Universitat
Autònoma
de Barcelona

Master in Computer Vision and Artificial Intelligence
Report of the Master Project
Option: Computer Vision

Real-time Hand Pose Recognition using Depth Sensors combined with Spherical Shape Model Descriptor

Author: **Oscar Lopes**
Advisor: **Sérgio Escalera**
Co-Advisor: **Jordi Gonzàlez**

Acknowledgements

I would like to express my sincere gratitude to my both advisors Dr. Sergio Escalera and Dr. Jordi Gonzalez, for their great support in all aspects of development of this work, for their knowledgeable guidance, patience and their enthusiastic motivation.

Also I would like to thank Oblong Industries, specially Miguel Sanchez and Dan Chak, for their patience and support, during the stays at the Barcelona office, providing the opportunity to experience their state-of-art gesture recognition technologies, a simply unbelievable experience.

I would like to express my sincere gratitude to Obra Social Catalunya Caixa[1], for supporting me with a grant, that allowed me to pursuit this Master Course, and ultimately, perform this work.

Last but not least, I would like to thank my parents, for their unconditional support and motivation, during the difficult times of the development of this work. Thank you for always being there for me.

[1] http://obrasocial.catalunyacaixa.com

ABSTRACT

Hand pose recognition is a hard problem due to the inherent structure complexity of the hand, that can show a great variety of dynamic configurations and self occlusions. Some traditional approaches tackle this problem using RGB image information, which translates into a extremely hard problem given the ambiguities that arise from this representation. Depth sensor devices, namely the Kinect, have opened a new perspective for this problem since it deals with the 3D information of the object of interest, potentiating more robust and accurate approaches. In this work it is proposed a new light-weight spherical 3D descriptor model that encodes the information provided by the depth data sensor. The descriptor has been integrated in a selective hand data capture and classification pipeline capable of real-time hand pose recognition and successfully integrated with Oblong's pose recognition framework.

Keywords: *hand pose recognition, gesture, real-time, depth camera, spherical shape descriptor, support vector machines*

Contents

Chapter 1

Introduction

This master thesis was originated by the final graduation project of the Computer Vision Master and Artificial Intelligence Master organized by the Computer Vision Centre (CVC) Research Group of the Autonomous University of Barcelona. The project resulted from a collaboration of the CVC with Oblong Industries [1], a renowned company widely known for their state-of-art hand gesture recognition systems tailored for rich Human-Computer Interaction (HCI).

This chapter introduces some more aspects of the hand pose recognition problem. Moreover Section 1.1 discusses the problem description followed by research goal on Section 1.2. Lastly, it is provided the research method and outline of the overall thesis.

1.1 Problem Overview

In the recent years a great interest and progress in HCI, tailoring to create easier and more intuitive interfaces based in our innate communication skills have been performed, improving the overall user experience. When considering vision-based interfaces and interaction tools, among other body parts, the hand is the most effective general-purpose tool to perform non-contact interactions, due to its dexterity, both for communication and manipulation, enabling cursor control, rich navigation interfaces and dynamic set of gesture language. However, hand pose recognition represents a very hard task, associated to the natural complexity of the human hand structure and the high number possible configurations associated to its 26 degrees of freedom (DOF) [1] (Figure 1.1).

The main difficulties, according to [1], that surround a hand pose recognition system design include:

- High dimensionality problem: the hand is an highly articulated object with a high variability due to its great number of DOF. However, a human hand cannot express its theoretical number of

[1] http://oblong.com/

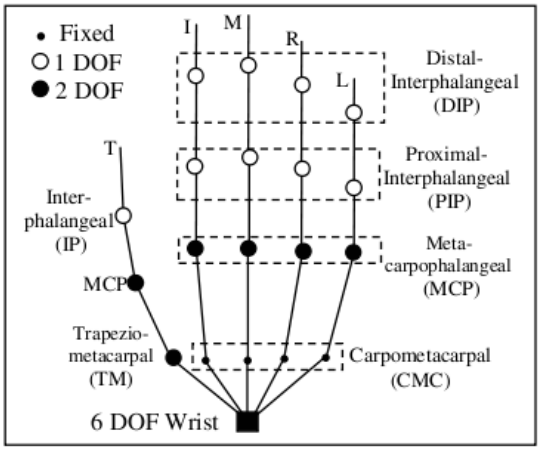


Figure 1.1: Hand Skeletal Model.

DOF, due to the interdependency constraints, it is not possible to use less than 6 dimension to model it.

- Self-occlusions: being the hand an articulated object, naturally some poses create occlusions that make harder to extract high level characterizing features.
- Uncontrolled environments: one of the major difficulties associated to an HCI system is the associated requirement of operating in non-controlled environments, such as, different backgrounds and broad range of lighting conditions.
- Hand motion velocity: human hand possesses a high motion and wrist rotation capabilities. This aspect combined introduces difficulties in the hand tracking and pose identification, even using depth devices capable of 30Hz frame rates, mainly because of the processing overhead imposed by algorithm. This implies that consecutive image depth frames become very uncorrelated when the hand motion speed increases.

Pose recognition, since the initial steps of Computer Vision, has sparkled a great interest on researchers, due to its potential applications on HCI, greatly enhancing user experience enabling the creation of command and control interfaces [2]. Object manipulation interfaces have attracted the interest of researchers, one of the seminal works in this area is the *"Put-that-there"* project from the MIT media labs [3], that enabled a user, via gesture recognition and speech commands, to control virtual elements over a canvas, allowing hand pointing functionality, selection and manipulation (Figure 1.2). The pointing functionality was obtained by means of a magnetic field, accordingly to the article [2]. It used two electronical devices: one acting as a transmitter and a second device working as a sensor. The transmitter radiates a magnetic field pointed at the sensor. When the pointing vector is correct, the strength of the received field is constant. When it is not, there will be an error signal consisting of the mutation frequency. This error is used to generate the output pointing angles, and to re-aim the

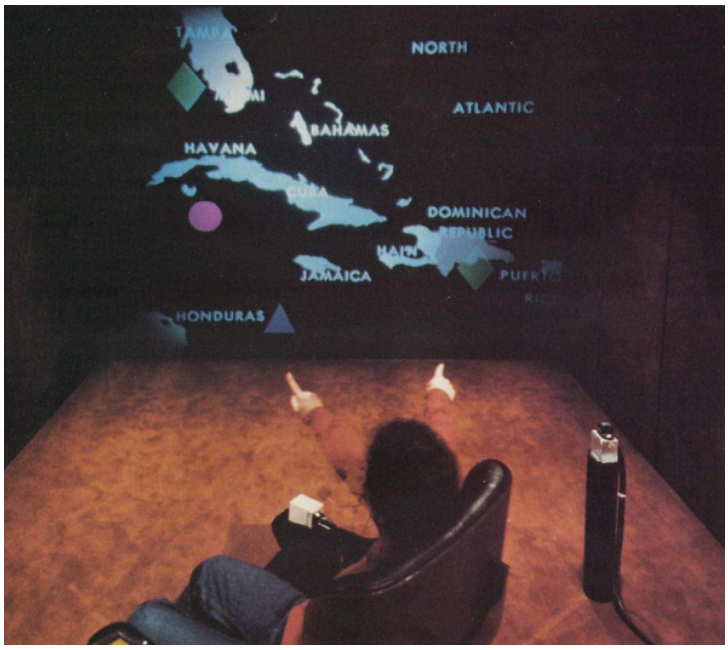


Figure 1.2: *Put-that-there* interaction system (1980).

transmitter.

Besides this major break-through, this system had the major drawback of limiting the mobility of the user, since the interaction was closely attached with the before mentioned magnetic sensors, plus the pointing functionality was not very precise. From today's standards this system may seem obsolete, but taking in account that was designed in 1980, when the computing power was no way near to what we have today, and computer graphics were on its beginnings, it is clear that these initial results were impressive.

Until recently, some of the best commercial hand pose recognition systems relied on glove based solutions [4], where one of the original projects in this field was carried out in MIT Media Labs, with the *Led Glove* and the *Data Glove* from VPL Research (Figure 1.3).

Currently Oblong Industries [5] has revolutionized this initial concept, taking it even further, creating a 3D hand gesture operating system, that enables the creation of rich HCI applications capable of millimetre precision pointing, virtual object manipulation, and navigation, handling massive amounts of data in real-time, in a totally distributed environment, or like it's creators prefer to call it - a *spatial operating environment* (Figure 1.4).

This system relies in a Motion Capture (MoCap) technology [6], usually used for full body tracking and virtual characters animation. In this case is used in a specially customized environment for hand pose recognition, using the marker gloves. The system relies on multiple Vicon [7] cameras (Figure 1.4(c)), that track both hand's position, and according to it's configuration pose, determined by infra-red tags placed on the fingertips, computes the hand pose.

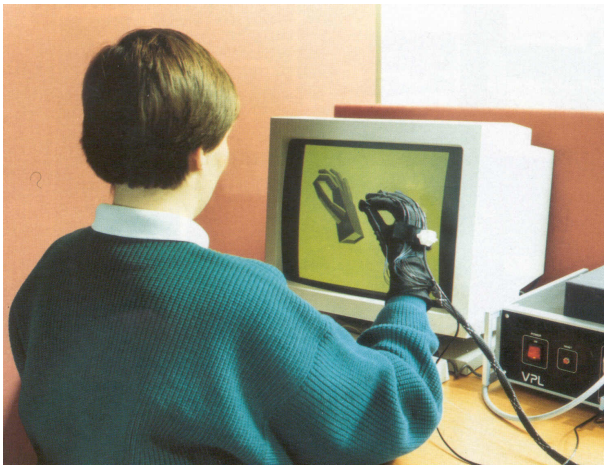
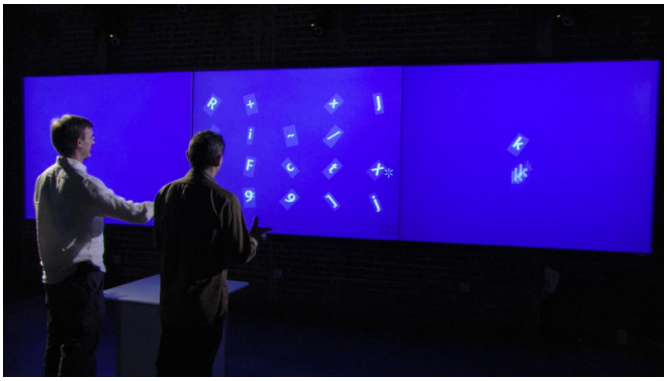


Figure 1.3: VPL Data Glove (1987).

More recently with appearance of depth sensors such as the Kinect™ (Figure 1.5), at an affordable price, a new spectrum of approaches were opened since it provided a new source of information that dealt more closely with the 3D nature of the objects of interest. This way it is now feasible to extract 3D information from the hand pose using a gloveless input source, and from then on, analyse and estimate the observed pose.

Computer hand gesture interaction opens a wide range of possibilities for innovative applications, since it provides a natural and intuitive language paradigm to interact with computer virtual objects, inspired in how humans interact with *real-world* objects. However, hand pose recognition imposes a extremely difficult problem, due to the highly dynamic range of configurations the hand can show, associated to the limitations of traditional approaches based solely on RGB data, since the 3D information of the scene is lost in its image representation. When depth sensors, namely the low-cost Microsoft® Kinect™ device, appeared in the market, a new perspective for these problems was open to be explored. One of the main features of this device consists in the capability of obtaining an aligned representation of depth and colour images, usually labelled as RGBD (RGB+Depth) data, at a pace of 30 fps (frames per second).

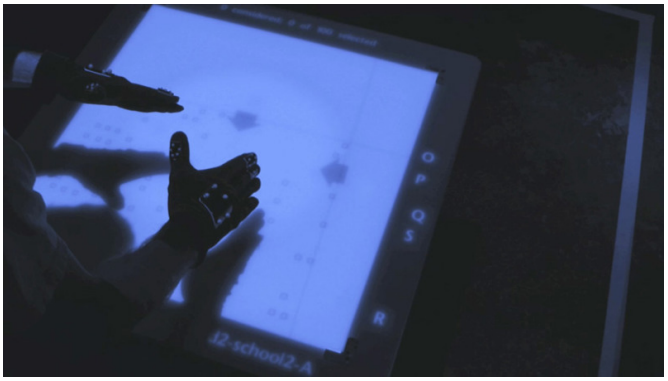
Basically the Kinect™ device it is based on the structured light technology, projecting an codified/structured matrix of points (Figure 1.6) of 300.000 points to the environment. The device consists on an infrared laser emiter, and infared camera and a RGB camera. The laser emitter projects a beam that is split by a diffraction grating, creating a constant dot pattern, then projected into the scene. The receiver then correlates the pattern observed with a reference pattern, that is stored in the device's memory and was obtained from a captured plane at a known distance from the sensor. This way, when a dot is project on an object positioned at a distance smaller or grater than the reference plane, the position of the dot in the infrared image is then shifted to the direction of the baseline between the laser projector and the perspective centre of the infrared camera. These shifts are correlated for all projected



(a) Multiple user gesture interaction.

(b) Marker gloves.

(c) Vicon camera.



(d) Close proximity gesture interaction.

Figure 1.4: Oblong HCI System.

dots creating a disparity matrix, from which, the real distance to the object can the be computed. Further details about Kinect underlying mathematical model can be found in [8].

Besides the Kinect™ device, there are other devices that include similar features such as, the Asus Xtion Pro™, which was released in 2010. This technology was developed and patented by Primesense [9, 10]. Commonly, this devices offer multi-modal capabilities combining RGBD + audio information, opening an even extensive functionalities, consolidating this field of research. Both devices offer similar features and capabilities. However, since the Kinect™ appeared first on the market, it is more widely supported by the Open Source community, offering better set of drivers across a wide range of supported operating systems.

In order to take the most of this technology several frameworks are available to developers, in a steady stream of API's and tools, providing a wider scope of applications that can be designed with depth sensor devices. In December 2010 OpenNI [11] released its own framework stack, comprising

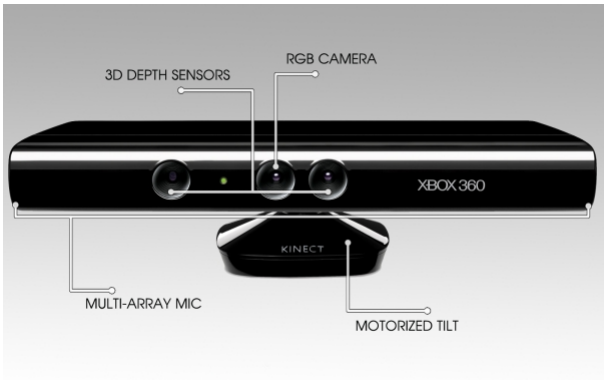


Figure 1.5: Kinect Depth Sensor.

Figure 1.6: Structured light pattern.

open source drivers and motion tracking middleware for PCs running Windows, Ubuntu and MacOSX. In June 2011 Microsoft® released a non-commercial Kinect™ software development kit (SDK) for windows, including Windows™ compatible drivers. As a natural consequence of the data representation provided by depth sensors in general, new libraries that process depth maps were developed, such as the Point Cloud Library (PCL) [12, 13], that underlying its core includes renowned libraries such as the Eigen[14] C++ template library for linear algebra and LibBoost [15] C++ general purpose portable source library, and obviously the OpenNI library, making PCL a very suitable development canvas, to create Kinect™ based applications.

1.2 Project Goals

The project goals are condensed in the following general statement:

> Create a robust gloveless hand pose recognition system, depth sensor based, capable of real-time performance to enhance HCI.

(a) Structured light pattern (b) Depth image (c) Point cloud

Figure 1.7: Depth Sensor Imaging.

To accomplish this general objective the following tasks must be accomplished:

- Create a hand detector based on depth sensor, to capture train and testing data.
- Collect a rich and broad hand pose dataset to train the overall system.
- Define a volumetric shape model descriptor, that takes the best use of the 3D data provided by the depth sensor, and capable of real-time processing.
- Create an end-to-end pipeline, using the previously defined tools, capable of real-time hand pose recognition.
- Integrate the developed system with Oblong's framework.

1.3 Research Method and Outline

In order to solve the previously defined goals and ultimately achieve create a viable solution to the stated problem, the following steps are taken:

- Perform a literature study on relevant concepts: Firstly a study on exiting literature on general Pose Recognition techniques must me performed, with special emphasis in those that can be transposed to the hand pose recognition problem. This should also take into account the performance of the algorithm in terms of recognition accuracy, training requirements and execution time. This aspects are presented in Chapter 2.
- Expose the solution's general architecture and its several components namely:
  - Hand tracking application: since this is the starting element of the solution's pipeline, it is crucial to define a set of methods capable of correctly segment both hands from the captured depth image provided by the Kinect™ device.

  - Define a broad hand pose dataset, with a rich set of poses, to perform the system training.
  - Define a hand description algorithm, that takes full advantage from the 3D information provided by the depth sensor. The design should also take into account the descriptor's performance since the final application must be able to process data in a real-time fashion.
  - Perform the system's training with the collected data.
  - Establish a global end-to-end pipeline, that using the before mentioned components, is capable of real-time hand pose recognition, using a live-feed from the depth sensor.

  All these aspects are fully explained in Chapter 3.

- Analyse the results obtained from the overall system design, assessing which is the best configuration setup, both performance and accuracy wise. These topics are exposed in Chapter 4.
- Establish some conclusions about the selected approaches, assessing if the obtained results have met requirements initially defined. Lastly plan some lines of future work to overcome the possible drawbacks of the system's design. This analysis is defined in Chapter 5.

Chapter 2

Related Work

In Section 1.1 are referenced, in a thirty years evolution time-frame, some HCI applications that have used successfully hand pose recognition techniques. However all the mentioned approaches in order to perform the pose recognition, rely either on external devices that limit the users mobility, or on marker glove solutions, which limits the field of application. This clearly implies that more requirements are imposed on the application's working environment, and this represents to some extent, a considerable drawback.

Other fields of research tried to tackle the pose recognition problem by using solely standard RGB cameras [16, 17, 18, 19] with a relative success. However, it is important to point out, that these approaches overall performance depend heavily on how well the environment is controlled, in terms of lighting conditions, position of the user in respect to the camera and background complexity. Plus, these techniques are only capable of recognizing a very limited set of hand poses, due to the difficulties imposed by the hand itself, like it was discussed in Chapter 1. Nonetheless, these techniques can be useful for some applications, but for what HCI is concerned, these serious limitations make it infeasible in practice.

Given the problem's complexity, in the past years, several works focused in circumventing some of the difficulties by using another technological approach, by the means of Time-of-Flight range cameras [20, 21, 22]. However other technologies are available to capture depth maps from the environment (Figure 2.1) which includes the structured light system of Microsoft®Kinect™.

The appearance of depth sensors like the Kinect™ have revolutionized the type of approaches that can be considered for the pose recognition problem in general, since the RGBD data provides the *missing link* for this problem - 3D geometric information of the interest object - removing the object's pose ambiguities that arise on the conventional RGB approaches.

Some techniques have also greatly benefited from the RGBD representation such as the reconstruction of dense surfaces and 3D object detection [23], augmented reality [24], improved descriptors for object recognition [25, 26], SLAM [27], and obviously human behavior analysis and pose recognition.

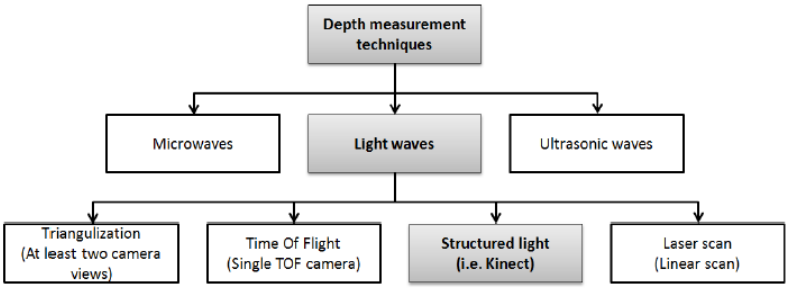


Figure 2.1: Structured light technologies.

An excellent overview about some of these technologies and techniques can be found in [28].

One of the most important contributions to pose recognition of the human body was provided by Microsoft® Research Team [29], proposing a method to quickly and accurately predict 3D positions of body joints from a single depth image (Figure 2.2). Due to the complexity of the task, they have designed intermediate body parts representation, converting the problem into a simple per-pixel classification, using Random Decision Forest (RDF) [30]. To train the system it was used a synthetic dataset composed of 1 million images. The final testing results of the system achieve high accuracy with a high performance, achieving 200 frames per second (fps). However it requires a huge number samples and the computing time required to train the overall system (24 hours on a 1000 core cluster) has to be taken into account.

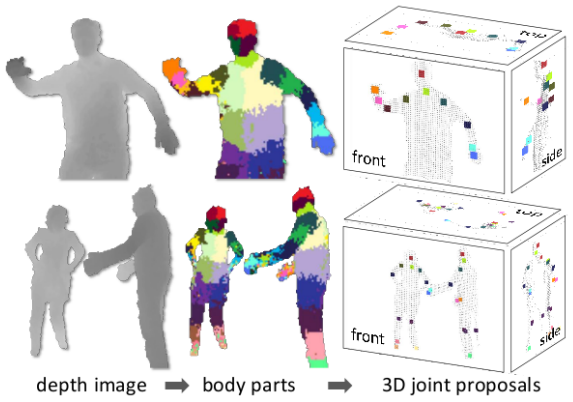
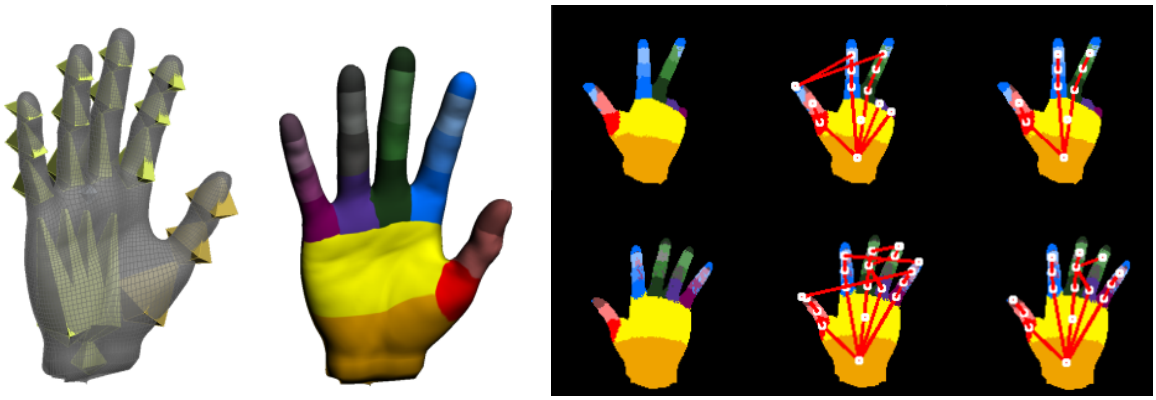


Figure 2.2: Body part recognition.

The idea previously mentioned was then extended to the hand pose recognition problem [31], using a synthetic pixel labelled dataset with 20.000 samples to train the system. They have also compared the accuracy results of RDF classifier versus a Support Vector Machine (SVM), showing that the later provided better results with approximately the same computational time (Figure 2.3).

Other lines of research also take advantage of the representation capabilities provided by depth



(a) Synthetic hand model (b) Hand joints classification

Figure 2.3: Hand pose.

image, however taking a slightly different approach. In [32], developed by Oblong Industries, three set of features are used in order to identify and characterize segment hand patches. The first feature set performs a global image statistics like the percentage of pixels that is covered by the blob contour, number of fingertips detected, the mean angle from the blob's centroid to those fingertips. A second descriptor is built from the number of pixels covered by every possible rectangle contained in the blob's bounding box and then normalized by its total size (Figure 2.4). A third feature set uses a similar grid as the second set however, instead of analysing the coverage within different rectangles, it is composed from the difference between the mean depth for each pair of individual cells (Figure 2.4). Finally a forth feature set is built by simply combining all of the features from the initial three sets.

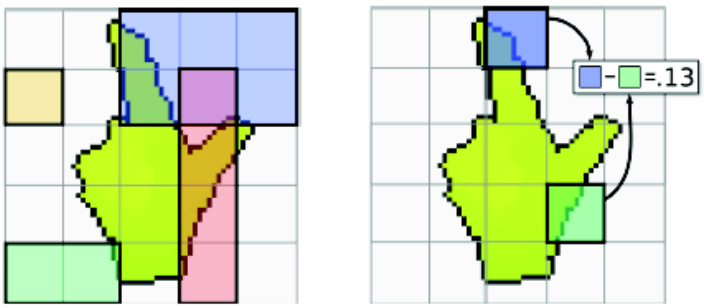


Figure 2.4: Statistical feature descriptor.

A total different approach is taken by [33] where it was considered a cylindrical shape model descriptor (Figure 2.5), that takes into account the geometric properties of the hand. This approach has the advantage that it can be trained considering the global hand shape samples without requiring further labelling. Nonetheless it still provides some fairly good accuracy results in recognizing the hand pose.

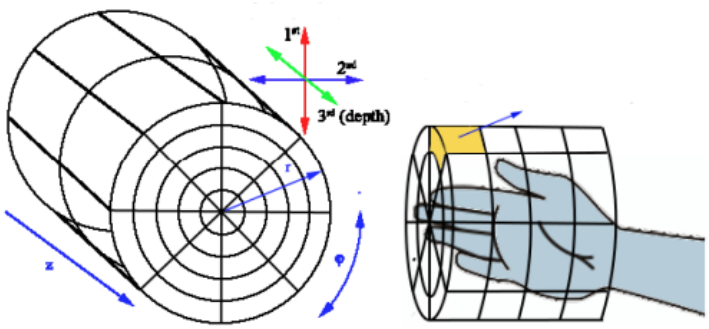


Figure 2.5: Cylinder shape model descriptor.

Like other approaches has some drawbacks, specially due to the way the descriptor was built, considering cylindrical sub-layers, which implies that for some similar poses, is not discriminative enough, as the results confirm. However it provides some ideas that can be explored in order to solve this problem, as it will be explained in Chapter 3.

Given the classical pipeline of hand pose recognition: detection, description and classification, it is clear that the main *bottleneck* to achieve a successful pose recognition heavily depends on the establishment of a highly discriminative descriptor with a low computational overhead. Although some new 3D descriptor have been developed, that complement the before mentioned (plus the Point Feature Histogram (PFH) among other available on the PCL, most of them based on the 3D gradient orientation distribution) the design of a new descriptor is necessary. Due to this fact, one of the main contributions of this work is the proposal of a new generic shape descriptor, fast to compute and highly discriminative, that can be used in conjunction with any state-of-art classifier like Adaboost, SVM or RDF.

Chapter 3

System Design

The developed system in order to accomplish the goals proposed in Section 1.2 is here explained in depth, and all of its components. A fully modular structure was followed, in order to make the system more customizable, manageable and adaptable to different set-ups and configurations.

3.1 System Overview

This section provides a major overview of the system's composition. It is built by three major modules namely, hand tracking and segmentation, hand point cloud description using the novel Spherical Blurred Shape Model Descriptor (SBSMD), and classification using Support Vector Machine library (libSVM). The system was trained with hand pose dataset specially crafted for this work, comprising over 12000 samples of hand poses, with high intra-class variability in terms of hand orientation.

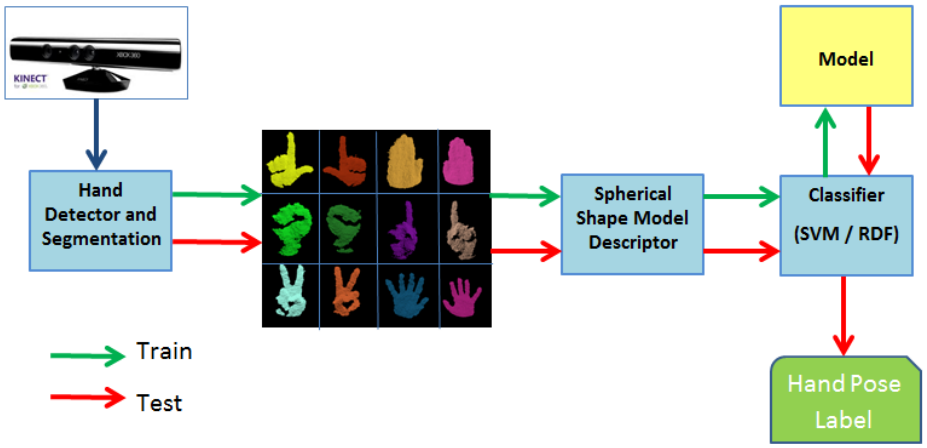


Figure 3.1: System architecture.

3.2 Hand Tracker

In order to achieve some a good overall pipeline recognition performance it is crucial to perform a reliable point cloud hand segmentation. The Kinect™ has the capability to capture depth data from 50cm to 3.5m. However, in the case of the hand, since it is a very small object, it is unreliable to identify its pose at a great distance since in this case the number of captured hand cloud points would be very small. Some approaches define a threshold zone distance where the object of interest is searched. In this approach, the criteria followed is the minimum number of point that surround the closest point towards the depth camera.

In a nutshell, the algorithm can be explained by the following way: first we try to find the closest point to sensor. Then from that point, perform a radius search of 10cm over the nearby points, if we have less than 100 points, probably we are observing noise. Otherwise we compute the centroid of these points, and with this we have found the first hand. Now we repeat the previous mentioned procedure to try to find the second hand under the following restrictions: the distance to camera must be within 30 cm of the first hand's closest distance and must more than 20 cm from the centroid of the first hand. An example of the hand tracker is provided in Figure 3.2.

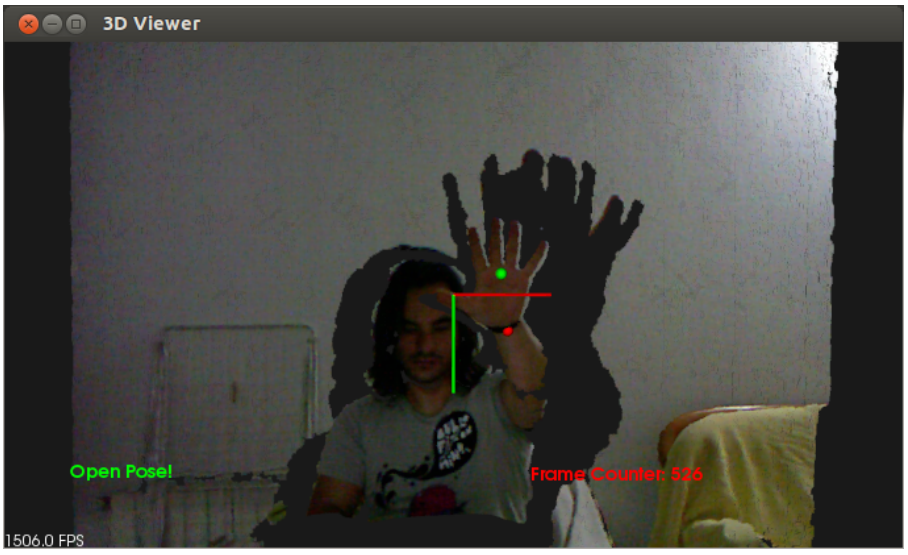


Figure 3.2: Hand Tracker.

3.3 Dataset Definition

The development of this work requires of a broad hand pose dataset, with perfectly segmented hand samples, and ultimately, based on the PCL library. Since currently this kind of dataset is not currently

available for the research community, it was required to build one that fulfils the requirements previously mentioned.

This task was greatly simplified by the tool referred on Section 3.2, that simultaneously detects, both hand point cloud blobs. So this tool was modified in order to disk store these samples.

The dataset comprises 6 hand pose classes, each one containing the pose of both hands, and high hand orientation variability, so the system can be trained in a highly robust fashion. A class containing non-pose elements, was also added to the dataset, it is formed by non-hand captured elements that are observed by the hand tracker. In Figure 3.3 some real point cloud samples of the dataset are presented.

3.4 Descriptor Definition

In this section it is proposed a novel shape model descriptor that can fully express the 3D characteristics of the described objects. It can be considered an extension of the *Blurred Shape Model Descriptor* described in [34], it defines a correlogram structure from the center of the object region, spatial arrangement of object parts is shared among regions defined by circles and sections the method codifies the spatial arrangement of object characteristics based on a prior blurring degree, which determines the shape deformation allowed to the object. Regions in a correlogram are used to vote object characteristics from neighbour regions. Contiguous regions share information about their containing object points, and thus, the descriptor is tolerant to irregular deformations.

Given the contours of an element represented in an image, the descriptor for each dot of the image contour, computes assigns a weight to a section of the correlogram where the dot is contained, propagating this weight to the closest neighbour sections, using a proportionally inverse relation in respect of the distance, of the considered dot to the centroids of the referred section (Figure 3.4).

This algorithm initially designed for images was extended, in order to enable the description of 3D point clouds. Instead of considering a circular correlogram surrounding the object of interest, it is considered a spherical correlogram, whose center is established in the point clouds centroid. In a nutshell this novel designed descriptor, defines a partition of the object of interest, in 3D spatial arrangement that divides it according to a series of concentric spheres and by sections, which are obtained by performing radial *cuts* across spheres (Figure 3.5). Each of these sections has an associated *weight*, that is incremented by one unit, for each cloud point P that is contained in that section S_P. This same point, propagates a weight to the surrounding 26 *neighbour* sections $N_i^P : 1 \leq i \leq 26$, that is inversely proportional to the distance, between the S_P and N_i^P. By performing this, we enable the sharing of information between neighbour section, and making the descriptor more tolerant to hand deformations.

A rigorous presentation of the SBSM descriptor method is provided in Algorithm 1.

Algorithm 1: Spherical Blurred Shape Model Descriptor

input : a point cloud $P = \{p_i \mid p_i \in \mathbb{R}^3\}$, point cloud centroid $P_{centroid}$, number radial layers N_L, number of θ angular divisions N_θ, number of ϕ angular divisions N_ϕ, sphere radius: S_{Radius}

output: Feature vector W of dimension $N_L \times N_\theta \times N_\phi$

Define: $S_R = S_{Radius}/N_L$ as the distance between consecutive layers, $S_\theta = 2\pi/N_\theta$, and $S_\phi = 2\pi/N_\phi$ the degrees in θ and ϕ polar coordinates components respectively, between consecutive sectors.

Define: Point cloud centroid, $P_{centroid} = \sum_{i=0}^{\#P} \frac{P_i}{\#P}$

Define: $P^* = P - P_{centroid}$, which now defines the center of coordinates as the cloud centroid.

Define: The set of bins B, for the spherical description of P^* $B = \{b_{\{0,0,0\}}, b_{\{1,0,0\}}, b_{\{2,0,0\}}, \ldots, b_{\{S_R,1,0\}}, b_{\{S_R,2,0\}}, \ldots, b_{\{S_R,S_\theta,1\}}, b_{\{S_R,S_\theta,2\}}, \ldots, b_{\{S_R,S\theta,S\phi\}}\}$ as the ordered set of bins for the spherical description of P, where $b_{\{p_R,p_\theta,p_\phi\}}$ is the bin of B, between distance $[p_R S_R, (p_R+1)S_R[$, $[p_\theta S_\theta,\ (p_\theta+1)S_\theta[$, and $[p_\phi S_\phi,\ (p_\phi+1)S_\phi[$. This way B defines a partition in $\mathbb{R}^3$ of the object of interest.

Define: $b^*_{\{i,j,k\}} = (iS_R + \frac{1}{2}S_R, jS_\theta + \frac{1}{2}S_\theta, jS_\phi + \frac{1}{2}S_\phi)$ the centroid of the section $b_{\{i,j,k\}}$

Define: $N(b_{\{i,j,k\}})$ as the neighbours of the section $b_{\{i,j,k\}}$ (Figure 3.5(b)).

Initialize: $W_n = 0, n \in \{1, \ldots, N_L N_\phi N_\phi\}$ as the weights associated to each one of the elements of B.

foreach $p_n \in P^*$ **do**
  $b_x : b_x \in B$ and $p_n \subset b_x$
  $W(b_x) = 1$
  foreach $b_{i,j,k} \in N(b_x)$ **do**
    $d_{i,j,k} = d(b_{i,j,k}, p_n) = ||p_n - b^*_x||$
    $W(b_{i,j,k}) = W(b_{i,j,k}) + \frac{1}{d_{i,j,k}}$
  end
end

Normalize the vector W as follows:
$\frac{W_i}{\#P}, i \in \{1, \ldots, N_L N_\phi N_\phi\}$

3.5 Classification

The classification phase is a crucial step of the overall system's performance, so it is mandatory fine tune its underlying parameters correctly, both the descriptor and classifier.At a first level, it is extremely important to assess the SBSM descriptor parameters, namely the number of required spheres (layers), and the number of angular divisions, since this has considerable implications on both the descriptor's discriminative capacity, and also in the runtime performance, since the final designed system must be able of real-time performance. In a second level, the classifiers performance depends heavily on the correct adjustment of its defining parameters.

At a first step, using the SBSM descriptor, we collected feature vectors for the following parameter combinations, namely $N_L = \{2, 4, 8\}$ and $N_\theta = N\phi = \{2, 4, 8, 16\}$, using 70% of each dataset class for training. selecting the samples randomly at each execution. In this phase the SVM classifier was used, more precisely libSVM [35] implementation. The assessment of the most adequate SVM configuration, was performed by using libSVM tools which are specially tailored for parameter selection of the C-SVM classification using the RBF (radial basis function) kernel. We use a 5-fold cross validation technique to estimate the accuracy of each parameter combination, namely *cost* and *gamma*.

The results achieved for each of the descriptors configurations are discussed in Chapter 4.

(a) Open hand pose (left).

(b) Open hand pose (right).

(c) Pointer pose (left)

(d) Pointer pose (right).

(e) Close hand pose (left).

(f) Close hand pose (right).

(g) Stop hand pose (left).

(h) Stop hand pose (right).

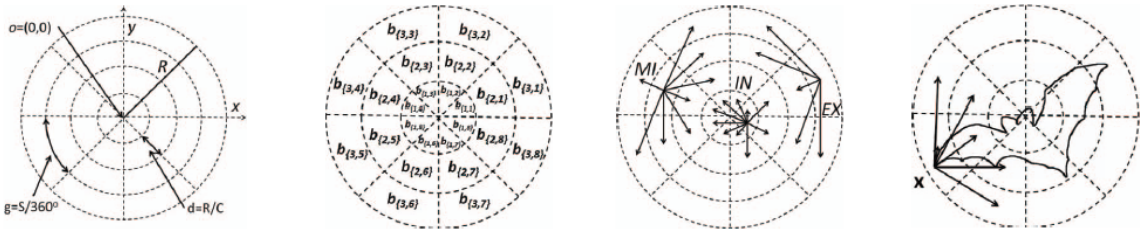
(i) Zoom pose (left).

(j) Zoom pose (right).

(k) Victory pose (left).

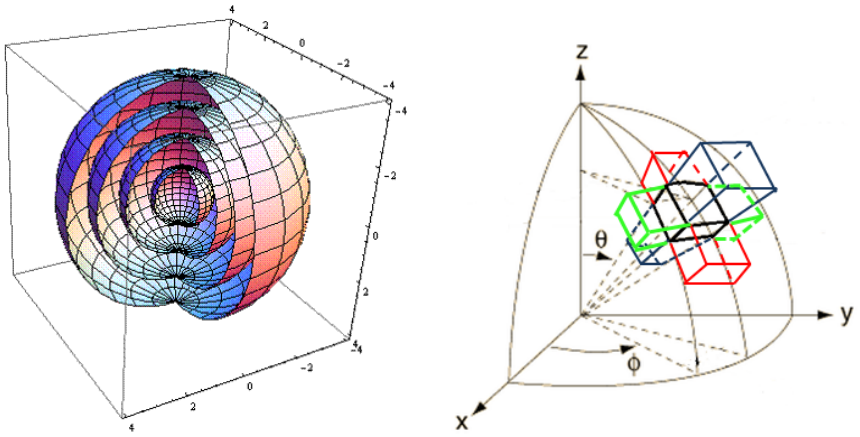
(l) Victory pose (right).

Figure 3.3: Dataset definition.



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

Figure 3.4: Circular blurred shape model descriptor.



(a) Descriptor example using 4 spheres

(b) Example of 6 of the 26 ($3^3 - 1$) section neighbours of the central reference section.

Figure 3.5: Descriptor components illustration.

Chapter 4

Results

In this chapter we analyse some of the achieved results, trying to evaluate one of the focal points of the whole system pipeline, the SBSM descriptor. This is extremely important since this is a novel descriptor, and it is crucial to understand the role of its underlying features in the final performance, both in terms of overall accuracy and execution time.

Finally, it is provided a general overview of the real-time pose recognition system prototype.

4.1 SBSM Descriptor Performance Evaluation

The descriptor testing is performed using the following parameter combination: $N_L = \{2, 4, 8\}$ and $N_\theta = N\phi = \{2, 4, 8, 16\}$. Each combination pair was executed 10 times in a cross-validation test, training with 70% of dataset class samples, randomly selected at each execution. In each test run, we also perform a cross-validation of the training data to find the best parameters - *cost* and *gamma*, of the SVM.

This modality is performed for two cases:

- Enforcing the descriptor weight propagation among the neighbour sections.
- Using Zoning, meaning that no weight propagation is used.

The results for the designed dataset are presented in Table 4.1 and Table 4.2 respectively.

The same results are aggregated graphically in Figure 4.2. However, it is more or less evident that weight propagation, has a very slight advantage over Zoning, and only when using a very large number of descriptor features. When considering a very small number of sections, the neighbour weight propagation introduces non-relevant data on the descriptor, and the Zoning modality achieves better

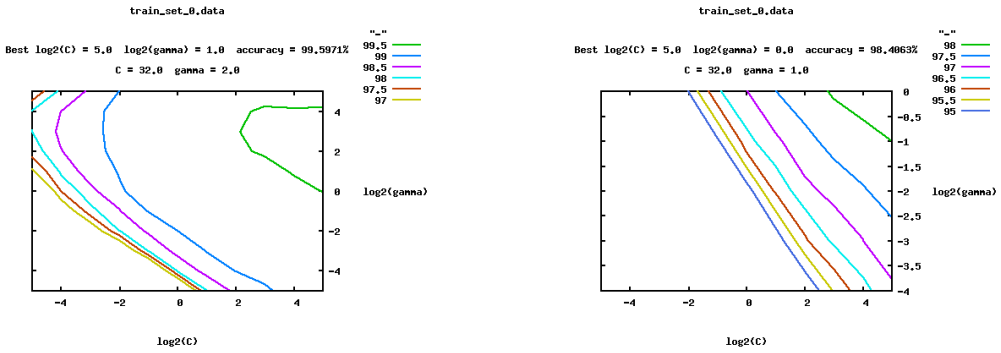


Figure 4.1: Train data cross validation using libSVM tools.

Descriptor Parameters			SVM Parameters			Results	
#Spheres	#Angles	#Descriptor Features	Cost	Gamma	Cross Validation Accuracy	Average Accuracy (μ)	Standard Deviation (σ)
2	2	9	8	0.25	77.8	78.42499	0.56188498
2	4	33	8	0.25	93.8	**94.02738**	0.502782219
2	8	129	1	0.25	97.6	98.04632	0.108874595
2	16	513	1	0.25	98.5	98.67991	0.130109855
4	2	17	32	1	98.4	98.59942	0.151137559
4	4	65	8	0.25	98.5	**98.63698**	0.128845366
4	8	257	8	1	99.8	**99.84706**	0.055197347
4	16	1025	1	1	99.8	**99.81756**	0.041554254
8	2	33	32	2	99.6	**99.65656**	0.102602601
8	4	129	8	8	99.7	**99.77193**	0.063560663
8	8	513	8	1	99.8	**99.83903**	0.065709834
8	16	2049	1	1	99.8	**99.78536**	0.043811850

Table 4.1: SBSM descriptor testing using weight propagation between neighbour sections.

global results in those cases. When the number of descriptor sections increases, we can observe that both descriptor modalities achieve more or less the same accuracy values, with a slight advantage for the weight propagation version. However, given that the discriminative power of the system is high, these small differences encourage the use of the propagation version of the descriptor algorithm.

In the 2D version of the descriptor [34], the weight propagation version achieved a better performance than the Zoning. However, in this case, the descriptor was applied over images, which implies, that every section had non-zero weights, and by this, the blurring effect provided by the weight propagation, helped to achieve better results and more discriminative power.

On the other hand, in this work the SBSM descriptor is applied over segmented 3D point clouds, which implies, that in some sections of the descriptor sphere won't contain any cloud point, neither receive weight propagations, implying that there will be a high number of sections with zero weight, which greatly simplifies the SVM classification task. These facts explain why the two modalities tested reveal approximate results, without a clear advantage of the weight propagation descriptor version, like

Descriptor Parameters			SVM Parameters			Results	
#Spheres	#Angles	#Descriptor Features	Cost	Gamma	Maximum Cross Validation Accuracy	Average Accuracy (μ)	Standard Deviation (σ)
2	2	9	8	8	88.5	**89.27825**	0.24172
2	4	33	1	1	89.5	90.79689	0.50751
2	8	129	1	1	98.6	**98.70871**	0.30883
2	16	513	1	1	99.4	**99.49826**	0.08203
4	2	17	8	8	96.9	**96.94123**	0.19056
4	4	65	8	8	99.5	99.53852	0.09533
4	8	257	1	1	99.5	99.47948	0.14266
4	16	1025	8	8	99.8	99.79611	0.06092
8	2	33	32	32	99.5	99.62711	0.06124
8	4	129	8	8	99.7	99.71291	0.05938
8	8	513	1	1	99.5	99.52676	0.08214
8	16	2049	1	1	99.5	99.59753	0.08854

Table 4.2: SBSM descriptor testing using Zoning (without weight propagation between neighbour sections).

it was expected from the 2D version of the descriptor. The other main reason is the complexity of the dataset. A future study on more difficult to learn datasets may show more clearly the benefits of the weight propagation procedure in the descriptor algorithm.

At this point, it is crucial to assess the computational performance of both descriptor modalities. For this we compute the average computational time of the descriptor when applied over the dataset. The results are expressed both in Table 4.3 and Figure 4.3. From these results it is easily observable the descriptor computation overweight caused by the weight propagation. In the case of the Zoning modality the computational cost increases slightly, due the initializing the descriptor structure, and a increasing number of sections. However, the computational cost for processing each the point cloud sample is approximately constant for all the descriptor configurations.

Descriptor Parameters			Processing Time (ms)	
Layers	Angles	Features	Weight Propagation	Zoning
2	2	9	3.828	1.787
2	4	33	4.026	2.517
2	8	129	7.741	2.645
2	16	513	22.914	2.764
4	2	17	4.709	2.215
4	4	65	5.904	2.329
4	8	257	10.327	2.646
4	16	1025	32.262	2.782
8	2	33	6.352	2.539
8	4	129	7.173	2.671
8	8	513	12.115	2.875
8	16	2049	38.136	2.985

Table 4.3: Descriptor average computation time per-sample.

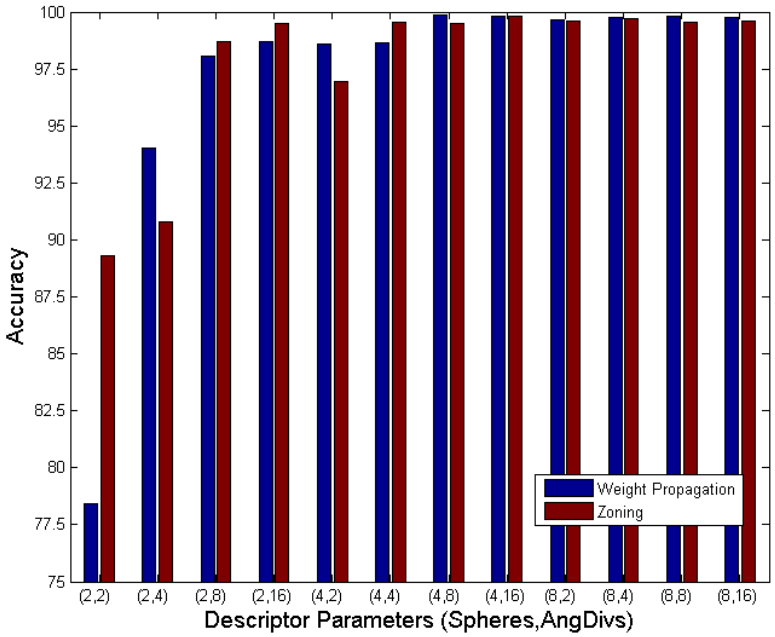


Figure 4.2: SBSM descriptor accuracy.

4.2 Real-time Pose Recognition System Prototype Overview

The pose recognition system prototype, uses the weighted SBSM descriptor, with a configuration of 8 spheres and 8 angular divisions. To perform the system training it was the full hand pose dataset, performing a cross-validation, using the libSVM tools, to fine tune the SVM cost and gamma parameters. The system's throughput achieves a performance of 14 fps when capturing a single hand, reducing to 11 fps when capturing both hands. The various hand pose dataset classes, derived by their their great orientation variability and SBSM discriminative power, are correctly recognized by the system like the experiments have shown. Here some screenshots of the real-time system are presented: open hand (Figure 4.4), pointer (Figure 4.5), zoom (Figure 4.6), close hand (Figure 4.7), victory (Figure 4.8) and stop (Figure 4.9) poses.

Also the system is capable of simultaneous hand pose recognition for both hands. Some screenshots are provided in Figure 4.10 and Figure 4.11

In Figure 4.12 provides a somewhat more visual understanding on what 3D data is extracted by the Kinect^{TM}depth sensor. The point cloud was slightly rotated to expose the captured body surfaces, while performing the real-time pose recognition.

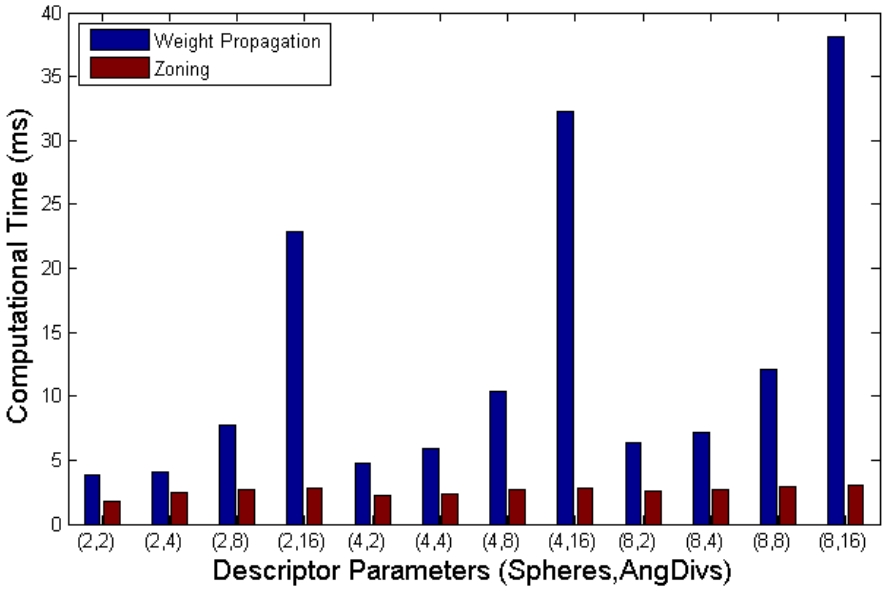


Figure 4.3: Computational performance of the SBSM descriptor .

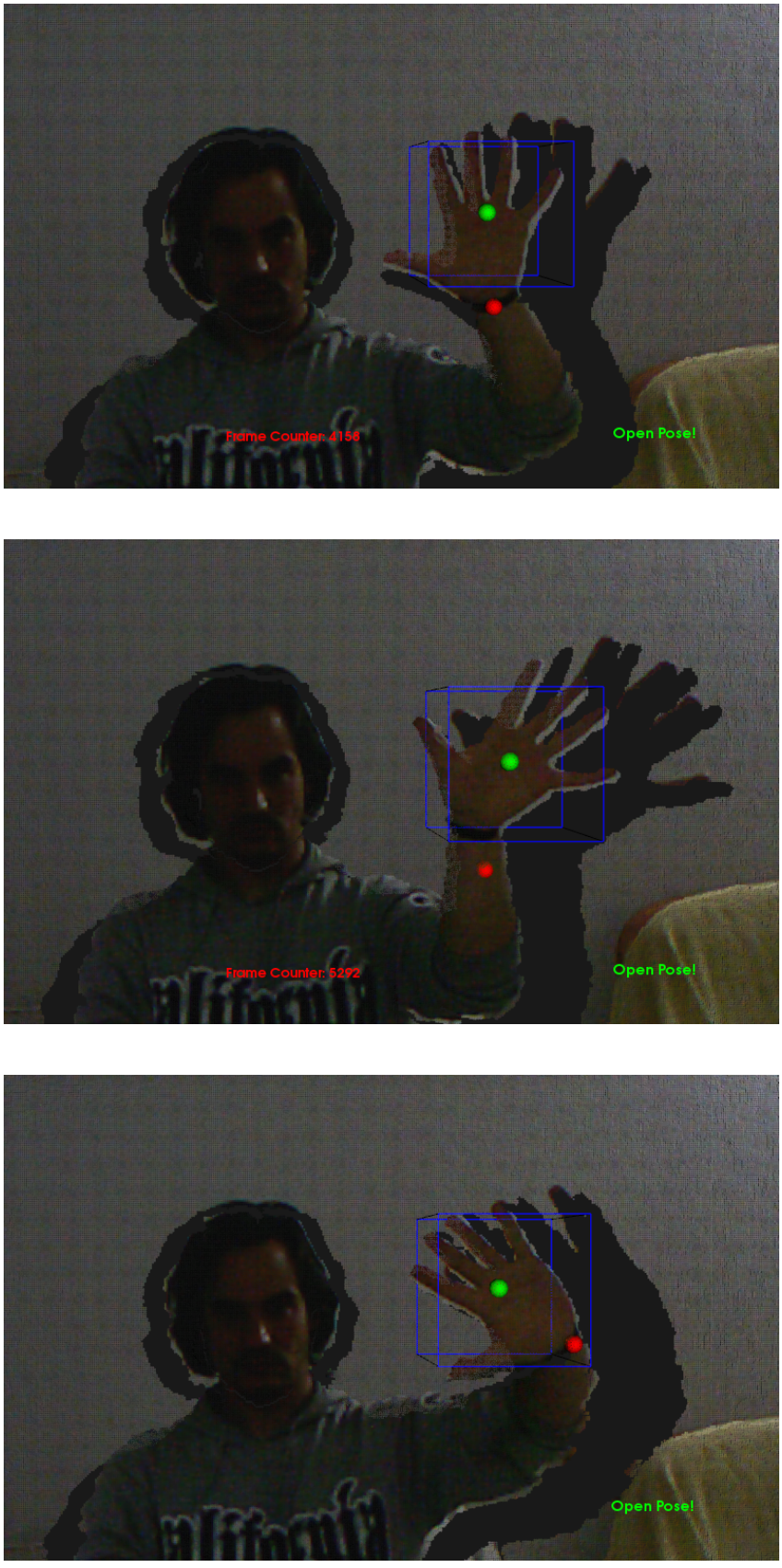


Figure 4.4: Open hand pose.

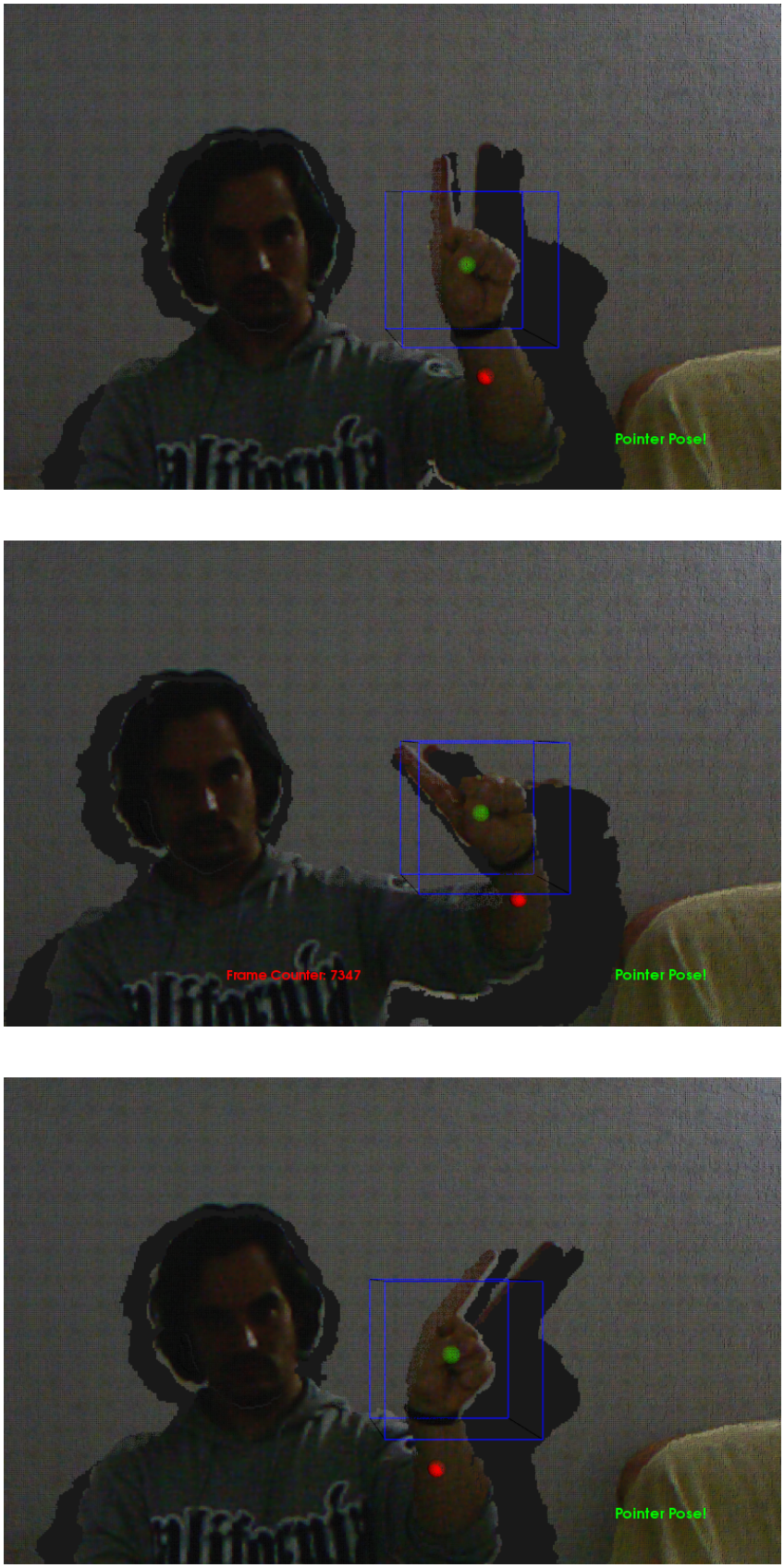


Figure 4.5: Pointer hand pose.

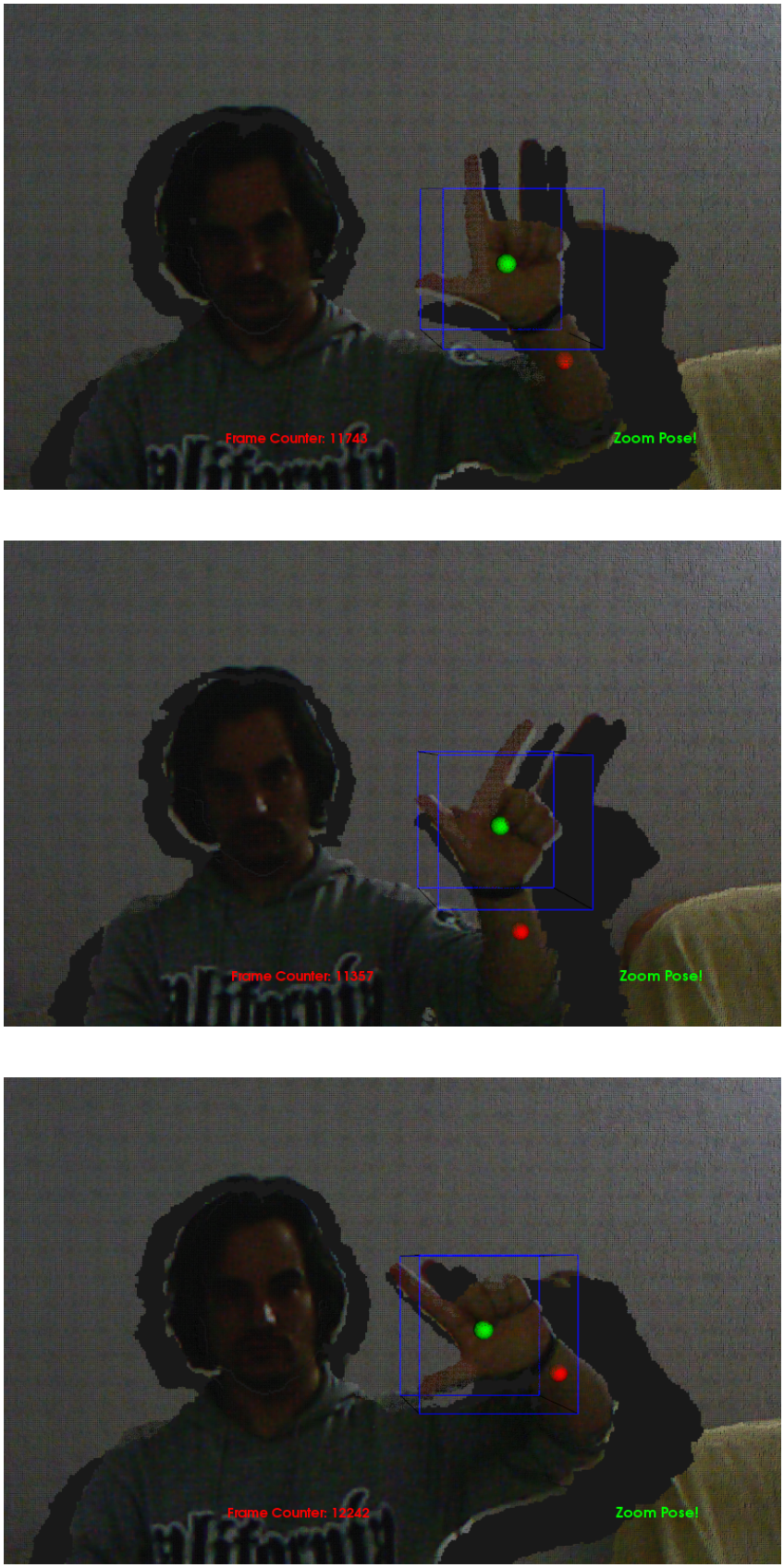


Figure 4.6: Zoom hand pose.

Figure 4.7: Close hand pose.

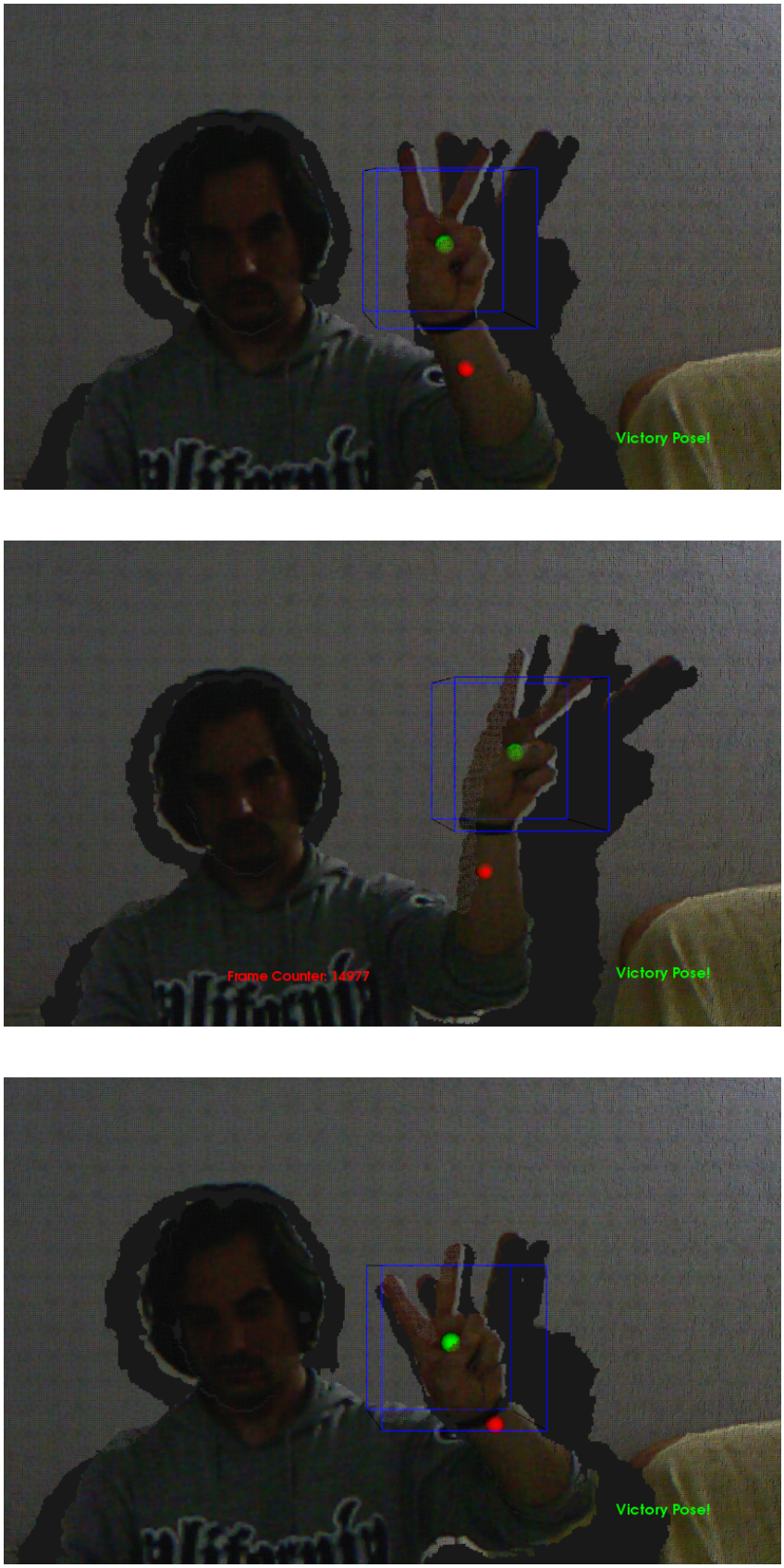


Figure 4.8: Victory hand pose.

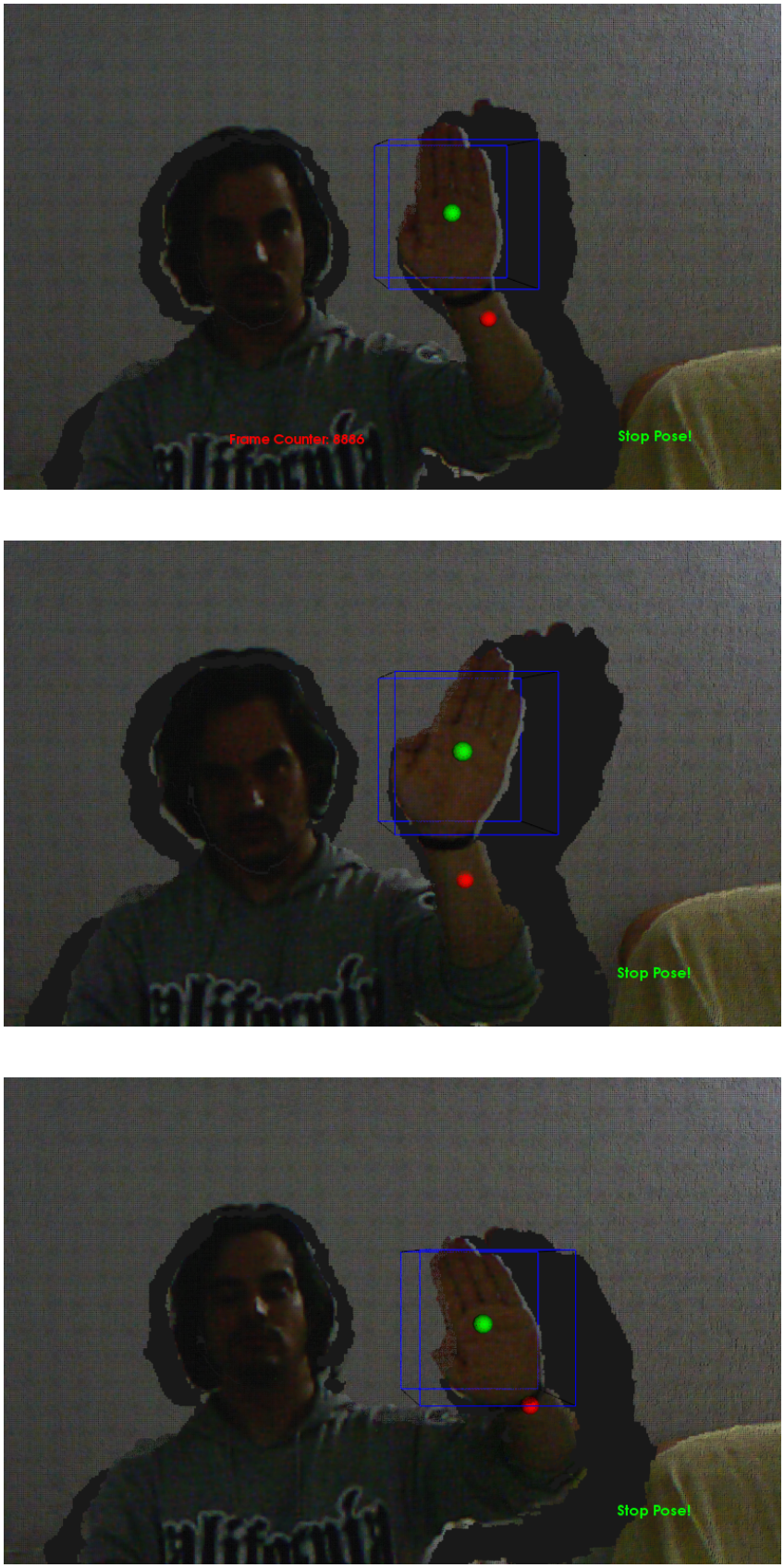


Figure 4.9: Stop hand pose.

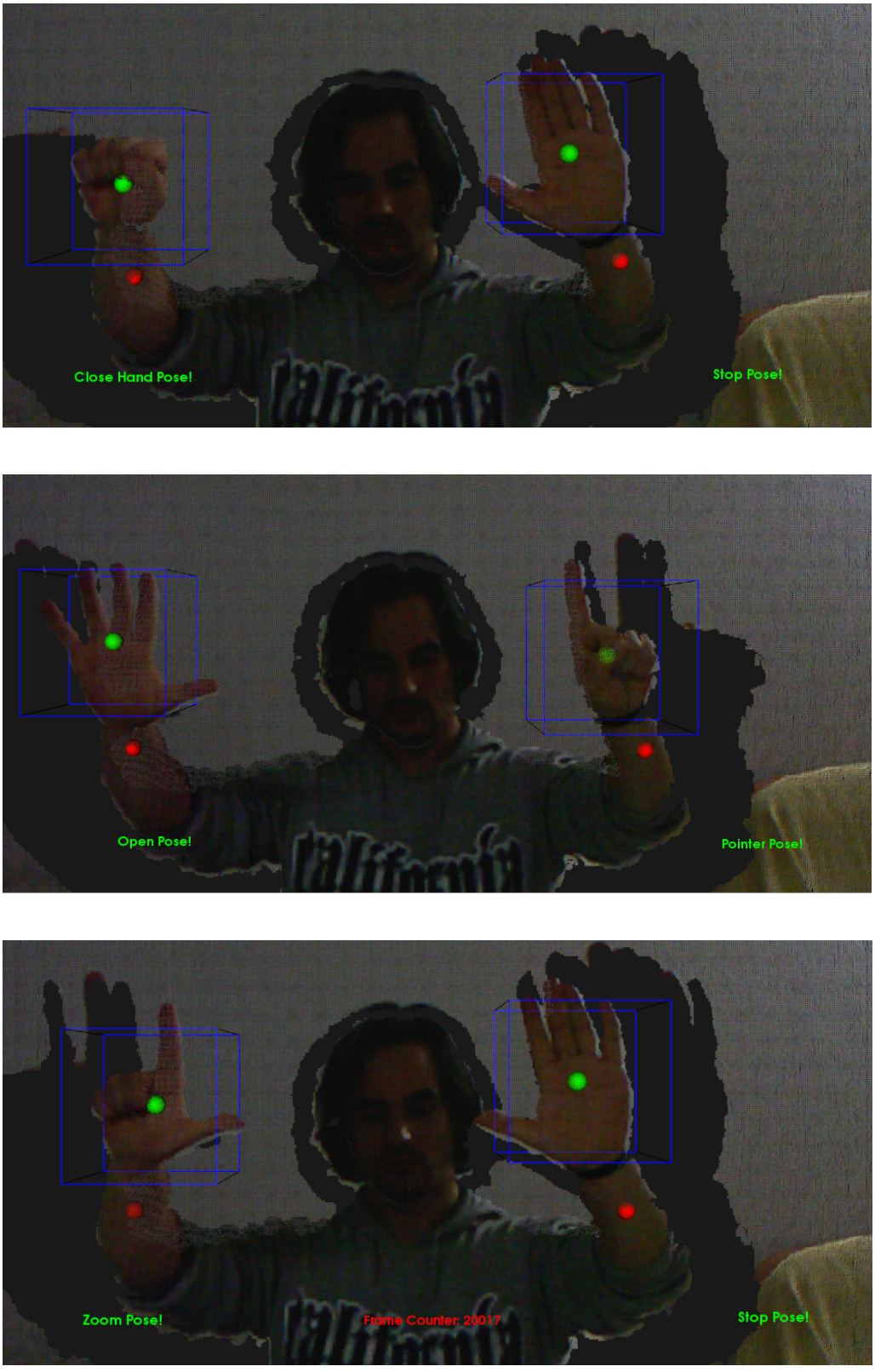


Figure 4.10: Simultaneous hand pose recognition.

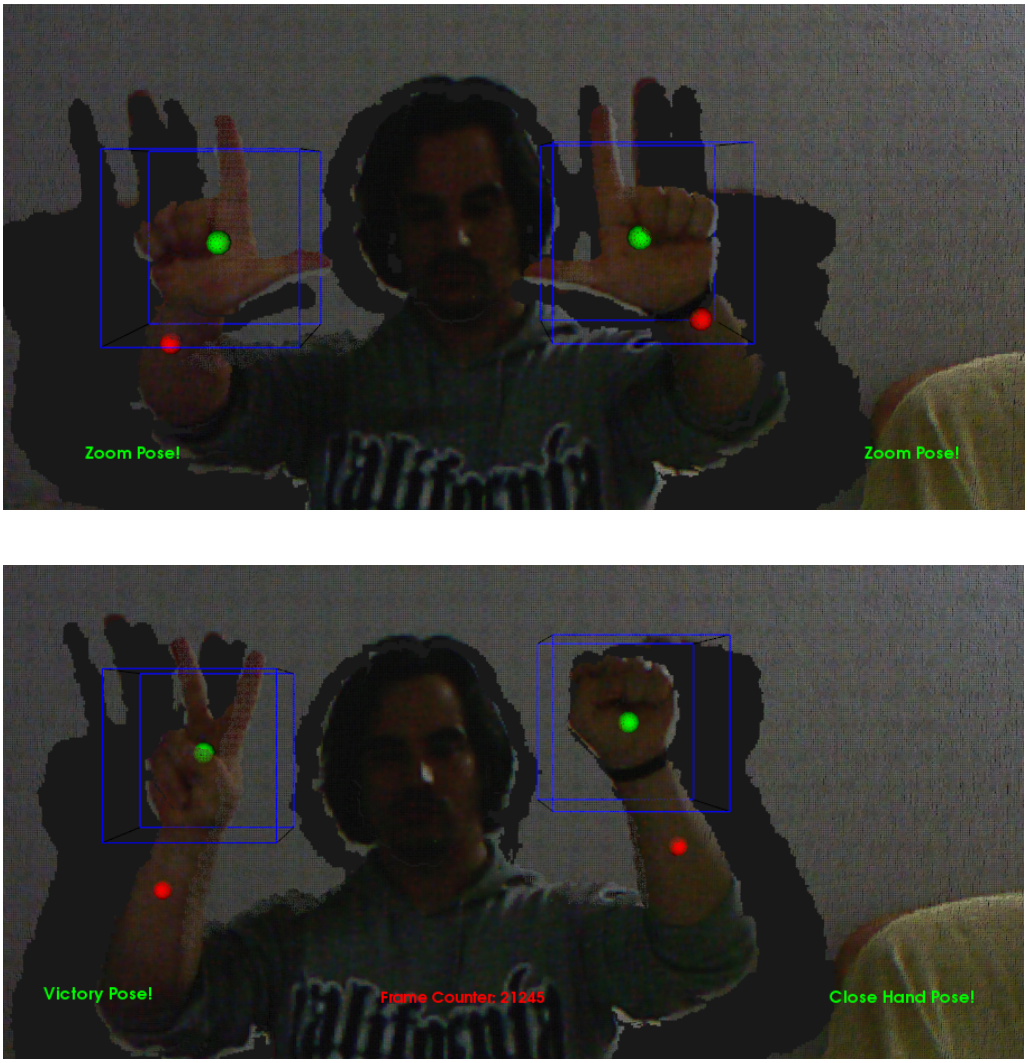


Figure 4.11: Simultaneous hand pose recognition.

Figure 4.12: Rotated point cloud.

Chapter 5

Conclusions and Future Work

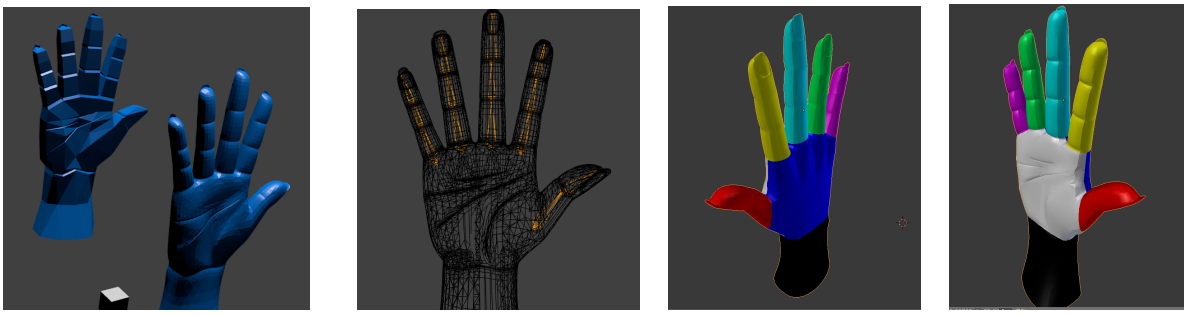
One of the focal points that contributed to the overall success of the followed approach is clearly the use of the Kinect^{TM}depth sensor, and the 3D scene information it extracts. This technology by itself, simplifies amazingly the pose recognition problem by removing ambiguities that cause tremendous problems and limitations in traditional RGB-based approaches, and thus, opening the door to new techniques that are more focused on the pose recognition issue, making the scene representation part of the solution, and not part of the problem.

This work presented an end-to-end, real-time pose recognition system pipeline. This contribution required the creation of a novel spherical shape model descriptor, with low computational overhead making it applicable in real-time architecture, and most importantly, capable of extracting relevant information from 3D point cloud data of an object of interest. On the other hand it was also presented a hand tracker solution, capable of capturing hand pose data, performing a robust hand segmentation, removing irrelevant information from the scene.

However, taking into account all the systems characteristics, some limitations have been identified: similar hand poses formed by different finger dispositions can generate wrong classifications, and this can be an issue if the wrong pose is not included in the *no-pose* class of the dataset.

This way, to ensure the system's robustness, the integration of a new module prior to the descriptor phase, should be considered. In this new component, a per-point classification would be performed over the previously segmented hand point cloud, using RDF classifier (loosely inspired in [31]). This module would provide, a labelled hand point cloud, where each finger would be clearly identified. Then in the descriptor phase some modifications would be required. Instead of having a *global weight* associated to each section, we would have weight histogram, where each bin would account the weights generated by a given finger, and the weight propagation among neighbour sections would occur solely among bins associated to the same finger label.

We believe that this improvement would amplify the descriptor's discriminative power, enabling the system to become more reliable. The labelled train data, in order to generate the RDF model, could



(a) Hand model base

(b) Articulated hand model mesh

(c) Hand model (back)

(d) Hand model (front)

Figure 5.1: Synthetic hand model.

be achieved by using synthetically generated labelled point clouds, from a hand model (Figure 5.1) that ultimately enables the creation of a dataset with a rich set of hand poses. This proposal is left as future work.

In terms of system performance throughput, the system achieves a high performance. However one of the system's bottlenecks is the descriptor computation, which is more demanding, when using a large number of sphere layers and angular divisions. The next evolution step will consist in the implementation of the descriptor's algorithm using the GPU (Graphics processing unit), that we consider, could lead to a major decrease of the computational time, boosting the performance of the overall system pipeline.

Nevertheless, the current system achieves a high result accuracy of hand pose recognition, and good runtime processing speed, which enabled the fulfilment of all the objectives initially established, and ultimately, making the system suitable for an integration on a more ambitious HCI system.

Bibliography

[1] A. Erol, G. Bebis, M. Nicolescu, R. Boyle, and X. Twombly, "A review on vision-based full dof hand motion estimation," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, june 2005, p. 75.

[2] F. Quek, "Unencumbered gestural interaction," *MultiMedia, IEEE*, vol. 3, no. 4, pp. 36 –47, winter 1996.

[3] "Put-that-there: Voice and gesture at the graphics interface," *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, vol. 14, no. 3, pp. 262–270, 1980. [Online]. Available: http://portal.acm.org/citation.cfm?id=800250.807503

[4] D. Sturman and D. Zeltzer, "A survey of glove-based input," *Computer Graphics and Applications, IEEE*, vol. 14, no. 1, pp. 30 –39, jan. 1994.

[5] (2012, August) Oblong industries. [Online]. Available: http://oblong.com

[6] J. S. Joon, "Reviewing principles and elements of animation for motion capture-based walk, run and jump," in *Computer Graphics, Imaging and Visualization (CGIV), 2010 Seventh International Conference on*, aug. 2010, pp. 55 –59.

[7] (2012) Vicon motion tracking. [Online]. Available: http://www.vicon.com

[8] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012. [Online]. Available: http://www.mdpi.com/1424-8220/12/2/1437

[9] (2012) Primesense natural interaction. [Online]. Available: http://www.primesense.com

[10] A. Shpunt and Z. Zalevsky, "Three-dimensional sensing using speckle patterns," Patent 20 090 096 783, April, 2009. [Online]. Available: http://www.freepatentsonline.com/y2009/0096783.html

[11] (2012) Openni development framework. [Online]. Available: http://www.openni.org

[12] (2012) Point cloud library. [Online]. Available: http://www.primesense.com

[13] R. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 1 –4.

[14] (2012) Eigen is a c++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms. [Online]. Available: http://eigen.tuxfamily.org

[15] (2012) Boost - portable c++ general purpose source libraries. [Online]. Available: http://www.boost.org

[16] J. Parker and M. Baumback, "Finger recognition for hand pose determination," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, oct. 2009, pp. 2492 –2497.

[17] M. Bhuyan, D. Neog, and M. Kar, "Hand pose recognition using geometric features," in *Communications (NCC), 2011 National Conference on*, jan. 2011, pp. 1 –5.

[18] A. El-Sallam, F. Sohel, and M. Bennamoun, "Robust pose invariant shape-based hand recognition," in *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*, june 2011, pp. 281 –286.

[19] S. Rautaray and A. Agrawal, "Interaction with virtual game through hand gesture recognition," in *Multimedia, Signal Processing and Communication Technologies (IMPACT), 2011 International Conference on*, dec. 2011, pp. 244 –247.

[20] B. Sabata, F. Arman, and J. Aggarwal, "Segmentation of 3d range images using pyramidal data structures," in *Computer Vision, 1990. Proceedings, Third International Conference on*, dec 1990, pp. 662 –666.

[21] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, "Real time motion capture using a single time-of-flight camera," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, june 2010, pp. 755 –762.

[22] J. Rodgers, D. Anguelov, H.-C. Pang, and D. Koller, "Object pose detection in range scan data," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 2006, pp. 2445 – 2452.

[23] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," phd, Tecnische Universitatet Muenchen, Munich, Germany, 10/2009 2009.

[24] R. Koch, I. Schiller, B. Bartczak, F. Kellner, and K. Koser, "Mixin3d: 3d mixed reality with tof-camera." in *Dyn3D*, ser. Lecture Notes in Computer Science, A. Kolb and R. Koch, Eds., vol. 5742. Springer, 2009, pp. 126–141.

[25] K. Lai, L. Bo, X. Ren, and D. Fox, "Sparse distance learning for object recognition combining rgb and depth information," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 4007 –4013.

[26] L. Bo, X. Ren, and D. Fox, "Depth kernel descriptors for object recognition," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, sept. 2011, pp. 821 –826.

[27] V. Castaneda, D. Mateus, and N. Navab, "Slam combining tof and high-resolution cameras," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, jan. 2011, pp. 672 –678.

[28] S. Escalera, "Human behavior analysis from depth maps," in *AMDO*, 2012, pp. 282–292.

[29] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, june 2011, pp. 1297 –1304.

[30] L. B. Statistics and L. Breiman, "Random forests," in *Machine Learning*, 2001, pp. 5–32.

[31] C. Keskin, F. Kirac, Y. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 1228 –1234.

[32] D. Minnen and Z. Zafrulla, "Towards robust cross-user hand tracking and shape recognition," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 1235 –1241.

[33] P. Suryanarayan, A. Subramanian, and D. Mandalapu, "Dynamic hand pose recognition using depth data," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, aug. 2010, pp. 3105 –3108.

[34] S. Escalera, A. Fornes, O. Pujol, A. Escudero, and P. Radeva, "Circular blurred shape model for symbol spotting in documents," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, nov. 2009, pp. 2005 –2008.

[35] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.