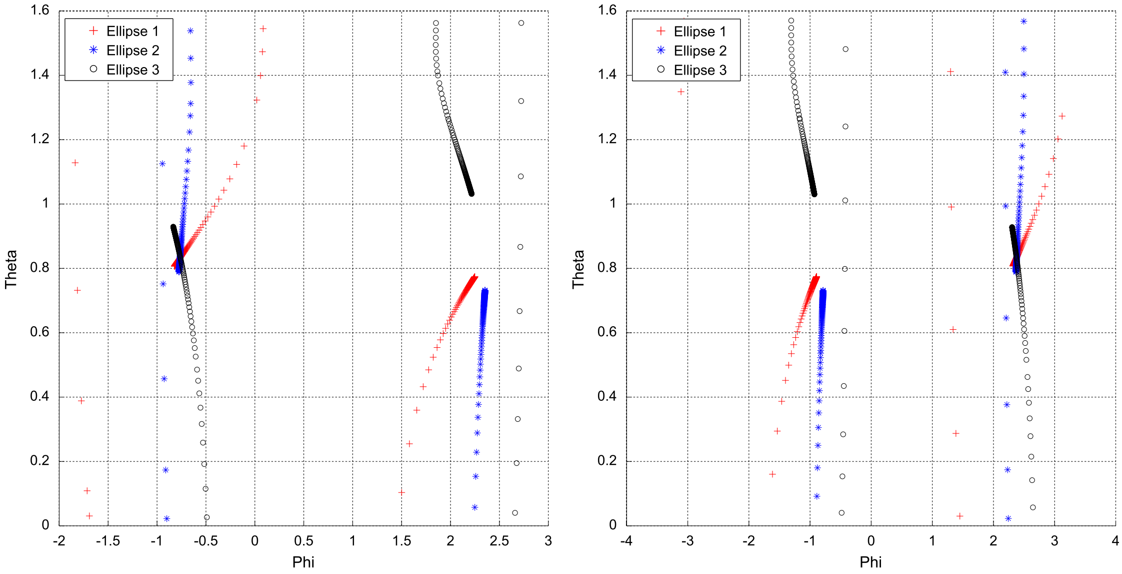


Figure 3.4: Estimated normals orientation in spherical coordinates of three coplanar ellipses spanning positive (Left) and negative (Right) focal length values. Note how one of the two possible orientations converge to a common direction while the other does the opposite.

circles assuming to know the optical center and varying the focal length. In fig. 3.4 (Left) we plotted the values of such orientations in spherical coordinates spanning positive values of f from almost zero to 5 times the known correct value. For the right plot we did the same procedure but with negative values. In general, each ellipse produces two different traces in (ϕ, θ)-plane as a function of the focal length. Since all correct orientations have to be parallel to each other when the correct focal length is used, traces that are relative to the correct orientation will converge to a same point as f get closer to the expected value. On the other hand, all other traces will follow different paths and will diverge to different directions. It's clear form the plot that for positive values of the focal length there is only one intersection point (in this example $\phi \simeq -0.76, \theta \simeq 0.83$). Also, since the other possible intersection only happens when f becomes negative, the wrong orientation will never be present in the set of feasible solutions.

This means that we can both estimate the correct focal length and extract sets of coplanar circles by solving a clustering problem among all the generated plane normals. However, there is no simple closed form solution to reverse the process and obtain the best possible focal length that would have produced a given normal. Therefore, we restrict our estimation to a discrete set of n_f possible focal length values $f_i, i = 1 \ldots n_f$ equally spaced inside the range $f_{\min} \ldots f_{\max}$. At this point, for each dot $\mathbf{Q}$ detected in a scene and for each f_i, exactly two feasible plane normals $N^1_{\mathbf{Q}_{\mathbf{f_i}}}, N^2_{\mathbf{Q}_{\mathbf{f_i}}}$ can be computed as described in section 3.1.1. All such normals will exhibit two degrees of freedom and hence can be easily parametrized in spherical coordinates with azimuth ϕ and elevation θ as vectors in $\mathbb{R}^2$. Then, all these vectors are collected in a 2D accumulator whose bins are equally divided into equal angular ranges.

Once the accumulator is completely filled with values extracted from all the dots, local maxima with respect of the cardinality of the bins will represent clusters of normals oriented almost in the same direction[1]. Finally, once a maxima is selected, we take the median focal length of all the candidates contained in a bin as our sought focal length estimate. Moreover, the candidates contained in a winning bin are all coplanar and thus the dots search phase can be restricted on such set.

An example of the proposed focal length estimation method is given in Fig. 3.5. We started by rendering a synthetic image of a RUNE-149 tag trough a virtual camera of known focal length $f_{vc} = 1000\ px$ and with principal point being exactly the center of the image (First row of Fig. 3.5). In the middle of the figure, we plotted the accumulator values projected on the front and back side of a hemisphere. As expected, a clear accumulation of votes can be observed in the bin containing the combination of ϕ, θ corresponding to the normal of the plane on which the tag lie. On the right, we plotted the distribution of the focal length candidates of the winning bin, highlighting a clear maximum around the correct value of f_{vc}. Conversely, we repeated the same test with two tags on the same image lying into two different planes (Second row of Fig. 3.5). This time, the accumulator shows two clear maxima corresponding to the plane normals of the two planes. Again, on the right side of the figure we plotted the distribution of the focal length

[1] more precisely, the variability inside the bin depends on the bin size itself

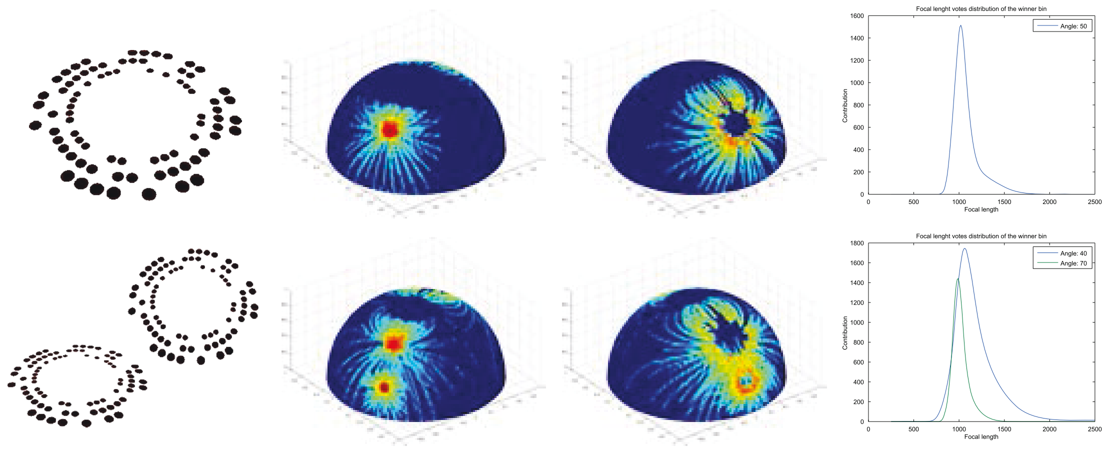


Figure 3.5: A synthetic representation of the marker dots normal voting scheme used to guess an initial value of the camera focal length. Left: RUNE-129 markers rendered by a virtual camera with known focal length and principal point. Center: the normal accumulator in spherical coordinates. Right: Focal length distribution of the bins. See the text for a complete discussion on the voting procedure.

candidates for the two winning bins. Two important observations can be made. First, both the two distributions show two clear maxima around f_{vc}, demonstrating that a focal length guess is the same regardless of the tag orientation. Second, the more a tag is angled the more the guess is near the expected value. This can be explained by noting that a tag perfectly parallel to the imaging plane has all the dots exactly appearing as circles and so no focal length can be recovered. Therefore, the correct focal length is better constrained when the eccentricity of the dots is low. In fact, from the accumulator can be noted that the maximum corresponding to the angled tag is far more sharp than the other.

Even if the focal length guess is somehow biased by the angle of the observed tag, we feel that this won't be a show-stopper as we can still obtain a focal length guess good enough to let the detection procedure work properly. To convince the reader furthermore, we recall that the focal length is used to obtain a good rotation matrix to transform all the dots into circles. The more the angle is low, the more the focal length become irrelevant to recover that rotation. In the extreme case, to detect a perfectly parallel tag the focal length is not necessary at all since all the dots (and so the whole tag) already appear as circles.

To conclude, in the uncalibrated case we require an initial camera intrinsic parameters guessing step able to produce values good enough to perform a subsequent tag detection. To do so, we guess the principal point as the image center and the focal length with a voting procedure among a discretized set of plausible focal length values.

Marker Recognition and Coding Strategies

Once a candidate marker has been detected, dots distribution among the slots produces a sequence of symbols that can be subsequently used to identify each tag. However, two coupled problems raise. First, we don't have a starting position of the symbols sequence since the detection step can only identify each candidate up to a rotation around the normal of the plane[2]. Consequently, any cyclic shift of the sequence is equally possible and must be recovered. Second, some dots may be missing or assigned to wrong slots thus requiring the identification procedure being somehow robust to this situations.

We decided to cast the problem into the solid mathematical framework of coding theory. Specifically, dot patterns of the tags corresponds to codes generated with specific properties and error-correcting capabilities. In section 3.1.1 we briefly discuss the mathematical theory involved in the generation of the codes while in section 3.1.1 we give a closed form solution to decode each code block in case of erasures and errors. We refer the reader to [147] for a in-depth investigation of the field.

Code generation

We start by defining a *block code* of length n over a set of symbols S as the set $C \subset S^n$. The elements of C are called *codewords*.

Let $q = p^k \in \mathbb{N}$ be a power of a prime number p and an integer $k > 1$. We denote with $\mathbb{F}_q$ the finite field with q elements. A *linear code* C is a $k-$dimensional vector sub-space of $(\mathbb{F}_q)^n$ where the symbols are taken over the field $\mathbb{F}_q$. A linear code is called *cyclic* if any cyclic shift of a codeword is still a codeword, i.e.

$$(c_0, \dots, c_{n-1}) \in C \Rightarrow (c_{n-1}, c_0, \dots, c_{n-2}) \in C$$

If we consider the field $\mathbb{F}_q[x]/(x^n - 1)$ obtained by the polynomial ring $\mathbb{F}_q[x]$ modulo division by $x^n - 1$, there exists a bijection to the vectors in $(\mathbb{F}_q)^n$:

$$(v_0, \dots, v_{n-1}) \Leftrightarrow v_0 + v_1 x + \dots + v_{n-1} x^{n-1}$$

Furthermore, C is a cyclic code if and only if C is an ideal of the quotient group of $\mathbb{F}_q[x]/(x^n - 1)$. This means that all cyclic codes in polynomial form are multiples of a monic *generator polynomial* $g(x)$ of degree $m < n$ which divides $x^n - 1$ in $\mathbb{F}_q[x]$. Since multiplying a polynomial form of a code by x modulo $x^n - 1$ corresponds to a cyclic shift

$$\begin{gathered} x(v_0 + v_1 x + \dots + v_{n-1} x^{n-1}) \bmod (x^n - 1) = \\ v_{n-1} + v_0 x + \dots + v_{n-2} x^{n-2} \end{gathered}$$

all codewords can be obtained by mapping any polynomial $p(x) \in \mathbb{F}_q[x]$ of degree almost $n - m - 1$ into $p(x)g(x) \bmod (x^n - 1)$.

[2] Note that, conversely, the verse of the sequence is induced by the counter-clockwise ordering of the sectors that is preserved since we always observe the frontal face of the marker plane.

Since all the cyclic shift are codes, we can group the codewords into *cyclic equivalence classes* such that two codewords are in the same class if and only if one can be obtained as a cyclic shift of the other. Since the number of elements in a cyclic equivalence class divides n, by choosing an n prime we only have classes either composed by a single element (constant codewords with n repetitions of the same symbol) or where all codewords are distinct. The first can be easily eliminated since it involves in at most q codewords.

In our marker setting, the identity of the marker is encoded by the cyclic equivalence class while the actual alignment of the circles (i.e. its rotation around the plane normal) can be obtained from the detected element within the class. Using coding theory enables us to balance the trade-off between the number of errors that can be handled with respect to the number of possible valid tags (i.e. the number of equivalence classes) granted. To our knowledge, is the first fiducial marker system that provides such feature at a geometrical level, modifying its shape to accommodate different requirements.

The *Hamming distance* $dH : S \times S \rightarrow \mathbb{N}$ is the number of symbols that differ between two codewords. Similarly, the Hamming distance of a code C is the minimum distance between all the codewords: $dH(C) = \min_{u,v \in C} dH(u, v)$. The Hamming distance plays a crucial role on the number of errors that can be detected and corrected. Indeed, a code with a Hamming distance d can detect $d-1$ errors and correct $\lfloor (d-1)/2 \rfloor$ erasures. When we consider a linear code of length n and dimension k, the singleton bound $d \leq n - k - 1$ holds. Thus, with a fixed code length n the error correcting codes capabilities are traded with a smallest number of available codewords. In our setting we restrict our analysis to the correction of random errors or erasures but the same mathematical framework can be used to improve the detection resilience while correcting burst errors (i.e. errors that are spatially coherent, like we have in case of occlusions).

For the proposed RUNE-Tags, we experiment on two specific codes instances. In the first one (RUNE-43) we encode the single-layer circular pattern as a vector in $(\mathbb{Z}_2)^{43}$, where $\mathbb{Z}_2$ is the remainder class modulo 2. The number 43 for the radial elements was chosen because it is a prime that leads to radial sectors of a reasonable size (slightly less than 10^o), but any cyclic code of prime length would work. The polynomial $x^{43} - 1$ factors into 4 prime polynomial in $\mathbb{Z}_2$, namely $x - 1$ and three polynomials of degree 14. By excluding $x - 1$ which generates only constant codes, and one of the prime polynomial of degree 14, we obtain a generator polynomial resulting in a cyclic code of dimension 15, with 762 different markers (equivalence classes) and a minimum Hamming distance of 13, allowing us to correct up to 6 errors. In particular, we used the polynomial (3.3) where the terms in brackets are two of the three degree 14 prime polynomials dividing $x^{43} - 1$.

$$\begin{aligned} g(x) = {} & (1 + x^2 + x^4 + x^7 + x^{10} + x^{12} + x^{14}) \\ & \qquad (1 + x + x^3 + x^7 + x^{11} + x^{13} + x^{14}) \end{aligned} \tag{3.3}$$

In the second (RUNE-129) we have 8 different patterns (since it is a 3-layer tag) in a sequence of 43 sectors. We hold out the pattern with no dots to detect erasures due to occlusions and we encode the remaining 7 patterns as vectors in $\mathbb{Z}_7$. For the whole target, the code is represented as vectors in $(\mathbb{Z}_7)^{43}$ using the generator polynomial (3.4).

$$\begin{aligned} g(x) = {} & (1+4x+x^2+6x^3+x^4+4x^5+x^6) \\ & (1+4x^2+6x^3+4x^4+x^6)(1+x+3x^2+5x^3+3x^4+x^5+x^6) \\ & (1+5x+5x^2+5x^4+5x^5+x^6)(1+6x+2x^3+6x^5+x^6) \\ & (1+6x+4x^2+3x^3+4x^4+6x^5+x^6) \end{aligned} \tag{3.4}$$

Again, this polynomial is produced excluding one prime factor of $x^{43}-1$. In particular, in $\mathbb{Z}_7$, $x^{43}-1$ factors into the usual $x-1$ and 7 prime polynomials of degree 6. By excluding $x-1$ and one of the degree 6 factors we obtain a cyclic code of dimension 7, giving 19152 different markers with a minimum Hamming distance of 30, and allowing us to correct up to 14 errors, or 29 erasure, or any combination of t errors and e erasures such that $2t+e \leq 29$. Unlike the case for (RUNE-43) where any choice of the prime factor to exclude leads to equivalent codes, here the choice of the prime factor to exclude was dictated by the need to have a generalized BCH code for fast decoding, as it will be explained in the next Section. Of the 6 prime polynomials, only 2 produced a BCH code correcting 14 errors, while the others had smaller correction capabilities through BCH decoding.

Decoding

The recognition of a tag is divided into two main stages: First the observed code sequence is decoded, i.e., we find the valid codewords that is closest to the observed sequence. Second, we align the codeword, extracting a unique representative of the cyclic class and the relative cyclic shift of the decoded codeword.

Given the relative high correction capabilities of the Codes, for the first stage we opted for an algebraic syndrome-based decoding.

Let $g(x)$ be the generator polynomial, and $w(x) = a(x)g(x)$ a codeword. Given an observed sequence $v(x) = w(x) + e(x)$ where $e(x)$ is the error, the goal of the decoding process is to recover the error $e(x)$ and consequently, the codeword $w(x)$ and the code $a(x)$.

Let $\mathbb{F}_{q^m}$ be an extension of $\mathbb{F}_q$ that splits x^n-1 into n linear terms $x^n-1 = \prod_{i=1}^{n}(x-a_i)$ where $a_i \in \mathbb{F}_{q^m}$ are n-th roots of unity. Further, let $\alpha \in \mathbb{F}_{q^m}$ be a primitive n-th root of unity, i.e., $\alpha^n = 1$ and $\alpha^k \neq 1$ for all $k < n$, then all the roots a_i of x^n-1 are of the form α^j with $j \in \{0,\dots,n-1\}$. Since the generator polynomial $g(x)$ divides x^n-1, some of these divide $g(x)$. Let

$$D = \{i \in 0,\dots,(n-1) \mid g(\alpha^i) = 0\} \tag{3.5}$$

$$N = \{i \in 0,\dots,(n-1) \mid g(\alpha^i) \neq 0\} \tag{3.6}$$

be the set of powers i for which α^i is and is not a root of $g(x)$, clearly, given correct codeword $w(x)$ we have

$$\forall i \in D,\; w(\alpha^i) = a(\alpha^i)g(\alpha^i) = 0\,. \tag{3.7}$$

We define the syndrome $(S_1, \ldots, S_n)$ of an observed sequence $v(x)$ as

$$S_i = v(\alpha^i) \tag{3.8}$$

we want to connect the values of the syndromes S_i, $i \in D$ with the (correctable) error $e(x)$.

Let $i_1, \ldots, i_w$ be the indices of the non-zero digits in the error sequence $e(x)$, where $w \leq t$ is the number of errors, t being the maximum number of errors correctable by the code. We define the *error locator polynomial* as

$$\Lambda(x) = \prod_{j=1}^{w} \left(1 - \alpha^{i_j} x\right) = 1 + \sum_{j=1}^{w} \lambda_j x^j \tag{3.9}$$

with $\lambda_j \in \mathbb{F}_{q^m}$. With the error locator at hand, we can locate the error digits by finding the powers of α that are roots of $\lambda(x)$, in fact $\lambda(\alpha^{-i}) = 0$ if and only if the i-th digit of v is wrong.

The Newton equations link the coefficients λ_i, $i = 0, \ldots, (n-1)$ of the locator polynomial with the syndromes S_j, $j = 0, \ldots, (n-1)$ of the error $e(x)$:

$$\begin{cases} S_i + \sum_{j=1}^{i-1} \lambda_j S_{i-j} + i\lambda_i = 0, & i \leq t \\ S_i + \sum_{j=1}^{w} \lambda_w S_{[i-j]_n} = 0, & t < i \leq n+t\,, \end{cases} \tag{3.10}$$

where $[x]_n$ is the remainder of the division of x by n.

note that of every index $i \in D$ the syndrome S_i of $e(x)$ is equal to the same-index syndrome of the observed sequance $v(x)$, in fact:

$$\forall i \in D,\; v(\alpha^i) = a(\alpha^i)g(\alpha^i) + e(\alpha^i) = e(\alpha^i)\,, \tag{3.11}$$

while for the indices in N the syndromes of $e(x)$ are unknown. Although there are approaches to solve the Newton equation in the general case [168], the most efficient algorithms are for special codes where there are sufficient equations in (3.10) that only use known syndromes to solve the system of linear equations in $\mathbb{F}_{q^m}$.

A cyclic code is *generalized BCH* correcting t errors if there are $2t$ consecutive powers of α $(\alpha^c, \ldots, \alpha^{c+2t-1})$ that are roots of $g(x)$. In this case we have exactly t of the Newton equations making use only of the known syndromes, i.e., the syndromes computed on those powers of α, thus we can solve (3.10) using those equations. In particular, the error locator as well as the actual error can be efficiently computed using Forney's algorithm [94]. Let $S(x) = S_c + S_{c+1}x + \cdots + S_{c+2t-1}x2t - 1$ be the *reduced syndrome* polynomial, there are two unique polynomials $\Lambda(x)$ and $\Omega(x)$ in $\mathbb{F}_{q^m}[x]$ with degree less than or equal to t for which

$$\Omega(x) = S(x)\Lambda(x) \mod x^{2t} . \tag{3.12}$$

We call $\Omega(x)$ the *error evaluator* polynomial, while $\Lambda(x)$ is the error locator polynomial (3.9). These polynomials can be computed efficiently noting that (3.12) can be re-written as

$$\Lambda(x)S(x) + f(x)x^{2t} = \Omega(x) . \tag{3.13}$$

Hence, $\Omega(x)$ is the gcd between $S(x)$ and x^t, and both $\Lambda(x)$ and $\Omega(x)$ can be computed using the (generalized) Euclidean algorithm.

Once the locations of the errors are computed finding the roots of $\Lambda(x)$ among the powers of α, the error values are computed as follows:

$$e_j = -\frac{\alpha^{-(c-1)j}\Omega(\alpha^{-j})}{\Lambda'(\alpha^{-j})} , \tag{3.14}$$

where $\Lambda'(x) = \sum_{i=1}^{2t-1} i\lambda_i x^{i-1}$ is the formal derivative of $\Lambda(x)$.

In the case of the presence of e erasures, a BCH code can correct up to $t' = t - \lceil e/2 \rceil$ errors. First we set the erased digits to 0, and then we consider the *error/erasure locator polynomial* $\Gamma(x) = \Lambda(x)E(x)$ where $E(x)$ is the *erasure locator polynomial*

$$E(x) = \prod_{j=1}^{e} \left(1 - \alpha^{i_j} x\right) . \tag{3.15}$$

In this context $i_1, \ldots, i_e$ are the indices of the e erasures.

With the error/erasure locator polynomial at hand, we re-write (3.12) as

$$\begin{aligned} \Omega(x) &= S(x)\Gamma(x) \mod x^{2t} \\ &= S(x)E(x)\Lambda(x) \mod x^{2t} , \end{aligned} \tag{3.16}$$

with $S(x)$ and $E(x)$ known, and $\Omega(x)$ and $\Gamma(x)$ of degree less than or equal to $t' + e$. As before, we use the Euclidean algorithm to compute $\Omega(x)$ and $\Lambda(x)$. Then, the roots of $\Gamma(x)$ will give us the locations of the errors, and their values are computed as follows:

$$e_j = -\frac{\alpha^{-(c-1)j}\Omega(\alpha^{-j})}{\Gamma'(\alpha^{-j})} . \tag{3.17}$$

It is easy to show that (RUNE-129) is a generalized BCH code correcting 14 errors, since $\alpha = x^5 + 4x^4 + 5x^2 + 6x$ defined over $\mathbb{Z}_7/(x^6 + 6x^5 + 2x^3 + 6x + 1) \simeq \mathbb{F}_{7^6}$ is a primitive 43-th root of unity, and the 28 consecutive powers $\alpha^8 \ldots \alpha^{35}$ are all roots of (3.4). This means that through Forney's algorithm we can decode t errors and e erasures up to $2t + e \leq 28$, thus quite close to the code's limit.

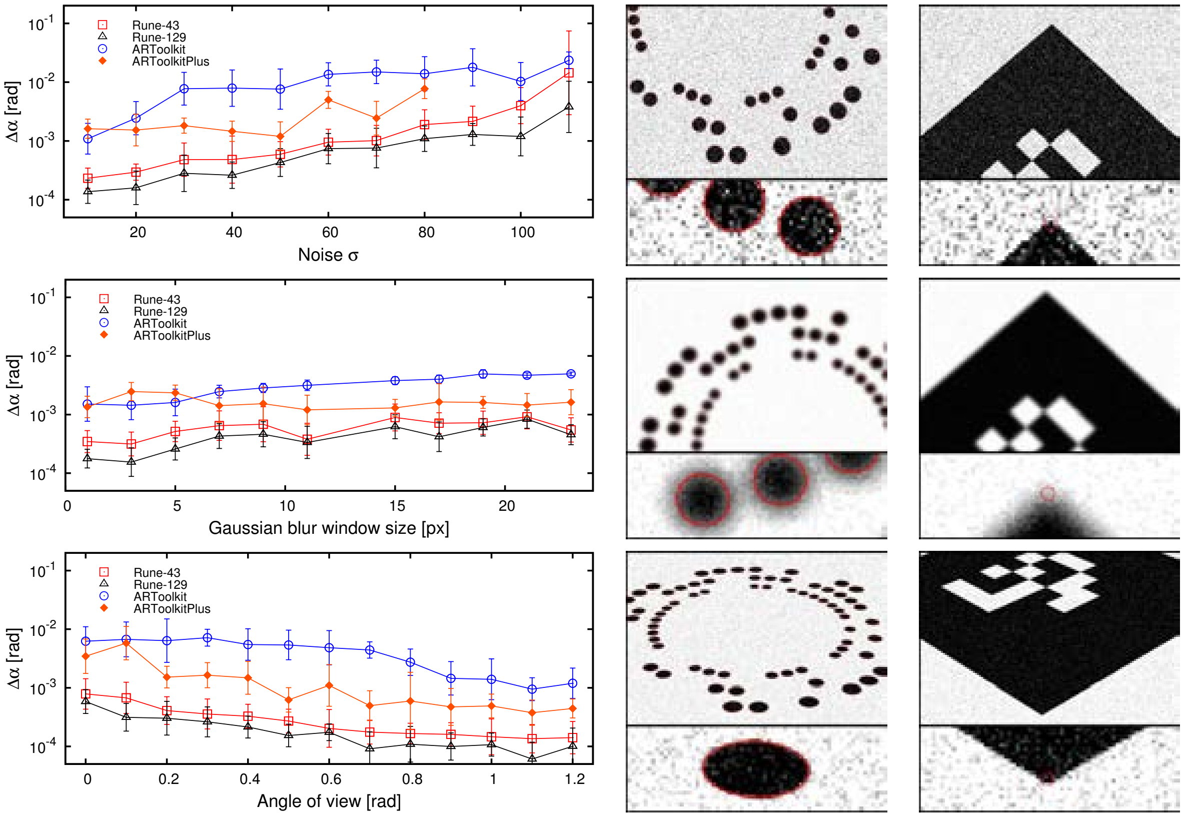


Figure 3.6: Evaluation of the accuracy in the camera pose estimation with respect to different scene conditions. Examples of the detected features are shown for RUNE-129 (first image column) and ARToolkitPlus (second image column).

In the case of (RUNE-43), the code cannot be reduced to a generalized BCH, however, $\alpha = x^7 + x^5 + x^4 + x^2 + x + 1$ defined over $\mathbb{Z}_2/(x^{14} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^4 + x^3 + 1) \simeq \mathbb{F}_{2^{14}}$ is a primitive 43-th root of unity, and has 3 ranges of 6 consecutive powers that are roots of (3.3), namely $\alpha^1 \ldots \alpha^6$, $\alpha^{19} \ldots \alpha^{24}$, and $\alpha^{37} \ldots \alpha^{42}$. Limiting the correction capabilities to 5 errors, we have 8 Newton equations using only known syndromes. Consequently, we can solve the system of linear equations in $\mathbb{F}_{2^{14}}$ to find the error locator polynomial and, thus, the error locations. Since the code is binary, the error values are all 1 and we only need to flip the codes at the error locations. Note that we can detect whether the sequence has more than 5 errors by the fact that the locator polynomial has roots that are neither 0 nor powers of α, leaving us with a margin around the decision boundary of the codewords where we detect but cannot correct the error. This is not a major problem in our application where it is arguably safer to ignore a tag with enough errors to be adjacent to the decision boundary.

Code Alignment

The alignment of the code is performed through an integer Fourier Transform. Let k be an integer such that $r = kn+1$ is prime, and let α be a primitive element of the multiplicative group $\mathbb{Z}_r\times$, i.e., $\alpha \neq 0$ and $\alpha^i \neq 1$ for all $i < r-1$. Under these assumptions α^k has a prime period n. Assume also that for the given k, α combination $\alpha^k > q$. In our cases with $n = 43$ we have $k = 4$, $r = 4 \cdot 43 + 1 = 173$ prime, and 2 a primitive element of $\mathbb{Z}_{173}$. Further $2^4 = 16 > 7 > 2$.

Given a codeword sequence $c_0, \dots, c_{n-1}$ of integers between 0 and $q-1$, we define the Fourier transform in $\mathbb{Z}_r$ of this sequence as

$$C_i = \sum_{j=0}^{n-1} \alpha^{kij} c_j \mod r\,. \tag{3.18}$$

We define the phase of a Fourier coefficient C_i as $\phi_i = \log_\alpha(C_i)$, i.e., the unique integer $0 \leq \phi_i \leq r-2$ such that $\alpha^{\phi_i} = C_i \mod r$. With this definition we compute the shift of the code as $l = \lfloor \phi_1 / k \rfloor$ and the Fourier Transform of the aligned code as

$$\hat{C}_i = \alpha^{-kli} C_i \mod r\,. \tag{3.19}$$

Note that for constant codewords, $C_1 = \cdots = C_{n-1} = 0$ so the shift is not well defined, while in all other cases we have $C_1 \neq 0$. Also, under this rotation $0 \leq \log_\alpha(\hat{C}_1) < k$, thus we are minimizing the phase of $\hat{C}_1$. Once the aligned Fourier Transform is at hand, we can compute the aligned codeword sequence $\hat{c}_0, \dots, \hat{c}_{n-1}$ using the inverse Fourier Transform in $\mathbb{Z}_r$:

$$\hat{c}_i = n^{-1} \sum_{j=0}^{n-1} \alpha^{-kij} \hat{C}_j \mod r\,. \tag{3.20}$$

3.1.2 Experimental Validation

We tested our proposed fiducial markers in many different ways. To start, in section 3.1.2 and 3.1.2 we assessed the pose estimation accuracy compared to the ARToolkit [123] and ARToolkitPlus [227] which are deemed as a de-facto standard markers for augmented reality applications. Such tests are performed synthetically by rendering different frames varying an additive Gaussian noise, blur, illumination gradient and random occlusions.

Furthermore, driven by the good localization accuracy and occlusion resilience of the composing circular features, we tested RUNE-Tags as targets for camera calibration and object measurement. In section 3.1.2 we simulated a mono camera calibration scenario while in 3.1.2 we compared the camera pose estimation for both the mono and the stereo case. Also, we assessed the repeatability achievable while estimating the distance between two joint tags moving in a scene.

Finally, in addition to the evaluation with synthetic images, in section 3.1.2 we performed some qualitative tests on real videos.

Accuracy and Baseline Comparisons

In Fig. 3.6 the accuracy of our markers with calibrated cameras is evaluated. In the first test, an additive Gaussian noise was added to images with an average view angle of 0.3 radians and no artificial blur added. The performance of all methods get worse with increasing levels of noise and ARToolkitPlus, while in general more accurate than ARToolkit, breaks when dealing with a noise with a std. dev. greater than 80 (pixel intensities goes from 0 to 255). Both RUNE-43 and RUNE-129 always recover a more faithful pose. We think that this is mainly due to the larger number of correspondences used to solve the PnP problem. In fact, we can observe that in all the experiments RUNE-129 performs consistently better than RUNE-43.

Unlike additive noise, Gaussian blur seems to have a more limited effect on all the techniques. This is mainly related to the fact that all of them performs a preliminary edge detection step, which in turn applies a convolution kernel. Thus is somewhat expected that an additional blur does not affect severely the marker localization. Finally, it is interesting to note that oblique angles lead to an higher accuracy (as long as the markers are still recognizable). This is explained by observing that the constraint of the reprojection increases with the angle of view. Overall, these experiments confirm that Rune-Tag always outperforms the other two tested techniques by about one order of magnitude. In practical terms the improvement is not negligible, in fact an error as low as 10^{-3} radians still produces a jitter of 1 millimeter when projected over a distance of 1 meter. While this is a reasonable performance for augmented reality applications, it can be unacceptable for obtaining precise contactless measures.

RUNE Tags for camera calibration

Since RUNE-129 provides an extremely robust yet precise way to localize many circular features we tried to use the proposed markers as a calibration target. In most cases, camera calibration is performed by exposing a well known pattern to the camera in many different point of views. This allows the gathering of many 3D-2D point correspondences used to simultaneously recover the target pose, camera intrinsics parameters, and the lens distortion. Most of the time, a chessboard pattern is used since it provides a good set of feature points in the form of image corners. However, a manual chessboard boundary identification process is required for the limited robustness of such patterns against occlusions or illumination gradients. As a consequence, our fiducial markers may provide a very interesting alternative when an automatic calibration procedure is sought or an optimal visibility of the target cannot be guaranteed.

In Fig.3.7 we show the calibration results while calibrating a camera using a single RUNE-129 as calibration target and by varying the number of exposures used for each calibration. Specifically, we divided the camera poses (as given by PnP) into 3 major groups with respect to the angle between the camera z-axis and the marker plane normal. For each group, more than 200 photos are taken and a random subset of them are selected for each test to compose the plot. The ground truth is provided by a calibration performed

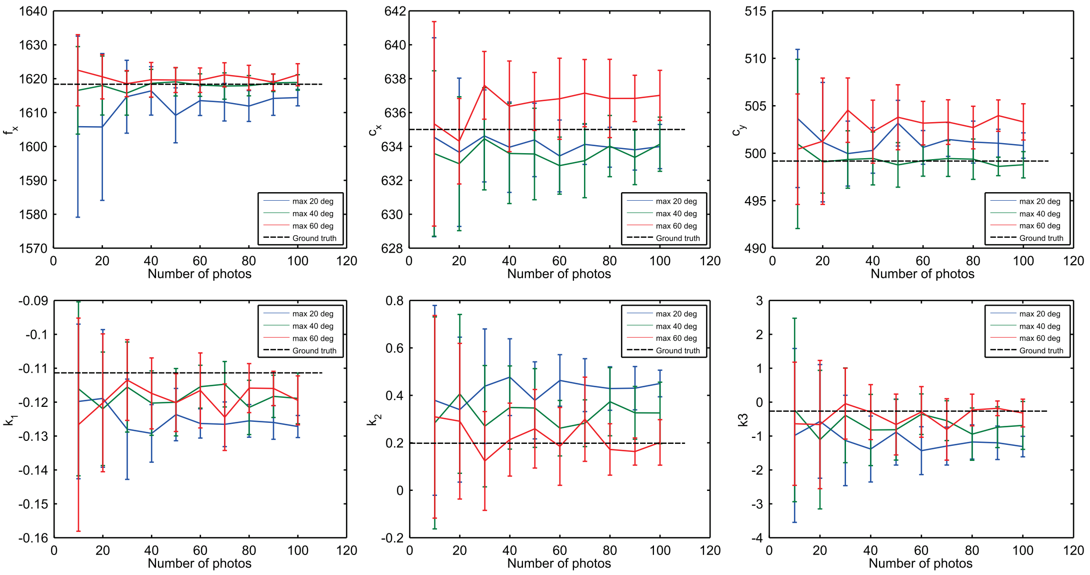


Figure 3.7: Accuracy of camera calibration when using a single RUNE-129 as a dot-based calibration target. Camera poses has been divided into 3 groups based on the maximum angle between the camera z-axis and the marker plane. A random subset of photos is used to test the calibration varying the number of target exposures. In all the experiments we achieve a good accuracy with a decreasing st.dev. when increasing the number of photos.

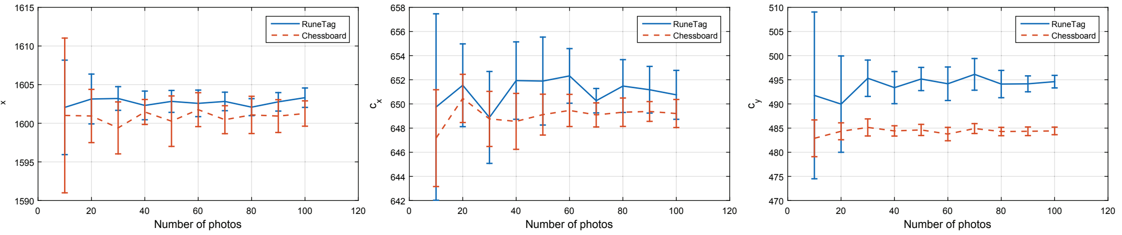


Figure 3.8: Comparison between a calibration performed with RUNETag and chessboard target

with a 20×20 chessboard target exposed in 200 different poses using the method described in [20] to limit the errors due to printing misalignments. Camera calibration is performed by using the common technique described in [251] implemented by the OpenCV library [43].

Some interesting facts can be observed. First, the standard deviation of all the estimated parameters decrease by increasing the number of photos. This is an expected behaviour that agrees with the accuracy experiments presented in section 3.1.2. Indeed, the more the number of target feature points given, the more the calibration error can be reduced by the non-linear optimization process. Second, the focal length estimation tends to be more accurate while considering the target poses spanning trough a greater range of angles (i.e. between 0 and 60 degrees). Differently, optical center seems to behave in the opposite way, giving better results when keeping the target plane more parallel to the image plane. This is probably due to the well known localization bias of the ellipse centers [150]. Third, the first two radial distortion parameters (i.e. k_1 and k_2) behave respectively like the optical center and the focal length. It has to be noted that a precise localization of the ellipse centers is only achievable in absence of distortion since the projective invariance of conics holds only for pure central projections. Therefore, we think that the calibration performance can be improved by estimating an initial calibration assuming no radial distortion followed by an iterative undistortion and re-localization of the circular features and a re-calibration of the camera intrinsics. Finally, we obtained no completely wrong calibrations due to mis-detections of the target thanks to the extremely resilient coding scheme used for markers identification.

In Fig.3.8 we compared the recovered camera instrinsic parameters varying the number of shots for RuneTag and a standard 10x10 Chessboard calibration target, calibrated with the method described in [251]. Both the two approaches shows comparable results, with the standard deviation decreasing while increasing the number of target exposures. Overall, chessboard target provides more stable results for optical center while RuneTag target performs better in the focal length estimation. This may be caused by the radial structure of RuneTag target that may partially hinder the optical center estimation.

Mono vs. Stereo Pose Estimation

To further test the camera pose estimation accuracy we compared the results achievable comparing a single camera setup (using PnP algorithm) with a calibrated stereo setup that can recover the pose by means of a 3D reconstruction of the marker followed by an estimation of the rigid motion with respect to the known model.

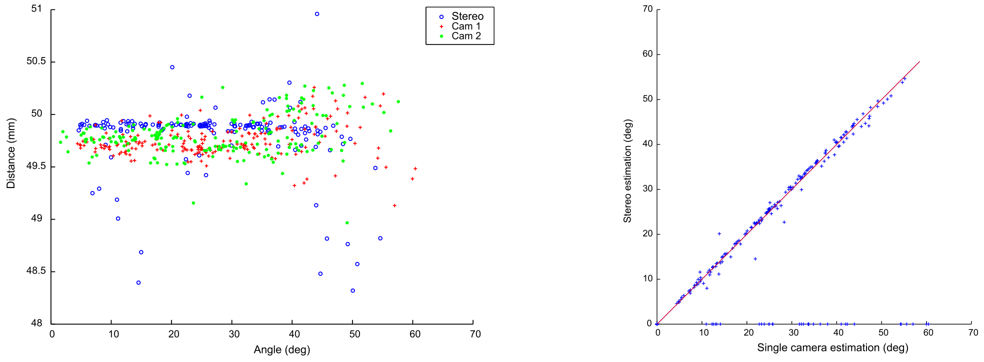


Figure 3.9: Comparison between the pose accuracy for a single or stereo camera setup. Left: distance between two jointly moving markers as a function of the angle with respect to the first camera. Right: Angle around the marker plane normal as estimated by the first camera versus the stereo setup. Ideally, all the measures should lie on the 45 degrees red line.

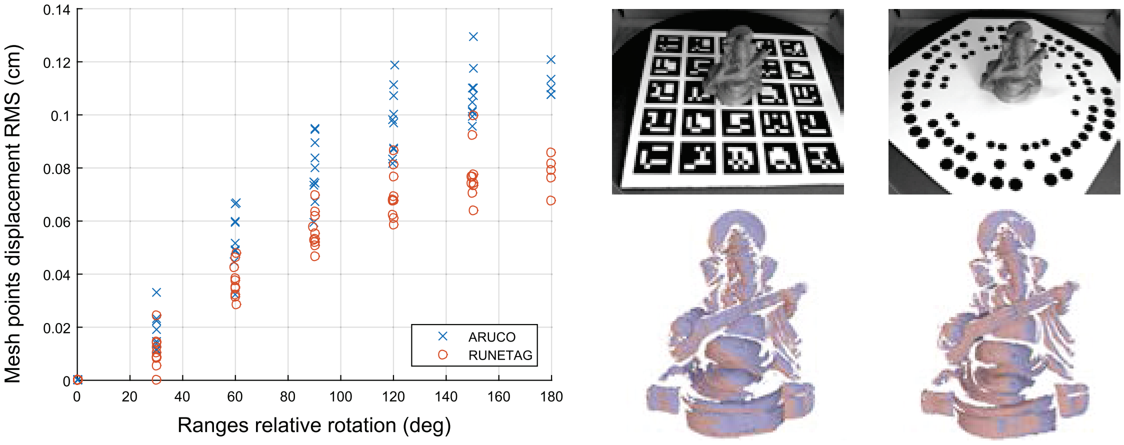


Figure 3.10: Real-world reconstruction scenario. ArUco and RUNETag were placed on top of a scanner turntable to provide the initial coarse estimation for rangemap alignment. See text for details.

We started by calibrating a stereo rig using a marker-based target as described in section 3.1.2. Then, we firmly positioned two RUNE-129 tags to a rigid metal rod so that they could only be moved without changing their relative position.

In the first experiment (Fig. 3.9, Left) we plotted the unknown distance between the two markers as estimated only by the first camera (in red), by the second (in green) and by using stereo reconstruction (blue) as a function of the angle between the first marker and the first camera. It can be noted that the stereo case exhibit lower variance with respect to single-camera scenarios with some sparse outliers happening when the entire marker is not visible by both the cameras. Moreover, the distance measured by the mono case tends to be a little lower than the stereo one if the angle is below 30 degrees while increasing significantly for higher angles. This behaviour is probably due to the PnP algorithm that suffers for a non-isotropic error with respect to the three camera axis (i.e. the localization error on the camera z-axis is higher than the other two).

In (Fig. 3.9, Right) we compared the angle around the plane normal of a single RUNE-129 tag for mono (using the first camera) versus the stereo case. Ideally, the ratio between the two measures should be exactly 1 and so all the points should be disposed on the 45 degrees red line shown in the plot. We can observe that most of the measures are equally distributed above and below such line indicating no significant bias. This behaviour is consistent for all the angles spanning between 10 and 60 degrees since the overall geometrical shape of all the dots (i.e. minor and major axis length) remains constant if a rotation around the marker plane normal is applied. This suggests that the proposed tags may be used as a coarse registration initialization for a 3D scanner turntable.

We tested this possible real-world scenario to compare our proposed tags against the recently developed ArUco Tags [96] which exhibit similar reliability under heavy occlusions. In (Fig. 3.10, Top-Right) we show two pictures of our setup, with a test object placed on top of one of the two kind of tags covering the entire turntable. To test the

Occlusion	0%	10%	20%	50%	70%
RUNE-43	100%	69%	40%	0%	0%
RUNE-129	100%	100%	100%	100%	67%

Table 3.2: Recognition rate of the two proposed marker configurations with respect to the percentage of area occluded.

rangemap registration accuracy, we took 10 ranges spanning a complete 360 turn of the turntable. Then, for each couple of ranges, pose recovered with each tag was used to provide an initial coarse alignment which has been further refined with ICP. In (Fig. 3.10, Left) we plotted the distance between the initial coarse and the refined configuration of each range couple, in terms of root mean square distance of corresponding rangemap points. The rationale is that the worse the pose provided by the tag, the more ICP has to move each range for the fine alignment. In the plot we can clearly see how RuneTag provides lower RMS (so less ICP displacement was necessary) at any initial relative rotation. For both the tags, the initial relative rotation proportionally affects the pose estimation error. Finally, in (Fig. 3.10, Bottom-Right) we show a qualitative example of the coarse estimation provided by ArUco (Left) and RuneTag (Right).

Resilience to Occlusion and Illumination

One of the main characteristics of Rune-Tag is that it is very robust to occlusion. In section 3.1.1 we observed that RUNE-129 can be used to distinguish between about 20.000 different tags and still be robust to occlusions as large as about 67% of the dots. By choosing different cyclic coding schemes is even possible to push this robustness even further, at the price of a lower number of available tags. In the first column of Fig. 3.11 we show how occlusion affects the accuracy of the pose estimation (i.e. how well the pose is estimated with fewer dots regardless to the ability of recognize the marker). Albeit a linear decreasing of the accuracy with respect to the occlusion can be observed, the precision is still quite reasonable also when most of the dots are not visible.

In Table 3.2 we show the recognition rate of the two proposed designs with respect to the percentage of marker area occluded. In the second column of Fig. 3.11 the robustness

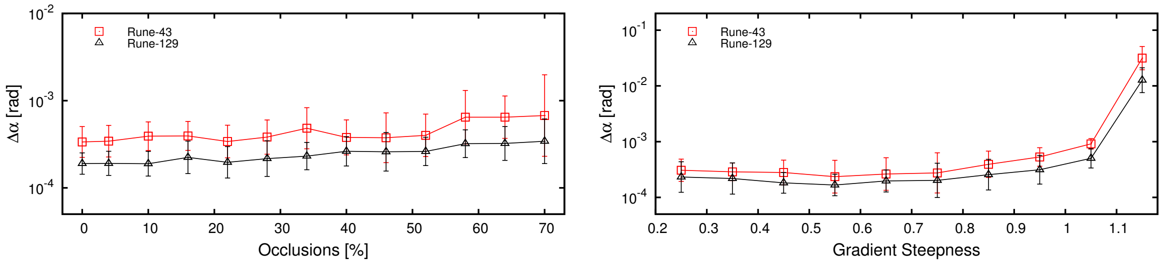


Figure 3.11: Evaluation of the accuracy in the camera pose estimation of RUNE-Tag with respect to occlusion (left column) and illumination gradient (right column).

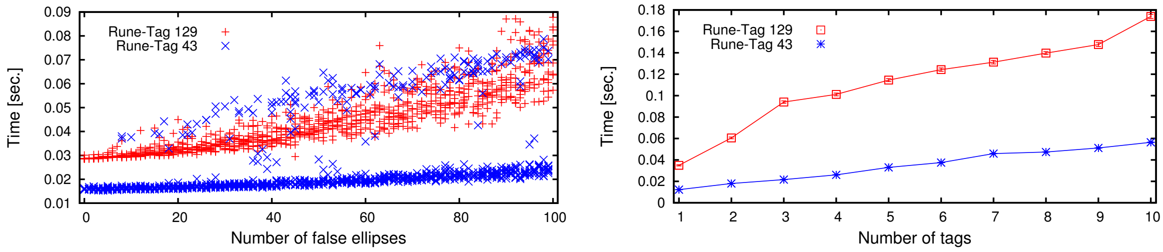


Figure 3.12: Evaluation of the recognition time respectively when adding artificial false ellipses in the scene (left column) and with several markers (right column).

to illumination gradient is examined. The gradient itself is measured unit per pixel (i.e. quantity to add to each pixel value for a each pixel of distance from the image center). Overall, the proposed methods are not affected very much by the illumination gradient and break only when it become very large (in our setup an illumination gradient of 1 implies that pixels are completely saturated at 255 pixels from the image center). This agrees with the fact that a precise sub-pixel ellipse contour estimation is quite robust to steep changes in scene illumination.

Performance Evaluation

Our tag system is designed for improved accuracy and robustness rather than for high detection speed. This is quite apparent in Fig. 3.12, where we can see that the recognition could require from a minimum of about 15 ms (RUNE-43 with one tag an no noise) to a maximum of about 180 ms (RUNE-129 with 10 tags) with a consumer Core2 Duo PC, 2Ghz clock. By comparison ARToolkitPlus is about an order of magnitude faster [227]. However, it should be noted that, despite being slower, the frame rates reachable by Rune-Tag (from 60 to about 10 fps) can still be deemed as usable even for real-time applications (in particular when few markers are viewed at the same time).

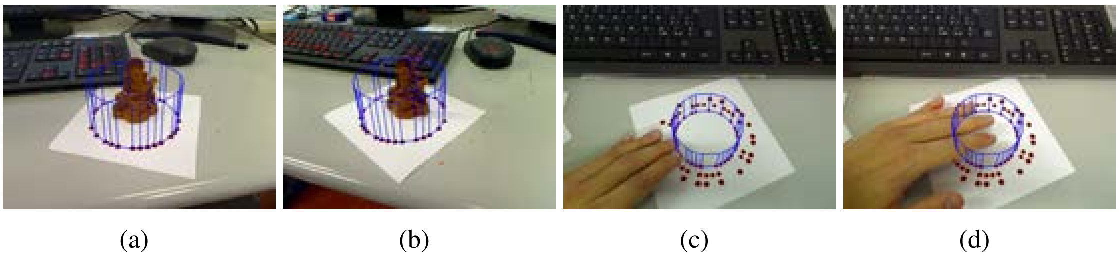

(a) (b) (c) (d)

Figure 3.13: Some examples of behaviour in real videos with occlusion. In (a) and (b) an object is placed inside the marker and the setup is rotated. In (c) and (d) the pose is recovered after medium and severe occlusion.

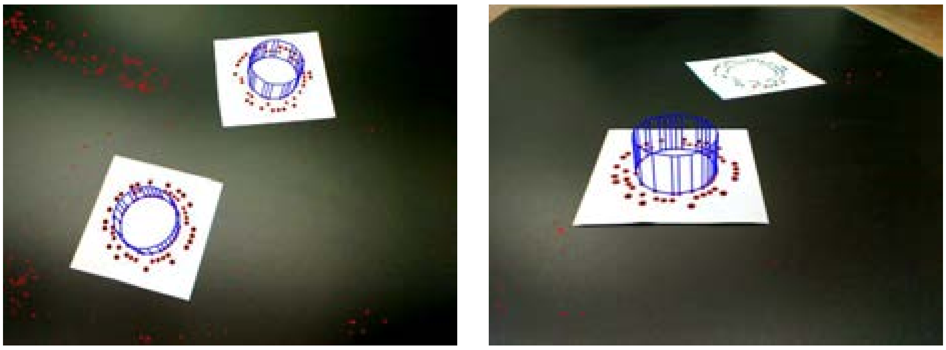


Figure 3.14: Recognition fails when the marker is angled and far away from the camera and the ellipses blends together.

Shortcomings and Limitations

In Fig. 3.13 some experiments with common occlusion scenarios are presented. In the first two shots an object is placed inside a RUNE-43 marker in a typical setup used for image-based shape reconstruction. In the following two frames a RUNE-129 marker is tested for its robustness to moderate and severe occlusion. At last, in Fig. 3.14 an inherent shortcoming of our design is highlighted. The high density exhibited by the more packed markers may result in a failure of the ellipse detector whereas the tag is far away from the camera or very angled, causing the dots to become too small or blended.

3.2 A Robust Multi-Camera 3D Ellipse Fitting

Ellipses are a widely used cue in many 2D and 3D object recognition pipelines. In fact, they exhibit a number of useful properties. First, they are naturally occurring in many man-made objects. Second, the projective invariance of the class of ellipses makes them detectable even without any knowledge of the acquisition parameters. Finally, they can be represented by a compact set of parameters that can be easily adopted within optimization tasks. While a large body of work exists in the literature about the localization of ellipses as 2D entities in images, less effort has been put in the direct localization of ellipses in 3D, exploiting images coming from a known camera network. We propose a novel technique for fitting elliptical shapes in 3D space, by performing an initial 2D guess on each image followed by a multi-camera optimization refining a 3D ellipse simultaneously on all the calibrated views. The proposed method is validated both with synthetic data and by measuring real objects captured by a specially crafted imaging head. Finally, to evaluate the feasibility of the approach within real-time industrial scenarios, we tested the performance of a GPU-based implementation of the algorithm.

3.2.1 Introduction

Among all the visual cues, ellipses offer several advantages that prompt their adoption within many machine vision tasks. To begin with, the class of ellipses is invariant to projective transformations, thus an elliptical shape remains so when it is captured from any viewpoint by a pinhole camera [74]. This property makes easy to recognize objects that contain ellipses [104, 152] or partially elliptical features [205]. When the parameters of one or more coplanar 3D ellipses that originated the projection are known, the class of homographies that make it orthonormal to the image plane can be retrieved. This is a useful step for many tasks, such as the recognition of fiducial markers [34, 162], orthonormalization of playfields [101], forensic analysis of organic stains [239] or any other planar metric rectification [61]. Furthermore, ellipses (including circles) are regular shapes that often appear in manufactured objects and can be used as optical landmarks for tracking and manipulation [246] or measured for accurate in-line quality assurance [192].

Because of their usefulness and broad range of applicability, it is not surprising that ellipse detection and fitting methods abound in the literature. In particular, when points belonging to the ellipse are known, they are often fitted through ellipse-specific least square methods [93]. In order to find co-elliptical points in images, traditional parameter-space search schemas, such as RANSAC or Hough Transform, can be employed. Unfortunately, the significantly high dimensionality of 2D ellipse parametrization (which counts 5 degrees of freedom) makes the direct application of those techniques not feasible. For this reason a lot of efficient variants have appeared. Some try to reduce the number of samples for a successful RANSAC selection [204,241]. Others attempt to escape from the curse of dimensionality that plagues the Hough accumulator [64, 154]. If high accuracy is sought, point-fitted ellipses can be used as an initial guess to be refined through intensity-based methods. Those approaches allow to obtain a sub-pixel estimation by exploiting the raw

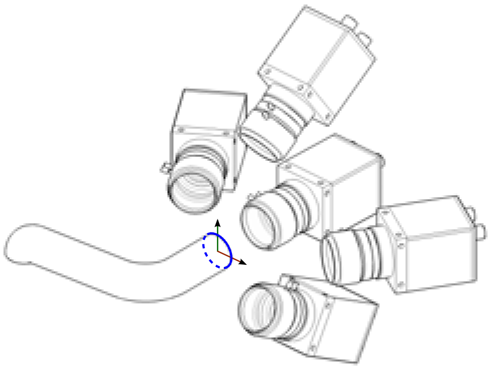


Figure 3.15: Schematic representation of a multi-camera system for industrial in-line pipes inspection.

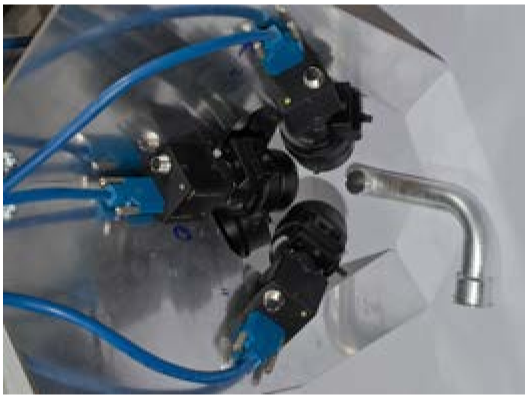


Figure 3.16: The experimental Multiple-camera imaging head.

gradient of the image [170] or by preserving quantities such as intensity moments and gradients [108]. Multiple view geometry has also been exploited to get a better 3D ellipse estimation. In [220], multiple cameras are used to track an elliptical feature on a glove to obtain the estimation of the hand pose. The ellipses fitted in the images are triangulated with the algorithm proposed in [178] and the best pair is selected. In [149], holes in metal plates and industrial components are captured by a couple of calibrated cameras and the resulting conics are then used to reconstruct the hole in the Euclidean space. Also in [85] the intersection of two independently extracted conics is obtained through a closed form. All these approaches, however, exploit 3D constraints in an indirect manner, as triangulation always happens on the basis of the ellipses fitted over 2D data.

In this work we present a rather different technique that works directly in 3D space. Specifically, we adopt a parametric level-set appraoch, where the parameters of a single elliptical object that is observed by a calibrated network of multiple cameras (see Fig.3.15) are optimized with respect to an energy function that simultaneously accounts for each point of view. The goal of our method is to bind the 2D intensity and gradient-based energy maximization that happens within each image to a common 3D ellipse model. The performance of the solution has been assessed through both synthetic experiment and by applying it to a real world scenario. Finally, to make the approach feasible regardless of the high computational requirements, we propose a GPU implementation which performance has been compared with a well optimized CPU-based version.

3.2.2 Multiple Camera Ellipse Fitting

In our approach we are not seeking for independent optima over each image plane, as is the case with most ellipse fitting methods. Rather, our search domain is the parametrization of an ellipse in the 3D Euclidean space, and the optimum is sought with respect to its concurrent 2D reprojections over the captured images. In order to perform such optimization we need to sort out a number of issues. The first problem is the definition of a

3D ellipse parametrization that is well suitable for the task (that is, it makes easy to relate the parameters with the 2D projections). The second one, is the definition of an energy function that is robust and accounts for the usual cues for curve detection (namely the magnitude and direction of the intensity gradient). The last issue is the computation of the derivative of the energy function with respect to the 3D ellipse parameters to be able to perform a gradient descent.

Parameterization of the 3D Ellipse

In its general case, any 2-dimensional ellipse in the image plane is defined by 5 parameters, namely: the length of the two axes, the angle of rotation and a translation vector with respect to the origin.

In matrix form it can be expressed by the locus of points $\mathbf{x} = \begin{pmatrix} x_1 & x_2 & 1 \end{pmatrix}^T$ in homogeneous coordinates for which the equation $\mathbf{x}^T A \mathbf{x}^T = 0$ holds, for

$$A = \begin{pmatrix} a & b & d \\ b & c & f \\ d & f & g \end{pmatrix} \tag{3.21}$$

with $\det(A) < 0$ and $ac - b^2 > 0$.
In the 3-dimensional case it is subjected to 3 more degrees of freedom (i.e. rotation around two more axes and the z-component of the translation vector). More directly, we can define the ellipse by first defining the plane T it resides on and then defining the 2D equation of the ellipse on a parametrization of such plane. In particular, let $\mathbf{c} = (c_1, c_2, c_3, 1)^T \in T$ be the origin of the parametrization, and $\mathbf{u} = (u_1, u_2, u_3, 0)^T$, $\mathbf{v} = (v_1, v_2, v_3, 0)^T$ be the generators of the linear subspace defining T, then each point on the 3D ellipse will be of the form $\mathbf{o} + \alpha\mathbf{u} + \beta\mathbf{v}$ with α and β satisfying the equation of an ellipse.

By setting the origin $\mathbf{o}$ to be at the center of the ellipse and selecting the directions $\mathbf{u}$ and $\mathbf{v}$ appropriately, we can transform the equation of the ellipse on the plane coordinates in such a way that it will take the form of the equation of a circle. Hence, allowing the 3D ellipse to be fully defined by the parametrization of the plane on which the ellipse resides. However, this representation has still one more parameter than the actual degrees of freedom of the ellipse. To solve this we can, without any loss of generality, set $u_3 = 0$, thus, by defining the matrix

$$\mathbf{U_c} = \begin{pmatrix} u_1 & v_1 & c_1 \\ u_2 & v_2 & c_2 \\ 0 & v_3 & c_3 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.22}$$

and the vector $\mathbf{x} = (\alpha, \beta, 1)^T$, we can express any point p in the 3D ellipse as:

$$\mathbf{p} = \mathbf{U_c}\mathbf{x} \quad \text{subject to} \quad \mathbf{x}^T \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \mathbf{x} = 0\,. \tag{3.23}$$

Even if $\mathbf{U_c}$ embeds all the parameters needed to describe any 3d ellipse, it is often the case that an explicit representation through center $\mathbf{c}$ and axes $\mathbf{a_1}, \mathbf{a_2} \in R^3$ is needed. Let $\mathbf{U}$ be the 3×2 matrix composed by the first two columns of $\mathbf{U_C}$. The two axes $\mathbf{a_1}, \mathbf{a_2}$ can be extracted as the two columns of the matrix:

$$\mathbf{K} = \begin{pmatrix} | & | \\ \mathbf{a_1} & \mathbf{a_2} \\ | & | \end{pmatrix} = \mathbf{U}\phi^{\mathbf{T}}$$

where $\phi^{\mathbf{T}}$ is the matrix of left singular vectors of $\mathbf{U^T U}$ computed via SVD decomposition. The vector $\mathbf{c}$ is trivially composed by the parameters $\begin{pmatrix} c_1 & c_2 & c_3 \end{pmatrix}^T$.

Conversely, from two axes $\mathbf{a_1}, \mathbf{a_2}$, the matrix $\mathbf{U}$ can be expressed as:

$$\mathbf{U} = \mathbf{K} \begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$$

by imposing that $\begin{cases} \alpha K_{31} + \beta K_{32} = 0 \\ \alpha^2 + \beta^2 = 1 \end{cases}$. Finally, once $\mathbf{U}$ has been computed, the 3D ellipse matrix can be composed in the following way:

$$\mathbf{U_c} = \begin{pmatrix} \mathbf{U} & \mathbf{c} \\ \mathbf{0} & 1 \end{pmatrix}$$

Finally, with this parametrization it is very easy to obtain the equation of the ellipse projected onto any camera. Given a projection matrix $\mathbf{P}$, the matrix $\mathbf{A_P}$ describing the 2-dimensional ellipse after the projection can be expressed as:

$$\mathbf{A_P} = (\mathbf{PU_c})^{-T} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} (\mathbf{PU_c})^{-1} \tag{3.24}$$

Energy Function over the Image

To estimate the equation of the 3D-ellipse we set-up a level-set based optimization schema that updates the ellipse matrix $\mathbf{U_c}$ by simultaneously taking into account its re-projection in every camera of the network. The advantages of this approach are essentially threefold. First, the equation of the 3D ellipse estimated and the re-projection in all cameras are always consistent. Second, erroneous calibrations that affects the camera network itself can be effectively attenuated, as shown in the experimental section. Third, the ellipse can be partially occluded in one or more camera images without heavily hindering the fitting accuracy.

In order to evolve the 3D ellipse geometry to fit the observation, we need to define the level set functions $\varphi_i : R^2 \rightarrow R$ describing the shape of the ellipse $\mathbf{U_c}$ re-projected to the

i^{th} camera. Given each level set, we cast the multiview fitting problem as the problem of maximizing the energy function:

$$E_{I_1 \dots I_n}(\mathbf{U_c}) = \sum_{i=1}^{n} E_{I_i}(\mathbf{U_c}) \tag{3.25}$$

Which sums the energy contributions of each camera:

$$\begin{aligned} E_{I_i}(\mathbf{U_c}) &= \int_{R^2} \langle \nabla H(\varphi(\mathbf{x})), \nabla I_i(\mathbf{x}) \rangle^2 \mathrm{d}x & (3.26) \\ &= \int_{R^2} \langle H'(\varphi(\mathbf{x})) \nabla \varphi(\mathbf{x}), \nabla I_i(\mathbf{x}) \rangle^2 \mathrm{d}x \,, & (3.27) \end{aligned}$$

where H is a suitable relaxation of the Heavyside function. In our implementation, we used:

$$H(t) = \frac{1}{1 + e^{-\frac{t}{\sigma}}} \tag{3.28}$$

where parameter σ models the band size (in pixels) of the ellipse region to be considered. By varying σ we can manage the trade-off between the need of a regularization term in the energy function to handle noise in the image gradient and the estimation precision that has to be achieved.

The level set for a generic ellipse is rather complicated and cannot be easily expressed in closed form, however, since it appears only within the Heavyside function and its derivative, we only need to have a good analytic approximation in the boundary around the ellipse. We approximate the level set in the boundary region as:

$$\varphi_i(\mathbf{x}) \approx \frac{\mathbf{x}^T \mathbf{A}_i \mathbf{x}}{2\sqrt{\mathbf{x}^T \mathbf{A_i}^T \mathbf{I_0} \mathbf{A_i} \mathbf{x}}} \tag{3.29}$$

Where $I_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ and $\mathbf{A_i}$ is the re-projection of the ellipse $\mathbf{U_c}$ into the i^{th} camera computed using equation (3.24). The function has negative values outside the boundaries of the ellipse, positive values inside and is exactly 0 for each point $\{\mathbf{x} | \mathbf{x}^T \mathbf{U_c} \mathbf{x} = 0\}$.

The gradient of the level set function $\nabla \varphi : R^2 \to R^2$ can actually be defined exactly in closed form:

$$\nabla \varphi_i(\mathbf{x}) = \frac{\mathbf{A}_i \mathbf{x}}{\sqrt{\mathbf{x}^T \mathbf{A_i}^T \mathbf{I_0} \mathbf{A_i} \mathbf{x}}} \tag{3.30}$$

.

Starting from an initial estimation, given by a simple triangulation of 2d-ellipses between just two cameras, we maximize the energy function (3.25) over the plane parameters $\mathbf{U_c}$ by means of a gradient scheme.

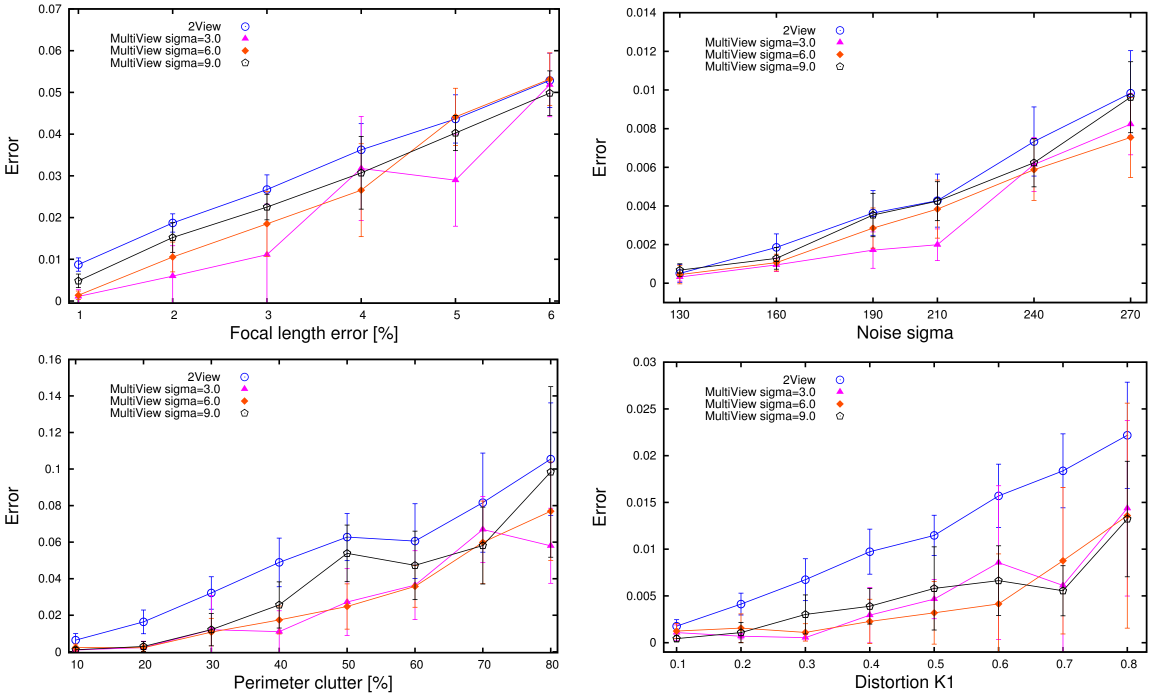


Figure 3.17: Evaluation of the accuracy of the proposed method with respect to different noise sources. The metric adopted is the relative error between the minor axis of the ground truth and of the fitted ellipse.

Gradient of the Energy Function

The gradient of the energy function can be computed as a summation of the gradient of each energy term. This gradient can be obtained by analytically computing the partial derivatives of equation (3.26) with respect to the eight parameters $(p_1 \dots p_8) = (u_1, v_1, c_1, u_2, v_2, c_2, v_3, c_3)$:

$$\begin{aligned}
\frac{\partial}{\partial p_i} E_{I_i}(\mathbf{U_c}) &= \frac{\partial}{\partial p_i} \int\limits_{R^2} E_{I_i}(\mathbf{U_c}, \mathbf{x})^2 \mathrm{d}x \\
&= \int\limits_{R^2} 2E_{I_i}(\mathbf{U_c}, \mathbf{x}) \frac{\partial}{\partial p_i} E_{I_i}(\mathbf{U_c}, \mathbf{x}) \mathrm{d}x
\end{aligned}$$

Where:

$$E_{I_i}(\mathbf{U_c}, \mathbf{x}) = \langle H'(\varphi(\mathbf{x})) \nabla \varphi(\mathbf{x}), \nabla I_i(\mathbf{x}) \rangle$$

and

$$\begin{aligned}
\frac{\partial}{\partial p_i} E_{I_i}(\mathbf{U_c}, \mathbf{x}) =& (\frac{\partial}{\partial p_i} H'(\varphi(x))) \langle \nabla \varphi(\mathbf{x}), \nabla I_i(\mathbf{x}) \rangle + \\
& + H'(\varphi(\mathbf{x})) \langle (\frac{\partial}{\partial p_i} \nabla \varphi(\mathbf{x})), \nabla I_i(\mathbf{x}) \rangle \, .
\end{aligned}$$

The derivatives of the parametric level set functions can be computed analytically. At the beginning of each iteration we compute the derivative of the projected ellipse matrices $\mathbf{A_i}$ which are constant with respect to $\mathbf{x}$:

$$\frac{\partial}{\partial p_i}\mathbf{A_i} = \mathbf{T} + \mathbf{T}^T \tag{3.31}$$

where

$$T = (\frac{\partial}{\partial p_i}[(\mathbf{P_i}\mathbf{U_c})^{-1}])^T \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} (\mathbf{P_i}\mathbf{U_c})^{-1} \tag{3.32}$$

and

$$\frac{\partial}{\partial p_i}[(\mathbf{P_i}\mathbf{U_c})^{-1}] = -(\mathbf{P_i}\mathbf{U_c})^{-1}(\mathbf{P_i}\frac{\partial}{\partial p_i}\mathbf{U_c})(\mathbf{P_i}\mathbf{U_c})^{-1}\,. \tag{3.33}$$

Then, using (3.31), we can compute the level set derivatives for each pixel:

$$\begin{aligned} \frac{\partial}{\partial p_i}\nabla\varphi(\mathbf{x}) = & \frac{(\frac{\partial}{\partial p_i}\mathbf{A_i})\mathbf{x}}{\sqrt{\mathbf{x}^T\mathbf{A_i}^T\mathbf{I_0}\mathbf{A_i}\mathbf{x}}} - \\ - & \frac{\mathbf{A_i}\mathbf{x}(\mathbf{x}^T(\frac{\partial}{\partial p_i}\mathbf{A_i})^T\mathbf{I_0}\mathbf{A_i}\mathbf{x} + \mathbf{x}^T\mathbf{A_i}^T\mathbf{I_0}(\frac{\partial}{\partial p_i}\mathbf{A_i})\mathbf{x})}{2(\mathbf{x}^T\mathbf{A_i}^T\mathbf{I_0}\mathbf{A}\mathbf{x})^{\frac{3}{2}}} \end{aligned} \tag{3.34}$$

$$\frac{\partial}{\partial p_i}\varphi(\mathbf{x}) = \frac{1}{2}\langle\mathbf{x}, \frac{\partial}{\partial p_i}\nabla\varphi(\mathbf{x})\rangle \tag{3.35}$$

$$\frac{\partial}{\partial p_i}H'(\varphi(\mathbf{x})) = H''(\varphi(\mathbf{x}))\frac{\partial}{\partial p_i}\varphi(\mathbf{x})\,. \tag{3.36}$$

By summing the derivative $\frac{\partial}{\partial p_i}E_{I_i}(\mathbf{U_c}, \mathbf{x})$ over all images and all pixels in the active band in each image, we obtain the gradient $\mathbf{G} = \nabla E_{I_1 \ldots I_n}(\mathbf{U_c})$. At this point, we update the 3D ellipse matrix $\mathbf{U_c}$ through the gradient step

$$\mathbf{U_c}^{(t+1)} = \mathbf{U_c}^{(t)} + \eta\mathbf{G} \tag{3.37}$$

where η is a constant step size.

3.2.3 Experimental evaluation

We evaluated the proposed approach both on a set of synthetic tests and on a real world quality control task where we measure the diameter of a pipe with a calibrated multi-camera setup. In both cases, lacking a similar 3D based optimization framework, we compared the accuracy of our method with respect to the results obtained by triangulating ellipses optimally fitted over the single images. The rationale of the synthetic experiments is to be able to evaluate the accuracy of the measure with an exactly known ground truth (which is very difficult to obtain on real objects with very high accuracy). Further,

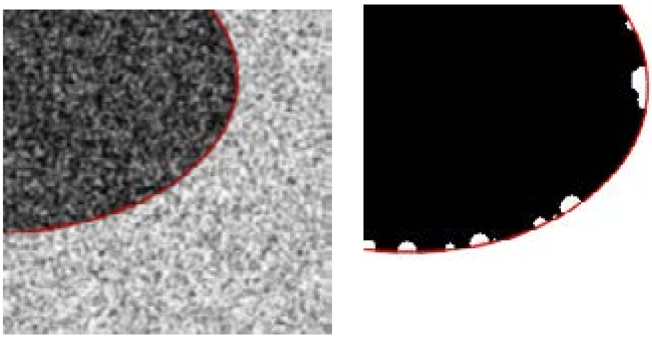


Figure 3.18: Examples of images with artificial noise added. Respectively additive Gaussian noise and blur in the left image and occlusion in the right image. The red line shows the fitted ellipse.

the synthetically generated imagery permits us to control the exact nature and amount of noise, allowing for a separate and independent evaluation for each noise source. By contrast, the setup employing real cameras does not give an accurate control over the scene, nevertheless it is fundamental to asses the ability of the approach to deal with the complex set of distractors that arise from the imaging process (such as reflections, variable contrast, defects of the object, bad focusing and so on). In both cases the ellipse detection is performed by extracting horizontal and vertical image gradients with an oriented derivative of Gaussian filter. Edge pixels are then found by non-maxima suppression and by applying a very permissive threshold (no hysteresis is applied). The obtained edge pixels are thus grouped into contiguos curves, which are in turn fitted to find ellipses candidates. The candidate that exhibit the higher energy is selected and refined using [170]. The refined ellipses are then triangulated using the two images that score the lower triangulation error. The obtained 3D ellipse is finally used both as the result of the baseline method (labeled as *2view* in the following experiments) and as the initialization ellipse for our refinement process (labeled as *multiview*). All the experiments have been performed with 3Mp images and the processing is done with a modern 3.2 Ghz Intel Core i7 PC equipped with Windows 7 Operating System. The CPU implementation was written in C++ and the GPU implementation uses the CUDA library. The video card used was based on the Nvidia 670 chipset with 1344 CUDA cores.

Synthetic Experiments

For this set of experiments we chose to evaluate the effect of four different noise sources over the optimization process. Specifically, we investigated the sensitivity of the approach to errors on the estimation of the focal length and of the radial distortion parameters of the camera and the influence of Gaussian noise and clutter corrupting the images. In Fig. 3.18 examples of Gaussian noise and clutter are shown (note that these are details of the images, in the experiments the ellipse was viewed in full). For each test we created 5 synthetic snapshots of a black disc as seen from 5 different cameras looking at the disk from different points of view (see Fig. 3.15 and Fig. 3.16). The corruption by Gaussian noise has been produced by adding to each pixel a normal distributed additive error of variable value of σ, followed by a blurring of the image with a Gaussian kernel with $\sigma = 6$. The artificial clutter has been created by occluding the perimeter of the disc with a set of random white circles until a given percentage of the original border was corrupted. This simulates the effect of local imaging effect such as the presence of specular highlights that severely affect the edge detection process. The focal length error was obtained by changing the correct focal length of the central camera by a given percentage. Finally, the distortion error was introduced by adding an increasing amount to the correct radial distortion parameter K1.

In Fig. 3.17 we show the results obtained using the baseline triangulation and our optimization with different values of the parameter σ used for the heavyside function (respectively 3, 6 and 9 pixels). As expected, in all the tests performed the relative error grows with the level of noise. In general, all the methods seem to be minimally sensitive to Gaussian noise, whereas the clutter has a big effect even at low percentages. The baseline method performs consistently worse and, among the multiview configurations, the one with lower heavyside band appears to be the most robust for almost all noise levels. This is probably due to the fact that the images have already been smoothed by the gradient calculation step, and thus further smoothing is not required and, to some degree, leads to a more prominent signal displacement.

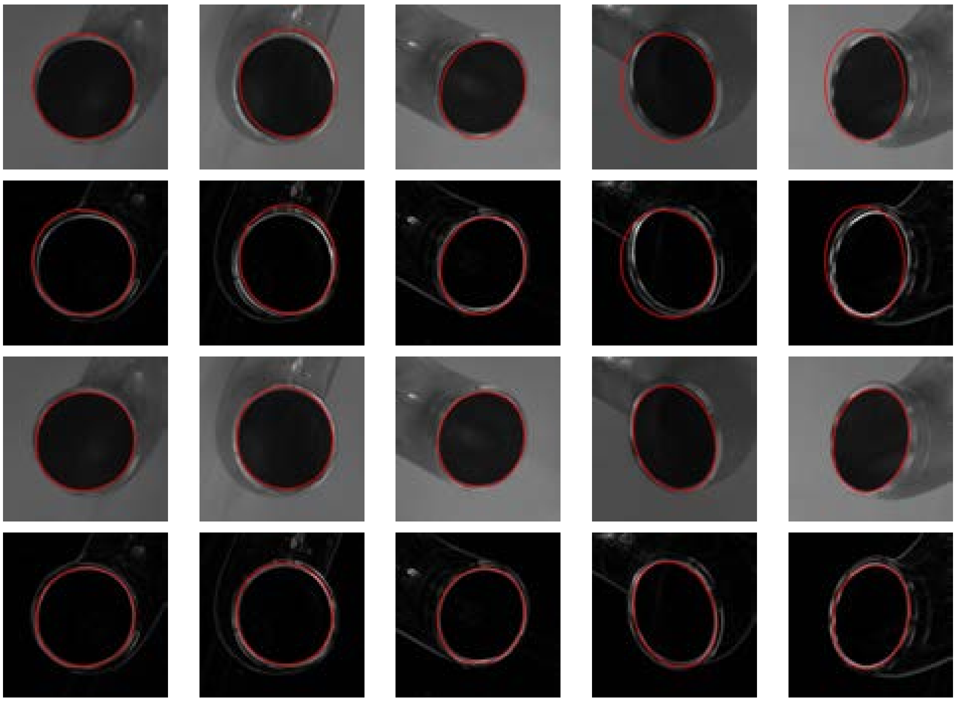


Figure 3.19: Comparison between the accuracy of the initial 2D fitting and the proposed 3D optimization.

Real World Application

For the experiments with real images we built an imaging device that hold 5 PointGrey Flea3 3.2Mp Monochrome USB3 machine vision cameras (see Fig. 3.16). The 5 cameras were calibrated for both intrinsic and extrinsic parameters. We used an aluminium pipe for air conditioning system as the object to be measured, and the imaging head has been supplemented with four high power LEDs in order to get an even illumination of the rim. This is a typical scenario for in-line inspection in manufacturing lines. Additionally, the smooth and polished surface of the pipe offers especially challenging conditions for ellipse detection and refinement, since reflections and changes of contrast tend to create a lot of false elliptical sectors and some highly structured noise.

If Fig. 3.19 a complete qualitative example of the refinement process is shown. In the first two rows of the figure the reprojection of the initially estimated 3D ellipse is overlayed to both the original images and the intensity-coded gradient magnitude. In the remaining rows the reprojection of the optimized 3D ellipse is overlayed over the same images. The images used for the initial triangulation in this specific case were the first and the third. Specifically, the initial guess for the first image was a slightly off-center ellipse fitted in between the edge response produced by the inner and outer rims of the pipe opening (see the gradient image). As a matter of fact, it is immediate to note that these two images exhibits the lower reprojection error, especially for the central camera. However, the other reprojections are rather grossly misaligned with the remaining three points of view. By contrast, almost all the misalignment has been corrected after performing the 3D refinement procedure. While some degree of displacement is still visible in some images, we think that this is mainly due to miscalibration of the extrinsic parameters of the imaging head.

We manually measured the internal and external diameter of the pipe with a caliper (with ± 0.1mm accuracy) obtaining respectively 13.9 and 16.1 mm. However, since the optimization process aim to converge toward the middle of the two rims, it would make no sense to evaluate directly the measurement error committed. Still, the standard de-

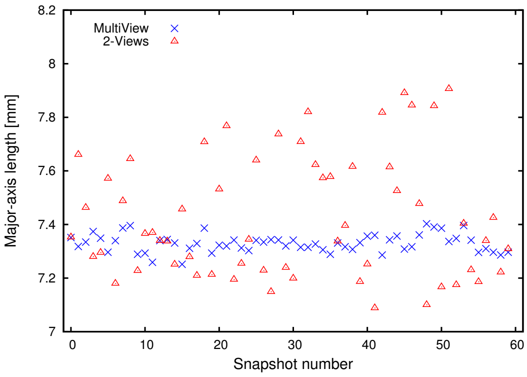


Figure 3.20: Quantitative assessment of the improvement in accuracy.

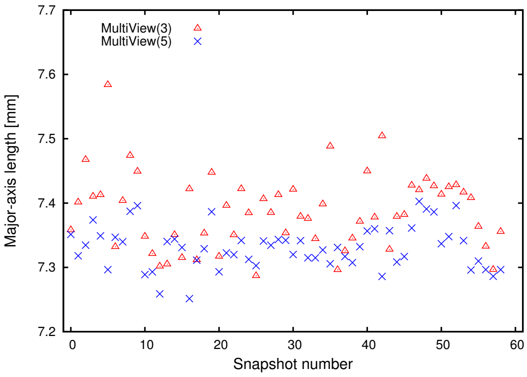


Figure 3.21: Effect of the number of views over the measure quality.

viation of the data with respect to several subsequent measures of the same object from slightly different angles can be considered a good indication of measurement error. Indeed, even if the final measure can be affected by systematic errors, they can be estimated and corrected a-posteriori. In Fig. 3.20 we plotted the measured length of the major axis of the detected 3D ellipse for 60 repeated shots of the pipe opening. The improvement in uncertainty reduction after the refinement step is clearly noticeable as the variance of the measurements is strongly reduced. Indeed, the standard deviation went from 0.23 to 0.03.

All the refinements performed so far have been conducted using 5 points of view. In order to complete our tests it would have been interesting to evaluate if similar accuracy could be obtained using a smaller number of cameras. To this end we disabled two cameras and took further 60 shots of the pipe. The results are plotted in Fig. 3.21. While the dispersion of the measurements is a little higher using only three points of view, it is still noticeably smaller than the one obtained without the optimization step (note that the scales of Fig. 3.20 and Fig. 3.21 are different).

3.2.4 GPU-based Implementation

In a naive implementation, the optimization scheme proposed is quite intensive in terms of raw computing power. Especially for the gradient computation, which requires several matrix and vector multiplications that may easily sum up to an unacceptable total computation time.

However, the intrinsic structure of the problem leads naturally to an implementation in which every pixel in the area of interest defined by the Heavyside function are computed in parallel. After this computation, that can be performed with no required synchronization between each view, a reduction step is needed to aggregate all terms and obtain the final value of the energy and gradient in each iteration.

We implemented the algorithm in C++ with no additional external libraries except for

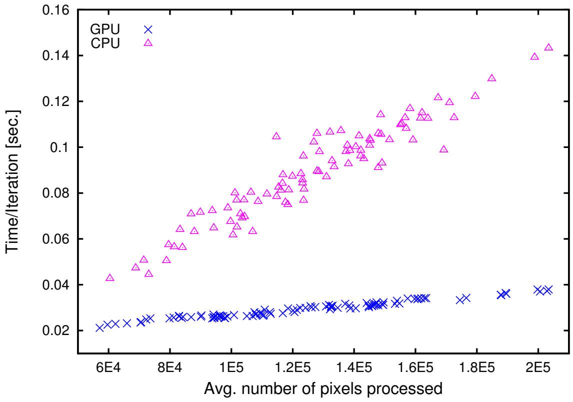


Figure 3.22: Comparison between the running time of the CPU and GPU-based implementations of the multiview algorithm. Times are plotted with respect to the number of pixels in the evaluated mask (i.e. size of the ellipse to be refined).

OpenCV for image IO, OpenMP for CPU parallelization and CUDA for GPU computing. Both the CPU and GPU based implementations are essentially the same, except for the fact that the latter can exploit the massive computing power of modern graphics cards. For every algorithm's iteration, a CPU-based function computes a list of pixel for each image that will be affected by the computation. This list is generated by considering a band around each 2d-ellipse reprojection with a thickness of 5σ pixels and is uploaded to the device memory, together with the optimized parameters and the pre-computed image gradient for each pixel in the list. Once the upload is completed, all available stream processors are used to compute the energy and the energy gradient terms. At the end of the computation steps, all threads are synchronized and 9 values are reduced (energy and the 8 terms of the gradient) to obtain the final values. The total energy is used to track the optimization status and trigger a termination criteria, the gradient is used to adjust the 3d ellipse that is being optimized, moving toward a local maxima.

We tested the execution time per iteration for both the CPU and GPU based implementation of our algorithm (see Fig.3.22) with respect to the average number of pixel processed. In both cases, the process is fast enough to handle a real-time optimization in 3 megapixels images with the fitted ellipse spanning into about 50% of the image. As expected, the GPU implementation performs better than the CPU and exhibits a more consistent running time throughout the tests. This is probably due to the fact that we are dealing with a dedicated hardware. Finally, the synchronization overhead caused by the reductions decreases the performance gap between the two implementations when a relatively low number of pixels are processed, which in turn becomes dramatic when an optimization involving more than 10^5 pixels is needed.

4
Model free camera calibration

Although a wide variety of model-based calibration techniques exist to approximate the imaging system of almost all existing cameras and optics categories, they will eventually lead to hard constrain the calibration result to the limits of the assumptions on which the model bases itself. In [33] Bergamasco et al. show how their fully unconstrained model outperform the standard calibration technique even for standard pinhole plus distortion cameras.

Following their results we improved and applied their method in different situations: In section 4.1 we exploit the unconstrined model to on-line calibrate the projector of a structured-light low cost scanner. The outlier filtering strategy adopted allows us to retain only the good rays observation and increase the scanned surface with respect to the original camera pair, improving at the same time the triangulation accuracy; In section 4.2 we calibrate a light-field camera and use it to re-construct the capture surface with a single shot . This is the ideal field of use of an unconstrained calibration technique since the complex lens setup makes difficult the construction even of a specific model.

Finally, in section 4.3 we present a free distortion model. In this work we step back from the fully unconstrained model and make the only assumption of having a central camera. This allows us to take advantage of the whole set of tools provided by the projective geometry, while maintaining an higher calibration accuracy with respect to the pinhole based calibration techniques.

4.1 High-Coverage 3D Scanning through Online Structured Light Calibration

Many 3D scanning techniques rely on two or more well calibrated imaging cameras and a structured light source. Within these setups the light source does not need any calibration. In fact the shape of the target surface can be inferred by the cameras geometry alone, while the structured light is only exploited to establish stereo correspondences. Unfortunately, this approach requires each reconstructed point to exhibit an unobstructed line of sight from three independent points of views. This requirement limits the amount of scene points that can be effectively captured with each shot. To overcome this restriction, several systems that combine a single camera with a calibrated projector have been proposed. However, this type of calibration is more complex to be performed and its accuracy is hindered by both the indirect measures involved and the lower precision of projector optics.

In this section we propose a calibration method for structured light sources that computes the projector parameters concurrently with regular scanning shots and does not require to adopt special procedures or targets as, for instance, in [27, 57, 128, 136]. In fact, our approach is an online method that can be performed directly during the normal system usage. Differently from other online methods [242] it is able to automatically recover the scene scale and to deal even with severely distorted lens, increasing both surface coverage and triangulation accuracy.

4.1.1 High-Coverage 3D Scanning

The main idea of our approach is to exploit the 3D points triangulated by two calibrated cameras to get insight about the projector geometry. Basically, this happens by collecting among subsequent shots the coordinates 3D points that reproject exactly over the same projector pixel and then using a simple least square fitting to fix the parameters of the projector ray associated to that pixel. In order to easily perform this step and to cope well with commercial quality projector optics, we adopted the general unconstrained camera model. In [33] the authors already studied a method for effectively calibrating such model and now we are extending it to deal with this new application.

Unconstrained Camera Model

With the term *unconstrained camera model* we mean a completely free imaging model where each pixel (i.e. imaging sensor for cameras or light emitter for projectors) is associated to an independent ray. More formally, the ray associated with camera pixel i can be written as $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{p}_i)$, where $\mathbf{d}_i, \mathbf{p}_i \in \mathbb{R}^3$ represent direction and position of the ray respectively. These vectors satisfy $||\mathbf{d}_i|| = 1$, (normalized direction) and $\mathbf{d}_i^T \mathbf{p}_i = 0$ (orthogonal position vector). Any point $\mathbf{x}$ in the ray $\mathbf{r}_i$ satisfies the parametric equation $\mathbf{x} = \mathbf{d}_i t + \mathbf{p}_i$ for some $t \in R$.

This kind of model comprises literally millions of free parameters, thus it is very hard to calibrate with standard target-based approaches. In [33] we propose a practical method to calibrate it. This is done by taking several shots of a computer monitor showing both a vertical and horizontal phase shift pattern sequence [143]. These is exactly the same type of phase shift coding that is used during the scanning process. The monitor is placed in a total of s position, each one characterized by a pose $\left(\Theta\right)_s = (\left(R\right)_s, \mathbf{t}_s)$ where $\left(R\right)_s$ and $\mathbf{t}_s$ are the rotation and translation of the monitor reference system with respect to the camera world. Once the pattern sequences are decoded, for each pose of the target monitor $\left(\Theta\right)_s$, we are able to assign an observed code $\left(Co\right)$ to each camera pixel i that was inside the reprojection area of the monitor surface. After estimating the poses $\left(\Theta\right)_s$ (see [33] for details), it is also possible to compute the expected code for each pixel $\left(Ce\right)(\mathbf{r}_i| \left(\Theta\right)_s) = \left(P\right)_{\mathbf{uv}}(\mathbf{d}_i t_{\text{int}} + \mathbf{p}_i)$ where $\left(P\right)_{\mathbf{uv}}$ denotes the projection onto the (u, v) planar coordinates of the monitor reference frame, and $t_{\text{int}} = \frac{\mathbf{n}_s^T(\mathbf{t}_s - \mathbf{p}_i)}{\mathbf{n}_s^T \mathbf{d}_i}$ is the intersecting parameter for the equation of ray $\mathbf{r}_i$, i.e., the value such that $\mathbf{d}_i t_{\text{int}} + \mathbf{p}_i$ lies on the monitor plane.

Under these premises, the best estimate for each ray $\mathbf{r}_i = (\mathbf{d}_i, \mathbf{p}_i)$ is the one that minimizes the sum of the squared Mahalanobis lengths $\sum_k \varepsilon \left(\Sigma\right)_k^{-1} \varepsilon^T$ of the residuals $\varepsilon = \left(Ce\right) - \left(Co\right)$, where $\left(\Sigma\right)_i^s$ is the (conditional) error covariance matrix under the given ray-pose combination:

$$(\left(\Sigma\right)_i^s)^{-1} = I + (\cos^2 \phi_i^s - 1)\mathbf{r}^{\|}{}_i^s(\mathbf{r}^{\|}{}_i^s)^T \tag{4.1}$$

where ϕ_i^s is the angle between the target and the monitor. Note that, with the pose parameters $\left(\Theta\right)_s$ at hand, these observed 2D coordinates can be transformed into 3D points in the camera coordinate frame. We can divide the residual $\varepsilon = \left(Ce\right) - \left(Co\right)$ into the orthogonal vectors $\varepsilon^{\|} = \left(Ce\right) - \left(Co\right)^{\|}$ and $\varepsilon^{\perp} = \left(Co\right)^{\|} - \left(Co\right)$, where $\varepsilon^{\|}$ is parallel to $\mathbf{r}^{\|}$. Clearly, since $\varepsilon^{\perp}$ is orthogonal to the plane spanned by $\mathbf{d}$ and $\mathbf{n}$, the point in $\mathbf{r}$ closest to $\left(Co\right)$ is the one closest to $\left(Co\right)^{\|}$. Further, let $\mathbf{h}$ be this point, we have

$$||\mathbf{h} - \left(Co\right)||^2 = ||\mathbf{h} - \left(Co\right)^{\|}||^2 + ||\varepsilon^{\perp}||^2 . \tag{4.2}$$

It is easy to show that, $||\mathbf{h} - \left(Co\right)^{\|}|| = \cos\phi||\varepsilon^{\|}||$, where ϕ is the angle between $\mathbf{d}$ and $\mathbf{n}$. Hence, the squared distance between $\mathbf{r}$ and $\left(Co\right)$ equals

$$d^2(\mathbf{r}, \left(Co\right)) = \cos^2\phi||\varepsilon^{\|}||^2 + ||\varepsilon^{\perp}||^2 = \varepsilon^T \left(\Sigma\right)^{-1} \varepsilon\,, \tag{4.3}$$

thus the generalized least squares formulation with respect to the target coordinates corresponds to the standard linear least squares with respect to the 3D points associated with each ray. The linear least squares problem is then solved by a ray with parametric equation $\bar{\mathbf{x}} + \mathbf{w}t$, where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$ is the barycenter of the observed 3D points, and $\mathbf{w}$ is the eigenvector of their covariance matrix corresponding to the smallest eigenvalue.

Online Projector Calibration

Our goal is to calibrate the rays of the projector as if it was a camera that always recaptures the exact pattern that is projected. This assumption, which is similar to many other projector calibration methods, allows to implicitly know exactly the observed codes (Co). By contrast, the expected code (Ce) must be obtained from the scene. Most approaches found in literature solve this problem by projecting some pattern on a known target and using the features of the target as references. In our case, since we have two calibrated cameras available, we can obtain the 3D reference points by collecting them during the scanning process. The triangulation of each point of the scene happens by finding the same code (computed from the sequence of patterns [143]) on both cameras. This will produce 3D points that are associated to codes observed by camera pixels, that in general do not correspond to the expected code (Co) for any projector ray. For this reason, we cannot directly use the 3D points belonging to the acquired surface, rather we must produce additional points corresponding exactly to the (virtually) observed codes. This can be done easily by interpolating the camera rays whose codes encompass the observed code (Co) with weights inversely proportional to the distance from (Co) of their respective measured codes. To solve the ray interpolation problem for the unconstrained camera model, we generalize bi-linear interpolation to the manifold of 3D lines. Under our parametrization, in fact, a line is represented as a point in $\mathbb{R}^6$. However, the normal direction and orthogonal position constraints force the lines to lay in a 4-dimensional manifold. We can generalize (weighted) means over a manifold through the notion of Fréchet means: a point $\mathbf{x}$ residing in manifold $\mathcal{M}$ is the average of points $\mathbf{x}_i \in \mathcal{M}$ with weights w_i if it solves

$$\underset{\mathbf{x}\in\mathcal{M}}{\operatorname{argmin}} \sum_i w_i d^2_{\mathcal{M}}(\mathbf{x}, \mathbf{x}_i) \tag{4.4}$$

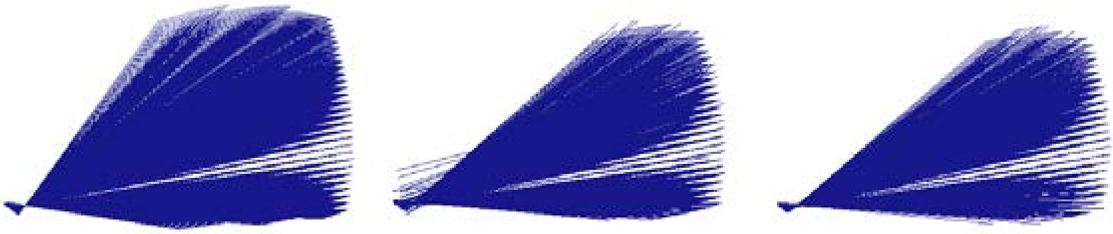


Figure 4.1: The bundles of rays that can be obtained after calibration of the projector using the reconstructed 3D points. In the first image we adopted the pinhole+distortion model. The second and third image show the results obtained using the unconstrained model respectively with and without outlier correction. Note that the pinhole model is able to calibrate all the rays, while the unconstrained model can be populated only by the rays that hit the scanned surface, thus they are a bit less. Also note that all the miscalibrated rays have (apparently) disappeared after outlier removal.

where $d_{\mathcal{M}}$ is the geodesic distance over $\mathcal{M}$. A similar approach has previously been applied to the interpolation of rotations and rigid transformation [124, 199] and, similarly to those approaches, it can be shown that the interpolation is invariant to the frame of reference and interpolates the rays through a minimal path with constant linear and angular velocity. For performance reasons, just like in [124], we approximate the Fréchet mean by taking a linear average in $\mathbb{R}^6$ followed by a projection onto the manifold. These newly obtained virtual rays can finally be used to triangulate the expected point (Ce) corresponding to (Co) and use it to perform the same optimization described in section 4.1.1. Note, however, that equation 4.3 holds only for locally planar surfaces. Generally speaking, this is not guaranteed if the points (Ce) are obtained from the scanning of random objects. Still, since both man-made and natural objects usually exhibits several low frequency areas, it is reasonable to guess that at least a sizeable portion of the obtained expected points will be accurate enough. Moreover, the scanned object will likely move during different shots, enhancing the coverage of the projector frustum and eventually adding redundancy. Finally, even if some (Ce) could suffer from bad estimation, in the next section we suggest an apt method to filter outliers.

Outliers Filtering

After estimating the projector rays, we can assess their quality by means of their fitting residual. Specifically, we can set a badness threshold that can be used to deem as unreliable rays that obtain a bigger residual. Such rays can be removed, in which case they will be simply unavailable for triangulation (note that there is no need to estimate all of the projector rays). Otherwise, it is indeed possible to still recover them by filtering inaccurate (Ce) points that could possibly be the cause of the bad estimation. To do this, we crate a tentative ray candidate by applying equation 4.4 with its available neighbours.

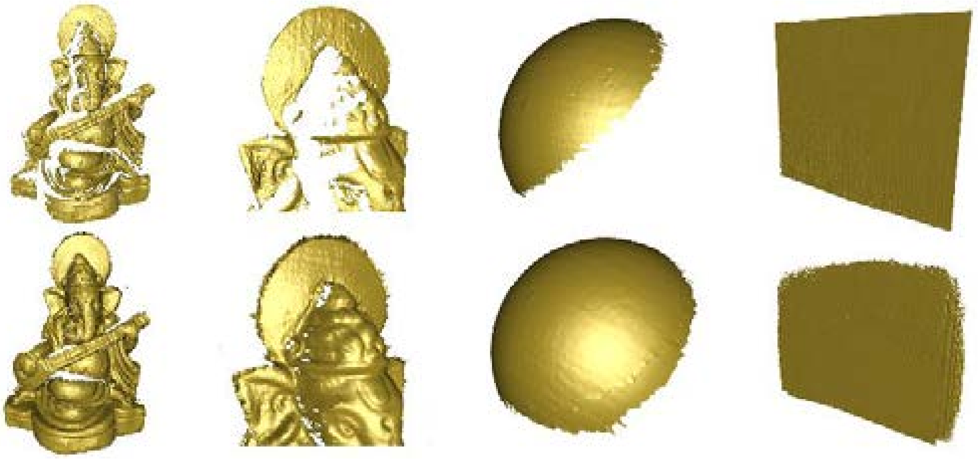


Figure 4.2: Coverage difference between the baseline (top row) and the unconstrained method (bottom row) for some different subjects.

Afterwards, this candidate is used to gather associated (Ce) that are near a given threshold to it, which in turn can be used to obtain a new estimate for the original ray. The rationale of this method is that the interpolation of the neighbours (if any) would result in a putative ray good enough for an effective inlier selection. In Fig. 4.1 we show the bundles of projector rays obtained after calibration. The first image depicts the bundle obtained by calibrating the projector with a standard pinhole model. This happens by using the estimated (Ce) as the 3D points of a virtual calibration objects and the associated projector codes as their reprojections. While this could seem a good approximation, we will show in the experimental section that the pinhole model is not able to fully deal with the imperfection of commercial quality lenses (as also observed in [33]). The other two show the bundles obtained using the described unconstrained model respectively before and after outlier filtering and ray correction.

4.1.2 Experimental Evaluation

In order to evaluate the proposed method we built an experimental setup similar to many off-the-shelf structured light 3D scanners. We accurately calibrated the two 1280x1024 pixels cameras for both intrinsic and extrinsic parameters according to both the pinhole model and to the unconstrained model. The projector used is an SVGA Dlp micro projector. We implemented three reconstruction models:

- *Baseline*: the unconstrained stereo camera reconstruction model that works without needing projector calibration presented in [33]. We expect this to be the most accurate but to exhibit less coverage;
- *Pinhole*: a reconstruction configuration that uses the projector calibrated according to the pinhole model (including distortion) to enhance coverage. We expect this to be less accurate due to the limitation of the pinhole model, especially for the cheap optics of commercial-quality projectors;
- *Unconstrained*: the reconstruction model using the unconstrained projector calibrated with the approach proposed.

We tested these models by scanning three different objects: a small figurine of Ganesha, which exhibits small details and thus many high frequency areas, a regular sphere, which is rather smooth and includes only low frequencies, and finally a flat plane, used as a featureless reference object. For each object we acquired about 100 scans covering almost all the projector frustum, and we compared the results obtained by calibrating the projector with different amounts of randomly selected shots subsets. We adopted four different evaluation criteria that are described in the following subsections.

Enhanced Coverage

With this test we measure the ratio between the area of the surface obtained with the calibrated projector methods and with baseline. This metric represents the enhancement in terms of coverage. In Fig. 4.4 we plot the coverage increment with respect to the number of shots used to calibrate (which correspond to the total number of scans, since the method is online). Of course we show only the curves for the Sphere and Ganesha objects, since there is no increment for the plane (which is equally well seen by both cameras). Note that the latter object obtains a larger advantage since it contains many convoluted areas that are hard to capture at the same time by two cameras and the projector. Note also that the pinhole model reaches immediately the maximum increment while the unconstrained model requires from 15 to 30 shots to perform equally well. This is expected since in this case the calibration includes all the rays from the start. However, we will see in the next test that this advantage comes at the cost of a lower accuracy. Some qualitative examples of the coverage are shown in Fig. 4.2. Here the scattered edges of the plane are due to the fact that not all the projector rays have been recovered. This happens simply because the rays on the periphery of the frustum appears in fewer scans of the subject, which is expected. If a full coverage of the projection must be guaranteed, this can be obtained offline using a bigger planar object encompassing the whole frustum.

Reconstruction accuracy

To give a reasonable accuracy measure, we decided to adopt the baseline method as a reference. This is a reasonable choice since we already discussed in [33] the accuracy of a camera pair calibrated with the unconstrained model. In this regard, we express the accuracy as the RMS error, after ICP registration [37], between the acquired surface and the ”ground truth” offered by the baseline. Note that such RMS is expressed in world unit, which, since the cameras have been calibrated with a computer monitor, corresponds to the size of a pixel on that specific screen (approximately 0.2 mm). In Fig. 4.5 we show the obtained accuracy after different amounts of scans. The pinhole method requires few shots to reach its maximum accuracy. However it always performs worse that the unconstrained method. Furthermore the standard deviation of the pinhole curve is narrower. These

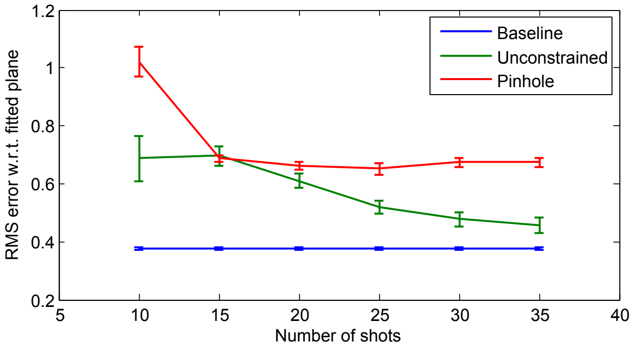


Figure 4.3: Coherence of the reconstructed surface with a planar reference target.

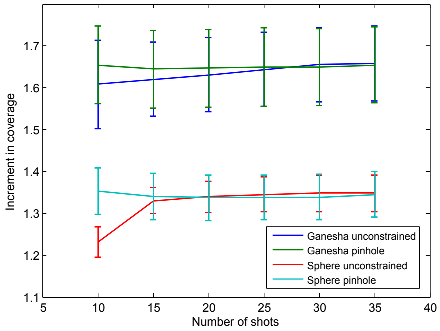


Figure 4.4: Increment in the coverage with respect to the number of scans.

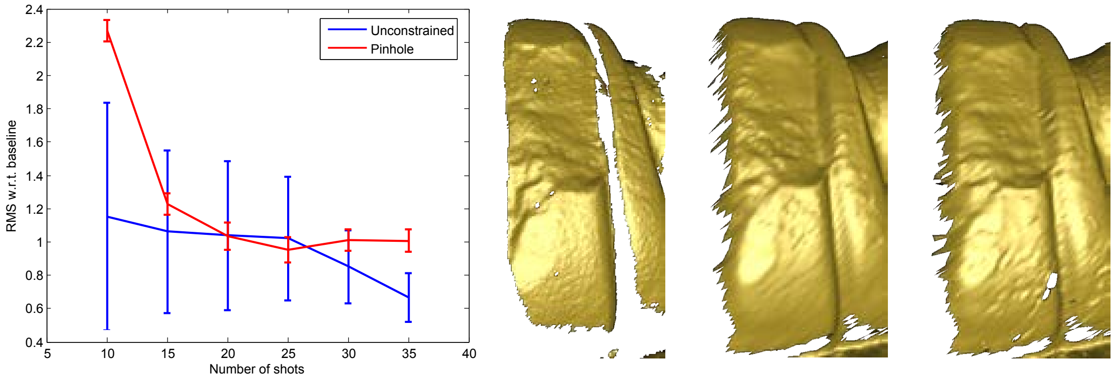


Figure 4.5: Accuracy of the reconstruction with respect to the baseline method. The close-ups on the left part of the figure show a detail of the reconstruction obtained respectively with the baseline, unconstrained and pinhole methods.

phenomena can be explained respectively by the fact that the pinhole model is not able to fully handle the imperfections of a real lens and that its statistical nature makes it very stable with respect to the set of shots selected for calibration. The unconstrained method, albeit after several shots, allows for a significantly better accuracy.

Surface Repeatability

While the accuracy measures the compliance of the results with respect to the ground truth, we are also interested in the repeatability of the reconstruction within the same method. To evaluate this metric we took several scans of the same subject with slightly different poses and we computed the average RMS error, after ICP registration, between the surfaced acquired using the same method. Basically, this measure gives us an insight about the resilience of the method to random noise and to aliasing error generated by the interplay between camera and projector rays. In Fig. 4.6 we plot such measure for the baseline method (which appears as a horizontal line since it does not depends on the number of scans) and of the other two methods. We can conclude that all the tested approaches exhibit a good repeatability, in the order of hundredths of a millimetre. This repeatability appears to be not strongly sensitive to the number of scan used to calibrate, with the possible exception of the pinhole method that performs less well with few shots.

Planarity

Finally, we measured the planarity of the reconstruction of a reference plane made by coating with matte paint a float glass. This is done by computing the average RMS error with respect to a a general plane that has been fitted to the data. The rationale of this measure is to assess spatial distortions that usually characterizes imperfect calibrations. The results obtained are shown in Fig. 4.3. We can observe that the pinhole method produces the surface with larger distortion. This is certainly attributable to the inherently

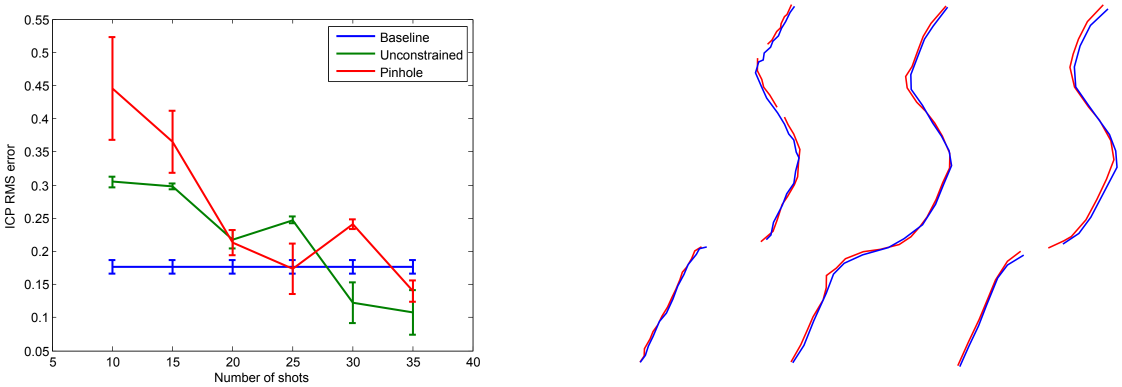


Figure 4.6: Repeatability of the reconstruction for different scans of the same subject. On the right part of the figure we show some slices from the acquired meshes to illustrate the alignment between subsequent scans respectively with the baseline, pinhole and unconstrained methods.

imperfect correction of the distortion.

4.2 Adopting an Unconstrained Ray Model in Light-field Cameras for 3D Shape Reconstruction

Light-field cameras have been recently popularized thanks to the recent introduction of commercial devices such as Lytro and Raytrix models. Such interest in light-field imaging devices is well justified by its numerous applications, ranging from depth map estimation [38,56,99] to super resolution [39,231], refocusing after shooting [165] and creation of virtual points of view [138,139].

One of the main hurdles in plenoptic photography derives from the composite imaging formation process which limits the ability to exploit the well consolidated stack of calibration methods that are available for traditional cameras. All the existing plenoptic camera calibration methods in literature adopt a multi-pinhole model [40,65,78,78,221]. In this section we analyze the use of a calibration method that escapes the need to adopt a parametric model by exploiting dense correspondences generated using phase coding technique [33]. While dense calibration has been already used in literature, this is the first time that it is attempted with light-field cameras and its correct behavior is not guaranteed. We further propose a generic triangulation technique that is suitable for the unconstrained calibrations as well as in the model based ones.

4.2.1 Light-Field Calibration

All the calibration methods for a light-field camera must deal with a common hurdle: Each micro-lens, being pinhole, could be calibrated independently using standard target-based methods, ranging from the basic approach proposed by Tsai [218], to more advance model that could account for the high distortion that micro-lenses usually exhibit [80,117]. However, these approaches would incur in the disadvantage that the recovery of the target pose could be very ill-posed when performed on the basis of a single micro-lens image. In fact, each micro-lens only counts few pixels spanning a large view angle, resulting in poor angular resolution of each micro-lens in isolation. Furthermore, most approaches adopt

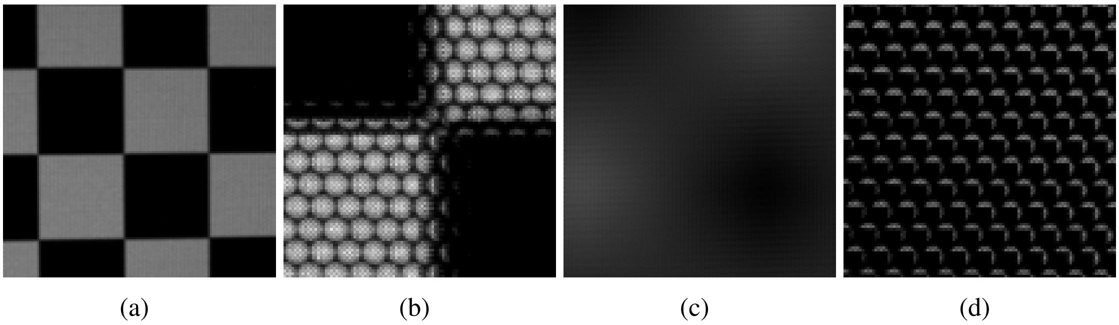

(a) (b) (c) (d)

Figure 4.7: A chessboard pattern captured by a Lytro lightfield camera respectively at minimum and maximum zoom level.

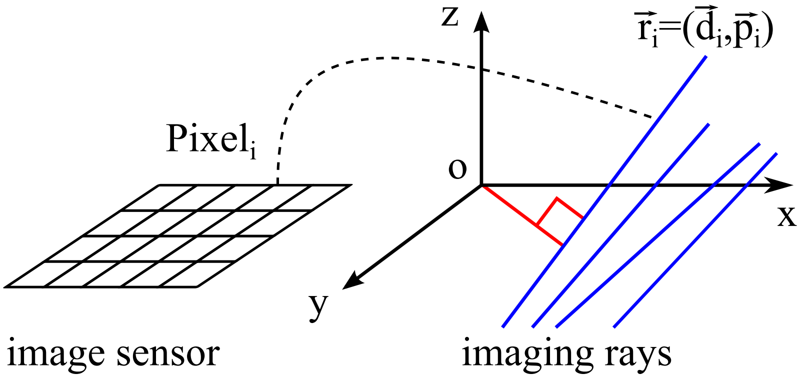


Figure 4.8: The non-parametric imaging model adopted (see text for details).

image space techniques to localize target features, thus the limited span of each microlens image would severely reduce the number of recoverable correspondences and their accuracy. This problem can be better understood by looking at Figure 4.7, where the images of a chessboard acquired by a commercial Lytro lightfield camera are shown. In Subfigure 4.7(a) and 4.7(b) we show an overall frame and a detail of the image created on the CCD when the zoom of the main lens is set to the minimum value. The overall frame (about 3000 pixels wide) appears similar to what would be obtained using a standard projective camera. However, by looking at the detail of each microlens (about 10 pixels wide), it can be observed that most of them capture a fully black or fully white field and just a few see a chessboard corner. Under these conditions, classical calibration methods that need to relate target points to observed features are useless, hence the need for specialized technique. The behavior is even more extreme within Subfigure 4.7(d) and 4.7(c), where we display images of the same chessboard obtained with the maximum zoom level of the main lens. Here, the overall frame lost any connection to a projective transform and the images produced by the microlens are so wide that they extend beyond the span of a single check. The solution we are proposing is to refrain to use any global parametric model and to independently assess the characterization of every single imaging ray that insists on the camera sensor. To this end, we apply a dense target localization method which works in the time rather than the space domain, thus escaping the aforementioned hindrances. Such dense correspondences, in turn, enable the adoption of a parameter-free optimization for non-central cameras.

The Parameter-Free Camera Model

Following [33], we adopt a non-parametric camera model where each ray is modeled as an independent line within a common reference frame. Such reference frame is not directly related to the physical sensor. In fact, according to this model, image coordinates can be considered just labels for the imaging rays, which are not related to them by means of any analytic function (see Figure 4.8). More formally, index $i \in \{1..n\}$ ranges over all the n pixels of the camera sensor (in no particular order). The ray associated to pixel i

can be written as $\mathbf{r_i} = (\mathbf{d_i}, \mathbf{p_i})$, where $\mathbf{d_i}, \mathbf{p_i} \in \mathbb{R}$ represent direction and position of the ray respectively. These vectors satisfy $||\mathbf{d_i}|| = 1$ (normalized direction) and $\mathbf{d_i}^T\mathbf{p_i} = 0$ (orthogonal position vector). Any point $\mathbf{x}$ in the ray $\mathbf{r_i}$ satisfies the parametric equation $\mathbf{x} = \mathbf{d_i}t + \mathbf{pi}$ for some $t \in \mathbb{R}$. Lacking any explicit structural relation between rays, this model counts 4 degrees of freedom for each pixel, resulting in several millions unknowns.

A solution space this large needs an exceptional number of observations, and this can only be obtained using a dense coding strategy, which assigns to each image pixel (i.e. to each ray) a pair of coordinates on the calibration target. There are several ways to do this, we follow [33] adopting a flat monitor as the target [100, 207] and using a multi period phase shift coding [143] in order to obtain dense target coordinates on a Lytro camera sensor. The coding has been performed both horizontally and vertically. In Figure 4.9 an example of such dense coding can be seen. We used a scale of red and green values to show the recovered horizontal and vertical coordinates of the monitor overlaid to a white light imaging of the scene (resulting in a color blending). The code appears to be smooth, but of course this is an effect due to detail level of the figure, in practice the image presents the same repetition effects that can be seen in Figure 4.7 and that will be discussed in detail in the next section. The dense correspondences acquired over several poses of the target can be used to feed the iterative optimization method presented in [33] obtaining the characterization of each ray that has been correctly codified within a large enough number of different poses. Such method, however, has been designed to work on quasi-pinhole cameras and there is no guarantee that it works with a plenoptic camera. Neither it is obvious that the dense coding would work well with the considered imaging process, especially for the higher camera zoom levels shown in Subfigures 4.7(d) and 4.7(c).

In the next two sections we will study in detail the performance of the dense coding and of independent rays calibration when applied to a lightfield camera.

Dense Target Acquisition

All our experiments have been performed using a first generation Lytro plenoptic camera, equipped with a 3280x3280 pixels sensor. Throughout all the tests we used two zoom

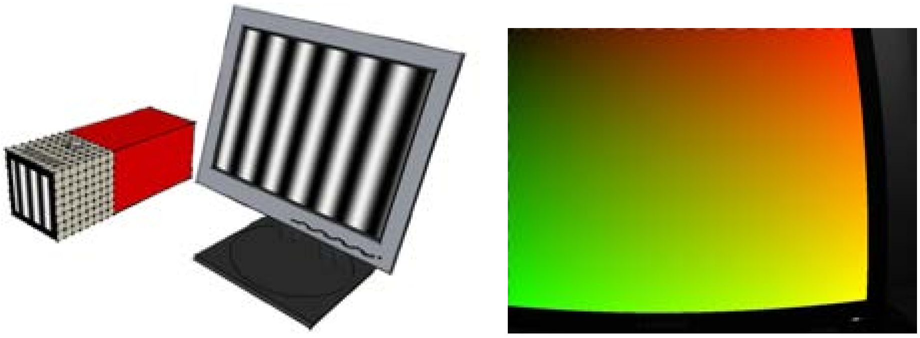


Figure 4.9: Dense calibration requires the user to acquire a sequence of horizontal and vertical patterns from a standard monitor.

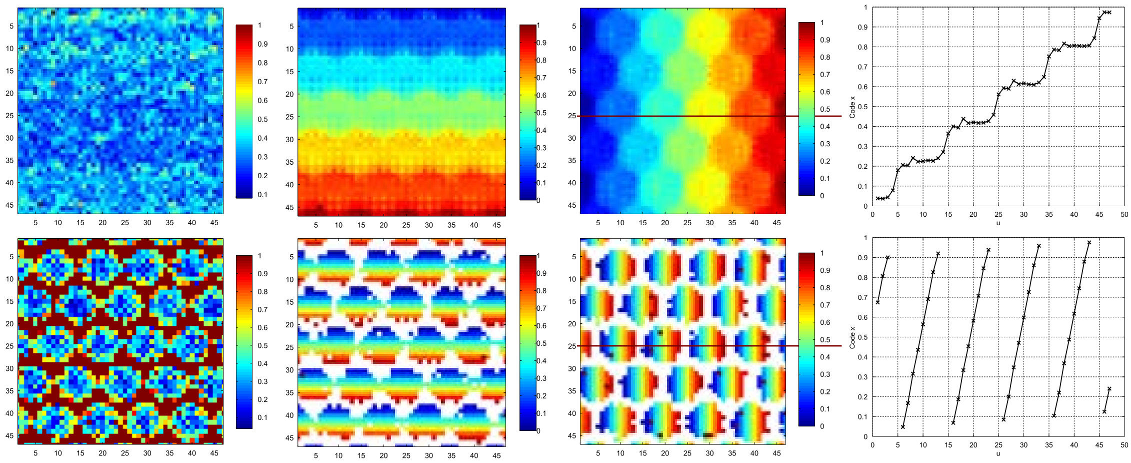


Figure 4.10: Behavior of the dense coding method applied to a Lytro lightfield camera at different zoom levels (see text for details).

levels for the main lens, which we will refer to as *minzoom*, corresponding to the 1.0 level on the camera, and *maxzoom* corresponding to the 5.5 level (the reader can refer to Figure 4.7 to get an idea about what results can be obtained at such zoom levels).

Horizontal and vertical coding of the target has been performed using multi-period phase shift on a 21 inches monitor place at about 1 meter from the camera, with period lengths respectively 11, 13 and 17 monitor pixels and capturing 15 samples for each period (see [143] for details).

In Figure 4.10 we show the acquired codes within a small portion of the imaging sensor measuring about 50x50 pixels respectively for minzoon (upper row) and maxzoom (lower row) (note that the code has been normalized from 0 to 1 for better visualization). The first column shows the coding error according to [143], a lower value means a more accurate target position recovery, a value of 1 means that no recovery has been possible for that particular pixel. At minzoom level the coding error is low and isotropic, this is due to the low distortion of the microlens images and to the overall quasi-pinhole behavior of the camera. Note that, although it may be counter-intuitive, at minzoom even the usually disregarded space *between* microlens is correctly coded, thus recovering the intersection between the associated ray and the target.

This allows to effectively calibrate the full sensor, including inter-lens pixels. In the next section we will show that the resulting calibration exhibits a good accuracy. This is in strong contrast with standard parametric light-field camera calibration methods, which, even when capable of capturing features between microlens, would still be unable describe the behavior of those rays with their models.

Conversely, at maxzoom the coding error is lower around the centers of the microlens and increases a bit when moving toward the edges. Note that outside the microlenses, the code is not recovered. This is not due to measuring errors, but rather to the low signals that reaches the sensor as a result of the strong vignetting (which can be observer also on Subimage 4.7(c)). In the two central columns of Figure 4.10 we show respectively the recovered horizontal and vertical codes. Both are dense and quite smooth, however the maxzoom setting clearly result in micro images that cover a wider area of the target and include several repetitions of the same code. This property, obviously, is very useful if the data is to be used for surface reconstruction, since multiple and well separated observations of the same point will result (in principle) in a more accurate triangulation. The code distribution is even more apparent in the last column of the figure, where a slice of the coding has been plotted (corresponding to the black horizontal line on the third column of the figure).

The different distribution of codes can be understood very well also by looking at Figure 4.11. In the first row we overlay to a coded image obtained at minzoom the pixels with a code less than one code unit far from a given pair of coordinates, showing two level of details. In the second row we plot the same information over a coded image obtained at maxzoom setting. It can be seen that a minimum zoom level a point can be observed at most by one or two microlenses, while at the maximum level the same code is repeated over ad over throughout a large number of different microlenses spanning a (relatively) large portion of the imaging sensor.

To this end we can affirm that, while lower zoom levels can be used for tasks such as refocusing of subaperture images creation, higher zoom levels are best suited for surface reconstruction tasks, where the larger disparity of the captured light rays results in a more accurate and robust triangulation. Specifically, in such cases accuracy would be granted by the wider angle between rays used for triangulation and robustness can be achieved out of their large number. In Sections 4.2.2 and 4.2.2 we will substantiate these statements.

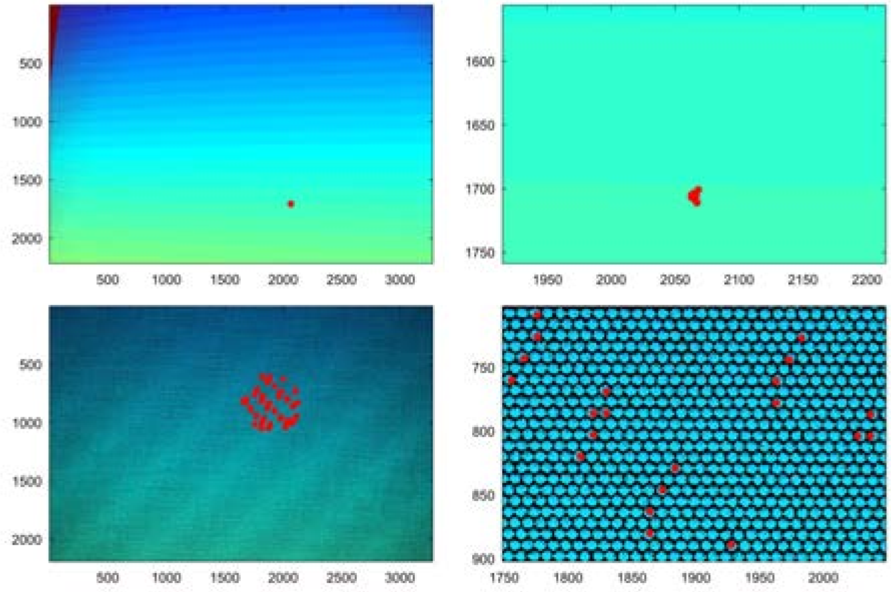


Figure 4.11: Distribution of points with similar code over the imaging plane.

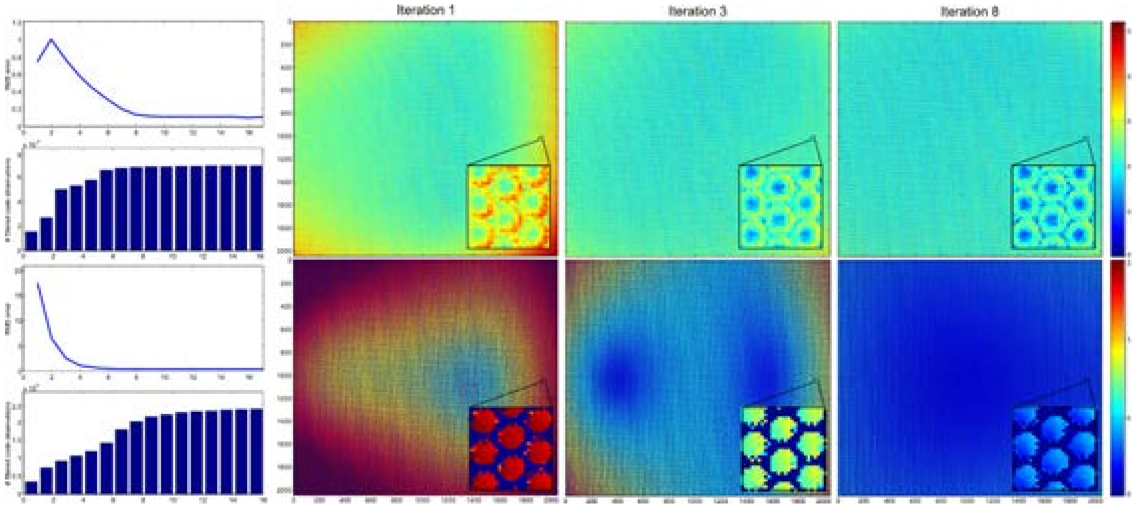


Figure 4.12: Evolution of the RMS error of rays during the optimization and number of culled outliers (see text for details).

Rays Optimization

In order to apply the ray optimization procedure presented in [33] we first need an initial estimation for the poses of the target and for the rays. We solved this problem by creating a quite raw sub-aperture image obtained by grouping all the microlens centers. With a very coarse approximation, such image can be considered as if it were produced by a pinhole camera, to this end we can get an initial estimate using the standard calibration procedure made available by OpenCV [43]. The optimization procedure has been performed using a total of 10 target poses. Since the non-parametric model does not provide an image plane, the ray optimization does not proceed by minimizing some reprojection error, as its common with standard calibration methods. Instead, [33] minimizes, with respect to ray and poses parameters, the sum of the squared distances between the 3D coordinates of a target point and the ray that observed it. Since our code advances of one unit for each monitor pixel, the RMS error is a metric measure in the Euclidean spaces expressed in units of pixels, that in our case, corresponds to about 0.25mm.

In Figure 4.12 we show the trend of this RMS error with respect to subsequent steps of the optimization process. As for the previous section, the first row represents the minzoom and the second the maxzoom setup (note that both for the plots and for the color-coded images the scales are different). In both cases the RMS error converges after just a few iterations to a final value that is well below one monitor pixel, representing in practice just a few hundredths of millimeter. Such accuracy should be put in context, considering that during the calibration procedure the target is placed about one meter far away from the camera. The colored plots express the RMS error associated to each pixel of the camera sensors. While with our model pixels are just indexes, it is still interesting to see how the error is distributed on the sensor, both at a global and microlens scale. Specifically, we can observe that both zoom levels start with an anisotropic error distribution (probably due to the incorrect handling of the global distortion)and rapidly reach a lower and better distributed RMS error. This is a well-known feature of parameter-free models, which are very well capable at accommodating both local and global distortions.

Finally, the histograms of Figure 4.12 report the total number of ray outliers that have been culled after each iteration. Such culling happens when a ray received a valid code, but it cannot be correctly justified using the estimated poses, that is when it cannot pass close enough to the target codes it has observed. This can happen for a variety of reasons, including occlusions (especially on the border of the monitor) or wrong code recovery due to measurement errors. Note that, even at convergence, the removed rays are in a magnitude order between 10^4 and 10^5, since the overall number of pixel is in the order of 10^7 we can conclude that more than 99% of sensor elements are correctly calibrated.

4.2.2 3D Shape reconstruction

Generally speaking, the 3D position of an observed point can be recovered by triangulation [106] if it is observed by different points of view by means of an imaging process of known geometry. To this end, three sub-problems must be addressed:

1. the point must be identified for each point of view;
2. all the viewing direction must be recovered;
3. an intersection between them must be computed;

Sub-problem 1 can be solved in many different ways, ranging from image-based correspondences to structured light coding. Since the goal of this work is not to introduce a matching method, and we are interested in factoring out most error sources that are not related to calibration. To this end, we solve the point identification problem using the same phase shift coding described in the previous section, which we have shown to be feasible and robust. On the other hand, with respect to subproblems 2 and 3 we introduce two task specific solutions.

Rays interpolation

One reason that makes constrained camera models such as [40] effective in practice is that exists a continuous mapping between any point (u, v) in the image plane and the corresponding ray exiting the camera. Consequently, 3D point triangulation can be solved by searching multiple occurrences of the same feature among the micro-lenses and intersecting the corresponding rays originating from the feature coordinates. In the case of phase-shift structured light coding, the set of projected codes is known but is extremely unlikely that the camera probing rays would sample exactly such codes. However, under the assumption of locally planar 3D surface, each feature location (u, v) can be recovered by interpolating the observed codes in the image plane.

Conversely, if we model our camera as a generic sparse bundle of unconstrained probing rays, there is no trivial way to recover the ray $\mathbf{r}_\ell$ exiting the imaging device at any (possibly sub-pixel) image point p. Further, there is not even a concept of image plane but just some existing calibrated rays in space each one sampling an intensity value or, if we use a structured-light system, a two-dimensional phase code. In other words, the interpolation cannot be performed on the image plane but on a set of already known rays whose contribution in the estimation of $\mathbf{r}_\ell$ depends on what those rays are observing.

In the following Section we give a solution to the rays interpolation problem. Then, in Section 4.2.2 we describe in detail our proposed triangulation process for light-field cameras.

Rays manifold interpolation function Let $R_d = \{\mathbf{r}_i\}$ a set of n known camera rays, and $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{R}^n$, $\sum_{i=1}^{n} w_i = 1$ a convex combination of weights.

We pose the ray interpolation problem in terms of rigid motions blending. Let $K \in SE(3), K(\mathbf{r}_a) = \mathbf{r}_b$ be a the rigid motion that transforms a ray $\mathbf{r}_a$ into $\mathbf{r}_b$. A famous result by Chasles [55] states that any rigid transformation is in fact a screw motion, i.e., a rotation around an axis placed anywhere in the 3D space, and a translation along the direction of the axis. when applied to rays $\mathbf{r}_a$ and $\mathbf{r}_b$, the screw motion of all the points under a pure translation is limited by the length of the translation, while the motion induced by the rotation in unbounded. For this reason, we chose the rigid motion aligning $\mathbf{r}_a$ to $\mathbf{r}_b$ of minimal rotation as interpolant K_{ab} of the two rays. It is straightforward to see that the best possible rotation angle is the one between the two vectors $\mathbf{d}_a$ and $\mathbf{d}_b$ (i.e. $\text{acos}(\mathbf{d}_a^T \mathbf{d}_b)$) that rotates the first ray around the axis given by $\mathbf{d}_a \times \mathbf{d}_b$. When the rotation angle and axis is chosen, the optimal translation is the one moving $\mathbf{r}_a$ according to a vector T orthogonal to $\mathbf{d}'_a = \mathbf{d}_b$ whose length is equal of the distance between the two rays. In other terms, the best translation is the vector that connects the two nearest points $\mathbf{s}_1$ and $\mathbf{s}_2$ lying on $\mathbf{r}_1$ and $\mathbf{r}_2$ respectively. To summarize, given two rays $\mathbf{r}_a$ and $\mathbf{r}_b$, we choose the interpolant K_{ab} as:

1. The rotation R_K around the axis $\mathbf{d}_a \times \mathbf{d}_b$ with angle $\text{acos}(\mathbf{d}_a^T \mathbf{d}_b)$
2. The translation $T_K = \mathbf{s}_2 - \mathbf{s}_1$

Given a set of interpolants mapping rays to rays, the problem of ray interpolation can be cast as one of averaging in the manifold of rigid transformations $SE(3)$. This is the path taken by *Dual-quaternion Iterative Blending* (DIB) that interpolates roto-translations in terms of a screw motion represented in terms of dual quaternion [124] and can be interpreted as computing a manifold averaging in SE(3) endowed with the screw motion metric. More formally, DIB takes a set of rigid motions K_i with $i = 1, \ldots, n$, and a set of weights w_i and finds the unique motion K^* that satisfies

$$\sum_{i=1}^{n} w_i \log\left(K_i K^{*-1}\right) = 0\,, \tag{4.5}$$

where $\log$ is the logarithm map of the group $SE(3)$. This interpolation approach exhibits many useful properties such being *constant speed*, *shortest path* and *coordinate system* *independent*. Adopting this approach, given a set of rays $\{\mathbf{r}_1, \dots, \mathbf{r}_n\}$, we initialize the interpolated ray $\mathbf{r}_\ell = (\mathbf{d}_\ell, \mathbf{p}_\ell)$ as their weighted linear combination followed by a reprojection on the rays manifold:

$$\mathbf{d}_\ell = \frac{\sum_{i=1}^n w_i \mathbf{d}_i}{||\sum_{i=1}^n w_i \mathbf{d}_i||} \tag{4.6}$$

$$\mathbf{p}_\ell = \frac{\sum_{i=1}^n w_i \mathbf{p}_i}{||\sum_{i=1}^n w_i \mathbf{d}_i||} - \mathbf{d}_\ell \left(\mathbf{d}_\ell^T \frac{\sum_{i=1}^n w_i \mathbf{p}_i}{||\sum_{i=1}^n w_i \mathbf{d}_i||} \right) \tag{4.7}$$

Then, we compute the rigid transformations $K_{\ell,i}$ as the screw motion between $\mathbf{r}_\ell$ and each $\mathbf{r}_i$ according to the procedure stated before. Once computed, all the $K_{\ell,i}$ are averaged via DIB with the weights $\mathbf{w}$ to obtain K_{avg}. Finally, K_{avg} is applied to $\mathbf{r}_\ell$ to obtain a better estimate, and the procedure is repeated until convergence.

Interpolation weights estimation In the structured light case, we base the weights estimation for the set of rays R_d in terms of the codes $c_1 \dots c_n \in \mathbb{R}^2$ observed by each $\mathbf{r}_1 \dots \mathbf{r}_n$.

In this work, we cast the weight estimation as a regularized barycentric interpolation. Following [197] we adopted inverse squared distance weight, but add a regularization factor λ. Specifically, let $\mathbf{D}$ be the $n \times n$ diagonal matrix whose diagonal entries $d_{ii}, i = 1 \dots n$ are the squared distances between each observed code c_i and co. Then, the weight vector $\mathbf{w} = \left(w_1 \dots w_n\right)^T$ can be estimated as:

$$\begin{array}{ll} \min\limits_{\mathbf{w}} & \frac{1}{2}\mathbf{w}^T(\mathbf{D} + \lambda\mathbf{I})^{-1}\mathbf{w} \\ \text{subject to} & \mathbf{C}\mathbf{w} = co, \\ & \mathbf{1}^T\mathbf{w} = 1 \end{array} \tag{4.8}$$

where $\mathbf{C} = \begin{pmatrix} | & & | \\ c_1 & \dots & c_n \\ | & & | \end{pmatrix}$.

Problem 4.8 can be solved as a generalized least squared problem, yielding:

$$\mathbf{w} = \mathbf{A}\mathbf{C}^* \left(\mathbf{C}^* \mathbf{A} \mathbf{C}^{*T}\right)^{-1} co^* \tag{4.9}$$

where $\mathbf{A} = (\mathbf{D} + \lambda\mathbf{I})^{-1}$, $\mathbf{C}^* = \begin{pmatrix} \mathbf{C} \\ \mathbf{1}^T \end{pmatrix}$ and $co^* = \begin{pmatrix} co \\ 1 \end{pmatrix}$.

Ray Selection and Triangulation

Since we are dealing with light-field cameras, we expect each known projected code co being visible in many different micro-lenses (see Fig. 4.11). Therefore, we start by searching in the acquired coded image all the pixel locations $\mathbf{u}_1 \dots \mathbf{u}_m$ whose codes $c(\mathbf{u}_1) \dots c(\mathbf{u}_m)$

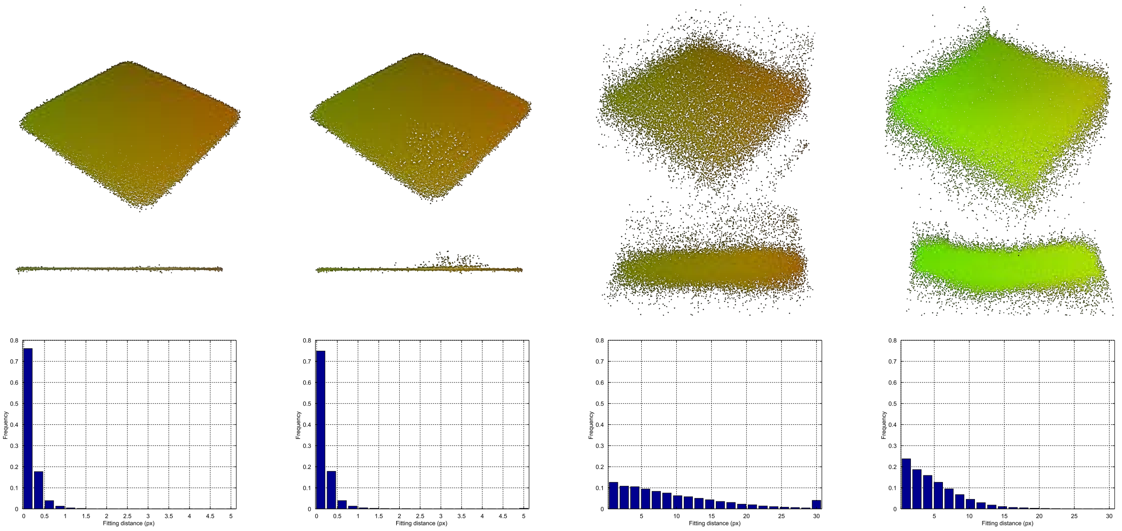


Figure 4.13: 3D reconstructions of the target plane using respectively our full method, disabling the RANSAC selection, disabling the interpolation and, finally, using the parametric calibration obtained with [40].

are closer than a threshold dst_c to the projector code co. Due to their pinhole nature, we expect each $\mathbf{u}_i$ to lie in a different micro-lens.

At this point, for each $\mathbf{u}_i$ our aim is to create a new ray $\mathbf{r}_{ui}$ that would have observed the code co from the same micro-lens where $\mathbf{u}_i$ lies. Therefore, we look at the adjacent 8 neighbor pixels for valid codes and rays to be used as interpolation data. Then the ray is interpolated as explained above.

Once all the virtual rays $R_t = \{\mathbf{r}_{u1} \dots \mathbf{r}_{un}\}$ are collected, a robust least-square estimation is used to triangulate the 3D point P_{co} associated to the projected code co. Specifically, we adopt a RANSAC scheme to select a subset $R_{ti} \subseteq R_t$ producing a point P_{co} whose squared distance between each ray in R_{ti} is less than a threshold dst_s.

Quantitative Analysis and Comparisons

To assess the accuracy of the 3D surface reconstruction (and hence of the calibration) we need to define some proper error measure with respect to a ground truth. Here we opt to use the same monitor that has been used for the calibration by creating a set of additional shots and by triangulating its surface. This approach has the advantage of producing a planar surface that can be easily fitted allowing to compute the displacement error between the plane model and each triangulated point.

We tested a total of four setups at the maximum zoom level (which is best suited for reconstruction for the reasons described in previous sections). The first one adopted the full method described in this Section. Subsequently, we disabled respectively the RANSAC selection and the ray interpolation, to study their role in the overall accuracy.

Finally, we adopted the calibration method proposed in [40], which has been introduced very recently and can be considered among the state-of-the art for parametric light-field calibration. This latter setup has been obtained using the software made available by the authors to calibrate the camera and, subsequently, by producing single rays r_i according to the imaging model presented in the original paper. This way the obtained rays can be used directly within our pipeline. Note also that we applied both RANSAC and interpolation to these rays, in order to make the results comparable with the best results obtained with our method.

In Figure 4.13 we show the surfaces obtained from different points of view. We also show the distribution of the fitting errors for each method (note that the horizontal scales are different). The first two columns show the results obtained with our method with and without the RANSAC selection. The error distribution is quite similar, in fact just a few misplaced points appear when the consensus between ray is not enforced. This is somewhat expected since outlier culling has already been performed during calibration and we think that these inaccurate points result from coding errors that happened during the reconstruction shot. Conversely, ray interpolation is key to an accurate reconstruction. In fact, since a microlens covers a wide angle, we expect to observe quite large jumps in the codes between pixels (and rays), thus making the observation of exactly the same code (or a code near enough) from the different camera pixels very unlikely.

Finally, the triangulation obtained using the rays calibrated with [40] has a performance that is very similar to the one obtained without interpolation (even if, in this case, we are indeed interpolating the rays before triangulation). Moreover, a bit of global bending can be seen, probably due to a non perfect compensation of the distortion of the main lens. It should be noted that, while [40] is not really meant for 3D reconstruction, but only for image synthesis, it is still one of the most relevant and recent calibration methods against which to compare.

4.3 Non-Parametric Lens Distortion Estimation for Central Cameras

The inability of describing the imaging model by simple linear projective operators severely limits the use of standard algorithms with unconstrained models. On the other hand, unconstrained non-parametric models, despite being originally proposed to handle specialty cameras, have recently been shown to outperform the pinhole model, even with the simpler setups.

In this section we propose a parameter-free camera model where each imaging ray is constrained to a common optical center, forcing the camera to be central. Such model can be easily calibrated with a practical procedure which provides a convenient undistortion map that can be used to obtain a virtual pinhole camera. We apply our method also to the calibration of a stereo rig with a displacement map that simultaneously provides stereo rectification and corrects lens distortion.

4.3.1 Unconstrained Distortion Calibration

To estimate a dense non-parametric lens distortion we start by recovering the 3D light rays associated to each image pixel. Specifically, we formalize the light path entering the lens and hitting the sensor at discrete pixel coordinate (u, v) with the straight line $r_{(u,v)} = (o, d_{(u,v)})$ passing trough the point o and oriented along the direction $d_{(u,v)}$. The common point o constrains our model to be central while no restriction is enforced on the directions $d_{(u,v)}$. Also, the uniform-spaced grid of the CCD provides an ordering on the rays spatial topology.

Since the model implies 3 degrees of freedom for each pixel, plus additional 3 for the optical center o, standard point-to-point calibration approaches like [68, 218, 251] cannot provide enough data to recover a valid solution. We solve this by adopting the dense structured-light target proposed in [33]. Specifically, we use an high-resolution LCD screen displaying phase shift patterns to uniquely localize each target point seen by the camera.

This has a strong advantage with respect to common calibration targets composed by a discrete set of features: unlike methods based on corner or ellipse localization, we can reason in terms of discrete camera coordinates. Indeed, for each camera pixel (u, v) a precise sub-pixel localization of the 2D target-space coordinate of the observed point can be recovered from the phase unwrapping process. In addition, the coding theory approach is robust against possible visual artifacts that may appear on target surface like illumination gradients, specular highlights and shadows. Finally, we take advantage of the precise manufacturing of modern LCD displays to get an affordable target far more accurate that the ones created through ink printing process.

To estimate the optical center and the direction of each ray, the calibration target is exposed to the camera in different positions and orientations. We denote with $\mathbf{RT}_s$ the 3×4 matrix describing the roto-translation of the target with respect to the camera in the

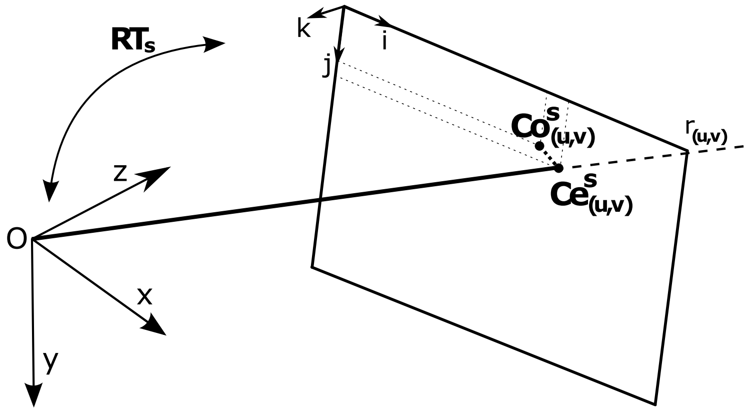


Figure 4.14: Schema of the optimized camera model involving the optical center o, the ray direction, the observed and expected code and a target pose.

pose s, assuming the target reference frame located in the upper left monitor pixel with $\mathbf{i}, \mathbf{j}, \mathbf{k}$ versors being respectively the monitor columns, rows and normal. For each pose, let $\mathbf{Co}^s_{(u,v)} \in \mathbb{R}^2$ be the code observed by the ray $r_{(u,v)}$ in shot s.

Since the LCD geometry and the displayed codes are known, we can take advantage of a pose RT_s to map each code to a 3D point in the camera space, viceversa. For instance, the intersection between a ray $r_{(u,v)}$ and the target plane defined by a pose $\mathbf{RT}_s$ yields to the expected code $\mathrm{Ce}(r_{(u,v)}, \mathbf{RT}_s) \in \mathbb{R}^2$ that the ray should have observed (Fig. 4.14).

Single Camera calibration

Following [33], we recover the geometry of the rays entering the camera as the generalized least-squares minimization problem:

$$\underset{\mathbf{r}_{(\mathbf{u},\mathbf{v})}, \mathbf{RT}_s}{\operatorname{argmin}} \sum_{u,v,s} (\varepsilon^s_{(u,v)})^T (\Sigma^s_{(u,v)})^{-1} \varepsilon^s_{(u,v)} \tag{4.10}$$

where $\varepsilon^s_{(u,v)} = \mathbf{Co}^s_{(u,v)} - \mathrm{Ce}(r_{(u,v)}, \mathbf{RT}_s)$ are the residuals on the target plane between the observed and expected codes and $(\Sigma^s_{(u,v)})^{-1}$ is the error covariance matrix for the given ray-pose combination that accounts for errors heteroscedasticity in the image plane.

In our setting, we aim to simultaneously minimize the optical center o, the direction $d_{(u,v)}$ of all rays and the pose RT_s for each exposure of the target. Similarly to [33], we can also take advantage of the conditional independence of the parameters to implement an alternating optimization scheme that seeks optimal o and $d_{(u,v)}$ assuming last estimation of RT_s fixed, and vice-versa. While our optimization involves less parameters, the optimization itself is more complex since the common optical center introduces a coupling between the rays which cannot be estimated independently anymore. As a consequence, the rays optimization step simultaneously estimates the optical center o and the ray directions $d_{(u,v)}$ given all the poses. For the pose estimation step we adopt the same ICP-based optimization introduced in [33]. The former step is discussed in detail in section 4.3.1 while, for the latter, we refer the reader to the original paper.

To start the alternating optimization we need a good initial approximation for the involved parameters. To this end, we gather a set of 3D-2D point correspondences assuming a discrete grid of target points similar to what can be commonly obtained with a chessboard. Then, we use *calibrateCamera* function provided by OpenCV [43] to obtain target poses for each exposure and the direction of each ray. Note that starting condition is not a critical aspect since most baseline calibration methods are more than adequate to provide a reliable initial configuration, especially when dealing with cameras that can be assumed to be almost central (differently, the method proposed in [33] should be preferred). In addition, the iterative method is based on a least-square minimization which is guaranteed to converge to a (possibly local) minimum.

Optical Center and Rays Direction Optimization In the o and $d_{(u,v)}$ optimization step we consider target poses constant. Let

$$\mathbf{x}^s_{(u,v)} = RT_s \begin{pmatrix} \mathbf{Co}^s_{(u,v)} \\ 0 \\ 1 \end{pmatrix}$$

be the 3D coordinates of the observed code $\mathbf{Co}^s_{(u,v)}$ transformed trough the pose RT_s. As shown in [33], the generalized least squares formulation with respect to the target coordinates corresponds to a linear least squares with the distance of each ray and its associated 3D point $\mathbf{x}^s_{(u,v)}$, we can formulate the estimation of the optical center o as:

$$\underset{o}{\operatorname{argmin}} \sum_{u,v} \min_{d_{(u,v)}} \sum_{s} \|(h^s_{(u,v)})^T (I - d_{(u,v)} d^T_{(u,v)})\|^2 \tag{4.11}$$

where $h^s_{(u,v)} = (\mathbf{x}^s_{(u,v)} - o)$, and the internal minimization apt to find the best $d_{(u,v)}$ minimizing the sum of squared distances between $r_{(u,v)}$ and all the $\mathbf{x}^s_{(u,v)}$. We start by re-writing the squared norm in (4.11) as $(h^s_{(u,v)})^T (I - d_{(u,v)} d^T_{(u,v)}) h^s_{(u,v)}$ to obtain

$$\underset{o}{\operatorname{argmin}} \sum_{u,v} \sum_{s} \|h^s_{(u,v)}\|^2 - \max_{d_{(u,v)}} \sum_{s} \left(d^T_{(u,v)} h^s_{(u,v)}\right)^2 \tag{4.12}$$

Let $\bar{x}_{(u,v)}$ be the centroid of the point cloud generated by the intersections of the ray $r_{(u,v)}$ and the target for each observed pose. Also, let $\bar{h}_{(u,v)} = (\bar{x}_{(u,v)} - o)$ be the distance vector between o with such centroid. By expressing $h^s_{(u,v)}$ as the summation of the two components:

$$h^s_{(u,v)} = (\mathbf{x}^s_{(u,v)} - \bar{x}_{(u,v)}) + \bar{h}_{(u,v)}$$

and expanding the formulation in (4.12) we obtain:

$$\underset{o}{\operatorname{argmin}} \sum_{u,v} N_{(u,v)} \left(tr(\mathbf{S}_{(u,v)}) + \|\bar{h}_{(u,v)}\|^2\right) - \tag{4.13}$$

$$- \max_{d_{(u,v)}} d^T_{(u,v)} \left(N_{(u,v)} \mathbf{S}_{(u,v)} + N_{(u,v)} \bar{h}_{(u,v)} \bar{h}^T_{(u,v)}\right) d_{(u,v)} \tag{4.14}$$

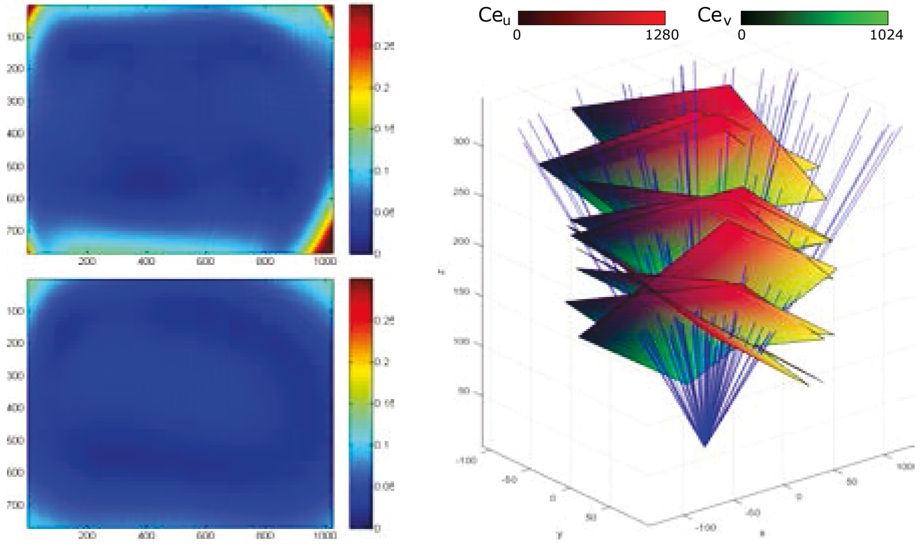


Figure 4.15: Left: RMSe between observed and expected codes for each $r_{(u,v)}$ at the first (top) and last (bottom) iteration. Right: Rays and calibration target poses recovered by the optimization. Only a subset of the camera rays are plotted for visualization purposes.

where $\mathbf{S}_{(u,v)}$ and $N_{(u,v)}$ are respectively the covariance matrix and the size of the point cloud generated by $r_{(u,v)}$.

Since we start our optimization with a configuration close to the optimum, we expect that the distance between each ray and its expected code is as small as few target pixels. This implies that the spatial extent of each point cloud is order of magnitude smaller than the distance $\|\bar{h}_{(u,v)}\|_2$. Under this assumption, an approximate maximizer for (4.14) is given by

$$d_{(u,v)} = \frac{\bar{h}_{(u,v)}}{\|\bar{h}_{(u,v)}\|^2} \tag{4.15}$$

By substituting (4.15) into (4.13) and (4.14), after some simplifications, we obtain the following alternative formulation

$$\underset{o}{\operatorname{argmax}} \sum_{u,v} N_{(u,v)} \frac{\bar{h}^T_{(u,v)} \mathbf{S}_{(u,v)} \bar{h}_{(u,v)}}{\|\bar{h}_{(u,v)}\|^2} \tag{4.16}$$

Problem (4.16) cannot be solved in a closed form. To provide a good approximate solution we compute the derivative with respect to o:

$$\begin{aligned} &\tfrac{\partial}{\partial o} \textstyle\sum_{u,v} N_{(u,v)} \frac{\bar{h}^T_{(u,v)} \mathbf{S}_{(u,v)} \bar{h}_{(u,v)}}{\|\bar{h}_{(u,v)}\|^2} \\ &= \textstyle\sum_{u,v} 2 N_{(u,v)} K_{(u,v)} \bar{h}_{(u,v)} \end{aligned} \tag{4.17}$$

$$K_{(u,v)} = \frac{\left(-\mathbf{S}_{(u,v)} \|\bar{h}_{(u,v)}\|^2 + \mathbf{I} \left(\bar{h}^T_{(u,v)} \mathbf{S}_{(u,v)} \bar{h}_{(u,v)} \right) \right)}{\|\bar{h}_{(u,v)}\|^4} \tag{4.18}$$

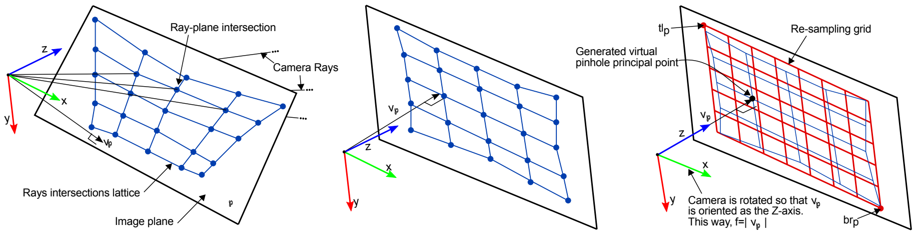


Figure 4.16: Left: the intersection of the camera rays with an image plane $\wp$ generates a projective distorted lattice. Center: An optimal plane is estimated to let the lattice be as regular as possible. Right: Camera reference system is rotated so that the image plane is orthogonal to the optical axis. Then, the lattice is re-sampled on an uniform grid to compute the undistortion function.

If $K_{(u,v)}$ is known, o can be obtained by setting to zero Equation (4.17) and solving the resulting linear system:

$$\sum_{u,v} 2N_{(u,v)}K_{(u,v)}o = \sum_{u,v} 2N_{(u,v)}K_{(u,v)}\bar{x}_{(u,v)} \tag{4.19}$$

Since $K_{(u,v)}$ is itself a function of o, the maximization problem (4.16) is tackled iteratively by computing $K_{(u,v)}$ with the estimate of o at iteration $t-1$ and then solving (4.19) to obtain a new estimate at iteration t and repeating this process until $\|o^{(t)} - o^{(t-1)}\| < \epsilon$. When the optical center is found, the direction of each ray is computed with equation (4.15). A qualitative result of the effect of the optimization process is shown in Fig. 4.15.

From ray bundle to virtual pinhole camera After the optimization of rays, optical center and poses we obtain a detailed model describing the light path entering the camera. Next, we need to choose a convenient image plane that define the intrinsic parameters of a new virtual pinhole camera, along with a non-parametric dense undistortion function to be applied to the acquired images.

As a preliminary step, all rays are translated so that their unique intersection point o lies at the origin. After that, we define a plane $\wp$ as the locus of points x satisfying $\langle x - v_\wp, n_\wp \rangle = 0$, with $n_\wp = \frac{v_\wp}{\|v_\wp\|}$. As soon as an image plane $\wp$ is chosen, it generates a virtual camera with the z-axis oriented as the vector $n_\wp$ and with a focal length $f = \|v_\wp\|$. When choosing any $\wp$ intersecting the ray bundle, all the intersection points inherit the lattice topology from the camera sensor (Fig. 4.16, Left). However, the lattice projective distortion and size are affected by the orientation and distance of $\wp$, respectively. Trough an optimization process we first find the best image plane to minimize the lattice projective distortion. Then, we generate the image undistortion function by re-sampling the lattice in a uniform grid. The grid position and size will define the projection of the optical center on the image plane (i.e. the pinhole parameters c_x and c_y) and the undistorted image size (see Fig. 4.16 for a complete description of the process).

Image Plane Estimation Choosing a good plane $\wp$ is crucial to ensure the lattice be as regular as possible and increase the quality of the subsequent re-sampling and interpolation processes. The optimal image plane is computed in two steps. First, we estimate the plane orientation $n_\wp$ to minimize the variance of the squared distance between each plane-ray intersection point and its neighbours. This ensure the lattice to be as regularly shaped as possible (Fig. 4.16, Center). After that, the scaling factor $\|v_\wp\|$ (i.e. the plane distance from the origin) is computed so that the average distance between all the points is equal to 1.

Let $I_d \subset \mathbb{R}^2$ be the set of (u, v) indices of the rays. Let $U(i \in I_d) = U(u, v) = \{(u-1, v), (u+1, v), (u, v-1), (u, v+1)\}$ the function defining the set of four neighbours of a ray indexed by i. The squared distance between the 3D intersections generated by rays r_i and $r_{j \in I_d}$ with a plane $\wp$ lying at unitary distance from the origin is given by:

$$D^2_{i,j} = \|\frac{d_i}{n^T_\wp d_i} - \frac{d_j}{n^T_\wp d_j}\|^2 \tag{4.20}$$

Consequently, the variance of the squared distances $D^2_{i,j}$ between each ray and its neighbours is given by the function

$$f_D = \sum_i \sum_{j \in U(i)} \left(D^2_{i,j}\right)^2 - \left(\sum_i \sum_{j \in U(i)} D^2_{i,j}\right)^2 \tag{4.21}$$

We cast the plane orientation problem as

$$\begin{aligned} &\underset{n_\wp}{\text{argmin}} && f_D \\ &\text{such that} && \|n_\wp\| = 1 \end{aligned} \tag{4.22}$$

solved via geodesic steepest descent. We start with an initial estimate of $n^{(0)}_\wp = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T$. For each iteration t, we update the estimate of $n^{(t)}_\wp$ enforcing the constraint of $\|n_\wp\| = 1$ by rotating $n^{(t)}_\wp$ around the rotation axis $\Psi = \nabla f^{(t-1)}_D \times n^{(t-1)}_\wp$ for an angle $\theta = \lambda \min(\|\Psi\|, \epsilon)$. The constant λ affects the speed of the gradient descent while ϵ gives an upper bound on the amount of rotation to avoid instabilities.

To perform effectively the optimization, ∇f_D can be analytically computed as follows:

$$\begin{aligned} \nabla f_D &= \sum_i \sum_{j \in U(i)} 2 D^2_{i,j} \frac{\partial}{\partial n_\wp} D^2_{i,j} - \\ &- \left(\sum_i \sum_{j \in U(i)} D^2_{i,j}\right)\left(\sum_i \sum_{j \in U(i)} \frac{\partial}{\partial n_\wp} D^2_{i,j}\right) \end{aligned} \tag{4.23}$$

$$\begin{aligned} \frac{\partial}{\partial n_\wp} D^2_{i,j} &= \frac{2}{(n^T_\wp d_i)^2}\left(\frac{d^T_i d_j}{n^T_\wp d_j} - \frac{d^T_i d_i}{n^T_\wp d_i}\right) d_i + \\ &+ \frac{2}{(n^T_\wp d_j)^2}\left(\frac{d^T_j d_i}{n^T_\wp d_i} - \frac{d^T_j d_j}{n^T_\wp d_j}\right) d_j \end{aligned} \tag{4.24}$$

Once $n_\wp$ has been recovered, we set $f = \|v_\wp\| = \frac{1}{2N}\sum_i \sum_{j\in U(i)} D_{i,j}$ where N is the number of lattice edges.

Generating the Undistortion Map Once an optimal plane has been found, we setup an interpolation procedure to re-sample the points on a regular grid. Let $p_{(u,v)}$ be the intersection of $r_{(u,v)}$ with the optimized plane $\wp$. First, all the points $p_{(u,v)}$ are rotated around the origin so that the vector $v_\wp$ coincides with the z-axis (Fig. 4.16, Right). After discarding the third component of all the points (all equal to 1 after the rotation), we compute the integral coordinates of the top-left $tl_p \in \mathbb{Z}^2$ and bottom right $br_p \in \mathbb{Z}^2$ corners of the containing bounding-box. At this point, we can provide the intrinsic matrix of the new virtual pinhole camera as:

$$K = \left(\begin{array}{cc|c} \|v_\wp\| & 0 & -tl_p \\ 0 & \|v_\wp\| & \\ \hline 0 & 0 & 1 \end{array}\right) \tag{4.25}$$

The undistorted image associated to the camera described by K corresponds to a unit-spaced grid inside the area of the bounding box. This leads to the construction of a dense displacement function $U_d : B \to \mathbb{R}^2$ that maps the coordinates of the output undistorted image $B \subset \mathbb{N}^2$ to sub-pixel coordinates of the input image[1].

To produce the displacement map U_d, we generate the quadrilaterals $q_1 \dots q_n$ formed when considering the 4-neighbours connectivity of the points $p_{(u,v)}$ with the topology induced by the rays lattice. For each quadrilateral q_i, we compute the homography H_i transforming the inner space bounded by its four vertices into the square defined by the CCD location of the four rays associated to each vertex. Then, the displacement map U_d can be obtained by:

$$U_d(u', v') = H_{Q(u',v')}(u', v', 1)^T \tag{4.26}$$

where $Q(u', v')$ is the function that returns the index of the quadrilateral containing the point $\begin{pmatrix} u' & v' \end{pmatrix}^T$, if exists.

Filtering data outliers Apart for being central, our model gives no constraint on the relative position of the rays. As a consequence, erroneous data caused by failures in the phase decoding process may lead to outliers in the ray estimation. Since no regularization is involved, we included a data filtering step at each alternation of rays-poses optimization. Specifically, we define the function $E(u, v)_s : I_d \to \mathbb{R}$ as the point-line distance between the ray $r_{(u,v)}$ and the point $x^s_{(u,v)}$. We then filter the observed codes $\mathbf{Co}^s_{(u,v)}$ by considering the median of the error function E in a squared neighbourhood of each point (u, v). If $E(u, v)_s$ is greater than κ times the median, $\mathbf{Co}^s_{(u,v)}$ is marked as invalid and not used any more in the subsequent iterations. Rays with less than 5 observed codes are completely removed from the optimization. A qualitative example of the output of the filtering process is shown in Fig. 4.17. Even if we filter erroneous observed codes, it may

[1] The obtained undistorted image has the same size of the bounding box

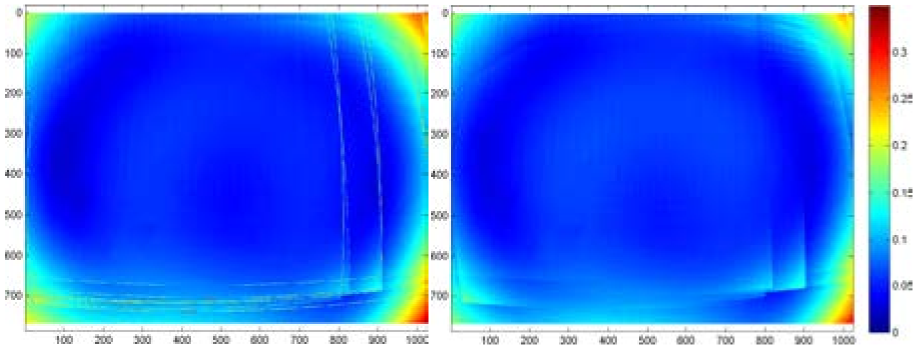


Figure 4.17: The effect of codes filtering step displayed by accumulating the value $E(u, v)$ among all the poses s. Left: original data may contain clear outliers near the boundary of the target. Right: after the filter almost all the spikes are no more visible.Note that remaining artefacts are not topological outliers (as for 4.3.1) and are probably due to the target non-planarity at the corners.

happen to obtain some quadrilaterals q_i for which the topological order of the vertices is not coherent with the order of the relative rays. Since this would lead to a non-injective displacement map U_d, such rays are marked as outlier and subsequently replaced by a linear interpolation performed over all the unmarked neighbours.

Dealing with Stereo Cameras

Since each ray acts independently with respect to the others, our approach can be easily extended to simultaneously calibrate and rectify a stereo rig. The pose optimization step remains exactly the same with the foresight to merge the two bundle of rays associated to each camera. Conversely, the optical centre and rays direction optimization can be performed independently on the two sets operating the same instance of target poses.

As a starting configuration for the subsequent optimization we performed the intrinsic and extrinsic calibration of the camera rig using the function provided by OpenCV library. Then, we are creating a single virtual imaging device by adopting the reference frame of the first camera for both. At the end of the optimization, we obtain an estimate of the two optical centres o_1 and o_2 and the directions of the rays in the two bundles. From this point, we roto-translate the rays to let o_1 coincide with the origin and the epipole $e = (o_2 - o_1)$ being oriented along the x-axis.

Rectification and Undistortion Map If we constrain the image plane optimization so that $n_\wp$ remains orthogonal to e, the estimated plane would have the property to keep all the epipolar lines for the left and right cameras being parallel. To achieve this, we slightly modify the optimization discussed in section 4.3.1 by fixing the rotation axis $\Psi = \frac{e}{\|e\|}$ and the rotation angle to $\theta = \lambda \min(\langle \nabla f_D^{(t-1)}, \Psi \times n_\wp^{(t-1)} \rangle, \epsilon)$.

After image plane optimization, two sets of points are generated by the intersection of the two ray bundles with the plane. The set of points generated by the right camera is

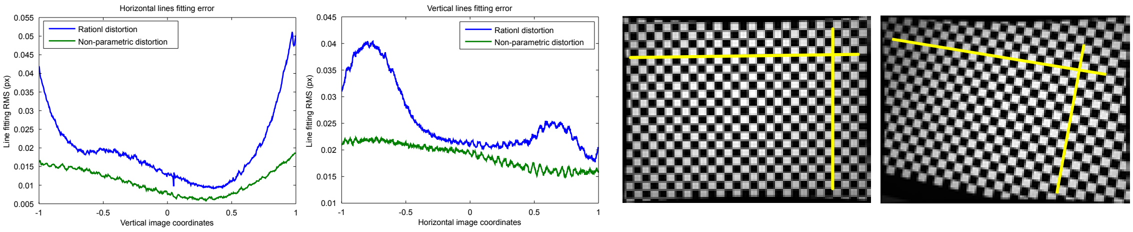


Figure 4.18: Left plots show a comparison of our non-parametric distortion model compared with Fitzgibbon's rational model. On the right, two examples of an undistorted images generated with our method. Note that black artifacts on the edges are due to the lack of clipping after undistortion (yellow guides have been superimposed in order to help to better appreciate rectification).

translated in the opposite direction of the x-axis by the length of the baseline $T = \|e\|$ to let the right optical center coincide with the left one. Subsequently, two different bounding boxes are generated with the two sets of points. The height of the two boxes (i.e. the vertical coordinates of the top-left and bottom-right corners) are forced to be equal so that the epipolar lines are coincident with the rows of the two images. To keep the largest available common area between the two images, the left edge of the merged bounding box is taken from the bounding box of the right point set. Symmetrically, the right edge is taken from the left point set. Note that, this way, the intrinsic matrices of the two cameras are exactly the same.

Finally, we compute the re-projection matrix

$$Q = \left(\begin{array}{ccc|c} 1 & 0 & 0 & -tl_p \\ 0 & 1 & 0 & \\ \cline{4-4} 0 & 0 & 1 & \|v_\wp\| \\ 0 & 0 & 1/T & 0 \end{array}\right) \tag{4.27}$$

so that, given any dimensional image point (u', v') and its associated disparity d, it can be projected into four-dimensional projective space with

$$\begin{pmatrix} x & y & z & w \end{pmatrix}^T = Q \begin{pmatrix} u' & v' & d & 1 \end{pmatrix}^T \tag{4.28}$$

4.3.2 Experimental Section

In order to assess the performance of the proposed approach, we compared it against the unconstrained model [33] and the rational distortion model proposed by Claus and Fitzgibbon[2] [68] in both single camera and stereo setups[3].

Our test setup included two PointGrey Flea3 1Mp grayscale cameras with approximately 60^o field of view, fastened to a common rigid bar with a disparity of about 5cm.

[2] The most accurate method available from the OpenCV library

[3] Source code and datasets available at: *anonymized URL*

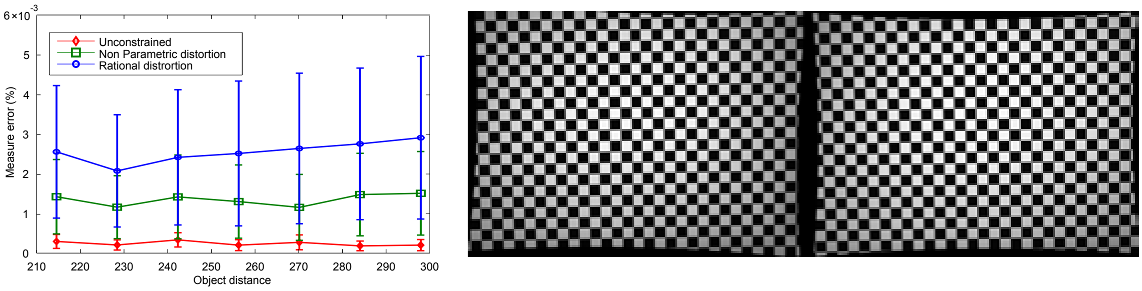


Figure 4.19: In the left-most plot we show the average relative error between the expected and measured distance of two triangulated points. In the right we show an example of a stereo-rectified image pair.

The calibration target was a 380×304mm commercial LCD monitor with a resolution of 1280×1024 pixels. The cameras have been calibrated using a set of 20 shots and tested over a set composed of 40 different shots of the same active target. These shots have been acquired both with a single camera and with the complete camera pair, taking care to cover as much as possible of the respective fields of view. The target has been acquired at random poses with a distance from the cameras ranging from roughly 100 to 300 mm and a rotation with respect to the optical axis ranging from 0 to about $\frac{\pi}{4}$ radians. (Fig. 4.15, Right) Using the same data sets, we performed three different calibrations, using respectively the fully unconstrained model (Unconstrained), the non-parametric distortion proposed in this work (Non-Parametric) and the rational distortion (Rational). Average convergence time for both Unconstrained and Non-Parametric calibration was about 10 minutes, while Rational always required less than 2 minutes.

Finally, we compared the performance of these methods by means of two experiments, assessing respectively the ability of providing a strictly projective virtual camera and to perform an accurate triangulation in the 3D space.

Image Undistortion

With this experiment we are testing the quality of the undistortion, that is how well the virtual pinhole camera obtained with the different methods approximate an ideal projective geometry. To this end, we exploited the projective invariance of straight lines. Specifically, for each horizontal and vertical scanline of the undistorted camera we collect the observed codes and we fit a straight line on them. Since we can assume the screen pixels to be regular and equally spaced, better pinhole approximation should exhibit a lower RMS error to the fitted line. Since a virtual pinhole cannot be produced with the fully unconstrained model, in Fig. 4.18 we plotted only the results for the Non-Parametric and the Rational model. While both methods are affected by some error, it is clear that the approximation given by the Non-Parametric approach yields less distorted lines. Furthermore, the structured nature of the RMS error obtained with the Rational model strongly suggests a systematic error due to the inability of the model to properly fit the data.

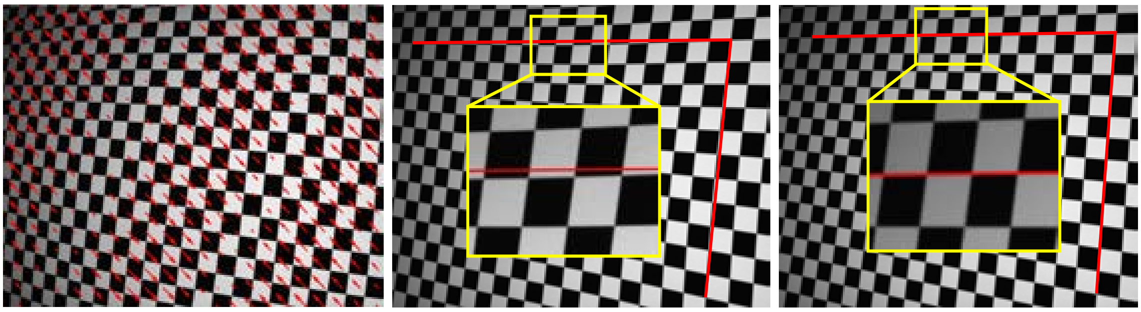


Figure 4.20: A small synthetic sinusoidal pixel displacement has been further applied to the acquired images (left). As expected, the rational model is unable to handle this distortion (center), which can be corrected with our non-parametric distortion model (right).

Note that, while our approach yields a much smaller undistortion error than the Rational model, there is still a strong spatial coherency in the error. We think the non-centrality of the camera to be the predominant error source, since the RMS consistently grows towards the image boundaries. We suggest that the non-planarity could also contribute, being the borders of the target usually located near the border of the captured image.

3D Measurement

In our second experiment we investigate the calibration quality of the camera pair. Using the rectificated image pair, we triangulate the 3D position of two random screen pixels observed by both cameras. We repeated the experiment for several shots with different positions of the target screen. In Fig. 4.19 we plotted the average relative error between the expected and measured distance of two triangulated points with respect to the distance of the target from the camera pair. In this case, the Unconstrained model shows the best performance as it produces a lower error at any distance and more repeatable measures. The Non-Parametric model exhibits a slightly higher error. This proves that the additional constraint hinders a perfect calibration. Still it is noticeably more reliable than the Rational model, thus it can be deemed as a reasonable alternative to the totally free model when high accuracy is needed, but it is not desirable to lose the advantages of the pinhole model. Finally, in Fig. 4.21 we give a qualitative example of a reconstructed range-map after stereo rectification provided by the OpenCV *stereoRectify* function and our calibration pipeline. The better alignment of epipolar lines with image rows gives a more precise and dense reconstruction, especially on grazing surfaces.

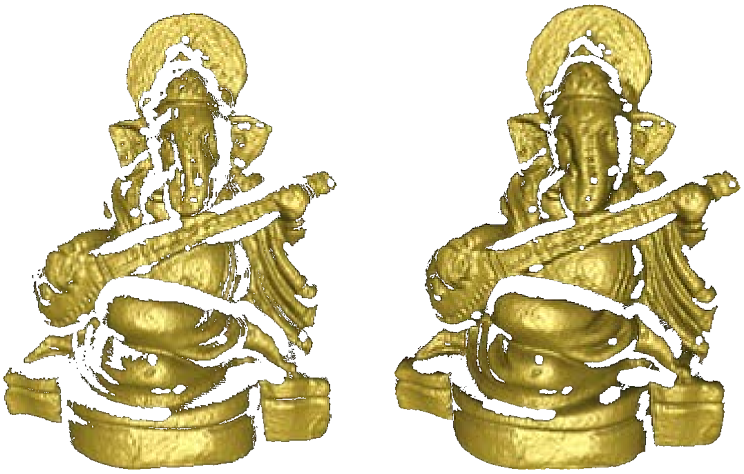


Figure 4.21: Reconstructed range-map triangulated with OpenCV calibration and rectification (Left) and our proposed method (Right).

II

Non-Rigid matching

5

Related Work

In the previous part of the thesis we have seen how finding matching points between two objects is one of the fundamental task in the processing of data in computer vision tasks. In that specific case objects were represented by images acquired by one or more cameras, and we tried to find correspondences minimizing some measure of euclidean distance distortion.

Another challenging family of matching problems is that of ”deformable” shape matching. In this case we allow objects to undertake Non-Rigid transformations, that is, euclidean distances in the embedded space are not preserved by the deformation. Researchers have focused they attention in the particular class of quasi-isometric deformations in which some notion of *intrinsic* distance is preserved by the transformation. This is the case, for instance, of articulated bodies (e.g. humans) in different poses [48, 126, 127].

Although 2D Non-Rigid shape matching is a relevant field in the Computer Vision and Geometry Processing communities [89, 167], we focus our attention on the three-dimensional case.
As opposed to images matching, usually 3D shapes lack of local informations like texture and colors, making feature descriptors of shapes not as distinctive. Moreover when we aim to be invariant to quasi-isometric deformations looking at the preservation of Euclidean distances in the $\mathbb{R}^3$ embedding of the shape is not an options, we instead look at the preservation of some intrinsic metric (we consider shapes as a destigmatization of a smooth Riemmanian manifolds), that is some distance function on the manifold which is independent to the embedding space and depends only on the Riemannian structure of the shape.

In this scenario, a straightforward measure of the correspondence quality is the distortion of the intrinsic metric, and the correspondence problem can be easily formulated in term of finding the correspondence that minimizes this metric distortion. The simplest construction of a intrinsic metric is the geodesic metric, which corresponds to the length of the shortest curve on the surface (minimal geodesic) connecting two given points. Despite its natural interpretation as distance over the surface, geodesic distance is very sensitive to noise, other intrinsic metrics have been proposed trying to mitigate this problem based on the *diffusion geometry* [70].

5.1 Partial shape matching

So far we assumed that the two shapes to be matched was a complete representation of the same ”object”. Even more challenging is *partial correspondence*, where one is shown only a subset of the shape and has to match it to a deformed full version thereof. Partial correspondence problems arise in numerous applications that involve real data acquisition by 3D sensors, which inevitably lead to missing parts due to occlusions or partial view.

For rigid partial correspondence problems, arising *e.g.*, in 3D scan completion applications, many versions of regularized iterative closest point (ICP) approaches exist, see for example [18, 22].

Attempts to extend these ideas to the non-rigid case in the form of non-rigid or piece-wise rigid ICP have been explored in recent years [140]. By nature of the ICP algorithm, these methods rely on the assumption that the given shapes can be placed in approximate rigid alignment to initiate the matching process. As a result, they tend to work well under small deformations (*e.g.*, when matching neighbouring frames of a sequence), but performance deteriorates quickly when this assumption does not hold.

For the non-rigid setting, several metric approaches centred around the notion of minimum distortion correspondence [48] have been proposed. Bronstein *et al.* [45, 47] combine metric distortion minimization with optimization over matching parts, showing an algorithm that simultaneously seeks for a correspondence and maximizes the *regularity* of corresponding parts in the given shapes.

Rodolà *et al.* [182] subsequently relaxed the regularity requirement by allowing sparse correspondences, and later introduced a mechanism to explicitly control the degree of sparsity of the solution [181]. Finally, in [189] the authors proposed a voting-based formulation to match shape extremities, which are assumed to be preserved by the partiality transformation. Being based on spectral features and metric preservation, the accuracy of the aforementioned methods suffers at high levels of partiality, where the computation of these quantities becomes unreliable due to boundary effects and meshing artifacts. Furthermore, these methods suffer from high computational complexity and generally provide only a *sparse correspondence*.

Pokrass *et al.* [177] proposed a descriptor-based partial matching approach where the optimization over parts is done to maximize the matching of bags of local descriptors. The main drawback of this approach is that it only finds similar parts, without providing a correspondence between them.

Windheuser *et al.* [237] formulated the shape matching problem as one of seeking minimal surfaces in the product space of two given shapes; the formulation notably allows for a linear programming discretization and provides guaranteed continuous and orientation-preserving solutions. The method was shown to work well with partial shapes, but requires watertight surfaces as input (*e.g.*, via hole filling).

Brunton *et al.* [51] used alignment of tangent spaces for partial correspondence. In their method, a sparse set of correspondences is first computed by matching feature descriptors; the matches are then propagated in an isometric fashion so as to cover the largest possible regions on the two shapes. Since the quality of the final solution directly depends on

the initial matches, the method is understood as a “densification” method to complement other sparse approaches.
Other recent works include the design of robust descriptors for partial matching [223]. In the context of collections of shapes, partial correspondence has been considered in [60, 73, 222].

All the aforementioned works are based on the notion of point-wise correspondence between shapes. Recently, Ovsjanikov *et al.* [171] proposed the *functional maps* framework, in which shape correspondence is modeled as a linear operator between spaces of functions on the shapes.
The main advantage of functional maps is that finding correspondence boils down to a simple algebraic problem, as opposed to difficult combinatorial-type problems arising in, *e.g.*, the computation of minimum-distortion maps.
While several recent works showed that functional maps can be made resilient to missing parts or incomplete data [114, 131] overall this framework is not suitable for dealing with partial correspondence.

In section 6.2 we propose a method for computing partial functional correspondence between non-rigid shapes. We use perturbation analysis to show how removal of shape parts changes the Laplace-Beltrami eigenfunctions, and exploit it as a prior on the spectral representation of the correspondence. We treat corresponding parts as optimization variables and use them to weight the functional correspondence.

5.2 Joint shape matching

While all the aforementioned works tackle the problem of shape matching from a pairwise perspective, there are some recent attempts trying to take advantage of the information brought in by matching a collection of shapes simultaneously [60, 115]. The key idea is that considering multiple shapes at once can provide information about which parts of the shapes are characteristic of the model and which ones has to be considered as outliers.
One natural requirement is cycle-consistency – namely the fact that map composition should give the same result regardless of the path taken in the shape collection.

Probably the earliest attempt to tackle multiple shape matching in a principled way is the synchronistic matching approach of Schmidt *et al.* [193]. Given a collection of planar shapes, the authors model the joint matching problem as the search of a shortest path in their product space. Due to the resulting intractability, the problem is relaxed to a series of pairwise sub-problems, and the cycle-consistency criterion introduced as a regularizer. The method allows to improve initial pairwise solutions, but consistency is not satisfied exactly and the method operates under the assumption that all shapes in the collection are similar.
Pachauri *et al.* [172] took a similar perspective by formulating the matching problem using the language of combinatorial optimization; due to the spectral relaxation they perform, the method tends to be sensitive to noise and outliers.
Recently, Yan *et al.* [243] formulated the problem as one of simultaneous multi-graph

matching [203], but similarly to [172, 193], cycle-consistency is relaxed and gradually infused in a pairwise matching process as a regularizer.
Zach *et al.* [248] were probably the first to make an explicit attempt at finding solutions meeting the cycle-consistency requirement. Starting from an initial graph of potential pairwise associations among the objects in the collection, they detect and remove erroneous edges as the ones giving rise to inconsistent loops in the graph. As an extension to this approach, Nguyen *et al.* [166] apply global optimization to select cycle-consistent maps while at the same time allowing edges to be replaced by better map compositions. The method performs well when the *full* point-to-point correspondence is known and accurate for all pairs of objects. Huang *et al.* [116] improved upon [166] by allowing sparse correspondences; however, the approach does not apply when the shapes being matched are only *partially* similar.
Finally, Sahillioğlu and Yemez [190] proposed a greedy algorithm, based on dynamic programming, that seeks nearly-isometric consistent solutions across all shapes in the collection. The approach only works well when matching shape extremities, and it is susceptible to outlier shapes and partiality. In particular, its accuracy depends on the specific ordering of the shapes in the collection.
All the methods outlined above demonstrate good practical performance in controlled settings, however there has been a general lack of theoretical guarantees that ensure correctness of the final correspondence under unfavorable conditions. First steps in this direction are taken by Huang and Guibas [115], who formulate a convex relaxation to the joint matching problem using the language of semi-definite programming. The authors derive theoretical guarantees on the recovery of the correct joint correspondence from possibly noisy input maps [127].
Very recent works in the field of information theory explore this direction more abstractly [59], giving conditions for perfect recovery under large outlier ratios.
Chen *et al.* [60] recently applied this analysis to match partially similar objects from a small fraction of densely corrupted pairwise maps. To our knowledge, their algorithm currently represents the state of the art within this family of approaches.

Although most of these approaches work well provided that the input maps are sufficiently accurate, they still suffer in the presence of noise (incorrect matches) in the maps themselves, or outlier (extra-class) shapes in the collection. Further, due to the combinatorial difficulty of imposing the consistency requirement, many of the existing schemes provide no guarantee that cycle-consistency is satisfied exactly.

In this section 6.1, we introduce a new method for the joint matching of multiple deformable shapes in a collection. Unlike the common approach outlined above, we do *not* require any pairwise correspondence to be given as input, and instead formulate the problem as an optimization directly over the space of cycle-consistent (multi-way) matches. As such, our method produces matches that are consistent by construction.

6

Shape matching under quasi-isometric deformations

Deformable shape matching is undoubtedly a challenging family of matching problems. Objects are allowed to undertake Non-Rigid transformations, that is, euclidean distances in the embedded space are not preserved by the deformation. Even when remaining in the $\mathbb{R}^3$ domain, and thus not applying the image projection transformation, finding correspondences between shapes under quasi-isometric deformations is a very ambitious problem which has received a great deal of attention by the scientific community in the last years. Even more difficult is when we have to deal with partiality on one or both (or more) of the shapes to be matched. One of the main complication in the non-rigid scenario is, indeed, the lack of meaningful local descriptors. If in the image and 3D rigid domain we can rely in texture information and/or strict spatial invariants, when we allow for quasi-isometric transformations we are left only with intrinsic properties of the surface. Even if various local descriptors have been proposed in the literature [49, 130, 238], the more successful descriptor still remain the WKS [26] which rely on global informations of the surface.

We can partition approaches to the non-rigid matching into two classes: sparse and dense matching. Algorithm adopting the first approach seek correspondences only between a restricted subset of points of the shapes, this means that a subsequent densification step is needed to diffuse correspondences over the whole shape. Dense matching approaches, conversely, try to solve the correspondence problem for the whole shape at once.

In this chapter we try to tackle the *Non-rigid Partial matching* problem from both perspectives. In section 6.1 we cast the problem of multiple shape matching as an optimization over the space of all the cycle-consistent matches. The L1 optimization adopted allows us to be robust to shape partiality and outliers in the collection at the cost of retrieving only sparse correspondences. In section 6.2, conversely, we formulate the matching problem as the pursuit of a functional map, exploiting the behavior of the Laplacian eigendecomposition under partiality. This allows us to retrieve the point-to-point map between two partial shapes.

6.1 Consistent Partial Matching of Shape Collections via Sparse Modeling

Finding matches among multiple objects is a research topic that has received a good deal of attention in recent years. In its most common formulation, it translates to the problem of determining point-to-point maps between all shapes in a collection, subject to the requirement that the extracted correspondence be in some way “consistent”. To this end, a natural and widely accepted criterion is *cycle-consistency* [248], namely that composition of maps along loops in the collection should approximate the identity.

In this section, we introduce a novel technique to construct accurate, consistent correspondences within shape collections. Our formulation has the following key properties:

- The method operates by optimizing directly over the space of cycle-consistent correspondences, without requiring pairwise maps to be given as input. As a result, cycle-consistency is satisfied *exactly* by construction.
- We employ sparsity techniques in order to cope with *partially similar* as well as *outlier* shapes in the collection – an aspect that has received limited interest so far, but that can frequently occur in practical scenarios.

6.1.1 Multi-way matches

We model shapes as compact two-dimensional Riemannian manifolds S_i (possibly with boundary) embedded in $\mathbb{R}^3$, equipped with the intrinsic distance function d_i.

Let us be given a collection $\mathcal{C} = \{S_1, S_2, \dots, S_n\}$ of n shapes. The product space $S_1 \times \cdots \times S_n$ consists of all possible n-way (*i.e.*, joint) matches between the shapes in $\mathcal{C}$. However, in practical settings it is often the case that outliers (*e.g.*, shapes belonging to different classes) as well as partially similar shapes (*e.g.*, man and centaur) are present in

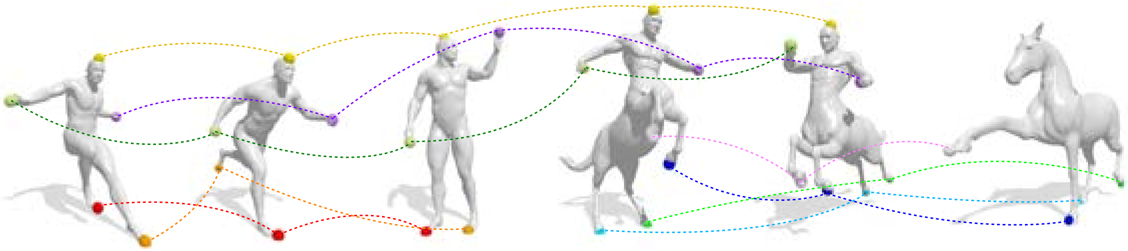


Figure 6.1: A partial multi-way correspondence obtained with our approach on a heterogeneous collection of shapes. Our method does not require initial pairwise maps as input, as it actively seeks a reliable correspondence by operating directly over the space of joint, cycle-consistent matches. Partially-similar as well as outlier shapes are automatically detected and accounted for by adopting a sparse model for the joint correspondence. A subset of all matches is shown for visualization purposes.

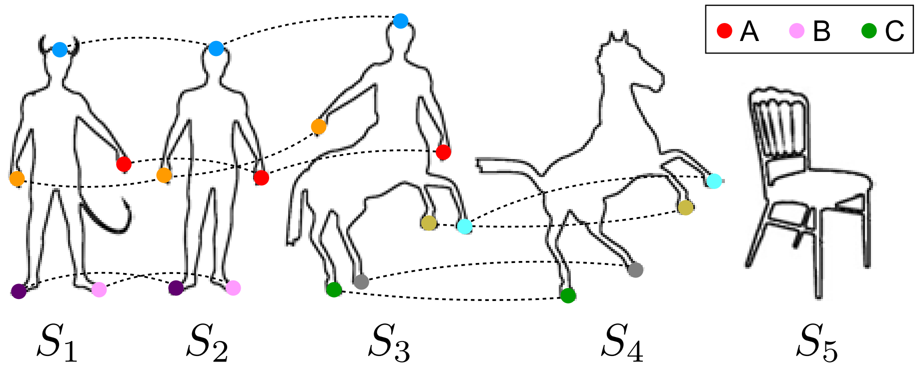


Figure 6.2: A collection of shapes may carry partiality at different levels. Our method allows to extract consistent correspondences reliably under partial similarity (*e.g.*, $S_1/S_2/S_3$) and at the same time detect and avoid outlier shapes (S_5).

the collection (see Fig. 6.2). In order to deal with such cases, we extend the set of possible joint matches as follows. Formally, we consider the set constructed as the union

$$\breve{\Gamma} = \bigcup_{k \in I} \prod_{j \in k} S_j \,, \tag{6.1}$$

where I is a collection of index sets k defined by the power set (denoted by $\mathcal{P}$) relation

$$I = \{k : k \in \mathcal{P}(\{1, 2, \ldots, n\}) \wedge |k| > 1\} \,. \tag{6.2}$$

In other words, $\breve{\Gamma}$ is the set of all possible m-fold Cartesian products between the shapes in $\mathcal{C}$, with $1 < m \leq n$. Clearly, this set also includes $S_1 \times \cdots \times S_n$ and in particular $|\breve{\Gamma}|$ grows exponentially with the number of shapes. Each element $\gamma \in \breve{\Gamma}$ with $|\gamma| = d \leq n$ now represents a joint match between a subset of d shapes from the collection.

Definition 1. *We define a* multi-way match *among $d \leq n$ shapes to be any element $\gamma \in \breve{\Gamma}$. A multi-way match is represented as the ordered d-tuple*

$$\gamma = (p_i)_{i \in k} \;\; \text{with } k \in I \text{ and } p_i \in S_i \,,$$

where k is a sequence of shape indices*, denoting the shapes matched by γ. We will write $p_i \in \gamma$ to say that the vertex p_i is matched via γ.*

Note that two multi-way matches $\gamma, \gamma' \in \breve{\Gamma}$ may in general have different lengths d and d'. In particular, they may or may not have shapes in common. We will therefore define the *overlap* $\gamma \cap \gamma'$ as the longest common subsequence of their shape indices. For example, in Fig. 6.2 we have the multi-way matches $A, B, C \in \breve{\Gamma}$. For A and B we have the overlap $A \cap B = (1, 2)$, whereas $A \cap C = (3)$ and $B \cap C = \emptyset$.

For our purposes, we are interested in subsets of $\breve{\Gamma}$ that satisfy certain properties, as described in the following:

Definition 2. *A* multi-way correspondence *between the n shapes in $\mathcal{C}$ is a subset $\Gamma \subset \breve{\Gamma}$ satisfying: for every $S_i \in \mathcal{C}$ and for every $p_i \in S_i$, there exists at least one $\gamma \in \Gamma$ such that $p_i \in \gamma$.*

The above definition ensures that in a multi-way correspondence each vertex of each shape is matched to corresponding vertices on (a subset of) the other shapes. We now define what is the meaning of cycle-consistency in our setting.

Definition 3. *We say that a multi-way correspondence Γ between shapes in the collection $\mathcal{C} = \{S_i\}_{i=1}^n$ is* cycle-consistent *if, for any $j, k, \ell \in \{1, \dots, n\}$, whenever Γ matches $p_j \in S_j$ to $p_k \in S_k$ and matches p_k to $p_\ell \in S_\ell$, then Γ also matches p_j to p_ℓ.*

Remark 1
A multi-way match is always cycle-consistent by construction, since it is an element of a product set. This applies to any cycle, with length possibly longer than 3.

Note that while individual multi-way matches are always cycle-consistent, a fixed point $p_i \in S_i$ might be mapped to different points on the other shapes via different multi-way matches. We therefore introduce the following notion:

Definition 4. *Two distinct multi-way matches $\gamma, \gamma' \in \Gamma$ are said to be* incompatible *whenever $p_i \in \gamma$ and $p_i \in \gamma'$ for some $p_i \in S_i$ and $i \in \{1, \dots, n\}$.*

An illustration of incompatible matches is given in Fig. 6.3.

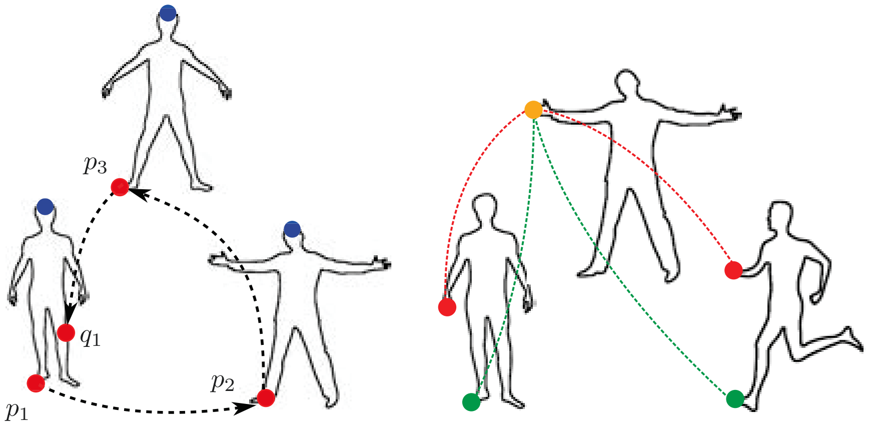


Figure 6.3: *Left*: The red matches violate cycle-consistency, since $p_1 \neq q_1$. *Right*: Example of two incompatible multi-way matches (red and green): the hand in the middle shape is assigned to multiple distinct points on the other shapes.

6.1.2 Optimizing over the space of multi-way matches

The goal of joint object matching is to determine a (possibly dense) correspondence among multiple shapes in a collection, with the requirement that the correspondence be consistent along cycles of any length. In this Section we formulate the joint matching problem as one of *minimum-distortion correspondence* [155]. Differently from most other approaches [116, 166, 172, 190, 193, 243, 248] our formulation comes with the *theoretical guarantee* of cycle-consistency, and additionally deals with partially similar as well as outlier shapes in a natural way.

Metric distortion. Suppose we are given two multi-way matches $\gamma, \gamma' \in \Gamma$ respectively putting $|\gamma|$ and $|\gamma'|$ points into correspondence, where in general $|\gamma| \neq |\gamma'|$. Then, we can quantify the quality of the correspondence by the cost function $\epsilon : \Gamma \times \Gamma \to \mathbb{R}_+ \cup \{\infty\}$ defined as:

$$\epsilon(\gamma, \gamma') = \max_{\substack{p_k, p_\ell \in \gamma \\ p'_k, p'_\ell \in \gamma'}} |d_k(p_k, p'_k) - d_\ell(p_\ell, p'_\ell)| \,. \tag{6.3}$$

Here we tacitly assume that the multi-way matches are compared only on their overlap, *i.e.*, over the shapes in common. In (6.3) we put $\epsilon(\gamma, \gamma') = \infty$ whenever γ and γ' are incompatible (see Fig. 6.3) or non-overlapping. This definition of cost encodes the maximum metric distortion attained by the two multi-way matches across the shape collection.

Multi-way L^p distortion. A multi-way correspondence $\Gamma \subset \breve{\Gamma}$ can be alternatively modeled as a binary function $g : \breve{\Gamma} \to \{0, 1\}$ such that for every $S_i \in \mathcal{C}$ and for every $q \in S_i$,

$$\sum_{\substack{\gamma \in \breve{\Gamma} \\ \text{s.t. } q \in \gamma}} g(\gamma) \geq 1 \,. \tag{6.4}$$

Note that function g can be seen as an indicator function over the space of all possible multi-way matches. Then, the condition above simply asks that each point in each shape is contained in at least one $\gamma \in \breve{\Gamma}$ for which $g(\gamma) = 1$, thus being a strict requirement to match all points in all shapes.

The overall metric distortion caused by a correspondence Γ can then be measured by the L^p distortion:

$$\|\epsilon\|^p_{L^p(g \times g)} = \sum_{\gamma, \gamma' \in \breve{\Gamma}} \epsilon^p(\gamma, \gamma') g(\gamma) g(\gamma') \,, \tag{6.5}$$

with $p \geq 1$. Now, determining a correspondence of minimum distortion amounts to seeking a minimizer (not unique in general) to:

$$\min_{g : \breve{\Gamma} \to \{0,1\}} \|\epsilon\|^p_{L^p(g \times g)} \tag{6.6}$$

where g ranges over all correspondences $\Gamma \subset \breve{\Gamma}$.

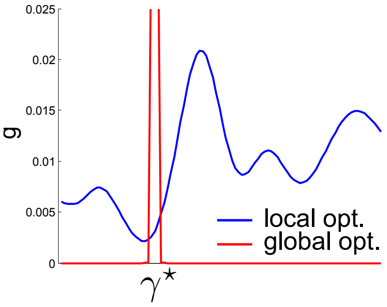
Example. In the specific case where $n = 2$, a multi-way match $\gamma = (p_1, p_2)$ reduces to a pair of points and the error criterion of Eq. (6.3) simplifies to the absolute metric distortion $\epsilon((p_1, p_2), (q_1, q_2)) = |d_1(p_1, q_1) - d_2(p_2, q_2)|$. Then, by taking the limit for $p \to \infty$ the expression (6.6) yields the classical Gromov-Hausdorff distance between metric spaces (S_1, d_1) and (S_2, d_2) [155].

Dealing with partiality. The combinatorial complexity of optimizing over all possible multi-way correspondences $\Gamma \subset \breve{\Gamma}$ makes the problem intractable even for small collections. Partial remedy to this issue is provided by relaxing the binary map to take continuous values, *i.e.*, $g : \breve{\Gamma} \to [0, 1]$.

We further note that, although Eq. (6.1) enables us to better deal with partiality, the constraint defined in Eq. (6.4) requires us to match all points in all shapes. However, we would like outlier shapes to not partake to the final correspondence. Furthermore, we want to allow *individual* shape points to be left unmatched if they do not find suitable matches throughout the collection.

We model this requirement by introducing a *sparse model* for the correspondence. To this end, we relax condition (6.4) by demanding $\sum_\gamma g(\gamma) = 1$ over $\breve{\Gamma}$. This requirement gives us an interpretation of g as a discrete probability distribution over the space of all multi-way matches. Importantly, the L^1-like constraint on g has a *sparsity-promoting* effect on the solution, hence modeling local forms of partiality.

Unfortunately, directly minimizing a problem of the form given in Eq. (6.6) subject to $\sum_\gamma g(\gamma) = 1$ would yield trivial solutions. Specifically, we can characterize the global minimizers by: $g(\gamma) = 1$ for $\gamma = \gamma^\star$ and $g(\gamma) = 0$ otherwise, where $\gamma^\star$ is taken to be any $\gamma \in \breve{\Gamma}$. This amounts to concentrating the whole mass of g into one single multi-way match, as illustrated in the inset figure.



We sidestep this issue by passing to a maximization problem. Suppose we are given, as opposed to the cost ϵ, a *similarity* function $s : \breve{\Gamma} \times \breve{\Gamma} \to \mathbb{R}_+$ measuring the extent to which two given multi-way matches preserve pairwise distances. A possible choice is given by the Gaussian score:

$$s(\gamma, \gamma') = e^{-\frac{1}{\mu^2}\epsilon^2(\gamma,\gamma')}\,, \tag{6.7}$$

where $\mu^2 \in \mathbb{R}_+$ is the variance of s. Note that $s(\gamma, \gamma') = 0$ whenever $\epsilon(\gamma, \gamma') = \infty$; that is, incompatible matches are assigned zero similarity. We get to the following optimization problem:

Problem 1
(Partial multi-way correspondence). Given a collection of shapes $\mathcal{C}$, we seek a *partial* multi-way correspondence among them as a maximizer to:

$$\max_{g:\breve{\Gamma}\to[0,1]} \quad \sum_{\gamma,\gamma'\in\breve{\Gamma}} s(\gamma,\gamma')g(\gamma)g(\gamma') \tag{6.8}$$

$$\text{s.t.} \quad \sum_{\gamma\in\breve{\Gamma}} g(\gamma) = 1 \tag{6.9}$$

$$\bar{c}(\gamma,\gamma')g(\gamma)g(\gamma') = 0 \quad \forall\, \gamma,\gamma' \in \breve{\Gamma}\,, \tag{6.10}$$

where we set $\bar{c}(\gamma, \gamma') = 1$ if the two matches are incompatible, and $\bar{c}(\gamma, \gamma') = 0$ otherwise.

Eq. (6.10) ensures that incompatible matches will not appear in any local optimum.

The transition to a maximization problem has a regularizing effect on its optima, as there are no trivial maximizers meeting the constraints in this case.

Remark 2
Any local solution to Problem 1 satisfies the key requirements of a multi-way correspondence: 1) it is always cycle-consistent (by construction of $\breve{\Gamma}$); 2) shape points are activated at most once by the correspondence (by Eq. (6.10)); and 3) partial matches are allowed (by $\breve{\Gamma}$ and Eq. (6.9)).

A note on symmetries. In case the shapes in the collection carry bilateral symmetries, mapping either side would in principle yield the same optimum for Problem 1. We deem correct such symmetric solutions as long as they remain consistent across all pairs of shapes (see Fig. 6.4).

6.1.3 Numerical optimization

Problem 1 is a non-convex quadratic program with $O(|\breve{\Gamma}|)$ variables; as such, it is in general very difficult to solve and to give guarantees on the optimality of the solution. In this Section we develop an efficient strategy to get good local solutions to this problem. The general strategy is to decompose it into two sub-problems: a robust process to get good match candidates (Section 6.1.3), and a restriction of the original problem to the reduced feasible set (Section 6.1.3).

First sub-problem (reducing the feasible set)

The first sub-problem is aimed at reducing the size of the feasible set $\breve{\Gamma}$ to a smaller subset of “stable” candidates $\Lambda \subset \breve{\Gamma}$. Then, we will directly optimize Problem 1 over the reduced feasible set Λ.

Outline. The general idea behind our formulation is that, given a collection of shapes, it is relatively easy and inexpensive to solve for one *single* multi-way match between them.

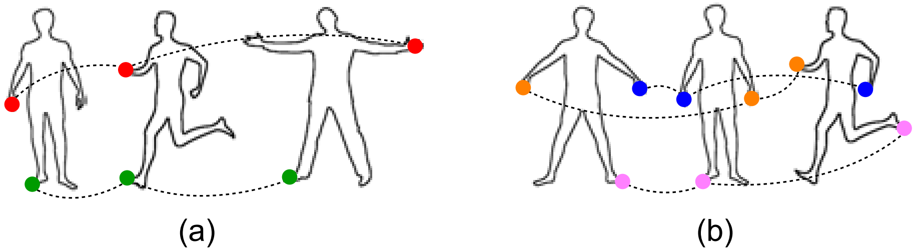


Figure 6.4: *(a)* Incorrect correspondence due to inconsistent handling of the symmetry. *(b)* Even if the solution is not orientation-preserving, symmetries are treated consistently.

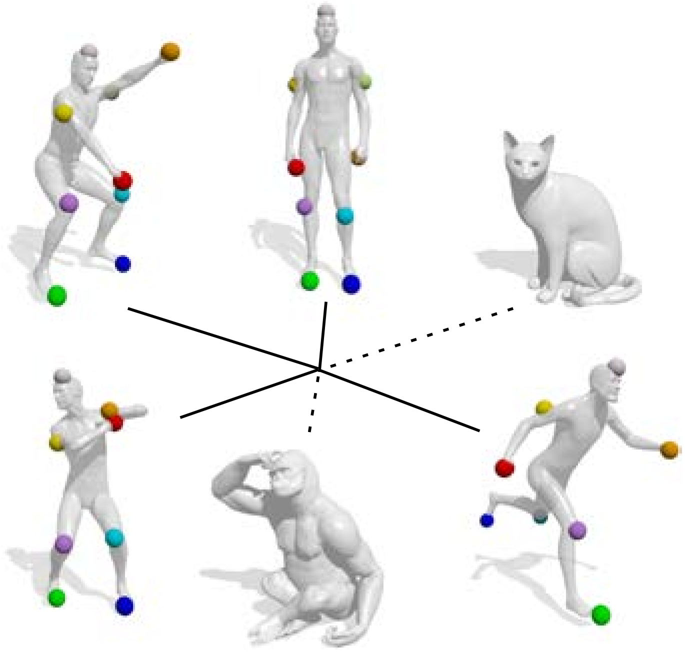


Figure 6.5: Outlier shapes are automatically excluded by our approach, as they do not find support from the other shapes in the collection. Note that the human shapes appearing in this example come from different datasets (TOSCA, SCAPE, SHREC'14). A subset of all matches is shown.

Specifically, we seek for a multi-way match $\gamma \in \breve{\Gamma}$ that maximizes a measure of point-wise similarity across several shapes, hence taking advantage of the stability induced by the whole shape collection. The final goal is to keep in the feasible set only multi-way matches maximizing this measure of similarity, since they are expected to be accurate and stable against outliers, as shown in Fig. 6.5.

This problem can be formulated as a series of quadratic programs with sparsity constraints (Eq. (6.11)), each yielding a multi-way match $\gamma \in \Lambda$. Note that mapping constraints are imposed such that *only* cycle-consistent matches are allowed to be local optima.

Solving for a single multi-way match. Assume for simplicity that $|S_i| = N$ for all $i = 1, \ldots, n$. Further, let us be given a point-wise similarity function $\tau : S_k \times S_\ell \to \mathbb{R}_+$, measuring the similarity of some descriptor defined at shape points (an example is given in section 6.1.4). Note that this function is not the same as the one defined in Eq. (6.7), which instead measures the similarity between multi-way matches.

We introduce the vector $\mathbf{x} \in [0,1]^{nN}$, representing a probability distribution over all points in $\bigcup_i S_i$. Then, consider the L^1-regularized non-convex quadratic program:

$$\max_{\mathbf{x} \geq 0} \mathbf{x}^\top \mathbf{A} \mathbf{x} \quad \text{s.t. } \mathbf{x}^\top \mathbf{1} = 1 \,. \tag{6.11}$$

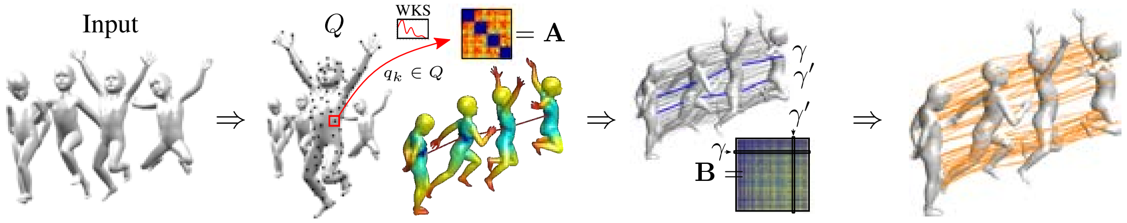


Figure 6.6: Our matching pipeline. First sub-problem (from left): Given a collection of shapes as input, a set Q of queries are generated (*e.g.*, by farthest point sampling in the joint WKS space); we then compute distance maps (shown here as heat maps over the shapes) in descriptor space from each shape point to each query $q_k \in Q$, and keep the vertices having distance smaller than a threshold; finally, a single multi-way match is extracted by solving problem (6.11). Second sub-problem: The multi-way matches extracted in the previous step are compared using a measure of metric distortion; the final solution (in orange) is obtained by solving problem (6.13) over the reduced feasible set.

Here, matrix $\mathbf{A}$ is a symmetric similarity matrix:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{S}^{1,2} & \cdots & \mathbf{S}^{1,n} \\ \mathbf{S}^{1,2} & \mathbf{0} & \cdots & \cdots \\ \vdots & \vdots & \mathbf{0} & \mathbf{S}^{n-1,n} \\ \mathbf{S}^{1,n} & \vdots & \mathbf{S}^{n-1,n} & \mathbf{0} \end{pmatrix}, \qquad (6.12)$$

where each symmetric block $\mathbf{S}^{k,\ell} \in \mathbb{R}^{N \times N}$ contains the similarity values between the points in S_k and S_ℓ, according to function τ. The reason for the zero blocks along the diagonal will become clear with Theorem 1. Note that the matrix above is not related to the block matrix appearing in [60, 115], which instead represents a collection of pairwise maps (ideally permutations).

The key result of this Section is that the support of any local maximizer to the problem above (*i.e.*, the set of points for which $\mathbf{x}_i \neq 0$) is guaranteed to be a *single* partial multi-way match $\gamma \in \breve{\Gamma}$ between the shapes in the collection, as we state in the following theorem.

Theorem 1. *Let* $\mathbf{x}$ *be a strict local maximizer of problem* (6.11)*, where* $\mathbf{A} = \mathbf{A}^\top$ *and* $\mathbf{A}_{ii} = 0$ *for all* $i = 1, \ldots, nN$*. Then,* $\mathbf{A}_{ij} > 0$ *for all* i, j *such that* $\mathbf{x}_i \neq 0$, $\mathbf{x}_j \neq 0$.

According to Thm. 1, local solutions to (6.11) cannot simultaneously activate points with zero similarity. This gives us a powerful means to restrict feasibility to solutions that activate *at most* one point per shape: It is sufficient to set $\mathbf{A}_{ij} = 0$ whenever indices i and j correspond to points on the same shape, *i.e.*, matrix $\mathbf{A}$ must have zero blocks on the diagonal.

Remark 3

Since local solutions to problem (6.11) are guaranteed to be multi-way matches, they are always cycle-consistent by definition.

A series of quadratic problems. Clearly, in order to construct the reduced set $\Lambda \subset \breve{\Gamma}$ we need a way to enumerate the local optima of problem (6.11). We do so by solving a sequence of problems of this form, each with a different data matrix (6.12). Specifically, for each problem we compute similarities from a reference descriptor (or "query") to all shape points, and we discard all dissimilar points.

Suppose we are given a collection Q of queries to compare against. A family of problems of the form (6.11) can then be generated as follows. Given a query $q_k \in Q$, for each shape $S_i \in \mathcal{C}$ we only consider the vertices $p \in S_i$ such that $\tau(p, q_k) > \xi$ for some threshold $\xi > 0$. In other words, each query selects a different subset of vertices from each shape; since we can generate and solve as many problems (6.11) as there are queries q_k, we can proceed constructively and store each solution in our reduced feasible set Λ, which will have size $|\Lambda| = |Q|$.

Note that each of these problems will be quite small, since the number N' of shape points that are similar to each query is significantly smaller than the total number of points N. We refer to Sec. 6.1.4 for an empirical evaluation. We also note that this approach is different from previous approaches which require pairwise maps as input or which require geometrically consistent samples to be pre-selected across the shapes [60, 115, 116, 166, 190].

Example. Suppose we are given a shape descriptor function $f : \bigcup_i S_i \to \mathbb{R}^m$, providing an embedding of all shapes in $\mathbb{R}^m$. The query set Q can be defined implicitly by a k-means clustering or by farthest point sampling directly in $\mathrm{Im} f$.

Numerical solution. It is worthwhile to note that problems of this form have a natural interpretation from the point of view of evolutionary game theory [23, 182]. We leverage this connection by adopting the infection-immunization dynamics algorithm [186], an efficient local optimizer with convergence guarantees that exploits the specific structure of problem (6.11).

Second sub-problem (correspondence)

We are now ready to solve a smaller version of Problem 1 by replacing $\breve{\Gamma}$ with the reduced set Λ. We proceed by directly rewriting the problem in matrix notation.

Suppose we solved $|\Lambda| = M$ instances (one per query) of problem (6.11), hence we have partial multi-way matches γ_i for $i = 1, \dots, M$ at our disposal. We can now compose the similarity terms $s(\gamma_i, \gamma_j)$ into a similarity matrix $\mathbf{B} \in \mathbb{R}_+^{M \times M}$ such that $\mathbf{B}_{ij} = \mathbf{B}_{ji} = s(\gamma_i, \gamma_j)$, and we set $\mathbf{B}_{ii} = 0$ for all $i = 1, \dots, M$ by Theorem 1. The correspondence function g can simply be represented by a vector $\mathrm{g} \in [0, 1]^M$. Similarly to the previous case, we arrive at the quadratic program:

$$\max_{\mathbf{g} \geq 0} \mathbf{g}^\top \mathbf{B} \mathbf{g} \quad \text{s.t. } \mathbf{g}^\top \mathbf{1} = 1 \,. \tag{6.13}$$

Note that the mapping constraints (6.10), which impose that incompatible matches cannot be part of the final solution, are already incorporated in the data matrix $\mathbf{B}$. This is because we set $\mathbf{B}_{ij} = 0$ whenever γ_i and γ_j are incompatible (by Eq. (6.7)).

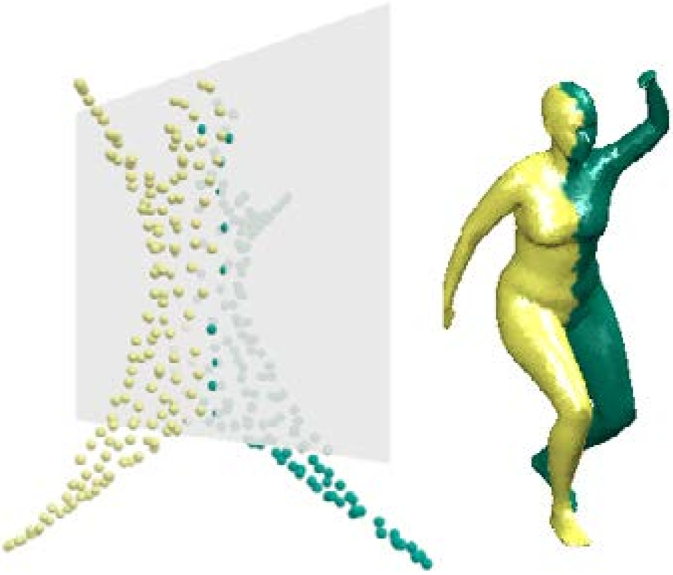


Figure 6.7: **Symmetries.** In order to favor symmetry-consistent solutions (Fig. 6.4), we assume to be given left-right maps for the shapes in the collection, associating each shape point to either side. The maps are then used to augment the shape descriptors. While there are robust approaches to perform this task [144], for our purposes it is enough to have a rough estimate so as to avoid obviously inconsistent solutions.

A problem of this form for the simple case of two shapes was previously considered in [182]. Local solutions to (6.13) (obtained again with [186]) will be accurate, although sparse. However, since the candidate set Λ is likely to contain good match hypotheses due to the previous optimization, there is hope to elicit a larger correspondence from it. To this end, we consider three simple approaches:

Grouped sparse. Following [21,187], we proceed by iteratively solving updated versions of problem (6.13). Whenever a local optimum is reached, the matches resulting from the optimizer g are stored, and the data matrix is modified by setting $\mathbf{B}_{i\star} = \mathbf{B}_{\star i} = 0$ for all i such that $\mathrm{g}_i \neq 0$. By Theorem 1, this amounts to reducing the feasible set to the remaining candidates in Λ. The iterations stop when the objective $\mathrm{g}^\top \mathbf{B}\mathrm{g}$ falls below a certain threshold.

Spectral relaxation. A different way to approach the problem consists in replacing the L^1 constraint $\mathrm{g}^\top \mathbf{1} = 1$ by a L^2 counterpart $\mathrm{g}^\top \mathrm{g} = 1$. This type of constraint acts as a Tikhonov regularizer, which tends to yield denser solutions for this kind of problems. A *global* optimum can then be computed by Rayleigh's ratio as the principal eigenvector of $\mathbf{B}$. This comes at the price of sacrificing the mapping constraints guaranteed by Theorem 1, which must be imposed by a post-processing of the obtained solution [137].

Elastic net. Finally, one may introduce a form of controllable sparsity into the problem by elastic net regularization [181]. In this case, the L^1 constraint is replaced by the convex combination $(1-\alpha)\mathrm{g}^\top \mathbf{1} + \alpha \mathrm{g}^\top \mathrm{g} = 1$, where parameter $\alpha \in [0,1]$ allows to tran-

```
Data: Shape collection $\mathcal{C}$ of $n$ shapes
Result: Partial multi-way correspondence $\Gamma$ among the shapes in $\mathcal{C}$
pre-processing (Sec. 6.1.4);
$Q \leftarrow$ generated as in the Example of Sec. 6.1.3;
$\Lambda \leftarrow \emptyset$;
forall the $q \in Q$ do
    find points $p \in S_i$ s.t. $\tau(p, q) > \xi$ for $i = 1, \dots, n$;
    construct similarity matrix $\mathbf{A}$ as in Eq. (6.12);
    $\mathbf{x} \leftarrow$ solve problem (6.11) using [186];
    $\gamma \leftarrow$ support of $\mathbf{x}$;
    update $\Lambda \leftarrow \Lambda \cup \{\gamma\}$;
end
construct similarity matrix $\mathbf{B}$ as in Sec. 6.1.3;
$\mathbf{g} \leftarrow$ solve problem (6.13) "grouped" as in Sec. 6.1.3;
$\Gamma \leftarrow$ support of $\mathbf{g}$;
```

Algorithm 1: Full pipeline of our method for consistent partial matching of shape collections.

sition smoothly from a formulation equivalent to (6.13) (hence sparse) to a purely spectral solution (denser).

In Fig. 6.8 we show a full comparison of the three alternatives on the TOSCA dataset, using the cumulative error measure defined in Sec. 6.1.4. Finally, the main steps describing our matching pipeline are given in Algorithm 1 and Fig. 6.6.

Complexity and scalability

We conclude the theoretical part with a complexity analysis of our method. Suppose our collection $\mathcal{C}$ is made of n shapes, each shape has N points, and M is the number of queries.

For a single query, computing the similarity matrix $\mathbf{A}$ takes $O((nN)^2)$ operations. In practice, since for each query we have $N' \ll N$, this is a fast operation of the order $O((nN')^2)$. Optimization of problem (6.11) using evolutionary dynamics [186] is a $O(nN')$ step. The complexity of the first sub-problem (generation of Λ) is thus $O(M \cdot (nN')^2)$.

Next, constructing matrix $\mathbf{B}$ is a $O(M^2)$ operation; this also involves computing geodesic distances among all points in Λ, which can be done efficiently via fast marching [233]. Since optimizing problem (6.13) is a $O(M)$ process, the overall complexity of the second sub-problem is $O(M^2)$. Note that in all our experiments we have once again $M \ll N$, hence this step of the pipeline is typically very fast. We refer to Sec. 6.1.4 for an experimental evaluation.

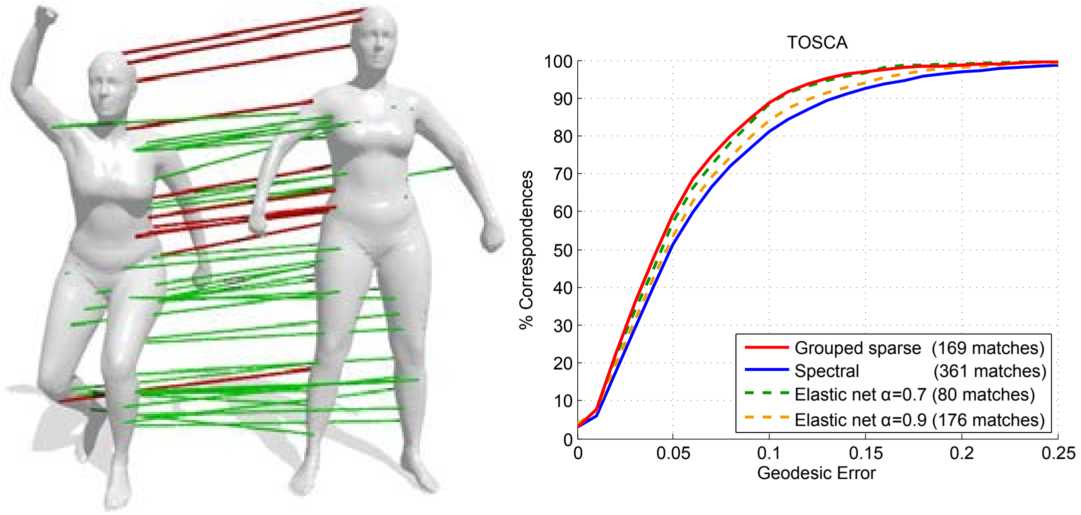


Figure 6.8: Comparison between different numerical approaches to solve problem (6.13). *Left*: Matches visualized on two scanned shapes from the SHREC'14 dataset, extracted from a multi-way correspondence of length 7; the spectral relaxation yields 43 matches (in green), while the elastic net with $\alpha = 0.7$ only 11 matches (in red), although less noisy. *Right*: Quantitative comparison on the entire TOSCA dataset; the iterated L^1 approach provides the best combination of size and accuracy.

6.1.4 Experimental results and applications

We performed a wide range of experiments on several benchmarks, namely: TOSCA [46], SCAPE [24], KIDS [184], and SHREC'14 [173]. These datasets consist of multiple classes of nearly-isometric shapes, with some intra-class variation in the case of KIDS and SHREC'14. All datasets with the exception of SHREC'14 come with ground-truth correspondences within each category. In all the experiments, we ran our matching algorithm using $M = 500$ queries in descriptor space.

Pre-processing. WKS descriptors [26] are precomputed for all the meshes. Where not specified otherwise, in our experiments we run the matching pipeline on $N = 300$ farthest samples per shape (using the Euclidean metric). This is done in order to avoid solutions that unduly aggregate in small regions. Note that we do *not* assume samples to be compatible across shapes as in [115], hence some local error in the computed matches is to be expected.

Error measure. We quantify the quality of the correspondence by using the same measure of error defined in [115]. Specifically, in our plots we show the percent of matches p_e which have geodesic error (*i.e.*, distance from the ground-truth) smaller than a threshold e. This cumulative distribution is computed and aggregated over all the pairwise matches induced by the obtained multi-way correspondences. The geodesic error is normalized by the square root of the area of each shape. As in [115], we also report values for $p_{0.16}$ and $p_{0.02}$, which respectively capture the global and local accuracy of the matching method.

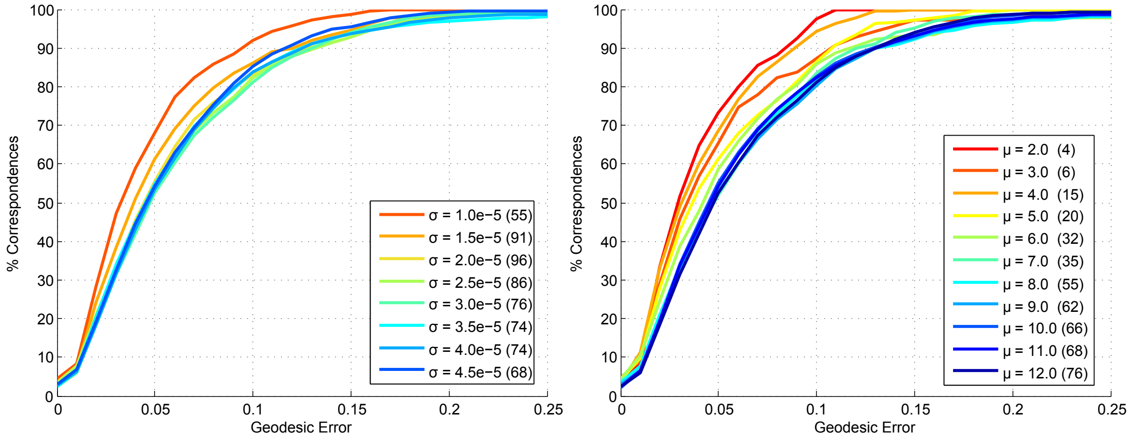


Figure 6.9: Sensitivity experiments on a subset of TOSCA. Here we plot the error curves under different choices of point-wise similarity parameter σ (left), and metric distortion parameter μ (right). In both graphs, the resulting number of multi-way matches is reported in parentheses.

Sensitivity analysis

The first set of experiments is aimed at analyzing the sensitivity of our method to different parametrizations. In order to reduce overfitting, these experiments are performed on a representative subset of the TOSCA dataset, consisting of the *victoria* (12 shapes) and *cat* (11 shapes) classes.

Point-wise similarity. We measure the similarity between points on different shapes by the similarity between their associated WKS descriptors. Each signature is computed on the shape samples using 100 eigenpairs, 100 energy levels and variance equal to 6.0 (default parameters as provided by the authors). Given two points $p \in S_k$ and $q \in S_\ell$, we define their similarity by the Gaussian weight

$$\tau(p,q) = e^{-\frac{1}{\sigma^2}\|WKS(p)-WKS(q)\|_2^2} . \tag{6.14}$$

In Fig. 6.9 (left) we plot the error curves under different choices of $\sigma \in \mathbb{R}$. Note that smaller values of σ tend to yield more accurate solutions. The choice of σ also has an effect, although not very pronounced, on the final number of multi-way matches (reported in parentheses).

Metric distortion. As described in Eq. (6.7), penalizing the metric distortion of a pair of multi-way matches is done by means of a control parameter μ. As shown in Fig. 6.9 (right), changing the value of μ allows to control the size/accuracy trade-off of the final correspondence: as μ is increased, distorted matches are tolerated and included in the solution. Further illustration of this behavior is given in Fig. 6.10, where we show how the *worst-case* metric distortion over the shape collection can be bounded by an appropriate choice of μ. The choice of this upper bound is ultimately driven by the application; *e.g.*, it makes sense in shape exploration applications (see Sec. 6.1.4) to require more accurate, although sparser matches in order to obtain a better clustering.

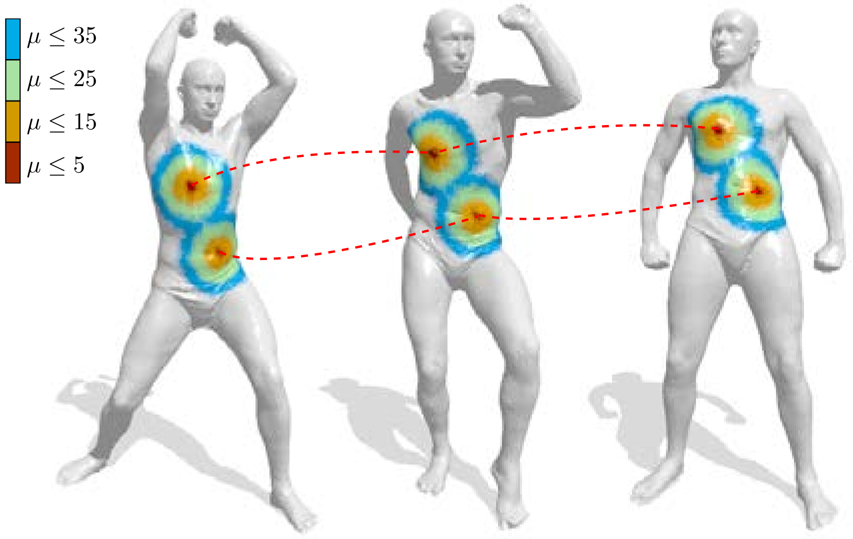


Figure 6.10: Effect of parameter μ on the metric distortion term of Eq. (6.7). Increasing the value of μ makes geometric validation more tolerant to distorted matches. In this real example on the SCAPE dataset, the colored regions show the admitted metric distortion for a pair of multi-way matches at different values of μ. An optimal value for this parameter can be chosen such that a prescribed metric distortion is not exceeded (*e.g.*, constrained to the orange area).

In a separate set of experiments, we investigate the effect of different similarity functions s. Namely, we consider both the Gaussian function of Eq. (6.7) as well as a modified version of it, given by replacing the worst-case cost of Eq. (6.3) by:

$$\epsilon^2(\gamma,\gamma') = \sum_{\substack{p_k,p_\ell\in\gamma\\ p'_k,p'_\ell\in\gamma'}} |d_k(p_k,p'_k) - d_\ell(p_\ell,p'_\ell)|^2 . \tag{6.15}$$

Following [182], we also include a relative (Lipschitz) notion of similarity in the comparison, defined as:

$$s(\gamma,\gamma') = \frac{\min_k d_k(p_k,p'_k)^\mu}{\max_k d_k(p_k,p'_k)^\mu} . \tag{6.16}$$

Since a fixed value of μ will in general scale differently in the three cases, each variant is parametrized so as to yield 30 multi-way matches on average. The results of this experiment are summarized in Table 6.1.

Comparisons

We compare our method with MatchLift, the convex relaxation approach of Chen *et al.* [60]. This method represents, to the best of our knowledge, the state of the art for this class of problems. In Fig. 6.11 we report the results on the TOSCA and KIDS datasets.

	L^∞	L^2	Lipschitz
Local ($p_{0.02}$)	**26.81**	16.01	20.04
Global ($p_{0.16}$)	**96.21**	95.49	91.05

Table 6.1: Comparison between different metric distortion measures on a subset of TOSCA. The best results (in bold) are obtained when we penalize the worst-case absolute metric error. Interestingly, there is no clear advantage in using a relative error as opposed to its absolute counterpart.

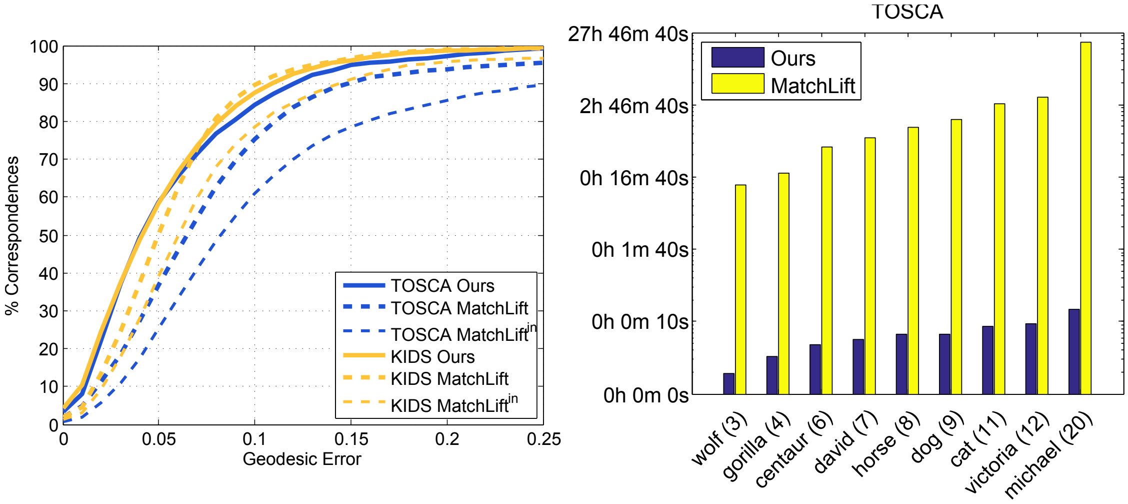


Figure 6.11: *Left*: Comparison between our method and the state-of-the-art method of [60] on the TOSCA and KIDS datasets; the quality of the input maps of MatchLift is also reported. *Right*: Comparison of execution times. In all the comparisons, the two methods generated a comparable amount of matches.

Note that MatchLift did not previously appear in these benchmarks; in the same figure we also report a runtime comparison of the two methods on collections of increasing size. Finally, in Table 6.2 we show additional comparisons with the methods described in [115] and [116] on the TOSCA and SCAPE datasets.

We remind the reader that all methods included in the comparisons, except for ours, require pairwise maps to be given as input (also evaluated in the comparisons), hence

	Ours	[115]	[115]^{in}	[116]
TOSCA ($p_{0.16}$)	97.7	100	84.1	97.2
SCAPE ($p_{0.16}$)	95.9	99.1	83.2	99.3
TOSCA ($p_{0.02}$)	21.9	35.7	-	38.4
SCAPE ($p_{0.02}$)	50.6	42.1	-	44.4

Table 6.2: Comparisons with other recent methods in terms of global ($p_{0.16}$) and local ($p_{0.02}$) accuracy. The ^{in} column reports the quality of the input maps [127].

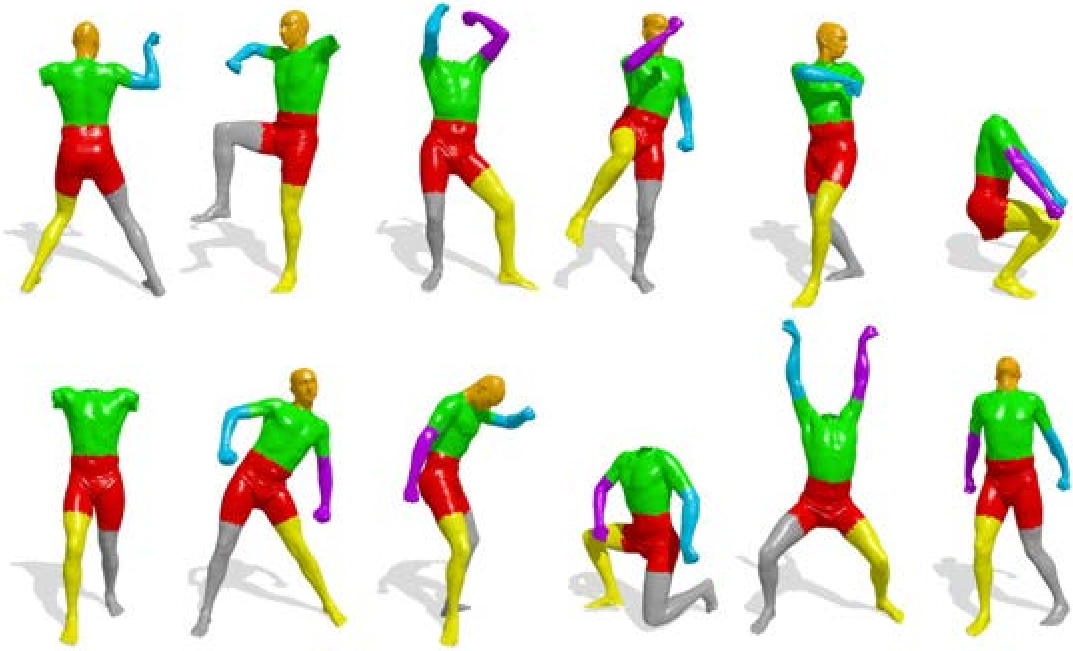


Figure 6.12: Joint region matching on SCAPE (only a subset shown). The optimization process automatically excluded shape regions having incompatible segmentations with respect to the rest of the collection (*e.g.*, two segments per arm). Our pipeline took 2 sec. to produce these results.

acting more like global regularizers rather than "pure" multiple shape matching methods. Also note that our method was not tuned to perform well in the comparisons, as our sensitivity analysis was only executed on a small subset of TOSCA.

Region matching

Our method can be trivially modified to work with region-wise rather than point-wise correspondences, assuming a (possibly noisy) segmentation is provided for the input shapes. The modification boils down to define a proper similarity measure among regions. To this end we use the simple Gaussian score of Eq. (6.14), where the cost term is replaced by the L^2 distance between the area-weighted average WKS of each region. Regions are computed by consensus segmentation [183], using the code provided by the authors.

Note that since most shapes typically contain only 5 to 15 regions, a *full* similarity matrix $\mathbf{A}$ can be constructed which encodes the pairwise similarities among all regions in the collection (*i.e.*, we do not need to define queries). We can then solve the resulting problem (6.11) iteratively, each time reducing the feasible set by removing solutions from the past iterates (this is done by putting rows and columns of $\mathbf{A}$ to zero, as per Theorem 1). In Fig. 6.12 we show some qualitative results produced by this simple procedure when applied to a noisy version of SCAPE, in which 10 random outlier shapes from TOSCA

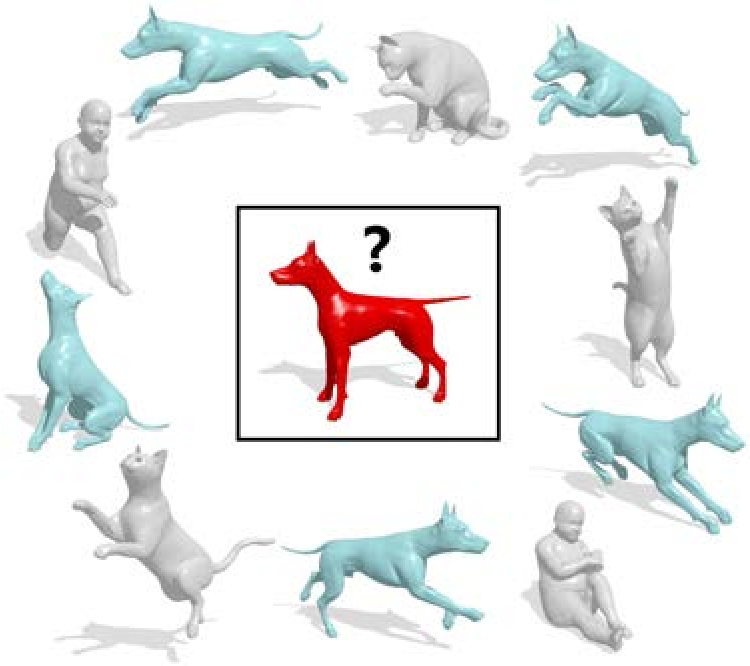


Figure 6.13: An example of shape retrieval. The query shape is matched jointly to the shapes in the database, forming a cluster with the shapes from the same class.

were introduced.

Other applications

Our approach is robust to the presence of outliers by design, and we can always extract an accurate solution as long as the outliers do not have a structure.

Consider the example in Fig. 6.1. As problem (6.11) is iteratively solved, the candidate set Λ is updated with matches that put the horse parts into correspondence, in addition to matches that only relate the human bodies. The subsequent optimization of (6.13) then extracts two intra-similar clusters of matches, one for each semantic group. In this case, it is clear that there is technically no reason to treat either of the two solutions as noise. Consider now a collection of shapes of a given class, which has been corrupted by introducing other shapes (Fig. 6.5). Since the extra-class objects fail to form stable matches with any other object in the collection, they will not appear in the final solution. This key feature of our framework suggests, among others, two applications:

Shape exploration and clustering. Consider once again the example in Fig. 6.5, and suppose both outlier shapes actually belong to the same class. This scenario can be seen as an instance of structured noise – in fact, we now have *two* semantic classes forming intra-similar groups, and it would be desirable to separate them into two disjoint clusters. We do so by a simple iterative procedure: 1) run Algorithm 1 on the whole collection; 2) relabel the resulting multi-way correspondence into clusters, based *solely* on the shape indices; 3) remove the matched shapes from the collection and repeat. Note that the clustering step is especially efficient, as it boils down to detecting connected components in a graph where each node represents a shape, and an edge exists between two nodes whenever there exist (at least 3) matches connecting the respective shapes. Running this procedure on the TOSCA dataset gives the results reported in Fig. 6.15.

Shape retrieval. The approach described above can be directly applied to shape retrieval applications. Given a query shape $S_q \notin \mathcal{C}$, the task is to detect the subset $\mathcal{C}_q \subseteq \mathcal{C}$ containing shapes that belong to the same class as S_q. This can be done by seeking a multi-way correspondence on the augmented set $\mathcal{C} \cup \{S_q\}$, and by retaining the cluster of shapes that contains S_q in the final solution (see Fig. 6.13).

Runtime

One of the key advantages of our matching method lies in its computational efficiency. In Fig. 6.14 we show a breakdown of the runtimes across the whole pipeline. Observe that the first optimization can be easily parallelized (we used 7 cores in our tests), as it amounts to solving independent instances of problem (6.11), one problem per query.

Our method takes around 1 min. 30 sec. to match the *entire* SCAPE collection when we use 300 samples per shape. Further runtime comparisons with the method of [60] are given in Fig. 6.11. All experiments were coded in Matlab/C++ and run on an Intel Core i7 4900MQ with 32GB memory, using publicly available code for [60] and for the optimization step [186].

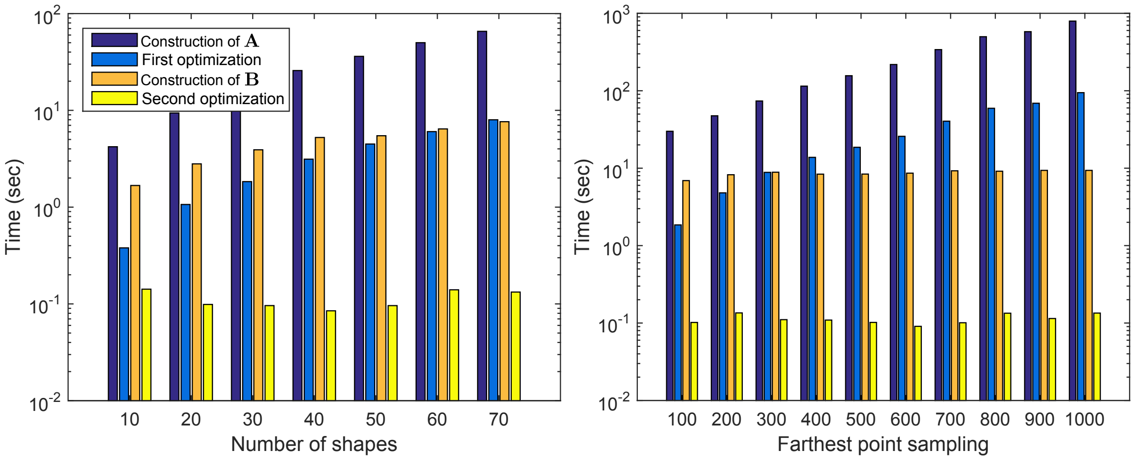


Figure 6.14: Breakdown of our computational times over the SCAPE dataset. *Left*: Runtime as a function of a subset of shapes in the collection, with 300 samples per shape. The first and second optimizations refer to solving problems (6.11) and (6.13) respectively. The runtime for the first problem is accumulated over $M = 500$ queries. *Right*: Runtime as a function of farthest samples per shape, over the entire collection (71 shapes). Note the different scales among the two graphs.

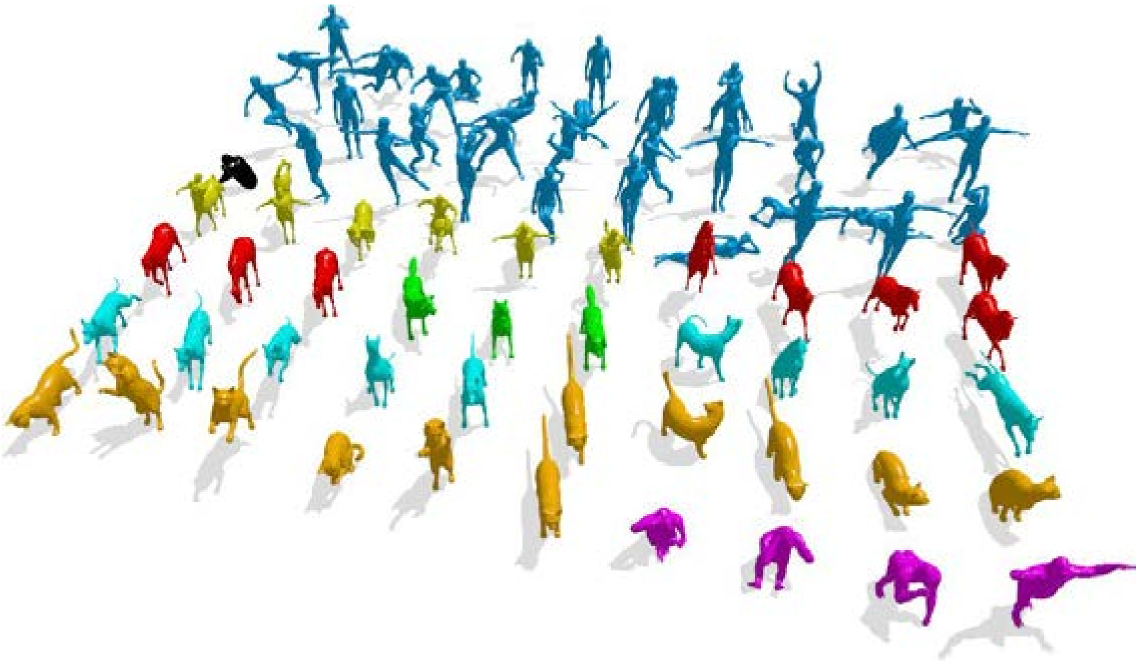


Figure 6.15: An example of shape clustering of the TOSCA dataset, obtained by running the matching algorithm followed by extraction of connected components. Classes are encoded by color; note how all humans except for one *victoria* pose (in black) have been clustered together. Total running time is around 1 min. 30 sec.

6.1.5 Limitations and future works

Our approach does have a few shortcomings. While the sparse model allows to successfully deal with partial similarity at different levels, this partiality is not easily controllable and it might well be that incomplete matches are extracted even within outlier-free collections. This is related to our necessity to establish a similarity criterion that acts globally on the whole collection, hence driving longer, but less globally-similar matches to be cut out from the solution even if correct. Enforcing specific shapes to partake in the final solution is a possible direction of future work.

6.2 Partial Functional Correspondence

In this section, we propose a method for computing partial correspondence between non-rigid shapes extending the functional correspondence framework to deal with partial correspondence. Specifically, we consider a scenario of matching a part of a deformed shape to some full model. Such scenarios are very common for instance in robotics applications, where one has to match an object acquired by means of a 3D scanner (and thus partially occluded) with a reference object known in advance. We use an explicit part model over which optimization is performed as in [45, 47], as well as a regularization on the spectral representation of the functional correspondence accounting for a special structure of the Laplacian eigenfunctions as a result of part removal. Theoretical study of this behavior based on perturbation analysis of Laplacian matrices is another contribution of our work.

The section is organized as follows. In 6.2.1, we review the basic concepts in the spectral geometry and describe the functional correspondence approach. We then study the behavior of Laplacian eigenfunctions in the case of missing parts, motivating the regularizations used in the subsequent sections in 6.2.2. Finally, before presenting the experimental results in 6.2.5, in 6.2.3 we introduces our partial correspondence model, and describes its implementation details in 6.2.4.

6.2.1 Spectral Geometry

We model shapes as compact connected two-dimensional Riemannian manifolds $\mathcal{M}$, possibly with boundary $\partial\mathcal{M}$. Given $f, g : \mathcal{M} \to \mathbb{R}$ some real scalar fields on the manifold, we define the standard inner product $\langle f, g\rangle_{\mathcal{M}} = \int_{\mathcal{M}} f(x)g(x)dx$, where integration is done using the area element induced by the Riemannian metric. We denote by $L^2(\mathcal{M}) = \{f : \mathcal{M} \to \mathbb{R} \mid \langle f, f\rangle_{\mathcal{M}} < \infty\}$ the space of square-integrable functions on $\mathcal{M}$.

The *intrinsic gradient* $\nabla_{\mathcal{M}} f$ and the positive semi-definite *Laplace-Beltrami operator* $\Delta_{\mathcal{M}} f = -\text{div}_{\mathcal{M}}(\nabla_{\mathcal{M}} f)$ generalize the notions of gradient and Laplacian to manifolds.

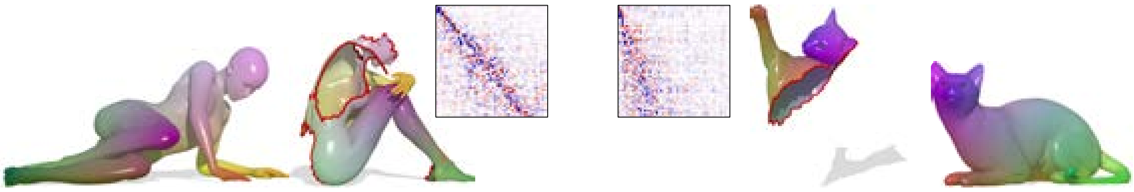


Figure 6.16: Partial functional correspondence between two pairs of shapes with large missing parts. For each pair we show the matrix $\mathbf{C}$ representing the functional map in the spectral domain, and the action of the map by transferring colors from one shape to the other. The special slanted-diagonal structure of $\mathbf{C}$ induced by the partiality transformation is first estimated from spectral properties of the two shapes, and then exploited to drive the matching process.

The Laplace-Beltrami operator admits an eigen-decomposition

$$\Delta_{\mathcal{M}}\phi_i(x) = \lambda_i\phi_i(x) \qquad x \in \mathrm{int}(\mathcal{M}) \tag{6.17}$$

$$\langle \nabla_{\mathcal{M}}\phi_i(x), \hat{n}(x)\rangle = 0 \qquad x \in \partial\mathcal{M}, \tag{6.18}$$

with homogeneous Neumann boundary conditions (6.18) if $\mathcal{M}$ has a boundary (here $\hat{n}$ denotes the normal vector to the boundary), where $0 = \lambda_1 < \lambda_2 \leq \ldots$ are eigenvalues and $\phi_1, \phi_2, \ldots$ are the corresponding eigenfunctions (or eigenvectors). The eigenfunctions form an orthonormal basis on $L^2(\mathcal{M})$, *i.e.*, $\langle\phi_i, \phi_j\rangle_{\mathcal{M}} = \delta_{ij}$, generalizing the classical Fourier analysis: a function $f \in L^2(\mathcal{M})$ can be expanded into the *Fourier series* as

$$f(x) \;=\; \sum_{i\geq 1}\langle f, \phi_i\rangle_{\mathcal{M}}\phi_i(x)\,. \tag{6.19}$$

Functional correspondence. Let us be now given two manifolds, $\mathcal{N}$ and $\mathcal{M}$. Ovsjanikov *et al.* [171]roposed modeling *functional correspondence* between shapes as a linear operator $T : L^2(\mathcal{N}) \to L^2(\mathcal{M})$. One can easily see that classical vertex-wise correspondence is a particular setting where T maps delta-functions to delta-functions.

Assuming to be given two orthonormal bases $\{\phi_i\}_{i\geq 1}$ and $\{\psi_i\}_{i\geq 1}$ on $L^2(\mathcal{N})$ and $L^2(\mathcal{M})$ respectively, the functional correspondence can be expressed w.r.t. to these bases as follows:

$$\begin{aligned} Tf &= T\sum_{i\geq 1}\langle f, \phi_i\rangle_{\mathcal{N}}\phi_i = \sum_{i\geq 1}\langle f, \phi_i\rangle_{\mathcal{N}}T\phi_i \\ &= \sum_{ij\geq 1}\langle f, \phi_i\rangle_{\mathcal{N}}\underbrace{\langle T\phi_i, \psi_j\rangle_{\mathcal{M}}}_{c_{ij}}\psi_j\,, \end{aligned} \tag{6.20}$$

Thus, T amounts to a linear transformation of the Fourier coefficients of f from basis $\{\phi_i\}_{i\geq 1}$ to basis $\{\psi_i\}_{i\geq 1}$, which is captured by the coefficients c_{ij}. Truncating the Fourier series (6.20) at the first k coefficients, one obtains a rank-k approximation of T, represented in the bases $\{\phi_i, \psi_i\}_{i\geq 1}$ as a $k \times k$ matrix $\mathbf{C} = (c_{ij})$.

In order to compute $\mathbf{C}$, Ovsjanikov *et al.* [171]ssume to be given a set of q corresponding functions $\{f_1, \ldots, f_q\} \subseteq L^2(\mathcal{N})$ and $\{g_1, \ldots, g_q\} \subseteq L^2(\mathcal{M})$. Denoting by $a_{ij} = \langle f_j, \phi_i\rangle_{\mathcal{N}}$ and $b_{ij} = \langle g_j, \psi_i\rangle_{\mathcal{M}}$ the $k \times q$ matrices of the respective Fourier coefficients, functional correspondence boils down to the linear system

$$\mathbf{CA} \;=\; \mathbf{B}\,. \tag{6.21}$$

If $q \geq k$, the system (6.21) is (over-)determined and is solved in the least squares sense to find $\mathbf{C}$.

Structure of C. We note that the coefficients $\mathbf{C}$ depend on the choice of the bases. In particular, it is convenient to use the eigenfunctions of the Laplace-Beltrami operators of

$\mathcal{N}$ and $\mathcal{M}$ as the bases $\{\phi_i, \psi_i\}_{i\geq 1}$; truncating the series at the first k coefficients has the effect of 'low-pass' filtering thus producing smooth correspondences. In the following, this will be our tacit basis choice.

Furthermore, note that the system (6.21) has qk equations and k^2 variables. However, in many situations the actual number of variables is significantly smaller, as $\mathbf{C}$ manifests a certain structure which can be taken advantage of. In particular, if $\mathcal{N}$ and $\mathcal{M}$ are isometric and have simple spectrum (*i.e.*, the Laplace-Beltrami eigenvalues have no multiplicity), then $T\phi_i = \pm\psi_i$, or in other words, $c_{ij} = \pm\delta_{ij}$. In more realistic scenarios (approximately isometric shapes), the matrix $\mathbf{C}$ would manifest a funnel-shaped structure, with the majority of elements distant from the diagonal close to zero.

Discretization. In the discrete setting, the manifold $\mathcal{N}$ is sampled at n points $x_1, \ldots, x_n$ which are connected by edges $E = E_\mathrm{i} \cup E_\mathrm{b}$ and faces F, forming a manifold triangular mesh (V, E, F). We denote by E_i and E_b the interior and boundary edges respectively. A function on the manifold is represented by an n-dimensional vector $\mathbf{f} = (f(x_1), \ldots, f(x_n))^\top$. The discretization of the Laplacian takes the form of an $n \times n$ sparse matrix $\mathbf{L} = -\mathbf{S}^{-1}\mathbf{W}$ using the classical cotangent formula [84, 146, 176],

$$w_{ij} = \begin{cases} (\cot\alpha_{ij} + \cot\beta_{ij})/2 & ij \in E_\mathrm{i}; \\ (\cot\alpha_{ij})/2 & ij \in E_\mathrm{b}; \\ -\sum_{k\neq i} w_{ik} & i = j; \\ 0 & \text{else}; \end{cases} \tag{6.22}$$

where $\mathbf{S} = \mathrm{diag}(s_1, \ldots, s_n)$, $s_i = \frac{1}{3}\sum_{jk:ijk\in F} s_{ijk}$ denotes the local area element at vertex i, s_{ijk} denotes the area of triangle ijk, and α_{ij}, β_{ij} denote the angles $\angle ikj, \angle jhi$ of the triangles sharing the edge ij (see Fig. 6.17).

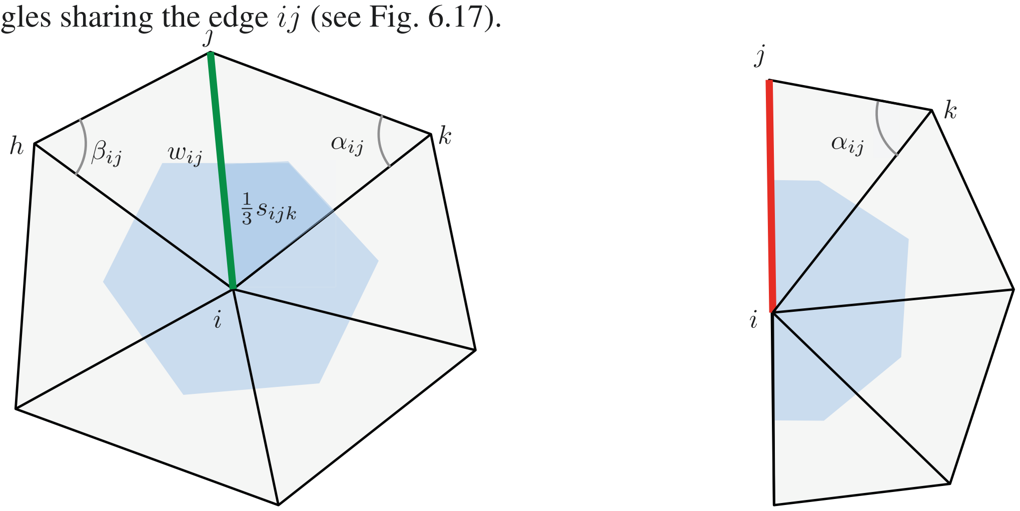


Figure 6.17: Discretization of the Laplace-Beltrami operator on a triangular mesh for interior edges (green, left) and boundary edges (red, right).

The first k eigenfunctions and eigenvalues of the Laplacian are computed by performing the generalized eigen-decomposition $\mathbf{W\Phi} = \mathbf{S\Phi\Lambda}$, where $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_k)$ is an

$n \times k$ matrix containing as columns the discretized eigenfunctions and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$ is the diagonal matrix of the corresponding eigenvalues. The computation of Fourier coefficients is performed by $\mathbf{a} = \mathbf{\Phi}^\top \mathbf{S}\mathbf{f}$.

6.2.2 Laplacian eigenvectors and eigenvalues under partiality

When one of the two shapes has missing parts, the assumption of approximate isometry does not hold anymore and a direct application of the method of Ovsjanikov *et al.* (*i.e.*, solving system (6.21)) would not produce meaningful results. However, as we show in this section, the matrix $\mathbf{C}$ still exhibits a particular structure which can be exploited to drive the matching process.

We assume to be given a full shape $\mathcal{M}$ and a part thereof $\mathcal{N} \subset \mathcal{M}$. We further denote by $\overline{\mathcal{N}} = \mathcal{M} \setminus \mathcal{N}$ the remaining vertices of $\mathcal{M}$. The manifolds $\mathcal{M}$ and $\mathcal{N}$ are discretized as triangular meshes with m and n vertices, respectively, and $\bar{n} = m - n$. The scenario we consider in concerns the problem of matching an approximately isometric deformation of part $\mathcal{N}$ to the full shape $\mathcal{M}$ (part-to-whole matching). Our goal is to characterize the eigenvalues and eigenvectors of the Laplacian $\mathbf{L}_\mathcal{M}$ in terms of perturbations of the eigenvalues and eigenvectors of the Laplacians $\mathbf{L}_\mathcal{N}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$ [159]. We tacitly assume that homogeneous Neumann boundary conditions (6.18) apply.

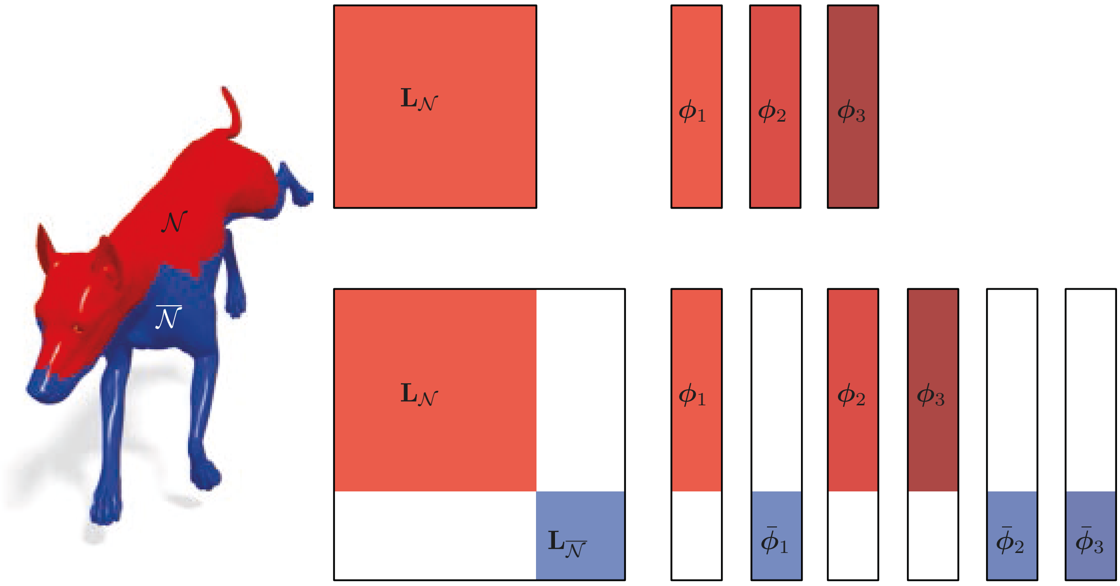


Figure 6.18: The eigenvalues and eigenvectors of a block-diagonal Laplacian $\mathbf{L}_\mathcal{M}$ are an interleaved sequence of the eigenpairs from the two blocks $\mathbf{L}_\mathcal{N}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$.

Block-diagonal case

For the simplicity of analysis, let us first consider a simplified scenario in which $\mathcal{N}$ and $\overline{\mathcal{N}}$ are *disconnected*, *i.e.*, there exist no links between the respective boundaries $\partial\mathcal{N}$ and $\partial\overline{\mathcal{N}}$. W.l.o.g., we can assume that the vertices in $\mathcal{M}$ are ordered such that the vertices in $\mathcal{N}$ come before those in $\overline{\mathcal{N}}$. With this ordering, the $m \times m$ Laplacian matrix $\mathbf{L}_{\mathcal{M}}$ is block-diagonal, with an $n \times n$ block $\mathbf{L}_{\mathcal{N}}$ and an $\bar{n} \times \bar{n}$ block $\mathbf{L}_{\overline{\mathcal{N}}}$. The (sorted) eigenvalues of $\mathbf{L}_{\mathcal{M}}$ form a mixed sequence composed of the eigenvalues from $\mathbf{L}_{\mathcal{N}}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$. Similarly, the eigenvectors of $\mathbf{L}_{\mathcal{M}}$ correspond to the eigenvectors of the two sub-matrices, zero-padded to the correct size (Fig. 6.18).

Structure of C under partiality. Suppose we are given the first k Laplace-Beltrami eigenvalues of the full shape $\mathcal{M}$ and its part $\mathcal{N}$. Since the spectrum of $\mathbf{L}_{\mathcal{M}}$ is an interleaved sequence of the eigenvalues of $\mathbf{L}_{\mathcal{N}}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$, only the first $r < k$ eigenvalues of $\mathbf{L}_{\mathcal{N}}$ will appear among the first k eigenvalues of $\mathbf{L}_{\mathcal{M}}$. The remaining $k - r$ eigenvalues of $\mathbf{L}_{\mathcal{N}}$ will only appear further along the spectrum of $\mathbf{L}_{\mathcal{M}}$ (see Fig. 6.21 for an example where $k = 50$ and $r = 21$). The same argument holds for the associated eigenfunctions, as illustrated in Fig. 6.19: if $\boldsymbol{\phi}_i$ is a (restricted) eigenvector of $\mathbf{L}_{\mathcal{N}}$, then $\mathbf{L}_{\mathcal{M}}$ also has an eigenvector $\boldsymbol{\psi}_j$ such that $\boldsymbol{\phi}_i = \mathbf{T}\boldsymbol{\psi}_j$, where $\mathbf{T} = (\mathbf{I}_{n\times n}, \mathbf{0})^\top$ and $i < j$.

This analysis leads us to the following simple observation: the partial functional map between $\mathcal{N}$ and $\mathcal{M}$ is represented in the spectral domain by the matrix of inner products $c_{ij} = \langle \mathbf{T}\boldsymbol{\phi}_i, \boldsymbol{\psi}_j \rangle_{\mathcal{M}}$, which has a *slanted-diagonal* structure with a slope r/k (see examples in Figs. 6.16, 6.19 where this structure is manifested approximately). Consequently, the last $k - r$ columns of matrix $\mathbf{C}$ are zero such that $r = \mathrm{rank}(\mathbf{C})$. The value r can be estimated by simply comparing the spectra of the two shapes, as shown in Fig. 6.21. Note that this behavior is consistent with Weyl's asymptotic law [235], according to which the Laplacian eigenvalues grow linearly, with rate inversely proportional to surface area.

Perturbation analysis

We will now show that these properties still approximately hold when the Laplacian matrix $\mathbf{L}_{\mathcal{M}}$ is not perfectly block-diagonal, *i.e.*, when $\mathcal{N}$ and $\overline{\mathcal{N}}$ are joined along their boundaries. Roughly speaking, the main observation is that in this case as well the matrix $\mathbf{C}$ has a slanted diagonal structure, where the diagonal angle depends on the relative area of the part, and the diagonal 'sharpness' depends on the position and length of the cut.

We assume w.l.o.g. that within $\mathcal{N}$ the boundary vertices $\partial\mathcal{N}$ are indexed at the end, while within $\overline{\mathcal{N}}$ the boundary vertices $\partial\overline{\mathcal{N}}$ are indexed at the beginning. Then, there is a boundary band $\mathcal{B} = \partial\mathcal{N} \cup \partial\overline{\mathcal{N}}$ such that only the entries of the Laplacians $\mathbf{L}_{\mathcal{N}}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$ between vertices in $\mathcal{B}$ are affected by the cut (Fig. 6.20).

We define the parametric matrix

$$\mathbf{L}(t) = \left(\begin{array}{c|c} \mathbf{L}_{\mathcal{N}} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{L}_{\overline{\mathcal{N}}} \end{array}\right) + t \left(\begin{array}{c|c} \mathbf{P}_{\mathcal{N}} & \mathbf{P} \\ \hline \mathbf{P}^\top & \mathbf{P}_{\overline{\mathcal{N}}} \end{array}\right), \tag{6.23}$$

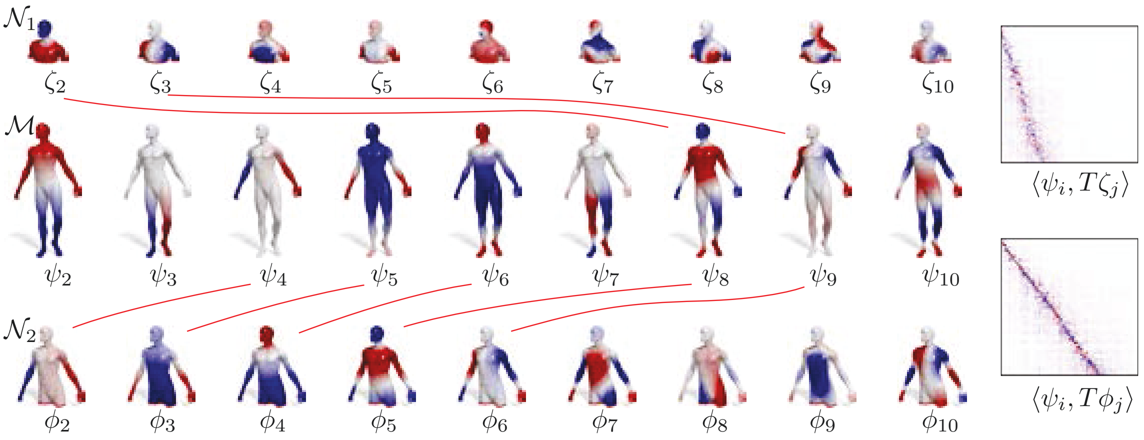

$\mathcal{N}_1$
ζ_2 ζ_3 ζ_4 ζ_5 ζ_6 ζ_7 ζ_8 ζ_9 ζ_{10}
$\mathcal{M}$
ψ_2 ψ_3 ψ_4 ψ_5 ψ_6 ψ_7 ψ_8 ψ_9 ψ_{10}
$\mathcal{N}_2$
ϕ_2 ϕ_3 ϕ_4 ϕ_5 ϕ_6 ϕ_7 ϕ_8 ϕ_9 ϕ_{10}
$\langle \psi_i, T\zeta_j \rangle$
$\langle \psi_i, T\phi_j \rangle$

Figure 6.19: First ten eigenfunctions of a full shape $\mathcal{M}$ and two parts $\mathcal{N}_1, \mathcal{N}_2 \subset \mathcal{M}$ with different surface area. All eigenfunctions of the partial shapes have a corresponding eigenfunction ψ_i on the full shape for some i; the correspondence between eigenfunctions follows from the correspondence between eigenvalues (see also Fig. 6.21). This is reflected in functional maps with different diagonal slopes, where the slope depends on the area ratios of the two surfaces (by Weyl's law).

where

$$\mathbf{P}_{\mathcal{N}} = \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{D}_{\mathcal{N}} \end{pmatrix}, \quad \mathbf{P}_{\overline{\mathcal{N}}} = \begin{pmatrix} \mathbf{D}_{\overline{\mathcal{N}}} & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathbf{P} = \begin{pmatrix} 0 & 0 \\ \mathbf{E} & 0 \end{pmatrix}$$

are matrices of size $n \times n$, $\bar{n} \times \bar{n}$, and $n \times \bar{n}$, respectively. Here $\mathbf{D}_{\mathcal{N}}$ and $\mathbf{D}_{\overline{\mathcal{N}}}$ represent the variations of the Laplacians $\mathbf{L}_{\mathcal{N}}$ and $\mathbf{L}_{\overline{\mathcal{N}}}$ *within* nodes in $\partial\mathcal{N}$ and $\partial\overline{\mathcal{N}}$ respectively, while $\mathbf{E}$ represents the variations *across* the boundary. The parameter t is such that $\mathbf{L}(1) = \mathbf{L}_{\mathcal{M}}$,

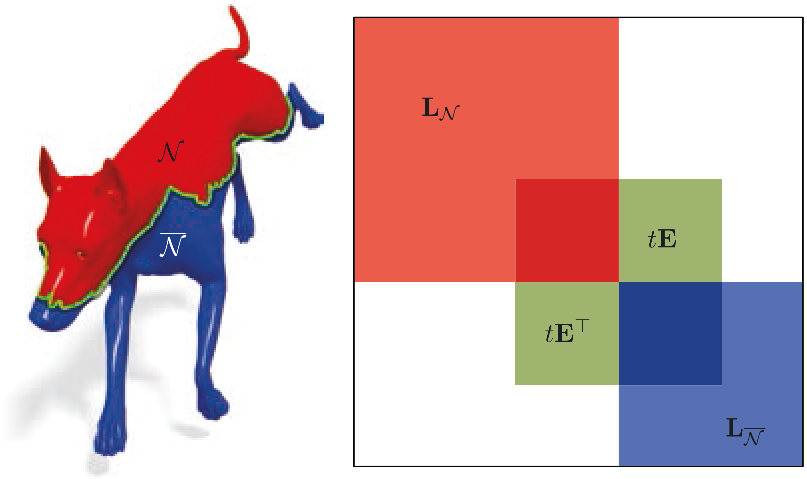


Figure 6.20: The matrix $\mathbf{L}(t)$ is obtained as a perturbation of the block-diagonal Laplacian in the boundary band (shown in green).

while for $t = 0$ we get back to the disconnected case of Fig. 6.18.

Also note that with the appropriate ordering of the vertices, the matrices $\mathbf{D}_{\mathcal{N}}$ and $\mathbf{D}_{\overline{\mathcal{N}}}$ will have a band-diagonal structure. In fact, a cut through an edge will affect the values of the discrete Laplacian matrix $\mathbf{L}$ only at the entries corresponding to the vertices at the extremities of the edge, and to the edges laying in the same triangle as the cut edge. For example, looking at Fig. 6.17, a cut through edge (i, j) will affect the diagonal entries l_{ii} and l_{jj} as well as the off-diagonal entries l_{ih}, l_{ik}, l_{jh}, and l_{jk}. Note also that the continuity of the cut implies that two of the four off-diagonal entries will be cut as well, leaving no more than two affected edges on any side of the cut. As a result, the entries of the Laplacian affected by the cut correspond to the nodes and edges in a path along the boundary of the cut.

Theorem 2. *Let $\mathbf{L}_{\mathcal{N}} + t\mathbf{P}_{\mathcal{N}} = \mathbf{\Phi}(t)^\top \mathbf{\Lambda}(t) \mathbf{\Phi}(t)$, where $\mathbf{\Lambda}(t) = \mathrm{diag}(\lambda_1(t), \ldots, \lambda_n(t))$ is a diagonal matrix of eigenvalues, and $\mathbf{\Phi}(t)$ are the corresponding eigenvectors. The derivative of the non-trivial eigenvalues is given by*

$$\frac{d}{dt}\lambda_i = \sum_{v,w \in \partial\mathcal{N}} (\mathbf{P}_{\mathcal{N}})_{vw} \phi_{iv} \phi_{iw} = \boldsymbol{\phi}_i^\top \mathbf{P}_{\mathcal{N}} \boldsymbol{\phi}_i. \tag{6.24}$$

Theorem 2 establishes that the (first-order) change in the eigenvalues of the partial shape $\mathcal{N}$ only depends on the change in the Dirichlet energy of the corresponding eigenvectors along the boundary $\partial\mathcal{N}$. In other words, the eigenvalues are perturbed depending on the *length* and *position* of the cut. By virtue of this result, we can establish approximate correspondence between the eigenvalues of $\mathbf{L}_{\mathcal{N}}$ and a subset of the eigenvalues of $\mathbf{L}_{\mathcal{M}}$ (which are now not exactly equal as in the block-diagonal case), required to estimate the slope of $\mathbf{C}$ (see Fig. 6.21).

Theorem 3. *Assume that $\mathbf{L}_{\mathcal{N}}$ has distinct eigenvalues ($\lambda_i \neq \lambda_j$ for $i \neq j$), and furthermore, the non-zero eigenvalues are all distinct from the eigenvalues of $\mathbf{L}_{\overline{\mathcal{N}}}$ ($\lambda_i \neq \overline{\lambda}_j$ for all i, j). Let $\mathbf{L}_{\mathcal{N}} + t\mathbf{P}_{\mathcal{N}} = \mathbf{\Phi}(t)^\top \mathbf{\Lambda}(t) \mathbf{\Phi}(t)$, where $\mathbf{\Lambda}(t) = \mathrm{diag}(\lambda_1(t), \ldots, \lambda_n(t))$ is a diagonal matrix of eigenvalues, and $\mathbf{\Phi}(t)$ are the corresponding eigenvectors. Then, the derivative of the non-constant eigenvector is given by*

$$\frac{d}{dt}\boldsymbol{\phi}_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{\boldsymbol{\phi}_i^\top \mathbf{P}_{\mathcal{N}} \boldsymbol{\phi}_j}{\lambda_i - \lambda_j} \boldsymbol{\phi}_j + \sum_{j=1}^{\overline{n}} \frac{\boldsymbol{\phi}_i^\top \mathbf{P}\ \overline{\boldsymbol{\phi}}_j}{\lambda_i - \overline{\lambda}_j} \overline{\boldsymbol{\phi}}_j\,. \tag{6.25}$$

Remark. If $\mathbf{L}_{\overline{\mathcal{N}}}$ shares some eigenvalues with $\mathbf{L}_{\mathcal{N}}$, the second sum in (6.25) would be slightly different [159], but would still only have support over $\overline{\mathcal{N}}$.

We conclude from Theorem 3 that the perturbation associated with the partiality transformation gives rise to a mixing of eigenspaces. The second summation in (6.25) has support over $\overline{\mathcal{N}}$ and thus provides the completion of the eigenfunction on the missing part. The first summation in (6.25) is responsible for the modifications of the eigenvectors over the nodes in $\mathcal{N}$. Here the numerator has a term $\boldsymbol{\phi}_i^\top \mathbf{P}_{\mathcal{N}} \boldsymbol{\phi}_j$ which, since $\mathbf{D}_{\mathcal{N}}$ is

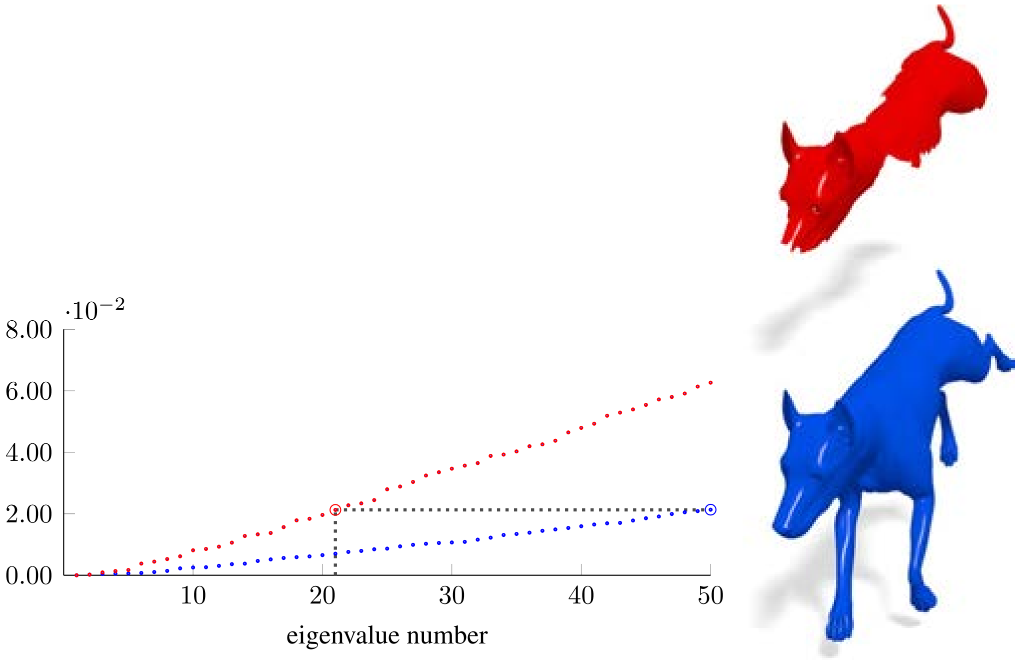


Figure 6.21: Neumann spectra of a full shape and a part of it. The eigenvalues of the partial shape (in red) are approximately preserved under the partiality transformation (see Theorem 2), and appear perturbed in the spectrum of the full shape (in blue). This simple observation allows us to estimate the diagonal slope of the functional map relating the two shapes; in this example, the slope is equal to $21/50$.

band-diagonal and diagonally dominant, acts as a dot product of the eigenvectors over the boundary band. This points to large mixing of eigenvectors with a strong co-presence near the boundary. In turn, the term $\lambda_i - \lambda_j$ at the denominator forces a strong mixing of eigenvectors corresponding to similar eigenvalues. This results in an amplification of the variation for higher eigenvalues, as eigenvalues tend to densify on the higher end of the spectrum, and explains the funnel-shaped spread of the matrix $\mathbf{C}$ visible at high frequencies (see Fig. 6.23).

Similarly to the case of eigenvalues, the eigenvectors are also perturbed depending on the length and position of the cut. The variation of the eigenvectors due to the mixing within the partial shape can be reduced either by shortening the boundary of the cut, or by reducing the strength of the boundary interaction. The latter can be achieved by selecting a boundary along which eigenvectors with similar eigenvalues are either orthogonal, or both small. The *boundary interaction strength* can be quantified by considering the following function:

$$f(v) = \sum_{\substack{i,j=1 \\ j\neq i}}^{n} \left(\frac{\phi_{iv}\phi_{jv}}{\lambda_i - \lambda_j} \right)^2 . \tag{6.26}$$

Fig. 6.22 shows an example of two different cuts with different interaction strengths,

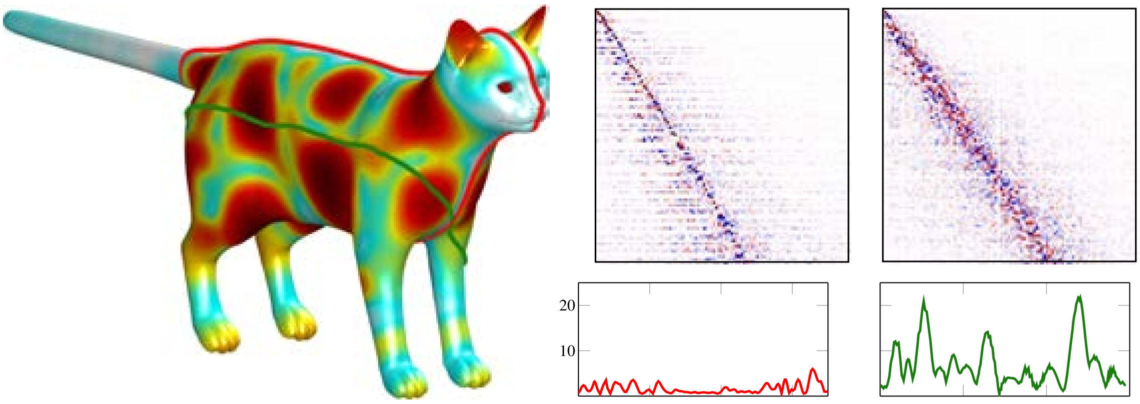


Figure 6.22: *Left*: A model is cut in two different ways (red and green curves) with cuts of same length. The off-diagonal dispersion depends mainly on the position of each cut. Function f (6.26) is plotted over the model. *Middle*: Ground-truth functional map between the complete model and the partial shape produced by the red cut (top), and values of f along the cut (bottom). *Right*: Plots associated to the green cut.

where the function f is plotted on top of the cat model. The cuts plotted in the figure have the same length, but one cut goes along a symmetry axis of the shape and through low values of f, while the other goes through rather high values of f. This is manifested in the dispersion of the slanted diagonal structure of the matrix $\mathbf{C}$ (larger in the second case).

6.2.3 Partial functional maps

As stated before, throughout this section we consider the setting where we are given a full model shape $\mathcal{M}$ and another query shape $\mathcal{N}$ that corresponds to an approximately isometrically deformed part $\mathcal{M}' \subset \mathcal{M}$.

Following [47], we model the part $\mathcal{M}'$ by means of an indicator function $v : \mathcal{M} \to \{0, 1\}$ such that $v(x) = 1$ if $x \in \mathcal{M}'$ and zero otherwise. Assuming that v is known, the *partial functional correspondence* between $\mathcal{N}$ and $\mathcal{M}$ can be expressed as $Tf = vg$, where v can be regarded as a kind of mask, and anything outside the region where $v = 1$ should be ignored. Expressed w.r.t. bases $\{\phi_i\}_{i\geq 1}$ and $\{\psi_i\}_{i\geq 1}$, the partial functional correspondence takes the form $\mathbf{CA} = \mathbf{B}(v)$, where $\mathbf{B}(v)$ denotes a matrix of weighted inner products with elements given by $b_{ij}(v) = \int_{\mathcal{M}} v(x)\psi_i(x)g_j(x)dx$ (when $v(x) \equiv 1$, $\mathbf{B}$ is simply the matrix of Fourier coefficients defined in (6.20)).

This brings us to the problem we are considering, involving optimization w.r.t. correspondence (encoded by the coefficients $\mathbf{C}$) and the part v,

$$\min_{\mathbf{C},v} \|\mathbf{CA} - \mathbf{B}(\eta(v))\|_{2,1} + \rho_{\text{corr}}(\mathbf{C}) + \rho_{\text{part}}(v)\,, \tag{6.27}$$

where $\eta(t) = \frac{1}{2}(\tanh(2t-1)+1)$ saturates the part indicator function between zero and one (see below). Here ρ_{corr} and ρ_{part} denote regularization terms for the correspondence and the part, respectively; these terms are explained below. We use the $L_{2,1}$ matrix norm (equal to the sum of L_2-norms of matrix columns) to handle possible outliers in the corresponding data, as such a norm promotes column-sparse matrices. A similar norm was adopted in [114] to handle spurious maps in shape collections.

Note that in order to avoid a combinatorial optimization over binary-valued v, we use a continuous v with values in the range $(-\infty, +\infty)$, saturated by the non-linearity η. This way, $\eta(v)$ becomes a soft membership function with values in the range $[0, 1]$.

Part regularization. Similarly to [47, 177], we try to find the part with area closest to that of the query and with shortest boundary. This can be expressed as

$$\begin{aligned}\rho_{\mathrm{part}}(v) &= \mu_1 \left(\mathrm{area}(\mathcal{N}) - \int_{\mathcal{M}} \eta(v) dx \right)^2 \\ &+ \mu_2 \int_{\mathcal{M}} \xi(v) \|\nabla_{\mathcal{M}} v\| dx \,,\end{aligned} \tag{6.28}$$

where $\xi(t) \approx \delta\left(\eta(t) - \frac{1}{2}\right)$ and the norm is on the tangent space. The μ_2-term in (6.28) is an intrinsic version of the *Mumford-Shah functional* [158], measuring the length of the boundary of a part represented by a (soft) membership function. This functional was used previously in image segmentation applications [225].

Correspondence regularization. For the correspondence, we use the penalty

$$\begin{aligned}\rho_{\mathrm{corr}}(\mathbf{C}) &= \mu_3 \|\mathbf{C} \circ \mathbf{W}\|_{\mathrm{F}}^2 + \mu_4 \sum_{i \neq j} (\mathbf{C}^\top \mathbf{C})_{ij}^2 \\ &+ \mu_5 \sum_i ((\mathbf{C}^\top \mathbf{C})_{ii} - d_i)^2 \,,\end{aligned} \tag{6.29}$$

where $\circ$ denotes Hadamard (element-wise) matrix product. The μ_3-term models the special slanted-diagonal structure of $\mathbf{C}$ that we observe in partial matching problems (see Fig. 6.23); the theoretical motivation for this behavior was presented in Sec. 6.2.2. Here, $\mathbf{W}$ is a weight matrix with zeros along the slanted diagonal and large values outside (see Fig. 6.23).

The μ_4-term promotes orthogonality of $\mathbf{C}$ by penalizing the off-diagonal elements of $\mathbf{C}^\top\mathbf{C}$. The reason is that for isometric shapes, the functional map is volume-preserving, and this is manifested in orthogonal $\mathbf{C}$ [171]. Note that differently from the classical case (*i.e.*, full shapes), in our setting we can only require area preservation going in the direction from partial to complete model, as also expressed by the μ_1-term in (6.28). For this reason, we do not impose any restrictions on $\mathbf{C}\mathbf{C}^\top$ and we say that the matrix is *semi*-orthogonal.

Finally, note that due to the low-rank nature of $\mathbf{C}$ we can not expect the product $\mathbf{C}^\top\mathbf{C}$ to be full rank. Indeed, we expect elements off the slanted diagonal of $\mathbf{C}$ to be close to zero and thus $\mathbf{C}^\top\mathbf{C} \approx \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$. The μ_5-term in (6.29) models this behavior, where vector $\mathbf{d} = (d_1, \ldots, d_k)$ determines how many singular values of $\mathbf{C}$ are non-zero.

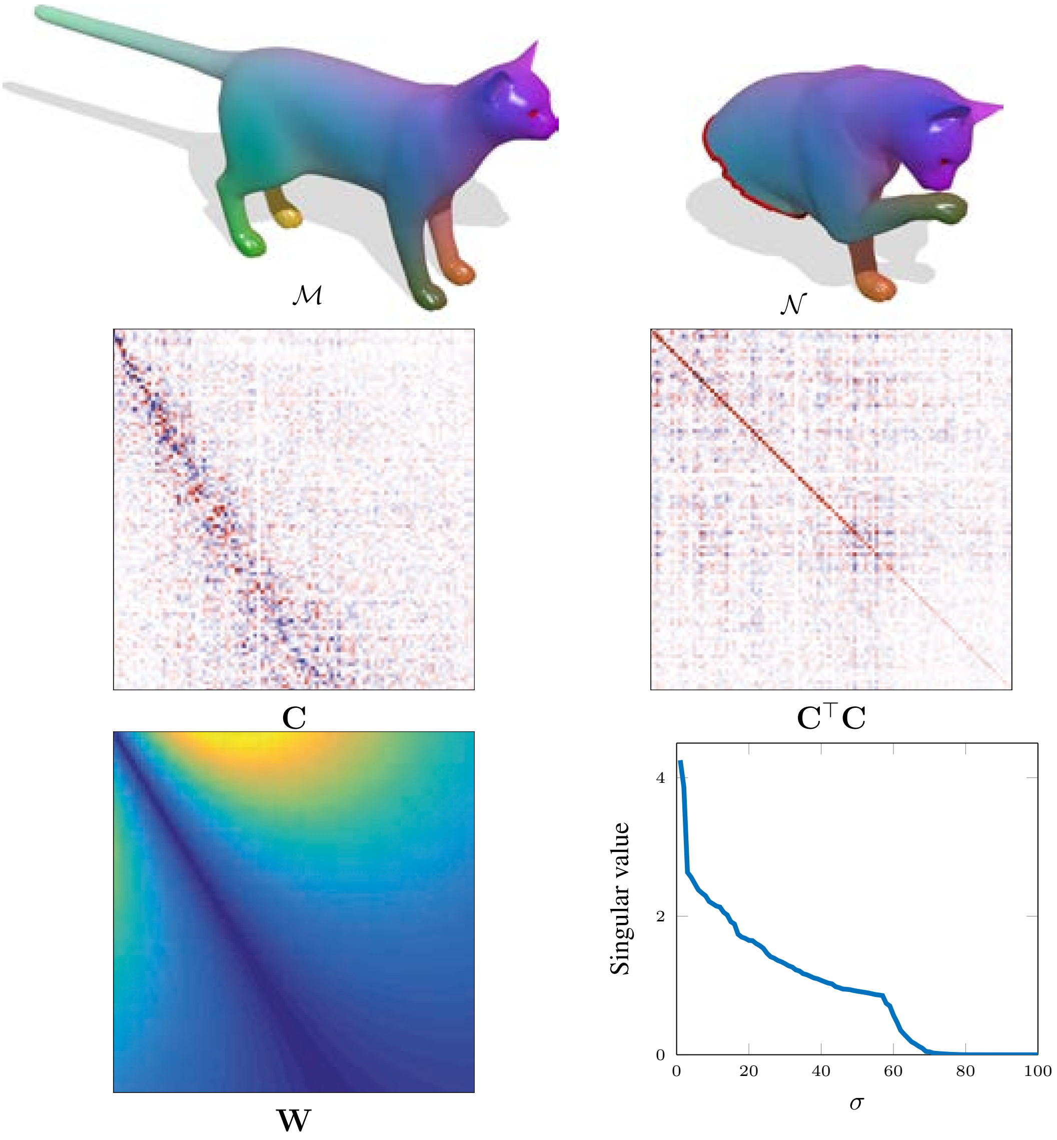


Figure 6.23: *Left*: A partial functional map $\mathbf{C}$ from $\mathcal{N}$ to $\mathcal{M}$ is low-rank and manifests a slanted-diagonal structure. *Right*: If the map is volume-preserving, then its full-rank sub-matrix is semi-orthogonal. Observe the trail of small values along the diagonal of $\mathbf{C}^\top\mathbf{C}$.

Remark. The fact that matrix $\mathbf{C}$ is low-rank is a direct consequence of partiality. This can be understood by recalling from Eq. (6.20) that the (non-truncated) functional map representation amounts to an orthogonal change of basis; since in the standard basis the correspondence matrix is low-rank (as it contains zero-sum rows), this property is preserved by the change of basis.

In Fig. 6.23 we show an example of a ground-truth partial functional map $\mathbf{C}$, illustrating its main properties.

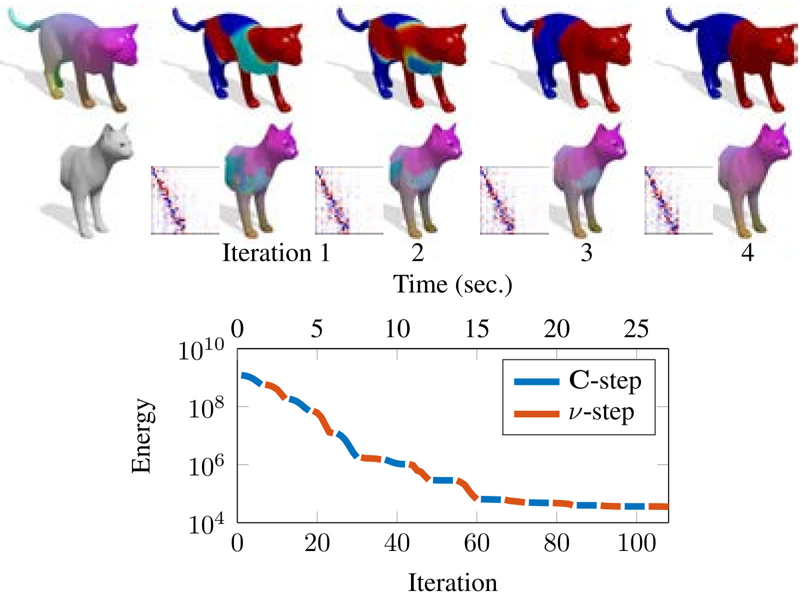


Figure 6.24: An example of the matching process operating on two shapes from TOSCA. The algorithm alternatingly optimizes over corresponding part (top row) and functional correspondence (bottom row). Corresponding points between full and partial shape are shown with the same color. This solution was obtained by using 30 eigenfunctions on both manifolds.

Alternating scheme. To solve the optimization problem (6.27), we perform an alternating optimization w.r.t. to $\mathbf{C}$ and v, repeating the following steps until convergence:

C-step: Fix v^*, solve for correspondence $\mathbf{C}$

$$\min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}(\eta(v^*))\|_{2,1} + \rho_{\text{corr}}(\mathbf{C})\,. \tag{6.30}$$

V-step: Fix $\mathbf{C}^*$, solve for part v

$$\min_{v} \|\mathbf{C}^*\mathbf{A} - \mathbf{B}(\eta(v))\|_{2,1} + \rho_{\text{part}}(v)\,. \tag{6.31}$$

A practical example of the alternating scheme applied to a pair of shapes is shown in Fig. 6.24.

6.2.4 Implementation

Numerical optimization. We implemented our matching framework in Matlab/C++ using the manifold optimization toolbox [42]. Each optimization step was performed by the method of nonlinear conjugate gradients. We initialize the alternating scheme by fixing $\mathbf{v}^* = \mathbf{1}$ (a vector of m ones), $\mathbf{C} = \mathbf{W}$, and by optimizing over $\mathbf{C}$. In all our experiments we observed convergence in 3-5 outer iterations (around 5 mins. for a pair of shapes).

Refinement. In order to account for noisy data, we also run a refinement step after each *C-step*. Specifically, assume $\mathbf{C}^*$ is a local optimum of problem (6.30). Then, we solve again for $\mathbf{C}$ by setting $\mu_3 = 0$ and replacing the data term with $\|\mathbf{C}\mathbf{\Phi}^\top - \mathbf{\Psi}^\top\mathbf{\Pi}\|_F$, where $\mathbf{\Pi}$ is a left-stochastic binary matrix aligning columns of $\mathbf{\Psi}^\top$ with columns of $\mathbf{C}^*\mathbf{\Phi}^\top$. By doing so, we disregard descriptor preservation (encoded in the original data term) and instead attempt to improve the alignment between the functional embeddings of the two shapes. We construct $\mathbf{\Pi}$ by solving for each column independently, boiling down to n nearest-neighbor searches in $\mathbb{R}^k$. Then, we solve for the semi-orthogonal $\mathbf{C}$ minimizing the new objective, and repeat the procedure until convergence. This refinement step can be seen as a generalization to partial maps of the ICP-like technique found in [171].

Conversion to point-wise map. Given a functional map between partial shape $\mathcal{N}$ and full shape $\mathcal{M}$ represented in the frequency domain as $\mathbf{T} = \mathbf{\Psi}\mathbf{C}\mathbf{\Phi}^\top$, we convert it to a point-wise map $f : \mathcal{N} \to \mathcal{M}$ using the nearest-neighbor approach described in [171].

6.2.5 Experimental results

Datasets. As base models, we use shapes from the TOSCA dataset [46], consisting of 76 nearly-isometric shapes subdivided into 8 classes. Each class comes with a “null” shape in a standard pose (extrinsically bilaterally symmetric), and ground-truth correspondences are provided for all shapes within the same class. In order to make the datasets more challenging and avoid compatible triangulations, all shapes were remeshed to 10K vertices by iterative pair contractions [95]. Then, missing parts were introduced in the following ways:[1]

Regular cuts. The null shape of each class was cut with a plane at 6 different orientations, including an exact cut along the symmetry plane. The six cuts were then transferred to the remaining poses using the ground-truth correspondence, resulting in 456 partial shapes in total. Some examples are shown in Fig. 6.21 and 6.23.

Irregular holes. Given a shape and an “area budget” determining the fraction of area to keep (40%, 70%, and 90%), we produced additional shapes by an erosion process applied to the surface. Specifically, seed holes were placed at 5, 25, and 50 farthest samples over the shape; the holes were then enlarged to meet the specified area budget. The total number of shapes produced this way was 684. Examples of this dataset are shown in Fig. 6.26 and 6.28.

Range images. We simulated range images by taking orthographic projections of the original TOSCA shapes from different viewpoints. Each range image was produced via ray casting from a regular grid with a resolution of 100×150 pixels. Examples are shown in Fig. 6.28.

Point clouds. Point clouds were generated by taking a subset of shapes from the first two datasets. Each partial shape was then resampled uniformly to 1000 farthest points,

[1] The datasets together with code for our method are available for download at `http://vision.in.tum.de/data/datasets/partial`.

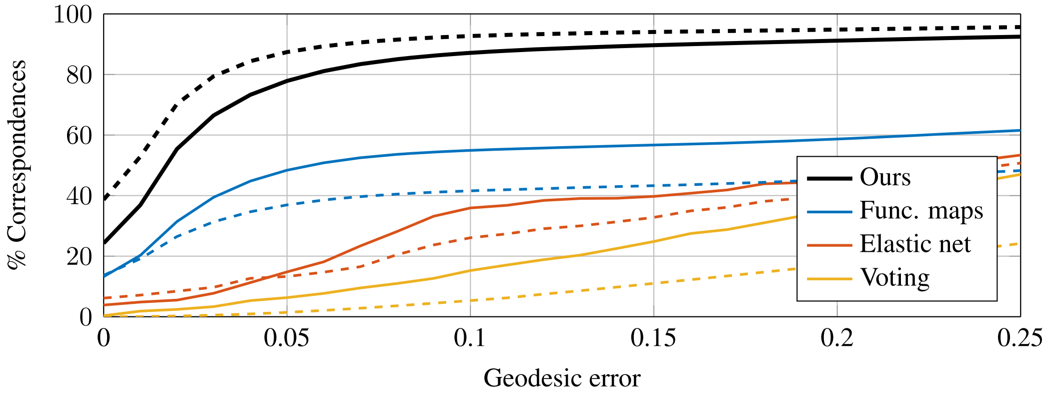


Figure 6.25: Correspondence quality of different methods evaluated using the Princeton protocol on partial TOSCA shapes with regular cuts (solid) and irregular holes (dotted).

and the tessellation removed. See Fig. 6.28 for examples.

Where not specified otherwise, we use 120 random partial shapes for the first dataset and 80 for the second, equally distributed among the different classes. Each partial shape is then matched to the null shape of the corresponding class.

Error measure. For the evaluation of the correspondence quality, we used the Princeton benchmark protocol [127] for point-wise maps. Assume that a correspondence algorithm produces a pair of points $(x, y) \in \mathcal{N} \times \mathcal{M}$, whereas the ground-truth correspondence is (x, y^*). Then, the inaccuracy of the correspondence is measured as

$$\epsilon(x) = \frac{d_{\mathcal{M}}(y, y^*)}{\text{area}(\mathcal{M})^{1/2}}, \tag{6.32}$$

and has units of normalized length on $\mathcal{M}$ (ideally, zero). Here $d_{\mathcal{M}}$ is the geodesic distance on $\mathcal{M}$. The value $\epsilon(x)$ is averaged over all shapes $\mathcal{N}$. We plot cumulative curves showing the percent of matches which have error smaller than a variable threshold.

Methods. We compared the proposed method with (full) functional maps [171], elastic net [185], and the voting method of [189] using the code provided by the respective authors.

Local descriptors. Due to the particular nature of the problem, in all our experiments we only make use of dense, *local* descriptors as a data term. This is in contrast with the more common scenario in which full shapes are being matched – thus allowing to employ more robust, globally-aware features such as landmark matches, repeatable surface regions, and various spectral quantities [171]. In our experiments, we used the extrinsic SHOT [216] descriptor, computed using 10 normal bins (352 dimensions in total). As opposed to [22, 177] which ignore points close to the boundary in order to avoid boundary effects, in our formulation we retain all shape points.

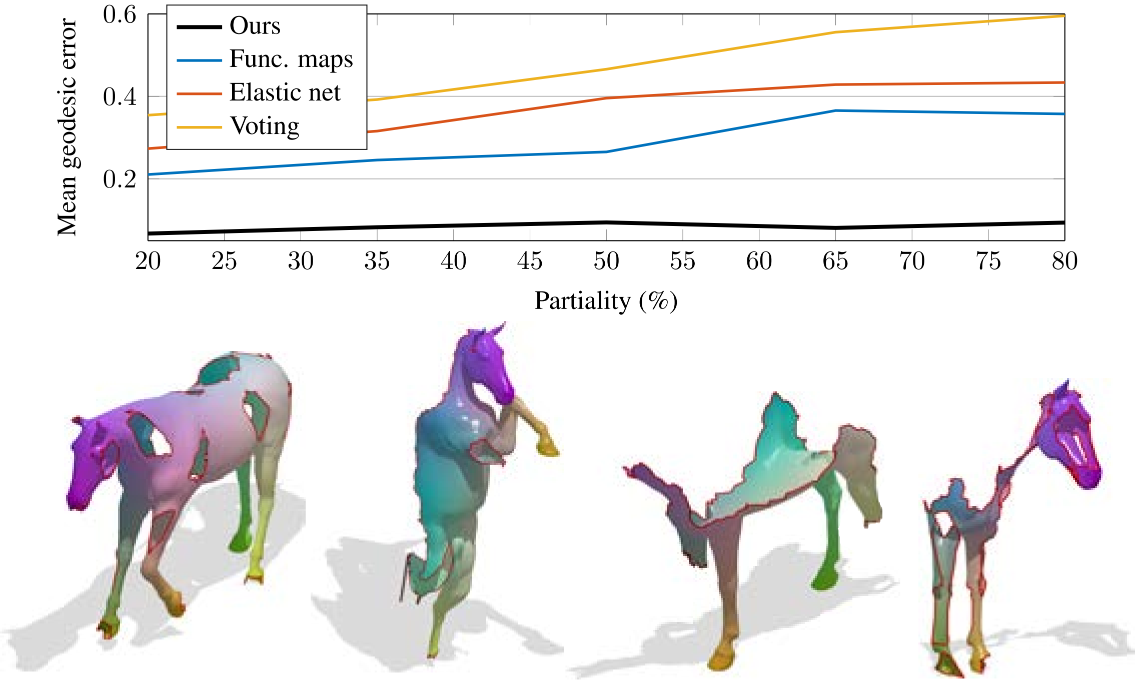


Figure 6.26: Correspondence quality (in terms of mean geodesic error, in % of diameter) obtained by different methods at increasing levels of partiality. Other methods show significant performance drop with increasing partiality, while the performance of our method is nearly-constant.

Sensitivity analysis

We conducted a set of experiments aimed at evaluating the sensitivity of our approach to different parametrizations. In order to reduce overfitting we only used a subset of TOSCA (regular cuts), namely composed of the *cat* and *victoria* shape classes (20 pairs).

Rank. In the first experiment we study the change in accuracy as the rank of the functional map is increased; this corresponds to using an increasing number of basis functions for the two shapes being matched. For this experiment we compare with the baseline method of Ovsjanikov *et al.* [171] by using the same dense descriptors as ours. For fair comparisons, we did not impose map orthogonality or operator commutativity constraints [171], which cannot obviously be satisfied due to partiality. The results of this experiment are reported in Fig. 6.27. As we can see from the plots, our method allows to obtain more accurate solutions as the rank increases, while an opposite behavior is observed for the other method.

Representation. Our method is general enough to be applied to different shape representations, as long as a proper discretization of the Laplace operator is available. In Fig. 6.28 we show some qualitative examples of correspondences produced by our algorithm on simulated point clouds and depth maps. Here we use the method described in [30] to construct a discrete Laplacian on the point clouds. This is traditionally considered a particularly challenging problem in robotics and vision applications, with few methods currently capable of giving satisfactory solutions without exploiting controlled conditions or domain-specific information (*e.g.*, the knowledge that the shape being matched is that of a human). These are, to the best of our knowledge, the best results to be published so far for this class of problems.

Comparisons

We compared our method on the *cuts* and *holes* datasets (200 shape pairs in total) ; the results are shown in Fig. 6.25. As an additional experiment, we ran comparisons against [171] across increasing amounts of partiality. The rationale behind this experiment is to show that, at little or no partiality, our approach converges to the one described in [171], currently among the state of the art in non-rigid shape matching. However, as partiality increases so does the sensitivity of the latter method. Fig. 6.26 shows the results of this experiment.

Parameters for our method were chosen on the basis of the sensitivity analysis. Specifically, we used $k = 100$ eigenfunctions per shape, and set $\mu_1 = \mu_3 = 1$, $\mu_4 = \mu_5 = 10^3$, and $\mu_2 = 10^2$. The different orders of magnitude for the μ coefficients are due to the fact that the regularizing terms operate at different scales. We also experimented with other values, but in all our tests we did not observe significant changes in accuracy. Additional examples of partial matchings obtained with our method are shown in Fig. 6.28.

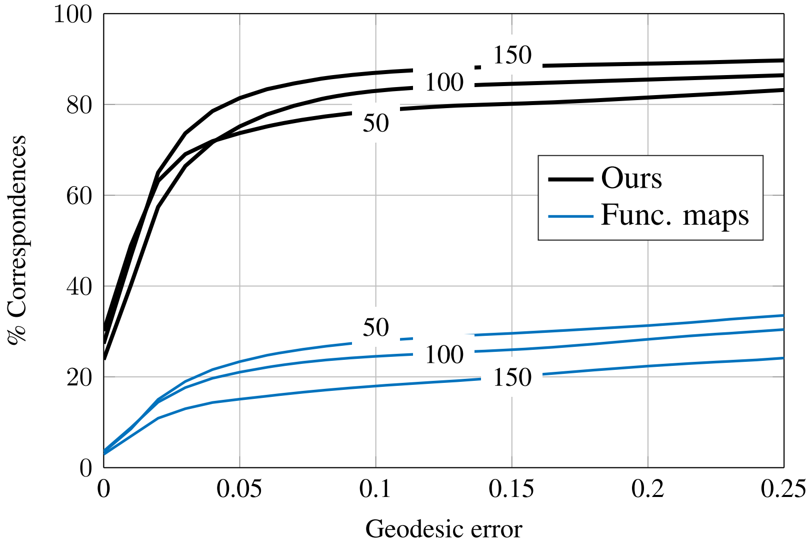


Figure 6.27: Correspondence quality obtained on a subset of TOSCA at increasing rank (reported as labels on top of the curves). Note the opposite behavior of the baseline approach and our regularized partial matching.

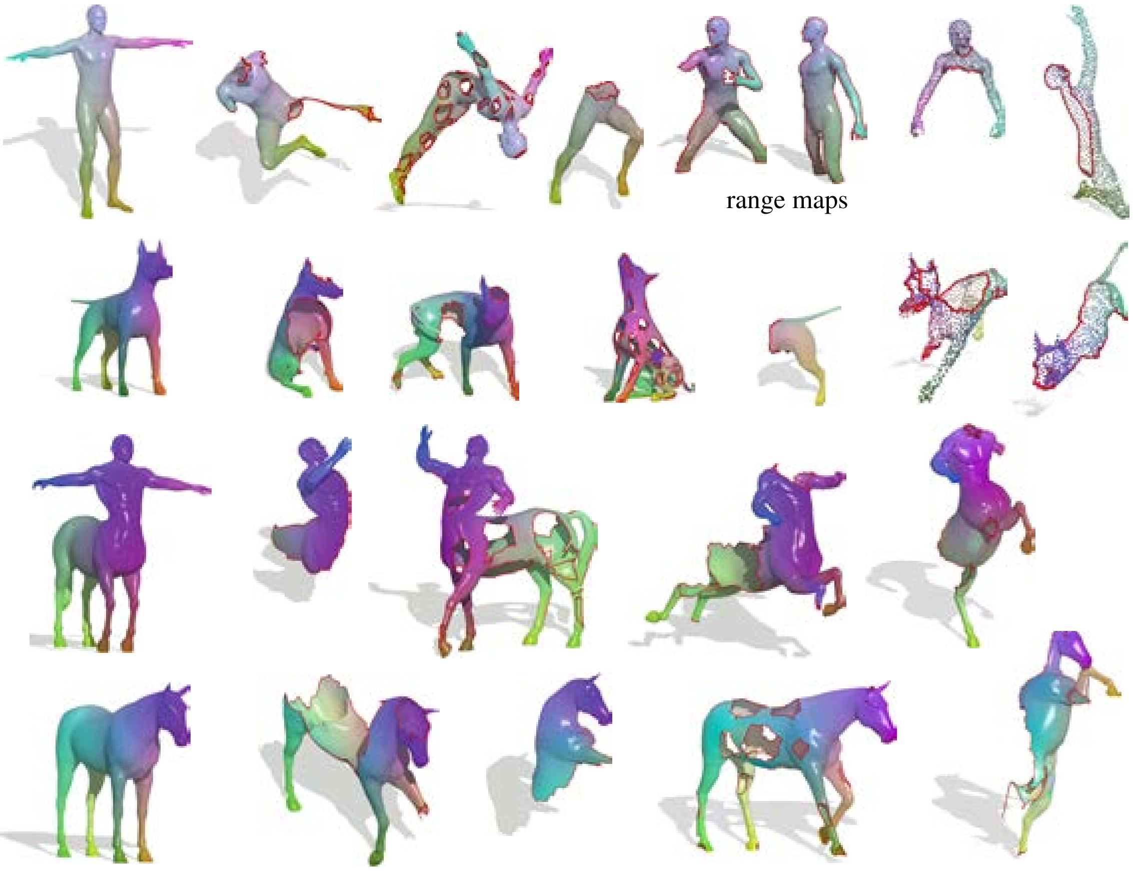


Figure 6.28: Examples of partial functional correspondence obtained with our method on meshes and point clouds from the proposed datasets. Notice how regions close to the boundary are still accurately matched despite the noisy descriptors.

6.2.6 Limitations and future works

One of the main issues of our method concerns the existence of multiple optima, which is in turn related to the presence of non-trivial self-isometries on the considered manifolds. Since most natural shapes are endowed with intrinsic symmetries, one may leverage this knowledge in order to avoid inconsistent matchings. For example, including a smoothness prior on the correspondence might alleviate such imperfections and thus provide better-behaved solutions. Secondly, since the main focus of this work is on tackling partiality rather than general deformations, our current formulation does not explicitly address the cases of topological changes and inter-class similarity (*e.g.*, matching a man to a gorilla).

Bidirectional partiality

We are trying to extend our approach to deal with partiality in both directions. In this case the correspondence is defined only over a part of both shapes: $T : \hat{M} \subset M \to \hat{N} \subset N$. Although the general idea of preserving some eigenfunctions between the two bases is still valid, we lost some properties on the diagonal structure of the functional map. For instance, since the overlapping part is now an unknown of our problem, we lost the ability to retrieve the slope of the diagonal looking at the spectral properties of the decomposition.

From Fig. 6.30 we can see that the ground truth functional map between the two partial shapes still preserves a diagonal structure, but with the decreasing of overlapping part some rows of C vanish resulting in the corresponding element of the diagonal of $C'C$ to be almost zero. Note also that the slope of the diagonal structure is not directly related to the percentage of overlapping part.

As a first attempt, based of the observations in Fig. 6.30, we tried to modify the *C-step* penalizing the $L1$ norm of the diagonal of $C'C$ and express the slope s of the W mask as an optimization variable leading to the following formulation:

$$\min_{C,s} \|CA - B\|_{2,1} + \mu_1 \sum_{i \neq j} (C^T C)^2_{ij} + \mu_2 \sum_i (C^T C)_{ii} + \mu_3 \sum_{i,j} (C_{ij}(j - is)^2(1 + s^2))^2 \tag{6.33}$$

In 6.29 we show some promising qualitative result of optimizing the *C-step* as in equation 6.33.

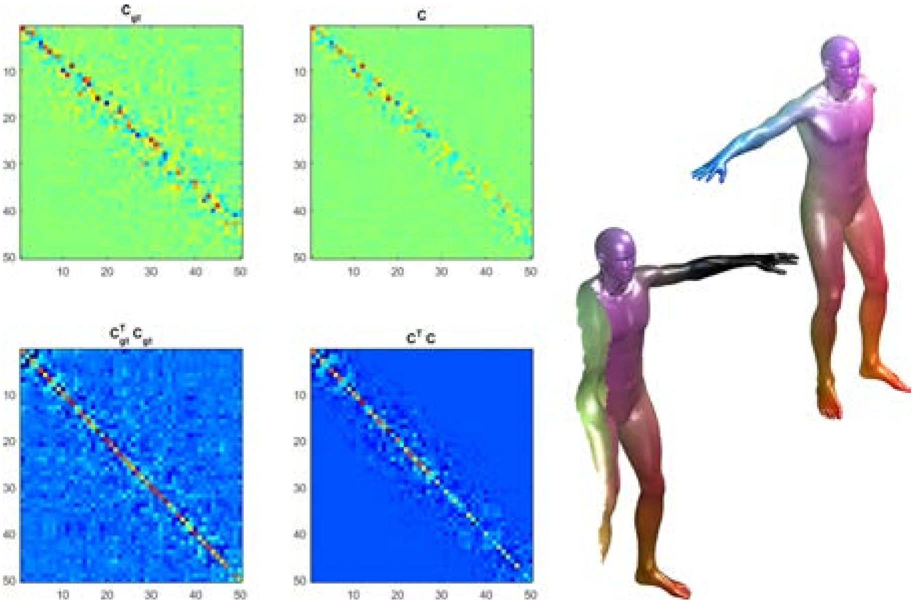


Figure 6.29: Functional map obtained with 6.33 compared with the Ground Truth (left). A function transferred using the obtained functional map (right).

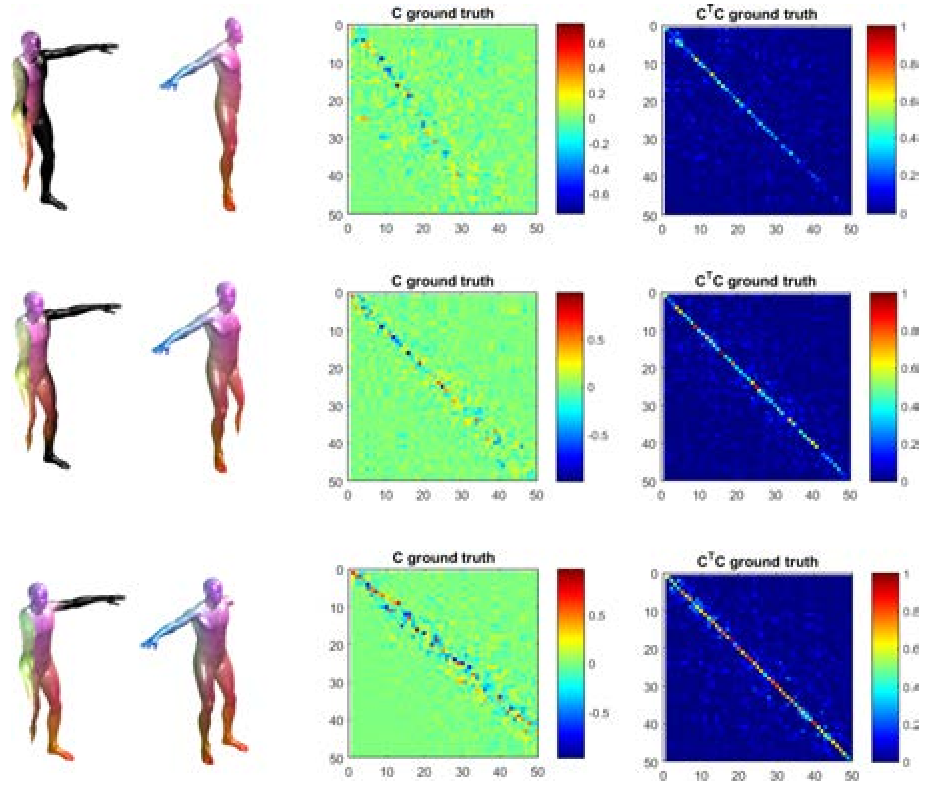


Figure 6.30: Example of partial matching between two parts of the same shape (Michael in null pose). The functional map preserves its (slanted)diagonal structure and $C^T C$ is still almost diagonal. We can notice that with the decrease of overlapping between the two parts some rows of C vanish resulting in the corresponding values on the diagonal of $C^T C$ to become zero.

III

Applications

7 Related Work

In the last years there have been an increasing diffusion of devices to track users in their pose and gestures. This is probably due to the investment of big game industry companies in the development of new input systems such, for instance, the tracked controllers *PlayStation Move*$^{(\mathrm{TM})}$ and *Wii Remote* or the depth sensor *Microsoft Kinect*®. As a consequence the HCI community increased their effort on exploiting such devices to provide users new paradigms to interact with computer systems in a more natural and intuitive way.

7.1 Interactive tables

One thriving field of application of such techniques is the so called *Interactive Tables*. they have proved to be a viable system to foster user participation and interest in many shared environments, among which educational and cultural environments such as museums and exhibitions have a leading role. They favor interaction among users and induce a sort of serendipitous discovery of knowledge by observing the information exploration performed by other users. Their use has been analyzed and evaluated in entertainment as well as in educational applications [25, 67, 113, 119].

Since their introduction in 1983 by Myron Krueger [132] a lot of experimental and commercial products have been proposed. Among the early multitouch, multiuser surface implementations *DiamondTouch*™, an interactive table produced by Mitsubishi Electric Research Labs (MERL), was able to recognize up to four different users by matching signals captured at users' touch by small antennas placed under the table surface with receivers capacitively coupled to the users through their seats [81]. Such a user identification technique, while effective and robust, is oriented to a structured collaboration among the users and is not easily applied to highly dynamic environments like museums and exhibitions.

A different technology is used in the Microsoft *Surface*® interactive table, which uses five cameras and a rear projector to recognizes finger gestures as well as objects equipped with tags placed on the table surface. Gloves equipped with fiduciary tags are proposed by Marquardt et al. [151] to identify both what part of the hand and which user caused the touch.

The *Frustrated Total Internal Reflection* (FTIR) technology [103], has received greater attention as a cost effective technology, able to trace many users with high frequency response; FTIR is based on infrared lateral illumination of a translucid surface, able to reveal small deformations caused by finger pressure. The problem of matching touches with users must, however, be solved by additional tracking systems based on analysis of the user position with additional external cameras and is subject to errors [82].

In more recent years the widespread of mobile devices with rich interaction capabilities has suggested to couple personal (small) and public (large) screens for enhanced multiuser interaction. The personal devices are used both for input, allowing each of several users to provide direct interaction and own information to a shared system, and for output of local private information; the large shared screen acts as a collaboration and information sharing environment, guided by the input provided by the single users [36, 83, 86, 194, 226].

In [76] a technique that uses accelerometer data to build a gesture based control schema was presented. In [208, 240] accelerometer data are given as input to SVM classifier to perform a semi-supervised gesture recognition.

In section 8.1 a setup of a large Interactive Table is presented. Computer vision techniques for blob detection and accelerometers data are coupled in order to allow for multiple users interaction in a both shared and personal environments.

7.2 Interactive Whiteboard

The term *interactive whiteboard* (IWB) usually refers to a wide class of hardware and software bundles that share the common goal of serving as a technological substitute for traditional blackboards (or flipcharts) in both offices and educational settings. Some studies find them to be a key component to enhance the performance of both teachers and learners in the classroom [141]. Other surveys are more critical, as they expose the limits of current implementations [200, 201]. Anyhow, their increasing adoption spurs the interest of both researchers and practitioners in the design of innovative supporting technologies, interaction models and teaching practices [198].

Since the inception of the IWB concept by Xerox Parc in the early nineties [213, 214], its basic elements have remained substantially unchanged. Generally speaking, an IWB is almost invariably made up of four components: a projection surface, a pointing device (that might well be the speaker's fingers), a set of sensors (usually dependent on the pointing device), and a set of software packages that enables different kinds of interaction models. Given the increasing success with the adoption of such devices, it is not surprising that a large amount of research has been recently investigating novel and diverse approaches to their implementation.

When considering the projection technique to adopt, the possible choices are basically two: front [44, 236] and rear [87] projection. The first approach is more prone to occlusion, whilst the second is more demanding in terms of room space. However they are basically identical from a human-machine interaction point of view.

Most of the current research is in fact focused on the pointing approach, that is the combination of hardware and software adopted to translate real-world actions into events on the whiteboard plane. A classical solution is to borrow resistive [202, 249] or capacitive [81, 125] technologies that are traditionally applied to touchscreen devices. Both of these approaches require to embed a sensing layer in the projection surface, but while resistive systems are able to detect both touches from objects and body parts, capacitive solution are limited to fingers, which can be an advantage or disadvantage depending on the intended usage. Other sensing technologies adopt electromagnetic sensors built in the pen or embedded in the whiteboard surface [141]. All these sensing approaches are based on reliable and proven technologies that offer a good accuracy and are easily scalable with the size of the whiteboard. However, the need for embedding the sensing layer within the projection surface contributes to high production costs. For this reason a great deal of effort has been spent in the study of alternative sensing technologies. Among them the most successful are based on an optical process, which allows to perform the pen (or finger) detection without a direct contact with the sensor. To this end, some widely accepted methods include infrared LED or laser curtains [29, 160], where the position detection happens by observing the occlusion produced with respect to LEDs or linear lasers illuminating an array of silicon photodiodes. Recently, even cheaper options have been brought to the table by researchers, in particular exploiting off-the-shelf hardware such as video cameras [135, 175] or video game hardware such as Microsoft Kinect [54, 250] and Nintendo Wiimote [41, 230].

Regardless of the adopted technological substrate, the practical utility and the impact of an interactive whiteboard are actually dependent on the associated control software and the underlying interaction model. Most IWB implementations share a common support for basic tasks, such as mouse emulation and on-screen writing. However, the study of novel and advanced interaction models is a very active research topic. Some authors propose to combine pen and touch-based interaction to better suit many presentation needs [228].

It should be noticed, however, that essentially all the proposed paradigms are deeply bound to original whiteboard metaphor. That is, the interaction happens almost entirely in the 2D space of the surface (with the possible exception of a recently proposed model where the proximity to the surface is exploited to enable additional gestures [212]). Still, there is no technical or philosophical reason to not extend the degrees of freedom of the system to the full 3D space, enabling an extended class of gestures and controls not conceivable with the basic 2D paradigm. In our method we propose to treat the pen as a 3D pointer, suggesting some possible applicatons and introducing a technical design to effectively detect and track it.

A natural choice for the detection of the pen and for the estimation of its position in the 3D space would be to use fiducial markers [34, 92], that is small 2D patterns that can be used to recover 3D orientation both with monocular views or through triangulation. Nevertheless, while marker-based approaches are widely adopted within augmented reality applications, they exhibit several drawbacks with respect to our specific requirements. Namely, printed markers are hard to detect and painful to recognize when the viewing angle is grazing, which fully happens as long as they are freely moved and rotated in 3D space. Moreover, a quandary emerges between the will to keep markers suitably small for the pointer to be ergonomic and the need for them to be large enough to be clearly seen by cameras, especially considering the large shift range expected. Finally, special care should be applied to avoid false positives originating from background clutter, which abounds in any classroom.

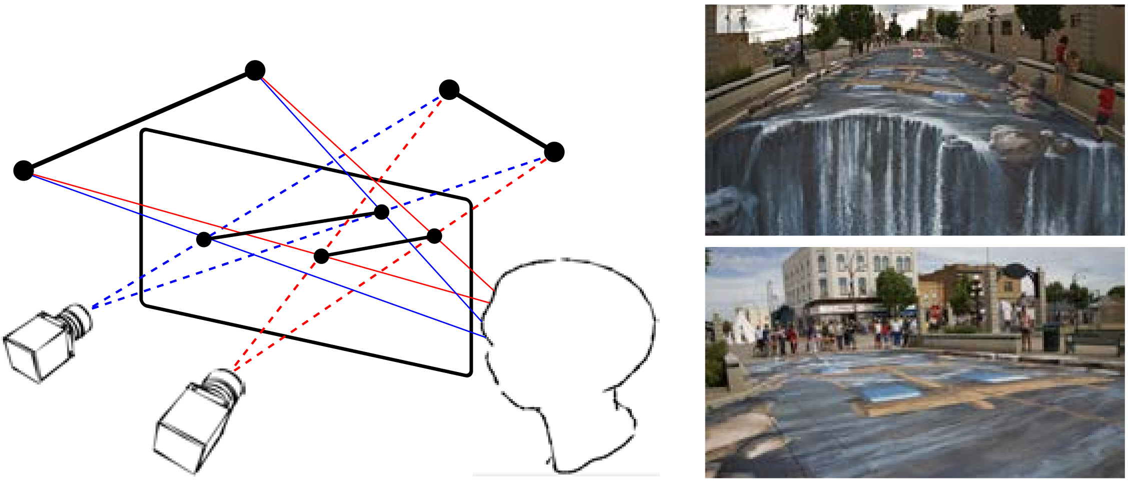
In order to overcome these drawbacks, in section 8.2 we describe a specially crafted setup made up of an active IR-emitting pointer and a camera network that reliably tracks it and computes its pose in the 3D space. By using a set of different frequencies, the system allows for many pointing devices to be used at once. The very same approach is also tapped to notify the push of the buttons available on the pointers.

7.3 Viewer dependent rendering

Traditional stereoscopic displays assume the viewer to be standing at a specific location, that is the same pose (relative to the screen) of the stereo camera pair that depicted the scene (physically or virtually). Even for basic applications, such as movies or games, this leads to visual inconsistencies as soon as the user moves his head. It is clear that the 3D illusion holds if and only if the optical system of the user exactly replicates the one that produced the images, otherwise, the perceived scene would be distorted as the 3D objects reconstructed (by the brain) will diverge from the original in size, position and proportions (see Fig. 7.1). Additionally, from a perception point of view, the effect can be further aggravated by the fact that points that would originally project into incident lines of sight, would probably result skewed when observed from the wrong point of view. The only viable solution to this limitation is to provide a rendering dependent on the user position and on his interocular distance.

Viewer-dependent displays have been extensively proposed in recent scientific literature, since they offer many advantages. For starters, they are able to guarantee that the viewed objects exhibit a correct size within the Euclidean space where the user resides, thus allowing to interact with them naturally and to make meaningful comparisons between virtual and physical prototypes. Moreover, viewer-dependent rendering lets the user walk around the scene, viewing it from different angles and enabling the same inspection dynamic that would be possible with a real object.

Such ideas are not new at all and have been widely developed in literature since their early implementations with the first immersive virtual reality and CAVE environments [75, 79]. More recently, Harish and Narayanan [105] combine several independent monitors arranged in a polyhedra to create a multiple-angle display and a fiducial marker system to track the user pose. In their system the object is visualized as if it was inside the solid space defined by the monitors. Garstka and Peters [97] used a single planar surface to display non-stereoscopic content according to the pose of the user head obtained with a Kinect sensor. A combination of Kinect devices and traditional range scanners have been adopted in a very similar approach by Pierard et al. [174].



(a) (b)

Figure 7.1: Left: The stereo inconsistency problem. Any stereo pair, when observed from a location different from the position of the capturing cameras, will result in impaired perception. Under these condition any observer will see an unpredictably distorted 3D object. Right: images from a work by Artist Edgard Muller. The simulated perspective only works if the observer stands in a very specific observation point (top).

It should be noted that, albeit implementing view-dependent solutions, the aforementioned approaches do not exploit stereoscopy. Stereo vision is exploited, for instance, by Hoang et al. [111], that used standard head tracking techniques to allow slight head movements when looking at a 3D scene on a monitor. The concept is very similar to the non-stereoscopic technique proposed a few years earlier by Buchanan and Green [52]. In those cases, while the correct projection is always offered to the user, he is not allowed to inspect the object by moving around it.

In section 8.4 the tracking device described in section 8.3 is used to enhance a pair of standard LCD shutter glasses, providing a setup to assemble a cheap viewer dependent display with off-the-shelf acquisition and projection devices. In addition we propose an evaluation method that can be used to assess the accuracy of similar view-dependent systems from a calibration error and delay point of view.

7.3.1 Stereoscopic perception

Even when a good calibration of all the components is provided, it is not straightforward to evaluate the perception of the users since other subjective aspects come into play.

The quality of visual perception depends on many factors, primarily depth perception and perspective rendering. Depth is primarily bound to binocular vision (*stereopsis*). Studies in physiology and neurophysiology have investigated the perception of depth in the real world under stationary and moving conditions; it is common experience to perceive depth even in absence of stereopsis through the motion parallax, i.e., by translating the observer's optical viewpoint and comparing the apparent shift of objects at different distance [163, 164]. The depth perception is based on the analysis of the stimuli received during pursuit eye movements; while the perception of the objects' ordering sequence is usually correct, the interpretation of the actual distances is generally inaccurate, unless integrated by information coming from other types of interaction between the observer and the world [88]. A first evaluation of the 3D perception in virtual reality applications is discussed by Ware et al [232] in the *Fish tank VR* system, made of a stereo display monitor and a mechanical sensor for tracking the user's head position. The relations between mono and binocular view under different conditions are analyzed by Yang-Mao et al [244] using a commercial see-through head mounted display. Quantitative (e.g., optimal viewing distance) as well as qualitative factors (e.g. size vs. shape analysis) helping a user to correctly perceive depth in a virtual scene are evaluated.

Except for [179, 244], little effort has been put to evaluate viewer dependant stereoscopic interfaces from a quantitative point of view. In section 8.5 we try to address the lack of quantitative evaluation methods by introducing an objective evaluation procedure: users are requested to measure with a physical ruler virtual object projected over a horizontal display. This study is aimed mainly in assessing the difference between the role of stereoscopic vision, compared to monocular vision, in the user perceptions of virtual object dimensions.

8

User tracking in HCI applications

In many interaction models with a computer system there is need to identify the position and the movements of a specific object to allow users to perform specific actions. In this chapter we describe some techniques to retrieve the position of a known device over the time and codify its movement with some specific input commands given to the system by the users. Specifically we have developed and evaluated two ”active” devices, together with the relative interaction paradigms, that allows the users to interact with the specific application a natural and seamless way.

In the description of the proposed methods we assume to be given a calibrated camera network, which is indeed the main goal of the methods described in the first part of this thesis.

The first device, presented in section 8.1, was used as input in a large *Interactive Table* exposed at the museum exhibition *”William Congdon in Venice (1948-1960: An American Look”*. Relating the tracked position of a mobile phone over a translucent plane with its accelerometers data it is able, after a learning phase, to handle multiple users in the selection of some interest points of a map and to stream the selected content in both a private and shared display.

With the second device we aimed to continuously track the device position and orientation (5 degrees of freedom). This allows the device to be used as a pointing device in an *Interactive Whiteboard.* The novelty of our proposal lies in the time domain identification of the device which allows to handle multiple users while maintaining a simple geometry of the device. It is in fact composed only by two IR LEDs which are clearly visible and distinguishable even with low end camera networks. It is presented in section 8.2, while in section 8.3 this it is further improved in its accuracy exploiting its phase-shift properties to correct the camera’s sync.

In section 8.4 we exploit this device to implement a position dependent stereoscopic rendering and we propose a technique to assess the perspective error and the delay impact in a general stereoscopic display setup. Finally, in section 8.5, we focus on the study of the human perception of stereoscopic vision and the ability of such a system to supply a coherent perception of spatial relation in the mixing of virtual scene and real background.

8.1 Using Multiple Sensors for Reliable Markerless Identification through Supervised Learning

In many interaction models involving an active surface, there is a need to identify the specific object that performs an action. This is the case, for instance, when interactive contents are selected through differently shaped physical objects, or when a two-way communication is sought as the result of a touch event. When the technological facility is based on image processing, fiducial markers become the weapon of choice in order to associate a tracked object to its identity. Such approach, however, requires a clear and unoccluded view of the marker itself, which is not always the case. We came across this kind of hurdle during the design of a very large multi-touch interactive table. In fact, the thickness of the glass and the printed surface, which were required for our system, produced both blurring and occlusion at a level such that markers were completely unreadable. To overcome these limitations we propose an identification approach based on SVM that exploits the correlation between the optical features of the blob, as seen by the camera, and the data coming from active sensors available on the physical object that interacts with the table. This way, the recognition has been cast into a classification problem that can be solved through a standard Machine Learning framework. The resulting approach seems to be general enough to be applied in most of the problems where disambiguation can be achieved through the comparison of partial data coming from multiple simultaneous sensor readings. Finally, an extensive experimental section assesses the reliability of the identification.

8.1.1 The context: a map based multiuser art browser

The art exhibition mentioned in the introduction provided an opportunity to design and build three large interactive tables equipped with the standard set of input and output devices such as cameras for blob detection and projectors for information display. While a focused effort has been made in order to create a generic and reusable system, the design

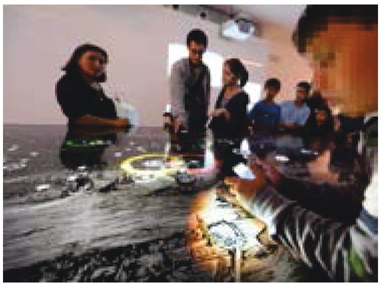


Figure 8.1: A group of people operating on the map based multiuser art browser described

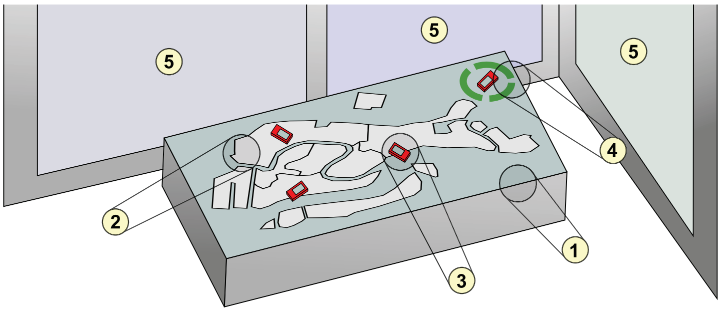


Figure 8.2: Schematic representation of the components of the proposed multiuser interactive table.

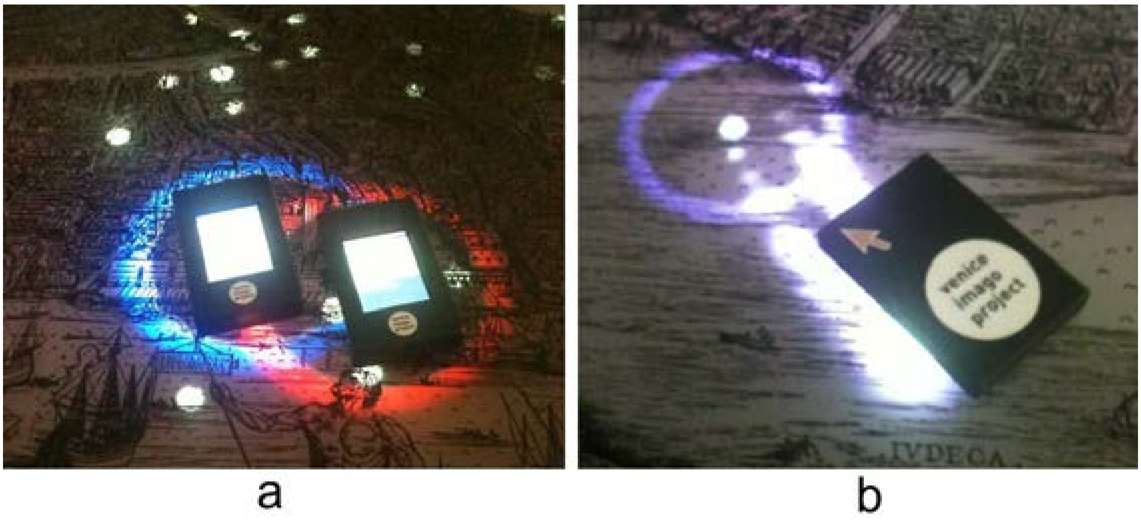


Figure 8.3: (a) The feedback on cursor location. (b) A visual suggestion to move.

of the table hardware and software is still the result of requirements partly bound to the interaction functions, partly imposed by the environment.

Interaction Model

Each table presents a high resolution diaphanous map of Venice (Fig. 8.2-2) printed on a thick glass surface (Fig. 8.2-1). A total of three tables has been built. Each one portraits a different period of the city history. The well-known lithography made by Jacopo De' Barbari [188] that represents an aerial view of the island of Venice has been selected to represent the XVI century. The Napoleonic cadastral map was chosen to provide an overview of the city during the XVIII century. Finally, a satellite view has been used to represent the modern era.

The required interaction is based on placing or moving on the table objects representing the virtual visitor position in the town. These objects, that in the following will be referred to as *cursors*, are smartphones that are equipped both with a display and some internal sensors, such as accelerometers and compass(Fig. 8.2-3).

Relevant places are associated to paintings by artists of different epoques, portraying the city views related to the location selected by the user and the historical period expressed by the specific map. They are shown as pulsating spots on the map, attracting the user attention (Fig. 8.3(a)).

As the user moves the cursor over a relevant place, the spot is highlighted to confirm its detection and the corresponding artwork is projected on the walls surrounding the installation(Fig. 8.2-5); relevant information about the author and the picture are presented on the display of the cursor.

Further, the cursors can be lifted from the table and used as a gesture-based remote control that allows the user to get near the projected paintings and still continue the browsing activity, updating the projection.

To allow more users to interact with the table and to experiment a simpler interaction style, suitable for less skilled users, a few *passive* cursors have also been used: they are small rectangular boxes decorated with the project logo, which can be placed and moved on the table like the active cursors, causing the same behavior of the active devices. In this case, obviously, there is no local information processing and display.

Additional visual cues are generated if the cursor is placed near to a spot but not moved over it. A stream of light is generated, moving from the cursor to the nearest spot, suggesting a move (Fig. 8.3(b))

Each user is independent; at most 5-6 users can operate at the same time, distributed along the table sides with a comfortable amount of surrounding space to experience an open view of a part of the map. Such distribution assures also that the user position around the table allows the placement of the video projections on the walls in a regular and pleasant layout.

Such layout is automatically arranged by an algorithm that tries to optimize space usage by dynamically resizing pictures and trying to place new artwork directly in front of the user that required its display. Since several new paintings could appear at the same time, special care must be applied to ensure that the user receives enough hints to visually associate the newly displayed painting with the action he just completed.

The tables are operated by the users without help, but the installations are guarded by cultural mediators, personnel available to visitors to help them in case of need, and ready to explain both the table functions and the associated content.

Optical System

As for any multitouch system, one of the most critical choices is related to the technology used to detect the user interaction with the table surface. Given the large active surface (measuring 300cm x 200cm) using a touch sensitive overlaid plane was not an option for economic and practical reasons. Also the adoption of an external vision-based tracking system was not viable, since the large number of simultaneous users would cause unpredictable occlusion conditions.

In this regard, our choice has been directed to classical blob detection by placing a number of cameras inside the table and oriented toward the surface. Specifically, we used infrared cameras (i.e. cameras that leave out the visible light) equipped with an 850nm thresholded filter.

This kind of camera is usually coupled with some source of infrared illumination, so that the visible light produced by the internal projector does not interfere with the blob-detection. To this end, two illumination techniques are usually adopted: the Frustrated Total Internal Reflection (FTIR) and the Diffused Illumination (DI).

FTIR is based on the interference between the object in contact with the surface and infrared light tangentially diffused inside the thickness of the surface layer and trapped by the total reflection angle. Such interference causes the light to change direction and thus to escape from the surface layer toward the camera.

By contrast DI implies the naive illumination of the objects via a diffused light that passes through the surface and is reflected as it encounters an infrared-reflective obstacle.

For this installation we used DI, which performed better than FTIR, as the large size of the table hampers the even diffusion of the light and the strength of the returned signal. This limitation is even more exacerbated by the presence on the lower side of the table surface of a diffuser layer that is needed to offer a screen for back-projection. The visualization itself happens by mean of two short throw projectors mounted inside the table.

Given the unfavourable ratio between the size of the table and its height (about 85cm), the use of a first-surface mirror has been necessary to create a suitable light path. The cursors consist of 6 Android based phones that communicate with the PC controlling the business logic via Bluetooth.

8.1.2 Unreliability of Marker Recognition

Since the proposed interaction model requires to associate each blob seen by the camera to a device, a reliable identification schema must be deployed. The adoption of Diffused Illumination would normally allow the use of fiducial markers to perform recognition. For this reason our first prototype was based on ARToolkit+ [227](see Fig. 8.5), a widely used extension to the well-known ARToolkit tag system [123].

Unfortunately, in our setup we faced several hurdles that compromised the viability of a marker-based recognition. The first problem is related to the size of the table. In fact, for the surface to be sturdy enough to be safe and do not flex under its weight, it has been necessary to use a glass pane 12mm thick. Since the diffuser layer is placed on the bottom side of the surface, this resulted in the marker pinted surface being at least 12mm

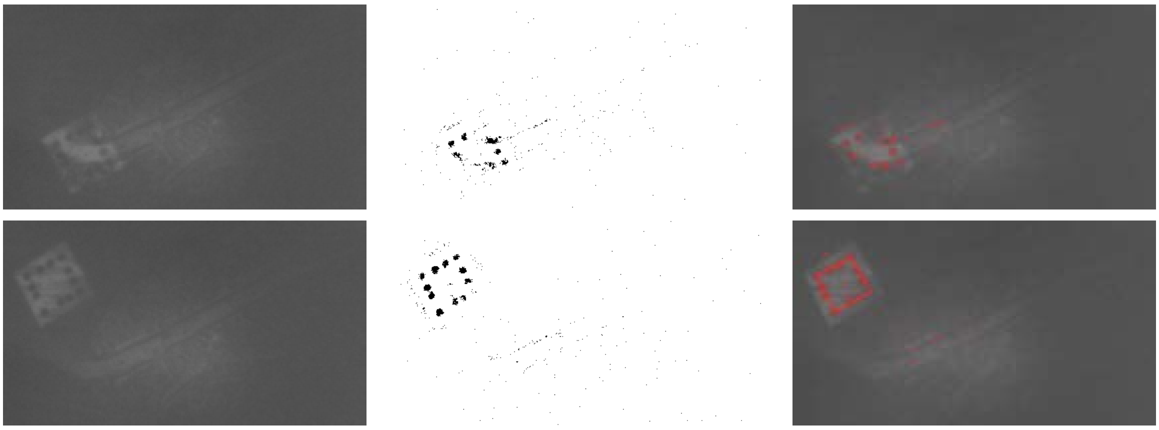


Figure 8.4: Two examples of the Pi-Tag recognition process with our setup. The original shot is shown in the first column. The other two columns show respectively the thresholded image and the detected ellipses (bad contrast is due to the low transmittance of the glass).

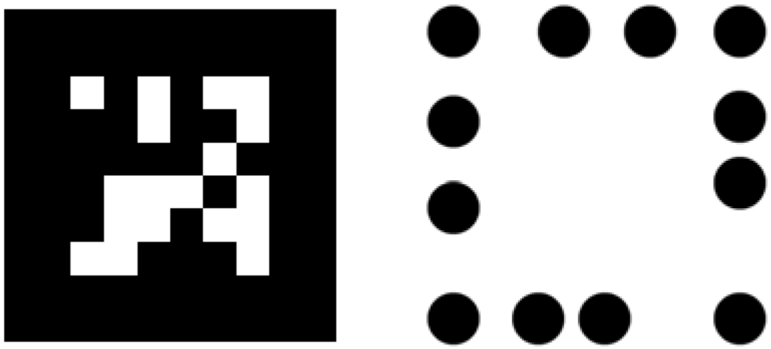


Figure 8.5: The two fiducial marker designs tested with our setup: ARToolkit+ (on the left) and Pi-Tag (on the right).

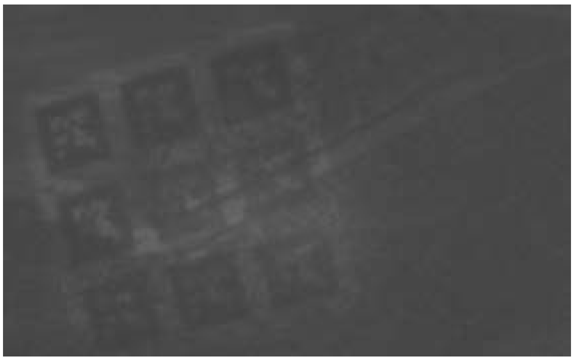


Figure 8.6: The effects of glass thickness and surface print over the readability of ARTookit+ tags (bad contrast is due to the low transmittance of the glass).

away from it, which in turn caused a strong blurring. This blur does not prevent to track the objects as blobs, however it inhibits even the most infrared-reflective markers to be distinguished reliably. Further, an additional source of noise and occlusion is represented by the map printed on the upper side of the surface.

In order to minimize such negative effects we took several measures: the markers have been printed on a substrate highly reflective with respect to infrared light, the light spots have been carefully calibrated to avoid blooming and reflections while still allowing an even and bright illumination, the camera exposure and gain have been manually optimized to get the best compromise between signal and noise. Finally we adopted a best-of-breed adaptive thresholding algorithm and we manually tuned it to get the best foreground/background separation.

In spite of these precautions, ARToolkit+ was not able to correctly recognized its markers, which appeared very faint and occluded to the camera (see Fig. 8.6).

As a last resort, we tried to change the fiducial marker system and we made some in depth experiments using Pi-Tag, a recently introduced fiducial tag that exhibits an ellipse-

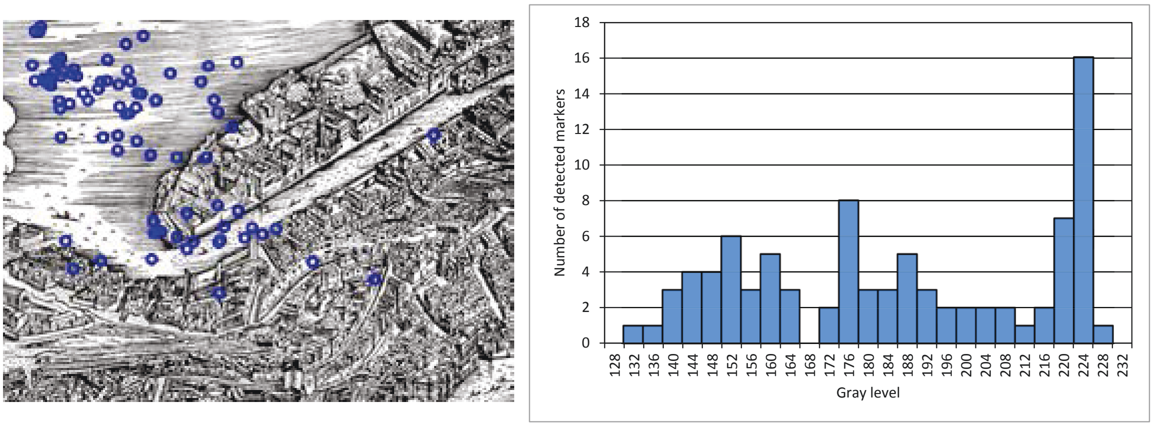


Figure 8.7: The locations of the detected markers in the test video overlayed to the printed map (on the left) and the distribution of the detections with respect to the average gray level.

based design that is moderately resilient to occlusion [34]. The adoption of and ellipse-based design makes a lot of sense within our setup, since ellipse detection is fairly robust to isotropic noise such as blur. Further, the Pi-Tag recognition algorithm is able to cope with some missing ellipses, which could help a lot when dealing with the non-uniform occlusion caused by the printed map overlay.

Regarding this latter problem, our first batch of qualitative tests revealed that the recognizability of the markers strongly depends on the local density of the printed overlay. In Fig. 8.4 we show two anecdotal examples. In the first one, the marker is seen through a non-uniform area where a Venice ”canal” separates two blocks of buildings (see also Fig. 8.7 for a bitmap image of the printed area). In this case only the ellipses on the clear area can be detected and the combination of blur and occlusion is too strong to allow recognition. By contrast, in the second example shown in Fig. 8.4, the marker is placed on the clear area of the ”lagoon” and, while an ellipse is still missing, the remaining signal is good enough for the tag to be detected and recognized.

Even from this simple qualitative evaluation it seems that the quality of recognition is still too unpredictable to be deemed as reliable. To get a quantitative assessment of this speculation we made a video a couple of minutes long that captures the marker moving over several locations. To obtain a fair evaluation for our setup, we tried to evenly cover the area. In Fig. 8.7 we plotted the location of the detected markers over a part of the original bitmap of the printed map. As expected, it can be observed that most of the successful recognitions happen within the less occluded areas. This behaviour is further characterized in the second column of Fig. 8.7. In this histogram recognition events are grouped according to the average grey level exhibited by the map area where they happen. Finally, the overall recognition rate was about 1%, since we obtained only 89 correct detection within a video with more than 7000 frames.

This very low level of reliability of the marker-based recognition, prompted us to explore different approaches in order to associate a blob to the device that produced it.

Specifically, we exploited the expected correlation between blob movements (as seen from the camera) and acceleration data (as measured by the device sensors). The details of the implemented solution and its effectiveness in a real scenario are described in the following sections.

8.1.3 Cursor/Table Communication

Before addressing the task of associating each tracked cursor to its identity, we first need to sort out a couple preliminary problems related to the data interchange between the active cursors and the table itself. Namely, we have to define a protocol for message transmission and a technique to obtain a reliable synchronization between the real time clock of the cursor and of the table. The latter is especially important, since we will adopt machine learning techniques that will relate specific data patterns gathered from the sensors with the information obtained from the camera. If proper synchronization does not happens both the learning and the recognition steps can be severely hindered since the mutual causality between the two phenomena could not hold any more.

Message exchange

All the communications throughout the system happen by exploiting the Serial Port Profile (SPP) of the Bluetooth standard. From a design point of view this is a reasonable choice for many reasons. For starters, Bluetooth requires much less power than Wi-Fi to work, and since the cursors should be able to run on battery power for a whole 8-hours day, energy saving must be seriously taken in account. Specifically, the device creates the server SPP socket, i.e. it presents itself as a serial port service in a similar way to what external GPS antennas or barcode readers generally do. Each device is first paired with the associated table, which scans at intervals for them. When a device is found, the table initiates the serial connection. The lack of connection for a long period indicates that a device is either malfunctioning or has been stolen, either way, a warning should be triggered. The communication protocol uses Consistent Overhead Byte Stuffing [62] to transmit packets made up of an header, that specifies a message type and the id of the sender, and a payload that is defined according to the characteristic of the exchanged data. There is a total of four types of messages that are transmitted within the system:

Type	*Sender*	*Content*
Sensors	Device	Accelerometer and compass data
Url	Table	Url of the content to display
Action	Device	Url selected by the user
Ping	Table	Timestamp of table real time clock
Pong	Device	Timestamp of device real time clock

The *Sensors* message is sent at regular intervals from the device to the table and contains the data gathered from the accelerometers and the magnetic field sensor. Since the actual update frequency of such sensors is usually very high on most Android devices, the data are not sent directly, rather an integration step is performed on board to make the update rate of the sensor data commensurate to the frame rate of the infrared camera (about 30fps). The integration step imply the additional advantage of an implicit noise reduction due to the averaging.

The *Url* message is sent by the table to control the display of the device. Each device contains a set of HTML pages that can be loaded by the local browser. Once the table identifies a device, it send the Url that selects a menu page related to the area of the map where the device has been placed. This is the only control action that the table performs with respect to the device.

The *Action* message is sent from the device when a user clicks on a link in the local browser. The click is intercepted by the application running on the device and the GET parameters (if any) are sent to the table. This protocol allows to define custom parameters to trigger actions by the table such as the display of an artwork, the highlight of interesting point on the map or any other interaction that can be added in the future.

Finally, the *Ping* and *Pong* messages are used to transmit the current time (in milliseconds) as measured by the real time clock respectively of the table and of the device. These two message are meant to be used to perform a round trip, initiated by the table, for internal clock synchronization. The details about how this synchronization happens will be given in the next section.

Time Synchronization

In order to properly correlate the data coming from all the devices we must be able to measure the value of all the sensors at certain times. Even if the delay from the camera acquisition to the blob identification is less than a couple of milliseconds and can be ignored, this is not true for the data coming from the devices accelerometers. Indeed, the delay introduced by the Bluetooth communication is in the order of tens of milliseconds and also, unfortunately, is not constant, so we cannot reliably compute the data time from the arrival time measured at the table PC. The importance of automatically synchronize

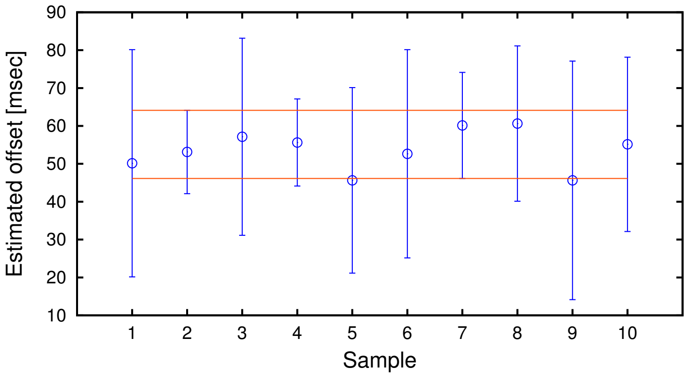


Figure 8.8: Example of the offset estimation via intersection of measured intervals.

the clock of all devices is twofold. First, the initial offset between each clock is not negligible and must be taken into account to properly estimate the status of the system. Second, due to the low accuracy of the quartz clock on modern devices that do not tick exactly with the same frequency, a continuously increasing drift is accumulated as time goes by.

To estimate with high accuracy the time offset between the server and each device we chose to adopt the same method used in SNTP. The synchronization process starts by collecting a set of pairs (o, d) where o is the offset between the server and a specific device, and d is the measured round trip time. To gather each of this data, a packet is sent to the device containing the current value of server time. On arrival, the device must reply attaching to the packet its own time. When the server receives the reply, is able to compute the offset between its time with respect to the device and the round trip time as the difference between the arrival time and the original send time contained into the packet. If we assume a symmetric unknown communication delay, the computed offset would be exact but this assumption is just unfeasible for Bluetooth communication devices. However, we can state for sure that the true offset value must be contained in the interval $[o - d/2...o + d/2]$. To restrict the interval as much as possible a lot of (o, d) pairs are collected and the largest common interval (see Fig. 8.8) is computed by using the algorithm proposed in [153].

8.1.4 Identification By Learning

In principle, several techniques could be adopted to identify blobs by combining sensor data and tracking information. For instance, a hand crafted decision tree with suitable thresholds could be applied to perform a direct verification of the compatibility between the blob status and the reported orientation of the compass. However, this kind of approach becomes cumbersome when the amount of information begins to grow, which is the case, for instance, if the history spanning the last few frames is considered. Further, it is not always obvious how to relate the data and how to weight the contribution of each source of information. Whenever a con-causal relation between different data sources exists, but it is not clear how to design and parametrize a direct algorithm to exploit such relation, resorting to some machine learning technique is a natural choice. In fact, given a reasonable feature selection, learning techniques have proven to be often more effective in classification tasks than manually crafted solution that exploit a direct knowledge of the problem domain [31, 195].

We decided to address the issue of blob-device association in terms of a non-probabilistic binary classification problem. In fact, during the normal usage of the system, two crucial class of events can occur in which inference from sensor data can be performed to disambiguate some of the pairs. In the following we will refer to these two events with the terms *appear* and *stop*. The *appear* event happens when a new blob starts to be tracked by the system. If the blob is generated by one of the connected devices, the time-synchronized signal produced by the accelerometers should be somehow related to the increasing area of the blob. Also, the absolute orientation of the device with respect to the magnetic north

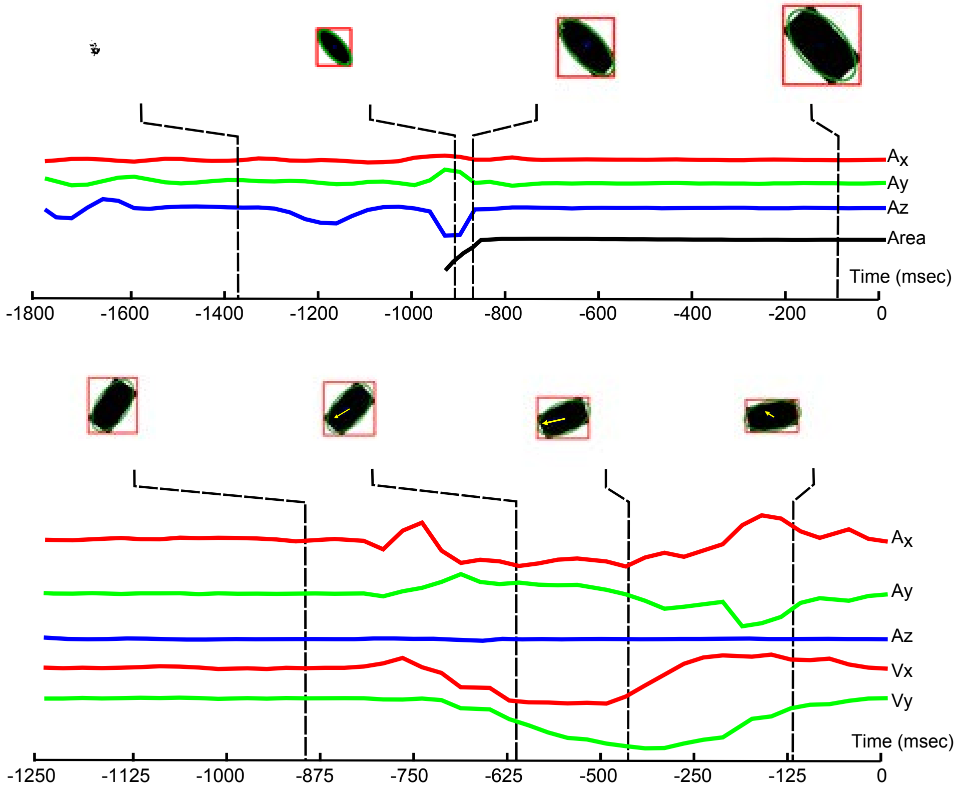


Figure 8.9: Mutual causality relations between the observed blobs and the data gathered from the sensors. In the first row an *appear* event is shown: note that as the object touches the table a sudden stop in the vertical acceleration (Az) can be observed, as well as a fast increase in the blob area. In the second row a *stop* event is detected as the accelerations (Ax, Ay, Az) and the velocities (Vx, Vy) measured toggle from a quiet state to an active state and then to a quiet state again. Note that the speed of the blob measured from the camera (yellow arrow) agrees with the data coming from the sensors.

should correspond to a specific orientation of the blob in the image frame (assuming that the table cannot be moved once calibrated). Differently, the event *stop* is triggered when a tracked blob stops moving. When this occurs, the blob velocity signal computed by the table will probably be coherent with the accelerometer data, both defining the same space-time trajectory.

In Fig. 8.9 an example of the signals coming from the aforementioned sensors is shown for an instance of the *appear* and *stop* events. Because of the non negligible sensor data correlation that is exhibited in this two particular events, a binary classifier should be able to answer the question ”Is this blob data related to this specific device data?” with very high accuracy. Many different types of binary classifiers have been proposed in literature, each with its own strengths and weakness. Due to the relatively high-dimensional well-separable sensor data we decided to use the well known *Support Vector Machine* method [53].

Machine Learning with SVM

Suppose that we are given training data

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\} \subseteq \mathbb{R}^d \times \{-1, 1\} \tag{8.1}$$

Where $\mathbb{R}^d$ denotes the space of the input patterns (sensor data in our case) and $y_i \in \{-1, +1\}$ indicates the binary class in which the point $\mathbf{x}_i$ belongs. In the simplest case, we can assume that there exist some hyperplanes which separate all points having $y_i = 1$ (positive examples) from those having $y_i = -1$ (negative examples). Any of those hyperplanes can be defined as the locus of points $\mathbf{x}$ satisfying the equation.

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{8.2}$$

Where $\mathbf{w}$ is the normal of the hyperplane and $|b|/\sqrt{\mathbf{w} \cdot \mathbf{w}}$ is the perpendicular distance from the hyperplane to the origin. A support vector algorithm simply looks for the hyperplane that maximizes the margin with respect to all points, defined as the shorted distance between the hyperplane and any of the negative or positive point. If the training data are linearly separable (this hyperplane exists), one can find a pair of hyperplanes such that no point lie between them and the following constraints are satisfied:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geqslant +1 \qquad \forall y_i = +1 \tag{8.3}$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leqslant -1 \qquad \forall y_i = -1 \tag{8.4}$$

that can be combined into:

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geqslant 0 \qquad \forall i \tag{8.5}$$

It is easy to demonstrate that the hyperplane with largest margin can be found by solving the following convex optimization problem:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}||\mathbf{w}||^2 \\ &\text{subject to} && y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geqslant 0 \quad \forall i \end{aligned} \tag{8.6}$$

This problem is feasible only under the assumption that such hyperplane actually exists. However, it may not be the case even if we know that the data should be linearly separable considering the presence of outliers or noise that may hinder that assumption. To this extent it is common to relax the constraints (8.3) and (8.4) by introducing positive slack variables $\xi_i, i = 1, \ldots, \ell$ transforming the formulation with the one proposed in [71]:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{\ell} \xi_i \\ &\text{subject to} && \mathbf{x}_i \cdot \mathbf{w} + b \geqslant +1 - \xi_i \qquad \forall y_i = +1 \\ & && \mathbf{x}_i \cdot \mathbf{w} + b \leqslant -1 + \xi_i \qquad \forall y_i = -1 \\ & && \xi_i \geqslant 0 \quad \forall i \end{aligned} \tag{8.7}$$

Roughly speaking, the constant $C > 0$ is a weighting term that determine how much we are interested to keep the hyperplane flat against the amount up to which we can tolerate mis-classifications in our training data. By introducing Lagrange multipliers $\alpha_i, i = 1, \dots, \ell$ the constrained problem (8.7) can be reformulated in the so called dual form as follows:

$$\begin{aligned} \text{Maximize} \quad & L_D \equiv \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to} \quad & 0 \leqslant \alpha_i \leqslant C \\ & \sum_i \alpha_i y_i \mathbf{x}_i = 0 \end{aligned} \tag{8.8}$$

and the solution $\mathbf{w}$ can be computed as:

$$\mathbf{w} = \sum_{i=1}^{N_s} \alpha_i y_i \mathbf{x}_i \tag{8.9}$$

Note that, albeit there exist one α_i for each training data, only few (N_s) α_i will be greater than zero. Those points for which $\alpha_i > 0$ are called *support vectors* and lie on one of the two separating hyperplanes. Moreover, switching to Lagrange formulation allows the training data to appear only in the form of dot products between vectors. This is an interesting property that can be used to generalize the method in cases where we want the decision function to be a non-linear function of the data.

Suppose to map the data in some other Euclidean space H through the mapping $\Phi : \mathbb{R}^d \mapsto H$ in which the points are linearly separable. The method can be generalized in term of *kernel function* by observing that is only required to define a kernel K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$.

Several kernels exists in literature with different characteristics. For our application we restricted to the evaluation of the linear kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j \tag{8.10}$$

and the gaussian kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-g||\mathbf{x}_i - \mathbf{x}_j||^2} \tag{8.11}$$

Cursor Identity Classifiers

To associate every device with its blob when *appear* and *stop* event are triggered, we trained two independent SVM-based classifiers. This choice is is due to the fact that the feature set used in the first event is slightly different from the second. Indeed, if the accelerometer and compass data are always necessary to describe the device state in both events, the rate of growth of blob area is relevant only when a new blob appears and the blob velocity is only applicable just after a motion on the table.

	Appear			
Stop	0.70	0.80	0.90	0.95
0.70	0.7840	0.8480	0.9180	0.9571
0.80	0.8470	0.8960	0.9450	0.9714
0.90	0.9100	0.9440	0.9720	0.9856
0.95	0.9415	0.9680	0.9855	0.9928

	Appear			
Stop	0.70	0.80	0.90	0.95
0.70	0.8369	0.9165	0.9752	0.9932
0.80	0.8810	0.9421	0.9833	0.9955
0.90	0.9251	0.9677	0.9914	0.9977
0.95	0.9472	0.9805	0.9955	0.9988

	Appear			
Stop	0.70	0.80	0.90	0.95
0.70	0.8740	0.9523	0.9922	0.9989
0.80	0.9064	0.9667	0.9947	0.9992
0.90	0.9388	0.9810	0.9973	0.9996
0.95	0.9550	0.9882	0.9985	0.9998

Figure 8.10: Improvement in accuracy with sequences of Stop events with different lengths.

Specifically, for the first classifier we collected vectors $\mathbf{x}_i \in \mathbb{R}^d$ composed by d distinct features. The first component of the vector is the angle difference in degrees between the orientation of the blob in the image space and the magnetic north measured by the device. Since a blob is seen as a rounded rectangle, only an undirected axis can be computed from its shape and so the difference is chosen as the minimum angle between the axis direction and the device orientation. It should be noted that just a rough orientation is required, thus small deformations of the observed shape (due to occlusion or to the hand holding the object) should have minimal influence. Once the number of samples s and history length h (in seconds) are chosen, the next $3s$ components of the vector $\mathbf{x}_i$ are just the concatenation of measured values of acceleration with respect to the three axis of the device. Each signal is linearly interpolated and re-sampled s times in the time span defined by h. Last s components of vector $\mathbf{x}_i$ are the measured values of blob area size in pixel, re-sampled in the time interval that spans from the first detection of the blob to the time in which the *appear* event is triggered.

In a similar way, for the second classifier we collected vectors whose first $3s + 1$

components are identical to the first case. Last $2s$ components are the concatenation of measured values of blob velocity computed by the table during its movement. Again, the measures are trimmed and re-sampled to match the corresponding acceleration signals.

Different values of s and h can be chosen to define the trade-off between the dimensionality of the vectors and the descriptiveness of the sensor data. Some possible combinations are proposed and evaluated in the experimental section.

Improved Reliability via Majority Voting

The accuracy of the linear and Gaussian kernel based classifiers for the *appear* and *stop* events is expected to be good enough to get a correct classification most of the time. However, given that several events could happen during the tracking lifespan of an object, a proper technique should be adopted to get advantage of the added information supplied by subsequent classifications. To this end, we propose a very simple majority voting method where a blob that enters the tracking system is first identified through the *appear* classifier, then, if such blob remains consistently tracked by the camera, each possible *stop* event will cast an additional vote that can confirm or deny the initial recognition. Since, in our schema, we trust more the *appear* event, the initial identification is kept as valid if the votes against are not the strict majority. This method permits to correct initial association error and prevents further individual wrong classifications from compromising the correctness of the association.

It is interesting to analyze this approach from a probabilistic point of view in order to asses the improvements that are to be expected by applying such correction measure. First of all, we define with the symbol P_a the accuracy of the classifier of the *appear* event and with P_s that associated to the *stop* event. The probability to observe exactly i correct *stop* classifications over a total of k events can be computed according to the Binomial distribution:

$$P_{exact}(k,i) = \binom{k}{i} P_s^i (1 - P_s)^{k-i} \tag{8.12}$$

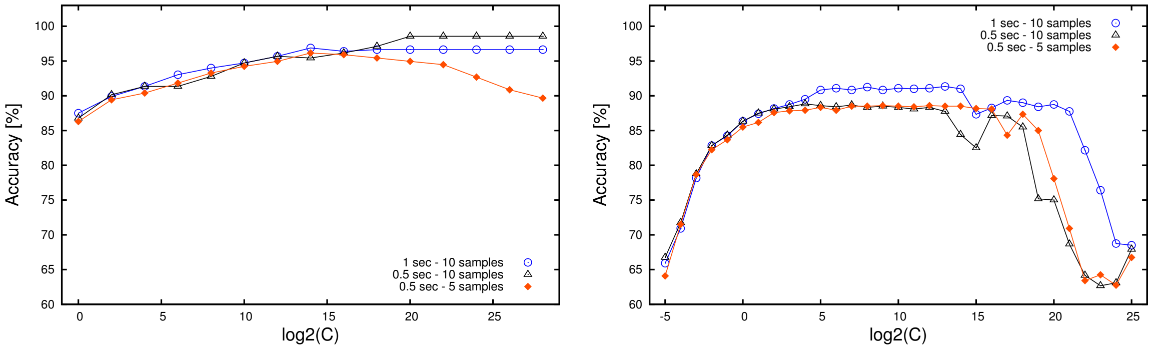


Figure 8.11: Evaluation of the accuracy of the linear kernel SVM classifier for both the Appear and Stop events.

Since all the $P_s(ki)$ are disjoint events, we can compute the probability of observing at least j correct classifications as:

$$P_{atleast}(k, j) = \sum_{i=j}^{i<k} \binom{k}{i} P_s^i (1 - P_s)^{k-i} \tag{8.13}$$

If we consider the first classification, we could have obtained a correct classification with probability P_a and a wrong one with probability $(1 - P_a)$.

Again, these events are clearly independent, thus the overall probability of getting a correct classification when applying majority voting after k observed *stop* events can be computed as:

$$P_a P_{atleast}\Big(k, \Big\lfloor \frac{k}{2} \Big\rfloor\Big) + (1 - P_a) P_{atleast}\Big(k, \Big\lceil \frac{k}{2} \Big\rceil + 1\Big) \tag{8.14}$$

In fact, at least $\lfloor \frac{k}{2} \rfloor$ correct *stop* event detections are needed to not spoil a good initial classification, while at least $\lceil \frac{k}{2} \rceil + 1$ are required to fix a wrong start.

In Fig. 8.10, we show the expected accuracy of a combined classifier respectively for 2, 4 and 6 correction steps and with respect to a range of different accuracy for the base classifiers.

8.1.5 Experimental Validation

The proposed approach has been tested by using it as a device identification for the multiuser map based art browser described in Sec. 8.1.1. Two quantitative aspects of the system have been studied separately: the performance of the classifier (both with a linear and Gaussian kernel) and the accuracy of the time synchronization protocol used.

Classification Performance

To assess the classification accuracy for both *appear* and *stop* classifiers a set of manually labelled data have first to be collected.

The positive examples (i.e. the ones for which the data refer to a correct device-blob association) are gathered by triggering *appear* and *stop* events keeping only one active device at a time. Data are recorded simulating a normal system usage with just a single device hence ensuring that the only blob visible will be the one produced by that device. Negative examples are collected in a similar way but exploiting inactive devices. Again, data are recorded during a normal system usage but placing on the table only inactive devices and using active devices on a fake non-interactive table. In this way, blobs seen by the table will never refer to their correct device.

We collected a thousand of samples for *appear* and *stop* events to be used for training and testing. To compute the classifier accuracy with respect to linear and Gaussian kernels we performed a K-fold cross-validation to our data. In k-fold validation the data set is randomly partitioned into k sub-samples. $k-1$ sub-samples are used to train the classifier

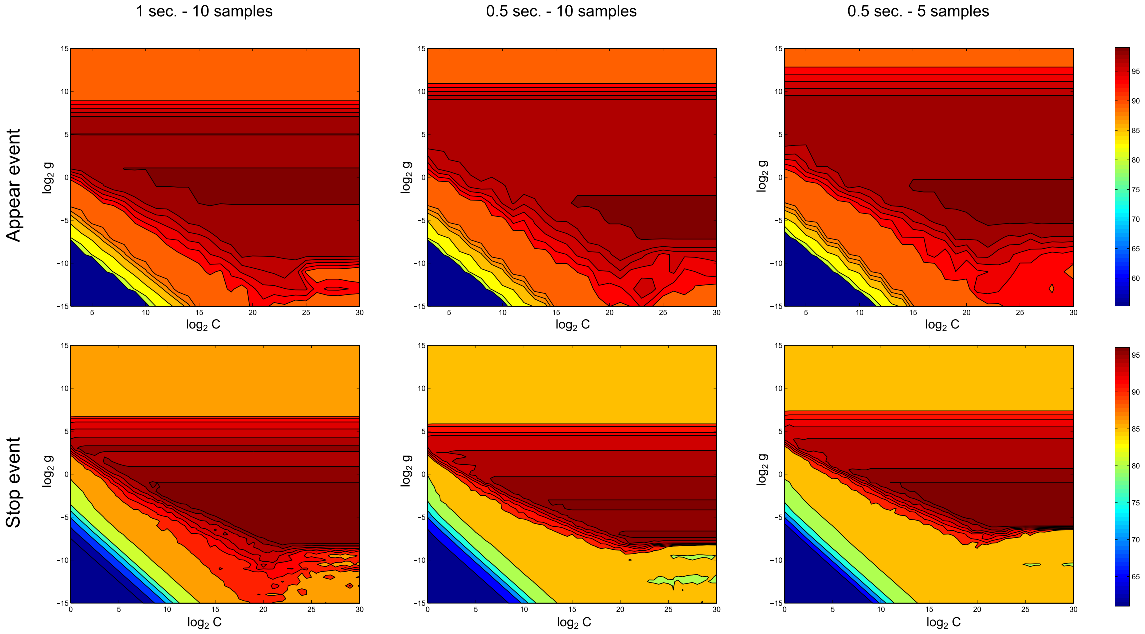


Figure 8.12: Evaluation of the accuracy of the Gaussian kernel SVM classifier for the Appear and Stop events.

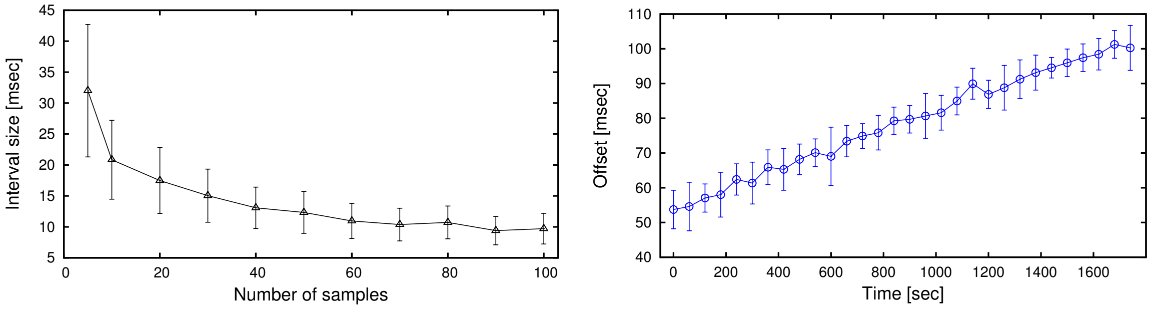


Figure 8.13: Evaluation of the accuracy of the time synchronization with respect to the number of samples taken and of the amount of drift between the two real-time clocks with respect to the time elapsed from the last synchronization.

and the remaining sub-sample is used as the validation set to test the learned model. This process is repeated k times and the average accuracy is returned. This limits the overfitting that may occur while learning the model and allows an effective exploration of method parameters. In all our experiments we chose $k = 5$.

For each of the two classifiers, linear and Gaussian kernel have been tested with three different type of data points, respectively using a signal timespan of 1 second with 10 samples, 0.5 seconds with 10 samples and 0.5 seconds with 5 samples.

In figure 8.11 the accuracy of two classifiers with linear kernel is shown with respect to the parameter C. The classification performance of *appear* event is better than *stop*, probably for the lower dimensionality of the points that allows the data to be more linearly separable. In the first case the best accuracy is $\approx 97\%$ while considering a signal timespan of half a second. In the *stop* case the best accuracy is $\approx 93\%$ achieved with a timespan of one second. In both cases the value of C is not crucial to obtain good performances demonstrating that the training data are probably well separable into the two classes.

In figure 8.12 the accuracy is examined as function of C and g parameter space. Overall, the non-linear classifier obtains better performance with respect to linear case. Best accuracy of 99.3% is obtained for the *appear* classifier with a timespan of 1 second with 10 samples. However it has to be noted that the portion of the C/g plane for which accuracy is above 95% is wider for a timespan of 0.5 seconds. For the *stop* event, the accuracy is a little bit lower and the best performance is achieved for a timespan of 1 second, which is a behaviour similar to the one that has been found when dealing with the linear case.

This level of accuracy is already good enough to be used in many practical scenarios that are tolerant to a negligible degree of misclassification. However, it should be noted that, according with the probabilistic analysis done in Sec. 8.10, considering the obtained accuracies, the combination of the two analyzed classifiers could easily reach an extremely reliable recognition rate.

Time Accuracy and Drift

In subsection 8.1.3 we described the technique adopted to synchronize the real time clock of the device with the time measured by the PC inside the table. In practice, this boils down to measure as precisely as possible the time offset between the two clocks.

In the right part of Fig. 8.13 we show the effect of the number of samples over the accuracy of the offset measure (i.e. the size of the intersection between all the measured intervals). It can be seen that after as few as 20 samples the accuracy is about 20ms and seems to be asymptotically approaching 10ms as the number of samples increases. An accuracy between 10ms and 20ms is acceptable for our application since it is in the same order of the camera sampling, which happens at 30fps (and thus, every 33ms).

From a theoretical point of view, a large number of samples could easily be obtained by sending time synchronization messages regularly when the device is not transmitting other data. Unfortunately, in practice this is not possible because of the drifting between the two clocks, i.e. the slight but significant difference of the internal oscillators. The drifting between a device and the table has been measured using a sliding samples window. The resulting data have been plotted in the left graph of Fig. 8.13. The drifting seems to be linear with the time (which is of course expected) and its value is in the order of about 50ms over a time span of 20 minutes. This is indeed a large value and it implies that for the synchronization to give reasonable results the probing messages should be exchanged in a few seconds span. Further, given the sizeable drifting, synchronization should happen quite often.

8.2 A 5 Degrees of Freedom Multi-User Pointing Device for Interactive Whiteboards

Interactive whiteboards are nowadays rather common equipments in classrooms as they provide large advantages in terms of expressive power. Despite the radical paradigm shift, their interaction model is firmly tied to the archetypal concept of strokes and gestures over a whiteboard. In this section we introduce a novel pointing device that enables one to escape the surface-based interaction, by means of a robust and occlusion resilient multi-camera 3D tracking. More precisely, we designed a frequencybased active pen. By means of a camera network such pen can be localized in a 3D frame featuring the same 5 degrees of freedom exposed by a real whiteboard marker. Our approach allows for using many pointers at the same time, by reliably assigning an unique and permanent identity to each one. By levering on these capabilities, interaction designers can conceive new and inventive interaction models. A few of them have been implemented within this study and are described in the experimental part of this work.

8.2.1 A Robust 3D pointing Device

The overall conception of our 3D pointing system has been driven by a handful of simple yet strongly characterizing design goals:

- The pointing device itself should be as similar to a pen as possible: small enough to be grasped between thumb and forefinger and quite light to be held for prolonged time without fatigue;
- More than one pointer should be usable at the same time, in order to enable a seamless multi-user interaction;
- The accuracy of the position and orientation assigned to each pointer should be high enough to enable classical whiteboard based actions;
- The pointers should work on a continuous basis, regardless of the position of the users (which can cause occlusion) and the nature of the background (which can produce clutter).

The key idea to satisfy those requirement was to ditch any kind of recognition technique in the spatial domain and to perform the identification within the time domain. To this end we designed a pointer augmented with two infrared LEDs that pulse at a given frequency (see Fig. 8.14). By analysing the signal throughout a sequence of frames we are able to locate each pointer and to recognize its ID. Since we set a $\frac{\pi}{2}$ radians phase difference between the head and the tail LED, it is also easy to assign an orientation to each pointer, totalizing of 5 degrees of freedom. In the remaining part of this section we will assign those degrees of freedoms to the euclidean position of the head LED $[xyz]^T$

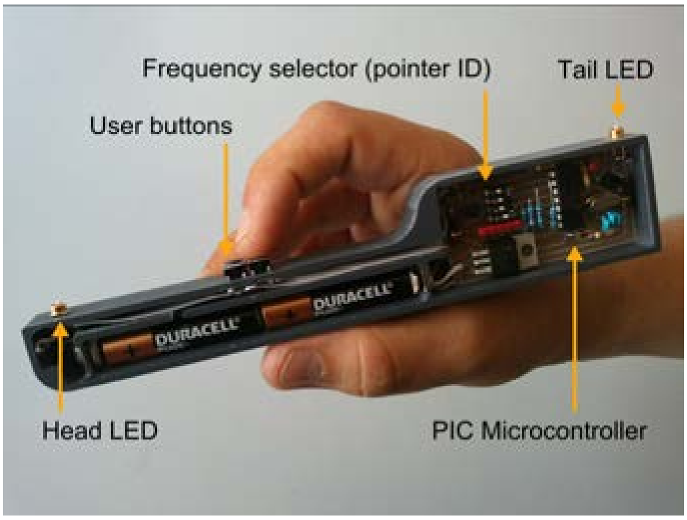


Figure 8.14: Key components of the active pointing device described (see text for details).

and to the direction of the pointer expressed as polar angles $[\theta\phi]^T$. Note that the 6^{th} degree of freedom, that would be the rotation around the pointer axis, cannot be recovered. However this kind of information is not needed for our intended usage. Also note that the current prototype mounts the LEDs on a single face of the device. We did not consider it a restriction on their visibility, since we placed the detecting cameras at ceiling level, and the used LEDs were able to spread a strong enough signal over a full 180 degrees field of view.

The pointer ID is selectable by means of a set of four switches that are accessible on the controller board (see Fig. 8.14), this allows for a total of 16 pointers. The two buttons on the top of the pointer are two additional bits that are chained to the pointer ID to produce a 6 bit word that is used by the microcontroller to choose the frequency of the LED pulse. This way, each pointer can emit up to 4 different frequencies: a base frequency when no button is pushed, and three additional frequencies when one or both buttons are pressed. Out of practical reasons, we assigned the frequencies monotonically with the values of the 6 bit word created with the combination of the pointer ID and of the buttons status. Differently from methods that perform identification by processing object appearance in a single image, our approach allows to use very small features (indeed point light sources) that are detected equally well both near or far from the camera. Of course the identification requires several frames, however, once a pointer has been recognized, its head and tail LEDs can be tracked on a frame-by-frame basis without needing a new recognition as long as the tracked does not miss. Additionally, the small size of the light sources grants a good precision with their localization and the use of infrared light makes it possible to filter out most of the scene clutter. Such accuracy, together with the resilience to occlusion, will be further enhanced by leveraging on a network of cameras that will offer both redundancy and statistical pose validation.

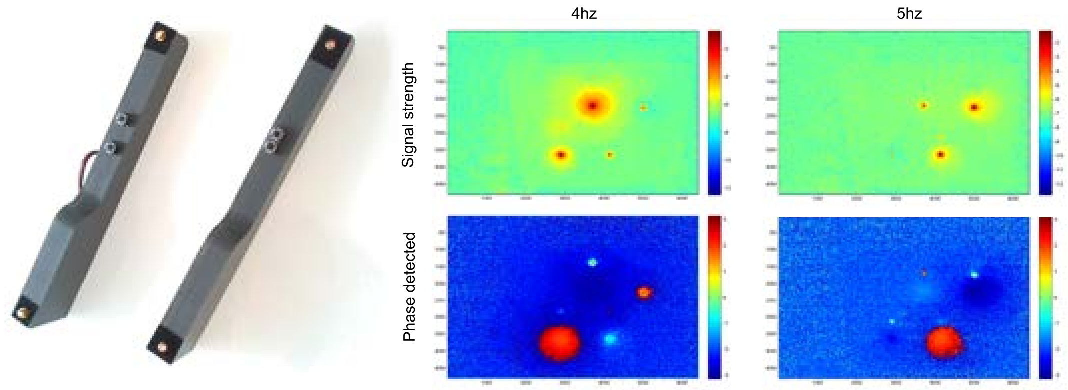


Figure 8.15: Filtering of the signal emitted by two pointers at different operating frequencies. Note that the response strength is highly sensitive to the filtering frequency. Furthermore, phase detection produces garbage data when not applied to a signal with corresponding frequency.

Frequency-based Pen Detection and Recognition

When a camera equipped with the proper infrared band-pass filter is used, only the signal generated by the pointing devices should be detected with non-negligible intensity. Such signal will produce unimodal intensity blobs whose size depend on the distance from the camera, and whose local maximum is located at the center of the LED.

Specifically, in our setup we use some PS3 Eye cameras equipped with a custom filter, modified for infrared, running at 75 frames per second. Each frame acquired is then thresholded with an adaptive method [169]. The resulting blobs are detected and fitted with parametric ellipses using the OpenCV library [43]. Such ellipses are thus refined on the original image with subpixel accuracy [170].

Each of their respective central points p is tracked between subsequent frames and labelled with the timestamp of the frame itself and the intensity of the associated signal. Such intensity has been computed as the average graylevel detected by the camera within a radius of three pixels from the tracked point. We refer to the timestamp (in second) of point p in frame i as $t(p, i)$ and to its intensity as $I(p, i)$.

Since the signal emitted by each LED is characterized by a specific frequency, we are now able to probe the identity of each point p by correlation with a sinus wave of the expected frequency f. This is performed by computing over a total of n subsequent frames, the vector:

$$\mathbf{C_p}(f) = \begin{pmatrix} \sum_{i=1}^{n} cos(2\pi f t(p,i)) I(p,i) \\ \sum_{i=1}^{n} sin(2\pi f t(p,i)) I(p,i) \end{pmatrix} \tag{8.15}$$

The length of $\mathbf{C_p}(f)$ is proportional to the correlation between the signal associated to p and the reference frequency f. Further, the angle between $\mathbf{C_p}(f)$ and the horizontal axis can be regarded as the phase of the signal emitted by p. This two facts offer a practical tool for rejecting false positive points that are not generated by pulsating LEDs, and also for discriminating each pointer from the others in multiuser scenarios.

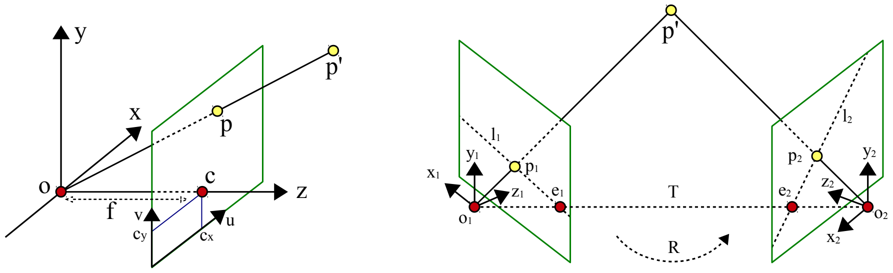
In details, our recognition pipeline performs the following steps:

- A frame is acquired by the camera and the center points of the LED candidates are located and tracked with respect to the previous frame;
- The current timestamp and intensity are added to the history of the successfully tracked points;
- For each frequency f to be probed and for each point p we compute $\mathbf{C_p}(f)$ over an history of n frames;
- If the two points with maximum response are above a given threshold, we consider the phase difference between them. If such difference is near to $\frac{\pi}{2}$ radians the two points are deemed as belonging to the same pointer.

Note that the number of frequencies to be probed and their values depend on the set of pointers that are expected to be in the scene. However, the probing step implies only a handful of computations and it is very fast, even when a large number of simultaneous pointers (and thus frequencies) are to be probed. The number of frames, n, required to compute $\mathbf{C_p}(f)$, depends on the trade-off between accuracy recognition speed. While 3 frames are the theoretical minimum, in our test we found that 9 frames are a good choice to cope with the unavoidable noise coming from the imaging process. Within our setup we provided 16 possible frequencies, spanning from 1hz to 16hz with 1hz incremental steps. The top 16hz frequency is a safe value with respect to the Nyquist sampling theorem, considering the 75hz sampling rate of the cameras used. The 1hz increment was experimentally shown to be large enough to separate well different pointers. In Fig. 8.15 we show the effectiveness of the frequency probing to tell one from the others, and to validate it by means of the phase difference between the two LEDs. We observed two pointers standing on an horizontal surface emmitting signal respectively at 4hz and 5hz. The first row in Fig. 8.31 shows the signal strength computed through Eq. 8.15 (logarithmic scale). The second row shows the detected phase, which is stable and correct only when the LEDs are filtered through the correct frequency slot.

Reliable Estimation of the Pointer Pose from Multiple Views

Since the size of the detected blob depends on the distance of the LED from the camera, a single camera would theoretically grab enough information to assess the 3D position of the pointer. This approach, although not usual within the Computer Vision community, has been deployed in actual commercial products, such as the Playstation Move game controller [209]. Still, given the small size of the adopted LEDs, the high accuracy required and the need for an occlusion-resilient tracking, we believe that a camera network would be a more suitable choice for our applications.



(a)

(b)

Figure 8.16: Illustration scheme of the pinhole camera model adopted (a) and of the triangulation process (b). See text for details.

With the term *camera network* we refer to any number of fixed cameras that are calibrated for both intrinsic and extrinsic parameters. Specifically, we require the distortion coefficients of the cameras to be known (and thus factored out with apt image processing steps) and the focal lengths $(\mathbf{f})$ and principal points $(\mathbf{c_x}, \mathbf{c_y})$ to be estimated with good accuracy. Further, we need to know the position and orientation of each camera, that is the rotation matrix $(\mathbf{R})$ and translation vector $(\mathbf{T})$ that move absolute coordinates to the reference system of the camera. These information allow to compute the image projection $\mathbf{p}$ of a 3D point $\mathbf{p}'$ (in homogeneous coordinates) as:

$$\mathbf{p} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{R}|\mathbf{T}]\mathbf{p}' = \mathbf{K}[\mathbf{R}|\mathbf{T}]\mathbf{p}' = \mathbf{P}\mathbf{p}'$$

where $\mathbf{K}$ is usually called the *camera matrix* and $\mathbf{P}$ the *projection matrix*.

Under these assumptions each camera can be modelled with a simple pinhole-based projective imaging model (see Fig. 8.16-a) where each blob center p is on the line of sight of the actual LED p' through the projective center O. Given a pair of cameras that detect two blob centers p_1 and p_2 (see Fig. 8.16-b), it is theoretically possible to find the exact location of the LED p' by solving a simple linear system. Of course, calibration and measurement errors hinder the ability to find an exact solution for such system, which motivates the need for a triangulation method that minimizes some meaningful error metric. To this end, classical least square is not that popular, since it does only account for an algebraic residual which is not directly related to the actual measures observed. Rather, most of the (very large) literature about triangulation aims to the minimization of the reprojection error, that is the distance (in pixels) between the actually observed blob centers p_1 and p_2 and the reprojections onto the image planes of the estimated point p'. Among the deluge of triangulation methods, we adopted the seminal (but still effective) technique proposed by Hartley and Sturm [107]. This way, we are able to obtain a hypothesis about the position of the LED for each pair of of cameras in the network.

Since with N cameras we can generate up to $\binom{N}{2}$ hypoteses, we gain some level of redundancy to occlusion and of robustness to noise, which can result from some kind of averaging of the independently estimated positions. In principle, the hypotheses could be joined with a mathematical average of the coordinates, however this approach would not benefit from the a priori knowledge of a very significant constraint, that is the physical distance between the head and tail LEDs. If one is only interested in the optimal positioning (with respect to reprojection error) of unconstrained points, simultaneous multiple camera triangulation methods already abound in literature [28, 63, 206]. Differently from such methods, we treat the whole pointer as a rigid object of known length and we proceed to find the rigid transform that minimizes the squared distance between its LEDs and a point cloud of hypotheses. Such cloud is in fact made up by the triangulated positions of the head and tail LED obtained by different camera pairs. This transform can be easily found through the closed-form method proposed by Horn [112]. Finally, since the Horn's registration method seamlessly allows to give a weight to each point in the hypotheses cloud, we exploited for that purpose two information pieces coming from the triangulation process. The first information is the maximum reprojection error committed under hypotesis $\mathbf{p}'$, that is:

$$Re(\mathbf{p}') = max(\|\mathbf{P_1}\mathbf{p}' - \mathbf{p_1}\|, \|\mathbf{P_2}\mathbf{p}' - \mathbf{p_2}\|)$$

where $\mathbf{P_1}$ and $\mathbf{P_2}$ are the projection matrices of the first and second camera that produced the triangulation, $\mathbf{p_1}$ and $\mathbf{p_2}$ are the respective image points. The second error measurement is the time difference (in seconds) between the two acquired frames:

$$Te(\mathbf{p}') = t(\mathbf{p_1}) - t(\mathbf{p_2}).$$

Cameras are not guaranteed to be synchronized, hence the triangulation can be performed between two shots belonging to slightly different samples in time. Since the two error sources are independent, we can model the weight of the estimated point $\mathbf{p}'$ as the density of a a mixture of two orthogonal zero-mean Gaussians:

$$w(\mathbf{p}') = e^{-\frac{1}{2}(\frac{Re(\mathbf{p}')^2}{\sigma_{re}} + \frac{Te(\mathbf{p}')^2}{\sigma_{te}})}$$

where σ_{re} and σ_{te} are the standard deviations of the reprojection and time error distributions.

Although the estimation of the pointer pose tends to be very reliable, due to the number of hypotheses made in the camera network, it might be the case that external factors hinder the ability to provide a smooth and continuous tracking of the motion.

For instance, in practical usage scenarios it is quite common to get one or both LEDs occluded by the user during the normal operation as a whiteboard pointer. Moreover, when the object appear and disappear from a particular network camera, the estimated position may change abruptly (due to the averaging implicitly performed by Horn method) resulting in a noisy estimation. To overcome these limitations, we can model the motion of the pointer as a time varying system. The complexity of the state evolution is tackled by considering its state as stochastic (thus including some intrinsic degree of uncertainty) and only observable through noisy *measurements*.

In our formulation, we modelled the unknown state $\mathbf{x} = (x, y, z, v_x, v_y, v_z)^T$ of each LED as a six-dimensional gaussian distribution with mean μ and covariance $\mathbf{P}$. The first three components define the spatial position of the LED, the least three its velocity. We describe the evolution of the state, called *process*, through each discrete time step k, with duration dt, as a linear function of the state at previous step $k-1$, perturbed by gaussian noise $\mathbf{q}$:

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{A}\mathbf{x}_{k-1} + \mathbf{q} \\
\mathbf{q} &\sim N(\mathbf{0}, \mathbf{Q}) \\
\mathbf{A} &= \begin{pmatrix} 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}
\end{aligned}$$

Whenever a camera pair in the network successfully triangulate a LED, a new measurement vector $\mathbf{y}_k = (m_x, m_y, m_z)_k{}^T$ is generated and used to reduce the uncertainty we have about the state of the system. Similar to the process, also measurements are assumed to be perturbed by an additive gaussian noise $\mathbf{r}$. As a consequence, the model that describes how each measurement $\mathbf{y}_k$ depends on the current state $\mathbf{x}_k$ is defined as:

$$\begin{aligned}
\mathbf{y}_k &= \mathbf{H}\mathbf{x}_k + \mathbf{r} \\
\mathbf{r} &\sim N(\mathbf{0}, \mathbf{R}) \\
\mathbf{H} &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}$$

Assuming to know the prior distribution of the state $\mathbf{x}$ at time $k = 0$ and the set of measurements prior of each time step k: $(\mathbf{y}_1 \dots \mathbf{y}_k)$, the task is to estimate the posterior distribution of hidden states $p(\mathbf{x}_k | \mathbf{y}_1 \dots \mathbf{y}_k)$ for each $k = 1, 2, \dots$. The state distribution is assumed to be Gaussian and process and measurements models are linear. Hence, a very efficient closed form solution is given by the *"Kalman Filter"* [121]. During the tracking of the pointer, the state is estimated at a fixed rate by cycling two consecutive steps: *prediction* and *update*. In the *prediction* step the predicted distribution at time k is computed from the one at time $k-1$ following the process model:

$$\begin{aligned}
\mu_k^- &= \mathbf{A}\mu_{k-1} \\
\mathbf{P}_k^- &= \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q}
\end{aligned}$$

In the *update* step, measurements coming from each available camera pair are assimilated to the current estimate and the posterior distribution is computed as:

$$
\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathbf{H}_k \mu_k^- \\
\mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \\
\mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T \mathbf{S}_k^{-1} \\
\mu_k &= \mu_k^- + \mathbf{K}_k \mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T
\end{aligned}
$$

There are many advantages in the usage of a stochastic filtering approach. First, it offers an elegant way to simplify the dynamics of a system (in our case we consider just position and velocity) by formally including uncertainty in the model. Second, a clue of the error introduced by the filter is obtained through the covariance matrix $\mathbf{P}$ that is estimated along with the state. Finally, the amount of filtering can be adjusted by acting on the covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. Process noise is very hard to define in practice because it depends on the actual behaviour of the pointer that may exhibit an unpredictable oscillating trajectory since is moved by a human. On the other hand, measurement noise depends on the characteristics of the acquisition device. It could be estimated exactly if a ground truth of the pointer is known. For our experiments, we just tuned the diagonal values of both matrices by hand (assuming a stochastic independence between each state component) to a good tradeoff between the lag on the filter and the smoothing introduced.

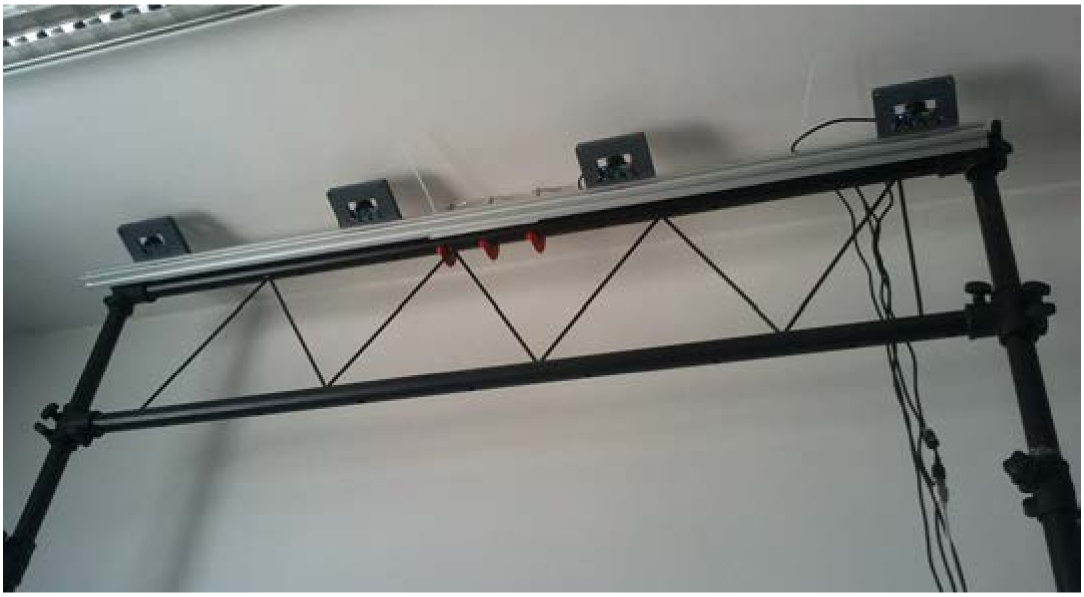


Figure 8.17: The simple camera network rig built for our setup.

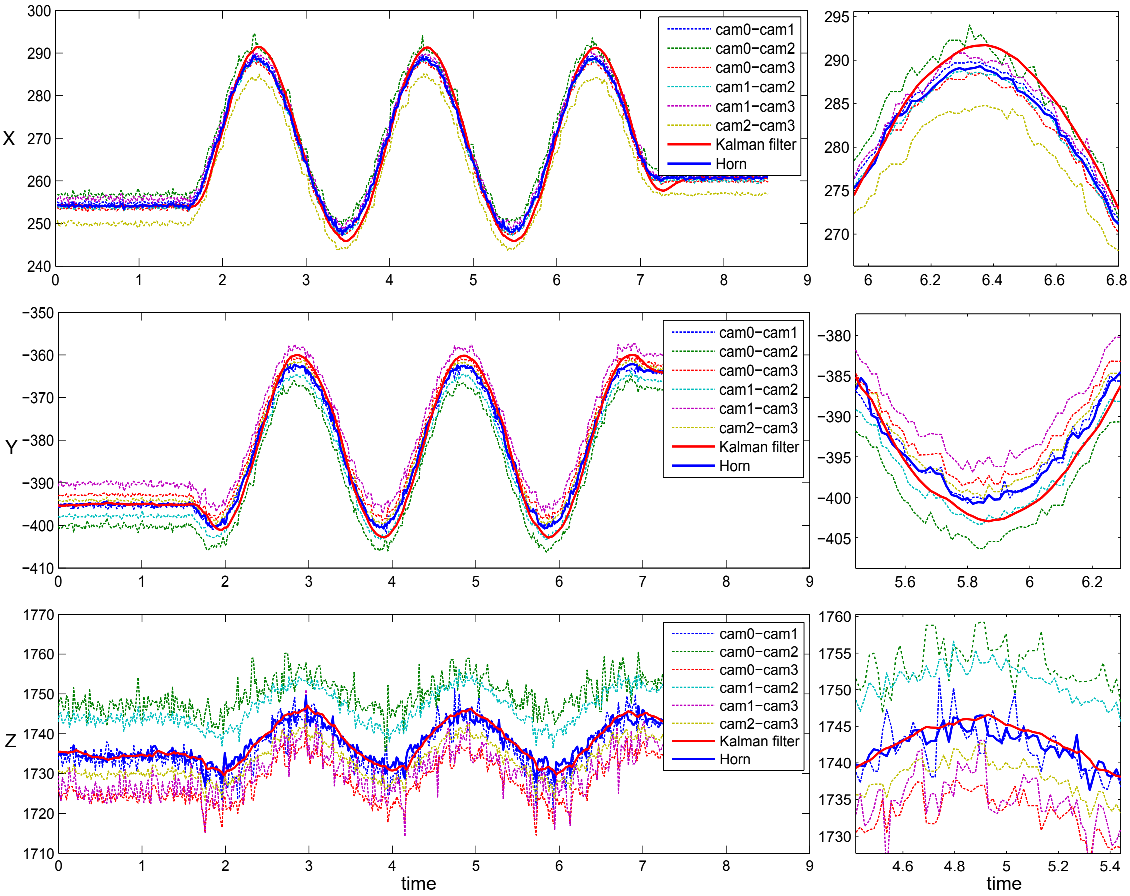


Figure 8.18: Plots of the X, Y and Z coordinates of the pointer triangulated by 4 cameras (6 independent pairs) over a few seconds time interval. Right frames offer a zoomed view of a portion of left frames.

8.2.2 Performance under Real-World Scenarios

To experimentally assess the effectiveness of the proposed pipeline, we built a simple test rig using 4 cameras (see Fig. 8.17). The cameras were calibrated both intrinsically and extrinsically using OpenCV [43]. The obtained extrinsic parameters were further regularized using graph diffusion techniques [217]. The cameras are not synchronized and they run steadily at 75 frames per second.

We can model each frame grab as an uniform random variable between 0 and $\frac{1}{75}$ seconds, hence $Te(\mathbf{p}')$ is the standard deviation of the distance between two such variables. Under these assumption $\sigma_{te} = \frac{1}{75\sqrt{18}} \simeq 3\ 10^{-3}$ seconds. Differently from σ_{te}, there is no easy theoretical method to estimate σ_{re}. For this reason we measured it over a collection of about 20.000 samples obtained by moving the pointer within the view frustum common to all the camera. We obtained $\sigma_{re} \simeq 1.2$ pixels.

In Fig. 8.18 we show the measure of the coordinates of the pointer over a few seconds of capturing. The motion was produced by fixing the pointer to a motorized rotating stage, thus allowing to easily cover a meaningful spatial range with a smooth and unbiased movement. We chose to display three separate plots, one for each coordinate. The noise on X and Y coordinates is much lower than the noise on the Z coordinate. Such difference is explained by the two major error sources (calibration and blob localization) being both amplified by the triangulation along the Z axis of the cameras.

The continuous blue line in Fig. 8.18 shows the result of the rigid registration obtained with Horn method. Exploitation of multiple independent measures and the rigidity constraint allow for a significantly lower variance and thus to a more accurate localization. The continuous red line shows the effect of Kalman filter. The position estimate is clearly smoother, which is a very important feature within interactive applications that include free-hand drawing or writing, object manipulation or remote control of interfaces. This additional smoothness comes at the price of a slight lag and inertia, as clearly shown in the zoomed graphs of Fig. 8.18. In Fig. 8.20, we show a plot depicting the estimated orientation of the pointer as a parametric curve extending over the polar coordinates ϕ and θ. The aforementioned experiments give a semi quantitative idea about accuracy of the pointer localization. However, we are lacking a proper ground truth, since the rotating stage used is not repeatable enough, and it is not accurately calibrated with respect to the camera network.

To produce a strictly quantitative evaluation with a proper ground-truth and to gain a better insight about the measurement error we designed an additional experiment. We fixed two pointers to a rigid bar about 10cm long and we manually moved such artefact inside the whole active volume of the camera network. The measured distance between the two head LEDs was exactly 86mm and the two pointers were mounted parallel one to the other. Note however, that we are more interested in the steadiness of the estimates

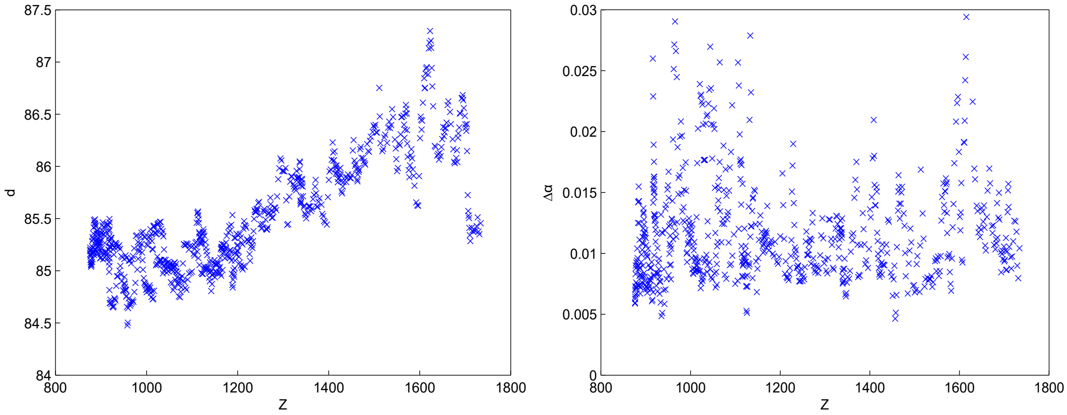


Figure 8.19: Distance (left) and angle (right) between two rigidly linked pointers with respect to the distance from the camera array (Z coordinate).

rather than in their absolute values. In fact, it can be easily shown that an ideal metric measurement system which operates through triangulation is characterized by a null space deformation inside the working volume. To this end, we can ascribe all the fluctuations to detection or calibration inaccuracies. In Fig. 8.19 we show respectively a scatter plot of the distance between head LEDs and angle between the two pointers with respect to the Z coordinate of the first head LED (which is proportional to the distance from the camera network). While the measured gap between pointers seems to be quite stable at a given distance from the camera network, some bias can be observed along the Z axis. The range of such bias, which is probably due to slight inaccuracies with the extrinsic calibration, exhibits a span of about 2mm in the whole volume, which is more than acceptable for any interactive IWB application.

8.2.3 Applications Prototypes

To complete our study, we developed three prototype applications that have been explicitly designed to take advantage of the 3D capabilities of the pointers and of the multiuser nature of the whole system. For each application we used a short throw projector to display the interactive content. We calibrated a rigid transform from the camera network to the projected surface reference frame, in order to get a proper position and orientation for

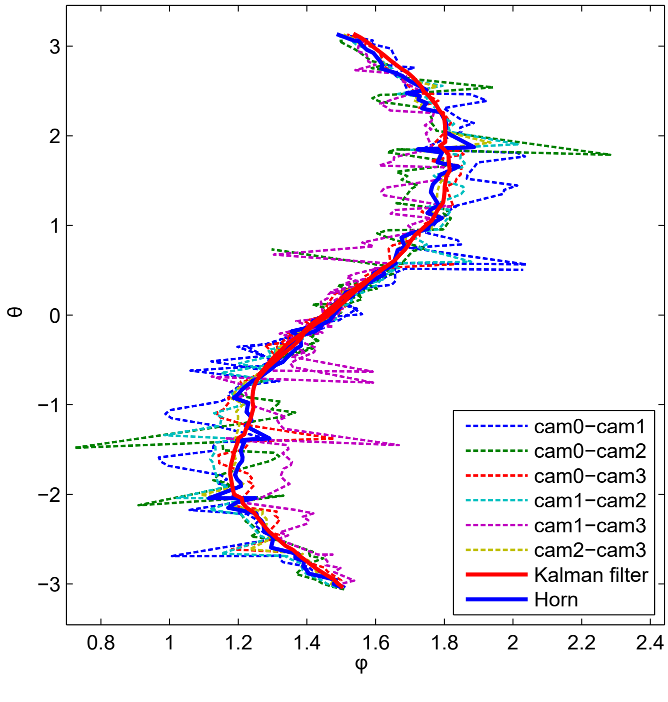
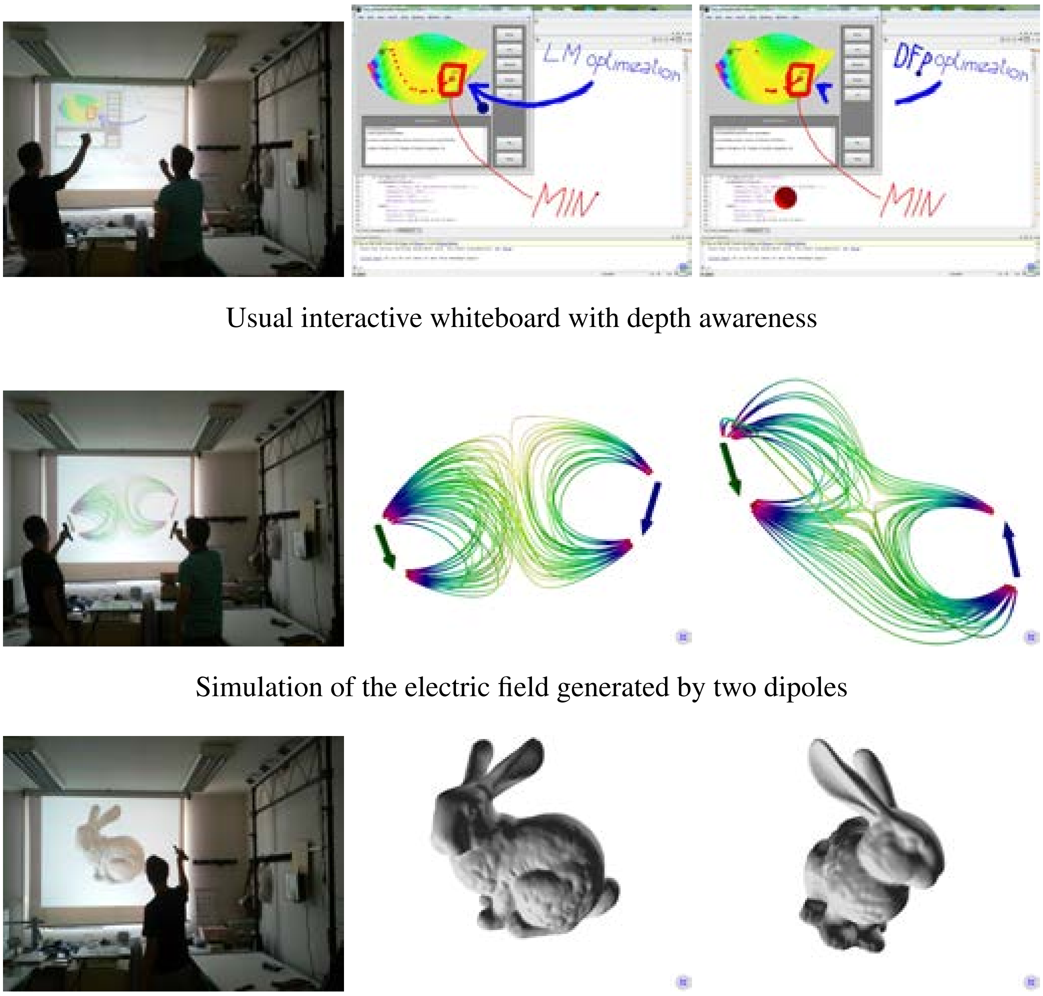


Figure 8.20: Plot of the orientation of the pointer (in polar coordinates) as estimated by each independent camera pair, or by the Horn-based averaging, or through the described filter.



Usual interactive whiteboard with depth awareness

Simulation of the electric field generated by two dipoles

Manipulation of 3D objects using the pointer as a 3D mouse

Figure 8.21: Three examples of possible usages of the described pointers as input devices for IWB applications (see the text for details).

the pointers.

Overlay Writing

The first proof-of-concept application is a simple writing system that allows several users to annotate the screen by hand-free writing. The interaction paradigm is similar to the one found with traditional interactive whiteboards, however the user operates at a distance and his/her pointer does not lay on an actual surface. This introduces additional freedom, as the user is free to move around in the classroom and can operate remotely. For instance, learners and instructors could interact over the same surface while the former does not leave its seat.

We exploited the Z axis as a mean to define the thickness of the line drawn and of the diameter of the eraser (see the first row of Fig. 8.21). Specifically, lines are drawn by pressing the first button of the pointer. Each line is colored according to the user, and it is more thick as the pointer approaches the virtual blackboard. Similarly, lines can be deleted using the second button of the pointer. The size of the eraser is bigger when the pointer is near to the surface.

While the additional freedom of the system opens new usage scenarios, we also noticed that the unavailability of a hard surface upon which pressing the pointer somewhat hinders the ability of the user to write comfortably. We partially mitigated this drawback by adding some visual feedback in the form of colored balls that visually indicate where the drawing or erasing action would take place and which size would change according to the size of the virtual tool.

Physical Simulation

Our second application example is especially crafted to benefit from both the multiuser and 3D capabilities of the system. The two pointers are used to model two electrical unit charge dipoles. Two (or more) users are invited to freely move such dipoles within the working volume of the camera network. As the dipoles are moved, their position is reproduced in a projective scene and the stream lines of the electric field are visualized using a color scale to express the intensity (see the second row of Fig. 8.21).

This is a basic example of a collaborative applications where multiple users cooperate to perform a common task in the 3D space, by means of a real time visual feedback. Other examples could include concurrent editing of cad models, the control of remotely operated devices, or even videogames.

3D Object Manipulation

The last test application adopts the pointer as a 3D manipulation device. A single user moves the pointer and rotates it in front of the display. As a result, the 3D scene or object model shown in the application follows the movements performed by the user, which is thus allowed to intuitively control the object position, rotate it, push it farther or drag it toward him (see the last row of Fig. 8.21).

From a functional standpoint, the pointer is used to perform exactly the same set of actions that could be obtained using a 3D mouse interface, however, in this case, there is a direct physical connection between the position and orientation of the pointer and the manipulated object.

Qualitative survey

Differently from the accuracy of the triangulation, analysed in the previous section, the described applications are difficult to evaluate in a quantitative manner. However, since the final goal for our interactive whiteboard is to be operated by end users, it would still be

	Question	Avg.	Dev.
Overlay	Overall, how much the proposed task has been easy to complete ?	3.91	0.79
	Did you found the accuracy of the pointing device to be adequate for the task ?	4.16	0.71
	Did you perceived a satisfying and timely feedback between the action performed and the system behaviour ?	3.83	1.02
Physical	Overall, how much the proposed task has been easy to complete ?	4.66	0.49
	Did you found the accuracy of the pointing device to be adequate for the task ?	4.83	0.38
	Did you perceived a satisfying and timely feedback between the action performed and the system behaviour ?	4.58	0.66
Manipulation	Overall, how much the proposed task has been easy to complete ?	4.75	0.62
	Did you found the accuracy of the pointing device to be adequate for the task ?	4.83	0.57
	Did you perceived a satisfying and timely feedback between the action performed and the system behaviour ?	4.66	0.77

Table 8.1: Results of a brief survey submitted to a group of students.

useful to assess the perceived quality and usability of the system. To this end we asked 12 volunteer students (8 males and 4 females, aged 19–26) to perform one specially crafted task for each application scenario. Specifically, the tasks were:

- *Overlay Drawing:* draw two circles, the first one using a thin line and the second one with a thick stroke. Note that thickness can be adjusted by moving the pen towards or away from the surface;
- *Physical Simulation:* place the two charged wands parallel and with concordant direction, then turn one wand 90 degrees. Tell which configuration produces the more intense magnetic field. Note that field intensity is coded using color temperature;
- *Object Manipulation:* Rotate and move the rabbit in order to count all the fingers that are present on its paws;

Each user has been briefly instructed about how to perform the task using the pointer, but he has not be allowed to try it in advance. Additionally, no user was able to watch the others perform the tasks.

After completing the tasks, the user was presented with a questionnaire which contained the same three questions for each activity. We decided to use the same questions in order to make the results comparable among tasks and to get some insight about how well the pointer is suited for each one. Each question was asking the user appreciation of several aspects of the experience in a 5 points Likert scale, with appreciation increasing with the answer value. Table 8.1 shows the questionnaire and the answers' average and standard deviation. We observed, for each activity, three aspects of the system. The first one, corresponding to the first question, was a general inquiry aiming at assessing the overall comfort level with the assigned task. The other two questions were designed to evaluate respectively the perceived accuracy and the quality of the feedback (which, in turn, is mostly related to the responsiveness of the system loop).

As can be observed in Table 8.1, the *Overlay* task, albeit receiving a positive average score, resulted to be less easy to complete with respect to the other two tasks. We postulated that this might be due to the difficulties related to performing drawing operations in mid air. Indeed, this is partially confirmed by the (relatively) low score obtained with the third question. To better investigate this aspect we informally asked to the students that gave lowest scores the reasons for their discomfort. Essentially, they confirmed the lack of tactile feedback to be a problem. This is a hindrance associated with the chosen design and it is not clear if it can be mitigated as the user get more accustomed with a contactless interaction model. To this end, we feel that it would be interesting to perform a focused study as a future work. Additionally, it would be also useful to investigate the impact of an artificial feedback mechanism, such as a vibration associated with the contact between the pen and a virtual mid air surface. Still, despite the weak feedback offered to the user, the accuracy of the pointing device was found to be satisfactory. Note that the proposed task was designed to test accuracy through two intrinsic challenges: being able to make the circle regular and properly closing start and end point of the shape.

Regarding the other two tasks (*Physical* and *Manipulation*), the users were almost completely satisfied with all the aspects of the device and of the interaction and the found also the feedback and responsiveness to be more than adequate. We think that this is partially due to the somehow less challenging characteristics of the tasks, which basically are designed adopt the tracked device as a 3D mouse rather than as a complete pointing system.

8.3 Phase-Based Spatio-Temporal Interpolation for accurate 3D Localization in Camera Networks

In this section we propose a simple interpolation schema that can be used to further increase the accuracy of the tracking device introduced in section 8.2 by exploiting its pulsing LEDs to obtain a better synchronization of the cameras involved in the tracking process.

Even with perfect calibration and feature localization, indeed, any triangulation would produce unpredictable results if the imaging devices are not synchronized. This effect is described in Fig. 8.22. An object (the red ball) moving along a line is captured by two non-synchronized imaging devices. Between the shots performed by camera 1 and camera 2 the object position has been displaced by vector S, moving from t_1 to t_2. As a result, the projection of the object on the image plane of camera 1 through its projection matrix P_1 will be P_1t_1 and the projection observed by camera 2 will be P_2t_2. Given the time discrepancy, when the rays passing through P_1t_1 and P_2t_2 are used to compute the original location of the feature, they will not intersect (even with perfect calibration and localization) and the reconstructed 3D coordinates of the ball will not correspond to any of the two original positions (see r in Fig. 8.22). A rather standard way to measure the inaccuracy of the reconstructed position is to reproject it back on the image planes through the same projection matrices P_1 and P_2 and to measure the drift from the original imaged features. Such measure is usually named reprojection error (e_1 and e_2 in figure).

In many practical scenarios camera synchronization is not a problem, since it is easy

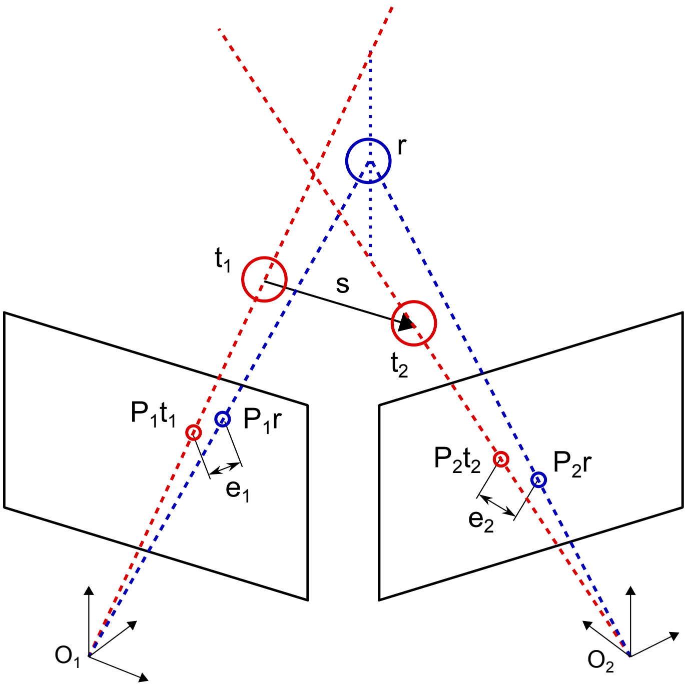


Figure 8.22: A schematic representation of the triangulation error resulting from not correctly synchronized observations.

to build specialized setups where a hardware triggering line guarantees that all the expositions would start and end at the same time. However, this could not be the case with a network of independent cameras, where practical an technical considerations could limit the amount and the type of communication between them. Furthermore, most off-the-shelf low cost hardware lack specialized synchronization controls and require to resort to some software-based solution accounting for time drifts. This is the case, for instance, with the increasingly popular solution based on low-cost embedded solutions such as the Raspberry Pi boards or other kind of acquisition devices, including smart phones. Many algorithms exist to obtain time synchronization between connected devices but network latencies can severely hinder the accuracy, especially on low powered and distributed devices. Finally, even if perfect timer synchronization happens between devices, there is no guarantee (especially with low cost embedded hardware) that the timestamp associated by the device to the obtained frame actually corresponds to the time of physical acquisition. In fact, unpredictable delays due to the capturing hardware and buffering can easily happen.

After reviewing the characteristics of the tracked device device and illustrating the proposed method, we test its relevance with a specially crafted set of experiments. Such evaluation includes both synthetic setups, used to assess the effectiveness of intrinsic synchronization, and a couple of applications, designed to study the performance of the method within real-world scenarios.

8.3.1 Spatio-Temporal Interpolation

The method proposed does not require specialized camera hardware and can work with off-the-shelf components since it base its feasibility on a specific implementation of the tracked device. The overall conception of such device has been driven by a handful of simple yet strongly characterizing design goals:

- The device itself should be as simple as possible and should be easy to implement with different approaches;
- More than one device should be usable at the same time, thus enabling multi-user scenarios;
- The device should offer to the acquiring cameras enough information to perform time synchronization.

The key idea to satisfy those requirement was to ditch any kind of recognition technique in the spatial domain and to perform the identification within the time domain. To this end we propose a design providing two light sources pulsating at the same frequency f. These two light sources are characterized by a phase shift of $\frac{\pi}{2}$ radians, allowing to distinguish between a *head* (with nominal phase $\phi = 0$) and a *tail* (with nominal phase $\phi = \frac{\pi}{2}$). This conventional orientation allows the recovery of device position and attitude up to any rotation around the head-tail axis, for a total of 5 degrees of freedom, which should be enough for most applications. A third light source with a different phase shift

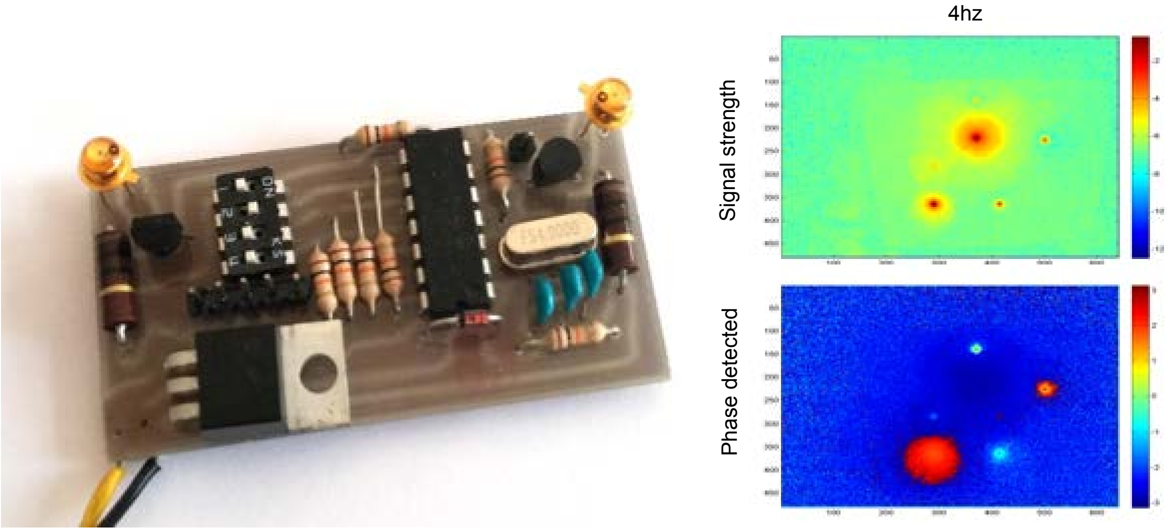


Figure 8.23: A prototype of the pulsating LED device proposed in this section (left) and two plots expressing using JET scale the strength of the recovered signal and its phase (right). Signal strength is shown as the logarithm of the normalized intensity and phase is in radians.

could be added to obtain a complete reference frame, however we feel that this would increase the complexity of the system for a marginal gain. Finally, by using different devices pulsating at different frequencies, multiple objects can be recognized. In Fig. 8.23 we show an actual implementation of the device using two infrared LEDs that can be made pulsating at a total of 16 different frequencies, selectable by means of a set of four switches that are accessible on the controller board. We used this device (and some physical modification) throughout all our experimental evaluation. It should be noted that the use of IR light is not a requirement of the design, nor it is mandatory to use LEDs. In fact, we will also demonstrate the feasibility of an implementation using a smartphone operating with visible light. Indeed the actual implementation of the light-emitting system should be chosen according to the specific application scenario. To this end, we also propose two quite different applications adopting the general approach.

Simultaneous position and time recovery

As stated before, the recognition and time recovery happens over a sequence of several frames. The exact number of frames to be used depends on the trade-off between latency and accuracy. In general we assume to work over n frames. We refer to the intensity of pixel p in frame i as I_{pi} and to the timestamp of frame i as T_i. Note that this timestamp does not necessarily report the exact capturing time and it is not assumed to be coherent among all cameras (due to the many reasons described in the introduction), in fact we will rely on additional information for camera synchronization. However, within the limited span of a few frames, and for each individual camera, we can assume that (locally) the drift due to time bias would be negligible. As the number of frames needed for detection is quite small, this is a reasonable assumption. Since the signal emitted by each light source is characterized by a specific frequency, we are now able to probe the identity of each pixel p by correlation with a sinus wave of the expected frequency f. This is performed by computing over a total of n subsequent frames, the complex number:

$$\mathbf{C_p}(f) = \sum_{i=1}^{n} cos(2\pi f T_i) I_{pi} + i \sum_{i=1}^{n} sin(2\pi f T_i) I_{pi} \tag{8.16}$$

From this number we can recover for each pixel p in the imaging plane both a measure of response intensity $\gamma_p(f) = |\mathbf{C_p}(f)|$ and a phase $\phi_p(f) = arg(\mathbf{C_p}(f))$. In Fig. 8.23 we show, using a JET color scale, an example of the scalar fields $\gamma(4)$ and $\phi(4)$ computed over a sequence of frames depicting two devices placed side-by-side, the first one with an operating frequency of $4hz$ and the other set at $5hz$. Notice that, even with a logarithmic scale, the second device is almost completely filtered out with respect to intensity and the recovered phase is unreliable. This effective filtering can be exploited to recognize each device, one at a time. Specifically, when probing frequency f we define for each pixel p the associated relevant blob as the set:

$$B_p(f) = \{\, q \mid \gamma_q(f) > \epsilon,\ q \sim_{\gamma > \epsilon} p \,\} \tag{8.17}$$

where $q \sim_{\gamma > \epsilon} p$ means that p and q are connected by a path within the 8-neighbor graph built over all the pixels with $\gamma > \epsilon$. Of course relevant blobs are a quotient set of the image plane since $q \in B_p(f) \Rightarrow B_q(f) = B_p(f)$. For each relevant blob we can compute its intensity and phase as:

$$\Gamma(B_p(f)) = \sum_{q \in B_p(f)} \gamma_q(f) \tag{8.18}$$

$$\Phi(B_p(f)) = \arg\left(\frac{1}{|B_p(f)|} \cdot \sum_{q \in B_p(f)} \exp(i \cdot \phi_q(f))\right) \tag{8.19}$$

The global intensity for a blob (Eq. 8.18) is not normalized by the number of pixels it contains. This is due to the fact that the blob with higher response are also the one which count a large number of compact members. Differently, small blobs containing a few spiking pixels would generate a lot of false positives. Note also that the complex formulation of Eq. 8.19 is required to avoid instability due to the periodicity of the measure.

We are now able to define the head h and the tail t of the device as two pixels for which the following conditions hold:

$$q \neq h, t \Rightarrow \Gamma(B_h(f)) \geq \Gamma(B_q(f)), \Gamma(B_t(f)) \geq \Gamma(B_q(f)) \tag{8.20}$$

$$\forall q \in B_h(f), \gamma_h(f) \geq \gamma_q(f) \tag{8.21}$$

$$\forall q \in B_t(f), \gamma_t(f) \geq \gamma_q(f) \tag{8.22}$$

$$sin(\Phi(B_t(f)) - \Phi(B_h(f))) > \alpha \tag{8.23}$$

Condition 8.20 guarantees that the selected head and tail blobs are the ones with higher overall intensity. Conditions 8.21 and 8.22 select the two single pixels with higher intensity for each blob. We used this criteria with all the experiments we performed in the evaluation section and we obtained good performances. Note, however, that different rules could be applied to extract head and tail features, including finding the sub-pixel barycenter of each blob or computing an average coordinate weighted on the intensity. Finally, condition 8.23 guarantees that the tail feature exhibits a phase shift of $\frac{\pi}{2}$ radians with respect to the head. Threshold α can be tuned to make the check more or less restrictive.

The interesting thing about our method is that the very same process used for detection is immediately available to compute the time skew between two cameras observing the same device. Given cameras a and b, with observed head phases $\Phi(B_h^a(f))$ and $\Phi(B_h^b(f))$ we can compute the time drift between them as:

$$\Delta_T^{ab}(f) = \begin{cases} \frac{\Phi(B_h^a(f)) - \Phi(B_h^b(f))}{2\pi f} & if\ \Phi(B_h^a(f)) - \Phi(B_h^b(f)) \leq \pi \\ \frac{\Phi(B_h^a(f)) - \Phi(B_h^b(f)) - 2\pi}{2\pi f} & if\ \Phi(B_h^a(f)) - \Phi(B_h^b(f)) > \pi \end{cases} \tag{8.24}$$

Indeed we are doing a clear assumption with Eq. 8.24. We assume that camera a does not anticipate or posticipate camera b for more than half a period, that is $\frac{1}{2f}$. In practice, it is very easy to meet such condition. In fact, for the sampling to be frequent enough to reconstruct the signal, the pulsating frequency of the device should be at least an order of magnitude lower than the sampling frequency of the cameras. This, in turn, means that we are seeking a coarse time alignment between cameras that can tolerate an error measured in several frames. In most scenarios obtaining such level of synchronization is trivial even adopting naive methods.

Image-Plane Interpolation and Triangulation

If we want to triangulate point p_a, observed by camera a, with point p_b captured with camera b at a slightly different time, we must first compute a virtual point $\hat{p_b}$ corresponding to the estimated position on the image plane b of the tracked object at the moment p_a was observed. In order to get a correct estimate we should know the position and direction of the feature in the Euclidean space, to compensate for the non-linearity of projective geometry. Unfortunately, this is exactly what we want to obtain through triangulation. However, since we can assume the displacement of the point to be small between two frames, it is quite reasonable to accept a linear approximation:

$$\begin{aligned} \hat{p_b} &= p_b + (p_b' - p_b)\Delta_T^{ab}(f) fr_b & (8.25) \\ &= (1 - \Delta_T^{ab}(f) fr_b)p_b + \Delta_T^{ab}(f) fr_b p_b' & (8.26) \end{aligned}$$

where p_b' is the tracked point in the following frame of camera b and fr_b is its frame rate (which can be different from the one kept by camera a). Note that, in order to get a more reliable interpolation (and to keep it convex), it would be better to have that $0 \leq$

$\Delta_T^{ab}(f) < \frac{1}{fr_b}$. Indeed, this is an easy condition to meet, in fact it is immediate to show that it is always possible to find an integer number of frames to be skipped or anticipated in camera b to make this happen. Note also that, if the frame rate of camera b is not known or its unstable, we can, once again, exploit the intrinsic synchronization method offered by our approach. Specifically we can estimate such frame rate on a frame-by-frame basis by computing the time shift between camera b and a conventional virtual camera b' that is always one frame ahead of b, obtaining the approximation $\hat{fr_b} = \frac{1}{\Delta_T^{bb'}(f)}$. Once both p_a and $\hat{p_b}$ are known, the 3D position of the point can be recovered by minimizing its reprojection error with respect to the two observed features [107]. Finally, if more than two cameras are available, an estimate feature position can be computed separately for each of them and a multi-camera triangulation method can be used [28, 63].

Implementation Details

Equation (8.16) must be computed for each pixel p of the frame, resulting in the continuous production of matrices representing the intensity of frequency response C (such as these shown in Fig. 8.23). Such computation, at frame number $n+1$, has to be performed over the history of the previous n frames. It is not a problem to have such frames at hand, since they are always stored in a circular buffer or similar data structures (depending on the camera driver). However the number of calculations involved could be quite large. In order to guarantee a real-time performance, we applied the following precautions:

- At each new frame, $cos(2\pi f T_i)I_{pi}$ and $sin(2\pi f T_i)I_{pi}$ are stored in two additional circular buffers. This way Equation (8.16) is computed incrementally by adding values from fame i and subtracting those from frame $i-n$;
- If the frame rate can be assumed to be constant within the n frames, then the values of $cos(2\pi f T_i)$ and $sin(2\pi f T_i)$ can be pre-computed in a fixed lookup table;
- If the camera captures only a small range of discrete intensities (i.e. from 8 to 12 bits), such lookup table could be made a lookup matrix accounting for I_{pi} on rows;

By means of these optimizations, the actual number of per-pixel operations became modest and with our setup (Core i7 4470, 640x480 camera, $n = 8$, single thread) the overhead was well below 1ms. If further performance is needed or the system must be implemented in low powered embedded devices its easy to implement this approach on GPU. This can be done easily even on the entry level ARM processors.

In addition to performance considerations, there is another key practical problem that must be addressed to achieve a robust implementation. Equation (8.16) is able to detect a stable signal only if a significant portion of image pixels captures the same LED throughout all the n frames in the observed sequence. While this does not necessarily imply the marker to be still, if it moves too fast the frequency detection step will miss, hindering the whole pipeline. Luckily this will not result in false positives, since it is quite unlikely that the sampling of random or partially random pixel values would result in a coherent enough signal. Moreover, even if this happens for a large enough blob, the phase shift check would still prevent improper detection. Nevertheless, while false positive are avoided, fast movements will still result in the inability to give a correct identity to the blobs and to assess the associated time drift. In our implementation we address this problem by performing standard blob tracking and, if unable to reliabily assign an identity to a blob, by falling back to the last known reliable label. In such conditions, Δ_T is approximated by assuming the camera framerate to be constant enough during the unreliable frames. Of course, the time drift will be estimated again as soon as the phase recovery will be feasible again. Finally, it should be noted that in our setup we are applying Equation (8.16) to the whole image and using always the image plane reference frame. A different option could be to compute $\mathbf{C_p}$ over each tracked blob, using its center as the origin of the frame. Albeit we did not needed to do so for the tested applications, this precaution does not require any specific methodological change and could help when dealing with never-stopping continuous motion.

8.3.2 Experimental Evaluation

All the following experiments, including the two described applications, have been performed using PS3 Eye cameras equipped with a removable infrared filter, running at 75 frames per second. All the cameras were connected to the same PC, due to the need of obtaining a ground-truth timestamp. This ground-truth can be deemed as accurate since the PS3 Eye cameras are well known for steady and constant capturing feeds and predictable latencies. Still there is no guarantee that two (or more) cameras will shoot at the same time. With all the following experiments we will use the term *GT* to refer to the correct timestamps, *Frame Synch* to refer to the synchronization obtained by assuming that frames have been shot at the same time (which is not the case in many setups) and *Phase Synch* to refer to the phase-based synchronization. Since our method could work with any light emitting device, we first made a test to study the impact of the light source. We modified the device shown in Fig. 8.23 changing the infrared LEDs with white ones, furthermore we also implemented a light emitting application on an Android phone. Such application simply displays two large dots on the screen, simulating the behaviour of the actual device (of course this is not practical in many scenarios, but it is worth testing). We placed the three devices on an horizontal surfaces with a characterizing frequency of $4hz$, and we captured a video of about one minute (5170 frames) by moving the camera

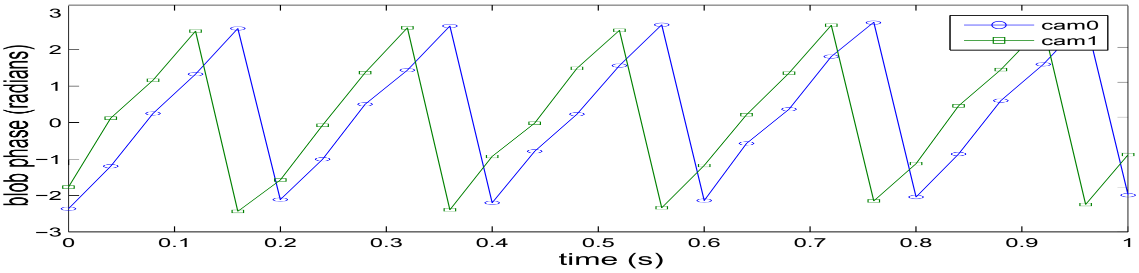


Figure 8.24: Discrepancy between estimated phase and uncorrected frame time.

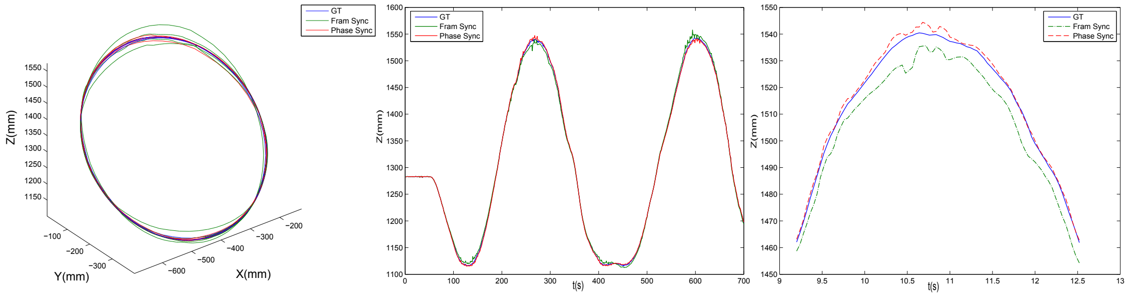


Figure 8.25: Reconstruction of the trajectory of a pulsating LED mounted on a rotating stage (left). Plot of the Z coord (center). Detail of the Z coord plot (right).

(thus obaining similar motions for all the devices). We probed frequencies of $3hz$, $4hz$ and $5hz$, obtaining the following errors over the whole video.

Device	Undetected	%	Misclassified	%
White LEDs	16	0.31	20	0.39
Infrared LEDs	9	0.17	18	0.35
Smartphone	32	0.62	61	1.18

Where *Undetected* means that the device was not recognized at any frequency (probably due to motion blur) and *Misclassified* means that it was recognized at the wrong frequency. According to these results, we chose to use the device based on infrared LEDs for our evaluation. Still, it should be noted that also the other two light sources are able to achieve very good performance levels, thus allowing for several implementations.

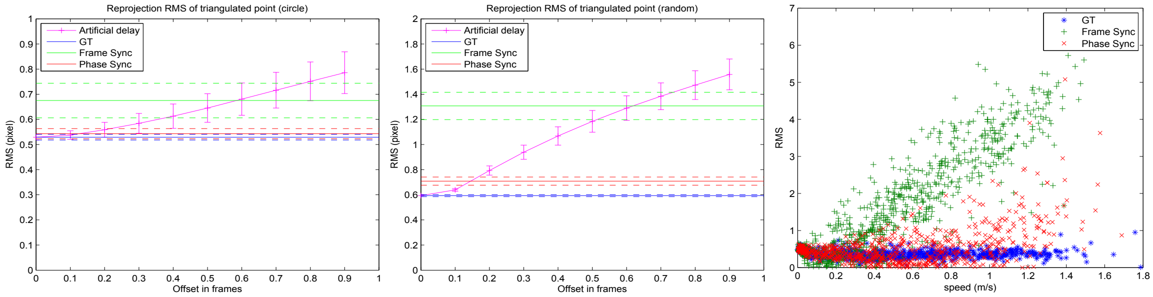


Figure 8.26: Reprojection error with the method described, the ground-truth, frame-based synchronization and artificial delay for circular motion and random trajectories (first two plots), and effect of the speed over accuracy (third plot).

Impact of Wrong Synchronization

In Fig. 8.24 we show the discrepancy between the frame time estimated assuming that the cameras are shooting at the same time and the phase actually recovered by the simultaneous localization an synchronization. It is apparent that the drift is about a couple hundredths of a second. Assuming that the tracked object is moving at a speed of 1 m/s (which is not particularly high) this would result in a spatial displacement in the Euclidean space of several cm, which could be very relevant, depending on the application scenario. To this end, we can state that the drift effect is measurable, relevant and its able to produce significant inaccuracies in position recovery. To qualitatively evaluate the effect of wrong synchronization over position reconstruction we mounted the device on a rotating stage and we captured a long video sequence. We show the trajectory reconstructed using the three described methods in Fig. 8.25. Note that the device performed several ”orbits” during the whole length of the video captured. This has been done to separate repeatable errors due to constant bias (i.e. camera calibration errors) from random uncertainty due to synchronization issues. Here, the trajectory produced without phase-based synchronization is by large the most unstable. Differently, the one produced using the proposed method is more coherent with the ground-truth. The small discrepancies are probably due to phase estimation errors related to the assumption of negligible time drift during the few frames used for phase estimation and, of course, to random sensor noise. In order to give an idea of the impact of time discrepancy, in Fig. 8.26 we plot the average and standard deviation of reprojection error obtained using GT, *Frame Synch* and *Phase Synch* over two large sets of video sequences capturing respectively a circular movement and a random trajectory with varied speeds. In addition we also plotted the error that would be obtained by adding to GT an artificial offset (ranging from 0.1 to 1 frames) to each frame of the video (*Artificial delay* curve). Note that the first three averages are represented

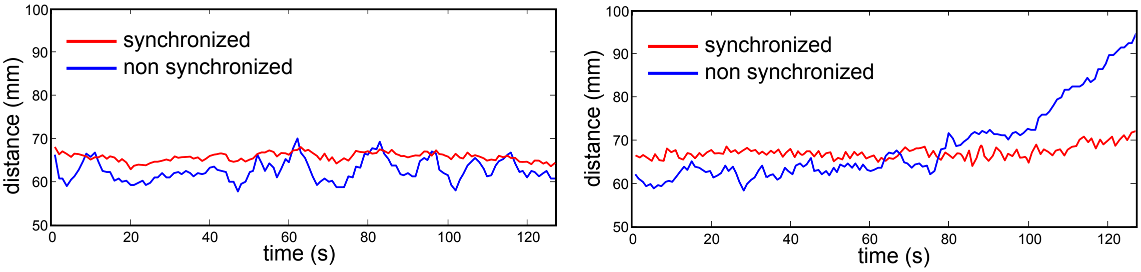


Figure 8.27: Effect of the synchronization on the stability of the tracking

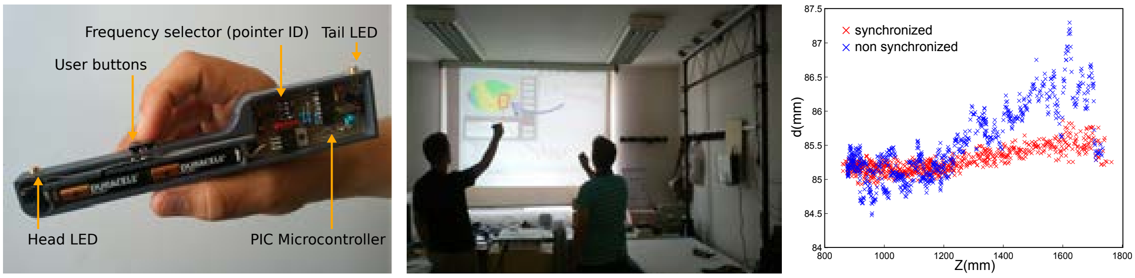


Figure 8.28: An interactive whiteboard system implements using the described design (left). Effect of the synchronization on repeatability (right)

by horizontal continuous lines and their standard deviation use dotted lines of the same colour. Not also that the overall *Frame Sync* values and the *Artificial delay* curve meet around 0.6 frames, which we could deem to be a coarse estimate of the drift between uncorrected cameras. From this test we can see that even a time drift as small as one fifth of frame (which could indeed result very easily from a network of independent cameras) can produce significant inaccuracies.

Effects of Speed

With the last part of Fig. 8.26 we study the effects of speed over reprojection error. To produce this scatter plot we used all the videos captured so far to obtain a wide range of different speeds. It can be seen that the proposed method can (on average) guarantee accurate results even with moderate speeds. Differently, naive frame-based synchronization introduces significant rms errors that seems to be directly proportional to the speed. Regarding the few points with large rms at higher speeds, we think that they are mainly due to image plane localization error due to motion blur rather than to wrong synchronization.

Effect of Distance and Blob Size

It could be interesting to evaluate the breaking condition of the system with respect to the size of the observed blobs, which, in turn, depends on the distance of the device from the cameras.

Distance	2 m	3 m	4 m	5 m
% of undetected markers	0.19	1.42	20.1	40.3
Average pixels in blob	42.1	15.3	7.81	4.12

We performed this test using the device based on infrared LEDs and for distances ranging from 2 meters (the standard usage distance for our setups) to 5 meters (the maximum distance before detection dropped below 50%). The performance seems to be satisfactory until the blobs contain a reasonable number of pixels. As this number drops the detection start to become unreliable with the marker undetected in more than one frame over five for a distance of 4 meters and more two frames over five at 5 meters. Of course different results can be obtained by using bigger or brighter LEDs and higher resolution cameras.

Dealing with a Dynamic Camera Network

Equation (8.24) makes the strong assumption that a coarse drift estimation is available and that it is more accurate than half a period. This condition could be difficult to achieve if cameras are on different networks and/or are dynamically added. To address such cases, we propose a very simple criteria:

- for each new camera b assume max_p to be an upper bound for the integer number of periods contained in the unknown correct value for the time drift Δ_T^{ab} between b and the reference camera a we want to use for triangulation;
- perform a different triangulation for each value of p between $-max_p$ and max_p, assuming drift $pT + \Delta_T^{ab}(f)$ where $\Delta_T^{ab}(f)$ is the last reliable value from Equation (8.24);
- keep the value of p yielding the triangulation with lower reprojection error as the sought coarse approximation.

If more cameras are already in the network, the criteria can be applied for each of them and majority voting can be performed. We tested this criteria by simulating the described scenario using a network of four cameras. All the cameras were connected to the same computer, thus their drift were known. However we added a random artificial delay to the fourth camera in order to simulate general multi-period drift. It is very interesting to note that, differently from identification, coarse synchronization would benefit from movement. Indeed, if the marker is still, there is no disambiguation through triangulation, since all the attempts will produce the same reprojection error. In the following table we show the average percentage of correct coarse synchronizations of the fourth camera (i.e. estimates for p) for different speeds of the marker.

Marker speed	0 m/s	0.1 m/s	0.5 m/s	1 m/s
% of correct estimates	4.7	96.3	100	100

The marker has been placed on a plane at about 2 meters from the network and recorder in a 3 minutes long video. The speed has been estimated over the image plane. The target frequency was $4hz$ and max_p was set to a value of 10. From these results it can be observed that synchronization is basically random when the marker is still, however even slow movements enable a correct estimate. Obviously, coarse synchronization does not need to be performed continuously as reliable values can be kept valid once recovered for a new camera.

Application: interactive whiteboards

The second example application we are proposing is an interactive whiteboard. In this case the controller and the two LEDs have been embedded in a pen-like object that can be used to draw or to interact with the content of the whiteboard (see Fig. 8.28). Moreover, the design has been changed a little in order to include user buttons. Such buttons have been implemented by connecting them to the dip switches of the controller that are used to set the operating frequency, thus allowing to change the device ID (and its function) on the fly. This specific setup relied on a network of 4 cameras. Each pair of cameras was triangulated separately and a rigid registration method has been adopted to estimate the position of the pen [112]. As for the previous application, its is difficult to obtain a reliable ground truth. We performed our evaluation by capturing two pens connected together by a rigid bar an by measuring the distance between their head LEDs. While the exact measure is not known, a well-behaving tracker is expected to yield the same value over time and throughout all the working volume. In Fig. 8.28 we show a scatter plot of such measure with respect to the z axis. The introduction of phase-based synchronization result in a noticeable enhancement of the measure repeatability.

8.4 Design and Evaluation of a Viewer-Dependent Stereoscopic Display

8.4.1 A Viewer-dependent Display System

The setup we are introducing is made up of three main components (see Fig. 8.30). The first one is a planar display. In our case it has been implemented with a short-throw digital projector placed under a translucent scattering surface. The second component is a pair of modified shutter glasses. They have been augmented with two pulsating IR LEDs that can be easily detected with a low false positive rate. The last component is a pair of IR enabled cameras, that will be used to track the two LEDs, and thus the user pose.

The main purpose of this setup is to give a correct and timely estimation of the user point of view and thus allowing to render the scene on the surface in a manner that offers the correct projection with respect to the user. In order to give an idea of the resulting effect we reported some qualitative results in Fig. 8.29. The first two columns show the scene as viewed by the user (putting the camera behind the glasses), while the remaining shots show how the displayed images appear from different points of view.

Tracking of the User Pose

While many other systems use some kind fiducial markers to estimate the user pose, we chose to avoid any kind of recognition technique in the spatial domain and to perform the identification within the time domain. To this end we modified a pair of shutter glasses by adding two infrared LEDs that pulse at a constant frequency (see Fig. 8.31). By using different frequencies we are able to locate each pair of LEDs and to assign an unique

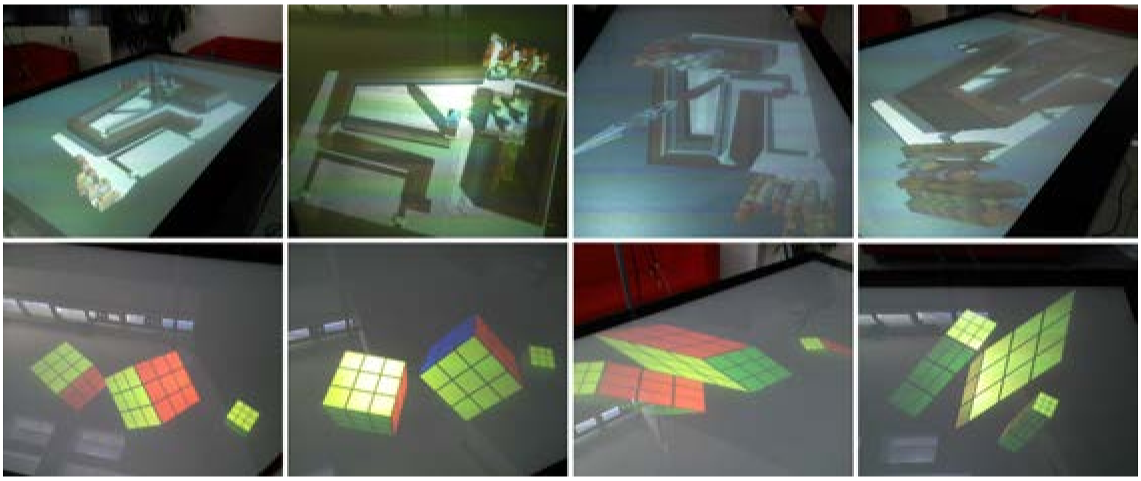


Figure 8.29: An object shown on our viewer-dependent display as seen from different angles. The images in the first two columns have been obtained by putting a camera behind a shutter glass lens.

label to them. Since we set a $\frac{\pi}{2}$ radians phase difference between the left and the right led, it is also easy to discriminate between each eye and thus to give a better estimation of their position. The led frequency is selectable by means of a set of four switches that are accessible on the controller board, allowing several concurrent users. Of course, in order to support more than a single user the shutter glasses and the projector must be configured to show different images for different users. This is supported by many recent mainstream display systems, however it is not the case for the system tested within this study.

Differently from methods that perform identification by processing object appearance in the image domain (i.e. fiducial markers), our approach allows to use very small features (indeed point light sources) that are detected equally well both near or far from the camera. Of course the identification requires several frames, however, once a pair of glasses has been recognized, its LEDs can be tracked on a frame-by-frame basis without needing a new recognition as long as the tracker does not miss. Additionally, the small size of the light sources grants a good precision with their localization and the use of infrared light makes it possible to filter out most of the scene clutter. In fact, when a camera equipped with the proper infrared band-pass filter is used, only the signal generated by the pointing devices should be detected with an intensity that is non-negligible. Such signal will produce mostly unimodal intensity blobs whose size depend on the distance from the camera, and whose local maximum is located at the center of the led.

In our setup we used some PS3 Eye cameras equipped with a custom filter, modified

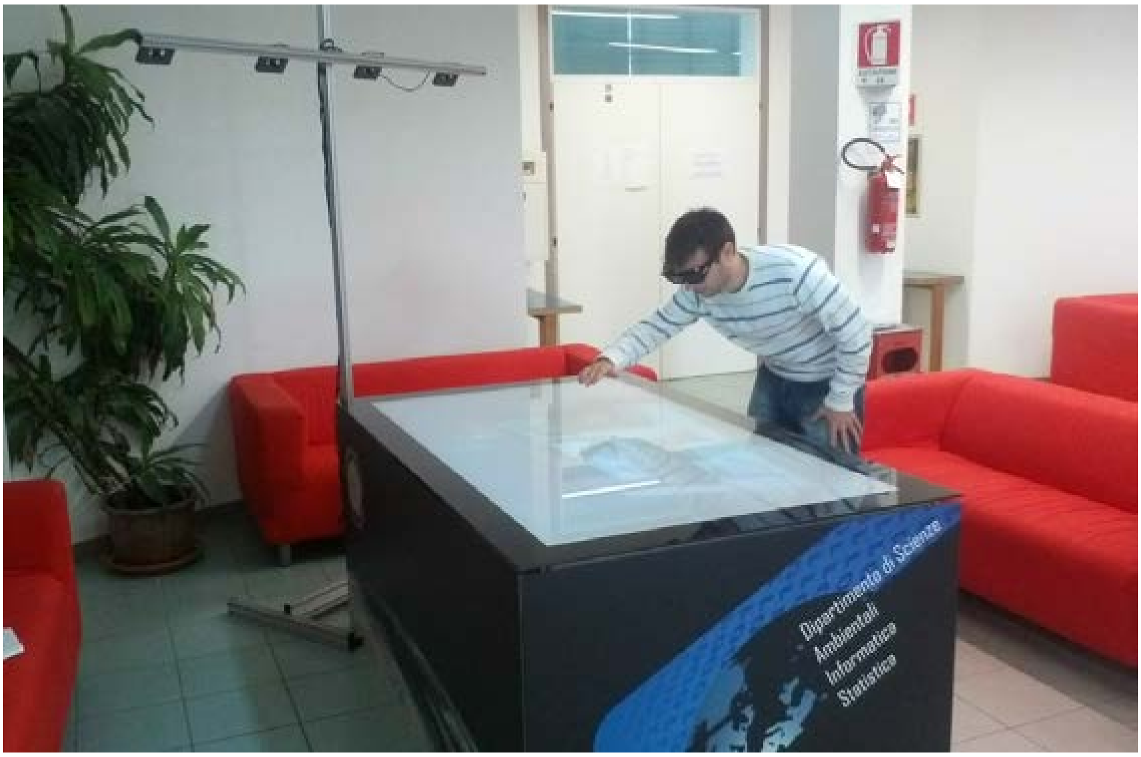


Figure 8.30: The view-dependent display and tracking system setup described.

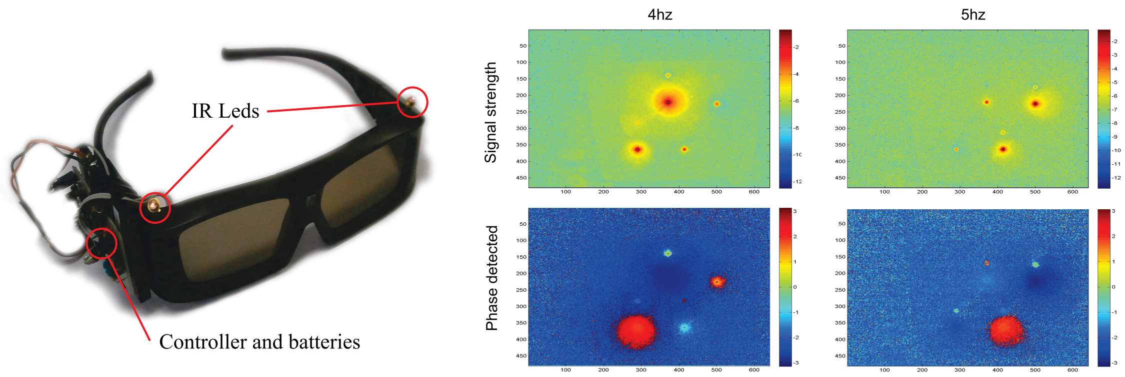


Figure 8.31: Filtering of the signal emitted by two shutter glasses at different operating frequencies. Note that the response strength is highly sensitive to the filtering frequency. Furthermore, phase detection produces garbage data when not applied to a signal with corresponding frequency.

for infrared, running at 75 frames per second. Each frame acquired is then thresholded with an adaptive method [169]. The resulting blobs are detected and fitted with parametric ellipses using the OpenCV library [43]. Such ellipses are thus refined on the original image with subpixel accuracy [170]. Each of their respective central points p is tracked between subsequent frames and labelled with the timestamp of the frame itself and the intensity of the associated signal. Such intensity has been computed as the average graylevel detected within a radius of three pixels from the tracked point. We refer to the timestamp (in seconds) of point p in frame i as $t(p, i)$ and to its intensity as $I(p, i)$.

Since the signal emitted by each led is characterized by a specific frequency, we are now able to probe the identity of each point p by correlation with a sinus wave of the expected frequency f. This is performed by computing over a total of n subsequent frames, the vector:

$$\mathbf{C_p}(f) = \begin{pmatrix} \sum_{i=1}^{n} cos(2\pi f t(p,i)) I(p,i) \\ \sum_{i=1}^{n} sin(2\pi f t(p,i)) I(p,i) \end{pmatrix} \tag{8.27}$$

The length of $\mathbf{C_p}(f)$ is proportional to the correlation between the signal associated to p and the reference frequency f. Further, the angle between $\mathbf{C_p}(f)$ and the horizontal axis can be regarded as the phase of the signal emitted by p. This two facts offer a practical tool for rejecting false positive points that are not generated by pulsating LEDs, and also for discriminating each user from the others in multiuser scenarios. The number of frames, n, required to compute $\mathbf{C_p}(f)$, depends on the trade-off between accuracy recognition speed. While 3 frames are the theoretical minimum, in our test we found that 9 frames are a good choice to cope with the unavoidable noise coming from the imaging process. In Fig. 8.31 we show the effectiveness of the frequency probing. We observed two glasses standing on a horizontal surface emmitting signal respectively at 4hz and 5hz. The first row in Fig. 8.31 shows the signal strength computed through Eq. 8.27 (logarithmic scale). The second row shows the detected phase, which is stable and correct only when the LEDs are filtered through the correct frequency slot.

Finally, the 3D positions of the detected LEDs are obtained by triangulation, using the method proposed by Hartley and Sturm [107] and the positional noise if filtered slightly by using a linear Kalman filter [121].

Viewer-dependent Rendering

After the triangulation of the IR LEDs, the position of each user eye can be determined along the line connecting them. This is a reasonable approximation since the LEDs are placed by construction on the side of the eyes. The exact distance between the LEDs and the estimated projection center of the eyes depends on the interocular distance of the user, which can be kept as a parameter.

For each eye the actual projection can be computed easily if we know its position in the world reference frame which we conveniently place on the display surface and aligned with its two main axis (see Fig. 8.33). In this case, being n and f respectively the z coordinates of the horizontal near and far planes, and (e_x, e_y, e_z) the position of the observer, the projection matrix for a general 3D point can be computed as:

$$\mathbf{P}_m = \begin{pmatrix} \frac{2}{w} & 0 & 0 & -1 \\ 0 & \frac{2}{h} & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & e_x/e_z & 0 \\ 0 & 1 & e_y/e_z & 0 \\ 0 & 0 & \frac{e_z-f}{e_z(f-n)} & \frac{(f-e_z)n}{e_z(f-n)} \\ 0 & 0 & 1/e_z & 1 \end{pmatrix}$$

The rightmost matrix projects the perspective frustum into normalized device coordinates, considering a display plane that lies on $z = 0$ and spans from -1 to 1 on both x and y directions. The leftmost matrix scales and translates the obtained x and y according to the size of the display surface. The matrix P_m can be applied directly with any display framework that supports projection matrices, such as OpenGL or Unity.

Finally, in order to compute the position of the observer with respect to this new world reference frame, we need to know the relative motion between such frame and the cameras used for triangulation. We compute this rigid transformation using the coordinates of the

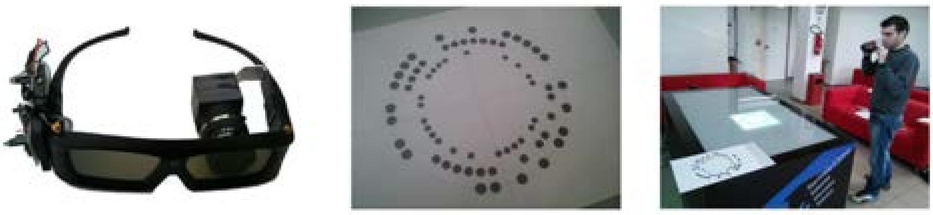


Figure 8.32: The modified glasses and the fiducial marker used for testing purposes.

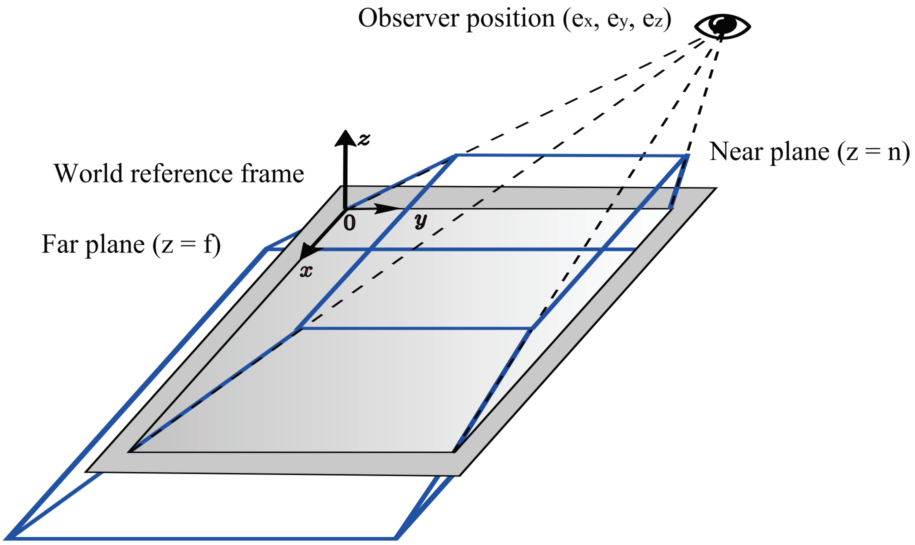


Figure 8.33: The relation between geometric entities within the setup.

four display corners in the cameras reference frames. These coordinates are obtained by triangulation of a single led placed alternately on the four corners.

8.4.2 Evaluating a Viewer-dependent Display

Given the subjective nature of this type of displays, it is very difficult to supply some quantitative assessment about their accuracy (or even to define what really does ”accuracy” mean). In fact, most of the literature limits the evaluation section to qualitative shots of the views or to subjective reporting of the quality perceived by the user. While this is perfectly fine for many application scenarios, we would like to propose a suitable method to measure the performance of a viewer-dependent rendering setup. To this end, we will account for the three main features that characterize this kind of systems: the accuracy of the user pose estimation, the compliance between the scene that the user is expected to observe and what he really sees, and finally the effect of the lag introduced by the whole pose estimation/display loop.

We propose to perform the evaluation by means of a specially crafted setup (see Fig. 8.32). This includes a further modified pair of shutter glasses, which we augmented with a camera mounted behind a lens, and a set of Rune-Tag fiducial markers [**?**]. The measuring experiment is carried on by placing a physical tag on the origin of the world coordinate system (i.e. the upper-left corner of the table) and by displaying a rendered tag inside the virtual scene. That can be in any position and with any angle. The typical experimental run involves the recording of a video while the camera is moving along some pattern. Within such video the camera should be able to capture both the reference physical marker on the table and the virtual marker displayed by the system. For each frame it is possible to compute:

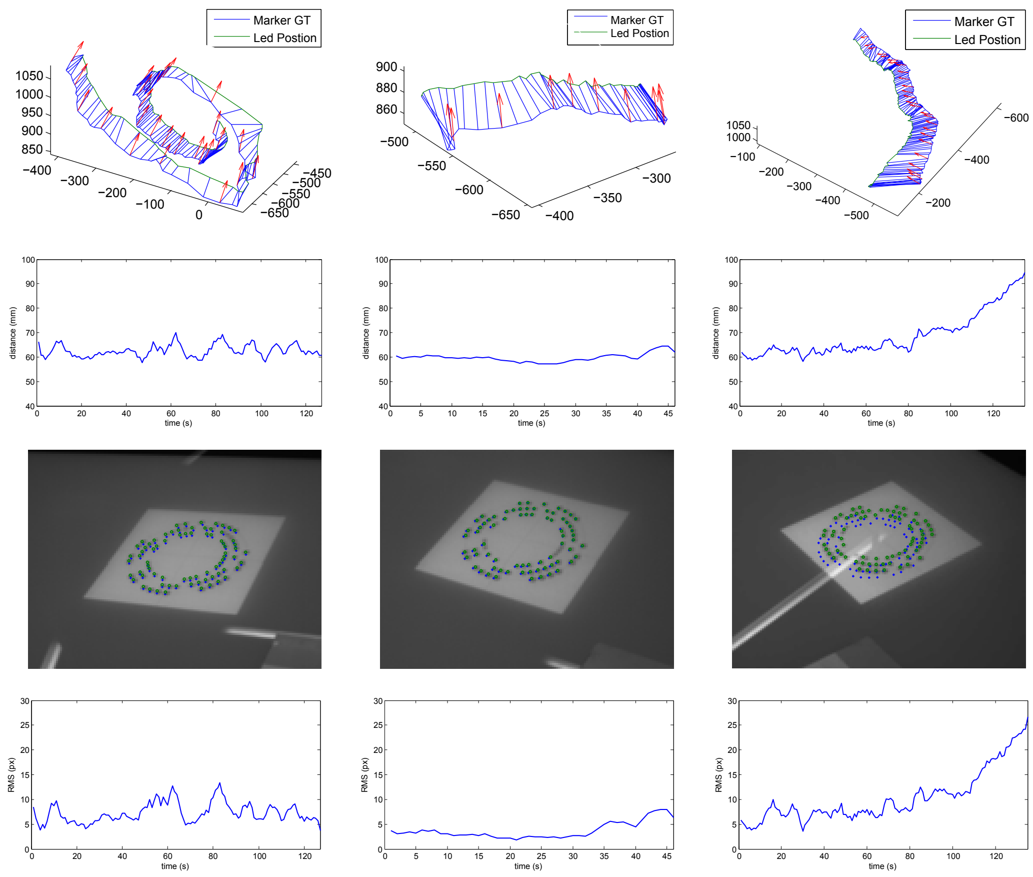


Figure 8.34: Evaluation of the accuracy in the pose estimation and positional error commited on the image plane.

- the pose of the camera center as resulting from the led tracking and triangulation (T_{pose});

- the pose of the camera center obtained using the physical marker (M_{pose});

- the centers of the ellipses on the image plane of the virtual marker as seen by the camera ($C_{centers}$);

- the centers of the ellipses on the image plane of the virtual marker as reprojected by considering the camera pose, its intrinsic parameters and the position of the virtual marker in the world coordinate system ($R_{centers}$). To this end, we use the location of the camera obtained with T_{pose} and the orientation obtained with M_{pose}. This way we guarantee the most faithful orientation of the image plane while still adopting the estimated point of view.

Note that M_{pose} is expected to be significantly more accurate than T_{pose}, since the Rune-Tag used, in opposite to the two LEDs used for T_{pose}, offers more than a hundred of ellipses that can be used to assess the camera pose. Moreover, errors in M_{pose} only depends on the intrinsic parameters of the camera (which is a high-end computer vision camera with low distortion), while T_{pose} if affected by the intrinsic calibration of each IR camera, by the calibration of their relative motion and also by the estimated location of the world reference frame. For this reasons we can consider M_{pose} as a reasonable ground-truth.

Pose accuracy

We propose to assess the accuracy of the pose estimation as the absolute distance between the camera center computed by M_{pose} and T_{pose}. Note that there is no point in considering the orientation of the camera, since it has no influence in the image formation process on the display. Note also that we expect M_{pose} and T_{pose} to be separated by a constant offset, since we cannot guarantee that the center of projection of the camera lies exactly on the line that goes through the two LEDs. This is also true for the user eyes and is a known approximation accepted by the approach (the effects of such approximation will be evaluated in the following section).

In the upper half of Fig. 8.34 we show this distance measured along three different types of motions: respectively a smooth movement along a curve, a slow movement along a straight line and an acceleration with a rotation around the same axis. The standard deviation of the distance under each condition is a good indicator of the ability of the tracking to be resilient to random error source (albeit still biased). We call this measure *pose accuracy*. In the three videos tested we obtained a pose accuracy respectively of 2.63, 1.72 and 8.67 mm.

Reprojection accuracy

The evaluation of the pose estimation accuracy, while assessing the stability of the tracker, gives little insight about the effects of various error sources on the perceived scene. To better study this aspect, which is the primary goal of a viewer-dependent display system, we propose to compute the RMS error between the points observed by the camera ($C_{centers}$) and the coordinates on the image plane obtained by reprojecting the centers of the ellipses belonging to the virtual marker ($R_{centers}$). In practice, this value gives a measure of the compliance between the scene that is actually observed and the scene that the system expects the user to observe. Ultimately, the reprojection accuracy accounts for all the error sources (including the pose estimation bias) and supplies a value that is meaningful also from a perception perspective.

In the lower half of Fig. 8.34 we show this measure, that we call *reprojection accuracy*, computed over time within the same videos that were used for measuring the pose accuracy. In the three video tested we obtained an average reprojection accuracy respectively of 7.06, 3.35 and 9.97 pixels (over a 1280x1024 pixels image).

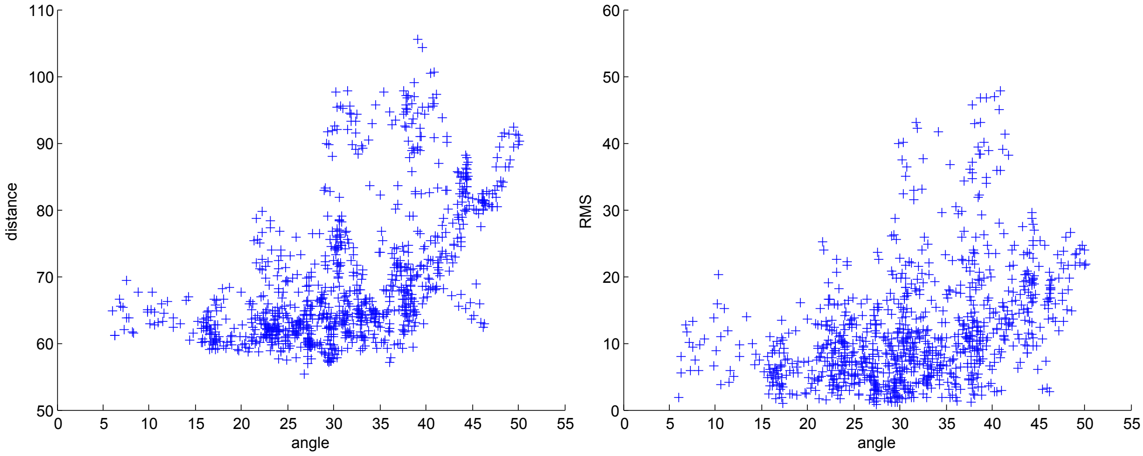


Figure 8.35: Evaluation of the effects of the angle between the projection on the epipolar plane of the line connecting the two camera centers and the two LEDs.

Other evaluations

We propose two additional tests that can be used to evaluate the performance of a viewer-dependent display system.

The first one is the study of the observed led distance and of the reprojection RMS with respect to the angle between the line between the two camera centers and the one connecting the two LEDs (projected on the epipolar plane). In Fig. 8.35, we show that at grazing angles both measures become less stable.

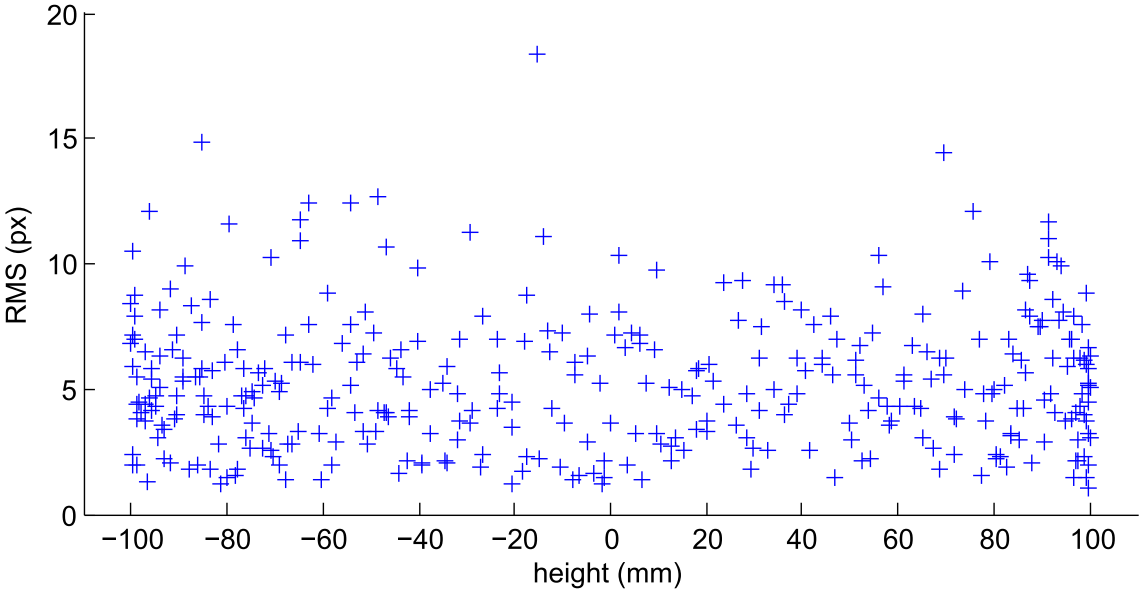


Figure 8.36: Evaluation of the effects on the reprojection error of different heights for the virtual marker.

The second test is designed to measure the isotropy of the perception throughout different zones of the (virtual) scene volume. To perform this evaluation, the virtual target has been randomly placed at various heights and the reprojection accuracy has been measured. In Fig. 8.35, we show a scatter plot of the reprojection accuracy with respect to the target height. It can be observed that there is no apparent relation between the distance of the target from the display and the perception error.

8.5 Evaluating Stereo Vision and User Tracking in Mixed Reality Tasks

In the context of mixed reality the quality of interaction between a user and a virtual scene is strongly affected by the correct perception of the virtual world and by its realistic integration with the real world. Two key factors enabling a correct perception of a virtual scene are the stereoscopic vision through proper displays showing a different image to each eye, and a correct perspective rendering matching the scene projection point of view with the user head position. A correct 3D perception holds if and only if the user's optical system exactly replicates the one that produced the scene projection; otherwise, the perceived scene will be distorted as the 3D objects reconstructed by the user's brain will diverge from the original in size, position and proportions.

Figure 8.37 shows some examples of correct and wrong perspective renderings. The first two images (top row) have been taken from a point of view corresponding to the projection point of view of the rendering system, and show a correct perspective. The last two images (bottom row) have been taken from point of views not corresponding to the projection point of view, hence the resulting perspective is wrong. The effect can be worsened by the fact that points that would originally project into incident lines of sight would probably result skewed when observed from the wrong point of view. This, in turn, would supply to the brain data about the spatial properties of the scene that cannot be correctly interpreted in any way, resulting in an unpleasant feeling of unreality. These shortcomings, not really addressed by the entertainment industry, are in fact the primary responsible for the fluctuating quality of the user experience in 3D theaters.

Context of the Evaluation

A visualization consistent with the user point of view enables interaction models based on the integration between the real and the virtual parts of a mixed reality environment, where objects and data can be actively inspected in an immersive way and directly manipulated with physical tools, as in tangible interfaces. A display that respects a geometrically correct projection allows the blending and comparison of physical and virtual objects, as they all belong to the same metric space. This, in turn, enables important mixed reality applications within the context of industrial design and prototype validation as well as training systems that mix real objects and tools with virtual ones. To this end, tracking systems are exploited to estimate the position of the user head (and of his/her eyes through interpolation) and to drive the correct rendering of the virtual scene. The resulting system is usually referred in literature as a *perspective-corrected* (or *subjective*) *display*.

Still, for this kind of application to be effective, the user perception of the scene should be good enough for such blending to be seamless, i.e., the user must be able to correctly perceive not only the correct position and perspective, but also the correct size and proportions of the virtual objects he/she interacts with. It is therefore of paramount importance to be able to evaluate these properties in an accurate and objective way. We think that

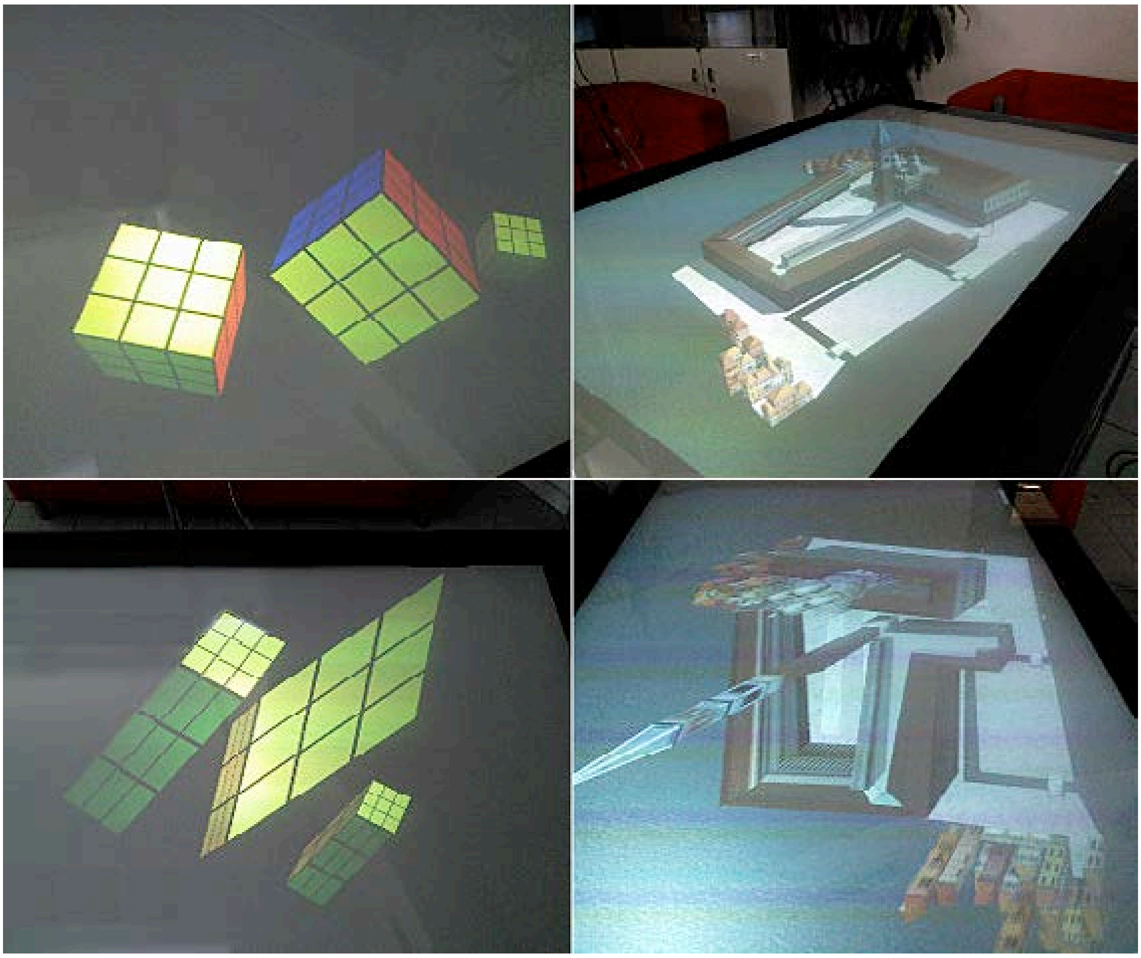


Figure 8.37: Correct view-dependent rendering (top row) and inaccurate scene perception resulting from wrong viewpoints (bottom row).

it is important to separately assess, in a subjective system, the role of each functionality (namely, stereoscopic vision and viewer-dependent rendering) in order to know, in the design and use of a 3D interface, to what extent the different functionalities affect the perception of a correct integration between the real and virtual objects.

We designed an evaluation procedure aiming at measuring the ability to perform some tasks requiring the interplay between real and virtual objects in a mixed reality context. The procedure is executed on an installation implemented using a baseline stereoscopic perspective-corrected display system, where we can accurately calibrate the user position and enable and disable independently each feature: 3D vision and user tracking. In such a way we are able to evaluate their role with respect to the performance of undertaken tasks in a quantitative an statistically meaningful way. While some studies have been recently proposed (see Section 7.3.1), they are for the most part qualitative or focused on specific features of the evaluated system. To our knowledge, this is the first time that a systematic analysis is presented.

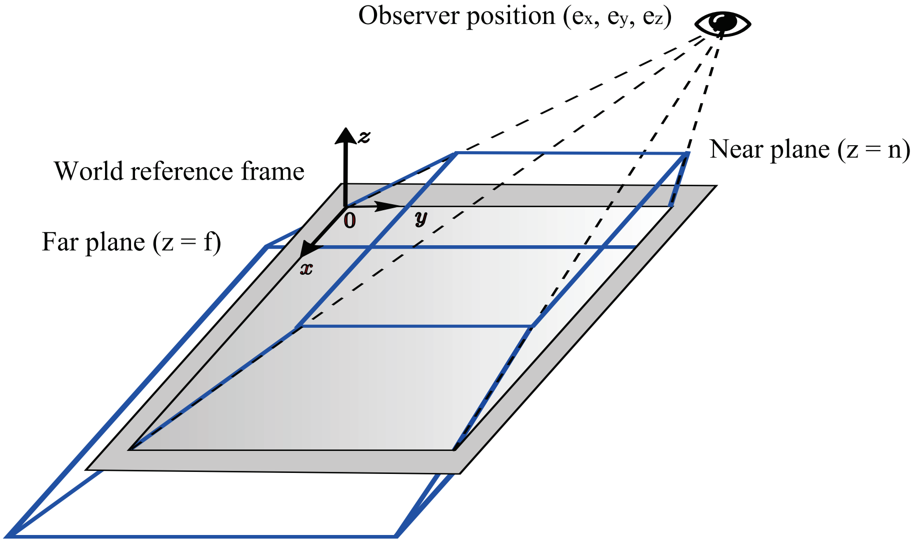


Figure 8.38: The relation between geometric entities within the setup.

Our work applies to environments where the user interacts with a mixed real/virtual scene in proximity, integrating real and virtual objects in the execution of tasks requiring the correct understanding of the spatial relations between them. These conditions largely correspond to the system described by Krueger et al in [133, 134], known as the *Responsive Workbench*, which models a wide range of activities in educational and professional applications.

After a brief description of the setup and projective principles involved in the implementation of the viewer dependent rendering display in 8.5.1, 8.5.2 briefly describes the evaluation procedure used to assess the quality of 3D correct perception. Finally, the actual evaluation on a set of test cases is done in 8.5.3.

8.5.1 Viewer-dependent Rendering

We have used the tracking setup described in 8 to augment a pair of shutter glasses to be used with a commercial DLP projector. Once the position of the two IR LEDs is found, the position of each user eye can be determined along the line connecting them. This is a reasonable approximation since the LEDs are placed by construction on the side of the eyes. The exact distance between the LEDs and the estimated projection center of the eyes depends on the interocular distance of the user, which can be kept as a parameter.

For each eye the actual projection can be computed easily if we know its position in the world reference frame which we conveniently place on the display surface and aligned with its two main axis (see Fig. 8.38). In this case, being n and f respectively the z coordinates of the horizontal near and far planes, and (e_x, e_y, e_z) the position of the observer, the projection matrix for a general 3D point can be computed as:

$$\mathbf{P}_m = \begin{pmatrix} \frac{2}{w} & 0 & 0 & -1 \\ 0 & \frac{2}{h} & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & e_x/e_z & 0 \\ 0 & 1 & e_y/e_z & 0 \\ 0 & 0 & \frac{e_z - f}{e_z(f-n)} & \frac{(f-e_z)n}{e_z(f-n)} \\ 0 & 0 & 1/e_z & 1 \end{pmatrix}$$

The rightmost matrix projects the perspective frustum into normalized device coordinates, considering a display plane that lies on $z = 0$ and spans from -1 to 1 on both x and y directions. The leftmost matrix scales and translates the obtained x and y according to the size of the display surface. The matrix P_m can be applied directly with any display framework that supports projection matrices, such as OpenGL or Unity.

Finally, in order to compute the position of the observer with respect to this new world reference frame, we need to know the relative motion between such frame and the cameras used for triangulation. We compute this rigid transformation using the coordinates of the four display corners in the cameras reference frames. These coordinates are obtained by triangulation of a single led placed alternately on the four corners.

8.5.2 Evaluation of perspective-corrected display on a responsive workbench

The quantitative evaluation of a 3D interface is a complex matter because the features on which the evaluation is done are based on the relations between the virtual scene and the user's physical world; evaluation depends also on the application.

In mixed reality contexts as discussed above, direct interaction and manipulation on a responsive workbench system become possible only if the rendering of the virtual objects seen by the user and the metric relations between them are compatible with the real world so that a comparison between the real and virtual part of a scene is meaningful and quantitatively correct, regardless of the user position within a reasonable area. In environments targeted to passive 3D viewing, such as theaters, the distortion for out of place perspective is less noticeable (yet still annoying), due to the limited range of view angles with respect to the projection size. In mixed reality applications, where the user interacts with the scene in proximity, the relations between the real and the virtual objects demand a precise user adapted rendering.

While the examples shown in Figure 8.37 have been acquired with a camera, in real cases the only capturing device in the loop is the user. For this reason, the effect of various error sources can be detected only by indirect measuring, that is by designing an evaluation procedure where the user is asked to perform a quantitative measure or estimate about what he/she sees.

Evaluation Procedure and Quantitative Metrics

We propose an evaluation procedure for quantitatively assessing the accuracy of perception offered by a 3D system regardless of the specific mixed reality application. For this reason we decided to involve users only in size and distance measuring operations, i.e., to assess the size of some virtual objects or the distance between them in virtual scenes by using a physical ruler. We are thus able to evaluate the accuracy and repeatability of direct measuring in a mixed real/virtual visual field. The rationale of this approach is that the act of measuring a virtual object using a physical ruler captures many of the aspects of a general mixed reality task, such as pointing at objects and space points, aligning virtual and real objects, using metric relations, perceiving depth, comparing sizes, searching and changing suitably the point of view, etc..

To translate the obtained measures into metrics useful for evaluation purposes, we performed the following three steps:

- all the data obtained are converted in relative errors with respect to the *correct* measure of the virtual object. The term *correct* is of course referred to the measure that the object should exhibit in the ideal working conditions of the system;
- a cumulative distribution of the error is computed, obtained by a direct sorting of the values and by computing for each sample the ratio between the number of samples that exhibit an error value smaller than it and the total number of samples gathered;
- finally, an error probability density function (error PDF) is estimated over the cumulative distribution, using a non-parametric Kernel Density Estimator (KDE) based on the Parzen-Rosenblatt window method. This is a rather standard statistical estimator that helps us in getting a more accurate idea about the overall error distribution that underlies the measure processes.

Measuring Bias Once the error PDF has been obtained, we compute the *measuring bias* as the average of such function. This metric expresses the ability of the system to allow the user to perceive unbiased visual representations of the scene, and is proportional to the total amount of systematic perception error.

It should be noted that this metric should be reasonably free from error sources coming from the system since, if the scenes have been designed correctly and no macroscopic errors are present, the translational error introduced by any system bias should not affect distance measurements.

Measuring Repeatability The *measuring repeatability* is computed as the standard deviation of the error PDF. It measures the error dispersion around the average, that is the ability of the system to allow the user to take accurate and repeatable measures.

Differently from the *measuring bias*, with the *measuring repeatability* the system could contribute to the metric. This is the case, for instance, if the estimate of the user head is very unstable and the scene shakes a lot. There is no way to avoid this contamination, however it is reasonable to think that, if the tracking system is designed properly, the instability should be negligible.

Testing Conditions

Regardless of the scenes used for testing purposes, we suggest to evaluate separately the role of stereoscopic vision and tracking. This would help in establishing which feature to include during the system design according to the content that is going to be displayed. To this end we propose to assess measurement bias and measurement repeatability under the following three viewing conditions, which are a subset of the conditions adopted in [232]:

- *Tracked binocular* (*stereo*) *view*: this view corresponds to the standard stereo display mode with the tracking system enabled. Under this condition the only distortions should be attributable to the unavoidable error sources in the tracking system, which in our experiment are small (see later) and produce unnoticeable effects on the test.
- *Untracked binocular view*: in this view condition stereo vision is enabled, but the virtual scene is projected from a fixed point of view without correcting the perspective according to the user position. This view condition provides a standard unadapted stereoscopic content, such as in consumer level movies and video games. The fixed point of view is determined using the initial position of the user standing at the display surface. In practice, the user can perceive an almost correct perspective by moving in search of the better viewing position with respect to the scene fidelity, based on his/her experience. Measure errors derive thus from the inability in finding the correct point of view used to project the scene.
- *Tracked monocular view*: this view condition is dual to the previous one, since stereo vision is disabled, but the perspective is corrected with respect to the user point of view. This is the approach adopted by many systems described in the literature to improve interaction with 2D diplays (e.g., [161]) and is similar in spirit to *trompe l'œil* images drawn on a flat surface. Lacking any stereoscopic vision, the depth perception is lost in principle, and left to cues like motion parallax, leading to larger or smaller measure errors according to the direction of measure with respect to the line of sight.

8.5.3 Experimental evaluation

In order to test the proposed metrics in a practical scenario and to discuss the role of tracking and stereo vision for accurate mixed reality tasks, we used an experimental setup based on a table similar to the *Responsive workbench*, equipped with a stereo retro-projection system and a user tracking system. The choice of a table, i.e., a horizontal surface, instead of a vertical surface like a stereo monitor or a wall screen, was dictated by the need to test the ability to perceive depth and perspective with a greater variability of viewing angles, letting the user moving around the table to observe the scene by several point of view; a screen would have presented to the user an almost central perspective with a much limited range of variations.

The custom built subjective display surface includes two main components: a *stereoscopic display system* made up of a large table (2.5 × 1.5 mt) equipped with a back-projection film where a 3D DLP projector displays a scene. The user observes the represented scene through a pair of DLP-link shutter glasses; a *user head tracking system* made by augmenting the shutter glasses with two infrared LEDs tracked by an array of four calibrated cameras to obtain an estimate of their position in space.

Error Sources Assessment

Putting aside macroscopic issues, such as misaligned cameras or swapped left and right eye frames, we can identify three different error sources:

- *Camera calibration errors*: this is a systematic bias that is due to statistical errors in the calibration procedure for the cameras. In practice, the focal length, center of projection and relative position between cameras are unavoidably estimated with some uncertainty. This, in turn, leads to a misplacement of the user position.
- *Led localization error*: this is a (usually) unbiased positional error due inaccuracies in the localization of the shutter lens LED blobs on the image plane. As for camera calibration error it produces a slight displacement of the observed scene; however its unbiased nature leads to zero mean distortions.
- *System lag*: the limited frame rate of the cameras, added to the projector response and the image processing time, introduces a lag between the user movements and the stabilization of the new viewing position. These distortions disappear completely when the user stops moving.

In practice, all these error sources sum up resulting in a slightly inaccurate estimate of the user head position, which has shown to be dependent on the speed the user moves at. Specifically, the standard deviation of the head pose ranges from a minimum of 2.6cm (when the user does not move) to a maximum of 8.6cm (when the user walks moderately fast at about 2m/s). Such relation between user movements and accuracy loss is due to the delay in the feedback loop between cameras and display. For this reason the increased misplacement only perdures while the user moves continuously and gets back to lower values as he/she stops. Normally, the user is expected to be almost still while performing most precision-bound tasks. Using an artificial marker [**?**, 33] with high repeatability we evaluated the impact of such head misplacement on the actual display surface. This evaluation has been made as objective as possible by attaching an actual camera behind a lens of the stereo glasses and using a registration procedure [19] to estimate the actual position of the marker with respect to the camera. Subsequently, a virtual marker is projected according to the viewing position estimated by the tracker and is observed by the very same camera. Under these assumptions, the average distance between the physical and virtual markers are a good indicator of the overall system-introduced error. Repeated

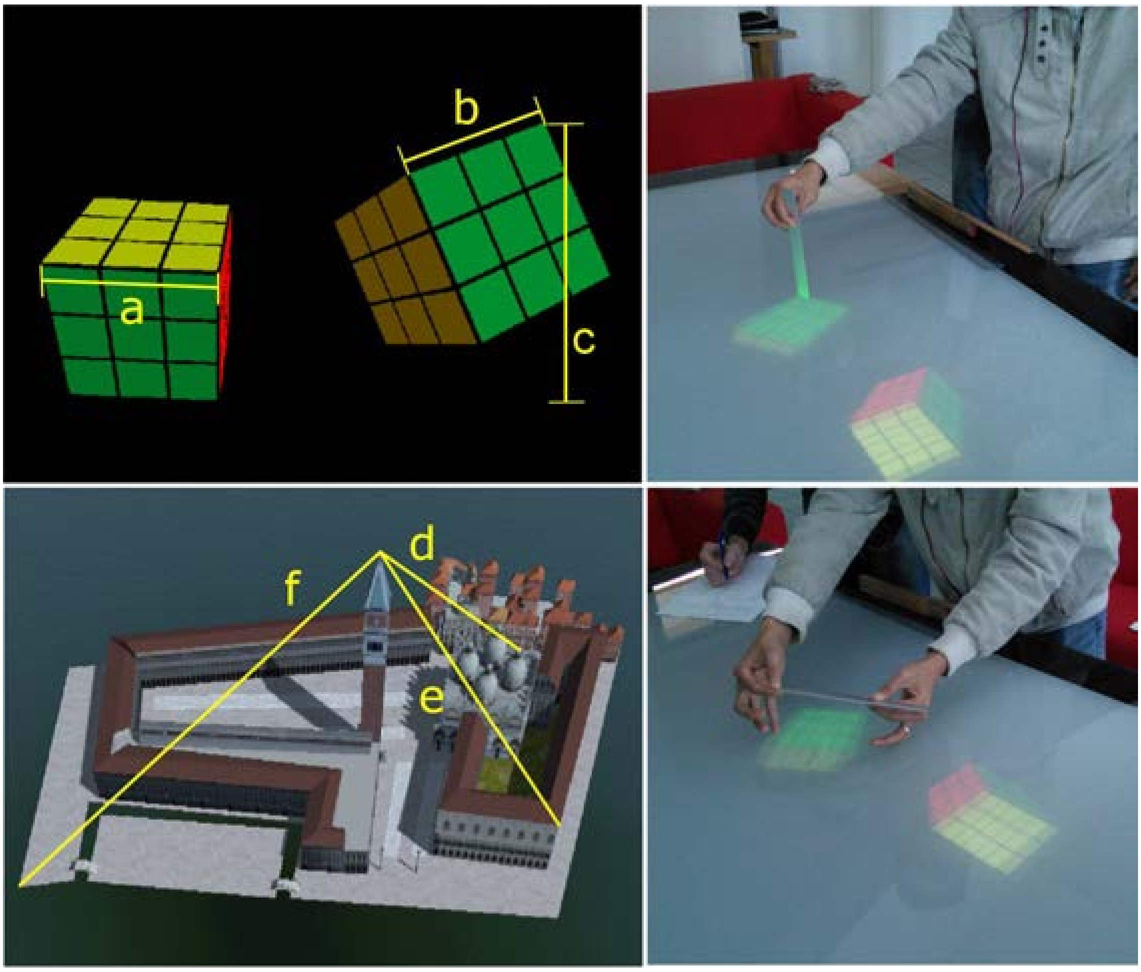


Figure 8.39: Scenes used for the quantitative evaluation. User have been asked to measure (with a physical ruler) the sizes and distances highlighted with the yellow lines.

tests measured an average displacement ranging from 3 to 10 pixels over a 1080p class display. While this can seem a large value, especially with big displays, it must be noted that it mostly results in a global shift of the whole scene: sizes, distances and proportions of the represented objects are practically unaffected. The reader interested in the details of the applied measuring methodology can refer to [72].

Measuring Test Design

We designed the test based on the measure of sizes and distances in two mixed reality scenes using a physical ruler freely chosen among three alternatives (a transparent ruler, a metallic ruler and a carpenter's rule, to adapt to different positions and light conditions). The two scenes have different features. A first scene is composed by two Rubik's cubes with a side of about 10 cm floating a few centimeters over the table surface; the second scene is a synthetic view picturing the famous Saint Mark's Square in Venice, a virtual

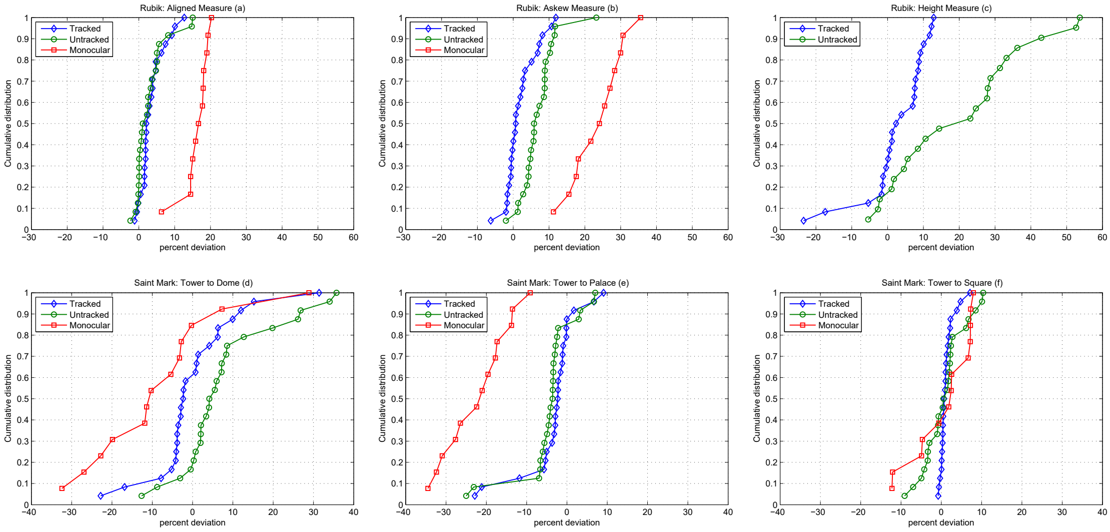


Figure 8.40: Cumulative relative error distributions as measured from direct experimental samples

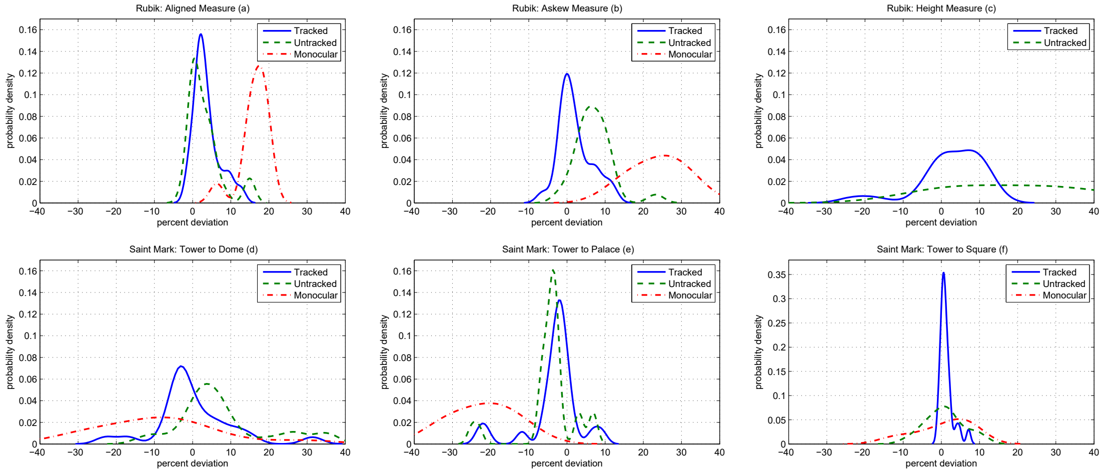


Figure 8.41: Relative error probability densities resulting from kernel density estimation computer over the experimental data.

maquette about 60 cm wide.

For each scene the user had to obtain three measures, for a total of six measures for each execution of the test. Each user performed two consecutive tests, a few made three tests. The tests were repeated under the viewing conditions described in Section 8.5.2 to investigate the distortions affecting the measures under each condition. Figure 8.39 shows the two scenes, the measures requested and two phases of the test execution.

The test involved 11 users (7 males and 4 females) aged 21–27 (avg. 24) for a total of 121 different measures: 60 on the Rubik's cubes scene, 61 on the Saint Mark's Square scene. All the users declared to have some previous experience with 3D stereo glasses and none of them was either stereo blind or color blind; the environmental light conditions were the same for all the tests. During the test the users were free to move looking for the best position to perform the measure.

The measures were almost evenly distributed among the three viewing conditions, with one exception: a height measure in the Rubik's cubes scene under monocular vision was excluded from the evaluation because it did not produce meaningful values; this situation is due to the lack of visual cues coming from motion parallax that could balance the lack of stereopsis. The scene, indeed, has a uniform background, a very simple geometry and no visible marks on the floor, causing the impossibility to find a correct reference point for the measure on the table surface. The reasons for the choice of a scene so unhelpful for the users will be discussed in Section 8.5.3.

For each test the size and position of the scenes were changed by small amounts to avoid memory effects in the users across the tests, and to guarantee measure independence under different viewing angles. Specifically, both scenes where randomly rotated by ± 10 degrees and scaled by ± 10 percent. To avoid the natural tendency to confirm repeated measurements of the same object, we explicitly told users that the scene would change size and orientation after each test. The scenes have been presented in the same order for all the users, starting with the *Rubik* setup. The measures have been performed in the same order as the letters $a - - f$ of Figure 8.39. All the measures were then converted to percentage errors in order to make them comparable among different scenes and different features.

Evaluation of Test Results

The results are shown in Figures 8.40 and 8.41. Figure 8.40 shows the cumulative distribution of the pure data gathered, which is useful to directly analyze the distribution of measurement errors and to obtain an overall idea about where most data are located. Moreover, extrema and outliers are more apparent, as well as the data dispersion. Also, in order to make them comparable, all the errors are shown as percentage offset with respect to the expected value. In Figure 8.41 we present the plots of the corresponding probability density distributions as estimated using a non-parametric Kernel Density Estimator (KDE) based on the Parzen-Rosenblatt window method.

Table 8.2 reports the p-value obtained with a one-way ANOVA test that we performed in order to further validate the impact of different viewing conditions over the proposed scenes and measurement tasks. In the following we comment individually the different cases and the errors revealed. Each case is identified by the name of the scene and the type of measure.

Rubik: aligned side. The first case (size a in Figure 8.39), whose result is plotted in Figures 8.40a and 8.41a, corresponds to measuring the side of a Rubik's cube parallel to the table edge, i.e., orthogonal to the line of sight of a user standing in front of the table.

With this scene we obtained respectively for the tracked, untracked and monocular renderings a measurement bias of 3.6, 3.0 and 16.3 and a measurement repeatability of 3.4, 4.4 and 3.7. In this case both tracked and untracked scene renderings produced accurate measurements, as their error probability density functions exhibit an almost zero-centered distribution and a relatively low deviation. This is due to the fact that the measured cube's side is orthogonal to the view frustum. In fact, the affine transform induced by the lack of tracking is a skew along the subspace complementary to the line of sight, which does not affect at all the segments that entirely lie in it.

Differently, the lack of parallax due to monocular vision severely hinders the measure, showing a clear bias that results in a consistent overestimation of the side length; because of perspective correction, the projection of the cube on the table surface is in fact regularly larger than the virtual object. At the same time, the lack of depth cues leads the user to place the ruler near to the table surface, the only real surface perceived, hence the overestimation. From this first set of observations, we can speculate that tracking is not crucial when the object of interest is orthogonal to the line of sight; on the other side, stereoscopic vision seems essential for properly relate a virtual object with the physical world.

Rubik: askew side In this scene the measure is done along a cube's side askew with respect to the line of sight (size b in Figure 8.39); we obtained a measurement bias of 2.1, 7.0 and 23.8 and a measurement repeatability of 4.3, 4.4 and 7.1, respectively for the tracked, untracked and monocular renderings. This second test confirmed the observations made above: we expect the deformation induced by the lack of tracking to affect the perceived size. In fact, as shown by Figures 8.40b and 8.41b, while the measure made on the tracked rendering maintains an accuracy similar to the previous experiment, the measure made on the untracked rendering has a noticeable bias, due to the slanting of the object if seen from a point of view not coherent with the rendering point of view. Not surprisingly, albeit being correct with respect to perspective, monocular vision is also inadequate, as it leads to very strong bias for the same reasons described in the previous test.

Rubik: corner height. This test is more complex than the previous two tests, as it relates more directly the virtual object with the embedding physical space. The user is asked to measure the height of the cube's topmost corner with respect to the table surface (size c in Figure 8.39). This implies putting the base of the ruler in contact with the physical table and aligning the measuring strip with the virtual cube. Monocular vision is unsuitable for this task due to the lack of depth cues, and no user was able to place the ruler in an even approximately correct position, therefore we excluded this vision condition from the evaluation. For the remaining viewing conditions we obtained respectively for the tracked and untracked renderings a measurement bias of 2.6 and 8.8 and a measurement repeatability of 20.0 and 18.1. As in the previous cases, the tracking in scene rendering is important (Figures 8.40c and 8.41c). While even under correct tracking the measures gathered exhibit a larger deviation, the lack of tracking leads to unreliable observations.

Saint Mark: tower to dome distance. The second set of tests are based on a more complex scene depicting a city landscape: a model of the Saint Mark's Square in Venice. Four points corresponding to architectural elements have been selected as ends of distances to measure: the top of the bell tower, the top of the church's dome, a corner of the palace and a corner of the square (Figure 8.39). For the tower to dome distance (d in Figure 8.39) we obtained respectively for the tracked, untracked and monocular renderings a measurement bias of -0.2, 8.0 and -8.5 and a measurement repeatability of 10.5, 12.3 and 16.0. The Saint Mark's tower to church's dome distance is measured through a slightly skewed angle and the distribution of the measures for both the tracked and untracked case (Figures 8.40d and 8.41d) confirms the conclusions postulated with the skewed Rubik's cube side measure. Monocular view, however, results in both a negatively biased measure and larger data dispersion. With respect to the Rubik's cube test, we believe that the larger error is due to the lack of a visible straight line, like the cube side, helping the user to position the ruler.

Saint Mark: tower to palace distance. With this distance (e in Figure 8.39) we obtained respectively for the tracked, untracked and monocular renderings a measurement bias of -3.4, -4.4 and -22.0 and a measurement repeatability of 7.0, 7.2 and 7.9. This measure is quite similar to the previous one (Figures 8.40e and 8.41e), albeit the line connecting the tower to the palace is a little less oblique, thus allowing for a lower dispersion and a smaller difference between the measures made with the tracked and the untracked renderings, confirming what above postulated.

Saint Mark: tower to square corner distance. This final test is different from the previous two as one end point for the measure, i.e., the square corner, actually lies on the table surface (f in Figure 8.39). Such point is indeed a physical reference, hence it is not affected by errors in tracking or stereo vision. The measuring conditions of this test are close to those of the Rubik's cube height measure, that was however different, because in that case no such reference point exists: being the table surface uniformly colored,

Scene	F	p-value
Rubik: Aligned Measure	53.09	$9.5e^{-14}$
Rubik: Askew Measure	71.1	$2.9e^{-16}$
Rubik: Height Measure	17.42	0.0001
Saint Mark: Tower to Dome Distance	7.55	0.0012
Saint Mark: Tower to Palace Distance	32.2	$3.9e^{-10}$
Saint Mark: Tower to Square Distance	0.09	0.9135

Table 8.2: ANOVA verification of the actual impact of the different viewing conditions on the measure accuracy.

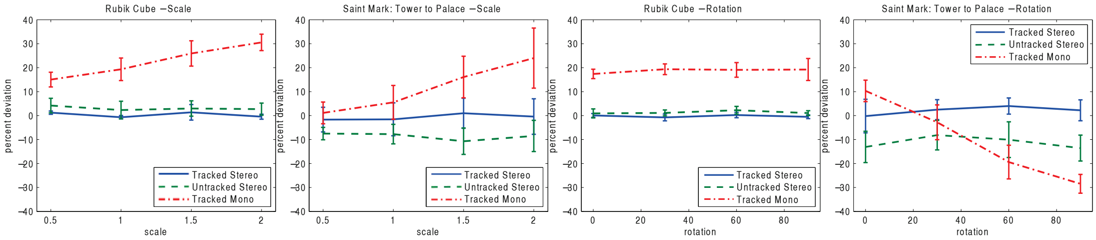


Figure 8.42: Effect of scale and rotation over measurement accuracy.

the user physical reference is in fact an area and not a point. Having a well identifiable reference point simplifies a lot the measuring and reduces the error sources, as shown in Figures 8.40f and 8.41f: all the viewing condition setups were able to produce almost unbiased results (note the different scale of the graph). In fact, we obtained respectively for the tracked, untracked and monocular renderings a measurement bias of 1.3, 0.8 and 0.7. It should be noted, however, that the error dispersion induced when disabling tracking or stereo vision is higher than the one obtained with both tracking and stereo enabled. In fact, the latter condition resulted in a measurement repeatability of 1.8, while the former two respectively of 7.2 and 7.2.

Statistical Validation

With this analysis we are substantiating the statistical significance the influence of different functionalities with respect to the measurement accuracy. To this end, we performed a one-way ANOVA test for each scene and we reported the computed F and the associated p-value on Table 8.2. Within this test, the p-value represents the probability of the null hypothesis, that is the probability that the different conditions have no influence on the ability of the user to perform correct measurements.

Overall the significance of our evaluation and the impact of the different factors studied are confirmed. Indeed, as can be easily observed, the error distributions under different viewing conditions can be deemed to be sharply different for all the scenes and tasks, with the outstanding exception of the last one. However, as can be seen in Figure 8.40, the distributions related to the Saint Mark's tower to square corner distance, albeit characterized by similar averages, are far from being similar. Indeed, this is a clear situation where the ANOVA test fails to detect the real differences between the measurement processes involved. This is due to the fact that such test is designed to identify differences in averages and not in data dispersion. In fact, to get best results, the samples gathered should come from phenomena characterized by similar standard deviations [156], which is a condition that is not met with this scene.

When taking in account this behavior, we can confirm that, while the ANOVA is a standard statistical test, the succinct descriptor that it supplies is not always well suited to capture all the aspects of an empirical evaluation. To get a better insight, the proposed measurement bias and repeatability appear to be a better choice. Furthermore, also the associated PDF can be analyzed as an additional source of useful information.

Effect of Scale and Rotation

Since all the tests have been evaluated using relative errors (expressed as percentages), it would be interesting to investigate if the performance depends on the size of the measured feature, i.e., if the measurement error is indeed relative or is just an offset. Furthermore, it would be interesting to study the effect of the viewing angle over the perception accuracy. To this end we involved 8 users (6 males and 2 females) aged 22–31 (avg. 27) to measure a side of the Rubik cube and the distance of the Saint Mark's tower to the palace with a more ample range of scales and angles. Figure 8.42 shows the results plotted together with the standard deviations.

We can observe that scale has a minimal influence with both stereo-based viewing conditions. Differently, the monoscopic display setup is strongly affected as the relative error undergoes an almost linear increase with the scale. Also changing the angle of the measured feature with respect to the user has negligible effect for stereo methods. With the Rubik scene the effect is also modest for the monoscopic view, while it is strong with the Saint Mark's scene. As already observed, this is probably due the lack of a reference edge to align with. Overall, the biases confirm the variations observed in the former experiments.

An assessment of the experimental evaluation

This study is a first attempt toward the evaluation of the contribution of tracking and stereo vision to the accuracy of interaction in mixed reality applications. We feel the overall methodology to be well suited for the task, however there are still a few issues to be considered to put the results into the correct perspective.

For starters, the rendering setup and the models adopted for building the virtual scene were rather basic, thus no advanced visual cues were available to help the user to correctly perceive depth with stereo vision disabled. Indeed, the role of visual cues is more apparent if we observe that monocular vision has a less negative impact with the more detailed Saint Mark's Square scene. We feel that artificial scenes with simple geometric shapes have a big role in mixed reality applications because they can be algorithmically manipulated, an important functionality when dealing with metaphorical objects to interact with the system and with data visualization scenarios. Still, in the future it would be important to study the actual impact of stereo and monocular view for different degrees of scene details and realism.

Some aspects of the experimental procedure could be enhanced: specifically, the unavoidable bias introduced by the variability in skills and attitude between users could be factored out by letting each user taking more measures and by evaluating their performance separately. As a good alternative or in addition to separate measurement batches, a baseline measuring experiment with real objects could be introduced to normalize each user.

Finally, it would also be interesting to repeat the test using different tracking and visualization systems. In fact, even if the bias introduced by the equipment cannot be avoided, increasing the number of mixed reality setups could yield more robust results.

9
Conclusions and Future Work

In this thesis we target the problems arising in low-cost Computer Vision based setup used to enhance the user's interaction experience by tracking its position and gestures. In the first part of the thesis we focused on the technical problems introduced by the adoption of consumer level optical devices. In section 3.1 we proposed a new Fiducial Marker formed of concentric rigs of circular features. We have seen that its flexibility in the number of dots and the robust cycle-coding based identification make its adoption ideal in different situation, from the camera Calibration, where a large number of features is desirable, to augmented reality applications where the occlusion robustness is a fundamental prerequisite. In section 3.2 we presented a method to accurately estimate the parameters of a 3D ellipse maximizing its reprojection consistency with respect to the gradient of the images acquired by a camera network. Although we formulated the optimization to optimize the ellipse parameter, we are working on a formulation to simultaneously optimize both the parameters of a set of ellipses and the extrinsic parameters of the cameras.

In sections 4.1, 4.2 and 4.3 we adopted the unconstrained camera model introduced by Bergamasco et al. [33] to handle the imperfections of low-cost image acquisition devices. In the first section we developed a outlier detection strategy that allows us to calibrate optical devices even when few observations of the calibration target are available. Together with the interpolation technique introduced in the next chapter it allows to calibrate even non central cameras (such the Lytro$^{\text{TM}}$ light field camera) with high accuracy and use them to triangulate the scene with only one shot.
Despite in section 4.1 we have proven that the unconstrained calibration technique allows to online calibrate with high accuracy the projector of a stereoscopic structured light scanner, it would be interesting to consider an even simpler setup composed by the projector and a single camera. This configuration is not easy to calibrate and we are working to provide a semi-automatic procedure procedure to calibrate it exploiting the unconstrained camera model for both the camera and the projector.

In the second part we presented two techniques to tackle the problem of finding correspondences between partial shapes under the presence of Non-Rigid quasi-isometric deformations. This problem can be found, for instance, when processing depth images obtained by the Kinect® sensor. In 6.1 we casted the matching problem as a L1 error minimization over the set of all the partial cycle-consistent multi-way matches over a collection of shapes. The limitation of this technique is mainly due to the inability to exactly control the density of the retrieved matches. Conversely, in section 6.2 we search for a dense matching between a full shape and a partial one using the Functional Map framework. Thanks to a perturbation analysis of the shape Laplacian under the presence of partiality, we have been able to add some soft constraints to the shape of the optimized functional map and the method has proven to outperform the state of the art algorithms of non-rigid dense matching. We are now looking to extend this approach to deal with partiality in both direction and apply it to deformable object in clutter scenarios.

In the last part we presented some Human Computer Interaction applications exploiting Computer Vision techniques to track user's pose and gestures. In section 8.1 we explained the setup of a big Interactive Table build for a museum exhibition and the interaction paradigm based on a active cursor thanks to the aggregation of image-based tracking and accelerometers data. Within sections 8.2 and 8.3 we introduce a robust tracking device that we used in two applications, as a pointing device in an interactive whiteboard and as a head tracking system in a viewer dependent display. The tracking system is composed by two IR pulsating LED tracked by a camera network. Thanks to the changing of the brightness intensity over the time we are able to retrieve the position and orientation of multiple devices exploiting only two LEDs. The last section is dedicated to the study of the human perception of the stereoscopy in viewer dependent displays with particular regard to the sizes ans spatial relation perception in augmented reality. This study highlighted very interesting aspects of the human perception which go beyond the simple projective geometry aspects. With the increasing diffusion of new portable displays it becomes more and more important to analyse the impact of a correct user interface design from the point of view of usability and comfort.

Bibliography

[17] The camera obscura: Aristotle to zahn. http://www.acmi.net.au/AIC/CAMERAOBSCURA.html, 2003.

[18] AIGER, D., MITRA, N. J., AND COHEN-OR, D. 4-points congruent sets for robust pairwise surface registration. *TOG 27*, 3 (2008), 85.

[19] ALBARELLI, A., RODOLÀ, E., AND TORSELLO, A. A game-theoretic approach to fine surface registration without initial motion estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), pp. 430–437.

[20] ALBARELLI, A., RODOLÀ, E., AND TORSELLO, A. Robust camera calibration using inaccurate targets. In *Proc. BMVC* (2010), pp. 16.1–10. doi:10.5244/C.24.16.

[21] ALBARELLI, A., RODOLÀ, E., AND TORSELLO, A. Imposing semi-local geometric constraints for accurate correspondences selection in structure from motion: A game-theoretic perspective. *Int. J. Comput. Vision 97*, 1 (2012), 36–53.

[22] ALBARELLI, A., RODOLÀ, E., AND TORSELLO, A. Fast and accurate surface alignment through an isometry-enforcing game. *Pattern Recognition 48* (2015), 2209–2226.

[23] ALBARELLI, A., ROTA BULÒ, S., TORSELLO, A., AND PELILLO, M. Matching as a non-cooperative game. In *Proc. ICCV* (2009), pp. 1319–1326.

[24] ANGUELOV, D., SRINIVASAN, P., KOLLER, D., THRUN, S., RODGERS, J., AND DAVIS, J. Scape: Shape completion and animation of people. *ACM Trans. Graph. 24*, 3 (2005), 408–416.

[25] ARDITO, C., COSTABILE, M. F., AND LANZILOTTI, R. Gameplay on a multi-touch screen to foster learning about historical sites. In *AVI '10, Proc. of the Int. Conf. on Advanced Visual Interfaces* (2010), ACM, pp. 75–78.

[26] AUBRY, M., SCHLICKEWEI, U., AND CREMERS, D. The wave kernel signature: A quantum mechanical approach to shape analysis. In *Proc. ICCV Workshops* (2011), pp. 1626–1633.

[27] AUDET, S., AND OKUTOMI, M. A user-friendly method to geometrically calibrate projector-camera systems. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on* (2009), pp. 47–54.

[28] Bartoli, A., and Lapresté, J.-T. Triangulation for points on lines. *Image Vision Comput. 26*, 2 (Feb. 2008), 315–324.

[29] Bartu, P., Neulinger, A., Jakoby, B., Bauer, S., and Koeppe, R. Light curtain for 2d large-area object detection. *Opt. Express 21*, 10 (May 2013), 12757–12766.

[30] Belkin, M., Sun, J., and Wang, Y. Constructing laplace operator from point clouds in rd. In *Proc. SODA* (2009), Society for Industrial and Applied Mathematics, pp. 1031–1040.

[31] Ben-David, A., and Mandel, J. Classification accuracy: Machine learning vs. explicit knowledge acquisition. *Machine Learning 18* (1995), 109–114.

[32] Bergamasco, F., Albarelli, A., Rodola, E., and Torsello, A. Runetag: A high accuracy fiducial marker with strong occlusion resilience. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on* (June 2011), pp. 113–120.

[33] Bergamasco, F., Albarelli, A., Rodola, E., and Torsello, A. Can a fully unconstrained imaging model be applied effectively to central cameras? In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (June 2013), pp. 1391–1398.

[34] Bergamasco, F., Albarelli, A., and Torsello, A. Pi-tag: A fast image-space marker design based on projective invariants. *Machine Vision and Applications* (2012).

[35] Bergamasco, F., Cosmo, L., Albarelli, A., and Torsello, A. Camera calibration from coplanar circles. In *22nd International Conference on Pattern Recognition (ICPR 2014)* (Aug 2014).

[36] Bergweiler, S., Deru, M., and Porta, D. Integrating a multitouch kiosk system with mobile devices and multimodal interaction. In *ACM International Conference on Interactive Tabletops and Surfaces* (New York, NY, USA, 2010), ITS '10, ACM, pp. 245–246.

[37] Besl, P. J., and McKay, N. D. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2 (1992), 239–256.

[38] Bishop, T., and Favaro, P. Plenoptic depth estimation from multiple aliased views. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on* (Sept 2009), pp. 1622–1629.

[39] Bishop, T., and Favaro, P. The light field camera: Extended depth of field, aliasing, and superresolution. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 34*, 5 (May 2012), 972–986.

[40] Bok, Y., Jeon, H.-G., and Kweon, I. Geometric calibration of micro-lens-based light-field cameras using line features. In *Computer Vision ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8694 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 47–61.

[41] Bosetti, M., Pilolli, P., Ruffoni, M., and Ronchetti, M. Interactive whiteboards based on the wiimote: Validation on the field. In *Interactive Collaborative Learning (ICL), 2011 14th International Conference on* (2011), pp. 269–273.

[42] Boumal, N., Mishra, B., Absil, P.-A., and Sepulchre, R. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research 15* (2014), 1455–1459.

[43] Bradski, G., and Kaehler, A. *Learning OpenCV: Computer Vision with the OpenCV Library*, 1st ed. O'Reilly Media, Inc., 2008.

[44] Brandl, P., Haller, M., Hurnaus, M., Lugmayr, V., Oberngruber, J., Oster, C., Schafleitner, C., and Billinghurst, M. An adaptable rear-projection screen using digital pens and hand gestures. In *Artificial Reality and Telexistence, 17th International Conference on* (2007), pp. 49–54.

[45] Bronstein, A., Bronstein, M., Bruckstein, A., and Kimmel, R. Partial similarity of objects, or how to compare a centaur to a horse. *IJCV 84*, 2 (2009), 163–183.

[46] Bronstein, A., Bronstein, M., and Kimmel, R. *Numerical Geometry of Non-Rigid Shapes*, 1 ed. Springer Publishing Company, Incorporated, 2008.

[47] Bronstein, A. M., and Bronstein, M. M. Not only size matters: regularized partial matching of nonrigid shapes. In *Proc. NORDIA* (2008).

[48] Bronstein, A. M., Bronstein, M. M., and Kimmel, R. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *PNAS 103*, 5 (2006), 1168–1172.

[49] Bronstein, M. M., and Kokkinos, I. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *In Proc. CVPR* (2010).

[50] Brown, D. C. Close-range camera calibration. *PHOTOGRAMMETRIC ENGINEERING 37*, 8 (1971), 855–866.

[51] Brunton, A., Wand, M., Wuhrer, S., Seidel, H.-P., and Weinkauf, T. A low-dimensional representation for robust partial isometric correspondences computation. *Graphical Models 76*, 2 (2014), 70–85.

[52] BUCHANAN, P., AND GREEN, R. Creating a view dependent rendering system for mainstream use. In *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference* (2008), pp. 1–6.

[53] BURGES, C. J. C. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov. 2*, 2 (June 1998), 121–167.

[54] CAMPLANI, M., SALGADO, L., AND CAMPLANI, R. Low-cost efficient interactive whiteboard. In *Consumer Electronics (ICCE), 2012 IEEE International Conference on* (2012), pp. 686–687.

[55] CHASLES, M. Note sur les propriétés générales du systme de deux corps semblables entr'eux et placés d'une manière quelconque dans l'espace; et sur le déplacement fini ou infiniment petit d'un corps solide libre. *Bulletin des Sciences Mathematiques, Astronomiques, Physiques et Chimiques 14* (1830), 321–326.

[56] CHEN, C., LIN, H., YU, Z., KANG, S. B., AND YU, J. Light field stereo matching using bilateral statistics of surface cameras. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on* (June 2014), pp. 1518–1525.

[57] CHEN, C.-Y., AND CHIEN, H.-J. An incremental target-adapted strategy for active geometric calibration of projector-camera systems. *Sensors 13*, 2 (2013), 2664–2681.

[58] CHEN, Q., WU, H., AND WADA, T. Camera calibration with two arbitrary coplanar circles. In *Proc. European Conf. Computer Vision* (2004), Inc, pp. 521–532.

[59] CHEN, Y., AND GOLDSMITH, A. Information recovery from pairwise measurements. In *Proc. ISIT* (2014), pp. 2012–2016.

[60] CHEN, Y., GUIBAS, L., AND HUANG, Q.-X. Near-optimal joint object matching via convex relaxation. In *Proc. ICML* (2014), pp. 100–108.

[61] CHEN, Y., AND IP, H. H. S. Planar metric rectification by algebraically estimating the image of the absolute conic. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 4 - Volume 04* (Washington, DC, USA, 2004), ICPR '04, IEEE Computer Society, pp. 88–91.

[62] CHESHIRE, S., AND BAKER, M. Consistent overhead byte stuffing. *IEEE/ACM Trans. Netw. 7*, 2 (Apr. 1999), 159–172.

[63] CHESI, G., AND HUNG, Y. S. Fast multiple-view l2 triangulation with occlusion handling. *Comput. Vis. Image Underst. 115*, 2 (Feb. 2011), 211–223.

[64] CHIA, A., LEUNG, M., ENG, H.-L., AND RAHARDJA, S. Ellipse detection with hough transform in one dimensional parametric space. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on* (16 2007-oct. 19 2007), vol. 5, pp. V –333 –V –336.

[65] Cho, D., Lee, M., Kim, S., and Tai, Y.-W. Modeling the calibration pipeline of the lytro camera for high quality light-field image reconstruction. In *Computer Vision (ICCV), 2013 IEEE International Conference on* (Dec 2013), pp. 3280–3287.

[66] Cho, Y., Lee, J., and Neumann, U. A multi-ring color fiducial system and a rule-based detection method for scalable fiducial-tracking augmented reality. In *Proceedings of International Workshop on Augmented Reality* (1998).

[67] Ciocca, G., Olivo, P., and Schettini, R. Browsing museum image collections on a multi-touch table. *Inf. Syst. 37*, 2 (Apr. 2012), 169–182.

[68] Claus, D., and Fitzgibbon, A. W. A rational function lens distortion model for general cameras. In *Proc. IEEE Computer Vision and Pattern Recognition* (2005), pp. 213–219.

[69] Claus, D., and Fitzgibbon, A. W. Reliable automatic calibration of a marker-based position tracking system. In *IEEE Workshop on Applications of Computer Vision* (2005).

[70] Coifman, R. R., and Lafon, S. Diffusion maps. *Applied and Computational Harmonic Analysis 21*, 1 (2006), 5 – 30. Special Issue: Diffusion Maps and Wavelets.

[71] Cortes, C., and Vapnik, V. Support-vector networks. In *Machine Learning* (1995), pp. 273–297.

[72] Cosmo, L., Albarelli, A., Bergamasco, F., and Torsello, A. Design and evaluation of a viewer-dependent stereoscopic display. In *22nd Int. Conf. on Pattern Recognition (ICPR)* (2014), pp. 2861–2866.

[73] Cosmo, L., Rodolà, E., Albarelli, A., Mémoli, F., and Cremers, D. Consistent partial matching of shape collections via sparse modeling. *Computer Graphics Forum* (2015). to appear.

[74] Coxeter, H. S. M. *Projective Geometry, 2nd ed.* Springer Verlag, 2003.

[75] Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. Surround-screen projection-based virtual reality: The design and implementation of the cave. In *Proc. of the 20th Annual Conf. on Computer Graphics and Interactive Techniques* (1993), SIGGRAPH '93, pp. 135–142.

[76] Dachselt, R., and Buchholz, R. Natural throw and tilt interaction between mobile phones and distant displays. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems* (New York, NY, USA, 2009), CHI EA '09, ACM, pp. 3253–3258.

[77] DAFTRY, S., MAURER, M., WENDEL, A., AND BISCHOF, H. Flexible and user-centric camera calibration using planar fiducial markers. In *Proceedings of the British Machine Vision Conference* (2013), BMVA Press.

[78] DANSEREAU, D., PIZARRO, O., AND WILLIAMS, S. Decoding, calibration and rectification for lenselet-based plenoptic cameras. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (June 2013), pp. 1027–1034.

[79] DEERING, M. High resolution virtual reality. In *19th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1992), SIGGRAPH '92, ACM, pp. 195–202.

[80] DEVERNAY, F., AND FAUGERAS, O. Straight lines have to be straight: automatic calibration and removal of distortion from scenes of structured environments. *Mach. Vision Appl. 13*, 1 (Aug. 2001), 14–24.

[81] DIETZ, P., AND LEIGH, D. Diamondtouch: A multi-user touch technology. In *UIST '01, Proc. of the 14th Annual ACM Symposium on User Interface Software and Technology* (2001), ACM, pp. 219–226.

[82] DOHSE, T., STILL, J., AND PARKHURST, D. Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. In *First Int. Conf. on Advances in Computer-Human Interaction* (2008), pp. 297–302.

[83] DÖRING, T., SHIRAZI, A., AND SCHMIDT, A. Exploring gesture-based interaction techniques in multi-display environments with mobile phones and a multi-touch table. In *Proc. of PPD '10, Workshop on Coupled Display Visual Interfaces, in conjunction with AVI 2010* (2010), pp. 47–54.

[84] DUFFIN, R. J. Distributed and lumped networks. *Journal of Mathematics and Mechanics 8*, 5 (1959), 793–826.

[85] DUFOURNAUD, Y., HORAUD, R., AND QUAN, L. Robot Stereo-hand Coordination for Grasping Curved Parts. In *9th British Machine Vision Conference (BMVC '98)* (Southampton, Royaume-Uni, 1998), J. N. Carter and M. S. Nixon, Eds., vol. 2, British Machine Vision Association, pp. 760–769.

[86] ECHTLER, F., NESTLER, S., DIPPON, A., AND KLINKER, G. Supporting casual interactions between board games on public tabletop displays and mobile devices. *Personal and Ubiquitous Computing 13* (2009), 609–617.

[87] ELROD, S., BRUCE, R., GOLD, R., GOLDBERG, D., HALASZ, F., JANSSEN, W., LEE, D., MCCALL, K., PEDERSEN, E., PIER, K., TANG, J. C., AND WELCH, B. B. Liveboard: A large interactive display supporting group meetings, presentations and remote collaboration. ACM Press, New York, pp. 599–607. Proc. Conf. on Human Factors in Computing Systems (CHI).

[88] ERIKSSON, E. S. Movement parallax during locomotion. *Perception & Psychophysics 16*, 1 (1974), 197–200.

[89] FELZENSZWALB, P., AND SCHWARTZ, J. Hierarchical matching of deformable shapes. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on* (June 2007), pp. 1–8.

[90] FIALA, M. Artag, a fiducial marker system using digital techniques. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Washington, DC, USA, 2005), CVPR '05, IEEE Computer Society.

[91] FIALA, M. Designing highly reliable fiducial markers. *IEEE Trans. Pattern Anal. Mach. Intel. 32*, 7 (2010).

[92] FIALA, M. Designing highly reliable fiducial markers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 32*, 7 (July 2010), 1317–1324.

[93] FITZGIBBON, A., PILU, M., AND FISHER, R. Direct least square fitting of ellipses. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 21*, 5 (may 1999), 476 –480.

[94] FORNEY, G. On decoding bch codes. *Information Theory, IEEE Transactions on 11*, 4 (Oct 1965), 549–557.

[95] GARLAND, M., AND HECKBERT, P. S. Surface simplification using quadric error metrics. In *Proc. SIGGRAPH* (1997), pp. 209–216.

[96] GARRIDO-JURADO, S., MUOZ-SALINAS, R., MADRID-CUEVAS, F., AND MARN-JIMNEZ, M. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition 47*, 6 (2014), 2280 – 2292.

[97] GARSTKA, J., AND PETERS, G. View-dependent 3d projection using depth-image-based head tracking. In *Proceedings of the 8th IEEE International Workshop on ProjectorCamera Systems* (2011), PROCAM '11, IEEE, pp. 41–47.

[98] GATRELL, L., HOFF, W., AND SKLAIR, C. Robust image features: Concentric contrasting circles and their image extraction. In *Proc. of Cooperative Intelligent Robotics in Space* (Washington, USA, 1991), SPIE.

[99] GOLDLUECKE, B. Globally consistent depth labeling of 4d light fields. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Washington, DC, USA, 2012), CVPR '12, IEEE Computer Society, pp. 41–48.

[100] GROSSBERG, M. D., AND NAYAR, S. K. A general imaging model and a method for finding its parameters. In *International Conference of Computer Vision* (2001), pp. 108–115.

[101] GUPTA, A., LITTLE, J., AND WOODHAM, R. Using line and ellipse features for rectification of broadcast hockey video. In *Computer and Robot Vision (CRV), 2011 Canadian Conference on* (may 2011), pp. 32 –39.

[102] HAGBI, N., BERGIG, O., EL-SANA, J., AND BILLINGHURST, M. Shape Recognition and Pose Estimation for Mobile Augmented Reality. In *8th IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2009)* (2009), IEEE Computer Press, pp. 65–71.

[103] HAN, J. Y. Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05, Proc. of the 18th Annual ACM Symposium on User Interface Software and Technology* (2005), ACM, pp. 115–118.

[104] HANSEN, D. W., AND JI, Q. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 3 (Mar. 2010), 478–500.

[105] HARISH, P., AND NARAYANAN, P. J. A view-dependent, polyhedral 3d display. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and Its Applications in Industry* (New York, NY, USA, 2009), VRCAI '09, ACM, pp. 71–75.

[106] HARTLEY, R., AND ZISSERMAN, A. *Multiple View Geometry in Computer Vision*, 2 ed. Cambridge University Press, New York, NY, USA, 2003.

[107] HARTLEY, R. I., AND STURM, P. Triangulation. *Comput. Vis. Image Underst. 68*, 2 (Nov. 1997), 146–157.

[108] HEIKKILA, J. Moment and curvature preserving technique for accurate ellipse boundary detection. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on* (aug 1998), vol. 1, pp. 734 –737 vol.1.

[109] HEIKKILÄ, J. Geometric camera calibration using circular control points. *IEEE Trans. Pattern Anal. Mach. Intell. 22*, 10 (Oct. 2000), 1066–1077.

[110] HEROUT, A., SZENTANDRASI, I., ZACHARIA, M., DUBSKA, M., AND KAJAN, R. Five shades of grey for fast and reliable camera pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on* (June 2013), pp. 1384–1390.

[111] HOANG, A. N., TRAN HOANG, V., AND KIM, D. A real-time rendering technique for view-dependent stereoscopy based on face tracking. In *Proceedings of the 13th International Conference on Computational Science and Its Applications - Volume 1* (Berlin, Heidelberg, 2013), ICCSA'13, Springer-Verlag, pp. 697–707.

[112] HORN, B. K. P. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A 4*, 4 (1987), 629–642.

[113] HORNECKER, E. “i don’t understand it either, but it is cool” — visitor interactions with a multi-touch table in a museum. In *TABLETOP 2008. 3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems* (2008), pp. 113–120.

[114] HUANG, Q., WANG, F., AND GUIBAS, L. J. Functional map networks for analyzing and exploring large shape collections. *TOG 33*, 4 (2014), 36.

[115] HUANG, Q.-X., AND GUIBAS, L. Consistent shape maps via semidefinite programming. *Computer Graphics Forum 32*, 5 (2013), 177–186.

[116] HUANG, Q.-X., ZHANG, G.-X., GAO, L., HU, S.-M., BUTSCHER, A., AND GUIBAS, L. An optimization approach for extracting and encoding consistent maps in a shape collection. *ACM Trans. Graph. 31*, 6 (2012), 167:1–167:11.

[117] HUGHES, C., DENNY, P., JONES, E., AND GLAVIN, M. Accuracy of fish-eye lens models. *Appl. Opt. 49*, 17 (Jun 2010), 3338–3347.

[118] HUYNH, D. Q. The cross ratio: A revisit to its probability density function. In *Proceedings of the British Machine Vision Conference BMVC 2000* (2000).

[119] JACUCCI, G., MORRISON, A., RICHARD, G., KLEIMOLA, J., PELTONEN, P., PARISI, L., AND LAITINEN, T. Worlds of information: Designing for engagement at a public multi-touch display. In *CHI ’10, Proc. of the 28th Int. Conf. on Human Factors in Computing Systems* (2010), ACM, pp. 2267–2276.

[120] JOHANNSEN, O., HEINZE, C., GOLDLUECKE, B., AND PERWA, C. On the calibration of focused plenoptic cameras. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, M. Grzegorzek, C. Theobalt, R. Koch, and A. Kolb, Eds., vol. 8200 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 302–317.

[121] KALMAN, R. E. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME—Journal of Basic Engineering 82 (Series D)* (1960), 35–45.

[122] KANG, S. B. Catadioptric self-calibration. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on* (2000), vol. 1, pp. 201–207 vol.1.

[123] KATO, H., AND BILLINGHURST, M. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality* (Washington, DC, USA, 1999), IEEE Computer Society.

[124] KAVAN, L., COLLINS, S., O'SULLIVAN, C., AND ŽÁRA, J. Dual quaternions for rigid transformation blending. Tech. Rep. TCD-CS-2006-46, Trinity College Dublin, 2006.

[125] KIM, S., CHOI, W., RIM, W., CHUN, Y., SHIM, H., KWON, H., KIM, J., KEE, I., KIM, S., LEE, S., AND PARK, J. A highly sensitive capacitive touch sensor integrated on a thin-film-encapsulated active-matrix oled for ultrathin displays. *Electron Devices, IEEE Transactions on 58*, 10 (2011), 3609–3615.

[126] KIM, V. G., LIPMAN, Y., CHEN, X., AND FUNKHOUSER, T. A. Möbius transformations for global intrinsic symmetry analysis. *Comput. Graph. Forum 29*, 5 (2010), 1689–1700.

[127] KIM, V. G., LIPMAN, Y., AND FUNKHOUSER, T. Blended intrinsic maps. *ACM Trans. Graph. 30*, 4 (2011), 79:1–79:12.

[128] KIMURA, M., MOCHIMARU, M., AND KANADE, T. Projector calibration using arbitrary planes and calibrated camera. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on* (2007), pp. 1–8.

[129] KNYAZ, V. A., GROUP, H. O., AND SIBIRYAKOV, R. V. The development of new coded targets for automated point identification and non-contact surface measurements. In *3D Surface Measurements, International Archives of Photogrammetry and Remote Sensing* (1998).

[130] KOKKINOS, I., BRONSTEIN, M. M., LITMAN, R., AND BRONSTEIN, A. M. Intrinsic shape context descriptors for deformable shapes. In *CVPR* (2012), IEEE Computer Society, pp. 159–166.

[131] KOVNATSKY, A., BRONSTEIN, M. M., BRESSON, X., AND VANDERGHEYNST, P. Functional correspondence by matrix completion. In *Proc. CVPR* (2015).

[132] KRUEGER, M. W. *Artificial Reality*. 1983.

[133] KRUEGER, W., AND FROEHLICH, B. The responsive workbench. *IEEE Comput. Graph. Appl. 14*, 3 (1994), 12–15.

[134] KRÜGER, W., BOHN, C.-A., FRÖHLICH, B., SCHÜTH, H., STRAUSS, W., AND WESCHE, G. The responsive workbench: A virtual work environment. *Computer 28*, 7 (1995), 42–48.

[135] LECH, M., AND KOSTEK, B. Gesture-based computer control system applied to the interactive whiteboard. In *Information Technology (ICIT), 2010 2nd International Conference on* (2010), pp. 75–78.

[136] LEE, J. C., DIETZ, P. H., MAYNES-AMINZADE, D., RASKAR, R., AND HUDSON, S. E. Automatic projector calibration with embedded light sensors. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2004), UIST '04, ACM, pp. 123–126.

[137] LEORDEANU, M., AND HEBERT, M. A spectral technique for correspondence problems using pairwise constraints. In *Proc. ICCV* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 1482–1489.

[138] LEVIN, A., AND DURAND, F. Linear view synthesis using a dimensionality gap light field prior. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (June 2010), pp. 1831–1838.

[139] LEVOY, M., AND HANRAHAN, P. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, ACM, pp. 31–42.

[140] LI, H., SUMNER, R. W., AND PAULY, M. Global correspondence optimization for non-rigid registration of depth scans. In *Proc. SGP* (2008), pp. 1421–1430.

[141] LIANG, T.-H., HUANG, Y.-M., AND TSAI, C.-C. An investigation of teaching and learning interaction factors for the use of the interactive whiteboard technology. *Educational Technology and Society 15*, 4 (2012), 356–367.

[142] LIERE, R. V., AND MULDER, J. D. Optical tracking using projective invariant marker pattern properties. In *Proceedings of the IEEE Virtual Reality Conference* (2003), IEEE Press.

[143] LILIENBLUM, E., AND MICHAELIS, B. Optical 3d surface reconstruction by a multi-period phase shift method. *JCP 2*, 2 (2007), 73–83.

[144] LIU, T., KIM, V. G., AND FUNKHOUSER, T. Finding surface correspondences using symmetry axis curves. *Computer Graphics Forum 31*, 5 (2012).

[145] MA, L., HEN, Y., AND MOORE, K. L. Flexible camera calibration using a new analytical radial undistortion formula with application to mobile robot localization. In *IEEE International Symposium on Intelligent Control* (2003), pp. 799–804.

[146] MACNEAL, R. H. *The solution of partial differential equations by means of electrical networks*. PhD thesis, California Institute of Technology, 1949.

[147] MACWILLIAMS, F., AND SLOANE, N. *The Theory of Error-Correcting Codes*, 2nd ed. North-holland Publishing Company, 1978.

[148] MAIDI, M., DIDIER, J.-Y., ABABSA, F., AND MALLEM, M. A performance study for camera pose estimation using visual marker based tracking. *Mach. Vision Appl. 21* (2010).

[149] MALASSIOTIS, S., AND STRINTZIS, M. Stereo vision system for precision dimensional inspection of 3d holes. *Machine Vision and Applications 15* (2003), 101–113.

[150] MALLON, J., AND WHELAN, P. F. Which pattern? biasing aspects of planar calibration patterns and detection methods. *Pattern Recogn. Lett. 28*, 8 (June 2007), 921–930.

[151] MARQUARDT, N., KIEMER, J., AND GREENBERG, S. What caused that touch?: expressive interaction with a surface through fiduciary-tagged gloves. In *ACM International Conference on Interactive Tabletops and Surfaces* (New York, NY, USA, 2010), ITS '10, ACM, pp. 139–142.

[152] MARTELLI, S., MARZOTTO, R., COLOMBARI, A., AND MURINO, V. Fpga-based robust ellipse estimation for circular road sign detection. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on* (june 2010), pp. 53 –60.

[153] MARZULLO, K. A. *Maintaining the time in a distributed system: an example of a loosely-coupled distributed service*. PhD thesis, Stanford University, Stanford, CA, USA, 1984.

[154] MCLAUGHLIN, R. A. Randomized hough transform: Improved ellipse detection with comparison. *Pattern Recognition Letters 19*, 34 (1998), 299 – 305.

[155] MÉMOLI, F. Gromov-Wasserstein distances and the metric approach to object matching. *Foundations of Computational Mathematics* (2011), 1–71.

[156] MONTGOMERY, D. C. *Design and Analysis of Experiments*. John Wiley & Sons, 2006.

[157] MOOSER, J., YOU, S., AND NEUMANN, U. Tricodes: A barcode-like fiducial design for augmented reality media. *Multimedia and Expo, IEEE International Conference on* (2006).

[158] MUMFORD, D., AND SHAH, J. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure and Applied Math. 42*, 5 (1989), 577–685.

[159] MURTHY, D. V., AND HAFTKA, R. T. Derivatives of eigenvalues and eigenvectors of a general complex matrix. *Intl. J. Numer. Met. Eng. 26*, 2 (1988), 293–311.

[160] MUSA, E. Laser-based light barrier. *Appl. Opt. 47*, 19 (Jul 2008), 3415–3422.

[161] NACENTA, M. A., SAKURAI, S., YAMAGUCHI, T., MIKI, Y., ITOH, Y., KITAMURA, Y., SUBRAMANIAN, S., AND GUTWIN, C. E-conic: A perspective-aware interface for multi-display environments. In *Proceedings of the 20th Annual ACM*

Symposium on User Interface Software and Technology (New York, NY, USA, 2007), UIST '07, ACM, pp. 279–288.

[162] NAIMARK, L., AND FOXLIN, E. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality* (Washington, DC, USA, 2002), ISMAR '02, IEEE Computer Society.

[163] NAWROT, M., AND JOYCE, L. The pursuit theory of motion parallax. *Vision Research 46* (2006), 4709–4725.

[164] NAWROT, M., RATZLAFF, M., LEONARD, Z., AND STROYAN, K. Modeling depth from motion parallax with the motion/pursuit ratio. *Frontiers in Psychology 5*, 1103 (2014).

[165] NG, R., LEVOY, M., BRÉDIF, M., DUVAL, G., HOROWITZ, M., AND HANRAHAN, P. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR 2*, 11 (2005).

[166] NGUYEN, A., BEN-CHEN, M., WELNICKA, K., YE, Y., AND GUIBAS, L. J. An optimization approach to improving collections of shape maps. *Computer Graphics Forum* (2011), 1481–1491.

[167] OIKONOMIDIS, I., AND ARGYROS, A. Deformable 2d shape matching based on shape contexts and dynamic programming. In *Advances in Visual Computing*, G. Bebis, R. Boyle, B. Parvin, D. Koracin, Y. Kuno, J. Wang, R. Pajarola, P. Lindstrom, A. Hinkenjann, M. Encarnao, C. Silva, and D. Coming, Eds., vol. 5876 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, pp. 460–469.

[168] ORSINI, E., AND SALA, M. Correcting errors and erasures via the syndrome variety . *Journal of Pure and Applied Algebra 200*, 1–2 (2005), 191–226.

[169] OTSU, N. A threshold selection method from gray-level histograms. *Systems, Man and Cybernetics, IEEE Transactions on 9*, 1 (1979), 62–66.

[170] OUELLET, J., AND HEBERT, P. Precise ellipse estimation without contour point extraction. *Mach. Vision Appl. 21* (2009).

[171] OVSJANIKOV, M., BEN-CHEN, M., SOLOMON, J., BUTSCHER, A., AND GUIBAS, L. Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph. 31*, 4 (2012), 30:1–30:11.

[172] PACHAURI, D., KONDOR, R., AND SINGH, V. Solving the multi-way matching problem by permutation synchronization. In *Proc. NIPS* (2013), pp. 1860–1868.

[173] PICKUP, D., SUN, X., ROSIN, P. L., ET AL. SHREC'14 track: Shape retrieval of non-rigid 3d human models. In *Proceedings of the 7th Eurographics workshop on 3D Object Retrieval* (2014), EG 3DOR'14, Eurographics Association.

[174] PIERARD, S., PIERLOT, V., LEJEUNE, A., AND VAN DROOGENBROECK, M. I-see-3d ! an interactive and immersive system that dynamically adapts 2d projections to the location of a user's eyes. In *3D Imaging (IC3D), 2012 International Conference on* (2012), pp. 1–8.

[175] PINHANEZ, C. Using a steerable projector and a camera to transform surfaces into interactive displays. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2001), CHI EA '01, ACM, pp. 369–370.

[176] PINKALL, U., AND POLTHIER, K. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics 2*, 1 (1993), 15–36.

[177] POKRASS, J., BRONSTEIN, A. M., AND BRONSTEIN, M. M. Partial shape matching without point-wise correspondence. *Numer. Math. Theor. Meth. Appl. 6* (2013), 223–244.

[178] QUAN, L. Conic reconstruction and correspondence from two views. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 18*, 2 (feb 1996), 151 –160.

[179] RAGAN, E. D., KOPPER, R., SCHUCHARDT, P., AND BOWMAN, D. A. Studying the effects of stereo, head tracking, and field of regard on a small-scale spatial judgment task. *IEEE Transactions on Visualization and Computer Graphics 19*, 5 (May 2013), 886–896.

[180] REKIMOTO, J., AND AYATSUKA, Y. CyberCode: designing augmented reality environments with visual tags. In *DARE '00: Proceedings of DARE 2000 on Designing augmented reality environments* (New York, NY, USA, 2000), ACM.

[181] RODOLÀ, E., ALBARELLI, A., BERGAMASCO, F., AND TORSELLO, A. A scale independent selection process for 3d object recognition in cluttered scenes. *Int. J. Comput. Vision 102*, 1-3 (2013), 129–145.

[182] RODOLÀ, E., BRONSTEIN, A., ALBARELLI, A., BERGAMASCO, F., AND TORSELLO, A. A game-theoretic approach to deformable shape matching. In *Proc. CVPR* (June 2012), pp. 182–189.

[183] RODOLÀ, E., ROTA BULÒ, S., AND CREMERS, D. Robust region detection via consensus segmentation of deformable shapes. *Computer Graphics Forum 33*, 5 (2014), 97–106.

[184] RODOLÀ, E., ROTA BULÒ, S., WINDHEUSER, T., VESTNER, M., AND CREMERS, D. Dense non-rigid shape correspondence using random forests. In *Proc. CVPR* (2014), pp. 4177–4184.

[185] RODOLÀ, E., TORSELLO, A., HARADA, T., KUNIYOSHI, Y., AND CREMERS, D. Elastic net constraints for shape matching. In *Proc. ICCV* (2013), pp. 1169–1176.

[186] ROTA BULÒ, S., AND BOMZE, I. M. Infection and immunization: A new class of evolutionary game dynamics. *Games and Economic Behavior 71*, 1 (2011), 193–211.

[187] ROTA BULÒ, S., TORSELLO, A., AND PELILLO, M. A game-theoretic approach to partial clique enumeration. *Image Vision Comput. 27*, 7 (2009), 911–922.

[188] S., B. *Six Centuries of Master Prints*. Cincinnati Art Museum, Cincinnati, Ohio, 1993.

[189] SAHILLIOĞLU, Y., AND YEMEZ, Y. Partial 3-d correspondence from shape extremities. *Computer Graphics Forum 33*, 6 (2014), 63–76.

[190] SAHILLIOĞLU, Y., AND YEMEZ, Y. Multiple shape correspondence by dynamic programming. *Computer Graphics Forum 33*, 7 (2014), 121–130.

[191] SCARAMUZZA, D., MARTINELLI, A., AND SIEGWART, R. A toolbox for easily calibrating omnidirectional cameras. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on* (Oct 2006), pp. 5695–5701.

[192] SCHLEICHER, D., AND ZAGAR, B. Image processing to estimate the ellipticity of steel coils using a concentric ellipse fitting algorithm. In *Signal Processing, 2008. ICSP 2008. 9th International Conference on* (oct. 2008), pp. 884 –890.

[193] SCHMIDT, F. R., TÖPPE, E., CREMERS, D., AND BOYKOV, Y. Intrinsic mean for semi-metrical shape retrieval via graph cuts. In *Proc. DAGM* (Berlin, Heidelberg, 2007), Springer-Verlag, pp. 446–455.

[194] SCHÖNING, J., ROHS, M., AND KRÜGER, A. Using mobile phones to spontaneously authenticate and interact with multi-?touch surfaces. In *Proc. of PPD 2008, Workshop on designing multi-??touch interaction techniques for coupled public and private displays* (2008).

[195] SEHGAL, A. K., DAS, S., NOTO, K., SAIER, M., AND ELKAN, C. Identifying relevant data for a biological database: Handcrafted rules versus machine learning. *IEEE/ACM Trans. Comput. Biol. Bioinformatics 8*, 3 (May 2011), 851–857.

[196] SHAH, S., AND AGGARWAL, J. Intrinsic parameter calibration procedure for a (high-distortion) fish-eye lens camera with distortion model and accuracy estimation*. *Pattern Recognition 29*, 11 (1996), 1775 – 1788.

[197] SHEPARD, D. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM National Conference* (New York, NY, USA, 1968), ACM '68, ACM, pp. 517–524.

[198] SHI, Y., YANG, Z., YANG, H. H., AND LIU, S. Study on the research hotspots of interactive whiteboards in education. In *ICIMCS* (2012), ACM, pp. 209–212.

[199] SHOEMAKE, K. Animating rotation with quaternion curves. In *Proc. of the 12th annual conference on Computer graphics and interactive techniques* (1985), pp. 245–254.

[200] SLAY, H., SIEBRGER, I., AND HODGKINSON-WILLIAMS, C. Interactive whiteboards: Real beauty or just lipstick? *Computers and Education 51*, 3 (2008), 1321 – 1341.

[201] SMITH, H. J., HIGGINS, S., WALL, K., AND MILLER, J. Interactive whiteboards: boon or bandwagon? a critical review of the literature. *Journal of Computer Assisted Learning 21*, 2 (2005), 91–101.

[202] SO, E., ZHANG, H., AND SHENG GUAN, Y. Sensing contact with analog resistive technology. In *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on* (1999), vol. 2, pp. 806–811 vol.2.

[203] SOLÉ-RIBALTA, A., AND SERRATOSA, F. Graduated assignment algorithm for multiple graph matching based on a common labeling. *Int. J. Patt. Recog. Art. Intel. 27*, 01 (2013).

[204] SONG, G., AND WANG, H. A fast and robust ellipse detection algorithm based on pseudo-random sample consensus. In *Computer Analysis of Images and Patterns*, vol. 4673 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2007, pp. 669–676.

[205] SRESTASATHIERN, P., AND YILMAZ, A. Planar shape representation and matching under projective transformation. *Computer Vision and Image Understanding 115*, 11 (2011), 1525 – 1535.

[206] STEWENIUS, H., SCHAFFALITZKY, F., AND NISTER, D. How hard is 3-view triangulation really? In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01* (Washington, DC, USA, 2005), ICCV '05, IEEE Computer Society, pp. 686–693.

[207] STURM, P., AND RAMALINGAM, S. A generic concept for camera calibration. In *Proc. European Conference on Computer Vision* (May 2004), vol. 2, Springer, pp. 1–13.

[208] SUN, L., ZHANG, D., LI, B., GUO, B., AND LI, S. Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations. In *Ubiquitous Intelligence and Computing*, Z. Yu, R. Liscano, G. Chen, D. Zhang, and X. Zhou, Eds., vol. 6406 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, pp. 548–562.

[209] SUNG, K. Recent videogame console technologies. *Computer 44*, 2 (2011), 91–93.

[210] SWAMINATHAN, R., AND NAYAR, S. K. Nonmetric calibration of wide-angle lenses and polycameras. *IEEE Trans. Pattern Anal. Mach. Intell. 22*, 10 (Oct. 2000), 1172–1178.

[211] SZENTANDRASI, I., ZACHARIAS, M., HAVEL, J., HEROUT, A., DUBSKA, M., AND KAJAN, R. Uniform marker fields: Camera localization by orientable de bruijn tori. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on* (Nov 2012), pp. 319–320.

[212] TAKEOKA, Y., MIYAKI, T., AND REKIMOTO, J. Z-touch: an infrastructure for 3d gesture interaction in the proximity of tabletop surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces* (New York, NY, USA, 2010), ITS '10, ACM, pp. 91–94.

[213] TANG, J. C., AND MINNEMAN, S. Videowhiteboard: video shadows to support remote collaboration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 1991), CHI '91, ACM, pp. 315–322.

[214] TANG, J. C., AND MINNEMAN, S. L. Videodraw: a video interface for collaborative drawing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 1990), CHI '90, ACM, pp. 313–320.

[215] TEIXEIRA, L., LOAIZA, M., RAPOSO, A., AND GATTASS, M. Augmented reality using projective invariant patterns. In *Advances in Visual Computing*, vol. 5358 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2008.

[216] TOMBARI, F., SALTI, S., AND DI STEFANO, L. Unique signatures of histograms for local surface description. In *Proc. ECCV* (2010), pp. 356–369.

[217] TORSELLO, A., RODOLÀ, E., AND ALBARELLI, A. Multiview registration via graph diffusion of dual quaternions. In *Proc. CVPR* (2011), pp. 2441–2448.

[218] TSAI, R. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation 3*, 4 (1987), 323–344.

[219] TSONISP, V. S., CH, K. V., AND TRAHANIASLJ, P. E. Landmark-based navigation using projective invariants. In *Proceedings of the 1998 IEEE Intl. Conf. on Intelligent Robots and Systems* (Victoria, Canada, 1998), IEEE Computer Society.

[220] USABIAGA, J., EROL, A., BEBIS, G., BOYLE, R., AND TWOMBLY, X. Global hand pose estimation by multiple camera ellipse tracking. *Machine Vision and Applications 21*, 1 (2009), 1–15.

[221] VAISH, V., WILBURN, B., JOSHI, N., AND LEVOY, M. Using plane + parallax for calibrating dense camera arrays. In *In Proc. CVPR* (2004), pp. 2–9.

[222] VAN KAICK, O., TAGLIASACCHI, A., SIDI, O., ZHANG, H., COHEN-OR, D., WOLF, L., AND HAMARNEH, G. Prior knowledge for part correspondence. *Computer Graphics Forum 30*, 2 (2011), 553–562.

[223] VAN KAICK, O., ZHANG, H., AND HAMARNEH, G. Bilateral maps for partial matching. *Computer Graphics Forum 32*, 6 (2013), 189–200.

[224] VAN RHIJN, A., AND MULDER, J. D. Optical tracking using line pencil fiducials. In *Proceedings of the eurographics symposium on virtual environments* (2004).

[225] VESE, L. A., AND CHAN, T. F. A multiphase level set framework for image segmentation using the Mumford and Shah model. *IJCV 50*, 3 (2002), 271–293.

[226] VILLANUEVA, P. G., GALLUD, J. A., AND TESORIERO, R. WallShare: A collaborative multi-pointer system for portable devices. In *Proc. of PPD 2010, Workshop on designing multi-??touch interaction techniques for coupled public and private displays* (2010), ACM, pp. 31–34.

[227] WAGNER, D., REITMAYR, G., MULLONI, A., DRUMMOND, T., AND SCHMALSTIEG, D. Real time detection and tracking for augmented reality on mobile phones. *IEEE Transactions on Visualization and Computer Graphics 99* (2010).

[228] WALNY, J., LEE, B., JOHNS, P., RICHE, N. H., AND CARPENDALE, S. Understanding pen and touch interaction for data exploration on interactive whiteboards. *IEEE Trans. Vis. Comput. Graph. 18*, 12 (2012), 2779–2788.

[229] WANG, J., SHI, F., ZHANG, J., AND LIU, Y. A new calibration model of camera lens distortion. *Pattern Recognition 41*, 2 (2008), 607 – 615.

[230] WANG, Z., AND LOUEY, J. Economical solution for an easy to use interactive whiteboard. In *Frontier of Computer Science and Technology, 2008. FCST '08. Japan-China Joint Workshop on* (2008), pp. 197–203.

[231] WANNER, S., AND GOLDLUECKE, B. Spatial and angular variational super-resolution of 4d light fields. In *Computer Vision ECCV 2012*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7576 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 608–621.

[232] WARE, C., ARTHUR, K., AND BOOTH, K. S. Fish tank virtual reality. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (1993), CHI '93, pp. 37–42.

[233] WEBER, O., DEVIR, Y. S., BRONSTEIN, A. M., BRONSTEIN, M. M., AND KIMMEL, R. Parallel algorithms for approximation of distance maps on parametric surfaces. *ACM Trans. Graph. 27*, 4 (2008), 104:1–104:16.

[234] WENG, J., COHEN, P., AND HERNIOU, M. Camera calibration with distortion models and accuracy evaluation. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 10 (Oct. 1992), 965–980.

[235] WEYL, H. Über die asymptotische verteilung der eigenwerte. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse 1911* (1911), 110–117.

[236] WILSON, A. D. Playanywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology* (New York, NY, USA, 2005), UIST '05, ACM, pp. 83–92.

[237] WINDHEUSER, T., SCHLICKEWEI, U., SCHMIDT, F. R., AND CREMERS, D. Large-scale integer linear programming for orientation preserving 3d shape matching. *Computer Graphics Forum 30*, 5 (2011), 1471–1480.

[238] WINDHEUSER, T., VESTNER, M., RODOLA, E., TRIEBEL, R., AND CREMERS, D. Optimal intrinsic descriptors for non-rigid shape analysis. In *Proceedings of the British Machine Vision Conference* (2014), BMVA Press.

[239] WRIGHT, J., WAGNER, A., RAO, S., AND MA, Y. Homography from coplanar ellipses with application to forensic blood splatter reconstruction. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (june 2006), vol. 1, pp. 1250 – 1257.

[240] WU, J., PAN, G., ZHANG, D., QI, G., AND LI, S. Gesture recognition with a 3-d accelerometer. In *6th International Conference on Ubiquitous Intelligence and Computing* (Berlin, Heidelberg, 2009), UIC '09, Springer-Verlag, pp. 25–38.

[241] XIE, Y., AND OHYA, J. Efficient detection of ellipses from an image by a guided modified ransac. In *Image Processing: Algorithms and Systems* (2009), vol. 7245 of *SPIE Proceedings*, SPIE.

[242] YAMAZAKI, S., MOCHIMARU, M., AND KANADE, T. Simultaneous self-calibration of a projector and a camera using structured light. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on* (2011), pp. 60–67.

[243] Yan, J., Li, Y., Liu, W., Zha, H., Yang, X., and Chu, S. M. Graduated consistency-regularized optimization for multi-graph matching. In *Proc. ECCV* (2014), vol. 8689, pp. 407–422.

[244] Yang-Mao, S.-F., Lin, Y.-T., Lin, M.-H., Zeng, W.-J., and lien Wang, Y. Evaluation of mono/binocular depth perception using virtual image display. In *HCI International, 15th International Conference. Towards Intelligent and Implicit Interaction* (2013), vol. 8008 of *Lecture Notes in Computer Science*, Springer, pp. 483–490.

[245] Ying, X., and Hu, Z. Catadioptric camera calibration using geometric invariants. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 26*, 10 (Oct 2004), 1260–1271.

[246] Yoon, Y., DeSouza, G., and Kak, A. Real-time tracking and pose estimation for industrial objects using geometric features. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on* (sept. 2003), vol. 3, pp. 3473 – 3478 vol.3.

[247] Yu, X., Leong, H. W., Xu, C., and Tian, Q. A robust and accumulator-free ellipse hough transform. In *Proceedings of the 12th annual ACM international conference on Multimedia* (New York, NY, USA, 2004), ACM.

[248] Zach, C., Klopschitz, M., and Pollefeys, M. Disambiguating visual relations using loop constraints. In *Proc. CVPR* (2010), pp. 1426–1433.

[249] Zhang, H., and So, E. Hybrid resistive tactile sensing. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 32*, 1 (2002), 57–65.

[250] Zhang, S., He, W., Yu, Q., and Zheng, X. Low-cost interactive whiteboard using the kinect. In *Image Analysis and Signal Processing (IASP), 2012 International Conference on* (2012), pp. 1–5.

[251] Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2000), 2000.