Universitat Politècnica de Catalunya

Human Body Analysis using Depth Data

Ph.D. Thesis

Author:

Xavier Suau Cuadros

Thesis Advisors:

Javier Ruiz Hidalgo

Josep R. Casas Pla

October 2013

Oficina de Doctorat
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Acta de qualificació de tesi doctoral

Curs acadèmic:

Nom i cognoms

DNI / NIE / Passaport

Programa de doctorat

Unitat estructural responsable del programa

Resolució del Tribunal

Reunit el Tribunal designat a l'efecte, el doctorand / la doctoranda exposa el tema de la seva tesi doctoral titulada

_____

_____.

Acabada la lectura i després de donar resposta a les qüestions formulades pels membres titulars del tribunal, aquest atorga la qualificació:

☐ APTA/E ☐ NO APTA/E

(Nom, cognoms i signatura) President/a		(Nom, cognoms i signatura) Secretari/ària
(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal	(Nom, cognoms i signatura) Vocal

_____, _____ d'/de _____ de _____

El resultat de l'escrutini dels vots emesos pels membres titulars del tribunal, efectuat per l'Escola de Doctorat, a instància de la Comissió de Doctorat de la UPC, atorga la MENCIÓ CUM LAUDE:

☐ SI ☐ NO

(Nom, cognoms i signatura) Presidenta de la Comissió de Doctorat	(Nom, cognoms i signatura) Secretària de la Comissió de Doctorat

Barcelona, _____ d'/de _____ de _____

Als meus pares i a la meva família, que sempre m'han empès a aprendre.

I especialment a l'Ariadna, sense la qual, no tan sols aquesta tesi seria diferent, sinó també la meva vida.

Summary

Human body analysis is one of the broadest areas within the computer vision field. Researchers have put a strong effort in the human body analysis area, specially over the last decade, due to the technological improvements in both video cameras and processing power. Human body analysis covers topics such as person detection and segmentation, human motion tracking or action and behavior recognition. Even if human beings perform all these tasks naturally, they build-up a challenging problem from a computer vision point of view. Adverse situations such as viewing perspective, clutter and occlusions, lighting conditions or variability of behavior amongst persons may turn human body analysis into an arduous task.

In the computer vision field, the evolution of research works is usually tightly related to the technological progress of camera sensors and computer processing power. Traditional human body analysis methods are based on color cameras. Thus, the information is extracted from the raw color data, strongly limiting the proposals. An interesting quality leap was achieved by introducing the *multiview* concept. That is to say, having multiple color cameras recording a single scene at the same time. With multiview approaches, 3D information is available by means of stereo matching algorithms. The fact of having 3D information is a key aspect in human motion analysis, since the human body moves in a three-dimensional space. Thus, problems such as occlusion and clutter may be overcome with 3D information.

The appearance of commercial depth cameras has supposed a second leap in the human body analysis field. While traditional multiview approaches required a cumbersome and expensive setup, as well as a fine camera calibration; novel depth cameras directly provide 3D information with a single camera sensor. Furthermore, depth cameras may be rapidly installed in a wide range of situations, enlarging the range of applications with respect to multiview approaches. Moreover, since depth cameras are based on infra-red light, they do not suffer from illumination variations.

In this thesis, we focus on the study of depth data applied to the human body analysis problem. We propose novel ways of describing depth data through specific descriptors, so that they emphasize helpful characteristics of the scene for further body analysis. These descriptors exploit the special 3D structure of depth data to outperform generalist 3D descriptors or color based ones. We also study the problem of person detection, proposing a highly robust and fast method to detect heads. Such method is extended to a hand tracker, which is used throughout the thesis as a helpful tool to enable further research. In the remainder of this dissertation, we focus on the hand analysis problem as a subarea of human body analysis. Given the recent appearance of depth cameras, there is a lack of public datasets. We contribute with a dataset for hand gesture recognition and fingertip localization using depth data. This dataset acts as a starting point of two proposals for hand gesture recognition and fingertip localization based on classification techniques. In these methods, we also exploit the above mentioned descriptor proposals to finely adapt to the nature of depth data.

Resum

L'anàlisi del cos humà és una de les àrees més àmplies del camp de la visió per computador. Els investigadors han posat un gran esforç en el camp de l'anàlisi del cos humà, sobretot durant la darrera dècada, degut als grans avenços tecnològics, tant pel que fa a les càmeres com a la potència de càlcul. L'anàlisi del cos humà engloba varis temes com la detecció i segmentació de persones, el seguiment del moviment del cos, o el reconeixement d'accions. Tot i que els éssers humans duen a terme aquestes tasques d'una manera natural, es converteixen en un difícil problema quan s'ataca des de l'òptica de la visió per computador. Situacions adverses, com poden ser la perspectiva del punt de vista, les oclusions, les condicions d'il·luminació o la variabilitat de comportament entre persones, converteixen l'anàlisi del cos humà en una tasca complicada.

En el camp de la visió per computador, l'evolució de la recerca va sovint lligada al progrés tecnològic, tant dels sensors com de la potència de càlcul dels ordinadors. Els mètodes tradicionals d'anàlisi del cos humà estan basats en càmeres de color. Això limita molt els enfocaments, ja que la informació disponible prové únicament de les dades de color. El concepte *multivista* va suposar salt de qualitat important. En els enfocaments multivista es tenen múltiples càmeres gravant una mateixa escena simultàniament, permetent utilitzar informació 3D gràcies a algorismes de combinació estèreo. El fet de disposar de informació 3D és un punt clau, ja que el cos humà es mou en un espai tri-dimensional. Així doncs, problemes com les oclusions es poden apaivagar si es disposa de informació 3D.

L'aparició de les càmeres de profunditat comercials ha suposat un segon salt en el camp de l'anàlisi del cos humà. Mentre els mètodes multivista tradicionals requereixen un muntatge pesat i car, i una calibració precisa de totes les càmeres; les noves càmeres de profunditat ofereixen informació 3D de forma directa amb un sol sensor. Aquestes càmeres es poden instal·lar ràpidament en una gran varietat d'entorns, ampliant enormement l'espectre d'aplicacions, que era molt reduït amb enfocaments multivista. A més a més, com que les càmeres de profunditat estan basades en llum infraroja, no pateixen problemes relacionats amb canvis d'il·luminació.

En aquesta tesi, ens centrem en l'estudi de la informació que ofereixen les càmeres de profunditat, i la seva aplicació al problema d'anàlisi del cos humà. Proposem noves vies per descriure les dades de profunditat mitjançant descriptors específics, capaços d'emfatitzar característiques de l'escena que seran útils de cara a una posterior anàlisi del cos humà. Aquests descriptors exploten l'estructura 3D de les dades de profunditat per superar descriptors 3D generalistes o basats en color. També estudiem el problema de detecció de persones, proposant un mètode per detectar caps robust i ràpid. Ampliem aquest mètode per obtenir un algorisme de seguiment de mans que ha estat utilitzat al llarg de la tesi. En la part final del document, ens centrem en l'anàlisi de les mans com a subàrea de l'anàlisi del cos humà. Degut a la recent aparició de les càmeres de profunditat, hi ha una manca de bases de dades públiques. Contribuïm amb una base de dades pensada per la localització de dits i el reconeixement de gestos utilitzant dades de profunditat. Aquesta base de dades és el punt de partida de dues contribucions sobre localització de dits i reconeixement de gestos basades en tècniques de classificació. En aquests mètodes, també explotem les ja mencionades propostes de descriptors per millor adaptar-nos a la naturalesa de les dades de profunditat.

Agraïments

Un doble agraïment dirigit als meus directors de tesi, Javier Ruiz i Josep R. Casas. En primer lloc, per les propostes aportades, la implicació i l'interès durant tota la tesi. I en segon lloc pel tracte personal i humà que han tingut amb mi, que podríem resumir en amistat.

Un agraïment especial al Marcel i l'Adolfo, companys de despatx per davant i per darrere literalment, per les llargues xerrades, idees i consells. També a l'Albert pel gran suport donat que ha fet que aquesta tesi no durés el doble de temps. I al Jordi Pont per tots els cafès creatius.

Als del despatx del costat, per endolcir la tesi amb una infinitat de pastissos.

Índex

Índex de figures

Índex de taules

1

Introduction

Humans naturally understand the human body. From childhood, one is intrinsically capable of interpreting a human body, being able to recognize its pose and to differentiate its parts. Moreover, it is fairly easy for humans to carry out such interpretation under unfavorable conditions, *i.e.* cropped views, bad illumination or schematic representations. For example, one may say whether a given part is a hand or a foot from an uncommon, hand-made drawing (Figure 1.1), and even which gesture is that hand performing.

Such *easy* understanding of the human body by humans has stirred up the interest of the computer vision community. During the last years, cameras have spread almost everywhere, from mobile phones and web-cams to shopping malls and operating rooms. Moreover, the explosion of internet has put an enormous amount of pictures and videos at a *click* distance, many of them containing human beings. These two recent facts have pushed Human Body Analysis to the position of being one of the main computer vision research topics nowadays.

The computer vision community has addressed the Human Body Analysis (HBA) problem from various perspectives over the last years. Some works study the human body

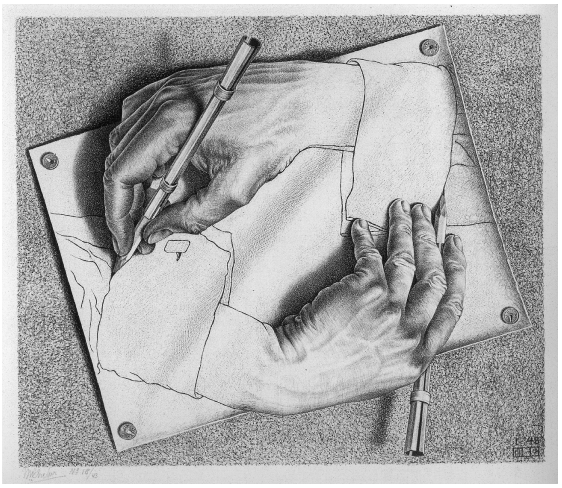

FIGURA 1.1: Hand-made drawing. Any human being is able to detect both hands and what are they doing, despite the uncommon representation.

using markers and other devices. Such approaches will not be mentioned in this document, focusing on the marker-less strategies. Indeed, in order to provide a truly immersive experience, marker-less body pose estimation is a must in this field [MHK06, Pop07].

The Human Body Analysis field is a wide research topic; however, one may identify some main research branches. Many works deal with human detection, that is to say, knowing where people are in an image or video capture. Other works focus on human parts detection and human pose estimation. Moreover, the analysis of specific body parts, mostly hands, has recently evolved as a main topic itself.

No matter the approach, being able to interpret the human body from a camera capture opens the door to completely new applications.

1.1 Humans and Human Body Analysis

Humans interact most of the time. Either with other humans, with animals or objects, interaction is omnipresent in our life. Interaction between humans is based on language, but also on signs and gesturing. Language understanding requires very strong hypotheses: both persons should speak the same language. However, gesture understanding is not that evident, since both interlocutors should have some cultural background in common. For example, in the Middle Ages, humans represented themselves to one another by means of gestures in political, religious and secular life [Sch90]. Hand gestures had a specific semantic meaning from the perspective of legal history in Middle Ages manuscripts [SCO11] (Figure 1.2). However, few people of the XXI century are able to understand such gestures. In general, a knowledge of gestures meaning and which parts of the body are representative is assumed in order to achieve gesture understanding.

(a) *pointing* (b) *swearing* (c) *speaking-b* (d) *speaking-f*

FIGURA 1.2: Medieval gestures, picture extracted from [SCO11].

This way, our interlocutor is able to *analyze* our body and try to *estimate* what we are communicating.

The same problem may be posed to Human-Computer Interaction (HCI). One could expect the same gesture understanding achieved between humans, and indeed, this is the main objective of computer vision approaches to Human Body Analysis: machines should understand human gestures as humans do. The eyes of the interlocutor are replaced by a camera sensor, whilst man intelligence is transferred to computer vision algorithms.

Applications are straightforward and extremely varied.

- Novel HCI interactive paradigms, where touch-less gestures become available.
- Sign language communication between a deaf person and a machine is a relevant application.
- Health related applications such as the analysis of athletic performance, as well as medical diagnostics.
- The capability to automatically monitor human activities using computers in security-sensitive areas such as airports or borders.
- Automatic annotation of large datasets for content-based queries in digital environments.
- Animation of virtual characters for movie production.
- And many applications to come.

1.2 The Emergence of Depth Data

Human Body Analysis techniques exploit the information provided by video cameras. Having 3D information at one's disposal is of great interest, since the human body moves in a 3-dimensional space. Traditional approaches use color, noted as RGB, as input information. By locating an array of cameras pointing to the object of interest (*i.e.*

human body), 3D information may be obtained by means of stereo triangulation, noted as RGBD (or color+depth). These family of methods are called *multiview triangulation* methods. Adding such spatial dimension to the available information enables a paradigm change in the definition of HBA algorithms. However, adopting multiview methods implies the verification of a series of hypotheses:

- The scene must be recorded with more than one camera. These cameras should have slightly different points of view.
- Cameras must be finely calibrated.
- Illumination should be controlled.
- The scene must have some kind of texture.

Such hypotheses are strongly restrictive in many situations. In addition, the recording setup turns out to be expensive and cumbersome, due to the number of cameras and the calibration procedure.

The appearance of depth cameras (Figure 1.3) has supposed a revolution, since a new dimension is added to the color information provided by conventional cameras. Indeed, these cameras provide an estimation of the distance D between the camera origin an the position of a given pixel in the scene.

A more in-depth overview of depth cameras and their characteristics is presented in Chapter 2.1.

FIGURA 1.3: Depth cameras have been commonly used in recent years. From left to right, PMD CamCube and MESA SR4000 (TOF cameras), Microsoft Kinect and ASUS Xtion (IR Stereo Cameras).

Depth cameras already provide 3D information using a single, low-cost, device. This property allows a considerable simplification of the above mentioned hypotheses for multiview methods. Furthermore, since recent depth cameras are based on matricial active triangulation and IR light coding [MZC12], illumination and texture are not as restrictive.

However, depth cameras present some important drawbacks to be taken into account:

- The operating range usually goes from 0.5 to 5 meters.
- Strong sunlight may degrade the depth estimation.
- Low image resolution (up to 640×480 pixels) if compared with color cameras.

Despite these drawbacks, the performance of RGB algorithms may be strongly improved by using depth information. Moreover, novel features and approaches may be developed solely based on depth data.

1.3 Contributions

This thesis summarizes a list of proposals for Human Motion Analysis using depth data. The work is divided into two main areas of contributions (Parts I and II), listed hereafter:

Part I **Description of Depth Data** How depth data is represented and described is a key aspect for a successful analysis. Description procedures aim at extracting specific features intrinsically available from the data that may be more useful than raw data for further analysis.

- An extension of Geometric Deformable Models to depth data, allowing to obtain end-effectors of a point-cloud, with special emphasis in the human body case.
- A descriptor for depth data analysis, called Oriented Radial Distribution (ORD). The proposed descriptor highlights extremal and prominent zones of a point cloud, as well as flat zones. Moreover, ORD is formulated in a multi-scale way, allowing size-wise filtering of prominent zones. We show that the ORD descriptor helps improving detection and classification methods using depth data.

Part II **Hand Analysis** Analyzing hand pose and gesturing has gradually become a sole research axis itself. Since humans intensely communicate through hands, being able to interpret them by means of Computer Vision techniques is crucial for the future of HCI. In this thesis we propose:

- ***HandBox*: A baseline method for hand detection** We address the problem of detecting human hands in a non-cumbersome way. The purpose of this method is to serve as a tool for further research on human hands. We propose the following contributions in this area:
  - A light-weight and robust head detector and tracker based on depth data.

  - A novel real-time hand tracking paradigm, called *HandBox*, which takes advantage of the previous head tracking proposal. HandBox enables automatic gesture segmentation and a wide range of interactive applications.

- *ColorTip*, a public dataset for hand gesture recognition and fingertip localization using depth data. We provide gesture and fingertip annotations for further training of classification methods or performance analysis of detection and tracking methods.
- A complete real-time framework for gesture recognition and fingertip localization, that exploits the Oriented Radial Distribution multi-scale feature for both tasks.
- A voting framework that jointly classifies object parts and global object classes. Special focus is put on fingertip localization.

A list of references to journal papers, international conference publications, contributions to projects and submitted publications is included in the conclusions Part III. Contributions discussed throughout the thesis document are also summarized in that part.

1.4 Thesis Organization

This thesis document is organized as follows:

Chapter 1 Introduction to the thesis subject and organization of the document.

Chapter 2 An overview of the relevant state-of-the-art publications and methods related to depth data acquisition and description, human body analysis and hand analysis.

Part I **Description of Depth Data** In this part we present our contributions to the field of depth data description.

Chapter 3 The 3D descriptor named Oriented Radial Distribution feature is presented in this Chapter.

Chapter 4 We propose a method to extend Geometrical Deformable Models to depth data

Chapter 5 Experimental results and conclusions about the two methods are presented.

Part II **Hand Analysis** This part contains the contributions to the Hand Analysis field using depth cameras.

Chapter 6 The HandBox approach is presented in this chapter, with special focus on the human body part detection case.

Chapter 7 A public dataset named *ColorTip* is presented.

Chapters 8 and 9 This chapters contains two proposed hand analysis methods.

Chapter 10 Experimental results about hand analysis are summarized in this chapter. Conclusions of these methods are also discussed.

Part III **Overall Conclusion** The final conclusions and list of contributions are included in this part.

Chapter 11 The contributions of the thesis are listed in this chapter.

Chapter 12 A final discussion on the work carried out in this thesis is included. Also, some future work directions are explored.

2

State of the Art

The computer vision community has addressed an increasing number of contributions to the problem of Human Body Analysis in recent years. The emergence of depth cameras has been one of the main driving forces. Some HBA topics are considered in this thesis: depth data acquisition, depth data description, body parts detection and tracking, and, finally, human hand analysis. We present hereafter a summary of the state-of-the-art approaches for each of these topics of interest.

2.1 Depth Data Acquisition

The first family of depth cameras was called *Time-of-Flight* (TOF) cameras. Such cameras measure the time of flight of a light signal between the camera and the subject for each point of the image. No complex processing is needed to convert a signal to an explicit distance, providing real-time captures of up to 100fps. One of the most used TOF cameras is the MESA SR4000 [tofcM], which provides a 176×144 pixel depth image.

In 2010, Microsoft released the Kinect sensor [Mic]. Kinect projects an IR pattern onto the scene, and applies 3D stereo techniques to recover the depth configuration of the scene. The depth estimation obtained with Kinect is more stable and accurate than

current TOF estimations. Moreover, Kinect provides a 640 × 480 pixel depth image, which means a considerably higher resolution than current TOF cameras [MZC12].

In this thesis, both the SR4000 and the Kinect sensor are used. However, most of the work is performed with the Kinect sensor, given its better characteristics. In addition, Kinect provides an RGB capture which is registered to the depth estimation. Thus, Kinect may be truly considered as an RGBD sensor.

Very recently, Kinect manufacturer PrimeSense has announced the release of a reduced version of Kinect, called Capri [Pri13]. Such device fits in regular tablets and mobile phones, and may be powered by USB, enabling novel applications that exploit the portability and power of tablets.

In 2012, a company called LeapMotion [Lea12] advertised a device able to provide hand tracking with sub-millimeter resolution. LeapMotion is also based on computer vision technology, consisting of 2 cameras and IR emitters. Its irruption caused a huge interest, however, after its very recent release [Rel13], some beta-testers indicate that the performance of LeapMotion is far from what it should be according to the advertisement [MIT13, Clu].

2.2 Depth Data Description

Depth data is obtained from the sensor as a depth map image. However, one may project each depth value to its 3D position. Thus, after back-projecting the values from a depth map, a 3D point cloud is obtained. Such point cloud contains 3D points which belong to the surfaces of the scene visible from the camera viewpoint. In this document, the term *depth image* refers to the 2D depth map, whilst the term *depth point cloud* refers to the 3D point cloud obtained from the projection of the values in the depth image.

2.2.1 3D Features

Depth point clouds may present specific characteristics which are barely observable from the raw data. In order to emphasize such characteristics, mathematical tools and methods can be specifically defined. Since we are faced with depth point clouds, the following summary considers features that take 3D structure into account.

One of the simplest 3D features is curvature, usually computed as $\frac{\lambda_0}{\lambda_0+\lambda_1+\lambda_2}$, where λ_i are the eigenvalues of the eigen-decomposition of a neighborhood of a given point. 3D Shape Contexts (3DSC) [FHK+04] capture, for a given point, the distribution over neighboring

positions of other shape points and summarizing global shape in a rich, local descriptor. Viewpoint Feature Histogram (VFH) [RBTH10] is a descriptor for depth point clouds that encodes geometry and viewpoint. More precisely VFH constructs a histogram that collects the pairwise pan, tilt and yaw angles between every pair of normals on a surface patch, and also between the viewpoint direction and each normal. Aldomà *et al.* have proposed the OUT-CVFH [ATRV12] extension to VFH that exploit color cues besides depth information . Tombari *et al.* [TSD10] propose the Signature of Histograms of OrienTations (SHOT) descriptor, using histograms of normals of the 3D volumes.

2.3 Body Parts Detection and Tracking

Computer vision methods have been applied to the field of Human Body Analysis with various objectives. Detection and segmentation of body parts is usually the foremost task of an HBA system, and usually, the rest of the processing strongly relies on the output of this first step. For that reason, detection and segmentation methods are still a major research topic within the computer vision community.

Human motion is understandable, most of the times, if analyzed over time. Thus, body part detection is tightly linked to the tracking of the body along time. That is to say, extracting motion information from a video sequence to establish matching between consecutive frames and exploiting their correlation in order to add robustness and to increase the performance of a system.

Depth cameras are being used both in generative and discriminative approaches for the detection and tracking of body parts.

Generative approaches use an explicit model to explain the data. Usually, an energy function is minimized to adapt the model to the observed data. The performance of these methods is tightly related to the model suitability (its ability to describe data), which turns the energy minimization step into an expensive task.

Discriminative approaches exploit machine learning techniques to directly infer parts from the data. Usually, no temporal information is used, resulting in frame-wise algorithms. Although we focus on depth-based approaches, interesting results could be obtained by adding color information.

Generative approaches may take advantage of discriminative techniques to handle model initialization and ambiguities [BMB+11].

2.3.1 Generative Approaches

In the literature, many of the works based on generative approaches come from the multiview problem, which consists in obtaining human body pose from a set of calibrated color cameras. Results in this area are impressive [SHG+11, WVT12] since complete 3D movement information is available. Gall *et al.* [GRBS08] introduce a multi-layer framework that combines stochastic optimization, filtering, and local optimization. In [GSD+09], Gall *et al.* go beyond pose estimation and cover possible non-rigid deformations of the body, such as moving clothes. Sundaresan and Chellappa [SC09] predict pose estimation from silhouettes and 2D/3D motion queues. Corazza *et al.* [CMG+09] generate a person-wise model which is updated through Iterative Closest Point (ICP) measures on visual-hull data. Pons-Moll *et al.* [PMBH+10] combine video images with a small number of inertial sensors to improve smoothness and precision of the human body pose estimation problem. Nevertheless, these 3D capture environments are very expensive and cumbersome to setup, since they require precise calibration and, usually, controlled illumination conditions. In addition, the computational cost of 3D methods is prohibitive and real-time is hardly achieved (*i.e.* we may have 24 parallel video streams).

On the other hand, very interesting works have studied how to extract human pose from single color cameras. In this direction, Guan *et al.* [GWBB09] obtain a synthesized shaded body. Body pose is estimated by searching into the learned poses, reflectance and scene lighting which most likely produced the observed pose. Yan and Pollefeys [YP08] recover the articulated structure of a body from single images with no prior information. In their work, trajectories of segmented body parts are mapped on linear subspaces to model the global body movement. Brubaker *et al.* [BFH09] use a simple lower-body model based on physical walking movement called *Antropomorphic Walker*, proposed by Kuo [Kuo02]. Hasler *et al.* [HAR+10] propose a pose estimation algorithm which performs on mono and multiple uncalibrated cameras. Unfortunately, single color cameras inherently provide poor information, due to information loss originated from perspective projection. Single-camera based methods are usually very specific and hardly generalize to different kinds of movement, scenes and view points.

Depth-based methods enable fast processing (1 video stream) and easy setups, at the expense of lower resolution and partial 3D information (single viewpoint). Recent depth-based methods take advantage of smart definition of an energy function, especifically designed for depth data, that may lead to impressive results [SM10, GPKT10a], and also extremely fast [GPKT12]. Schwarz *et al.* [SMMN11] robustly detect anatomical landmarks in the depth point cloud and fit a skeleton body model using constrained inverse kinematics. Zhu *et al.* [ZDF08] propose a tracking algorithm which exploits temporal consistency to estimate the pose of a constrained human model. Knoop *et al.*

[KVD06] propose a fitting of depth data with a 3D model by means of ICP. Grest *et al.* [GKK07] use a non-linear least squares estimation based on silhouette edges, which is able to track limbs in adverse background conditions. While many methods focus on upper-body pose, Plagemann *et al.* [PGKT10] present a fast method which localizes body parts on depth data at about 15 frames per second.

Baak *et al.* [BMB+11] combine local feature matching with a database look-up, achieving a fast and robust end-effector tracking. As mentioned, the human body model in [BMB+11] is initialized using a discriminative strategy, compensating the draw-back of one method with the second one.

Ganapathi *et al.* provide a useful dataset for human body analysis using depth data in [GPKT10a], which consists of 27 short sequences with increasing body motion and occlusion (increasing difficulty). Such dataset has widely been used during the last two years. Very recently, Ganapathi *et al.* published a second, much more challenging dataset [GPKT12].

Zhang *et al.* [ZSCL12] successfully propose one of the firsts attempts of multiview human body tracking using depth cameras.

2.3.2 Discriminative Approaches

Before diving into depth-based methods, we want to point out that interesting results are obtained on color images [VKMV09]. Hough Forests, proposed by Gall *et al.* [GYR+11], are also a successful example of the usage of a discriminative approach on color images for the detection of body parts.

Concerning depth information, the algorithm behind Kinect's human body pose estimation [SFC+11] trains a Random Forest to detect body parts. Recently, [TSSF12] take the part detection problem to the extreme, considering every pixel a body part. In a second step, they perform a one-step matching of a canonical body pose to the test pose, using the classified pixels as objective. All of these methods require a large training dataset (*i.e.* +900K images in [SFC+11]), usually cumbersome to obtain. The use of synthetic data is an interesting strategy to easily enlarge the dataset [SFC+11, KKKA12b].

López and Casas [AC12] propose to detect specific body gestures by means of an unbalanced Random Forest approach. Their approach is largely real-time and robust, allowing frame-wise tracking of these gestures over time.

2.4 Hand Analysis

Early works tackled only static hand gesture recognition on depth images. They where mainly based on local and very specific descriptors. Mo and Neumann propose in [MN06] one of the first works attempting hand gesture recognition using a laser-based camera to produce low-resolution depth images. They interpolate hand pose using basic sets of finger poses and inter-relations. Liu and Fujimura [LF04] recognize dynamic hand gestures using Time-of-Flight depth images. Hands are detected measuring shape similarity by means of the Chamfer distance. They analyze the trajectory of the hand and classify gestures using shape, location, trajectory, orientation and speed features.

Recent methods attempt to obtain hand parts and to track them over time, as well as recognizing hand poses. These recent works can also be classified into generative and discriminative.

2.4.1 Generative Approaches

As has happened with complete body tracking algorithms, multiview approaches have achieved great precision [BTG+12]. However, computational time is still a major drawback of these methods (*i.e.* about 30 seconds per frame in [BTG+12]).

A few works have recently obtained interesting results by means of generative approaches using depth cameras. Oikonomidis *et al.* [OKA11] formulate the hand pose recovery problem as an optimization approach, measuring the discrepancy between a model and the observed hand. Full hand pose is provided (including fingertips), requiring initialization at a known initial pose. On the other hand, their cost function relies on color information, reducing the performance to controlled scenarios.

A small number of works perform fingertip localization using a generative approach. The work in[HMB11, MIT11] perform fingertip detection, but do not distinguish between fingers. In the latter, palm and fingers orientation are estimated. Both approaches exploit geometrical features to detect fingertips on the hand point cloud.

2.4.2 Discriminative Approaches

More works have tackled the hand analysis problem from a discriminative point of view. Most of these works focus on two kinds of application:

- Interactive interfaces [SPHK08, VKMBV09, RYZ11, HMB11].

- Deaf language systems, with a strong effort put on the American Sign Language (ASL) [KKKA12a, ZYT13, ZW12, PB11, UGVV11].

Many authors have adopted a Nearest Neighbor (NN) classification strategy to overcome fingertip localization. In [SPHK08] a NN classification into five hand gestures is proposed, using geometrical descriptors. Color cues are added to depth data in [VV11] in order to build a NN classification framework. In [KPHB08], simple features are projected on two axis, and NN is then performed into 12 ASL letters. Novel descriptors like in [ZYT13] have also been recently proposed. In [RYZ11], the Earth Movers Distance is adapted to finger signatures and used as cost function for further NN classification.

Another commonly adopted strategy is to classify gestures using variations of the Random Forests (RF)[Bre01] approach. In *et al.* [KKKA11, KKKA12a] a set of randomized decision trees is used to classify hand shapes, in a similar manner than in [SFC+11], taking advantage of a large synthetic dataset. A multi-resolution Gabor filtering of the hand patch is used to train a RF classifier in [PB11].

Other classification strategies may also be adopted. For example, linear Support Vector Machines (SVM) are used in [ZW12], combining color and depth descriptors to predict the hand gesture.

Part I

Description of Depth Data

Overall introduction

Obtaining information of the human body from camera frames is a challenging task. There exists a large variety of 2D descriptors [BETV08, BTV06, Low04, AOV12, LCS11, RRKB11], widely used for color image description and matching. Such descriptors are based on color or intensity gradient at specific salient points of the input image. Thus, they require a detection step (*i.e.* FAST detector [RD06]) that selects the best suited points where to calculate the descriptor.

However, these 2D descriptors rely on rich images with a minimum amount of variation so that the detector is able to locate points, and the descriptor is selective enough. Depth images do not provide such richness. Indeed, depth images are usually composed of fairly flat patches (surfaces) [RHMA+11] surrounded by sharp edges (objects border). Color and intensity variations are lost during the depth map estimation procedure. This lack of texture makes classical 2D descriptors not sufficient at describing depth images.

We propose to use 3D features, that is to say, descriptors that take 3D structure into account. A summary of existing 3D features has been included in Chapter 2. These family of descriptors are able to characterize a point cloud by analyzing structural information, such as topology, salience, flatness or density.

We propose hereafter two novel descriptors on depth data. In chapter 3 we propose the Oriented Radial Distribution (ORD) descriptor, which emphasizes prominent zones of the input point cloud at various scales. We show the ability of ORD in describing the overall image, as well as local zones of the point cloud.

In chapter 4, we propose an extension of the Geometric Deformable Models theory that takes the 3D connectivity of depth point clouds into account, obtaining an alternative formulation of the geodesic distance.

The concepts of physical area and connectivity that appear in this thesis are detailed in Appendix A.

3

Oriented Radial Distribution

3.1 Introduction

We introduce in this chapter the Oriented Radial Distribution (ORD) descriptor for 3D point clouds. ORD exploits 3D points neighborhoods in order to characterize the distribution of points around the central one. Moreover, ORD adapts to the surface normals, providing an improved invariance with respect to other non-oriented methods [FHK+04]. Such feature presents high values on prominent zones of the point cloud, and low values on flat and interior zones. Therefore, it is convenient to use this feature to detect end-effectors on depth data.

We also provide a simple method to classify end-effectors into head, hand and foot solely based on the ORD output. Some probabilistic descriptors, which categorize position, size and shape, are computed over the ORD high-responsive zones. The obtained end-effectors are used to reinforce the detection in further frames, providing a temporal dimension which increases the robustness of the algorithm.

3.2 Related Work

Some authors have proposed 3D descriptors that are mainly used in the robotics field. For example, Rusu *et al.* [RBTH10] propose the Viewpoint Feature Histogram (VFH) descriptor, which is able to characterize a point cloud depending on its geometry, but also on the viewpoint. Some extensions to this world have recently been proposed. We remark the color extension proposed by Aldomà *et al.* [ATRV12]. Frome *et al.* [FHK+04] proposed the 3D Shape Contexts (3DSC) descriptor, providing a dense description of the shape of a point cloud using a partitioned sphere. Our proposal is inspired by this approach, replacing the sphere by a partitioned disk oriented coherently with the point cloud normals. Tombari *et al.* [TSD10] propose the Signature of Histograms of OrienTations (SHOT) feature. In the SHOT method, a set of local histograms of normals is calculated over the 3D volumes defined by a 3D grid superimposed on the support and then grouping together all local histograms to form the final descriptor. The normal estimation is based on the Eigenvalue Decomposition of a novel scatter matrix defined by a weighted linear combination of neighbor point distances to the feature point, lying within the spherical support. The eigen-vectors of this matrix define repeatable, orthogonal directions in presence of noise and clutter. In Section 10.4.1.3 we provide a comparison of ORD with the mentioned 3D features at the task of hand gesture recognition.

3.3 Description of the Oriented Radial Distribution

A depth point cloud is obtained after mapping the depth pixels in the real world coordinate system. Such point cloud corresponds to a sampling of the visible scene surface from the camera viewpoint. With a simple foreground segmentation, using a depth threshold with respect to the empty scene (background), the subset Ω containing the foreground objects is obtained (mainly human body, in this thesis).

After simple inspection of Ω (Figure 3.1), one may notice that the head, hands and feet are relatively visible on the depth map. Indeed, these five end-effectors share the common characteristics of being extrema of Ω, and also of having a similar size. Therefore, being able to determine whether a pixel is located in an extremal zone of a given size is convenient to detect the end-effectors of a human body.

With this purpose, we propose a feature $\Theta : \{\mathbf{z}, \Omega, \xi\} \longmapsto \mathbb{R}$ to measure the *Oriented Radial Distribution* of the neighborhood $\mathcal{N}_{\mathbf{z}}^{\rho}$ around a point $\mathbf{z} \in \Omega$ (an example is provided in Figure 3.1). More precisely, let the neighborhood $\mathcal{N}_{\mathbf{z}}^{\rho}$ of $\mathbf{z}$ be all those points $\{\mathbf{z}_i\} \in \Omega$ such that $|\mathbf{z} - \mathbf{z}_i| < \rho$. In other words, $\mathcal{N}_{\mathbf{z}}^{\rho}$ contains all the points located in

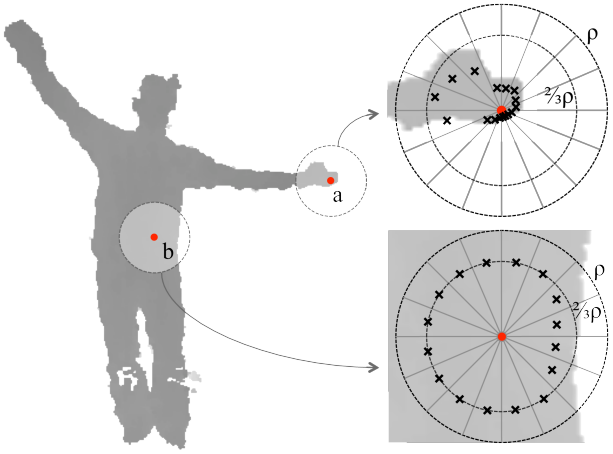

FIGURA 3.1: Oriented Radial Distribution feature computation example. The point a belongs to an extreme, the distances between the dark crosses $\bar{\delta}_j$ and the barycenter of each zone $\frac{1}{\sqrt{2}}\rho$ present high values. On the other hand, point b belongs to a relatively uniform area, and most of the $|\bar{\delta}_j - \frac{1}{\sqrt{2}}\rho|$ distance values are very low.

a ball of radius ρ centered at $\mathbf{z}$. Therefore, the radius ρ is a parameter of the proposed feature. Such radius, and other parameters explained in this chapter, are noted as ξ.

The tangential plane $\mathcal{T}_\mathbf{z}$ at $\mathbf{z}$ is roughly estimated by Principal Component Analysis (PCA) of the points surrounding $\mathbf{z}$, the two principal axis of the PCA determining $\mathcal{T}_\mathbf{z}$. Then, all the points $\mathbf{z}_i \in \mathcal{N}_\mathbf{z}^\rho$ are projected onto $\mathcal{T}_\mathbf{z}$. Therefore, a disk $\mathcal{D}_\mathbf{z}^\rho$ of radius ρ is obtained, containing all the projections of the points in the neighborhood of p. Projecting the neighborhood of every point in Ω onto its tangent plane is a key aspect of the proposed algorithm (Figure 3.2), making the feature Θ consistent over the whole point cloud Ω.

If the central point p is not located close to an extreme, it is likely to be surrounded by a regular amount of points in all directions. On the other hand, points located close to extremal zones only present neighbors in some directions. In order to measure whether point p is located close to an extremal zone of Ω or not, the content of $\mathcal{D}_\mathbf{z}^\rho$ is analyzed. Indeed, $\mathcal{D}_\mathbf{z}^\rho$ is divided into K equal zones as shown in Figure 3.1 ($K = 16$ in the example), where $K \in \xi$ is a parameter. These zones, noted Δ_j with $j = 1..K$, present two equal sides of length ρ and a third side of length $\frac{2\pi\rho}{K}$. The average distance $\bar{\delta}_j$ between the points in each zone Δ_j and the central point p is calculated, as shown in Figure 3.1. Therefore, a $\bar{\delta}_j$ value characterizes every zone Δ_j. Different situations may happen:

- Those Δ_j zones being completely filled with points will have a $\bar{\delta}_j$ value very close to $\frac{1}{\sqrt{2}}\rho$, which is the radius of barycenter of the zone (divides the zones into two equal areas).

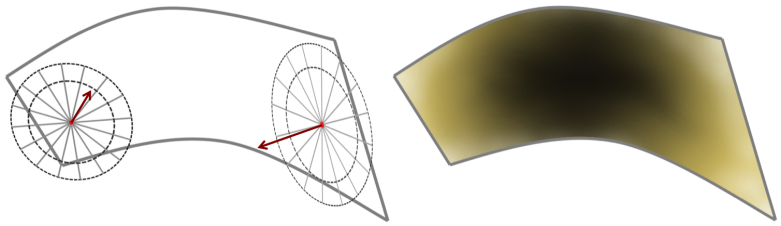

FIGURA 3.2: Orientation of the measure disks $\mathcal{D}^{\rho}_{\mathbf{z}}$ depending on the tangent plane to the depth surface at the measuring point (center of the disk). Two disks are illustrated in this example, and the resulting Θ feature on the right (lighter color corresponds to high Θ values).

- On the contrary, partially filled zones will result in a higher distance between $\bar{\delta}_j$ and $\frac{1}{\sqrt{2}}\rho$.

The *ORD* feature Θ is constructed as the average distance between the $\bar{\delta}_j$ and $\frac{1}{\sqrt{2}}\rho$, as shown in Equation (3.1). The obtained result is normalized with the maximal value $\frac{1}{\sqrt{2}}\rho$, so that $\Theta \in [\frac{1-\sqrt{2}}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$. Therefore, the dynamic range of Θ is 1. Only those zones Δ_j filled with more than 10 points are considered, resulting in a subset of K_f zones taken into account to compute Θ.

$$\Theta(\mathbf{z}, \Omega, \xi) = \frac{1}{\frac{1}{\sqrt{2}}\rho K_f} \sum_{j=0}^{K_f} \left(\bar{\delta}_j - \frac{1}{\sqrt{2}}\rho\right) \quad with \quad \xi = \{\rho, K\} \tag{3.1}$$

3.3.1 Effect of the parametrization ξ

Two main parameters are involved in the computation of Θ, noted as $\xi = \{\rho, K\}$. The radius ρ determines the size of the neighborhood around p which will be analyzed. Indeed, the feature Θ, parametrized with a given ρ, will return its greatest values at the extrema presenting a radius similar to ρ. Thus, small radius return high values at thin extrema of Ω, while larger radius detect larger extrema. Therefore, the radius value may be used to *filter* some extrema while preserving others. For example, to find a human head, a radius of about $\rho = 15\,cm$ should be appropriate, which will filter other noisy and smaller extrema. The second parameter in the calculation of Θ is the number of zones K into which the oriented disk $\mathcal{D}^{\rho}_{\mathbf{z}}$ is divided. Low values of K (*i.e.* K=2) lead to noisy and non-robust detection of end-effectors, the spatial resolution of the Θ feature being too poor. On the other hand, high values of K (*i.e.* K=128) provide a too smooth transition between end-effectors and the non-desired zones. Reasonable trade-off values is between 8 and 16 divisions (see Figure 3.3).

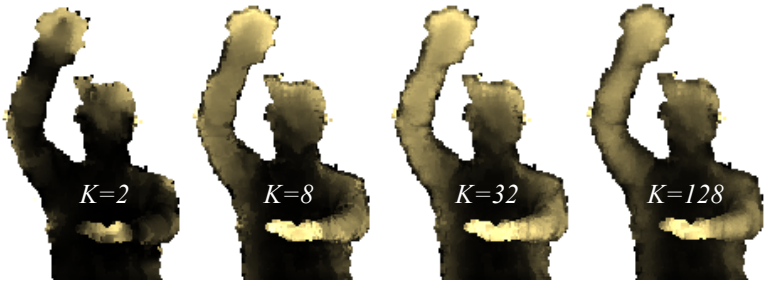

FIGURA 3.3: Effect of the parameter K on the ORD feature.

3.3.2 GPU implementation of ORD

The way the proposed ORD descriptor is defined nicely suits a parallel implementation. Indeed, ORD is computed for every point $\mathbf{z}$ in the point cloud, only using the neighborhood of $\mathbf{z}$ as input data. If we are capable of providing an input point and its neighborhood to a GPU core, we may compute the ORD descriptor in an extremely fast way.

In order to do so, we voxelize the input point cloud Ω into $N_{GPU} \times N_{GPU}$ voxels in the XY axes. These voxels cover the complete width and height of Ω, and all of them have a depth equal to maximal depth span of Ω.

We design the implementation so that a GPU kernel can access to the lists of points of a voxel V and its adjacent voxels $\{V_a\}$. Therefore, the N_{GPU} value is critical. A small value will include a large amount of points in every voxel, saturating the GPU memory and reducing the parallel processes. Thus, the maximal size of a voxel is limited by the hardware memory.

On the other side, a high value of N_{GPU} creates small voxels, cropping the neighborhoods of the central points. Since neighborhoods are altered, the output result is erroneous. The minimal size of a voxel will then be equal to the ORD radius ρ, so that any point in the central voxel has its complete neighborhood inside $V \cup \{V_a\}$.

3.4 Classification of end-effectors using probabilistic descriptors

A set of candidate end-effectors is obtained after the calculation of Θ on Ω. As shown in Figure 3.4, those points $\mathbf{z}$ over a threshold Θ_{min} (usually $\Theta_{min} \approx 0.15$) are labeled as candidate points, and [1]clustered into candidate blobs $\{\mathcal{B}\}$. We look for those five zones

[1]Euclidean Clustering

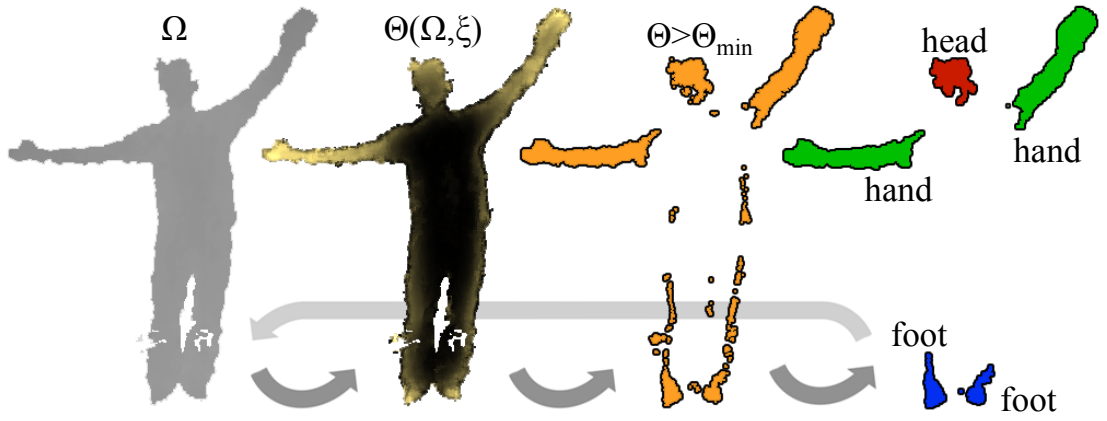

FIGURA 3.4: Summary of the proposed method. From left to right: Segmented depth map Ω, *Oriented Radial Distribution* values Θ, candidate points $\Theta > \Theta_{min}$ and labeled end-effectors.

with maximal Θ being large enough to be considered end-effectors. Therefore, all the very small blobs observed in Figure 3.4 (third column) are omitted, only keeping the largest ones, up to five candidate blobs. Remark that sometimes less than five blobs are obtained (*i.e.* in the case of a hidden hand).

We remark that the size, shape and position of the blobs $\{\mathcal{B}\}$ depends only on the input point cloud and the ORD feature. Therefore, we propose three descriptors that are calculated for these candidate blobs, in order to classify them into $\gamma_i = \{head, hand, foot\}$, as in [PGKT10]. These descriptors, for a given blob $\mathcal{B}$ with centroid $\mathcal{B}^c$, are:

Y **- Position** The relative height (vertical y axis) with respect to the centroid Ω^c of Ω is calculated as: $Y = (\mathcal{B}^c_y - \Omega^c_y)\, cm$ (vertical coordinates of $\mathcal{B}^c$ and Ω^c).

S **- Size** The estimated size of $\mathcal{B}$, calculated from the apparent area of $\mathcal{B}$ on the depth image. A quadratic law Γ relating this apparent area and the physical one is obtained empirically for the Kinect sensor. More precisely, the conversion from a single pixel at depth $\mathbf{z}$ to a real world surface is $\Gamma_{pix}(\mathbf{z}) \approx 1.12 \cdot 10^{-6} \cdot \mathbf{z}^2 + 8.41 \cdot 10^{-5} \cdot \mathbf{z} - 4.64 \cdot 10^{-3}$ (Appendix A.1). Therefore, the size descriptor of a blob $\mathcal{B}$ containing N_B points is: $S = (N_{\mathcal{B}} \cdot \Gamma(\mathcal{B}^c_z))\, cm^2$.

A **- Shape** The shape descriptor is defined as the relation between the second α_2 and first α_1 eigenvalues of the PCA decomposition of $\mathcal{B}$. Thus, $A = \frac{\alpha_2}{\alpha_1}$, which gives an idea of the *roundness* of $\mathcal{B}$. Very elliptical shapes result in low A values, while very round $\mathcal{B}$ shapes result in A values near to 1.

These descriptors $\lambda_k = \{Y, S, A\}$ are analyzed over 1300 frames containing various human poses with annotated head, hand and foot parts. The statistical moments (mean μ and variance σ^2) of the obtained blobs are calculated for every extremity group $\gamma_i = \{head, hand, foot\}$ and for every descriptor (Table 3.1).

We propose to construct the probability density functions (PDF or f) which evaluate the probability of a given descriptor to belong to a given extremity group. Such PDF are considered Gaussian, centered at $\mu_{\lambda_k}^{\gamma_i}$ with a standard deviation $\sigma_{\lambda_k}^{\gamma_i}$, as shown in Equation (3.2). Therefore, for each candidate blob $\mathcal{B}$ we may calculate the probability of belonging to a given group γ_i depending on the three descriptors. The combined probability of a blob belonging to a group γ_i is defined as the product of the separate probabilities of every descriptor of $\mathcal{B}$ belonging to γ_i (Equation (3.2)).

$$\begin{aligned} P(\mathcal{B} = \gamma_i) \quad &= P\Big((Y_B = \gamma_i) \wedge (S_B = \gamma_i) \wedge (A_B = \gamma_i)\Big) \\ &= f_Y^{\gamma_i}(\mathcal{B}) \cdot f_S^{\gamma_i}(\mathcal{B}) \cdot f_A^{\gamma_i}(\mathcal{B}) \\ \text{with PDF:} \quad f_{\lambda_k}^{\gamma_i}(\mathcal{B}) &= \tfrac{1}{\sigma_{\lambda_k}^{\gamma_i}\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\lambda_k(\mathcal{B}) - \mu_{\lambda_k}^{\gamma_i}}{\sigma_{\lambda_k}^{\gamma_i}}\right)^2} \end{aligned} \tag{3.2}$$

A decision about whether a blob belongs to any of the γ_i groups is taken, based on the obtained probabilities. Those candidate blobs with a probability of belonging to any of the groups γ_i smaller than $P(\mathcal{B} = \gamma_i) < 10^{-6}$ are not considered. The remaining candidate blobs are classified into $\{head, hand, foot\}$ depending on their probabilities, restricted to two feet, two hands and one head (Figure 3.4, farther right).

λ_k	$\mu_{\lambda_k}^{head}$	$\sigma_{\lambda_k}^{head}$	$\mu_{\lambda_k}^{hand}$	$\sigma_{\lambda_k}^{hand}$	$\mu_{\lambda_k}^{foot}$	$\sigma_{\lambda_k}^{foot}$
Y	62.18	7.48	29.43	29.06	-71.31	10.89
S	58.58	10.00	64.24	24.89	46.68	10.75
A	0.58	0.17	0.11	0.13	0.41	0.19

TAULA 3.1: Statistical moments of the descriptors

Temporal weighting of the Oriented Radial Distribution feature

The objective of this operation is to increase the robustness and consistence of the detection of end-effectors over time. The Θ values at time $t+1$ are weighted according to the location of the end-effectors at time t. The set of end-effectors locations is noted $\mathcal{L}$. More precisely, a distance factor $\tau \in [0, 0.25]$ is added to every $\Theta(\mathbf{z}, \Omega, \xi)$ value, where τ decreases exponentially with the distance between the point $\mathbf{z}$ and $\mathcal{L}$, as shown in Equation (3.3).

$$\tilde{\Theta}_{t+1}(\mathbf{z}, \Omega, \xi) = \Theta_{t+1}(\mathbf{z}, \Omega, \xi) + \tau_t \quad | \quad \tau_t = \frac{1}{4} e^{-\frac{|\mathbf{z} - \mathcal{L}|}{10}} \tag{3.3}$$

4

Extending Geometric Deformable Models to Depth Data

4.1 Introduction

Some priors may be used for the detection of human body parts. We know before-hand how a human body should look like, its probable size and distribution, the length of its limbs, etc. However, these magnitudes must be measured in some way, and usual distances (*i.e.* Euclidean) are sometimes not sufficient.

The use of geodesic distances provides interesting results for human pose estimation [CMMM08], thus methods able to compute such distance may be of interest in the research community. We propose a method to compute geodesic distances on point clouds from a root zone, by extending Geometric Deformable Models (GDM) to depth data. Furthermore, the method formulation enables the creation of a directed graph that may be exploited to detect end-effectors and infer human pose.

4.2 Related Work

Geometric deformable models, proposed independently by Caselles *et al.* [CCCD93] and Malladi *et al.* [MSV95], have proved performance and flexibility at describing topology, as stated by Han *et al.* [HXP03]. GDM have been widely applied in the field of image [MM09] and volume [WBMS01] segmentation and component analysis.

Even if the GDM theory is formulated on the continuum, it may be implemented in a discrete domain. An efficient and simple implementation of the GDM is known as the *Narrow Band Level Set* method (NBLS), introduced by Adalsteinsson and Sethian [AS95], which restricts computation in thin bands surrounding a zero level. Periodic updates of these bands gradually cover the full area (or volume) of the analyzed data set, preserving its topology. The NBLS method is defined for organized points in an evenly spaced grid (pixels in 2D, voxels in 3D), which limits accuracy due to re-sampling. Rosenthal *et al.* [RML10] propose an implementation of the NBLS method for unorganized 3D points, preserving their actual position.

4.3 Geodesic distance estimation using Geometric Deformable Models and Narrow Band Level Sets

Geometric deformable models are based on the theory of curve evolution and the level set method [Set99]. The basic idea is to deform an initial curve or contour, which is registered to the data domain, depending on some pre-defined external and internal forces. Internal forces will expand the actual curve over the data keeping it smooth. External forces are computed from the available data and have an effect on the curve evolution. By way of example, imagine a drop of corrosive acid eating an object. The initial curve will be the drop at time zero, internal forces depend on the amount of acid in the drop, its corrosive power, etc. and external forces depend on the resistance of the underlying material. Thus, corrosion will slow down in resistive zones of the material and advance faster in areas more prone to corrosion.

In our case, external forces are computed from the depth point cloud $D = \{\mathbf{z}_i\} \in \mathbb{R}^3$ and are defined to respect and preserve data features like topology or borders. We recall that D is the depth point cloud corresponding to the object of interest, obtained from the captured depth image $\mathcal{I}$.

FIGURA 4.1: Summary of the steps involved in the proposed algorithm for human pose estimation. From left to right: Foreground mask, R-NBLS propagation, R-NBLS filtering, end-effector graph nodes and end-effectors found with the associated skeleton.

We propose an adapted version of the NBLS method in the context of depth data. The input depth point cloud is analyzed as a sampling of the actual surfaces in the scene (from the camera viewpoint). The objective is to exploit connectivities over these depth surfaces in order to extract topological features. Generally speaking, the proposed method locates the end-effectors of any 3D object sampled into a 3D point cloud. Since our work focuses on body pose estimation, the end-effectors are mainly the four extremities of a human body (Figure 4.1). However, other 3D objects have been studied, showing and example with the human hand in Section 5.3.3.

Additionally, Human pose is inferred from the NBLS output together with the obtained end-effectors: firstly populating a graph from the previously computed NBLS and, secondly, extracting extremity pose with a shortest path algorithm from the end-effectors.

Let $\phi(\mathbf{z},t):\mathbb{R}^3\rightarrow\mathbb{R}$ be a level set function whose sole purpose is to provide an implicit representation of the evolving curve. Let also $\mathcal{C}(s,t):\mathbb{R}^2\rightarrow\mathbb{R}^3$ be a contour parametrized by s as the zero level set of $\phi(\mathbf{z},t)$ and $L_t^0\subset D$ be the subset enclosed by $\mathcal{C}(s,t)$. Remark that $\mathcal{C}$ is parametrized in a two dimensional space, which is particular to the depth data case. Equation (4.1) defines ϕ at a given time instant t. In the level set notation, time represents the advance of $\mathcal{C}(s,t)$, $t=0$ being the time-stamp of the initial curve. In order to efficiently perform nearest neighbor queries to evaluate the Euclidean distance $dist_E$ between points, D is organized as a *kd-tree* structure.

$$\phi(\mathbf{z},t)=\begin{cases}0 & \forall\mathbf{z}\in L_t^0\\ \min\{dist_E(\mathbf{z},\mathcal{C}(s,t))\} & \forall\mathbf{z}\notin L_t^0\end{cases} \tag{4.1}$$

The objective is to make the contour $\mathcal{C}$ evolve over D preserving the topological properties of the latter. As cited in [HXP03], the NBLS method is a simple solution to implement GDM evolution. An NBLS version for unorganized $\mathbb{R}^3$ points has also been presented in [RML10]. In the NBLS method, the level set function ϕ is evaluated in

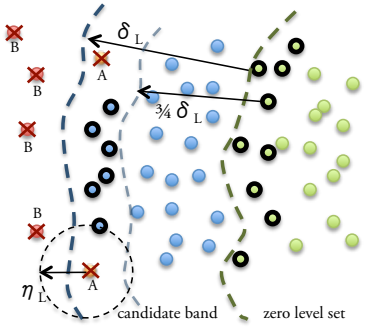

FIGURA 4.2: R-NBLS propagation example. The green points are the actual zero level set L_t^0, and those with a thick black boundary form the actual contour $\mathcal{C}(s,t)$. The blue points in the middle are the candidate narrow band of with δ_L, with its contour also marked with thick point boundaries. Points labeled A (orange) are rejected because of the density condition in Equation (4.2). Points labeled B (red) are rejected because of the proximity condition also in Equation (4.2).

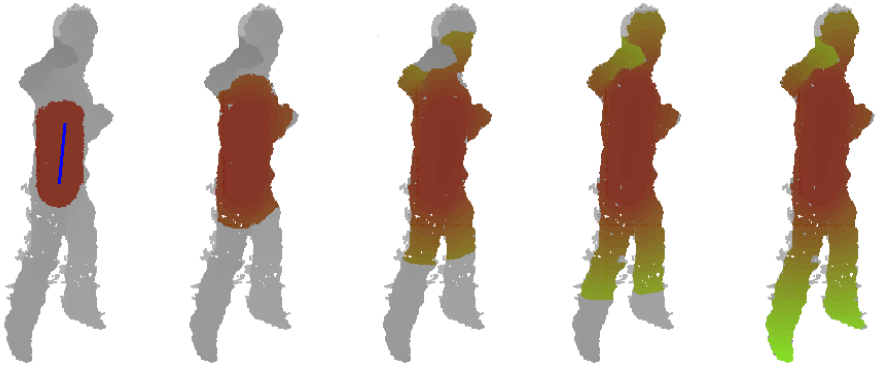

FIGURA 4.3: Example of the proposed R-NBLS propagation algorithm. The image on the left shows the foreground raw depth data D and the initial zero level set, in blue. From left to right, propagation iterations $k = 5, 10, 15, 23$ respectively. Propagation stopped at $k = 23$. Note the noise filtering at depth edges (zones in gray in the last column) as well as the correct propagation of the forearm.

a thin layer surrounding the actual zero level set in order to update the zero level set for the next time instant. Such approach limits the number of calculations to these few surrounding points.

We propose to add a density condition to the existing proximity condition. The role of this density condition is to filter the data, especially those points near depth edges. Using the above mentioned acid drop example, the density condition may be considered as an external force, since it is implicitly derived from the dataset. Propagation will slow down or stop in zones with low data density, and continue in highly populated zones. Thus, only those end-effectors *densely* connected to the main body will be considered, filtering sparsely represented or very thin ones.

We propose to update the zero level set according to Equation (4.2). We note that time t may be considered as a discrete time, where $t_k := t + k$ with $k \in \{0, \mathbb{N}\}$.

$$
\begin{aligned}
L_{t+1} = \{\mathbf{z}_i\} \quad \text{if} \quad & \begin{cases} \phi(\mathbf{z}_i, t) < \delta_L & \text{(proximity)} \\ & \textit{and} \\ \mathbf{z}_i \text{ is } (\eta_L, \delta_L)\text{-connected} & \text{(density)} \end{cases} \\
L^0_{t+1} = L^0_t \cup L_{t+1} &
\end{aligned}
\tag{4.2}
$$

The candidate narrow band is noted as L_{t+1} and its maximal width is δ_L, determined by the *proximity* condition. The connectivity property of $\mathbf{z}_i$ is used as *density* condition, ensuring that the space surrounding $\mathbf{z}_i$ is dense enough (at least η_L are close enough to $\mathbf{z}_i$), as shown in Figure 4.2. Therefore, δ_L and η_L are parameters of the proposed NLBS variant, called Restricted-NBLS or R-NBLS.

In order to complete the formulation, we should define how the contour $\mathcal{C}$ is updated in the depth context. In practice, the candidate points L_{t+1} which are farther from the previous zero level set are taken as the new contour $\mathcal{C}(s, t+1)$ as shown in Equation (4.3).

$$
\mathcal{C}(s, t+1) = \{\mathbf{z}_s\} \in L_{t+1} \quad \textit{with} \quad \phi(\mathbf{z}_s, t) \in \left[\tfrac{3}{4}\delta_L, \delta_L\right] \tag{4.3}
$$

Thus, iterating through Equations (4.1), (4.2) and (4.3) from an initial zero level set $L^0_{t_0=0}$, the sufficiently dense zones of D will be covered.

The geodesic distance between $L^0_{t_0}$ and a given point $\mathbf{z}_k$ which was added to L^0 at the time instant t_k (or iteration k) may be calculated with Equation (4.4).

$$
dist_G(L^0_{t_0}, \mathbf{z}_k) \approx k \cdot \delta_L + \phi(\mathbf{z}_k, t_k) \tag{4.4}
$$

The iterative R-NBLS method stops when the number of points N^C_t of the actual contour $\mathcal{C}(s, t)$ is smaller than a given stop threshold. The proposed iterative framework stops when $N^C_t = 0$ (see the second shape in Figure 4.1 for an example). An example of the R-NBLS method is shown in Figure 4.3, the narrow bands at every iteration being colored from the initial zero level set, which has been calculated as the contiguous region to the central line (see Section 4.5.1) of D.

4.4 Narrow band filtering by physical area

Zones of the scene being strongly oblique with respect to camera image plane will be sparsely sampled with 3D points, and will not be taken into account when constructing narrow bands. Consequently, the considered narrow band points are relatively parallel to the image plane axes, and the hypotheses in Appendix A.1 are valid.

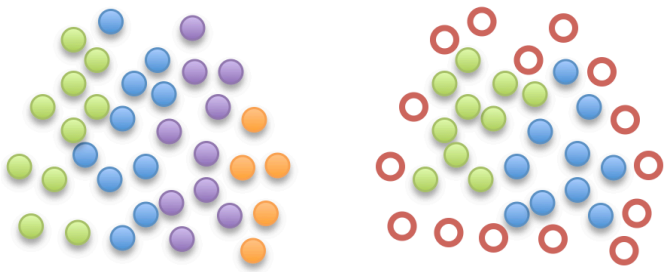

FIGURA 4.4: On the left, a narrow band filtering with a standalone A^{max} restriction, which results in a $N^{max} = 10$ maximal size. Note the residual region of 5 points (in orange). On the right, the filtering step with the additional restriction on connectivity with $\eta_L = 5$. The non-filled points have been filtered since they are not (η_L, δ_L)-connected.

Narrow bands cover the visible and connected parts of the scene surface. However, a given band may be composed of points from both arms, since they contain points at similar $dist_G$ (Equation (4.4)). In order to separate points of a same band that belong to different contexts, narrow bands are filtered depending on their physical area. Indeed, a maximal area A^{max} is set, so that any narrow band b with a physical area A^b larger than A^{max} is divided into a maximum of α regions, as shown in Equation (4.5).

$$N_{regions} \leqslant \alpha = \lceil \frac{A^b}{A^{max}} \rceil \tag{4.5}$$

Let $N(b)$ be the number of points in b and $\bar{\mathbf{z}}_b$ its mean depth. By applying the approximation described in Appendix A.1, there exists a maximum number of points $N^{max}(\mathbf{z}_b)$ which keeps α constant at every depth level $\mathbf{z}_b$ (Equation (4.6)). Therefore, if $\bar{\mathbf{z}}_b$ varies (*i.e.* a person moving towards or away from the camera), narrow bands will still be divided into no more than α regions.

$$N^{max}(\bar{\mathbf{z}}_b) = \frac{A^{max}}{\Gamma^C(\bar{\mathbf{z}}_b)} \tag{4.6}$$

Remark that a standalone A^{max} restriction could result in very small residual regions as shown in Figure 4.4. Therefore, besides the maximal area condition given by A^{max}, some additional restrictions must be verified in the narrow band filtering step. More precisely, the filtered regions must be (η_L, δ_L)-connected regions themselves (which implicitly forces a minimal region size of η_L points). After the filtering step, a set of regions is obtained for every narrow band, all of them being (η_L, δ_L)-connected. Some examples of the filtering step are presented in Figure 4.5.

FIGURA 4.5: Three examples of narrow band filtering. The obtained regions are randomly painted. In this case, an $A^{max} = 70\,cm^2$ has been applied together with a $(\eta_L = 30, \delta_L = 4\,cm)$-connectivity.

4.5 Detecting end-effectors in a topologically weighted graph

End-effectors are topologically prominent protuberances in the object under study, restricted to the viewpoint of the range camera. A method to detect end-effectors from the result of the R-NBLS method is discussed in this Section (see Figure 4.1).

4.5.1 Graph root

The proposed R-NBLS method requires the specification of an initial zero level set $L^0_{t_0=0}$ as starting point. Such origin region, called graph root, may be a single 3D point or a set of points. The definition of the graph root will strongly depend on the application. In this chapter, the cases of a whole human body and a human hand are studied, with their specific graph roots.

The human body case For this specific case, we propose to use a straight line as graph root (blue line in Figure 4.1). Such line connects the centroid $\mathbf{z}_C$ of D with $\mathbf{z}_M$, the latter being the midpoint between $\mathbf{z}_C$ and the head position $\mathbf{z}_H$, which is obtained with [SRHC12b]. Those points placed at a given distance δ_0 of l_C are labeled as initial zero level set $L^0_{t_0}$, from which the R-NBLS propagation can start. Despite head estimation, which exploits some temporal information to increase tracking robustness [SRHC12b], the rest of the proposed algorithm is frame-wise, without any temporal dependency.

4.5.2 End-effector graph construction

In general, R-NBLS filtered regions belong to prominent parts of the analyzed object (*i.e.* arms, legs) due to the band splitting considering connectivity (Section 4.4). Our objective being that of finding end-effectors, it seems reasonable to use these context-wise regions.

In a consecutive phase to the narrow band filtering (Section 4.4), a graph is constructed on the filtered regions. The centroid of every region is taken as a graph node, with an associated creation time t_k coming from the R-NBLS propagation step explained in Section 4.3.

Graph nodes are linked in pairs with graph edges. We propose to only include those edges which link a source node n_i with time t_i and a sink node n_j with time t_j such that $i < j$, resulting in a directed graph (from inner to outer narrow bands). This way, any path constructed on the graph will be consistent with the node creation instants, not linking nodes with previously created ones.

A distance weight $w_{i,j}$ is calculated for every edge linking nodes n_i and n_j, with an additional distance penalty depending on the time elapsed between the creation of the nodes. Such penalty limits the construction of graph paths with strong jumps in creation time, this effect happening only in strictly necessary occasions. A 10% gain is added to the penalty $\alpha = 1.1$, so that it penalizes slightly more than an integer number of jumps. The proposed node weights are calculated with Equation (4.7).

$$w_{i,j} = \underbrace{|n_i - n_j|}_{distance} + \underbrace{(j - i - 1) \cdot \delta_L \cdot \alpha}_{penalty} \tag{4.7}$$

4.5.3 End-effector Estimation: Shortest Path from Farthest Level

A Dijkstra shortest path algorithm is run on the graph constructed in Section 4.5.2. End-effectors are extracted as the shortest paths from the farthest nodes to the graph root. Paths are searched starting at the node with greater t_k and ending at $\mathbf{z}_M$ (arms) or $\mathbf{z}_C$ (legs). If it does not exist any path from that node, successive nodes are taken as path sources by decreasing t_k until all paths have been found. Indeed, two paths ending at $\mathbf{z}_M$ are searched and labeled as arms, and two other paths are searched to end at $\mathbf{z}_C$, which are taken as legs. End-effectors are often referred as extremities when working with the human body. Some conditions should be verified in order to accept a path as and end-effector of a human body.

- A path must have at least 3 segments, avoiding too short noisy detections.

- For a given graph, those nodes which belong to an already accepted path become inaccessible for further path estimations.
- Those nodes at a geodesic distance smaller than $30\,cm$ from the central line l_C are not taken into account, since we are looking for human extremities. Such restriction limits the detection of extremities starting close to the body centroid. We assume this draw-back of the algorithm, as shown in Figure 5.5.

When both arms and both legs have been found, path search stops. The result computed by the presented algorithm constitutes the end-effector positions along with a skeletal-like structure describing the limbs. It should be noted that some poses may result in undetectable extremities, since their topological prominence is not clear enough. Therefore, only those end-effectors which are sufficiently detached from the body will be detected. Figure 4.1 presents a summary of the proposed algorithm, containing from left to right: raw depth estimation, R-NBLS propagation with $\eta_L = 80$ and $\delta_L = 4\,cm$, narrow band filtering with $A^{max} = 50\,cm^2$, graph nodes, and the obtained end-effector estimation on the right of the figure. The head has been found as in [SRHC12b]. Nevertheless, note that it could be detected as an extra path from $\mathbf{z}_M$.

4.5.4 Right and Left extremity decision

Taking advantage of the extremity graph, a decision whether a limb corresponds to the right or left hand is taken. Remark that no temporal cues are involved in the decision.

For hands, the direction of first segment (t_0 to t_1) of every graph path (g_A and g_B) is calculated, obtaining two vectors $\mathbf{f}_A$ and $\mathbf{f}_B$. A simple decision depending on the orientation of $\mathbf{f}_A$ and $\mathbf{f}_B$ is performed, taking as right hand the path with $\mathbf{f}_i$ more oriented to the horizontal axis to the right (positive X coordinates). Remark that using the first graph segment is strongly invariant to the position of the end-effector associated to the graph path. A similar reasoning is done for feet, using their two graph paths.

Yet being a basic classification approach, experimental results show that the proposed decision framework is effective.

5

Experimental Results

5.1 Setup

The experiments are obtained with a Kinect sensor. The color information is discarded and only the depth estimation is exploited in the experiments below. Firstly, we provide experimental results of the proposed stand-alone methods, analyzing their parametrization and discussing the advantages and drawbacks observed during the evaluation. Such experiments are carried out on a set of self-recorded sequences *seqUPC*, containing various movements and body poses. In Section 5.4, the proposed methods are compared with two reference methods [SFC+11, GPKT10a] on the dataset provided by the latter.

We have uploaded videos of the experimental results in https://www.dropbox.com/sh/f6tls37lb9rezex/qRsTU5sjTh.

5.2 ORD stand-alone results

We consider the Kinect's original resolution of $640 \times 480\,px$, and also a down-sampled version by a factor $N = 4$ ($160 \times 120\,px$ respectively).

A frame-rate of 9 fps is achieved with the $N = 4$ solution, which is considered real-time. As expected, the full resolution version executes much slower, at about 0.05 fps.

The results presented hereafter correspond to the real-time version ($N = 4$) with a parametrization $\xi = \{\rho, K\} = \{15\,cm,\ 8\}$.

FIGURA 5.1: Compilation of different poses in the experiment sequence. The proposed algorithm performs properly in such challenging situations. Poses partially out of frame (farther left and right) are overcome. Remark that extremities are not detected when they are not prominent enough or when they are occluded (from left to right: 2nd, 7th and 8th poses).

We consider the manual annotations of *seqUPC* used in Section 5.3. The *seqUPC* sequence contains challenging poses, as well as some frames with partial body capture (person slightly out of frame). A compilation with some poses may be consulted in Figure 5.1.

An end-effector is considered properly detected when the distance to the ground-truth is smaller than 30 cm, as proposed in [GPKT10b].

The obtained *head* detection rate is of 97.7%, with an average error of 2.7 cm. As far as hands are concerned, none, one or two hands may appear marked in the ground-truth sequence. No distinction is considered between right and left hand, but between first (or only) hand and second hand (detection order). The first hand is detected in 90.31% of the cases, with an average error of 8.15 cm; while the second hand detection rate is of 76.31% with an average error of 9.2 cm. If an end-effector is detected as *hand*, and does not exist as ground-truth, a false positive is counted. About 8% of the overall *hand* detections are false positives in our experiments.

The detection rate of the first *foot* is of 86% with an average error of 11.2 cm. The second *foot* is detected in 71.6% of the cases, with an average error of 13.1 cm. The number of false positives is about 8.2% of the detections.

A confusion matrix is presented in Figure 5.2, built after the detected end-effectors, to give an overview of the precision of the classification. The values in brackets correspond to the full resolution version. Feet are the best-classified end-effectors, with only 1.07% of the detections being labeled as hands. The head is also well detected, being confused

	head	*hand*	*foot*
head	98.9% (95.8%)	1.12% (4.20%)	0% (0%)
hand	3.2% (5.1%)	96.3% (93.3%)	0.52% (1.61%)
foot	0%	1.07% (0%)	98.9% (100%)

FIGURA 5.2: Classification confusion matrix for the ORD real-time version, and the ORD full resolution version (in brackets).

with hands about 1.12% of the times. Cross-confusion between feet and head are not observed in our experiments. Hands are more often confused, even if achieving a high classification percentage of 96.3%, they are confused with the head (3.2%) and less often with feet (0.52%).

Slightly worse percentages are obtained with the full resolution version, due to the use of the $N = 4$ statistical moments in the experiment. However, the confusion matrix is still acceptable in both cases. The average errors of the full resolution version are $2.58\,cm$ (head), $4.13\,cm$ (first hand), $4.53\,cm$ (*second hand*), $5.81\,cm$ (first foot), $6.90\,cm$ (second foot). Therefore, the main advantage of the full resolution is a better average precision, which is about twice better for hands and feet.

The experiments show that the ORD approach performs better than the R-NBLS approach on the *seqUPC* dataset. Indeed, the ORD approach has been specifically trained with a subsequence of *seqUPC* (the statistical moments of the $\{Y, S, A\}$ descriptors). Therefore, it is expectable to obtain such better results with ORD than with the R-NBLS approach, which does not have a dedicated parametrization.

5.3 R-NBLS stand-alone results

5.3.1 Effect of the parametrization on the human pose estimation

Some aspects related to the tuning of the parameters of the proposed algorithm are shown in this section. We examine the parameters δ_L and A^{max}, since they are the most influential ones. η_L is a filtering parameter which may be kept invariant for a given sensor). For the Kinect camera, a value of 0.5 points per cm^2 during propagation has proven to be adequate. Therefore, the value of η_L is tightly related to the δ_L parameter ($\eta_L = 0.5 \cdot \pi \cdot {\delta_L}^2$).

The narrow band maximal width, δ_L, determines the resolution of the propagation, and also the areas which will be covered. A too low value of δ_L leads to propagation cuts,

the advancing contour being too poor (few or no points) at some topologically narrow zones. On the other hand, a high δ_L value affects precision and extremities are less frequently detected (greater topological prominence is needed). Figure 5.3.a shows two R-NBLS propagations with $\delta_L = 4\,cm$ and $\delta_L = 10\,cm$. Both arms are close to the body and their topological prominence cannot be detected with a large δ_L.

The maximal area A^{max} controls the population of the end-effector graph. The smaller A^{max}, the more nodes are included in the graph, allowing more freedom for finding plausible paths. However, extremity detection is less stable with very low A^{max} values. On the contrary, if A^{max} is increased, the graph is poorly populated, resulting in a very rigid extremity estimation. In Figure 5.3.b, three examples are shown to illustrate the effect of various A^{max} values on the extremity detection.

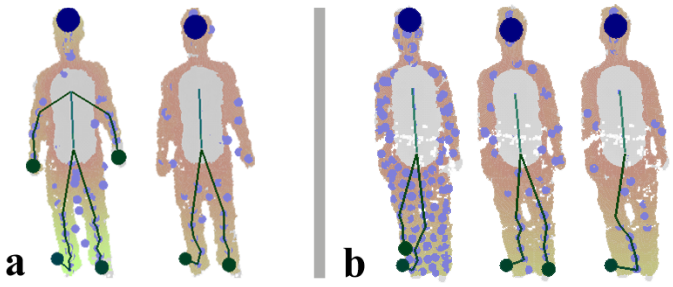

FIGURA 5.3: (a) Low values of δ_L provide more precision for the detection of topological prominence, even if the resulting paths are less straight (noticeable at the legs). (b) The maximal area parameter (left to right, $A^{max} = 20\,cm^2, 70\,cm^2$ and $200\,cm^2$) determines the population of the end-effector graph. Values of A^{max} which allow a proper detection of close legs are considered as trade-off values. Indeed, the narrow bands covering the legs will split into two regions, allowing the computation of two paths to $\mathbf{x}_C$. In the example, $20\,cm^2$ is too low and $200\,cm^2$ too high, while $70\,cm^2$ seems to be a convenient trade-off.

5.3.2 Effect of the parametrization on the detection error and detection rate

In order to evaluate the proposed method, more than 1000 depth frames of *seqUPC* have been manually marked (hands and feet) as ground-truth. Only those extremities noticeable to the naked eye on the depth images are marked, avoiding guessing limbs' position from the input data.

Results after various parametrizations are summarized in Figure 5.4. The error is measured as the Euclidean distance between the 3D ground-truth points and the estimated ones. In order to include the variance of the estimation, we plot $\bar{\varepsilon}$ and σ_ε on the horizontal axis, where $\bar{\varepsilon}$ is the average error and σ_ε the standard deviation. On the vertical axis, the percentage of detections over the number of ground-truth detections is plotted, taking into account only those detections with an error smaller than $30\,cm$. Therefore,

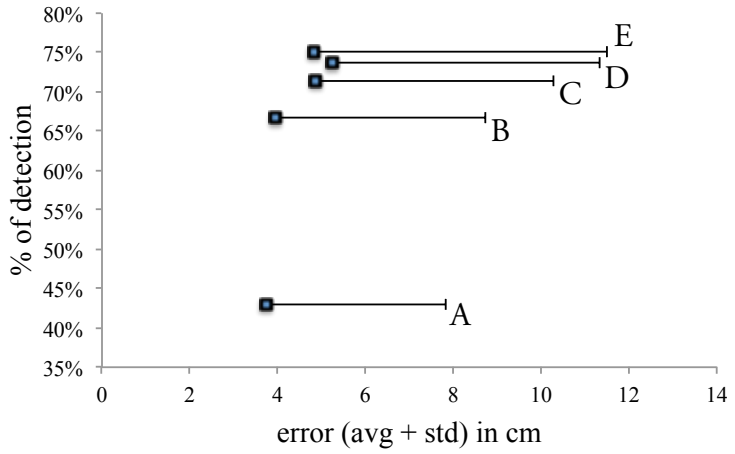

FIGURA 5.4: Detection error vs. detection rate (%) for various parametrizations (δ_L, A^{max}). $A = (10\,cm, 120\,cm^2)$, $B = (8\,cm, 70\,cm^2)$, $C = (6\,cm, 60\,cm^2)$, $D = (5\,cm, 50\,cm^2)$ and $E = (4\,cm, 40\,cm^2)$. Parametrization B seems to provide the best trade-off, with an average error of about $3.94\,cm$ and a standard deviation of $4.79\,cm$ for a detection rate of about 70%.

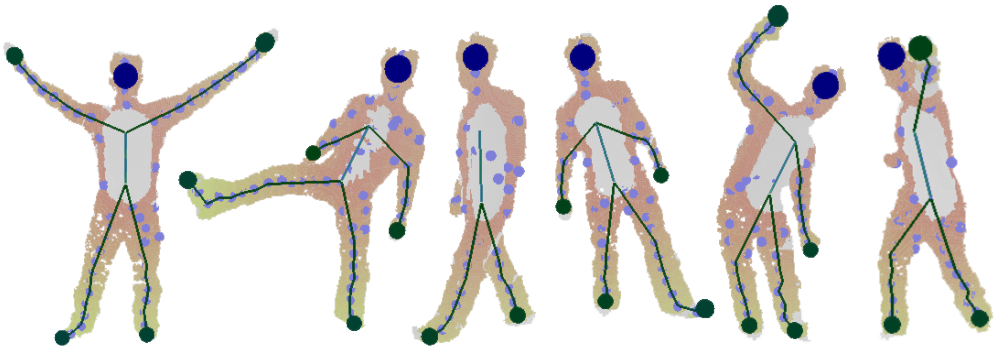

FIGURA 5.5: Some examples of the proposed human pose estimation algorithm. Remark that we do not perform any temporal tracking of the extremities, estimating human pose independently at each frame. One may notice how the strategy copes well with some adverse situations (*i.e.* . punching or bending the body). On the other hand, not prominent enough extremities are not detected with the proposed algorithm (*i.e.* third from the left, walking man).

parametrizations with higher detection rate and lower $(\bar{\varepsilon} + \sigma_\varepsilon)$ will provide the best results.

Experimental results show that parametrization $B = (\delta_L = 8\,cm, A^{max} = 70\,cm^2)$ obtains the best results with the Kinect camera. More precisely, it achieves extremity detection with an error smaller than $30\,cm$ (maximal size of a foot or hand) in about 77% of the over one thousand annotated frames. The average error is $\bar{\varepsilon} = 3.94\,cm$ and its standard deviation $\sigma_\varepsilon = 4.79\,cm$. The percentage of detection increases while the error does so. Such effect shows the trade-off between obtaining many poor detections or less precise detections.

A summary of different situations has been presented in Figure 5.5 to show how the proposed human pose estimation algorithm performs with various human poses. Both easy situations (cross pose, farther left) and more difficult ones (punching, farther right) are properly solved, providing a 3D estimation of the position of the extremities. When extremities are not topologically prominent (*i.e.* third pose from the left), they are not detected. This is a logical draw-back of the proposed method. Remark that these estimations are obtained without any temporal tracking of the extremities, providing a frame-wise solution to the human pose estimation problem.

5.3.3 Generalization to other objects

The proposed algorithm to detect end-effectors may be applied to other objects besides the already presented human body case. With a suitable zero level set $L^0_{t_0}$ and an adapted parametrization (δ_L, A^{max}), the prominent end-effectors of any object may be found.

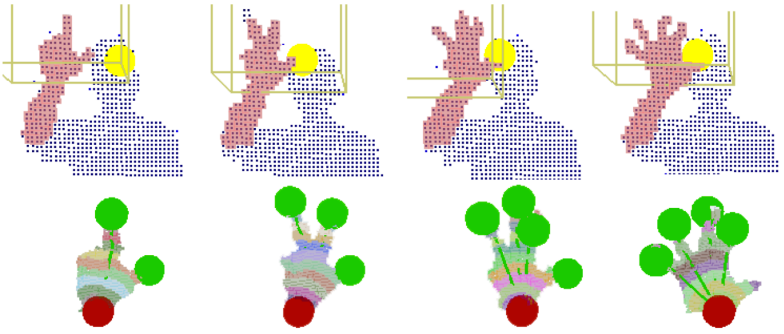

FIGURA 5.6: Obtention of end-effectors from a generalized object: example of finger detection.

In [SRHC12b], a fast and robust algorithm for head and hand detection is proposed. We utilize the obtained hand positions, onto which we apply the proposed R-NBLS end-effector detector, aiming to detect the number of extended fingers.

Some finger detection examples are shown in Figure 5.6, where one, two, three and four fingers are detected. In this example, the person is located about 2.5 meters far from the range camera. The initial zero level set $L^0_{t_0}$ is set as the centroid of the hand blob (graph root), and the parametrization $(\delta_L, A^{max}) = (2\,cm, 7\,cm^2)$.

5.4 Stanford'10 Dataset

Two reference methods are used to evaluate the proposed method. In [SFC+11], Shotton *et al.* propose a body part classification by means of a Random Forest strategy. Ganapathi *et al.* propose in [GPKT10a] a model-based approach exploiting temporal consistency. They also provide a dataset consisting of 27 sequences of increasing difficulty, recorded with a Time-of-Flight (TOF) camera. Moreover, ground-truth positions obtained with a motion capture device are provided. The experiments presented hereafter are obtained on the Stanford'10 Dataset [GPKT10a].

5.4.1 Classification Precision

The proposed methods detect head, hands and feet of a human body. Therefore, we select the subset of markers in the Stanford'10 Dataset that represent these body parts. In Figure 5.7, a summary of the obtained average precision (AP) is provided. The R-NBLS method outperforms [GPKT10a], only being slightly surpassed in the cases of the head and right foot. The method in [SFC+11] obtains slightly better results in average. However, it is a specific classification method, whilst the other two methods are focused on detection, making no classification effort. Regarding the ORD method, the classification results are poorer. As mentioned before, this method has been trained with data from the *seqUPC* sequence (different person, camera and viewpoint than in [GPKT10a]). Given the simplicity of the method and the specific training, the results on [GPKT10a] are acceptable.

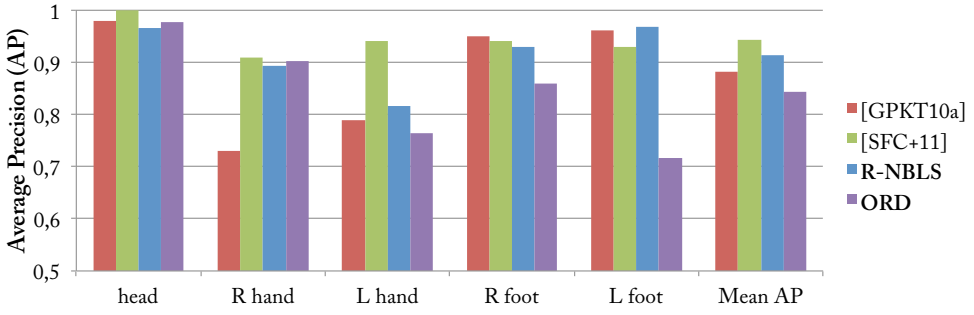

FIGURA 5.7: Average precision comparison with [SFC+11, GPKT10a] over the 27 sequences provided by [GPKT10a].

Regarding the average detection error, we compute the 3D error between the obtained end-effectors and the selected ground-truth markers. Results are presented in Figure 5.8, comparing the R-NBLS results to the results obtained by [GPKT10a]. The proposed method behaves in a similar manner over the whole dataset, obtaining an average error of about 9 cm even in the most challenging sequences (24-27). The method in [GPKT10a],

obtains a slightly better detection error in the first sequences, strongly degrading its results when facing the challenging sequences.

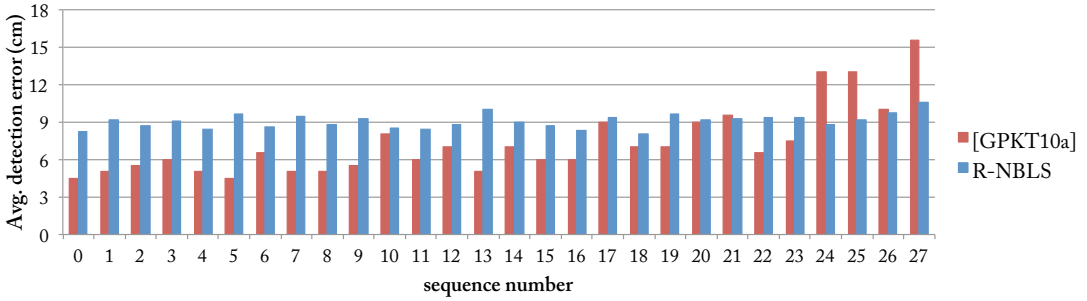

FIGURA 5.8: R-NBLS Detection 3D error comparison with [GPKT10a] over the 27 sequences provided by the Stanford'10 Dataset [GPKT10a].

5.5 Computational Performance

As far as processing speed is concerned, the R-NBLS method executes at about 57 fps using the dataset in [GPKT10a] (176×144 resolution). In this chapter, we have used a single core of an Intel Xeon CPU at 3GHz. The work in [GPKT10a] achieves a frame rate of about $4-10\,fps$ with a specific GPU implementation. The method in [SFC+11] claims a 50 fps execution frame-rate on full Kinect images ($640 \times 480\,px$) using an 8-core desktop CPU, and 200 fps using a dedicated powerful GPU. In [BMB+11], a frame-rate of 60 fps is achieved using a commercial CPU and a similar resolution.

The ORD proposal compares to the work in [GPKT10a], since their image resolution (SR4000 TOF camera, $176 \times 144\,px$) and objective (detection of human end-effectors) are similar as ours. Our proposal executes at 9 fps on a regular CPU core and at about 400 fps using the GPU implementation of ORD explained in Section 3.3.2. Since the proposal in [GPKT10a] achieves a frame-rate of $4-10\,fps$ on GPU, the ORD strategy performs much faster.

5.6 Conclusions

We propose the Oriented Radial Distribution descriptor for depth data. Its effectiveness at detecting end-effectors is shown, with special emphasis on the human body case. A fast classification strategy which exploits the statistics of local and global descriptors of these blobs, is also proposed.

Further usage of ORD is detailed in Chapters 8 and 9, where we exploit the multi-resolution characteristic of ORD for human hand analysis purposes. In these chapters

we also discuss the ability of ORD to both globally characterize an object and locally detect its end-effectors.

We also propose an extension of the Narrow Band Level Set formulation, named R-NBLS, to implement the GDM. A density restriction is added to the formulation, in order to better respect the topological properties of the object under analysis. The obtained level set provides a fast method to calculate geodesic distances over the original depth data.

Taking the R-NBLS results, we propose to build skeletal-like structure using a shortest path algorithm. We show how such structure is effective at detecting extremities in a frame-wise manner. In the specific human body case, the skeleton is tightly related to human pose. We also provide a simple model to initialize the R-NBLS method in the case of human body.

The proposed method performs about 5× to 50× faster than [GPKT10a], even in the adverse case of comparing our CPU implementation to the GPU implementation in the reference work. Using a similar resolution, a frame-rate of 60 fps is achieved by [BMB+11], taking advantage of using a pre-computed training dataset.

R-NBLS outperforms the method in [GPKT10a] in classification precision, and achieves similar results in terms of detection error. The method in [SFC+11] obtains a slightly better classification precision, taking advantage of a large training dataset and a dedicated classification task.

The R-NBLS method may be generalized to other objects.

Part II

Hand Analysis

Overall introduction

Humans move in countless ways, we run, jump, gesticulate, hold objects and a large etcetera. Many of these activities may be studied using by focusing on specific body parts, and analyzing their behavior (Figure 5.9). For example, for a human observer, it is fairly easy to tell if someone is walking by just taking a look at the movement of feet over a small time period of time.

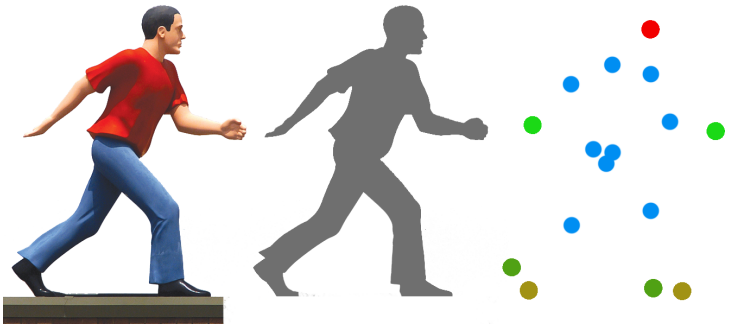

FIGURA 5.9: Human activities are still understandable by analyzing the silhouette, and even just some body parts. When observing motion, body parts are astonishingly representative of the activity.

Human activity can be represented as a compact set of body parts and their properties (position, trajectory, hierarchy, ...), which enables the usage of further computer vision methods to study human behavior. Therefore, some works have re-studied the problem of detecting body parts from the depth data point of view [SGF+12, GSK+11, BMB+11, GPKT10a] with impressive results. Some of these works estimate the position and label of different body parts based on a pre-processed dataset (discriminative approaches), whilst others fit a human body model onto the data in order to obtain body parts (generative approaches).

Amongst all body parts, hands occupy a top position in an expressivity ranking. We humans use hands to communicate, to express our emotions or to manipulate objects. Studying human hands from a computer vision point of view may enable novel human-machine interfaces, as well as providing intelligent communication systems for deaf people amongst other applications.

However, in order to apply hand recognition methods, we must know where hands are. Therefore, hand detectors are mandatory for the hand recognition task. Having a non-intrusive method to detect hand as a tool for further research on hand analysis is important. To be considered non-intrusive, such tool should be fast and lightweight (so that does not steal CPU power and memory to other methods), robust (to enable experiments over different datasets) and easy-to-setup.

Beyond hand detection

Recent successful interfaces like Apple's Trackpad [App12] or multi-touch devices, allow interaction by combining simple movements with finger configurations. However, device-based interaction is always limited, since the user must be *touching* the device.

Inspired by the way we interact with currently available multi-touch devices, we propose a touch-less interactive paradigm where hand gestures and fingertip configurations are combined with simple movements. In order to provide a similar usability, a precise and real-time detection of fingertip locations is required.

Detection of fingers or hand gestures is a complex task, given the high number of degrees-of-freedom of a hand, the usual presence of self-occlusions and the large amount of possible gestures. Few works have achieved performant fingertip detection results using Kinect [HMB11, MIT11, KKKA11, OKA11], mostly due to resolution problems and noisy depth estimations around fingers.

In addition, given the sudden appearance of the Kinect sensor, there is a lack of datasets using depth data for hand analysis, which makes the usage of discriminative methods more complicated.

In this part we firstly propose a baseline method called *HandBox* in Chapter 6. This approach is based on a method that detects human heads in a robust manner. Hands are consequently detected in a workspace zone attached to the obtained head estimate. The HandBox method is used as a low-level tool to localize hands in a depth data video sequence, given its robustness and fast execution. We show that the HandBox approach, yet simple in its concept, is highly effective in many real-time applications.

Regarding detailed hand analysis, we contribute with *ColorTip*, a public dataset for hand gesture recognition and fingertip localization on depth data. Details on the footage and obtention of the ground truth data are also included in Chapter 7.

Also, in Chapters 8 and 9 we propose two discriminative approaches for the problem of hand gesture recognition and fingertip localization (estimate *where* fingertips are and *which* finger is each):

- In the first approach, we propose a hand gesture classification method based on the ORD descriptor (Chapter 3). We show that ORD provides an effective representation for hand gesture recognition that helps reducing the search space for further fingertip localization. We make use of ColorTip as training dataset for both gesture recognition and fingertip localization.

- In the second method, we propose a collaborative voting framework that casts votes from ORD anchor points to fingertip positions, using ColorTip as training data. We include the hand gesture as an auxiliary variable in the problem formulation, obtaining it jointly with the fingertip positions. We show how this approach may be generalized to the classification of any object's parts and auxiliary properties.

6

HandBox: A baseline method for hand detection

6.1 Introduction

Obtaining the position of hands is an indispensable step for hand analysis methods. Yet mandatory, it is usually not a simple task, due to self-occlusions, clutter, hand-shape variability and speed of movement.

Moreover, hand detection algorithms that are intended to be used together with hand analysis methods should be robust and fast. Robust because erroneous detections are critical since hand analysis methods usually rely on consistent hand positions over time. And fast because hand analysis methods are usually thought to be run in real time, enabling a fluid human-machine interaction. The computational bottleneck, if exists, should be caused by the hand analysis step and not by the hand detection one.

With this purpose, we propose a hand detection algorithm that fulfills the robustness and speed requirements. Moreover, the fact of having our own method helps adapting to variations of the posterior hand analysis methods, especially during the research steps, where flexibility is an important factor to obtain interesting results.

6.2 Related Work

In the literature, some works explicitly tackle the head tracking problem. Haker *et al.* [HBMB07] compute principal curvatures on depth data as features to estimate the position of the head. Bohme *et al.* [BHMB09] improve Haker's proposal. Nichau and Blanz [NB10] present an algorithm to detect the tip of the nose in depth data by comparing the extracted silhouette to an average head profile template. Malassiotis *et al.* [MS05] estimate head pose by fitting an ellipse to the depth data and detecting the nose tip, which helps determining head orientation.

Most of the recent works concerning hand detection are focused on human pose extraction, the hands being a collateral result, since they are part of the body. The work by Bevilacqua *et al.* [BSA06] is one of the first attempts to track multiple persons in a crowded scene with a TOF zenital camera. Knoop *et al.* [KVD06] fit a 3D model to depth data by means of an adapted Iterative Closest Point (ICP) method. This way, hands are directly obtained from the model. Grest *et al.* [GKK07] use silhouette edges to track human extremities in complicated background conditions. They propose to use a non-linear least squares estimation. The tracking algorithm proposed by Zhu *et al.* [ZDF08] includes temporal consistency over frames to estimate the pose of a constrained human model. Lehment *et al.* [LKAR10] propose a model-based annealing particle filter approach on data coming from a TOF camera. More recently, Plagemann *et al.* [PGKT10] present a fast method which localizes body parts on depth point clouds at about 15 frames per second. Ganapathi *et al.* [GPKT10a] extend the work in [PGKT10] and extract full body pose by filtering the depth point cloud, using the body parts locations. In their work, they provide the *Stanford* dataset, which is used for evaluation in Chapter 10.4.3. Shotton *et al.* [SFC+11] presented recently the pose recognition algorithm running in the Microsoft Kinect depth sensor, which exploits an enormous dataset to perform a pixel-wise classification into more than 30 different body parts. In their work, they also make use of the Stanford dataset for evaluation. Strongly inspired by the work in [SFC+11], a Kinect SDK has been released by [Pri11]. They provide complete body pose in real-time.

6.3 Robust Head Tracking

Many persons at different distances from the camera may coincide in the same scene. Such persons may enter, exit and freely move around the camera field-of-view. Therefore, some aspects have to be taken into account when undertaking the head tracking problem.

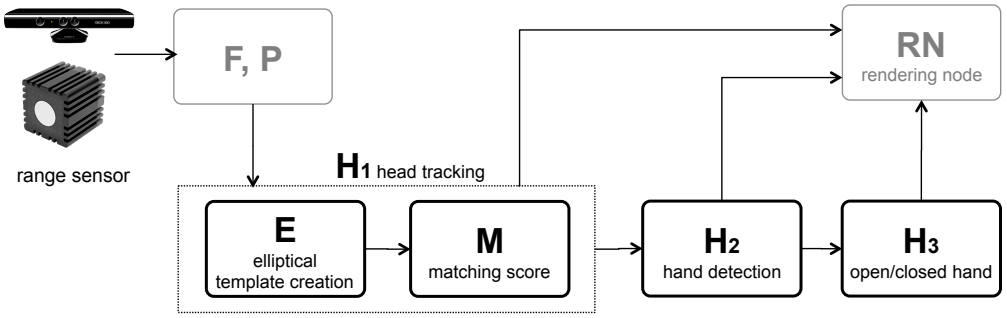

FIGURA 6.1: Summarized block diagram of the proposed head+hand tracking system, from the capture with a range camera to the final feed-back visualization on the rendering node (*i.e.* TV set). Both a Kinect sensor or a custom TOF camera have been used in this chapter, highlighting the flexibility of the proposed approach.

- **Occlusions**: Partial and total occlusions are a common problem of (single viewpoint) visual analysis systems, since moving objects may overlap.
- **Apparent head size**: User's heads may be placed at different depth levels, resulting in different head sizes when projected onto the image plane.

In order to overcome such problems, we propose to firstly estimate the size of the head on the depth image, step **(E)** in the scheme of Figure 6.1. Then, we estimate the head position **(M)**, composing the complete head tracking step $\mathbf{H_1}$.

6.3.1 (E) Head size estimation

Human heads may present many different sizes on the recorded images depending on the depth level where they are placed. Nevertheless, most of people's heads are likely to have an elliptical shape, no matter the distance they are away from the camera. Furthermore, the elliptical shape of heads is invariant to rotation of the human body around the vertical axis. Such invariant properties related to the elliptical shape of heads is exploited thanks to depth estimations from the range camera. We remark that hair could strongly change the shape of the head. However, the effect of hair is reduced, given its low reflectivity and high scattering of IR light.

Indeed, we assume that people will stand-up or be seated, laying down and other poses are not considered. Therefore, the depth of the whole body is likely to be similar to the depth of the head d_H.

An elliptical mask of the size of a regular adult head is placed at the calculated d_H depth level and projected onto the camera image plane, obtaining an ellipse of the apparent head size (H_x, H_y) in pixels (see Figure 6.5). Such ellipse, which is called *template* or $(\mathcal{E})$ is then used to find a head position estimate.

In order to better distinguish between the head ellipse and other elliptical shapes in the scene, some margins are added to $\mathcal{E}$. More precisely, the upper, right and left margins are extended with background pixels; while the lower margin is not modified, as shown in Figure 6.2.

FIGURA 6.2: On the right, a graphical example of the elliptical mask $\mathcal{E}$ used in the algorithm. On the left, a representation of a human head and its foreground mask $\mathcal{F}$, onto which a matching score is calculated.

6.3.2 (M) Head position estimation

With the aim of finding the image zone which better matches the elliptical shape, a matching score between the template ellipse ($\mathcal{E}$) and the global foreground mask ($\mathcal{F}$) is calculated at every pixel position (m, n) of the image. Such matching is performed within a rectangular search zone (see Section 6.3.3), by shifting the template ellipse across the image plane. The matching score is calculated according to conditions C^k presented in (6.1), where $bg = background$ and $fg = foreground$. Conditions are checked at every pixel position $(u, v) \in \mathcal{E}$ of the template, which is itself centered at (m, n). When a condition C^k is satisfied, $C^k = 1$, otherwise $C^k = 0$. The final matching score for the

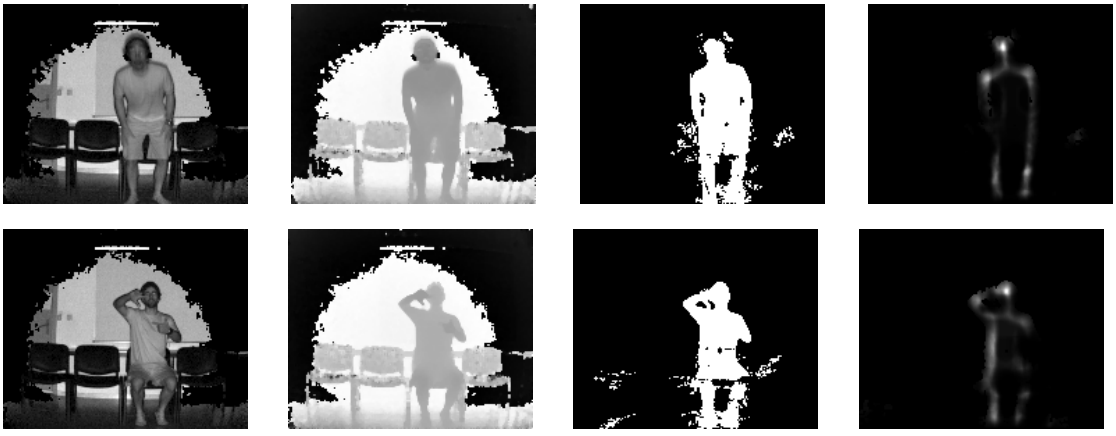

FIGURA 6.3: Head position estimation in two video frames (each row) obtained with a SR4000 TOF camera. From left column to right: IR amplitude, depth estimation, raw foreground mask and the obtained head matching score. The whitest zone is chosen as the most likely head position.

pixel (m, n) is calculated as the sum of all the scores obtained on the template pixels, as shown in Equation (6.1).

The pixel (m, n) with a higher score $\mathcal{M}^H = \max\{\mathcal{M}_{m,n}\}$ is selected, by simple max-pulling, as the best head position estimation $\hat{\mathbf{z}}_H$ in a given search zone. Conditions C^2 and C^3 provide robustness against partial occlusions. Indeed, the elliptical shape of the head would be very polluted by occlusions, since we are working on a foreground mask $\mathcal{F}$ which does not take depth into account. By means of conditions C^2 and C^3, depth is incorporated to the matching score, not taking into account those pixels which are not consistent with the calculated head depth d_H. Remark that a depth threshold d_{max} is used to decide whether a depth value is consistent or not.

$$
\begin{array}{lllllll}
C^1_{u,v}: & (\mathcal{E}_{u,v} = bg) & \wedge & (\mathcal{F}_{u,v} = bg) & & \\
C^2_{u,v}: & (\mathcal{E}_{u,v} = fg) & \wedge & (\mathcal{F}_{u,v} = fg) & \wedge & (|d_{u,v} - d_{H_i}| < d_{max}) \\
C^3_{u,v}: & (\mathcal{E}_{u,v} = bg) & \wedge & (\mathcal{F}_{u,v} = fg) & \wedge & (|d_{u,v} - d_{H_i}| > d_{max})
\end{array}
$$

$$
\mathcal{M}_{m,n} = \sum_{\forall (u,v) \in \mathcal{E}} (C^1_{u,v} + C^2_{u,v} + C^3_{u,v}) \tag{6.1}
$$

Two examples of the proposed solution are presented in Figure 6.3. It may be seen how, even with a noisy $\mathcal{F}$ mask, the algorithm manages to find the best estimate in the center of the head. Such inherent robustness is still increased with the search zone resizing step, presented in Section 6.3.3.

Furthermore, such approach is robust against horizontal head rotation and slight lateral head tilt, since the elliptical shape is still recognizable. As shown in Figure 6.4, the proposed algorithm succeeds when dealing with side and back views of the head (rotation along the vertical axis), as well as with long-haired heads, even if a slightly lower score is obtained in such case. Using either a SR4000 TOF camera or a Kinect camera has

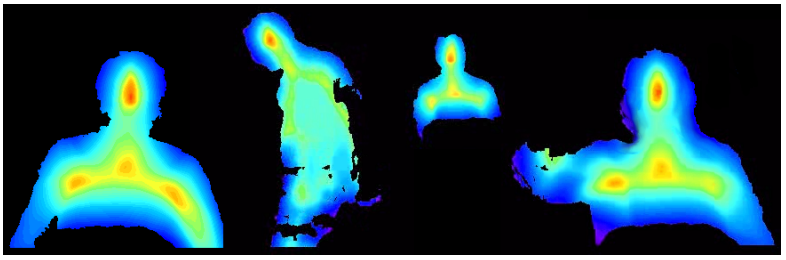

FIGURA 6.4: Head matching score in various situations obtained with a Kinect camera, including (from left to right): back view, side view (with slight head tilt), far person and long-haired person. In all these cases, the matching score presents a maximum in the head zone. Indeed, the user viewpoint does not strongly affect our algorithm, since the elliptical shape of heads does not substantially vary with vertical rotation.

insignificant impact on the proposed head estimation. Of course, the higher resolution provided by Kinect makes our algorithm slower. For real-time experiments, the Kinect

frames are down-sampled by 4, obtaining a $160 \times 120\,px$ images, which are similar in resolution to TOF images.

6.3.3 Search zone resizing

In order to increase the robustness of the presented head estimation algorithm we propose to resize the search zone where the matching score $\mathcal{M}_{m,n}$ is calculated. The fact of stretching the search zone to the previous estimate helps ignoring other possible elliptical shapes in the scene. For example, a second person entering the scene could lead to two similar maxima of $\mathcal{M}_{m,n}$. However, by limiting the search zone size, such second head is not taken into account in the matching score $\mathcal{M}_{m,n}$. A second thread could be run on the remaining pixels to find and track the second head, as shown in Figure 6.5.

Moreover, reducing the search zone drastically reduces the computational load of the algorithm, which processes the complete image only during the initialization frames.

The position and size of the head search zone is adapted to the head position variance, and also to the confidence on the estimation. More precisely, the new search zone size is calculated as a function of last matching score $\mathcal{M}^H$, the head size estimation (H_x, H_y), and the spatial variance of the estimation σ. The latter is calculated over the previous L frames, obtaining σ_x and σ_y for each axis. As for the matching score, it is normalized by the best achievable score for the current template $\mathcal{M}^{max}$ such that $\bar{\mathcal{M}} = \frac{\mathcal{M}^H}{\mathcal{M}^{max}} \in [0, 1]$.

The new rectangular search zone is centered at the last head position estimation, while the rectangle sides R_x and R_y are resized according to Equation (6.2).

$$\begin{aligned} R_x &= \sigma_x + (1+\mu) \cdot H_x \\ R_y &= \sigma_y + (1+\mu) \cdot H_y \end{aligned} \qquad with \qquad \mu = e^{\frac{-\bar{\mathcal{M}}}{1-\bar{\mathcal{M}}}} \tag{6.2}$$

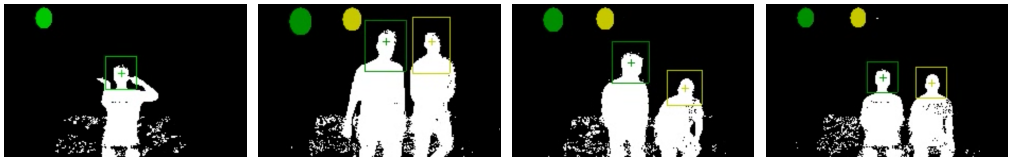

FIGURA 6.5: Head tracking snapshots from our experiments obtained with the SR4000 TOF camera. Head position is estimated by shape matching with an ellipse which is continuously resized depending on the distance between the camera and the person. Ellipses being currently used are presented at the upper-left corner of the image. The rectangles in the image correspond to the current search zones.

Such resizing is effective against fast head movements as the search zone is adapted to the variance of the estimations. For example, horizontal movements will enlarge the search zone along the horizontal axis.

Furthermore, including the matching score in Equation (6.2) makes the system robust against bad estimations, making the search zone slightly larger in case of bad matching. The objective is to include some more pixels to the matching score computation in case some better matches appear close to the previous processed zone.

Note that when the head estimation is stable ($\sigma \approx 0$) and confident ($\bar{\mathcal{M}} \approx 1$), the search zone is about the size of a human head $(R_x, R_y) \approx (H_x, H_y)$.

6.4 Hand Detection and Tracking

Hands are probably one of the most difficult parts of the body to track, given their mobility and size, but also one of the most important targets for many applications. Gesture recognition is tightly related to hand tracking, most of the information being obtained from hand movement. An accurate and robust hand tracking system is desirable to face complex gesture recognition, as well as to achieve interactive and immersive multimedia systems.

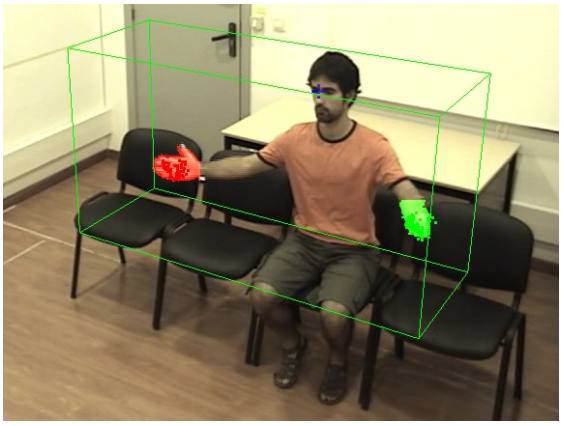

FIGURA 6.6: Snapshot of the proposed head+hand tracking system output. In this sequence, the head (blue cross) and both hands (green and red blobs) are being tracked. Movement is restricted to the HandBox (green box), which is attached to the estimated head position. Results are presented on a lateral view for visualization purposes.

The hand tracking system proposed in this section (block **H₂** in Figure 6.1) relies on the robust head estimation presented in Section 6.3. Hands are supposed to be *active* (performing gestures) in a zone placed in front of the body. Following this basic assumption, the HandBox $\Diamond$ is defined as a 3D box of size $\Diamond_x \times \Diamond_y \times \Diamond_z \, cm$, as shown in Figure 6.6. Hands are supposed to lay within it when moving. $\Diamond$ is attached to the head position $\hat{\mathbf{z}}_H$ so that $\Diamond$ follows the user's head at every time instant.

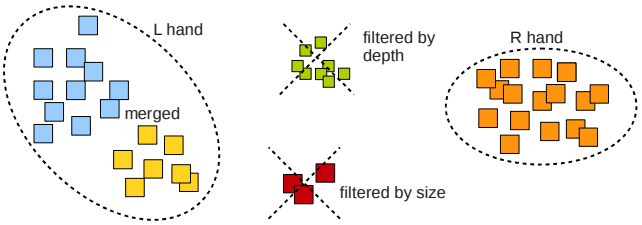

FIGURA 6.7: Example of cluster merging and filtering for hand detection. Color represents candidate clusters obtained with a kd-tree structure. The red cluster is too small. The blue and yellow clusters are merged since the Hausdorff distance between them is small enough. Three clusters remain, but the green one is filtered since it is placed farther in depth (represented with smaller squares). Thus, the remaining two clusters are labeled as R and L hands.

Hands are to be detected among the points $\{\mathbf{z}\} \in \Diamond$. Dense clusters are searched by means of a kd-tree structure, which allows fast neighbor queries among 3D point clouds [FBF77, MM99]. A list of candidate clusters is obtained and filtered according to the following sequential criteria:

1. **Merging**: Two clusters are merged as a single cluster if the Hausdorff distance between them is smaller than a given distance threshold δ_{min} (typically $\delta_{min} \approx 10\,cm$).

2. **Size filtering**: The resulting merged clusters are filtered by size (number of points in cluster), keeping the largest ones. A size threshold s_{min} (typically $s_{min} \approx 15\,cm^2$) is set to determine what clusters are accepted as hand candidates.

3. **Depth filtering**: Clusters that fulfill the previous criteria are sorted by depth. Those clusters placed closer to the camera are selected, knowing that a maximum of two clusters may be chosen.

Thresholds δ_{min} and s_{min} should be tuned depending on the type of camera and scene. A graphical illustration of these criteria is shown in Figure 6.7. The number of detected hands depends on the number of clusters that pass the merging and filtering steps, resulting in two, one or no hands being detected in $\Diamond$. For example, two hands are being detected in the example of Figure 6.6.

Following the proposed criteria, one could mislead the system by introducing, for example, an elbow in $\Diamond$. Such issue may be overcome by placing the $\Diamond$ box at a convenient distance of $\hat{\mathbf{z}}_H$. A distance of $25 - 30\,cm$ has proven to be robust enough in our experiments with non-trained users.

In addition, spatio-temporal coherence is included in the hand tracking scheme, which increases its robustness. Hand estimates at time $t+1$ are compared to those in time t

by means of Hausdorff distance measurements, in order to provide coherence to tracking and avoid right-left hand shifting. Furthermore, when only one hand is being detected, it is labeled (right / left) depending on the previous estimates and its relative position in $\Diamond$. For example, a cluster placed further right in $\Diamond$ probably corresponds to the right hand.

No significant differences have been appreciated between Kinect and TOF input data regarding hand detection.

7

ColorTip: A Dataset for Hand Analysis on Depth Data

7.1 Description

Despite the revolution that commercial depth cameras have brought, their recent irruption supposes a lack of public datasets. Ganapathi *et al.* [GPKT10b] provide a body pose estimation dataset using a Time-of-Flight (TOF) camera. Pugeault and Bowden [PB11] propose a hand gesture dataset using Kinect, which is intended for American Sign Language (ASL) purposes.

We propose ColorTip [Col], a public dataset for hand gesture recognition and fingertip localization captured with Kinect the sensor, which consists of a set of recordings and annotations with a two-fold objective. To provide a benchmark against which further research works may be assessed. But also, to enable novel interactive applications involving hand gesturing and fingertip localization.

In order to ease experimental setups, the ColorTip dataset is divided into folders according to:

- **Subject:** N subjects performing gestures like those shown in Fig. 7.1, ensuring intra user variability. Four of them are untrained users, which learned how to perform the gestures with a single and short explanation.
- **Challenge:** We consider that a given gesture may vary in orientation and translation. Therefore, raising 4 fingers is assumed as gesture number 4, but also moving these 4 fingers towards the camera, side views and hand rotations (Fig. 7.1). The amount of intra-gesture variability determines how challenging a given sequence is. The *Set A* sequences contain limited intra-gesture variation, which mainly consists in hand rotations on the vertical plane. On the other hand, the *Set B* sequences contain a higher intra-gesture variability, with free rotations and finger movement (as shown in Fig. 7.1).

In total, ColorTip contains a set of (7 subjects $\times$ 2 challenges) = 14 sequences of between 600 and 2000 frames each.

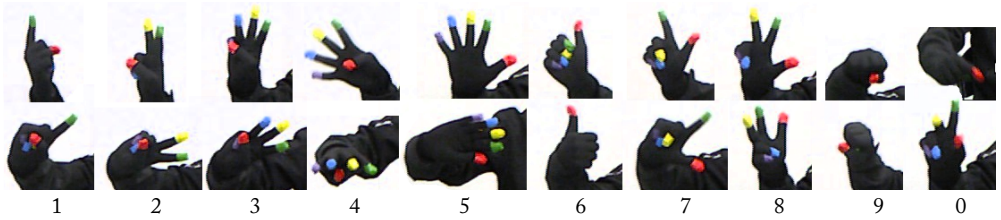

FIGURA 7.1: Sample of the annotated gestures in the ColorTip dataset. Two examples per gesture are shown (columns). These examples are extracted from a *Set B* sequence, with a high intra-gesture variation. Note the rotations and translations. Label 0 corresponds to *no gesture* (*i.e.* other gestures, transitions).

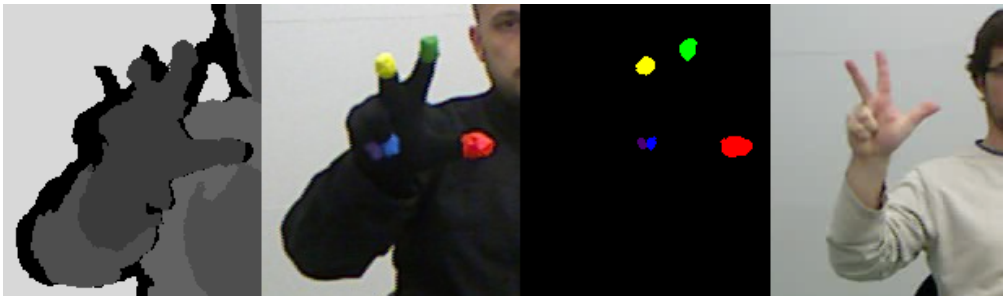

FIGURA 7.2: Snapshot of the ColorTip dataset content. From left to right: depth image, color image (remark the colored glove), segmented fingertips (colors are directly finger labels, and centroids are finger positions) and a similar gesture in a test sequence.

7.2 Annotations

Inspired by the work of Wang and Popović [WP09], a black glove with colored fingertips is used to capture the training sequences (see Fig. 7.2). In this way, we obtain a dataset

together with a fingertip annotation in a single footage without requiring expensive motion capture systems, like those used in [SFC+11, KKKA11]. Furthermore, one can easily record additional data to update the dataset. Actual fingertip locations are obtained by first segmenting the Kinect color images with a color-based Binary Partition Tree [SG00] (see Fig. 7.2) and then computing the region centroids. Color labels have an associated numerical label l.

Hand gestures are manually annotated among the 1-9 gestures, plus an extra label 0 for those frames with an unknown gesture. Also, a hand location annotation in image coordinates is provided.

8

NNGM: Nearest Neighbor + Graph Matching Approach for Fingertip Localization and Gesture Recognition

8.1 Introduction

The objective of the proposed method is to locate fingertips in real-time, that is, to know *where* fingers are placed, and also classify them to know *which* finger is each. Instead of facing the problem from raw data, as could be done with a similar approach to [SFC+11], we propose to use an intermediate step to restrict the search space. We exploit the statistical correlation between gestures and fingertip locations to perform such a restriction. This is very intuitive, since fingertip locations are conditioned by hand gestures, and at the same time allows a highly efficient fingertip inference.

We choose to use the hand gesture as a discriminative auxiliary variable in this intermediate step. Indeed, there exists a real necessity of detecting hand gestures, so we discard using other auxiliary variables without any semantic meaning. We remark that algorithmic decisions are strongly motivated by efficiency, since real-time is a strong objective for any interactive system.

In a second step, we infer the most probable fingertip locations conditioned on the obtained hand gesture. We propose a specific graph matching approach, which exploits fingertip structure, to undertake the fingertip localization task. Thus, both fingertip locations and hand gesture are obtained from the proposed overall scheme.

A novel usage of the Oriented Radial Distribution (ORD) descriptor (as explained in Chapter 3). The ORD descriptor characterizes a point cloud in such a way that its end-effectors are given a high ORD value, providing a high contrast between flat and extremal zones. Therefore, ORD is suitable to both globally characterize the structure of a hand gesture and to locally locate its end-effectors. Such ORD property nicely fits in the above mentioned two-step method. We propose to use the overall ORD structure for the gesture classification task, and to use local ORD extrema to feed the graph matching step. Therefore, a single ORD calculation is enough for both tasks.

The proposed method is evaluated with a recent 3D feature benchmark, revealing the convenience of using ORD. Furthermore, the gesture classification step is assessed with the ASL database provided by [PB11]. Fingertip localization results are successfully compared to a state-of-the-art Random Forest (RF) approach.

Summarizing, in this chapter we propose the following main contributions:

- A practical touch-less interaction concept, combining finger configurations, hand gesture and simple movements.
- A real-time method to obtain locations and labels, as well as hand gestures, using Kinect. We propose to exploit the statistical correlation between hand gestures and fingertip locations.
- A novel use of the Oriented Radial Distribution descriptor, exploiting its global structure for hand gesture characterization and its local values for fingertip detection.

8.2 Related Work

Many authors propose using depth cameras for human body analysis, ranging from full body pose estimation [SFC+11, BMB+11, GPKT10b, SMMN11] to hand gesture classification and fingertip localization.

Obtaining hand gestures with Nearest Neighbors (NN) classification has proven to be a promising approach when dealing with depth data [SPHK08, RYZ11, KPHB08]. However, most recent works use features that are not specifically designed for depth data.

Many authors have explored how to control a virtual environment with hands (*i.e.* PC desktop, 3D model). Such applications involve, in most of the cases, dynamic hand gesturing. In this direction, Soutschek *et al.* [SPHK08] propose a user interface for the navigation through 3D datasets using a Time-of-Flight (TOF) camera. They perform a polar crop of the hand over a distance threshold to the centroid, and a subsequent NN classification into five hand gestures. With a similar objective, Van den Berg and Van Gool [VV11] improve their work in [VKMBV09] by combining RGB and depth to construct classification vectors. Their alphabet consists of four gestures that enable selecting, rotating, panning and zooming of a 3D model on a screen. Hackenberg *et al.* [HMB11] estimate hand pose by identifying palm and finger candidates, after a pixel-wise classification into tips and pipes. The final hand structure is obtained with optical flow techniques. Ren *et al.* [RYZ11] segment the hand under some restrictive assumptions and adapt the Earth Movers Distance to a finger signature, finding the NN according to this metric. Malassiotis and Strintzis [MS08] extract PCA features from depth images of synthetic 3D hand models for training.

Other works have focused on finger-spelling using the American Sign Language (ASL). While still being an alphabet, the ASL contains 26 hand poses and their accurate classification becomes a challenging task. We remark that 24 of the 26 hand poses are static gestures and 2 of them are dynamic (involve trajectory). Most of the related works are focused on the static subset. Ceskin *et al.* [KKKA12a] take advantage of Randomized Decision Trees to classify hand shapes. Zhang *et al.* [ZYT13] recently propose a descriptor for depth data which encodes 3D facets into a histogram. They proof the suitability of this descriptor for hand gesture recognition on ASL datasets. Zhu and Wong[ZW12] propose to fuse common color and depth descriptors and use linear SVM's to predict the hand gesture. Kollorz *et al.* [KPHB08] obtain a fast NN classification using simple feature projection on two axis, which they apply to the first 12 letters of the ASL (static gestures). Uebersax *et al.* [UGVV11] perform an iterative hand segmentation by optimizing the center, orientation and size of the hand. They smartly aggregate three classifiers that take shape and orientation into account. Pugeault and Bowden [PB11] propose a multi-resolution Gabor filtering of the hand patch to train a Random Forest classifier. In their work, they provide a complete dataset of the 24 American Sign Language (ASL) static gestures captured with the Kinect sensor, with both color and depth information available. Their dataset contains patches roughly centered at the hand centroid.

Fewer works have tackled the fingertip localization problem. In [HMB11], fingertips are detected but not labeled, as well as in [MIT11] where also the palm and fingers orientation are estimated. Both approaches exploit geometrical features to detect fingertips on the hand point cloud. The body part classification approach proposed by Shotton *et al.* in [SFC+11] is applied to hand parts by Keskin *et al.* [KKKA11], obtaining full

hand poses at the expense of a costly training. Recently, Oikonomidis *et al.* [OKA11] formulate the hand pose recovery problem as an optimization approach, measuring the discrepancy between a model and the observed hand. Full hand pose is provided (including fingertips), requiring initialization at a known initial pose. On the other hand, their cost function relies on color information, reducing the performance to controlled scenarios.

8.3 Method Overview

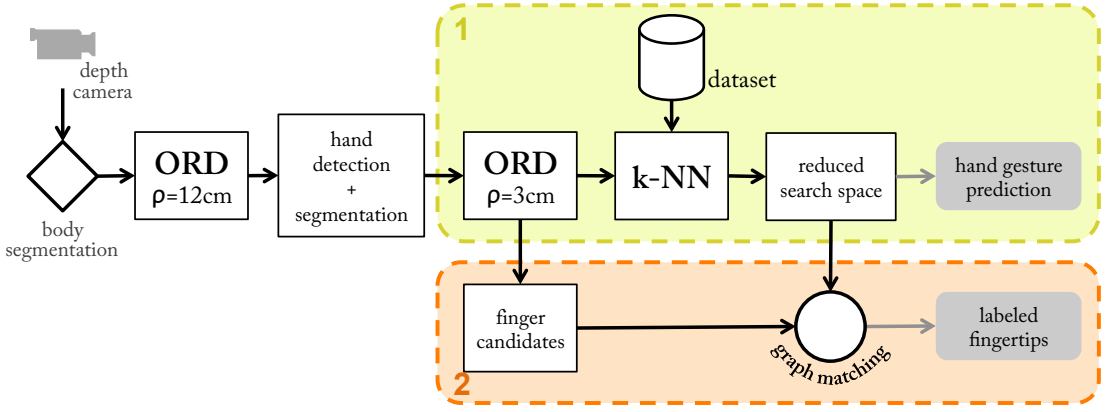

FIGURA 8.1: General scheme of the proposed NNGM (Nearest Neighbor + Graph Matching) method. Fingertip locations are obtained (2) through an intermediate step, where the hand gesture is obtained as auxiliary variable (1).

The scheme in Fig. 8.1 summarizes the main blocks involved in the proposed method. In a preliminary step, we perform a *body segmentation* by means of background subtraction with depth data. Then, we detect and segment the hand by using the ORD at *hand scale* (in this chapter $\rho = 12$ cm).

In case the application requires strict real-time, one may use fast methods to detect hands like that proposed in Chapter 6. However, a nice advantage of using ORD is that one may perform hand detection with a *quarter body* viewport, or even observing a single arm. In addition, ORD may be used to detect other body parts by tuning the scale.

ORD is very sensitive to depth gaps, so hands will still return high ORD values when placed few centimeters away from another body part. Of course, hands "touching" other parts (such as the head), will cause problems. At that precision level, both ORD and the depth camera resolution are pushed to the limit. We can say that hands placed at a distance $> \rho$ will be properly detected.

A two-step approach is proposed to perform hand gesture recognition and fingertip localization on the detected hand. We compute the ORD at *finger scale* ($\rho = 3$ cm) on a small patch containing the segmented hand, thus obtaining high ORD responses at fingertips and eventually at knuckles (see Fig. 8.2). The objective of this *finger scale* ORD is two-fold:

1. On the one hand, we use the ORD values to select the most likely hand poses (gestures) by computing distances between feature vectors, obtaining a subspace of likely hands from the ColorTip (Section 8.4). We note this step as *gesture recognition*.

2. On the other hand, higher ORD responses are used as sparse fingertip candidates, and serve us to infer fingertip locations (*fingertip localization*) conditioned on the previously selected subspace. A structured inference framework is proposed, formulated as a graph matching problem (Section 8.5).

The whole framework is based on the ORD feature data. We consider ORD a strong enough representation for this task, which, in addition, may be fast computed enabling real-time applications.

We introduce some notation hereafter, describing some of the variables handled in the proposed method. Training patches $\Pi = \{\pi_1, \ldots, \pi_i, \ldots, \pi_N\}$ are squared patches of different sizes containing depth data of the segmented hand. We start by computing the ORD(π_i) at *finger scale* on each training patch. Then, we re-sample into a regular grid of $m \times m$ blocks to characterize the training patches (Fig. 8.2). Each block gets the mean ORD value of the pixels inside it, obtaining a set of m^2-dimensional feature vectors $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$. Besides, let $\mathbf{r}_i \in \mathbb{R}^{2\times 5}$ denote the ground truth fingertip locations (in pixel coordinates) corresponding to the i-th training sample, being $\mathbf{r}_i[m] \in \mathbb{R}^2$ with $m = 1, \ldots, 5$ each fingertip location (used in Section 8.5). Additionally, let y_i be gesture labels. Then, training templates are defined as $\lambda_i = \{\mathbf{x}_i, \mathbf{r}_i, y_i\}$, and the complete training dataset $\mathcal{H}$.

Given a test patch π, the objective is to locate fingertip positions in it, that is $p(\mathbf{r}|\pi)$. We propose to break the problem of obtaining fingertips from data $p(\mathbf{r}|\pi)$ into two more tractable problems, that can be efficiently solved. In order to do so, we introduce the hand gesture y as an auxiliary variable. By doing so, the problem of inferring fingertip locations from data can be posed as:

$$p(\mathbf{r}|\pi) = \sum_y p(\mathbf{r}|y, \pi) \cdot p(y|\pi) \tag{8.1}$$

However, the marginalization of gestures implies a time consuming summation. Since real-time is a requirement, we approximate the problem in Equation (8.1) by firstly maximizing $p(y|\pi)$, obtaining the best candidate $\hat{\lambda} \in \mathcal{H}$. Secondly, we infer fingertip locations from the best template obtained after this maximization. The problem results as posed in Equation (8.2):

$$p(\mathbf{r}|\pi) \approx p(\mathbf{r}|\hat{\lambda}, \pi) \text{ with } \left\{ \hat{\lambda} \in \mathcal{H} \mid \hat{y} = \operatorname{argmax} y\{p(y|\pi)\} \right\} \tag{8.2}$$

We propose to solve the gesture recognition problem $p(y|\pi)$ using a k-Nearest Neighbors (k-NN) classifier. k-NN techniques are strongly sensitive to the data nature. Thus, an inappropriate feature selection could lead to a bad k-NN classification. Choosing a k-NN classifier helps to test the suitability of the ORD feature, as well as providing a fast classification taking advantage of a *kd-tree* [FBF77] structure.

We remark that the term *hand gesture* refers to specific hand poses with a given meaning. Such poses may be static (*i.e.* ASL dataset) or dynamic (*i.e.* ColorTip dataset). The posed gesture recognition problem is solved no matter the movement of the hand.

Concerning the fingertip localization problem $p(\mathbf{r}|y, \pi)$, we propose to solve it using a graph matching (GM) algorithm with a structure-based cost on edges. Such step is conditioned on the search space obtained from the $p(y|\pi)$ problem.

The overall proposal combining NN classification and GM is noted as NNGM.

8.4 Hand Gesture Recognition

In pattern recognition problems, the accuracy of a method ultimately depends on the distance metrics on the feature space, i.e., whether classes in the feature space appear separate enough to learn a robust classification rule. In this chapter, we propose a feature space based on the ORD descriptor. Feature vectors obtained by computing the ORD descriptor on the input data provide a representation of salient regions of the hand. In other words, ORD-feature vectors of hand poses can be seen as distribution of important parts of the hand and even interpreted as where the knuckles and fingers lay within the patch. For that reason, an ORD-based feature space is a suitable space for matching hand poses.

We choose to use a k-NN classifier for pose and gesture recognition. In this way, we show that even by simple matching techniques, the ORD feature space is adequate for hand analysis.

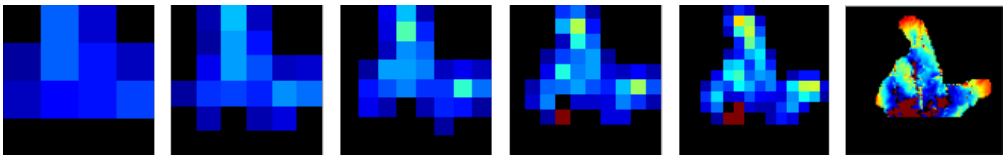

FIGURA 8.2: Examples of feature vectors at various m re-sampling values. From left to right, $m = \{4, 6, 8, 10, 14, \text{full ORD patch}\}$

To use a k-NN classifier on a large set of instances, we use a m^2-dimensional *kd-tree* [MD09] that efficiently organizes feature vectors, allowing fast NN queries. The L_2 norm is used in this chapter, since it offers a good trade-off between speed and performance.

For a test patch at a given time instant (we omit the temporal subscript t in this section for readability reasons) , the k-NN search returns a set of k training templates $\mathcal{H}^k = \{\lambda_1, \ldots, \lambda_j, \ldots \lambda_k\}$ with associated distances to the test patch $\delta : \mathcal{H}^k \mapsto \mathbb{R}$. Let $\Phi_y(\mathcal{H}^k)$ be the distribution of gestures obtained from $\mathcal{H}^k$.

We note $\hat{\lambda}$ the k-NN best match by majority, as specified in Eq. (8.3). The obtention of $\hat{\lambda}$ is conditioned on the gesture which maximizes $\Phi_y(\mathcal{H}^k)$. Therefore, maximizing $\Phi_y(\mathcal{H}^k)$ is equivalent to the maximization in (8.2). Remark that one may refer to 1-NN best match, which is the first nearest neighbor in the training dataset.

$$\hat{\lambda} = \lambda_j \in \mathcal{H}^k \quad | \quad j = \operatorname{argmin} j \Big\{ \delta(\lambda_j) \mid y_j = \operatorname{argmax} y\{\Phi_y(\mathcal{H}^k)\} \Big\} \tag{8.3}$$

The k-NN search may deliver false detections, resulting in a noisy gesture recognition. We propose hereafter to apply human dynamics restrictions to smooth the result of Eq. (8.3).

8.4.1 Dynamically Constrained k-NN

In many cases, we are subject to analyze video sequences, which intrinsically have a temporal consistency over consecutive frames. Hand dynamics are smooth, hence we assume that hand gestures are not instantly changing, but are maintained during a minimal number of frames.

In order to exploit such video consistency, we propose to keep a trace of the last Q predicted gestures $\hat{Y}_Q = \{\hat{y}_{t-Q}, \ldots, \hat{y}_{t-1}\}$, obtained from the gesture labels of $\mathcal{H}^Q = \{\hat{\lambda}_{t-Q}, \ldots, \hat{\lambda}_{t-1}\}$. Let $\tilde{y}_Q = \operatorname{argmax} y\{\Phi_y(\mathcal{H}^Q)\}$ be the statistical mode of the last gestures $\hat{Y}_Q$, and let $\hat{y}_t$ be the predicted gesture at time instant t, which is obtained as detailed in Equation (8.3).

In the cases where the predicted gesture differs from the statistical mode of the last Q gestures, we select the closest template among the k-NN set with a gesture equal to the mentioned mode. This way, transitions between gestures are smoothed, avoiding gesture *flickering*, as well as de-noising intra-gesture false detections.

In order to respond to gesture changes, when no gestures in the k-NN set equal the statistical mode, the 1-NN template is selected. The overall Dynamically-Constrained (DC) k-NN is detailed in Algorithm 1.

During the first frames (*i.e.* start of a live demo, begin of a video sequence, etc.) a simpler k-NN by majority is used, until the number of frames greater than Q. Thus, during these first Q frames, the performance of the gesture recognition part corresponds to that shown in Figure 10.7, with label *ORD*; and that of label *ORD-DC* after the first Q frames. By adopting this strategy, no gesture initialization is required.

Input:
$\mathcal{H}^k = \{\lambda_1, \ldots, \lambda_j, \ldots \lambda_k\} =$ k-NN set at time t
$\tilde{y}_Q =$ mode of the last predicted gestures $\hat{Y}_Q$
$\hat{y}_t =$ predicted gesture at t or $\operatorname{argmax} y\{f_y\}$
Output: $\hat{\lambda} =$ best k-NN

if $\hat{y}_t = \tilde{y}_Q$ **then**
    $\hat{\lambda} =$ k-NN by majority (Eq. (8.3))
else
    if $\exists\, y_j \in \hat{Y}_Q \mid y_j = \tilde{y}_Q$ **then**
        $\hat{\lambda} = \lambda_j \in \mathcal{H}^k \mid j = \operatorname{argmin}\{\delta(\lambda_j) \mid y_j = \tilde{y}_Q\}$
    else
        $\hat{\lambda} =$ 1-NN
    end if
end if

ALGORITHM 1: Dynamically Constrained k-NN search

8.5 Fingertip Localization

We address the problem of fingertip location by making use of the ORD descriptor in a structured inference framework. Maxima of the ORD of the input patch are likely to represent fingertip locations. However, as mentioned before, for some hand poses these maxima may correspond to other salient points of the hand. But, even if all the maxima correspond to finger locations, one should be able to classify which finger belongs to each maximum. Consequently, there is a need to exploit the global hand structure to overcome these issues.

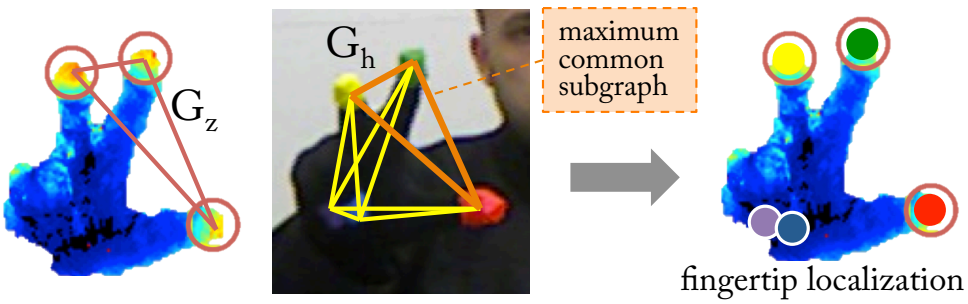

FIGURA 8.3: Fingertip localization scheme. Fingertip locations are inferred from the ground-truth graph G_h by computing the Maximum Common Subgraph with respect to the test graph G_z.

Fingertip localization on test patches takes advantage of the pose recognition scheme presented in Section 8.4. Let us recall that, in the training phase, we define templates $\lambda_i = \{\mathbf{x}_i, \mathbf{r}_i, y_i\}$ comprising the feature vectors, ground truth fingertip locations and gesture labels, respectively. Our method exploits the geometric structure of the ground truth fingertip locations $\hat{\mathbf{r}}$ of the best template match $\hat{\lambda}$ provided by the k-NN pose recognition block. The objective is to infer which ORD maxima of the test patch correspond to fingertip locations, and which are their finger classes. Let $G_h = (V_h, E_h)$ be a fully connected graph where vertices $v_h \in V_h$ correspond to the available fingertip coordinates in $\hat{\lambda}$, which we denote as $\mathbf{r}[v_h] \in \mathbb{R}^2$ (if a fingertip is not visible, such vertex is not considered). Let $G_z = (V_z, E_z)$ be the fully connected graph where vertices $v_z \in V_z$ correspond to the ORD maxima $\mathbf{s}$ of the test patch π_t, namely $\mathbf{s}[v_z] \in \mathbb{R}^2$. We obtain a correspondence between vertices in G_h and vertices in G_z by computing the *maximum common subgraph* [McG82] (Fig. 8.3). This process consists in obtaining the graph G with the maximum number of vertices such that there exist subgraph isomorphisms[1] from G to G_h and from G to G_z. Note that in general there exists more than one maximum common subgraph. From the set of maximum common subgraphs we choose the one that best satisfies a geometric constraint defined on its edges. Let us denote $G_{mcs} = (V', E')$ a graph from the set of maximum common subgraphs of G_h and G_z, which involves the mappings $f_h : V' \mapsto V_h$ and $f_z : V' \mapsto V_z$. Then, for each edge $(u, v) \in E'$ we can obtain the vectors $\mathbf{e}_h = \mathbf{r}[f_h(u)] - \mathbf{r}[f_h(v)]$ and $\mathbf{e}_z = \mathbf{s}[f_z(u)] - \mathbf{s}[f_z(v)]$ which characterize geometrically the graphs G_h and G_z. We propose to select the maximum common subgraph that minimizes the cost:

$$\mathcal{C} = \sum_{(u,v)\in E'} \|\mathbf{e}_h - \mathbf{e}_z\| + 1 - \frac{\mathbf{e}_h \cdot \mathbf{e}_z}{\|\mathbf{e}_h\| \, \|\mathbf{e}_z\|} \tag{8.4}$$

[1] A graph isomorphism of graphs G and H is a bijection f between the vertex sets of G and H such that any two vertices u and v of G are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in H.

The measure in Eq. (8.4) combines a cost proportional to the difference of relative distances between fingertips with a cost that penalizes matchings with distinct relative orientation between fingertips. In this manner, we take account of the geometrical structure of the whole fingertips set, of both the test and template match, which allow matching even in case of misses or false fingertip detections.

ORD maxima are found by clustering pixels depending on their thresholded ($> t_f$) ORD values into, at most, 5 clusters of a given minimal size s_f. For clustering, connectivity between pixels is verified, but also depth connectivity, thus we are subject to work with 3D data. For this purpose, we use the 3D Euclidean clustering proposed by Rusu in [Rus09]. Remark that t_f and s_f are parameters of the finger localization method. Summarizing, the method proceeds as follows, also depicted in Algorithm 2:

1. The test feature vector $\mathbf{x}_j$ is processed by the k-NN pose gesture recognition block. As a result, we match a template $\hat{\lambda}$ and build the graph G_h using the ground truth finger coordinates r.

2. Coordinates of ORD maxima $\mathbf{s}$ are computed using the clustering method and the graph G_z is built.

3. The maximum common subgraph G that minimizes the cost $\mathcal{C}$ in Eq. (8.4) is obtained, which defines the fingertips matching between the test patch and the template match.

4. Missing fingers in the test patch with respect to the template match are copied from the latter according to the average displacement between both sets of fingertip coordinates.

Input:
$\hat{\lambda}$ = matched template after k-NN
π_t = test template
Output: Test fingertip locations

Obtention of graph G_h from $\hat{\lambda}$, with n_h vertices
Obtention of graph G_z from $\hat{\mathbf{z}}$, with n_z vertices
Maximum common subgraph $G(G_h, G_z)$ than minimizes cost $\mathcal{C}$ in Eq. (8.4)
n_g fingers are obtained from G, displaced from G_h to G_z
if $n_g < n_h$ **then**
    the remaining $(n_h - n_g)$ fingers are copied from G_h, displaced with the average shift of the n_g first ones.
end if

ALGORITHM 2: Fingertip Localization

9

Collaborative Voting

9.1 Introduction

Object recognition algorithms have explored various ways of handling the available data. For example, in [TSSF12] a pixel-wise approach is proposed. That is to say, they process every pixel independently, obtaining the probability of that pixel belonging to a given part of a human body. When all pixels are classified, they are matched to a *canonical* body pose called the Vitruvian manifold. On the other side, the NNGM approach proposed in Chapter 8 handles the whole amount of information at once (depth point cloud of the hand) and exploits it to obtain the class (gesture) and some of its parts (fingertips). Between these two extrema, many approaches have been proposed, with the granularity of information being a major difference. For example, meaningful body parts are used in [FGMR10]. Bigger parts called pose-lets are used in [BM09]. These pose-lets are body parts tightly clustered in both appearance and configuration space.

We propose a discriminative approach which builds on the voting idea of Hough Forests [GYR+11], and also on the idea of describing object parts instead of the whole object, proposed in [FGMR10]. The objective of both methods is to detect an object given its parts. We propose to invert the formulation of the problem, trying to detect object parts given an object. For this purpose, we propose to describe object parts with the Oriented

Radial Distribution (ORD) feature [SRHC12a], which proved to be highly discriminative for hand gesture recognition using depth data [SALM+12]. In our method, object parts are meaningful from the ORD point of view. Each of these parts casts votes to other *annotated* parts, with a given confidence. Thus, a dataset of annotated object parts is required.

Fingertip localization of our Voting proposal is evaluated against a reference Random Forests method similar to [SFC+11], using the ColorTip dataset. The Voting method generalization to other objects is evaluated for the specific case of the human body. The Stanford dataset and its associated method [GPKT10a] is used, as well as the R-NBLS method proposed in Chapter 4.

We show experimental results on the ColorTip dataset and the Stanford human pose dataset [GPKT10a]. In addition, each object part may cast votes for additional information related to the object (*i.e.* overall pose). The Voting method ability to estimate a global property of the object under analysis is also evaluated. In this case, a benchmark of methods is used for comparison using the ASL dataset [ASL], showing how hand gestures are recognized.

9.2 Related Work

Object parts often are more characteristic of an object than global features of the object itself, as shown in [FGMR10]. With this idea, Gall *et al.* [GYR+11] propose a generalized method to obtain an object's centroid by casting votes from detected parts. These works use color images from a single camera.

Discriminative approaches have shown to be effective to detect parts of a given object using depth data. The Kinect's human body pose estimation [SFC+11] uses a Random Forest classifier to detect body parts (the object is a human body). In [GSK+11, SGF+12], such approach is extended for regression of human body parts. In [TSSF12], the classifier is pushed to the limit, considering every pixel a body part. In [KKKA12b], the approach proposed by [SFC+11] is extended to hand parts detection. The above mentioned methods require a large dataset in a training phase, which sometimes may be impossible to obtain or very cumbersome to manage in terms of memory allocation and processing power. Using a synthetic dataset has proven to be smart strategy to avoid endless shootings [SFC+11, KKKA12b]. However, the dataset size is does not decrease, and tends to increase due to the easily available data.

9.3 Voting Framework

In [GYR+11], each part of an object casts votes with the objective of finding the object's centroid. A dataset of objects with annotated object parts and votes is used for training a Hough Forest, which is utilized in the test phase to infer votes from the testing object parts.

Inspired by [GYR+11], we propose to cast votes from each object part to other object parts, constructing a collaborative object part detector.

9.3.1 Training Templates

A given object $\mathcal{O}$ in the database may contain N annotated parts at positions $\{\mathbf{g}_j\}$ with $j = 1..N$, and a maximum of $2M$ anchor points $\{\mathbf{a}_i\}$ with $i \leqslant 2M$. The object $\mathcal{O}$ is enclosed by an $L \times L$ pixel bounding box.

We define a training template λ_i, located at position $\mathbf{a}_i$ (anchor point), and enclosed by a smaller bounding box of size $l_i \times l_i$, as $\lambda_i = (\mathcal{I}_i, \mathbf{s}_i, \{\mathbf{v}_{i,j}\}, \{c_{i,j}\})$ (see Figure 9.1). More precisely,

$\mathcal{I}_i$ are the features extracted from the $l_i \times l_i$ image patch surrounding the object part,

$\mathbf{s}_i$ is the normalized position of the object part with respect to the center of the $L \times L$ object patch, such that $\mathbf{s}_i = \frac{\mathbf{a}_i}{L} - (\frac{1}{2}, \frac{1}{2}) \in \mathbb{R}^2$,

$\{\mathbf{v}(\mathbf{a}_i, \mathbf{g}_j)\}$ are the normalized votes from the current anchor point to the parts of the object, such that $\mathbf{v}(\mathbf{a}_i, \mathbf{g}_j) = \frac{\mathbf{g}_j - \mathbf{a}_i}{L} \in \mathbb{R}^2$. Remark that we vote at position $\mathbf{v}$ for a given class $c(\mathbf{g}_j)$,

$\{c_{i,j}\}$ are the class labels of the voted object parts. We consider N_C classes, belonging to the class space C (*i.e.* finger label amongst the 5 possible fingers in the case of human hand analysis).

In this chapter we refer to $\mathcal{I}$ for the ORD values of the pixels in the $L \times L$ object patch. Consequently, the $\mathcal{I}_i$ term contain the ORD values for the pixels in the $l_i \times l_i$ part patches. In order to normalize the result for further comparison with other templates, these ORD values are re-sampled to an $m \times m$ grid for each part, with $m < l_i \; \forall i$. Anchor points $\{\mathbf{a}_i\}$ of $\mathcal{O}$ are also obtained from the ORD calculation. We consider the ORD as a function $\text{ORD} : \mathbb{R}^2 \mapsto [0, 1]$. Those ORD values higher than 0.7 are thresholded and clustered, taking the centroids of the M most prominent clusters as $\{\mathbf{a}_i\}$ components, if they exist. In a similar way, those ORD values smaller than 0.3 are thresholded and

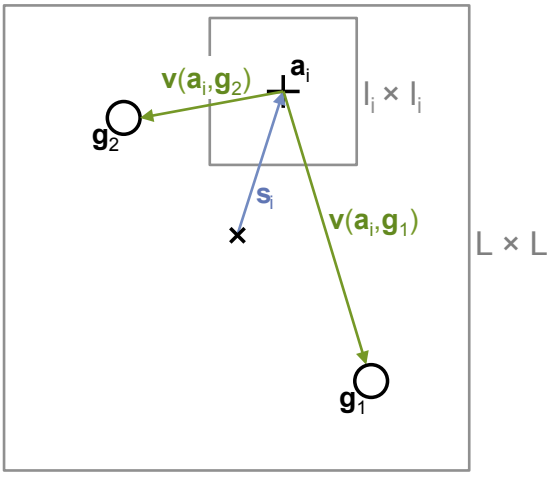

FIGURA 9.1: Training template definition. In this case, two votes $\{\mathbf{v}_{i,j}\}$ for $j = 1, 2$ are obtained from anchor point $\mathbf{a}_i$. The classes of $\mathbf{g}_1$ and $\mathbf{g}_2$ will compose $\{c_{i,j}\}$.

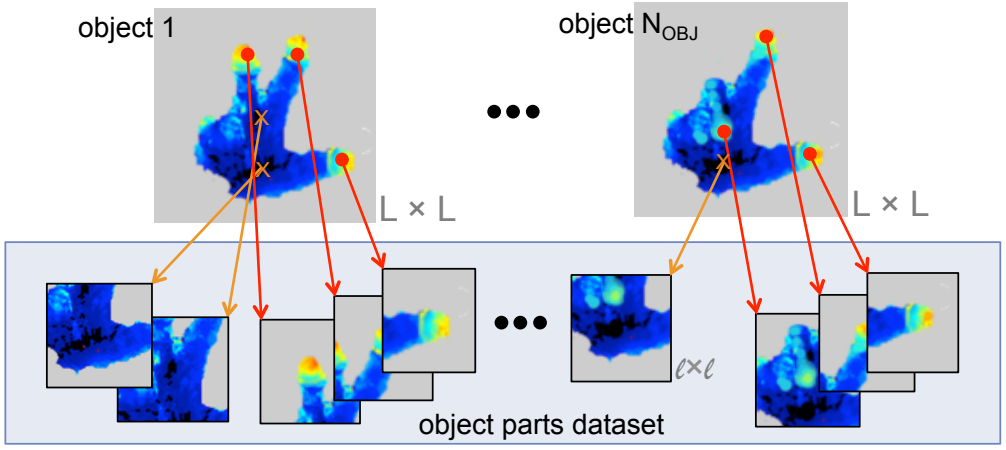

FIGURA 9.2: Object parts dataset construction. From a given object's ORD, anchor points are obtained as ORD maxima (red dots) and minima (orange crosses). A training template is built from every anchor point, composing the whole dataset. More details about training templates in Figure 9.1.

clustered, obtaining the remaining M components of $\{\mathbf{a}_i\}$ (Figure 9.2). Therefore, $\{\mathbf{a}_i\}$ will have, at most, $2M$ components. It may happen that less than $2M$ anchor points are obtained for a given object (*i.e.* few ORD values over 0.7). We have considered $M = 5$ in our work, resulting in a maximum of 10 anchor points per template.

9.3.2 Detection

In the detection step we consider a given testing object $\mathcal{O}$ with unknown parts $\{\hat{\mathbf{g}}_j\}$. Nevertheless, we can extract anchor points from $\mathcal{O}$ by applying the same ORD function used in the training step, and taking the M first maxima and M first minima. Thus, we obtain a maximum of $2M$ anchor points $\{\mathbf{a}_i\}$ which are suspicious of being similar the training ones, since they are obtained in the same manner.

The objective is to infer the unknown locations $\{\hat{\mathbf{g}}_j\}$ of the parts of the testing object, as well as their classes $\{\hat{c}_j\}$. Let $\hat{\mathbf{h}}(\{\hat{\mathbf{g}}_j\},\{\hat{c}_j\})$ be the hypothesis for the parts of the object $\hat{\mathcal{O}}$, located at positions $\{\hat{\mathbf{g}}_j\} \in \mathbb{R}^2$ of class $\{\hat{c}_j\} \in C$.

The object parts detection problem is then to maximize $p(\hat{\mathbf{h}}|\mathcal{I})$. The exact solution of this problem leads to evaluating the complete template dataset and extracting the best result that fulfills the hypothesis. To avoid such cumbersome calculation, we decompose the problem as shown in Equation (9.1)

$$\begin{aligned} p(\hat{\mathbf{h}}|\mathcal{I}) &= \sum_i p(\hat{\mathbf{h}}|\mathcal{I}_i,\mathbf{a}_i)\; p(\mathbf{a}_i|\mathcal{I}) \\ &= \sum_i p(\hat{\mathbf{h}}|\mathcal{I}_i,\mathbf{a}_i,\lambda)\; p(\lambda|\mathcal{I}_i,\mathbf{a}_i)\; p(\mathbf{a}_i|\mathcal{I}_i) \\ &= \sum_i p(\mathbf{a}_i|\mathcal{I}_i)\left(\sum_k p(\hat{\mathbf{h}}|\mathcal{I}_i,\mathbf{a}_i,\lambda_k)\; p(\lambda_k|\mathcal{I}_i,\mathbf{a}_i)\right) \end{aligned} \tag{9.1}$$

where λ is an auxiliary variable, measured directly from $\mathcal{I}$. We propose λ to be the closest K-Nearest Neighbors (K-NN) from the testing template. Thus, $\lambda = \{\lambda_k\}$ with $k = 1..K$, each element being a training template (Figure 9.3). In order to obtain λ we propose to use the cost function in Equation (9.2). The component $\mu_i(\lambda_k)$ is the L_2 distance[1] between the testing $\mathcal{I}_i$ and those in the whole dataset. We propose to add the spatial constraint $|\mathbf{s}_i - \mathbf{s}_k|$ to the cost function, which helps to select patches that occupy similar zones in the overall object patch. The K training templates are obtained by means of a *kd-tree* structure, selecting those with minimal $\mathcal{C}$.

$$\mathcal{C}_{i,k} = \frac{1}{2}\Big(\mu_i(\lambda_k) + |\mathbf{s}_i - \mathbf{s}_k|\Big) \tag{9.2}$$

Following with Equation (9.1), the term $p(\lambda_k|\mathcal{I}_i,\mathbf{a}_i)$ expresses the probability of selecting a given template λ_k. Thus, we may write $p(\lambda_k|\mathcal{I}_i,\mathbf{a}_i) = \mathcal{C}_{i,k}$

The term $p(\mathbf{a}_i|\mathcal{I})$ in Equation (9.1), noted as w_i, corresponds to the probability of obtaining the anchor point $\mathbf{a}_i$. Such probability is related to the value of the ORD maximum (or minimum) in it. More precisely, we note $w_i = 2\cdot|\mathrm{ORD}(\mathbf{a}_i) - \frac{1}{2}| \in [0,1]$. It is preferable to have weights with values between 0 and 1, so that their behave properly in Equation (9.3). Finally, the term $p(\hat{\mathbf{h}}|\mathcal{I}_i,\mathbf{a}_i,\lambda_k) = \sum_{\forall\lambda_k(\mathbf{g})} \mathbf{v}(\mathbf{a}_i,\lambda_k(\mathbf{g})) = \delta(\mathbf{a}_i,\lambda_k)$ may be explained as the normalized votes casted by the training template λ_k with origin at the current anchor point $\mathbf{a}_i$. Remark that $\delta(\mathbf{a}_i,\lambda_k)$ casts votes of different classes (the class of each $\lambda_k(\mathbf{g})$). Thus the initial problem is handled as a set of N_C scoring maps $\{\mathcal{S}\}$ of size $L\times L$, each one containing the votes of a given class $c \in C$.

[1] we have tried the Battacharyya distance, obtaining similar results with a higher computational cost

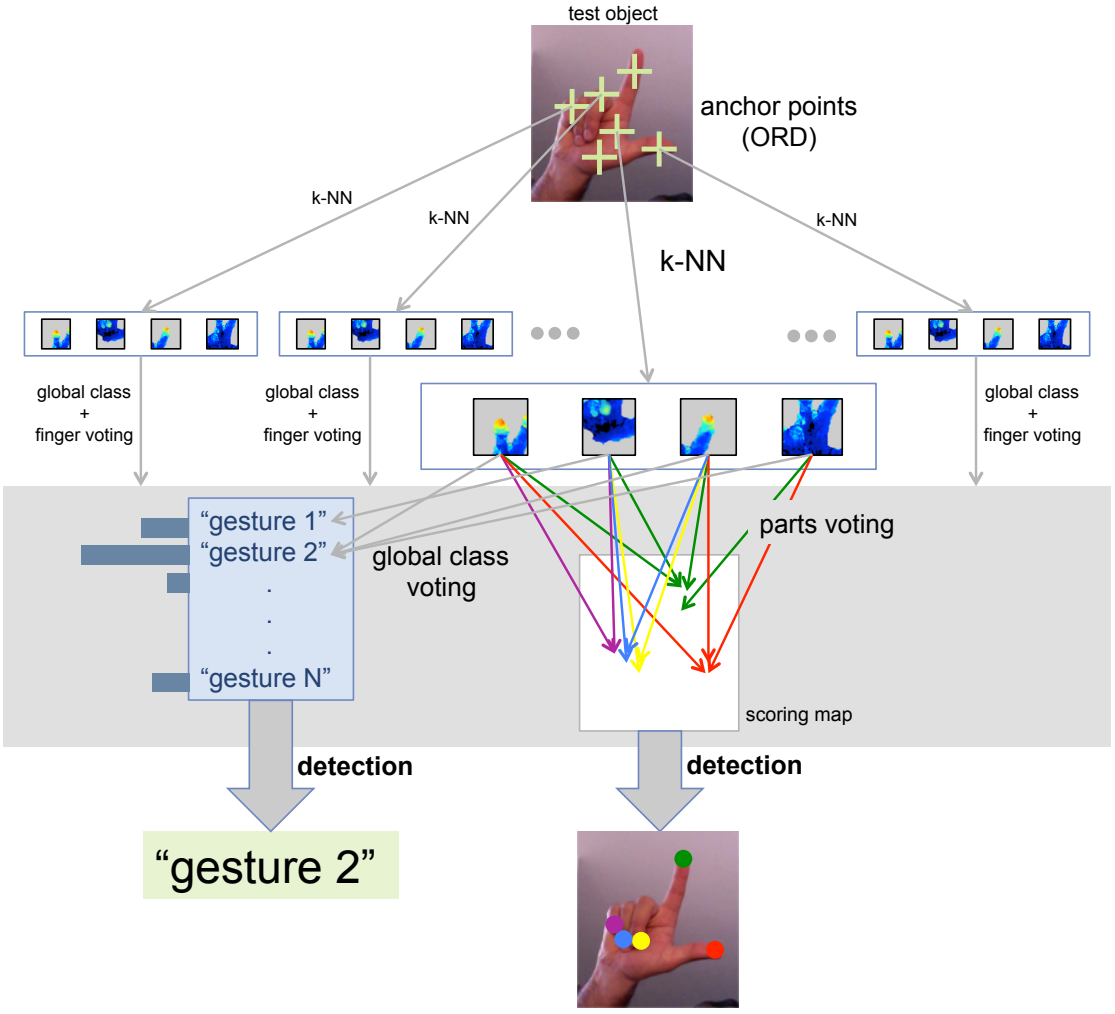

FIGURA 9.3: Graphical illustration of the proposed Voting framework. A set of anchor points is extracted from the testing object. The trained dataset is used in every k-NN operation. These obtained NN cast votes for parts on the scoring maps (see Figure 9.4 for an example). Finally, maxima of the scoring maps are found and parts are detected. The global class classification is also included in the scheme.

The initial problem may be re-written in a simplified form of known terms, as shown in Equation (9.3).

$$p(\hat{\mathbf{h}}|\mathcal{I}) = \{\mathcal{S}_c\} = \sum_i w_i \left(\sum_k \delta(\mathbf{a}_i, \lambda_k) \cdot \mathcal{C}_{i,k} \right) \tag{9.3}$$

Each of the scoring maps $\mathcal{S}_c$ contains a sparse set points which correspond to the votes of class c casted by $\{\lambda_k\}$ from $\{\mathbf{a}_i\}$. We apply a Gaussian filter with $\sigma = \frac{L}{10}$, obtaining a filtered map $\tilde{\mathcal{S}}_c$ like those in Figure 9.4. The locations of the unknown $\{\hat{\mathbf{g}}_j\}$ are the locations of the maximum of each $\tilde{\mathcal{S}}_c$, whilst $\hat{c}_j$ is directly the class of $\tilde{\mathcal{S}}_c$.

We note here that this formulation assumes $j = 1..N$ so that $N = N_c$ (equal number of parts and classes in C). Such assumption is valid for a fingertip classification problem,

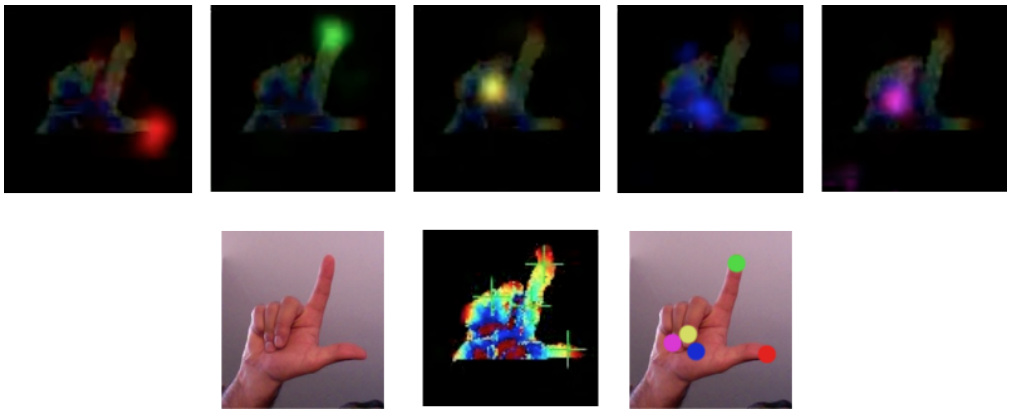

FIGURA 9.4: Fingertip localization example. The first row contains the filtered scoring maps $\tilde{\mathcal{S}}_c$ for each finer. The second row shows (left) the test hand, (middle) the obtained anchor points $\{\mathbf{a}_i\}$ on the ORD values and (right) the estimated fingertip positions.

for example. In case $N > Nc$, some parts belong to the same class. If we know how many classes are repeated in a given object, we may search for multiple maxima of $\tilde{\mathcal{S}}_c$ in order to obtain the locations of those $\{\hat{\mathbf{g}}\}$ of the same class c.

9.3.3 Global class inference

One could assign a global class $\xi \in D$ to an object, with N_D possible values. For example, in a fingertip classification problem, classes are the label of each finger (up to 5), whilst ξ may represent the overall hand gesture. More formally, an extra parameter ξ is added to the training templates, containing the type of object such part belongs. In

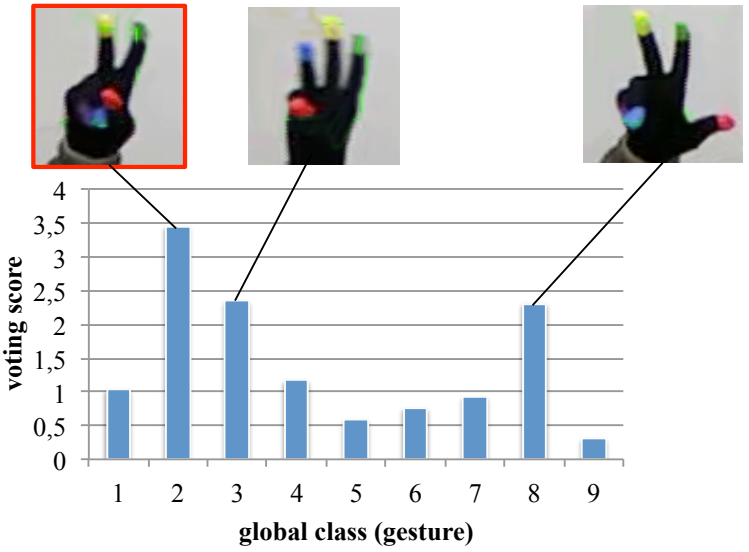

FIGURA 9.5: Example of the global class histogram S_ξ for a patch containing gesture number 2. We observe that gestures 3 and 8 also obtain a high score, since they have many similar parts with gesture 2.

order to estimate the unknown $\hat{\xi}$ of a testing object, we propose to analyze the obtained $\{\lambda_k\}$ (see Equations (9.1) and (9.3)). More precisely, a 1-dimensional score histogram S_ξ is constructed, with N_D bins. Thus, each bin $\lambda_k(\xi)$ is filled with the matching score $\sum_i \mathcal{C}_{i,k}$ between the template λ_k and all the anchor points $\{\mathbf{a}_i\}$ (see Figure 9.5).

The unknown $\hat{\xi}$ is straightforward obtained as the bin of S_ξ with maximal value.

10

Experimental Results

10.1 *Handbox* Head tracking results

In order to evaluate the performance of the proposed head tracking, we analyze the convenience of conditions C^2 and C^3 in Equation (6.1), and how robustness is increased by exploiting depth data. We compare the proposed algorithm with a reduced version which only verifies condition C^1. This way, the contribution of C^2 and C^3 is shown. The estimation error between a ground-truth (manually marked) head position g^H and the estimated head position is calculated as $\varepsilon = |\mathbf{g}^H - \hat{\mathbf{z}}^H|$, and presented in Figure 10.1 for the two versions of the algorithm. Note that, even if C^1 plays the role of 2D method, depth data is used for the initial head size estimation.

Figure 10.1 shows the estimation error of a sequence which contains two persons and three main events. Around frame 50, a single person waves hands before his head, hiding it. At frame 135, he performs a gesture with both hands which partially cover the head zone. At frame 300 the second person enters the scene and walks behind the first person. These events are illustrated in Figure 10.2, showing the successful behavior against occlusions and clutter.

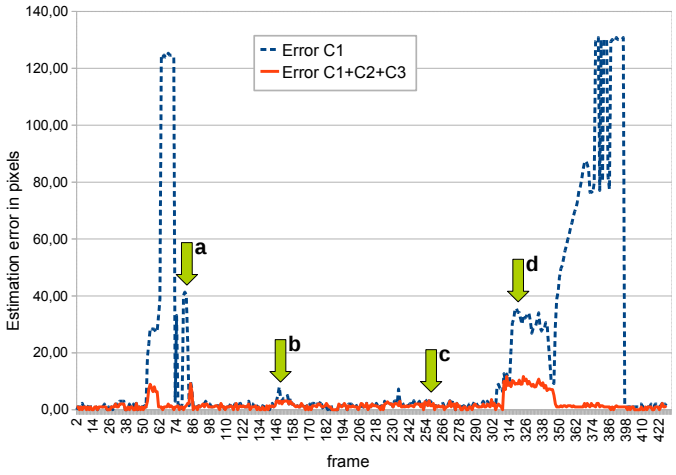

FIGURA 10.1: Error between the obtained head estimations and the ground-truth. $C^1 + C^2 + C^3$ error does not go above 10 pixels, which is about the head radius. The C^1 version loses target twice, a reset of the algorithm being needed (frames 74 and 398). The labeled arrows in Figure 10.1 correspond to the frames shown in Figure 10.2.

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

FIGURA 10.2: Head tracking snapshots (SR4000 TOF camera) of the sequence presented in Figure 10.1. In red, the C^1 version, in green the $C^1 + C^2 + C^3$ and in blue the ground-truth position. Note how the $C^1 + C^2 + C^3$ is more robust in these adverse situations (occlusions and clutter).

The proposed $C^1 + C^2 + C^3$ algorithm is able to track the first user's head despite these three polluting events, while the C^1 versions loses the target when the shape of the head is changed (hands moving, persons walking, etc.).

10.2 *Handbox* Head+Hands tracking results

As stated in Section 6.4, hand tracking depends on the robustness of head tracking. A set of gestures have been considered to analyze the performance of hand tracking, including gestures with one or two hands such as waving, pointing or separating hands apart (see Table 10.1). Hand centroids are manually marked on the depth video sequences. Thanks to the depth information, 3D ground-truth trajectories may be extracted from the 2D manually marked points. Such 3D ground-truth trajectories are used for comparison hereafter.

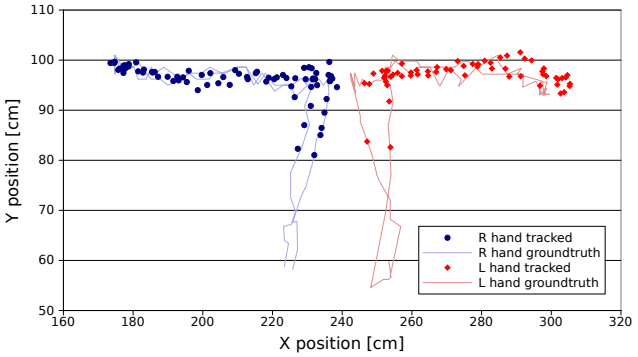

FIGURA 10.3: Ground-truth and estimated trajectories of the (R)ight and (L)eft hands. The estimated hand positions are fairly close to the reference ground-truth positions. Only the XY projection of these 3D trajectories is shown.

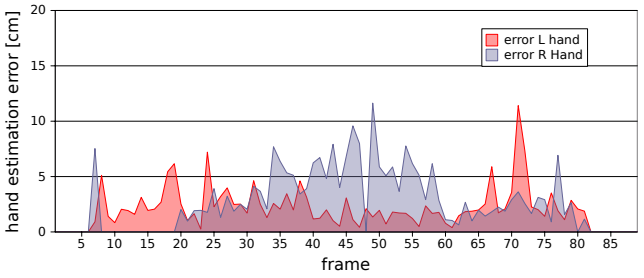

FIGURA 10.4: Hand estimation error, calculated as the Euclidean distance between the estimated and ground-truth 3D positions. The maximum error is about 10 *cm* for this sequence.

In Figure 10.3 the marked and estimated trajectories of both hands are presented corresponding to a gesture where each hand moves horizontally, like a slow hand clap (only the frontal projection of these 3D trajectories is shown for visual clarity). The error between the estimated and ground-truth trajectories is shown in Figure 10.4.

Hands are detected from the moment that they enter ◊ (frame 7). During the first 10 frames, both hands are touching each other, misleading the tracker which only *sees* one hand. However, once they are separated for a few *cm*, the tracker is able to detect and track both hands. It should be emphasized that both ground-truth and estimated trajectories are polluted by noise from the depth estimate. In addition, ground-truth trajectories have been extracted by hand, selecting a reasonable hand center which may vary in some *cm* along frames. Despite these adverse noisy conditions, the hand estimation error rarely goes above 10 *cm*.

Table 10.1 summarizes the average error for different one-handed and two-handed gestures gestures, computed as the average 3D error (with respect to the ground-truth hand positions) along the duration of the gestures. For example, the errors for *separate hands* gesture are calculated from the errors in Figure 10.4. Even if accuracy along the depth

TAULA 10.1: Hand detection 3D accuracy on different gestures

Gesture	# frames	error R hand	error L hand
push	30	2.62 *cm*	-
circle	30	6.61 *cm*	-
replay	35	2.86 *cm*	-
hand up-down	115	5.87 *cm*	-
separate hands	75	2.36 *cm*	3.80 *cm*

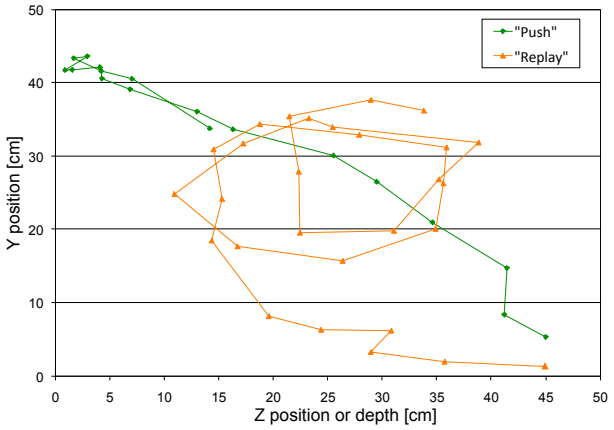

FIGURA 10.5: Trajectories obtained in a real-time experiment for the *push* and *replay* gestures. These results could be an interesting input for a classification step. Only the YZ plane is presented as movement is mainly contained in such plane.

axis depends on the type of sensor, we found it interesting to include it in the overall error, since both the Kinect and SR4000 used sensors provide a similar $\sim 1\,cm$ depth precision.

The average error is higher for fast movements such as the *circle* and the *up-down* gestures, resulting in about $6\,cm$ of error. For the other gestures, the error is of about $3\,cm$, which is fairly adequate given the size of a human hand.

For the sake of illustration, the trajectories corresponding to the *push* and *replay* gestures are presented in Figure 10.5. The *push* gesture consists in extending one arm from the body towards the camera and backwards. The *replay* gesture consists in describing circles with one hand in the YZ plane. Since both gestures are representative on the YZ plane, only these coordinates are presented, relative to the HandBox $\Diamond$ origin. The nature of the gesture may be easily derived from the trajectory, hence, such results may be interesting for further classification purposes.

10.3 *Handbox* vs. reference methods

Experimental results of the compared methods are summarized in Table 10.2. The proposed method largely overcomes the referred state of the art methods in terms of speed of execution. The fastest method is the work of Knoop *et al.* [KVD06], with a frame-rate up to 14 fps using a similar resolution. Our proposal achieves 68 fps on a regular CPU, which is about $4-5\times$ faster than the cited method. Moreover, some of the proposals in Table 10.2 use GPU implementations [LKAR10, GPKT10a], which should speed their performance up.

As for the accuracy of the hand detection and tracking, the proposed method performs with an average error of about $3-6\,cm$ while gestures are performed. Such error is similar to that claimed by Zhu *et al.* [ZDF08]. It should be remarked, in favor of the latter work, that hands are tracked at every time instant, no matter whether a gesture is being performed or not. This detail makes tracking more difficult, since hands may suffer of more occlusions or they may be located very near the body. Keeping an average error between $3-10\,cm$ is impressive.

Ganapathi *et al.* [GPKT10a] state that a $10\,cm$ error may be considered as a perfect match with the ground-truth, and that an error smaller than $30\,cm$ is good extremity match. In their work, they address the problem of a global full-body pose estimation and they detect all the visible extremities of a human body with an average error between $10\,cm$ and $20\,cm$. Our proposal takes advantage of a local and focused approach to overcome such results, as far as hand and head are concerned. The trade-off between accuracy and amount of information (full-body vs. only hands) is clearly observed after these results.

Recently, a Kinect SDK [Pri11] has been released, providing a complete human body pose estimation at about 20 fps. Head and hands may be extracted from the complete body, providing a comparable usage to the proposed method with a similar accuracy. However, the tracking in [Pri11] is slower than the proposed method, and it is partly performed in the Kinect hardware, reducing the flexibility of the approach.

10.3.1 Computational Performance

The main applications of the proposed head+hand tracking algorithm require real-time processing, otherwise natural interactivity would not be possible. In order to evaluate the real-time performance of our proposal, execution speed experiments have been carried out on a Intel Xeon 3GHz CPU.

TAULA 10.2: Comparative summary

Authors	Camera	Resolution	Full body	Speed	GPU	Accuracy
[KVD06]	SR4000	176×144	yes	$10 - 14\,fps$	no	-
[GKK07]	PMD	64×48	yes	$5\,fps$	no	-
[ZDF08]	SR3000	176×144	no	$10\,fps$	no	$3 - 10\,cm$
[LKAR10]	XB3	1280×960	yes	$2.5\,fps$	yes	-
[GPKT10a]	SR4000	128×128	yes	$4 - 6\,fps$	yes	$10 - 20\,cm$
HandBox	Kinect, SR4000	160×120	no	$68\,fps$	no	$3 - 6\,cm$

Real-time experiments are performed with a Kinect depth camera, which is down-sampled by a factor of 4, resulting in depth images of $R_4 = 160 \times 120$ pixels, which is the resolution used for the accuracy experiments presented in Sections 10.1 and 10.2. Experiments are run in sequences where the head and both hands are detected, which is the worst case in terms of computational load.

After hundreds of experiments with different two-handed gestures, we obtain a processing frame rate $f^P \approx 68\,fps$, which is largely enough for real-time applications. Moreover, it should be remarked that no GPU computing power is used in this proposal.

However, some experiments are also performed at other resolutions, specially $R_2 = 320 \times 240$ (down-sampling by 2) and $R_1 = 640 \times 480$ (original Kinect resolution). Real-time is slightly achieved with R_2, with a frame-rate of about $9\,fps$, while R_1 is too slow for any real-time purpose.

When dividing the overall processing time into tasks, it may be noticed that the head tracking task takes 92.4% of the total, while hand tracking is much faster (7.6%) of CPU time.

10.3.2 *Handbox* public demonstrations

The proposed method has been set up for many public demonstrations. Satisfactory qualitative results and feed-back of the users have been retained up to the moment. The reliability of the algorithm has specially been assessed in two main shows:

- The 2011 *International Conference on Multimedia and Expo* (ICME) in Barcelona. Over 70 persons where able to test the demonstration at ICME.
- The 2011 *International Broadcasting Convention* (IBC) in Amsterdam, as a part of the FascinatE project [Fas]. At IBC, the hand tracking positions where used to

control a panoramic video stream, being able to navigate within it (pan, tilt, zoom) and to grab the screen with the open/closed hand detector. Over 150 persons where able to interact with the demonstration with very encouraging results and feedback.

After these public experiments, some conclusions where drawn. The head detection algorithm performed with extreme robustness, only missing the head position at initialization for very few users (less than 1%). The hand tracker also proved to be robust with many different users (in height, shape, hand size, arm length, etc.). Open and closed hand distinction was somehow hard for users with very big or small hands. However, after some tries, users got used to the system operation, all of them managing to properly interact.

10.4 NNGM and Voting results

In this Section, we provide experimental results and evaluation of the hand analysis methods presented in this part, referred as NNGM (Chapter 8) and Voting (Chapter 9).

We show the performance of both methods using different datasets (*i.e.* ColorTip [Col], Stanford Dataset [GPKT10a] and Pugeault ASL Dataset [ASL]), which indeed helps showing the results obtained for various applications: fingertip localization (ColorTip), hand gesture recognition (ColorTip and ASL) and object part detection (Stanford).

Video results can be consulted in https://www.dropbox.com/sh/f6tls37lb9rezex/qRsTU5sjTh.

10.4.1 ColorTip

The ColorTip dataset, consisting of a total of 14 sequences (Chapter 7), is used for evaluation in the following experiments. We distinguish between *Set A* and *Set B* sequences in the results, given the considerable difference of intra-gesture variation.

10.4.1.1 Experimental Setup

Gesture classification results are obtained considering a *leave-one-subject-out-cross-validation* (LOSOCV) strategy. Thus, results for subject-i are obtained using as training dataset the remaining subjects' sequences. Remark that if we are considering the subject-i *Set A*, the sequence subject-i *Set B* is not used for training, and viceversa. The same strategy is used for fingertip localization results.

10.4.1.2 Selection of NNGM parameters

A study to select the most suitable parametrization is presented in this section. Parameters

- k : number of NN neighbors
- Q : temporal buffer of the kNN DC version
- m : grid sampling $m \times m$ of the ORD patch

are varied, maximizing the gesture detection F-Measure over the whole ColorTip dataset. For the sake of example, we provide experimental results for the selection of the number of k-NN and Q to be used (Figure 10.6). These results show that considering about $k = 15$ nearest neighbors and $Q = 5$ provides the best results. Parameter $m = 12$ was already fixed by experimentation.

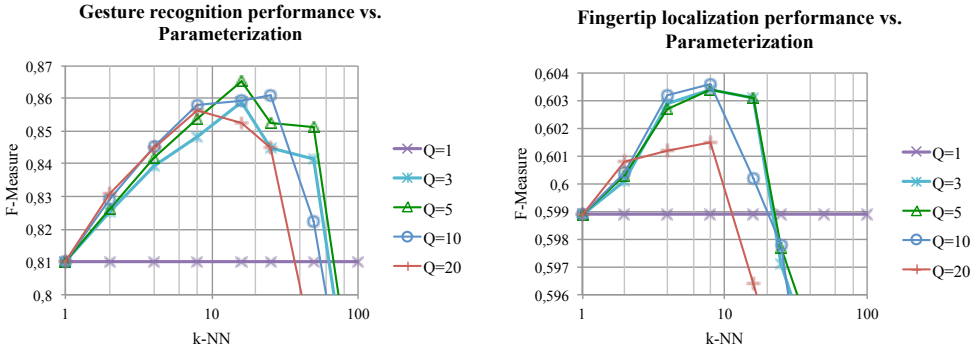

FIGURA 10.6: Selection of the optimal parametrization {k-NN, Q} to use. Experiments show that the best trade-off between gesture recognition and fingertip localization performance is k-NN=15 and $Q = 5$.

10.4.1.3 ORD vs. Benchmark

The suitability of using the ORD feature for gesture recognition is evaluated. With this purpose, we compare the results obtained with ORD using both proposals (NNGM and Voting) against a benchmark of various 3D features. We note $\mathcal{P}$ the 3D point cloud of the segmented hand, and $\rho(\mathbf{z})$ the neighborhood of radius ρ of a point $\mathbf{z}$. The proposed benchmark consists of:

- **Depth** Computed with respect to the average depth of $\mathcal{P}$.

- **Curvature** Computed as $\frac{\lambda_0}{\lambda_0+\lambda_1+\lambda_2}$, where λ_i are the eigenvalues of the eigen-decomposition of $\rho(\mathbf{z})$.
- **3DSC** 3D Shape Context, Frome *et al.* in [FHK+04].
- **VFH** Viewpoint Feature Histogram, Rusu *et al.* in [RBTH10].
- **SHOT** Signature of Histograms of OrienTations, Tombari *et al.* in [TSD10]

The 3DSC and SHOT features provide pixel-wise histograms. As proposed in [RHBB09], to obtain scalar values per pixel, we compute the Kullback-Leibler divergence between each histogram and the average histogram. Then, the same $m \times m$ sub-sampling as in Section 8.4 is performed, obtaining m^2 sized feature vectors. The depth and curvature features provide a scalar values per pixel, and the VFH feature already delivers a feature vector of size 308, which is used untouched.

In order to compare ORD against the benchmark, a k-NN by majority classification is performed with every feature (see Eq. (8.3)). For this experiment, we use our 14 training sequences, with a LOSOCV strategy. We present in Fig. 10.7 the average F-Measure of the 14 tested sequences (about 18000 frames). The F-Measure is calculated as $\frac{2 \cdot P \cdot R}{P+R}$, where P = precision and R = recall. Results obtained with Dynamically Constrained (DC) k-NN classification using ORD (Section 8.4.1) are also included.

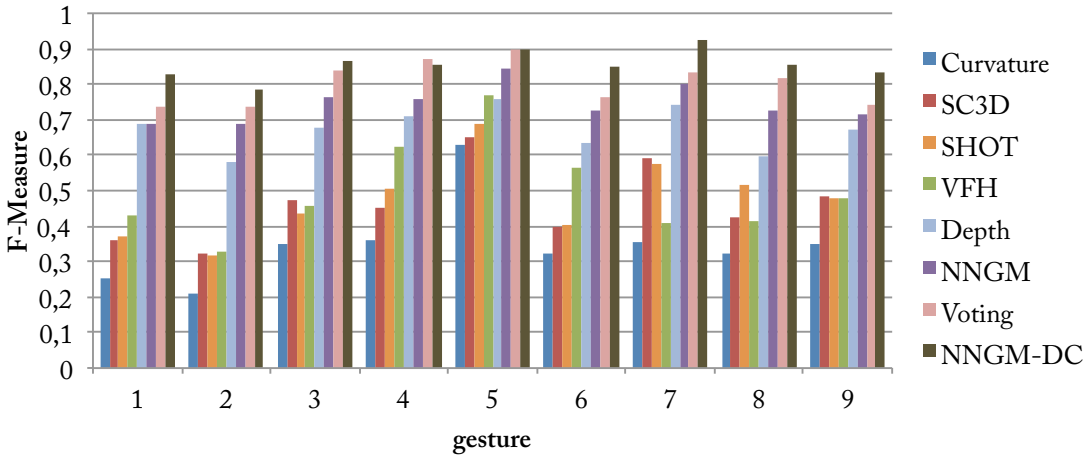

FIGURA 10.7: Comparison between using a 3D feature benchmark and ORD. All the experiments are obtained with k-NN classification by majority.

The ORD feature outperforms the benchmark, with an average F-Measure of 0.75 for the NNGM method and 80.7 with the Voting method. The fact that ORD is focused on characterizing 3D surfaces (by adapting orientation locally) helps achieving such results, since $\mathcal{P}$ is indeed a 3D surface. We remark that, looking at frame-wise methods, the

Voting strategy obtains better results than NNGM. We can state that the parts-based classification strategy adopted in the Voting method is better when using the ColorTip dataset.

The DC k-NN version of NNGM proposed in Section 8.4.1 helps increasing the F-Measure from 0.75 to 0.86 by exploiting video temporal consistency of gestures.

The best features in the benchmark are *depth* with an average F-Measure of 0.67 and *VFH* with 0.50. The benchmark features do not take into account the 3D surface nature of $\mathcal{P}$, and analyze it as it was a 3D point cloud.

10.4.1.4 Influence of the feature vector size

The dimension $m \times m$ of the feature vectors $\{\mathbf{x}_i\}$ has a noticeable impact on the hand gesture recognition results. In order to assess such effect, and with the objective of selecting an optimal value for m, we extract hand gesture recognition results for various values of m (Fig. 10.8) using the NNGM method. We recall that a feature vector consists of the re-sampling of an ORD patch to an $m \times m$ grid.

Experiments show that low $m \approx 4$ values lead to feature vectors which are not representative enough to distinguish between gestures. On the other hand, large $m > 14$ values lead to an over-fitting problem, since feature vectors become too related to data (usually noisy). In such case, the predictive performance degrades. Thus, values of $m \approx 12$ provide the best results in terms of hand gesture recognition.

We apply the same re-sampling of ORD in the Voting framework. The only difference being the meaning of the patch that is described with ORD. In NNGM, the patch covers the whole object, while in the Voting scheme, a patch covers the surrounding of an anchor point.

10.4.1.5 Influence of the dataset size using NNGM

The size of training datasets may suppose the bottleneck of a classification system. Designing scalable methods is crucial, allowing further incorporation of new training data if required. Furthermore, memory access and capacity problems may also occur due to large datasets.

We analyze in this experiment how the proposed NNGM method behaves with small training datasets. A basic clustering by Euclidean distance is performed to reduce the original training dataset $\mathcal{H}$, taking advantage of the already built *k-d tree*. More

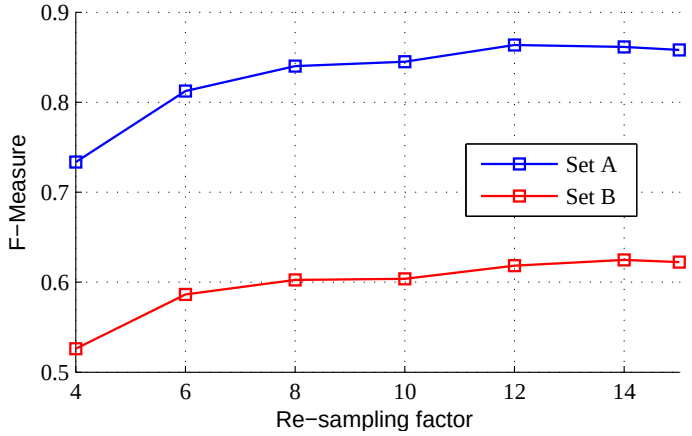

FIGURA 10.8: Effect of the re-sampling factor m on the hand gesture classification. We observe how re-samplings to $m \approx 12$ provide the best results.

precisely, a template $\lambda_j \in \mathcal{H}$ is randomly selected, grouping all those templates λ_i at a certain distance $\|\mathbf{x}_j - \mathbf{x}_i\| < D$ into a new average training template $\bar{\mathbf{h}}_j$. Such step is repeated until all the original templates are checked, obtaining the reduced dataset $\bar{\mathcal{H}} = \{\bar{\mathbf{h}}_j\}$. We note $F_{\%} = \frac{\bar{\mathcal{H}}}{\mathcal{H}}$ as reduction factor.

In Fig. 10.9 we present the F-Measure degradation as $\mathcal{H}$ is reduced, using a LOSOCV strategy. The original experiment at ($F_{\%} = 100\%$) consists of an average of 15200 training templates.

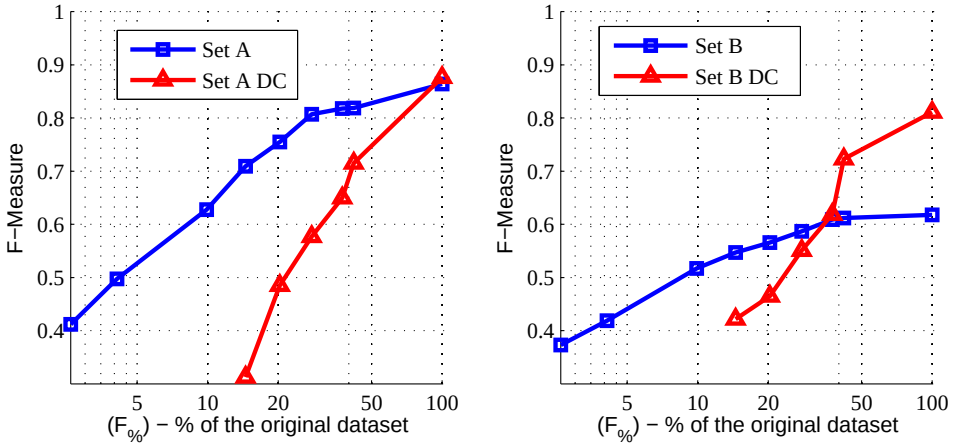

FIGURA 10.9: F-Measure degradation for various reduction factors of the training dataset. Remark that the k-NN search degrades slower than k-NN DC. However, the latter performs better with the complete dataset, as already shown in Fig. 10.7. Such effect is more visible in the *Set B* sequence.

The proposed NNGM method successfully tolerates drastic reductions of the training dataset. Such scalable behavior allows reducing the training dataset until $F_{\%} = 20\%$

(3040 templates) with a degradation of less than 5%.

The k-NN DC search performs better with the complete dataset, even if in the case of *Set A* sequences such effect is barely visible given the good performance of the stand-alone k-NN. We remark that, in the case of *Set A*, the performance without DC is already close to the annotation error due to transitions between gestures. However, we note that k-NN DC degrades faster.

In the *Set B* case, at $F_{\%} \approx 35\%$ the stand-alone k-NN search already outperforms the DC version, since the number of erroneous gestures being smoothed grows.

In our case, a training template $\lambda_i = \{\mathbf{x}_i, \mathbf{r}_i, y_i\}$ occupies $12 \cdot 12 \cdot 4 + 10 \cdot 4 + 1 = 587$ *bytes*. Thus, at $F_{\%} = 20\%$, the reduced dataset only occupies about $3040 \cdot 587 \approx 1.78$ *Mb*. Scalability is achieved taking advantage of the robustness against drastic reductions of the dataset, allowing the incorporation of new training sequences at low memory cost.

10.4.1.6 Fingertip Localization results

We conduct several experiments to evaluate the ORD and the proposed framework in the fingertip localization task. First, we compare the proposed fingertip inference method with a state-of-the-art fingertip detector based on Random Forests (RF). The RF method is also used to demonstrate the suitability of the ORD feature for hand analysis tasks. Then, we show the computational performance of the proposed method.

The fingertip evaluation protocol consists in a LOSOCV. We consider that a finger has been correctly localized if the estimated location and the ground-truth location are within a distance of 10 pixels.

In order to evaluate the proposed algorithm, we implement a fingertip localization method using Random Forests (RF) [Bre01]. The RF localization method is based on the successful system for detecting body parts from range data proposed by Shotton *et al.* [SFC+11]. We use very similar depth-invariant features, but in addition to depth data, we include the ORD feature.

We employ [1]RFs comprising 10 trees of maximum depth 15. Three baselines are trained: one using depth (D) information exclusively, another using ORD exclusively and a baseline combining both features. The precision and recall performance of the RF approach is evaluated with 50 different detection thresholds (Fig. 10.10). The experiments reveal that RF trained with the stand-alone ORD values provide the best results (in red in

[1]see a detailed explanation in Appendix B

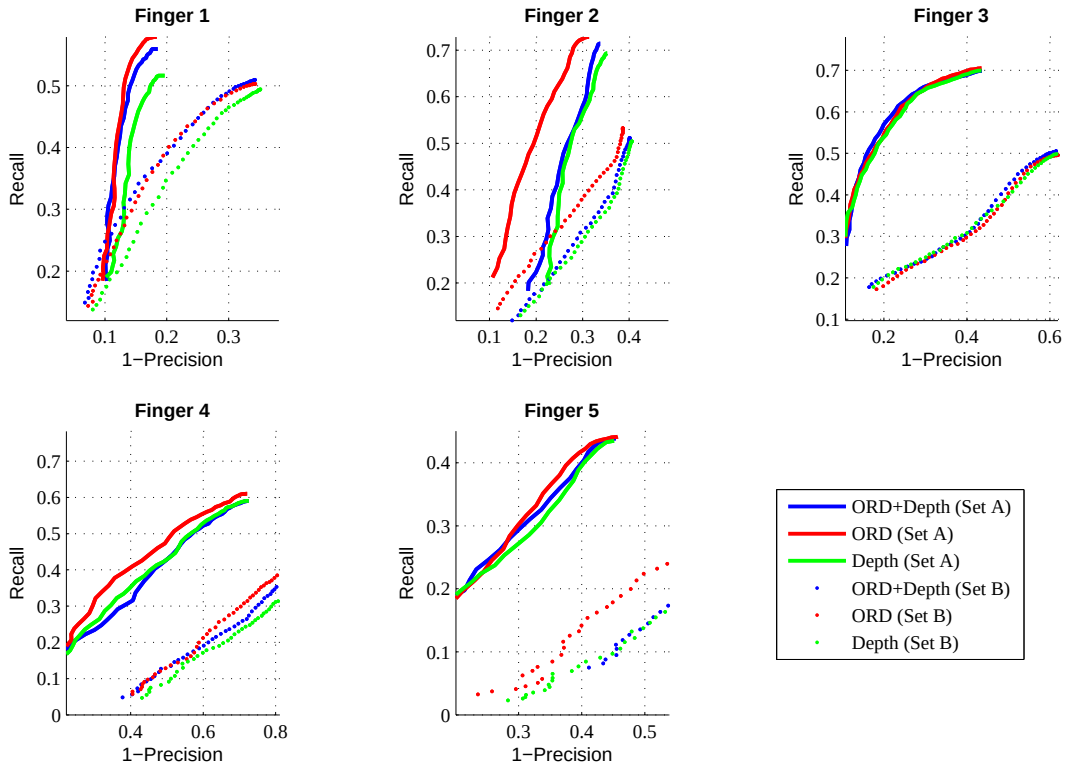

FIGURA 10.10: Fingertip classification results using the RF baseline approach. 50 different detection thresholds are used. Note that RF using the stand-alone ORD values obtain the best results.

Fig. 10.10), showing that ORD is also a suitable feature to locally describe parts of an object.

Results stepping over the gesture recognition are also included as onlyGM. In this case, we perform graph matching with the whole dataset templates, keeping the one with minimum cost $\mathcal{C}$ (see Equation (8.4)). The results show that, depending on the complexity of the test graph (G_z), doing graph matching with the complete dataset takes between 0.1 and 50 seconds per frame (ColorTip provides about 15000 templates). Also, this approach strongly degrades the performance of the fingertip localization task, obtaining an average precision/recall of 0.317/0.125, which means an F-Measure of 0.180. Such results are far worse (both in computational time and performance) than those obtained with the proposed combined approach.

Our approach is evaluated with 3 different t_f and 8 different s_f parameters, obtaining the best results with $t_f = 0.3$ and $s_f = 0.8\,cm^2$. Some visual results are provided in Fig. 10.11.

Comparative results between our approaches and the best RF baseline are presented in

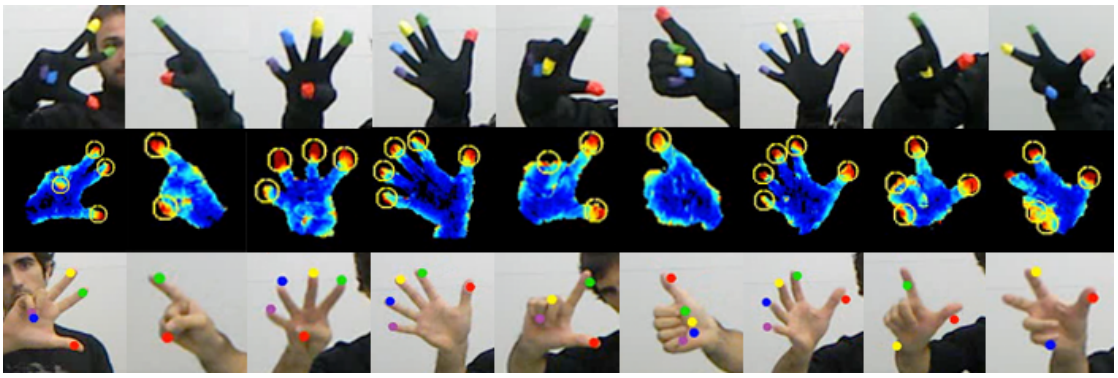

FIGURA 10.11: NNGM fingertip localization results (columns). The upper row contains the k-NN selected patch $\hat{\lambda}$ from the database, which intrinsically represents the recognized gesture. In the middle row, we show the ORD maxima to which fingers $\hat{\mathbf{r}}$ of $\hat{\lambda}$ are matched. The resulting fingertip localization on the testing hand is shown in the bottom row. Two erroneous examples are shown in the farthest right columns.

TAULA 10.3: **Set A** sequences - Comparative fingertip localization F-Measure.

	RF(D)	**RF(ORD)**	**RF(ORD+D)**	**onlyGM**	**NNGM**	**Voting**
f1	0.62	0.61	0.59	0.34	0.67	**0.80**
f2	0.64	0.69	0.64	0.26	0.66	**0.86**
f3	0.68	0.67	0.67	0.12	0.68	**0.83**
f4	0.46	0.49	0.46	0.09	0.54	**0.72**
f5	0.21	0.24	0.21	0.05	0.54	**0.80**
avg.	0.52	0.54	0.51	0.18	0.62	**0.81**

TAULA 10.4: **Set B** sequences - Comparative fingertip localization F-Measure.

	RF(D)	**RF(ORD)**	**RF(ORD+D)**	**onlyGM**	**NNGM**	**Voting**
f1	0.53	0.51	0.52	0.23	**0.59**	0.41
f2	0.47	0.50	0.46	0.17	**0.57**	0.35
f3	0.42	0.41	0.42	0.13	**0.51**	0.25
f4	0.27	0.29	0.26	0.06	**0.34**	0.30
f5	0.13	0.17	0.14	0.03	**0.37**	0.32
avg.	0.37	0.38	0.36	0.12	**0.48**	0.33

Table 10.3 (*Set A*) and Table 10.4 (*Set B*). The proposed methods consistently outperform all the RF baseline configurations. The main reason is the ability of our methods to infer fingertip locations using structured inference given a template pose (NNGM) or given a set of anchor points (Voting).

In the NNGM case, hand pose matching allows to robustly locate fingertips under several hand rotations, which is the main limitation of the RF approach. Moreover, the global structure of the hand pose helps to robustly detect fingertips even when there is weak

evidence of a finger location. In contrast, the RF approach requires each finger to have strong evidence in order to be robustly detected.

The Voting method provides impressive results (80.8% F-Measure) in the *Set A* sequence. An explanation for that is that a voting patch casts votes to the fingers that were present in the training step. Therefore, we have patches of different gestures casting strong votes for the actual fingers. A second point we may remark is that both ORD maxima and minima are taken as anchor points, exploiting the hand structure in a more complete way than in NNGM.

However, we appreciate that the Voting method degrades strongly when using the *Set B* sequence. Given the variability of hand poses in the sequence, the Voting framework struggles at detecting hand parts properly. We may conclude that the Voting parts detector presented in Chapter 9 is much more sensitive to dataset variability than the NNGM method.

The RF baseline results also show the suitability of the ORD for the fingertip localization task, in terms of F-Measure. Best RF performance is achieved when binary tests exclusively use the ORD descriptor. Interestingly, the ORD contributes to a significant increase in the index finger localization (finger 2).

In Figure 10.12 we show the behavior of the Voting fingertip localization in terms of F-Measure. We provide results with a fixed threshold (left) and a relative threshold ($th = \frac{\text{finger}^{max}}{factor}$). The best results are obtained with a fixed threshold of 3.0, achieving an average F-Measure of 80.8 (which is indeed the result reported in Table 10.3).

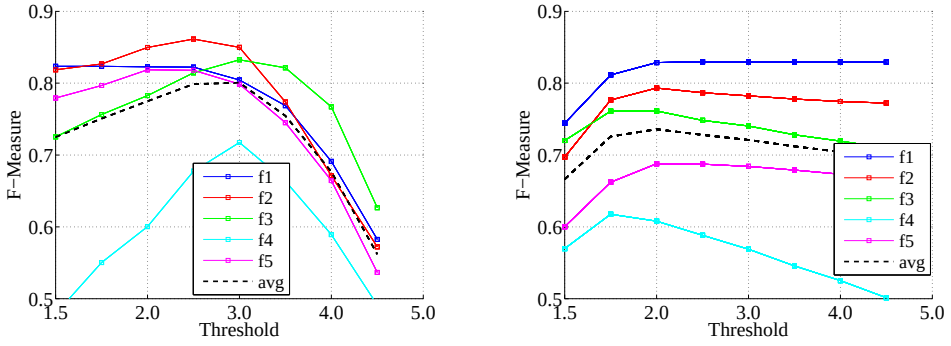

FIGURA 10.12: F-Measure of the fingertip localization performed with the Voting method.

10.4.2 ASL dataset

The ASL dataset provided by [PB11] in [ASL] is used in the following experiment. Such dataset contains annotated hand patches of 5 subjects recorded with the Kinect camera, performing 24 ASL alphabet gestures. Accuracy is used as a measure to be able to compare with other reference methods.

As in [PB11], the dataset is randomly split into equally sized training and test subsets. Doing so is advantageous for a k-NN strategy, since the probability of having a consecutive frame (very similar) in the training subset is very high. Such effect is reflected in the *Random* column of Table 10.5, achieving an accuracy of 99.3% against a 73% of [PB11] on the same dataset. Other classification methods achieve similar performance [ZYT13, KKKA12a].

A LOSOCV is also carried out. In that case, the Voting method achieves 78.1% and NNGM obtains an accuracy of 76.1%, still about 25 points higher than [PB11] (49%, results provided in [ASL]). Both the NNGM and Voting methods achieve state-of-the-art compared to methods [ZYT13, UGVV11], although results in [UGVV11] results are obtained in slightly different dataset. Only [KKKA12a] obtains significantly better results, taking advantage of a large training dataset built with synthetic data. We also include the confusion matrix of the Voting method (Figure 10.13) for a better visualization of the obtained classification.

It should be remarked that the reference methods [PB11, ZW12, UGVV11, ZYT13] are strictly gesture recognition methods, none of them provides fingertip localization. The proposed methods use the same ORD feature calculation to additionally provide such fingertip locations.

Method	Random	LOSOCV
[UGVV11]* Uebersax	88.0	76.0
[PB11] Pugeault (depth)	69.0	n/a
[PB11] Pugeault (depth+color)	73.0	49.0
[ZW12] Zhu (No Pyramid)	77.4	n/a
[ZW12] Zhu (Image Pyramid)	88.9	n/a
[KKKA12a] Keskin	97.8	**84.3**
[ZYT13] Zhang	98.9	73.3
NNGM	98.6	76.1
Voting	**99.3**	78.1

TAULA 10.5: Comparative ASL [ASL] hand gesture recognition accuracy
**Evaluated on a different dataset.*

Gesture	1	2	3	4	5	6	7	8	9	11	12	13	14	15	16	17	18	20	21	22	23	24
1	0,86	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,00	0,00	0,02	0,02	0,00	0,00	0,00	0,00	0,06	0,00	0,00	0,00	0,00
2	0,00	0,93	0,01	0,00	0,00	0,01	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00
3	0,01	0,03	0,87	0,00	0,02	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,01	0,00	0,01	0,02	0,00	0,00	0,00
4	0,00	0,00	0,00	0,93	0,01	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,01	0,00	0,00	0,02	0,00	0,00	0,00	0,00	0,01
5	0,01	0,00	0,00	0,00	0,60	0,00	0,00	0,00	0,01	0,00	0,00	0,08	0,02	0,06	0,00	0,00	0,00	0,09	0,00	0,00	0,00	0,00
6	0,00	0,01	0,00	0,00	0,00	0,94	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,02	0,00
7	0,01	0,00	0,01	0,00	0,00	0,00	0,65	0,25	0,00	0,02	0,00	0,01	0,01	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,01
8	0,00	0,00	0,00	0,00	0,00	0,00	0,04	0,95	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
9	0,09	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,81	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00
11	0,00	0,00	0,00	0,04	0,00	0,00	0,03	0,04	0,00	0,55	0,03	0,03	0,00	0,00	0,01	0,00	0,06	0,00	0,02	0,17	0,00	0,01
12	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,98	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
13	0,07	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,53	0,21	0,00	0,00	0,00	0,00	0,12	0,00	0,00	0,00	0,00
14	0,02	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,11	0,66	0,00	0,00	0,01	0,00	0,15	0,00	0,00	0,00	0,01
15	0,00	0,00	0,00	0,00	0,07	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,01	0,86	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00
16	0,00	0,01	0,00	0,06	0,01	0,00	0,01	0,03	0,00	0,00	0,00	0,01	0,00	0,03	0,64	0,12	0,00	0,00	0,00	0,00	0,02	0,03
17	0,00	0,00	0,02	0,01	0,02	0,00	0,01	0,01	0,01	0,00	0,00	0,00	0,07	0,08	0,07	0,65	0,00	0,00	0,00	0,00	0,01	0,00
18	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,02	0,01	0,00	0,00	0,00	0,00	0,00	0,76	0,00	0,19	0,00	0,00	0,00
20	0,10	0,00	0,00	0,00	0,01	0,00	0,00	0,01	0,00	0,00	0,00	0,07	0,23	0,00	0,00	0,00	0,00	0,47	0,00	0,00	0,00	0,00
21	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,05	0,00	0,93	0,00	0,00	0,00
22	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,05	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,00	0,01	0,90	0,01	0,00
23	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,96	0,00
24	0,00	0,00	0,00	0,03	0,00	0,01	0,02	0,00	0,00	0,05	0,00	0,00	0,05	0,01	0,04	0,03	0,07	0,01	0,01	0,01	0,02	0,63

FIGURA 10.13: Confusion matrix obtained with the Voting method on the ASL dataset.

10.4.3 Stanford'10 Dataset

We use the Stanford human pose dataset to show how the proposed Voting method may be generalized to obtain object parts, different than hands.

10.4.3.1 Experimental Setup

The Stanford dataset contains 28 sequences captured with a MESA SR4000 Time-of-Flight camera. A set of 23 ground truth markers is provided in the dataset, covering the main body parts such as head, hands, feet, elbows, shoulders, torso and pelvis. For these experiments, we use a LOSOCV strategy, using sequence i for testing and the complementary $j = 1 \dots 28$ $(j \neq i)$ sequences for training.

10.4.3.2 Body Parts Classification

The average detection error is provided in Figure 10.14. We have also included the results obtained with the R-NBLS method presented in Chapter 4. The method proposed by Ganapathi *et al.* [GPKT10a] in the same work where the Stanford dataset is published obtains an average error of 7.3 cm. The R-NBLS achieves 9.1 cm taking only 5 markers into account (head, hands and feet). The proposed Voting method obtains an average error of about 9.7 cm on the whole marker set. Therefore, we achieve an average error about 2.4 cm higher than [GPKT10a] with a generalized object parts detector. We remind the work in [GPKT10a] is fully dedicated to human body, incorporating a 3D model to fulfill these requirements.

Moreover, the Stanford dataset is composed of 28 *different* sequences. Therefore, it is difficult to find training examples similar to the test ones, since we are using the complementary sequences as training for a given test sequence. Despite this inconvenient setup, the proposed Voting approach manages to select appropriate training templates, achieving good classification results.

Results show that the most stable parts are torso, head, shoulders and pelvis (Figure 10.15), since they are more repeated in the complementary sequences. Such observation corroborates the above paragraph.

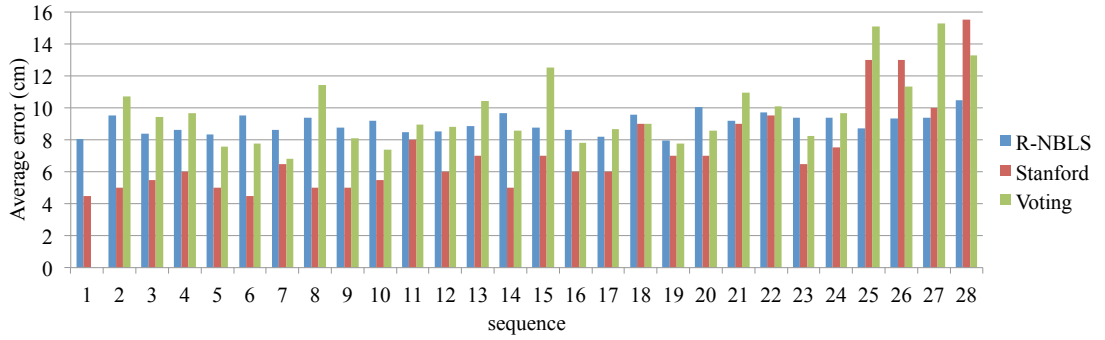

FIGURA 10.14: Average detection error in centimeters. We include in the comparison the R-NBLS method of Chapter 4.

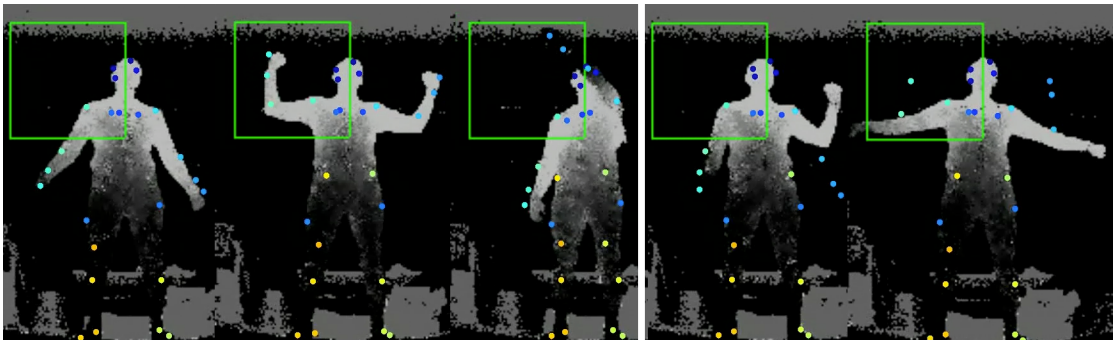

FIGURA 10.15: Examples of the body parts classification using the Voting framework. We show 3 satisfactory examples on the left, and 2 partially erroneous examples on the right. Note that errors are mainly located on arms (less similar examples in the training set). The green square represents the size of the voting patches.

10.4.4 Computational Performance

The hand analysis experiments are carried out on an Intel Core2 Duo CPU E7400 @ 2.80GHz. To calculate the ORD feature, we have coded a parallel implementation on a NVIDIA GeForce GTX 295 GPU, performing about $70-140\times$ faster than the CPU implementation. The overall NNGM approach (gesture recognition + fingertip localization) performs in real-time, at a frame-rate of about $15-17\,fps$. A frame-rate of $16\,fps$ is achieved by [UGVV11]. Remark that our proposal delivers fingertip positions

in addition to hand gestures. Moreover, a 176×144 camera is used in [UGVV11], with a smaller resolution than Kinect. Real-time is also attained by [PB11] for gesture recognition, using a state-of-the-art body tracker to detect hands. Public real-time demonstrations using NNGM have been carried out at ECCV'12 [SALM+12], and also within the FascinatE Project [Fas].

The Voting approach performs slower than NNGM since more processing steps are required. A higher amount of ORD re-samplings, distance calculations, update of score maps, etc. make the frame-rate slow down to about $6 - 7\, fps$. Moreover, in the Voting framework, the training dataset is composed of body parts, making it larger than in NNGM case. Even using a kd-tree for querying, the size of the dataset has some impact on the final processing time.

10.5 Conclusions

We have proposed two methods to locate fingertips and classify hand gestures. In both methods we make use of the ORD descriptor, which is used to characterize globally a hand point cloud, and to locally identify candidate fingertip locations.

In the NNGM proposal, the ORD output is used to feed a NN classifier, obtaining the estimated hand gesture. Based on this result, the search space where to find fingertip locations is reduced, limiting the search to a graph matching step. We use the ColorTip dataset to train the hand gesture classifier, and also to construct the graph to be matched in order to obtain fingertip positions and labels.

The second method consists of a voting framework to detect parts of a given object. We construct a parts dataset out from the ColorTip dataset, the overall ORD calculation at every part is used to obtain NN matches in a similar way than in the NNGM approach. Moreover, the ORD maxima and minima are selected as anchor points from where votes are casted to fingertip positions, depending on the obtained NN matches. This Voting method is indeed a general parts detector, showing successful results with hands but also with the whole human body. We propose to infer auxiliary variables from the Voting framework, such as the hand gesture. The Voting framework shows a performance degradation with strongly variable datasets such as ColorTip *Set B*

The ColorTip dataset itself is proposed to the research community, providing useful data for hand gesture recognition and fingertip localization.

Additionally, we have proposed HandBox, a fast and robust method to detect hands and head using depth data. This method has been used as a tool to extract hand positions for the above mentioned hand analysis methods.

Part III

Overall Conclusions

11

Contributions

11.1 Main contributions

1. **Description of Depth Data** : The sudden irruption of commercial depth cameras has enabled the incorporation of easy-to-obtain 3D information into existing works. However, classical color-based and 2D features may not be suited for such new data. We have focused on exploiting the depth information nature to propose novel ways to describe such data. Many present methods to analyze the human body rely on descriptors to enable a further classification step, using a discriminative strategy. Others, use descriptors to anchor a pre-defined body model, iteratively, in a generative way. In both cases, descriptors are crucial to obtain good results. It is interesting, then, to have adapted descriptors that extract as much information as possible from this new depth data.

   Our contributions to these problems and challenges are summarized as follows:

   - We propose an extension of the Narrow Bands Level Set to incorporate depth point clouds characteristics. More precisely, a density restriction is added to the formulation to better respect the topological properties of the analyzed object. We rely on the fact that current depth cameras provide point clouds with a typical structure and density, which is used as a prior in our proposal.

We also propose to filter the narrow bands depending on their physical area, which helps obtaining meaningful zones on the body under analysis. These zones may be used, for example, to populate a directed graph. We have shown that such graph may be exploited using shortest path algorithms to obtain the main end-effectors of a human body.

- The new Oriented Radial Distribution descriptor for depth is proposed. Such descriptor adapts to the local normal of a point cloud, treating it as a surface. The ORD returns high values at prominent parts of a point cloud, and low values on flat zones. Moreover, its multi scale formulation allows selecting prominent parts of a given size, filtering the remaining protuberances.

  We show that ORD also provides a good global representation of the point cloud, emphasizing its topology and structure.

  A GPU implementation of ORD is also proposed, which enables its usage in real-time applications.

Publications:

- [SRhC13] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Detecting End-Effectors on 2.5D data using Geometric Deformable Models: Application to Human Pose Estimation", Computer Vision and Image Understanding (CVIU), vol. 117, no. 3, 2013.*
- [SRHC12a] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Oriented radial distribution on depth data: Application to the detection of end-effectors", in IEEE International Conference on Acoustics, Speech, and Signal Processing, Kyoto, Japan, 2012.*

2. **A Baseline Method for Hand Detection** : Having an easy-to-setup and fast method to detect hands is an invaluable tool when doing research on hand analysis. Such tool should be tunable to fulfill the requirements of a given experimental setup. Also, it should not be time consuming, so that methods being analyzed are not affected by the hand detection memory and CPU consumption. Moreover, such a method should be robust, in order to minimize errors due to hand extraction in the final experimental results.

   Using the Kinect camera, there exist some SDK that provide hand positions. However, they are quite time consuming, not tunable and require full body frames to perform the detection.

   Our contributions to these topic are the following:

   - A fast and robust head detection algorithm based on depth cues. We propose to match human heads in the camera viewport with an elliptical template.

We incorporate an adaptive search zone that varies depending on the user's movement. Results have shown that, despite its simplicity, this method provides an extremely robust head detection.

- A hand detection method that relies on the above head detection proposal. This method is called HandBox, and basically consists in a 3D workspace placed in front of the user's head at every time instant. Some 3D heuristics are proposed to accept blobs in the HandBox as hands.

Publications:

- [SCRh11] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Real-time head and hand tracking based on 2.5D data", in IEEE International Conference on Multimedia and Expo (ICME), 2011, pp. 1–6.*
- [SRHC12b] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Real-time head and hand tracking based on 2.5D data", IEEE Transactions on Multimedia (TMM), vol. 14, no. 3, pp. 575-585 , 2012.*
- [ASM+] *M. Alcoverro, X. Suau, J.R. Morros, A. López-Méndez, A. Gil, J. Ruiz-Hidalgo, J.R. Casas, "Gesture Control Interface for immersive panoramic displays". Multimedia Tools and Applications, Springer. 2013.*

3. **Hand Analysis** : Hands are probably the most difficult part of a human body to analyze using Computer Vision. Indeed, hands may conform an enormous amount of poses, with many degrees of freedom in their movement. Occlusions and self-occlusions are another very common issue when doing hand analysis. Some works have focused on classifying hand gestures within a gesture dictionary. Very few works have addressed the problem of locating fingertips and classifying them, what we call fingertip localization.

   In this thesis, we have addressed both hand gesture recognition and fingertip localization, making the following contributions:

   - A public dataset for fingertip localization and hand gesture recognition is proposed, named ColorTip. The ColorTip dataset [Col] is composed of 7 users performing 9 different hand gestures. We provide fingertip annotations (position and label) as well as overall hand gesture annotations.
   - We propose to use the ORD descriptor to characterize depth point clouds for further classification purposes. We show that ORD is highly effective at describing a hand globally, enabling a precise gesture classification using both Nearest Neighbor and Random Forest strategies. A comparison with other existing features is provided, showing the suitability of using ORD for

such a task. Even with drastic reductions of the training dataset, the gesture classification is still acceptable using ORD.

- The ORD descriptor is also proposed to locally describe point clouds, obtaining its end-effectors. We exploit the multi scale formulation of ORD to detect hands and finger candidates.
- We have shown that the obtained finger candidates are stable enough to enable fingertip localization using a graph matching algorithm with ground truth finger positions. We propose a matching cost that exploits fingertip structure.
- A Voting framework to detect object parts using depth point clouds is proposed. In this framework, votes are casted from anchor points obtained from ORD maxima and minima, to different body parts. These votes are normalized and geometrically modified from the training template to the test template, to increase precision and robustness. Also, this framework accepts defining global classes, that may also receive votes, allowing global classification (*i.e.* hand gesture, body pose, etc). This way, we jointly obtain global information of an objects as well as its parts.

Publications:

- [SALM+12] *X. Suau, M. Alcoverro, A. A. López-Méndez, J. Ruiz-Hidalgo, and J. Casas, "INTAIRACT: Joint Hand Gesture and Fingertip Classification for Touchless Interaction", in Computer Vision – ECCV 2012, vol. 7585, Heidelberg: Springer, 2012, pp. 602-606.*
- [SALM+] *X. Suau, M. Alcoverro, A. López-Méndez, J. Ruiz-Hidalgo, J.R. Casas, "Real-time Fingertip Localization Conditioned on Hand Gesture Classification", Submitted to Image and Vision Computing.*
- *"A Colaborative Voting Framework for Parts Detection (journal submission in process)*

11.2 Side Contributions

Besides the main contributions mentioned above, additional work has been carried out during this thesis. Some of these works, even if not central to the research axis of this thesis, have provided interesting results for the research community, and have served as starting point for many of this thesis ideas.

- **Multi-resolution illumination compensation for foreground extraction**: Illumination changes may lead to false foreground segmentation and tracking results. Most of the existing foreground extraction algorithms obtain a background estimation from temporal statistical parameters. Such algorithms consider a quasi-static background which does not change but slowly. Therefore, fast illumination changes are not taken into account by the background estimator and they are considered as foreground. The aim of the proposed algorithm is to reduce illumination effects in video sequences in order to improve foreground segmentation performances. For that, we propose to compensate illumination in a zone-wise manner, by adapting their contrast and mean to canonical ones. We do this compensation at different resolutions (zone sizes).

  Publication:

  – [SCRH09] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Multi-resolution illumination compensation for foreground extraction", in 16th International Conference on Image Processing, 2009, pp. 3225-3228.*

- **Surface reconstruction by restricted and oriented propagation** : We address the problem of reconstructing a surface from a point cloud. More specifically, we propose a method which focuses on obtaining fast surface reconstructions for visual purposes. The proposed scheme is based on propagation in a voxelized space, which is performed in the directions defined by a propagation pattern, during an optimal number of iterations. Real-time applications are conceivable thanks to a low execution time and computational cost, keeping an acceptable visual quality of the reconstruction.

  Publication:

  – [SCRH10] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Surface reconstruction by restricted and oriented propagation", in 2010 IEEE International Conference on Image Processing, 2010, pp. 813-816.* **(Best Student Paper Award)**

- **From silhouettes to 3D points to mesh: towards free viewpoint video** In this proposal, we extend the previous work above on surface reconstruction. We propose a system for 3D reconstruction from video sequences acquired in multi-camera environments. In particular, the 3D surfaces of foreground objects in the scene are extracted and represented by polygon meshes. Three stages are concatenated to process multi-view data. First, a foreground segmentation method extracts silhouettes of objects of interest. Then, a 3D reconstruction strategy obtains a cloud of oriented points that lie on the surfaces of the objects of interest in a spatially bounded volume. Finally, a fast meshing algorithm provides a topologically correct interpolation of the surface points that can be used for both

visualization and further mesh processing purposes. The quality of the results (computational load) obtained by our system compares favorably against a baseline system built from state-of-the-art techniques for similar processing times (quality of the results).

Publication:

– [SSC10] *J. Salvador, X. Suau, and J. Casas, "From silhouettes to 3D points to mesh: towards free viewpoint video", in ACM Workshop on 3D Video Processing (3DVP), 2010, pp. 19-24.*

11.3 Summary of Contributions

11.3.1 Journal Articles

- [SRHC12b] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Real-time head and hand tracking based on 2.5D data", IEEE Transactions on Multimedia (TMM), vol. 14, no. 3, pp. 575-585 , 2012.*

- [SRhC13] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Detecting End-Effectors on 2.5D data using Geometric Deformable Models: Application to Human Pose Estimation", Computer Vision and Image Understanding (CVIU), vol. 117, no. 3, 2013.*

- [ASM+] *M. Alcoverro, X. Suau, J.R. Morros, A. López-Méndez, A. Gil, J. Ruiz-Hidalgo, J.R. Casas, "Gesture Control Interface for immersive panoramic displays". Multimedia Tools and Applications, Springer. 2013.*

* [SALM+] *X. Suau, M. Alcoverro, A. López-Méndez, J. Ruiz-Hidalgo, J.R. Casas, "Real-time Fingertip Localization Conditioned on Hand Gesture Classification", Submitted to Image and Vision Computing.*

11.3.2 Book Chapters

- [SALM+12] *X. Suau, M. Alcoverro, A. A. López-Méndez, J. Ruiz-Hidalgo, and J. Casas, "INTAIRACT: Joint Hand Gesture and Fingertip Classification for Touchless Interaction", in Computer Vision – ECCV 2012, vol. 7585, Heidelberg: Springer, 2012, pp. 602-606.*

11.3.3 Conference Papers

- [SCRH09] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Multi-resolution illumination compensation for foreground extraction", in 16th International Conference on Image Processing, 2009, pp. 3225-3228.*
- [SCRH10] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Surface reconstruction by restricted and oriented propagation", in 2010 IEEE International Conference on Image Processing, 2010, pp. 813-816.* **(Best Student Paper Award)**
- [SSC10] *J. Salvador, X. Suau, and J. Casas, "From silhouettes to 3D points to mesh: towards free viewpoint video", in ACM Workshop on 3D Video Processing (3DVP), 2010, pp. 19-24.*
- [SCRh11] *X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Real-time head and hand tracking based on 2.5D data", in IEEE International Conference on Multimedia and Expo (ICME), 2011, pp. 1–6.*
- [SRHC12a] *X. Suau, J. Ruiz-Hidalgo, and J. Casas, "Oriented radial distribution on depth data: Application to the detection of end-effectors", in IEEE International Conference on Acoustics, Speech, and Signal Processing, Kyoto, Japan, 2012.*

12

Discussion

The work carried out during this thesis has been focused on two main parts, (I) Description of Depth Data and (II) Hand Analysis, that are also the main parts of this document.

Regarding the description of depth data, we show that 3D descriptors are very effective at emphasizing specific characteristics of a depth point cloud. The proposed Oriented Radial Distribution feature proves to be a good alternative to detect end-effectors of depth point clouds, especially hands and feet (Chapter 3). Indeed, our ORD proposal behaves in a multi-resolution manner, which makes it very effective and flexible to detect end-effectors of a given size.

In Chapter 4 we propose to describe depth data using 3D connectivity properties, which leads to an alternative formulation of the geodesic distance adapting the Narrow Bands Level Set formulation (R-NBLS). We have shown that the proposed R-NBLS method is very effective in terms of computational cost. R-NBLS is combined with a shortest path algorithm that shows its performance at detecting end effectors (hand, feet and head). We make use of a very simple model for the human body, obtaining state-of-the-art detection results. These point out that generative methods strongly rely on the underlying data, so that, even with an extremely simple model, satisfying results are

achievable. The narrow bands obtained throughout the R-NBLS formulation prove to be very useful at feeding the body model.

With respect to the Hand Analysis part, in Chapters 8 and 9, we show that ORD is also able to globally describe a depth point cloud. More precisely, the objective in these chapters is to recognize various hand gestures. In order to do so, we apply classification strategies onto the ORD data, achieving satisfactory results in the hand gesture recognition task. Indeed, the proposed methods behave better than most of the reference works on a public ASL dataset.

In Chapter 8, we have proposed to localize fingertips using the recognized hand gesture as a prior, which helps reducing drastically the search space for fingertips. This so-called NNGM approach proves to be successful in various real-time demonstrations performed throughout this thesis. For this kind of applications, where the input data is a video sequence, we have proposed to exploit the temporal consistency of hand gestures to enhance both the gesture recognition and fingertip localization performance in about 10%.

In Chapter 9 we propose to invert the formulation of the fingertip localization problem. We tackle such task as it was a parts detection problem, which leads to a direct obtention of fingertip locations. The hand gesture is treated as an auxiliary property of the hand *object*. We have shown that the proposed Voting method obtains even better results than the NNGM method. Moreover, the Voting method may be generalized to any object represented by a point cloud, being able to detect its parts and estimate the posed auxiliary properties.

Both methods in Chapters 8 and 9 utilize the ColorTip dataset presented in Chapter 7, which is one of the main contributions of this thesis. ColorTip is a dataset for hand gesture recognition and fingertip localizations that can be used both for training purposes and evaluation. Given the recent irruption of depth cameras, there is a lack of public datasets, and, up to our knowledge, there does not exist any dataset providing the information ColorTip contains.

12.1 Future Work

Some computer vision problems have been addressed with satisfactory formulation and results. However, we foresee improvements in the various proposed methods, that we summarize here-after.

Including topological borders or frontiers is an interesting point for further work in the RNBLS formulation. Such borders could help to improve the propagation in order to better respect the topology. We envisage to use color information and other local descriptors to complement the depth estimation, which will help solving ambiguities and increase the detection rate. Also, physical magnitudes such as force fields may be incorporated to the RNBLS model, in order to slow down or speed-up propagation velocity depending on the confidence we have on a given part of the point cloud.

We have exploited the Oriented Radial Distribution using a pixel-wise value. But indeed, the ORD formulation involves the computation of a radial histogram of distribution, which could be exploited for feature matching for example. Using the histogram could also be extremely helpful for hand gesture recognition (in the NNGM and Voting methods, for example), at the expense of more processing power and time. Extending ORD to color images is another foreseen issue This formulation would drastically shift the paradigm, since the depth-based ORD is based on Euclidean distances that are meaningless on the RGB space.

The baseline HandBox method could be improved by re-enforcing the head detection algorithm. Strong head tilt should be handled, as well as leaning positions. Moreover, the hand detection heuristics may be improved using a discriminative approach like Random Forests. These, could be helpful to find hands if trained with a large amount of hand poses and using a relatively low acceptance threshold. Some experiments have been carried out in the last project FascinatE [Fas] demonstration substantiating this idea.

With respect to the NNGM approach, we have proposed to re-sample the ORD output to an $m \times m$ grid, in order to enable further comparison using usual distances. We foresee using various resolutions simultaneously (various grid sizes), implementing a learning method able to decide which resolutions are better suited to detect hand gestures depending on the feature output. This way, the classification result should be improved compared to using a fixed arbitrary resolution. This strategy could also be included in the Voting framework.

A

Preliminary Concepts on Depth Data

A.1 Apparent and Physical area on Depth Data

Generally speaking, a real world object is seen as a 2D area on the image plane when recorded with a camera. Such area, called apparent area $\tilde{A}$, depends on the distance between the object and the camera.

Since depth data is obtained from a single viewpoint, the concept of apparent area arises. Depth estimates are obtained in pixels, organized as images. If a surface S with a physical area A is captured by the depth sensor, its area A has to be translated into an area on the image plane, or apparent area $\tilde{A}$, which is expressed in pixels. Therefore, the size of $\tilde{A}$ will vary depending on the distance d between the camera and the recorded surface, which makes it impossible to recover A from the image without knowing the magnitude d.

Such inconsistency may be straightforwardly resolved in the case of depth data, since d is the estimated depth d_i for every pixel. A physical pixel-wise area A_i^{pix} may be assigned to a given pixel pix_i with depth d_i. Such assignment depends on the estimated depth d_i and the optical behavior of the camera.

An empirical law Γ^C is experimentally obtained for a given camera by means of recording a surface of a known physical area at different depth positions (Figure A.1).

The relationship between physical and apparent area, for a given pixel, which increases quadratically with depth, is formulated in Equation (A.1), providing a valid approximation for reasonable depth levels $d \in (80, 430)\, cm$. Equation (A.1) is the quadratic approximation of the samples in Figure A.1. Given the empirical nature of the approach, the camera parameters are not needed, so the approximation may be done without calibration.

$$A_i^{pix} = \Gamma^C(d_i) \approx 1.12 \cdot 10^{-6} \cdot d_i^2 + 8.41 \cdot 10^{-5} \cdot d_i - 4.64 \cdot 10^{-3} \tag{A.1}$$

Let S be a surface with an apparent area $\tilde{A}^S$. If S is sufficiently perpendicular to the camera, its physical area A^S may be approximated as shown in Equation (A.2).

$$A^S = \sum_{\forall pix_i \in S} \Gamma^C(d_i) \tag{A.2}$$

The depth of each pixel d_i may be replaced by the mean depth d^S of the observed region, simplifying the physical area calculation using Equation (A.3). The result finally depends on the number of points (or pixels) N^S in the area.

$$A^S \approx \sum_{\forall ppix_i \in S} \Gamma^C(d^S) = N^S \cdot d^S \tag{A.3}$$

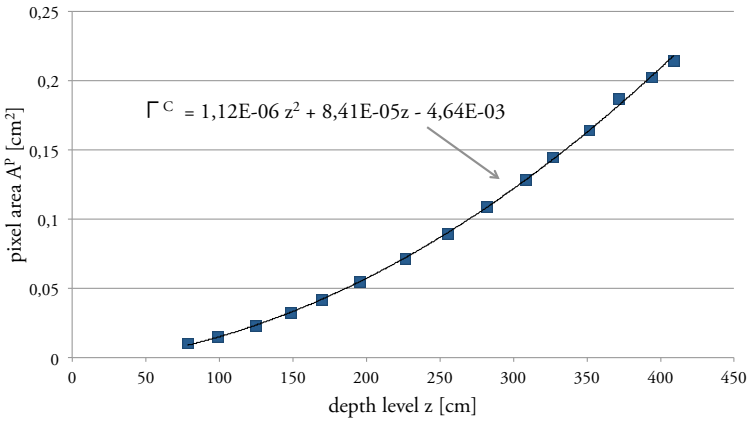

FIGURA A.1: Empirical estimation of the law Γ^C which gives the actual size of a pixel at a given depth level. Measurements have been carried out with a known flat surface (din A2 paper sheet) at various depth levels. The blue points are the division of the physical area of the paper sheet by the number of pixels it occupies on the image, which is equal to the physical area per pixel at that depth level. A quadratic approximation is shown in the figure.

A.2 (λ, ρ)-connectivity

In order to find connected regions in the point cloud, a connectivity condition should be defined. We state that a point p is (λ, ρ)-connected if the number of points in a ball of radius ρ centered at p is greater than λ. Thus, a region will be (λ, ρ)-connected if all its points are (λ, ρ)-connected too.

B

Description of the Random Forest baseline used to evaluate the fingertip localization accuracy

B.1 RF Fingertip baseline

In order to evaluate the proposed algorithms, we implement a fingertip localization method using Random Forests (RF) [Bre01]. The RF localization method is based on the successful system for detecting body parts from range data proposed by Shotton et al. [SFC+11]. We use very similar depth-invariant features, but in addition to depth data, we include the ORD feature.

Specifically, for an input patch π and a given pixel $\mathbf{p}$, the binary test has the following expression:

$$f(\pi, \mathbf{p}) = \phi_{\pi,c}\left(\mathbf{x} + \frac{\mathbf{m}}{d_\pi(\mathbf{p})}\right) - \phi_{\pi,c}\left(\mathbf{x} + \frac{\mathbf{n}}{d_\pi(\mathbf{p})}\right) \tag{B.1}$$

where d_π is the depth map associated to input patch π, $\phi_{\pi,c}$ denotes the c-th feature computed from π, and $\mathbf{m}$ and $\mathbf{n}$ are two randomly generated pixel displacements that fall within a patch size. Pixel displacements are normalized with the depth evaluated at pixel $\mathbf{p}$ in order to make the test features invariant to depth changes. In our implementation, the depth data is allocated in $\phi_{\pi,0}$ and the ORD responses in $\phi_{\pi,1}$.

We estimate the finger localization pseudo-probability density by means of a Parzen estimator with Gaussian kernel K:

$$F(l|\pi) = \sum_i p(l|\pi, \mathbf{p}_i) K(\mathbf{p} - \mathbf{p}_i) \tag{B.2}$$

Finally, we compute the l-th finger location $\mathbf{p}_l$ as the pixel location with maximum probability:

$$\mathbf{p}_l = \underset{\mathbf{p}}{\operatorname{argmax}} \quad F(l|\pi) \tag{B.3}$$

we ensure that this maximum represents a target gesture by thresholding the probability volume V computed by locally integrating the estimated pseudo-probability measure :

$$V = \sum_{\mathbf{p} \in \mathcal{S}} F(l|\pi(\mathbf{p})) \tag{B.4}$$

where $\mathcal{S}$ is a circular surface element of radius inversely proportional to the depth, and centered at the l-th finger maximum, i.e:

$$\mathcal{S} = \{\mathbf{n} \quad | \quad \|\mathbf{n} - \mathbf{p}_l\| < r(d_\pi(\mathbf{p}_l))\} \tag{B.5}$$

In this way, the fingertip localization is depth-invariant.

B.1.0.1 Benchmark

As explained in Section 10.4.1.6, we employ RFs comprising 10 trees of maximum depth 15. Three baselines are trained: one using depth information exclusively, another using ORD exclusively and a baseline combining both features. We test different parameters, and we keep the ones providing best accuracy in each of the 3 presented cases.

Bibliografia

[AC12] López-Méndez Adolfo and Josep R. Casas. Can our TV robustly understand human gestures? Real-time Gesture Localization on Range Data. In *CVMP*, pages 18–25, 2012.

[AOV12] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK : Fast Retina Keypoint. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2012.

[App12] Apple Inc. Magic Trackpad, 2012.

[AS95] D Adalsteinsson and J Sethian. A Fast Level Set Method for Propagating Interfaces. *Journal of Computational Physics*, 118(2):269–277, 1995.

[ASL] ASL Finger Spelling Dataset. http://info.ee.surrey.ac.uk/Personal/N.Pugeault/.

[ASM+] Marcel Alcoverro, Xavier Suau, Josep Ramon Morros, Adolfo López-Méndez, Albert Gil, Javier Ruiz-hidalgo, and Josep R. Casas. Gesture Control Interface for immersive panoramic displays. *Springer Multimedia Tools and Applications*.

[ATRV12] Aitor Aldoma, Federico Tombari, Radu Bogdan Rusu, and Markus Vincze. OUR-CVFH ,Äì Oriented, Unique and Repeatable Clustered Viewpoint Feature Histogram for Object Recognition and 6DOF Pose Estimation. *Lecture Notes in Computer Science*, 7476:113–122, 2012.

[BETV08] H Bay, A Ess, T Tuytelaars, and L Vangool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[BFH09] Marcus a. Brubaker, David J. Fleet, and Aaron Hertzmann. Physics-Based Person Tracking Using the Anthropomorphic Walker. *International Journal of Computer Vision*, 87(1-2):140–155, August 2009.

[BHMB09] Martin Bohme, Martin Haker, Thomas Martinetz, and Erhardt Barth. Head tracking with combined face and nose detection. *2009 International Symposium on Signals Circuits and Systems*, (July):1–4, 2009.

[BM09] Lubomir Bourdev and Jitendra Malik. Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations. In *International Conference on Computer Vision (ICCV)*, 2009.

[BMB+11] Andreas Baak, M Meinard, Gaurav Bharaj, Hans-peter Seidel, Christian Theobalt, and M P I Informatik. A Data-Driven Approach for Real-Time Full Body Pose Reconstruction from a Depth Camera. In *ICCV*. IEEE, 2011.

[Bre01] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[BSA06] Alessandro Bevilacqua, Luigi Stefano, and Pietro Azzari. People Tracking Using a Time-of-Flight Depth Sensor. *2006 IEEE International Conference on Video and Signal Based Surveillance*, pages 89–89, 2006.

[BTG+12] Luca Ballan, Aparna Taneja, Jürgen Gall, Luc Van Gool, and Marc Pollefeys. Motion Capture of Hands in Action Using Discriminative Salient Points. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Lecture Notes in Computer Science Computer Vision ,Äì ECCV 2012*, pages 640–653. Springer Berlin Heidelberg, 2012.

[BTV06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. *Computer Vision ,ÄìECCV 2006*, 3951(3):404–417, 2006.

[CCCD93] V Caselles, F Catté, T Coll, and F Dibos. A geometric model for active contours in image processing. *Numerische Mathematik*, 66(1):1–31, 1993.

[Clu] Leap Motion Clubic Review. http://www.clubic.com/technologies-d-avenir/article-575170-1-leap-motion-test.html.

[CMG+09] Stefano Corazza, Lars Mündermann, Emiliano Gambaretto, Giancarlo Ferrigno, and Thomas P. Andriacchi. Markerless Motion Capture through Visual Hull, Articulated ICP and Subject Specific Model Generation. *International Journal of Computer Vision*, 87(1-2):156–169, September 2009.

[CMMM08] Pedro Correa, Ferran Marqués, Xavier Marichal, and Benoit Macq. 3D posture estimation using geodesic distance maps. *Multimedia Tools and Applications*, 38(3):365–384, 2008.

[Col] ColorTip Dataset. https://imatge.upc.edu/web/?q=res/colortip.

[Fas] FascinatE Project. European Unions Seventh Framework Programme (FP7/2007-2013) Under Grant Agreement No U248138 (www.fascinate-project.eu).

[FBF77] Jerome H Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An Algorithm for Finding Best Matches in Logarithmic Expected Time. *Trans. on Mathematic Software*, 3(3):209–226, 1977.

[FGMR10] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 32(9):1627–1645, 2010.

[FHK+04] Andrea Frome, Daniel Huber, Ravi Kolluri, T. Bülow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *ECCV*, volume 1, pages 224–237, 2004.

[GKK07] Daniel Grest, Volker Krüger, and Reinhard Koch. Single view motion tracking by depth and silhouette information. *Lecture Notes in Computer Science*, 4522:719–729, 2007.

[GPKT10a] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real Time Motion Capture Using a Single Time-Of-Flight Camera. In *CVPR*, pages 755–762, 2010.

[GPKT10b] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real Time Motion Capture Using a Single Time-Of-Flight Camera. In *International Conference on Computer Vision and Pattern Recognition*, pages 755–762. IEEE, 2010.

[GPKT12] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real-Time Human Pose Tracking from Range Data. In *ECCV*, pages 738–751. Springer, 2012.

[GRBS08] Juergen Gall, Bodo Rosenhahn, Thomas Brox, and Hans-Peter Seidel. Optimization and Filtering for Human Motion Capture. *IJCV*, 87(1-2):75–92, 2008.

[GSD+09] J. Gall, C. Stoll, E. De Aguiar, C. Theobalt, B. Rosenhahn, and H.-P. Seidel. Motion capture using joint skeleton tracking and surface estimation. In *CVPR*, pages 1746–1753. IEEE, June 2009.

[GSK+11] Ross Girshick, Jamie Shotton, Pushmeet Kohli, Antonio Criminisi, and Andrew Fitzgibbon. Efficient regression of general-activity human poses

from depth images. In *CVPR*, pages 415–422. Microsoft Research Cambridge, USA, IEEE, 2011.

[GWBB09] Peng Guan, Alexander Weiss, A.O. Balan, and M.J. Black. Estimating human shape and pose from a single image. In *ICCV*, pages 1381–1388. IEEE, 2009.

[GYR+11] Juergen Gall, Angela Yao, Nima Razavi, Luc Van Gool, and Victor Lempitsky. Hough Forests for Object Detection, Tracking, and Action Recognition. *TPAMI*, 33(11):2188–2202, 2011.

[HAR+10] Nils Hasler, Hanno Ackermann, Bodo Rosenhahn, T. Thormahlen, and H.P. Seidel. Multilinear pose and body shape estimation of dressed subjects from image sets. In *CVPR*, pages 1823–1830. IEEE, 2010.

[HBMB07] Martin Haker, Martin Bohme, Thomas Martinetz, and Erhardt Barth. Geometric Invariants for Facial Feature Tracking with 3D TOF Cameras. *Circuits and Systems ISSCS 2007*, (July):3–6, 2007.

[HMB11] Georg Hackenberg, Rod McCall, and Wolfgang Broll. Lightweight Palm and Finger Tracking for Real-Time 3D Gesture Control. In *VR*, number March 2010, pages 19–26. IEEE, 2011.

[HXP03] X. Han, C. Xu, and J.L. Prince. A topology preserving level set method for geometric deformable models. *TPAMI*, 25(6):755–768, June 2003.

[KKKA11] Cem Keskin, Furkan Kirac, Yunus Emre Kara, and Lale Akarun. Real Time Hand Pose Estimation using Depth Sensors. In *ICCV-CDC4CV*, pages 1228–1234, 2011.

[KKKA12a] Cem Keskin, Furkan Kirac, Yunus Emre Kara, and Laie Akarun. Randomized decision forests for static and dynamic hand shape classification. In *2012 IEEE CVPR Workshops*, pages 31–36. IEEE, 2012.

[KKKA12b] Cem Keskin, Furkan Kiraç, Yunus Emre Kara, and Lale Akarun. Hand Pose Estimation and Hand Shape Classification Using Multi-layered Randomized Decision Forests. In *ECCV*, pages 852–863. Springer, 2012.

[KPHB08] Eva Kollorz, Jochen Penne, Joachim Hornegger, and Alexander Barke. Gesture recognition with a Time-Of-Flight camera. *Intl. J. of Intelligent Syst. Tech. and Applications*, (3/4):334, 2008.

[Kuo02] Arthur D. Kuo. Energetics of Actively Powered Locomotion Using the Simplest Walking Model. *Journal of Biomechanical Engineering*, 124(1):113, 2002.

[KVD06] S. Knoop, S. Vacek, and R. Dillmann. Sensor fusion for 3D human body tracking with an articulated 3D body model. In *ICRA*, number May, pages 1686–1691. IEEE, 2006.

[LCS11] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. BRISK: Binary Robust invariant scalable keypoints, 2011.

[Lea12] Leap. http://www.leapmotion.com, 2012.

[LF04] Xia Liu Xia Liu and K Fujimura. Hand gesture recognition using depth data, 2004.

[LKAR10] Nicolas H Lehment, Moritz Kaiser, Dejan Arsic, and Gerhard Rigoll. Cue-Independent Extending Inverse Kinematics For Robust Pose Estimation in 3D Point Clouds. In *ICIP*, pages 2465–2468, 2010.

[Low04] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[McG82] James J McGregor. Backtrack search algorithms and the maximal common subgraph problem. *Software: Practice and Experience*, 2(1):23–34, 1982.

[MD09] Marius Muja and Lowe G David. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. In *VISAPP*, 2009.

[MHK06] T B Moeslund, A Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90–126, 2006.

[Mic] Microsoft Corporation. Kinect for Xbox 360. http://www.xbox.com/en-US/kinect/default.htm.

[MIT11] MIT Finger Detection Demo. http://www.ros.org/wiki/mit-ros-pkg/KinectDemos/FingerDetection, 2011.

[MIT13] MIT. http://www.technologyreview.com/news/517331/look-before-you-leap-motion/, 2013.

[MM99] Songrit Maneewongvatana and David M. Mount. It,Äôs okay to be skinny, if your friends are fat. In *4th Annual CGC Workshop on Comptutational Geometry*, number October, pages 1–8, 1999.

[MM09] M. Maška and Pavel Matula. A fast level set-like algorithm with topology preserving constraint. In *Intl. Conf. on Computer Analysis of Images and Patterns*, pages 930–938. Springer, 2009.

[MN06] Zhenyao Mo and Ulrich Neumann. Real-time Hand Pose Recognition Using Low-Resolution Depth Images. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 2 CVPR06*, 2(c):1499–1505, 2006.

[MS05] Sotiris Malassiotis and Michael G Strintzis. Robust real-time 3D head pose estimation from range data. *Pattern Recognition*, 38(8):1153–1165, August 2005.

[MS08] S Malassiotis and M Strintzis. Real-time hand posture recognition using range data. *Image and Vision Computing*, 26(7):1027–1037, 2008.

[MSV95] R Malladi, J A Sethian, and B C Vemuri. Shape modeling with front propagation: a level set approach. *TPAMI*, 17(2):158–175, 1995.

[MZC12] C. Dal Mutto, P. Zanuttigh, and G.M. Cortelazzo. *Time-of-flight Cameras and Microsoft Kinect*. Springer, New York, springerbr edition, 2012.

[NB10] Marco Nichau and Volker Blanz. Pose-insensitive nose detection in TOF-scans. *Computer Vision and Image Understanding*, 114(12):1346–1352, December 2010.

[OKA11] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. Efficient Model-based 3D Tracking of Hand Articulations using Kinect. In *BMVC*, 2011.

[PB11] Nicolas Pugeault and Richard Bowden. Spelling It Out: Real,ÄìTime ASL Fingerspelling Recognition. In *ICCV-CDC4CV*, 2011.

[PGKT10] Christian Plagemann, Varun Ganapathi, Daphne Koller, and Sebastian Thrun. Real-time identification and localization of body parts from depth images. In *ICRA*, pages 3108–3113. IEEE, 2010.

[PMBH+10] G. Pons-Moll, Andreas Baak, Thomas Helten, M. Muller, H.P. Seidel, and Bodo Rosenhahn. Multisensor-fusion for 3d full-body human motion capture. *Elements*, pages 2–9, 2010.

[Pop07] Ronald Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108:4–18, 2007.

[Pri11] PrimeSense. OpenNI, 2011.

[Pri13] PrimeSense. http://www.engadget.com/2013/05/15/primesense-demonstrates-capri-3d-sensor/, 2013.

[RBTH10] R.B. Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *IROS*, pages 2155–2162, 2010.

[RD06] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV*, 2006.

[Rel13] Leap Motion Release. https://www.leapmotion.com/press_releases/leap-motion-launches-world-s-most-accurate-3-d-motion-control-technology-for-computing, 2013.

[RHBB09] Radu Bogdan Rusu, Andreas Holzbach, Michael Beetz, and Gary Bradski. Detecting and segmenting objects for mobile manipulation. In *S3DV-ICCV*, volume 71, pages 47–54, 2009.

[RHMA+11] J Ruiz-Hidalgo, J R Morros, P Aflaki, F Calderero, and F Marqués. Multiview depth coding based on combined color/depth segmentation. *Journal of visual communication and image representation*, 23(1):42{\textendash}52, 2011.

[RML10] Paul Rosenthal, Vladimir Molchanov, and Lars Linsen. A Narrow Band Level Set Method for Surface Extraction from Unstructured Point-based Volume Data. In *Intl. Conf. on Computer Graphics Visualization and Computer Vision*, pages 73–80, 2010.

[RRKB11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF, 2011.

[Rus09] Radu Bogdan Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments.* PhD thesis, Technische Universitaet Muenchen, 2009.

[RYZ11] Zhou Ren, Junsong Yuan, and Zhengyou Zhang. Robust Hand Gesture Recognition Based on Finger- Earth Mover,Äôs Distance with a Commodity Depth Camera. In *ACM-MM*, pages 1093–1096, 2011.

[SALM+] Xavier Suau, Marcel Alcoverro, Adolfo López-Méndez, Javier Ruiz-hidalgo, and Josep R. Casas. Real-time Fingertip Localization Conditioned on Hand Gesture Classification. *SIEEE ystems, Man and Cybernetics Part B.*

[SALM+12] Xavier Suau, Marcel Alcoverro, Adolfo López-Méndez, Javier Ruiz-Hidalgo, and Josep R. Casas. INTAIRACT: Joint Hand Gesture and Fingertip Classification for Touchless Interaction. *LNCS-ECCV*, 7585:602–606, 2012.

[SC09] Aravind Sundaresan and Rama Chellappa. Multicamera tracking of articulated human motion using shape and motion cues. *IEEE Transactions on Image Processing*, 18(9):2114–2126, 2009.

[Sch90] JC Schmitt. *La raison des gestes dans l,ÄôOccident mediéval*. Gallimard, Paris, 1990.

[SCO11] Joseph Schlecht, Bernd Carque, and Bjorn Ommer. Detecting gestures in medieval images. In *ICIP*, pages 1285–1288. IEEE, September 2011.

[SCRH09] Xavier Suau, Josep R Casas, and Javier Ruiz-Hidalgo. Multi-resolution illumination compensation for foreground extraction. In *IEEE International Conference on Image Processing*, pages 3225–3228, 2009.

[SCRH10] Xavier Suau, Josep R Casas, and Javier Ruiz-Hidalgo. Surface Reconstruction by Restricted and Oriented Propagation. In *IEEE International Conference on Image Processing*, pages 813–816, Hong Kong, 2010.

[SCRh11] Xavier Suau, Josep R Casas, and Javier Ruiz-hidalgo. Real-Time Head and Hand Tracking based on 2.5D data. In *ICME*, pages 1–6, Barcelona, 2011. IEEE.

[Set99] J A Sethian. *Level Set Methods and Fast Marching Methods*, volume 39 of *Cambridge Monograph on Applied and Computational Mathematics*. Cambridge University Press, 1999.

[SFC+11] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-Time Human Pose Recognition in Parts from Single Depth Images. In *CVPR*, pages 1297–1304, 2011.

[SG00] P Salembier and L Garrido. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *TIP*, 9(4):561–576, 2000.

[SGF+12] Jamie Shotton, Ross Girshick, Andrew Fitzgibbon, Toby Sharp, Mat Cook, Mark Finocchio, Richard Moore, Pushmeet Kohli, Antonio Criminisi, Alex Kipman, and Andrew Blake. Efficient Human Pose Estimation from Single Depth Images. *TPAMI*, 2012.

[SHG+11] Carsten Stoll, Nils Hasler, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt. Fast articulated motion tracking using a sums of Gaussians body model. In *ICCV*, volume 35, pages 951–958. MPI Informatik, Germany, IEEE, 2011.

[SM10] M Siddiqui and G Medioni. Human pose estimation from a single view point, real-time range sensor. In *CVPRW*, number August, pages 1–8. Ieee, 2010.

[SMMN11] Loren Arthur Schwarz, Artashes Mkhitaryan, Diana Mateus, and Nassir Navab. Human skeleton tracking from depth data using geodesic distances and optical flow. *Image and Vision Computing*, 2011.

[SPHK08] Stefan Soutschek, Jochen Penne, Joachim Hornegger, and Johannes Kornhuber. 3-D gesture-based scene navigation in medical imaging applications using Time-of-Flight cameras. In *CVPRW*, volume 1-3, pages 1–6. IEEE, 2008.

[SRHC12a] Xavier Suau, Javier Ruiz-Hidalgo, and Josep R Casas. Oriented Radial Distribution on Depth Data: Application to the Detection of End-Effectors. In *ICASSP*, 2012.

[SRHC12b] Xavier Suau, Javier Ruiz-Hidalgo, and Josep R. Casas. Real-Time Head and Hand Tracking based on 2.5D data. *IEEE Transactions on Multimedia*, (99):1, 2012.

[SRhC13] Xavier Suau, Javier Ruiz-hidalgo, and Josep R. Casas. Detecting End-Effectors on 2.5D data using Geometric Deformable Models: Application to Human Pose Estimation. *Computer Vision and Image Understanding (CVIU)*, 117(3), 2013.

[SSC10] Jordi Salvador, Xavier Suau, and Josep R Casas. From Silhouettes to 3D Points to Mesh: Towards Free Viewpoint Video. In *ACM Multimedia Workshop on 3D Video Processing*, pages 19–24, Firenze, 2010.

[tofcM] miniature 3D time-of-flight camera Mesa Imaging AG, SwissRanger SR4000. http://www.mesa-imaging.ch/swissranger4000.php.

[TSD10] Federico Tombari, Samuele Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, pages 356–369, 2010.

[TSSF12] Jonathan Taylor, Jamie Shotton, Toby Sharp, and Andrew W. Fitzgibbon. The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, pages 103–110. IEEE, 2012.

[UGVV11] Dominique Uebersax, Juergen Gall, M. Van den Bergh, and L. Van Gool. Real-time Sign Language Letter and Word Recognition from Depth Data. In *ICCV-HCI*, pages 1–8, 2011.

[VKMBV09] M Van Den Bergh, E Koller-Meier, F Bosche, and L Van Gool. Haarlet-based hand gesture recognition for 3D interaction. In *WACV*, pages 1–8. IEEE, 2009.

[VKMV09] Michael Van den Bergh, Esther Koller-Meier, and Luc Van Gool. Real-Time Body Pose Recognition Using 2D or 3D Haarlets. *IJCV*, 83(1):72–84, 2009.

[VV11] Michael Van den Berg and Luc Van Gool. Combining RGB and ToF Cameras for Real-time 3D Hand Gesture Interaction. In *WACV*, pages 66 – 72, 2011.

[WBMS01] R Whitaker, D Breen, K Museth, and N Soni. A framework for level set segmentation of volume datasets. In *International Workshop on Volume Graphics*, volume D, pages 159–68, 2001.

[WP09] Robert Y Wang and Jovan Popović. Real-time hand-tracking with a color glove. *ACM Transactions on Graphics*, 28(3):1, 2009.

[WVT12] Chenglei Wu, Kiran Varanasi, and Christian Theobalt. Full Body Performance Capture under Uncontrolled and Varying Illumination: A Shading-based Approach. In *ECCV*, pages 757–770. Springer, 2012.

[YP08] Jingyu Yan and Marc Pollefeys. A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video. *TPAMI*, 30(5):865–77, May 2008.

[ZDF08] Y. Zhu, Behzad Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. In *ICCV Workshops*, pages 1–8. IEEE, June 2008.

[ZSCL12] L. Zhang, J. Sturm, D. Cremers, and D. Lee. Real-time Human Motion Tracking using Multiple Depth Cameras. In *International Conference on Intelligent Robot Systems*, 2012.

[ZW12] Xiaolong Zhu and Kwan-Yee K. Wong. Single-Frame Hand Gesture Recognition Using Color and Depth Kernel Descriptors. In *ICPR*, pages 2989 – 2992, 2012.

[ZYT13] Chenyang Zhang, Xiaodong Yang, and Yingli Tian. Histogram of 3D Facets: A Characteristic Descriptor for Hand Gesture Recognition. In *FG*, 2013.