



POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
DOTTORATO DI RICERCA IN INGEGNERIA DELL'INFORMAZIONE

ANALISI E CLASSIFICAZIONE DI AZIONI UMANE TRAMITE UN SISTEMA DI ACQUISIZIONE MULTI-KINECT

Tesi di dottorato di:
Eliana Frigerio
ID: 753159

Relatore:

Prof. Marco Marcon

Tutore:

Prof. Andrea Monti Guarnieri

Coordinatore del programma di dottorato:

Prof. Carlo Fiorini

XXV ciclo

POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
Piazza Leonardo da Vinci 32 I 20133 — Milano



POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione
DOTTORATO DI RICERCA IN INGEGNERIA DELL'INFORMAZIONE

ANALYSIS AND CLASSIFICATION OF HUMAN ACTIONS WITH A MULTI-KINECT SYSTEM

Doctoral Dissertation of:
Eliana Frigerio
ID: 753159

Advisor:

Prof. Marco Marcon

Tutor:

Prof. Andrea Monti Guarnieri

Supervisor of the Doctoral Program:

Prof. Carlo Fiorini

XXV cycle

To all my family

Acknowledgments

I would like to thank my supervisor, Prof. Marco Marcon, who supported and encouraged me during the entire Ph.D. attending, offering me many interesting (and challenging) research activities. I am also grateful to Prof. Stefano Tubaro for his valuable advices.

I would like to thank Claudio Marchisio, Viviana D'Alto and all the ST Microelectronics team for welcoming me as a staff member and proposing me many “real world” problems, bringing me often to a more practical vision of research.

I would also like to thank Prof. Andrea Cavallaro of the Queen Mary University of London, for his kind hospitality and supervision during my staying at the School of Electronic Engineering and Computer Science. I'm also grateful to all the EECS staff, for their patience and assistance for making me feel like at home.

I greatly appreciated the helpful suggestions and improvements given by the thesis reviewers, Prof. Alberto Signoroni (Università degli studi di Brescia, Brescia, Italy) and Prof. Peter Eisert (Fraunhofer HHI, Berlin, Germany): many thanks for the careful work and interest.

I am also grateful to all the ISPG Lab people: it has been a pleasure to work with them, as well as sharing many funny (and also serious) moments.

The deepest and most grateful thank goes to my family. My parents Ida and Giuseppe, for the deep trust and the respect always shown in all these years. My sister Claudia, always present especially during my countless tantrum. Thank you so much Sister! My aunt Mary DisGrazia, for the infinity kindness that transmits in each moment of the day. Dear granny you are always in my heart and you know that I ask your help more often than I would like. The big Frigerio's family, thanks for the endearment. Without you I will never reach this goal and I will be never the person I am. Your support during my life has been invaluable.

I would like to thank all my friends for their cheer. Even if not all of you share my choices you are always on my side: thanks to everybody. You are the best and I love you them all.

A very special thank goes to Marco, for his great ability to make me laugh even when problems and worries seemed to be insurmountable and to always believe that I can do it ... we have done it!

I am also grateful to all the people that will read my work: I hope it will be helpful and interesting for you; it costs me sweat and tears, and I am deeply proud of it.

Sommario

In questa tesi si illustra una estensione innovativa degli operatori morfologici: dalla analisi ed elaborazione di immagini binarie, alla analisi ed elaborazione di immagini tridimensionali rappresentate tramite voxel. In particolare questi nuovi operatori morfologici (soprattutto l'estrazione di scheletro e l'operatore di thinning) sono stati applicati alla analisi e classificazione di un database di azioni umane in un vocabolario di gesti predefinito.

Partendo da una ricostruzione tridimensionale dell'attore presente nella scena, acquisita con una configurazione innovativa composta da sensori Microsoft Kinect per Xbox 360 e ottenuta discretizzando il volume osservato in voxel, gli operatori morfologici proposti si sono dimostrati essere uno strumento molto efficace, sia per ottenere una rappresentazione compatta del corpo umano, che per analizzarne i movimenti. Classici problemi legati alla *Computer Vision* o alla *Pattern Recognition* possono quindi essere affrontati con un *framework* comune che non richiede alcuna informazione a priori sull'oggetto in esame, nè impone alcun vincolo sulla qualità della ricostruzione su cui opera. L'unico input richiesto dagli operatori morfologici è un volume di voxel. Informazioni sulla natura del suddetto volume o vincoli di risoluzione sono legati solo alla particolare applicazione in cui gli operatori morfologici vengono utilizzati e non limitano l'applicabilità di nessuno degli operatori morfologici proposti.

Per quanto riguarda un classico problema di *Computer Vision*, l'algoritmo morfologico di estrazione di scheletro permette una ricostruzione della superficie dell'oggetto accurata e molto poco onerosa in termini computazionali. La superficie ricostruita può essere rappresentata, sia come unione di sfere, sia come isosuperficie di una funzione definita come combinazione lineare di funzioni elementari a supporto radiale. Questa rappresentazione risulta essere facilmente gestibile e interpretabile in termini gerarchici.

Per quanto riguarda un problema legato al *Pattern Recognition*, l'algoritmo mor-

fologico di thinning (estrazione del *curve skeleton*) permette di sottolineare meglio i movimenti umani. Il *curve skeleton*, definito come la rappresentazione larga 1 voxel di un volume 3D, riduce la dipendenza della rappresentazione dalla particolare struttura fisica e genere dell'attore che sta eseguendo l'azione. Questo permette di migliorare notevolmente le performance del sistema di classificazione di azioni umane.

La prima parte della tesi include inoltre la descrizione della costruzione di un nuovo sistema di acquisizione per le ricostruzioni tridimensionali basato sull'uso di Kinect, dispositivi a basso costo recentemente apparsi sul mercato. Abbiamo dimostrato come la scelta di utilizzare i Kinect permetta di sfruttare le informazioni provenienti dalle mappe di profondità, generate direttamente dai dispositivi insieme alle corrispondenti immagini a colori. La loro utilità si è dimostrata duplice: nel caso della calibrazione ha permesso di ottenere corrispondenze putative più robuste di quanto ottenibile con i classici metodi basati solo sulle immagini a colori; nel caso della ricostruzione a valle della fusione delle diverse viste ha permesso di ricostruire la scena anche nelle sue parti concave (solo il *Convex Hull* è ottenibile se si usa un approccio *Visual Hull* solo con camere RGB). Gli algoritmi utilizzati per l'estrazione di corrispondenze robuste ritornano poi utili anche per l'estrazione di *feature* di moto robuste e invarianti rispetto alla posizione dell'attore nella scena, permettendo di ottenere un'ottima percentuale di classificazione anche implementando un classificatore molto semplice come quello a minima distanza.

Le soluzioni proposte in questo lavoro trovano potenziali applicazioni in tutti i campi che si basano su una visione o ricostruzione tridimensionale della scena. In particolare sono state pensate per inserirsi, per esempio, negli ambiti della video sicurezza (assistenza e supporto domestico), interazione fra uomo e computer (con interfacce user-friendly) e video giochi immersivi.

Abstract

This thesis describes an innovative extension of morphological operators: from the analysis and processing of binary images, to the analysis and processing of three-dimensional images represented by voxels. In particular, these new morphological operators (especially the morphological skeleton extraction and the 3D thinning algorithms) are applied to the analysis and classification of a database of human actions in a predefined vocabulary of gestures.

Starting from a three-dimensional reconstruction of the actor in the scene, acquired with an innovative configuration composed of Microsoft Xbox 360 Kinect devices and obtained by discretizing the acquired volume using voxels, the morphological operators, proposed in this thesis, have proven to be a really effective tool both to obtain a compact representation of the human body and to analyze its movements. Classical problems of Computer Vision or of Pattern Recognition can be addressed with a common framework that, neither requires any prior information on the reconstructed volume, nor imposes any resolution requirement on the subject under analysis. The only input for morphological operators is a volume of voxels. Information on the volume nature or resolution requirements are related only to the particular application to which the morphological operators are applied, but do not limit the applicability of anyone of the proposed morphological operators.

With regard to a classical Computer Vision problem, the morphological skeleton extraction algorithm allows a reconstruction of the object surface that is accurate and very inexpensive in terms of computational time. The reconstructed surface is representable either as a union of balls or as an isosurface of a 3D function defined as a linear combination of elementary functions with radial support. Furthermore, the representation is easily processable and hierarchically interpretable.

With regard to a problem related to Pattern Recognition, the morphological 3D thinning algorithm (*curves skeleton* extraction) allows to better emphasize the human movement. The curve skeleton, defined as a one-voxel wide representation

of a 3D volume, reduces the dependence of the representation from the particular physical structure and from the gender of the performer. This greatly improves the performance of the human action classification system.

In addition, the first part of the thesis includes a description of the setting up of a new acquisition system for three-dimensional reconstructions using three Kinect devices. Kinect is a low-cost device recently appeared on the market. We show how to exploit the information coming from depth maps, directly produced by the device together with the corresponding color images. Their utility is twofold: in the case of calibration, it produces putative matches that are more robust than those achievable with traditional methods based only on RGB images; in the case of reconstruction merging different views, it allows to obtain even the concave parts of the scene (only the *Convex Hull* can be obtained using color images with a *Visual Hull* approach). The algorithm used for extracting robust correspondences is also useful to extract motion features that are robust and invariant respect to the actor position in the scene, allowing to obtain a good classification rate even implementing a very simple classifier as the minimum distance is.

The solutions proposed in this thesis find potential applications in all the fields that are based on a 3D reconstruction of the scene. In particular they are designed to fit in the area of video security (home care and support), human and computer interaction (with user-friendly interfaces) and immersing video games for example.

Contents

1	Introduction	1
1.1	Motivations	1
1.2	Examined Issues	3
1.3	Application Scenarios	5
1.4	Original Contributions	7
1.5	Application Fields	8
1.6	Thesis Outline	8
1.7	List of Publications	10
2	Acquisition System	11
2.1	Acquisition Setup	13
2.1.1	The Microsoft Xbox 360 Kinect Device	14
2.1.2	Time Synchronization	16
2.2	Kinect Calibration	18
2.2.1	RGB Camera Calibration	21
2.2.2	Depth Camera Calibration	22
2.2.3	RGB and Depth Cameras Calibration	23
2.2.4	Optimization	24
2.3	Kinect Pair Calibration	24
2.4	3 Kinect Net Calibration	28
2.5	Wide Baseline Correspondences using Depth Maps	32
2.5.1	Surface vs. Texture Relevant Points	34
2.5.2	Local Plane Extraction	35
2.5.3	Plane Rectification	37
2.5.4	Similarity Invariant Transform	41
2.5.5	Proposed Algorithm for Stereo-matching	44

2.5.6	Results and Discussion	45
3	Volumetric Human Body Data	49
3.1	Image Segmentation	51
3.2	Projection Cylinders	54
3.3	Volumetric Reconstruction	56
3.4	Database	58
3.4.1	Crouch	60
3.4.2	Grasp	61
3.4.3	Kick	62
3.4.4	March	63
3.4.5	Move	64
3.4.6	Open	65
3.4.7	PointAt	66
3.4.8	Pull	67
3.4.9	Push	68
3.4.10	Walk	69
4	3D Morphological Operators	71
4.1	Review on Digital Topology	72
4.2	Morphological Operators	74
4.2.1	Erosion	74
4.2.2	Dilation	75
4.2.3	Opening	75
4.2.4	Closing	76
4.2.5	Hit-or-Miss Transform	76
4.3	Morphological Algorithms	79
4.3.1	Surface Extraction	79
4.3.2	Extraction of Connected Elements	79
4.3.3	Morphological Skeleton Extraction	80
4.3.4	Thinning	81
5	Curve Skeleton	89
5.1	Skeletonization	90
5.2	Desirable Properties	93
5.2.1	Homotopy (topology preservation)	94

5.2.2	Connectivity	94
5.2.3	Invariance under Isometric Transformations	95
5.2.4	Reconstruction	95
5.2.5	Thinness (1D)	96
5.2.6	Centeredness	97
5.2.7	Robustness	98
5.2.8	Efficiency	98
5.2.9	Discussion	99
5.3	3D Thinning Algorithm to extract Curve Skeleton	100
5.3.1	Important Notions	100
5.3.2	Proposed 3D Thinning Algorithm	101
5.4	Results	104
6	Surface Representation	125
6.1	Simulated Acquisition Set-up	127
6.1.1	Building Synthetic Depth Maps using LightWave	127
6.1.2	Canonical Volumetric Reconstruction	128
6.1.3	Volumetric Integration	130
6.2	Surface Representation as Union of Maximal Balls	131
6.2.1	Medial Axis and Medial Axis Transform	132
6.2.2	Medial Axis Approximation by a Set of Poles	133
6.3	Surface Representation as Implicit Surface	134
6.4	Implemented Algorithms for Surface Representation	138
6.4.1	Union of Maximal Balls	139
6.4.2	Implicit Surface	142
6.4.3	Discussion	144
6.5	Surface Reconstruction from Curve Skeleton	146
7	Human Action Classification	151
7.1	Action Representation	153
7.1.1	Motion History Images	154
7.1.2	Motion History Volumes	155
7.2	Motion Descriptors	157
7.2.1	Invariant Representation	159
7.3	Action Classification	161
		IX

Contents

7.3.1	Dimensionality Reduction	161
7.3.2	Classification	164
7.4	Results	165
8	Conclusions and future research directions	171
8.1	Conclusions	171
8.1.1	Calibration	171
8.1.2	Volumetric Reconstruction	172
8.1.3	Morphology	173
8.1.4	Surface Reconstruction	173
8.1.5	Action Classification	173
8.2	Future Works	174
8.2.1	Segmentation in a Real Environment	174
8.2.2	Medial Surface Approximation	175
8.2.3	Exploring a Large Data Set	175
8.2.4	Segmenting Continuous Gesture Sequences	175
8.2.5	Gesture Recognition with Additional Information	176
8.2.6	Dealing with Multiple Actors and/or Occlusions	176

List of Figures

1.1	Block diagram of the proposed method for actor representation and classification of human actions captured with a multi-Kinect system.	9
2.1	Block diagram of the implemented calibration procedure developed for a net of 3 Kinect sensors.	12
2.2	Smart Space.	13
2.3	Block diagram of the MS Kinect sensor.	14
2.4	RGB and depth images acquired with a Kinect.	15
2.5	Kinect IR pattern used for the depth map estimation.	15
2.6	Scheme of the triangulation principle.	16
2.7	Examples of RGB and depth images acquired with a Kinect.	17
2.8	Sample calibration images.	19
2.9	Camera pin-hole model.	20
2.10	Reference frames and transformations present on a scene.	21
2.11	Epipolar geometry.	25
2.12	Camera matrix estimation.	28
2.13	Closed motion configuration.	29
2.14	Representation of the scene acquired by the proposed net of 3 calibrated Kinect devices.	31
2.15	Block diagram of the proposed matching algorithm.	32
2.16	Texture and surface corners.	35
2.17	Color images and corresponding depth maps acquired with Kinect device.	36
2.18	Clustering of different planes on a depth map.	37
2.19	Schematic representation of the plane rotation procedure.	38
2.20	Frontal view image obtained after the projection of the rotated plane on the camera image plane.	39
		XI

List of Figures

2.21	Synthetic images, corresponding depth maps, and images obtained after the rectification process.	40
2.22	Example of neighborhood of the interest point in cartesian coordinates and in log-polar coordinates.	42
2.23	Gray scale images of a box acquired from different viewpoints and corresponding depth maps.	46
2.24	Images of the rectified version of the same plane estimated from the depth map.	46
2.25	Point correspondences found by the SID and the SIFT descriptors.	47
3.1	Block diagram of the acquisition process till the 3D reconstruction of the actor pose in each frame.	50
3.2	Example of acquired image and its visualization in the Hue, Value, Saturation channels.	52
3.3	Block diagram of the segmentation process.	53
3.4	From the segmented depth map to the volumetric rendering.	54
3.5	Voxel sampling of the scene.	55
3.6	Block diagram of the 3D volume reconstruction process.	57
3.7	Actors in the default pose.	58
3.8	Crouch.	60
3.9	Grasp.	61
3.10	Kick.	62
3.11	March.	63
3.12	Move.	64
3.13	Open.	65
3.14	PointAt.	66
3.15	Pull.	67
3.16	Push.	68
3.17	Walk.	69
4.1	6-, 18-, and 26-neighborhoods of a voxel p.	72
4.2	The six major direction in the $\mathbb{Z}^3$ space.	73
4.3	Cavity and tunnel.	74
4.4	Morphological erosion using a structuring element E.	74
4.5	Morphological dilation using a structuring element E.	75

4.6	Morphological opening using a structuring element E	76
4.7	Morphological closing using a structuring element E	76
4.8	Example of structuring element, local neighborhood and local background.	77
4.9	Hit-or-Miss transform using a structuring element E	77
4.10	Simplified Hit-or-Miss transform using a structuring element E	78
4.11	Surface Extraction.	79
4.12	Step by step representation of the morphological skeleton extraction algorithm.	80
4.13	Thinning using a structuring element E	81
4.14	Thinning using structuring elements $\{E\}$	82
4.15	Example of structuring elements for thinning.	83
4.16	Thinning using a structuring element E designed for peeling in a specific direction.	84
4.17	The six basic structuring elements $E_U^1 - E_U^6$ assigned to the direction U of the thinning algorithm proposed by Palágyi and Kuba [93]. . .	85
4.18	Thinning using the Palágyi and Kuba algorithm [93].	86
4.19	Thinning using one type of structuring elements of the Palágyi and Kuba algorithm [93].	87
5.1	Surface and curve skeletons.	91
5.2	The four basic structuring elements $E_U^1 - E_U^4$ assigned to the direction U of the proposed thinning algorithm.	103
5.3	Thinning using the proposed thinning algorithm.	103
5.4	Thinning using one type of structuring elements of the proposed thinning algorithm.	104
5.5	Thinning using the proposed algorithm on six reconstructed frames from the action: Crouch	105
5.6	Thinning using the proposed algorithm on six reconstructed frames from the action: Grasp	106
5.7	Thinning using the proposed algorithm on six reconstructed frames from the action: Kick	107
5.8	Thinning using the proposed algorithm on six reconstructed frames from the action: March	108

List of Figures

5.9	Thinning using the proposed algorithm on six reconstructed frames from the action: Move	109
5.10	Thinning using the proposed algorithm on six reconstructed frames from the action: Open	110
5.11	Thinning using the proposed algorithm on six reconstructed frames from the action: PointAt	111
5.12	Thinning using the proposed algorithm on six reconstructed frames from the action: Pull	112
5.13	Thinning using the proposed algorithm on six reconstructed frames from the action: Push	113
5.14	Thinning using the proposed algorithm on six reconstructed frames from the action: Walk	114
5.15	Curve skeleton extraction algorithms comparison.	115
5.16	Curve skeleton topology preservation property.	116
5.17	Original objects corrupted with additive Gaussian noise ($\sigma = 0.2$) and their curve skeletons.	117
5.18	Original objects corrupted with additive Gaussian noise ($\sigma = 0.6$) and their curve skeletons.	118
5.19	Curve skeleton rotation invariance property.	119
5.20	Efficiency comparison of our and Palágyi and Kuba [93] thinning algorithms.	121
5.21	Surface extracted and volume recovered from the original volumes.	123
6.1	Simulated acquisition set up.	128
6.2	Depth map acquired and 3D surface points reconstructed.	129
6.3	Volumetric rendering from the acquired depth map.	130
6.4	Surface estimation of the volumetric reconstruction.	131
6.5	Surface representation as Union of balls [3].	134
6.6	Wendland's Radial Basis Function [119].	137
6.7	Surface representation as iso-surface of a function defined as a linear combination of Radial Basis Functions (RBF) [100].	138
6.8	Medial surface approximation with the morphological skeleton.	139
6.9	Surface approximation as Union of balls.	140
6.10	Morphological skeleton and balls filtered according to the minimum number of 26-connected elements composing each block.	141

6.11	Balls filtered according to the minimum allowable radius.	142
6.12	Balls filtered according to the minimum allowable number of intersecting balls.	142
6.13	Surface representation as zero level of a function defined as linear combination of Wendland's Radial Basis Functions.	143
6.14	Surface representation as an isosurface of a function defined as linear combination of Radial Basis Function with Gaussian support.	144
6.15	Surface representation as union of balls and as zero level of a function defined as linear combination of Wendland's Radial Basis Functions.	145
6.16	Block diagram of the implemented surface reconstruction procedure.	146
6.17	Surface reconstruction as union of balls from curve skeleton.	147
6.18	Two examples of curve skeletons extracted using our algorithm or the Palágyi <i>et al.</i> [93] algorithm.	148
6.19	Skeleton balls filtered according to the minimum allowable number of intersecting balls.	148
6.20	Examples of surface representation as an iso-surface of a function defined, using the curve skeleton voxels, as a linear combination of Wendland's Radial Basis Function.	149
7.1	Block diagram of the classification procedure from the 3D volumetric reconstruction of the actor pose in each frame.	152
7.2	Motion versus occupancy.	154
7.3	Motion History Volumes.	155
7.4	Block diagram of the action representation process.	156
7.5	Motion History Volumes computed directly on the voxel set or on the reconstructed voxel set after the thinning procedure.	158
7.6	Motion History Volumes in cylindrical coordinates.	159
7.7	1D-Fourier transform in cylindrical coordinates.	160
7.8	Block diagram of the computation of the motion descriptor.	161
7.9	Block diagram of the classification process from the computed feature vector to the definition of the action label.	163
7.10	Similarity matrices.	167
7.11	Motion History Volumes on the curve skeletons.	169

List of Tables

2.1	Matching results with different descriptors.	47
5.1	Efficiency comparison of our and Palágyi and Kuba [93] curve skeleton extraction algorithms.	121
5.2	Data reduction and representative power for our and the Palágyi and Kuba [93] thinning algorithms.	124
6.1	Volumetric Reconstruction using LightWave.	132
7.1	Action classification results.	168

1 Introduction

1.1 Motivations

The importance of realizing naturalistic and socially aware computing and interfaces, built for humans and based on models of human behaviors, is indisputable. Gestures and actions are among the principal ways through which a human interacts with the reality and the possibility of building an automatic system, capable of receiving and classifying this type of information, has been one of the most fascinating spur for the research community in recent years. Pattern Recognition is the field of study regarding the definition and codification of algorithms and theories dedicated to the design of automatic processes for signal classification. The goal is to mimic the complex human decisional processes involved in the classification of inputs into categories using comparatively simple analytical structures, easily translatable in machine language procedures. According to Neumann *et al.* [89] and from a computational perspective, actions are best defined as four-dimensional patterns in space and in time. The natural way to deal with action representation is, thus, in the 3D environment. Computer Vision technology provides solutions to the problem of reconstructing a 3D scene starting from its projection on multiple image planes.

Marker-less human motion analysis systems address the problem of estimating human body motions in non-cooperative environments. Techniques from Computer Vision and Pattern Recognition serve the purpose of extracting information on human body postures from video sequences, without the need of wearable markers. Multi-camera systems further enhance this kind of application providing frames from multiple viewpoints. In the last decade many multi-camera acquisition setups are developed using 5 [117] or 8 [95], [111] cameras for recovering a 3D representation of an actor in the scene. Multi-camera systems, which constituted the standard for dynamic 3D scenes reconstruction for many years, are being replaced with multi-stereo camera systems, or better multi-DIBR (Depth Image Based Rendering) device

1 Introduction

systems. Recent years have been characterized by an exponential growing of multimedia applications employing multi-IR-based sensor systems. The need of acquiring dynamic 3D scenes has driven both academic and industrial worlds towards the design of effective sensors and algorithms for geometry estimation. Time-of-Flight cameras [36], structured light 3D scanners [52], and multi-camera systems [95], for example, permit a real-time acquisition of both static and dynamic elements with different accuracies and resolutions. Among these, IR-based sensors have proved to be an interesting research field, because of the many possibilities they offer and the related problems that have not been completely solved. As a matter of fact, several innovative applications (*e.g.*, tracking and identification [64], human computer interactions [122], and reconstruction) have been proposed in the literature during the last years.

In this thesis we explore the use of a multi-Kinect acquisition system (composed by 3 Kinect) for the reconstruction and analysis of an indoor dynamic scene. It is a low cost implementation thought also for user-friendly and everyday applications. From the reconstructed information, the purposes of the research work described in this thesis are the design, implementation and testing of an automatic human action analysis and classification system. Two are the system constraints: the first is the restriction to the classification of actions performed by one actor at a time, while the second is to provide the system with sequences each one containing the execution of a single type of gesture at a time. Multiple actors situations and continue gestures classification can be considered, however, natural extensions of the system designed in this work.

After the acquisition scenario is implemented and the 3D volume information extracted, the next part of our research is related to Pattern Recognition area and it is focused on the development of new methodologies for the extraction of signals from volumetric data able to represent meaningful information. In particular, we dedicate our research to the analysis of the human body and the classification of sequences of 3D reconstructed human poses in a predefined vocabulary of gestures. Inside this framework, the many possible applications and the related problems are often faced with *ad-hoc* techniques in the literature. We believe, however, that a great amount of applications can highly benefit from a morphological volume processing, not yet deeply studied. Let us consider, for example, the suitable morphological algorithms for images. They are commonly used *e.g.* in Optical Character Recognition, Biometry (finger print enhancement), video restoration, and medical image analysis [38].

The same powerful properties of morphological operators can still be valid for binary volumes and 3D morphological operators can be successfully implemented in many applications. We develop several 3D morphological algorithms and apply them to different possible application scenarios, showing their efficiency and performances.

In the following Sections we introduce the examined issues, the application scenarios considered for testing the proposed methodologies, the main contributions, and possible application fields.

1.2 Examined Issues

Volumetric Reconstruction

The choice of the input devices has been the starting point of this work. According to the literature [53], image-based classification systems can be distinguished on the basis of the type of input device used. Monocular systems receive input images only from a single view of the scene. This solution suffers from the loss of spatial information when the 3D scene is projected on the camera image plane: an actor performing a posture can be represented by different image signals depending on his position relative to the camera. Multi-camera scenarios are designed to solve this problem providing multiple views of the scene. In order to truly represent information about postures, the choice of using a multi-camera system seems to be more profitable.

Moreover, thinking on an innovative and user friendly camera scenario, we decide to acquire sequences of human actions with a system composed by 3 Kinect devices. The first part of our research work is dedicated to the extraction of volumetric information starting from the acquisition system implemented. The methodologies that we propose are based on Computer Vision theory applied on color images and depth maps simultaneously. In order to obtain a 3D reconstruction of the acquired scene, the devices must be calibrated: intrinsic and extrinsic parameters must be known. Points matching between two or more images of a scene shot from different viewpoints is the crucial step to define epipolar geometry between views. Unfortunately, in most cases robust correspondences between points in different images can be defined only when small variations in viewpoint position, focal length, or lighting are present between images. In all the other conditions, *ad-hoc* assumptions on the 3D scene or just weak correspondences through statistical approaches can be

1 Introduction

used. In this research, we present a novel matching method where depth maps are integrated with RGB images to provide robust descriptors, even when wide baselines are present. We show how depth information can highly improve matching in wide-baseline contexts with respect to state of the art descriptors.

Once the system is completely calibrated, we extract the human silhouette from each color image, we use it to segment the corresponding depth map and we reproject the depth-pixels into the scene. An intersection of the reprojected points gives the 3D volumetric reconstruction of the actor in the scene. The use of a 3D reconstruction technique prior to any analysis or recognition routine, allows the recognition system to work directly on 3D data. Problems like viewpoint dependencies and motion ambiguities are inherently solved.

Morphological Operators

Frame-by-frame 3D representations of the scene in terms of voxels have been the input data for any other successive analysis and processing. Considering the power and the applicability of 2D morphological operators and algorithms [38], [106], we extend the most common operators to 3D, focalizing our attention on human body analysis and action classification as possible applications. In particular we extend the morphological skeleton extraction algorithm to 3D and we develop a new 3D thinning algorithm.

In shape analysis, the *topological skeleton* (TS) is defined as the locus of centers of maximal (inscribed) open balls (or disks in 2D) [66]. In 2D it is a set of curves, while in the 3D case the corresponding object is called *surface skeleton*, because it contains not only curves, but also surface patches [23]. In many applications, however, a concise representation of 3D objects with curve arcs or straight lines is desirable because of its simplicity. This line-like representation of a 3D object is called *curve skeleton* and it is a simplified one voxel wide representation of the original 3D object, consisting only of curves. Skeletons have several different mathematical definitions in the technical literature, [38], [106], [37] (see Chapter 5 for a summary). In the following we refer to *topological skeleton* (surface or curve) as the thin version of a shape that is equidistant from the boundaries and to *morphological skeleton* (MS) as the result of the 3D extension of the morphological skeleton extraction algorithm, as described in 2D by Gonzalez and Woods [38].

The 3D thinning algorithm is a morphological algorithm developed in order to give

as output an approximation of the topological curve skeleton. Several methods have been also suggested to compute and use the curve skeleton, but no one is able to completely solve the problem. Thinning is a frequently used method for extracting skeletons in discrete spaces. Most of the existing algorithms, however, are difficult to implement due to either complex checking conditions or lengthy matching templates involved [128]. Moreover the main disadvantage of the extracted curve skeleton is its sensitivity to noise. We focus on the design of a new 3D thinning algorithm for the computation of the curve skeleton of 3D digital objects. Our algorithm provides good results, preserves topology, is easy to implement and shows better noise robustness respect to the state of the art methods [93]. We also show that its application improves the performance of our action classification system.

1.3 Application Scenarios

The main methodologies proposed in this thesis are evaluated through a number of simulations and experiments, within the context of human-computer interaction. In particular, we consider two application scenarios, both with a 3D scene reconstruction system involving a single actor in an indoor scene: Surface Reconstruction and Human Action Recognition.

Surface Reconstruction

Surface reconstruction of a static 3D human pose is a Computer Vision problem. It has many possible applications in computer graphics (we think on an avatar representation) and reverse engineering [100]. This work aims to provide a method to reconstruct the actor body surface that is accurate and computationally inexpensive. In particular, using a net of Kinect devices, the resolution is on the order of centimeters and it is too low for many algorithms developed in the literature [2], [100]. However, working with morphological operators, no resolution requirements are imposed. This frees us from the type of devices used for the acquisition. Several surface extraction morphological algorithms can be used having the volumetric reconstruction as voxel-set. The result is similar to a set of scattered points sampled on a physical shape. We consider the problem of reconstructing the surface from this unorganized point cloud, having the advantage of knowing the object volumetric occupancy. The goal is the surface representation as a 3D function.

Extending the morphological skeleton extraction algorithm to 3D produces an approximation of the surface skeleton, that is used for surface rendering as union of balls. This representation is also hierarchically interpretable, a useful property in many applications. Moreover, having the center and radius of each ball (Medial Axis Transform), together with the sampled surface, allows to represent the surface as an iso-surface of a 3D function defined as linear combination of elementary functions with radial support. This method is inexpensive, in terms of computational time, and does not depend on the algorithm used to estimate the center and radius of the inscribed balls (see Chapter 6 for more details and experimental results).

Human Action Recognition

Automatic classification of sequences of human body movements into a pre-defined vocabulary of gestures is a Pattern Recognition Problem. Vision-based gesture recognizers allow the development of non-cooperative applications able to sense human movements at a distance with low-power and rather inexpensive devices. On the other hand, the problem of human gesture modeling, under a Pattern Recognition perspective, imposes a deep analysis of the structure of human body movements. In particular, using all the instruments previous developed, we investigate how to build spatio-temporal models of human actions. These models can support categorization and recognition of simple action classes, independently from body position and orientation inside the scene, actor gender and body sizes. Starting from sequences of human body reconstructions corresponding to particular actions, we first represent these four-dimensional patterns using Motion History Volumes (MHV) [117], to encode the history of motion occurrences in space and to make the system independent to the execution speed (impetus of the action). Moreover, the MHV is computed after a thinning and reconstruction procedure applied on each frame. This highlights the movement and reduces the body shape dependence, incrementing the similarity between sequences representing the same action, even if performed by actors with different gender or different body structure. Motion descriptors are then extracted. These descriptors must be invariant respect to locations, orientations and sizes. Similarity Invariant Descriptors, extended on 3D patterns, are used for this purpose. The classification, using a nearest neighbor classifier based on Mahalanobis distance, shows good results and the improvements using our thinning algorithm are demonstrated by the classification rate.

1.4 Original Contributions

The main results obtained in this work can be summarized as follows:

- Development of an innovative acquisition system based on color and depth images (using low cost Kinect devices) reducing the number of devices commonly used in multi-camera systems. Depth maps allow to reconstruct the 3D scene also in its concave parts, while using silhouettes extracted from color images only the Convex Hull is obtainable.
- Proposal of a novel approach, based on Similarity Invariant Descriptors, to define putative correspondences between images, where the integration between depth maps and color images strengths matching capabilities for wide base acquisition setups.
- Introduction of an unified theory for dealing with different types of analysis problems based on a 3D reconstruction of the scene. In particular, we develop morphology operators and algorithms for binary 3D images. They can be applied independently from the tools used to obtain the 3D reconstruction, from the surface sampling accuracy, and from the object represented.
- Definition of the structuring elements of a new 3D-thinning algorithm, developed in order to obtain an approximation of the curve skeleton that is: homotopic with the original object, geometrically centered within the object boundary, as close as possible to the object shape, smooth, robust to noise on the surface, as thin as possible, and efficient.
- Development of a 3D morphological operator to obtain a fast and accurate surface representation of the reconstructed object. This approach is suitable for a filtering procedure that allows to extract the most important components of the object, moving towards a hierarchical representation.
- Implementation of a novel approach for the representation of the action pattern (each sequence of reconstructed 3D poses is considered globally, avoiding the problem of frame alignment) that allows to highlight the movement while reducing the body shape dependence (gender or body structure). This approach improves the success in action classification.

1.5 Application Fields

The solutions proposed in this work find potential applications in several fields. As for the 2D case, morphological operators can in theory be applied to all the cases in which a discrete volumetric reconstruction is available, no matter the resolution or the object represented. In particular, curve skeleton is useful for many Computer Vision applications (such as virtual navigation, recognition, classification, surface approximation, shape analysis and metamorphosis, and animation) because it provides a local lower dimensional characterization of the rigid or non-rigid object.

Potential applications of human action classification projects can be easily found in the fields of automatic video surveillance, human-computer interaction researches, motion based medical diagnosis, and robot skill learning. Automatic recognition and classification of suspicious or dangerous movements in a domestic environment (domestic assistance at a distance) is perhaps one of the most important recent needs demanding for applications related to human action recognition technology.

The surface reconstruction method has many possible applications in computer graphics, human-computer interaction, immersing gaming, and reverse engineering.

The multi-Kinect calibration could be useful for many applications, as immersing gaming as well: the use of multiple devices can capture and reproduce the environment more accurately than using only one device, making the game experience more realistic and amazing.

1.6 Thesis Outline

This work spans the entire pipeline of the proposed method for the actor representation and the classification of human actions captured with a multi-Kinect system (as shown in Fig. 1.1). The thesis is organized as follows.

Chapter 2 presents the calibration procedure to estimate both intrinsic and extrinsic parameters. Robust stereo-correspondences between images acquired with wide baseline are estimated using the information provided by the corresponding depth maps.

Chapter 3 is devoted to the procedure for obtaining the 3D data. Starting from the extraction of the actor silhouette on the RGB image, this mask is applied on the corresponding depth map and the surveyed depth points are then reprojected and meshed into the scene.

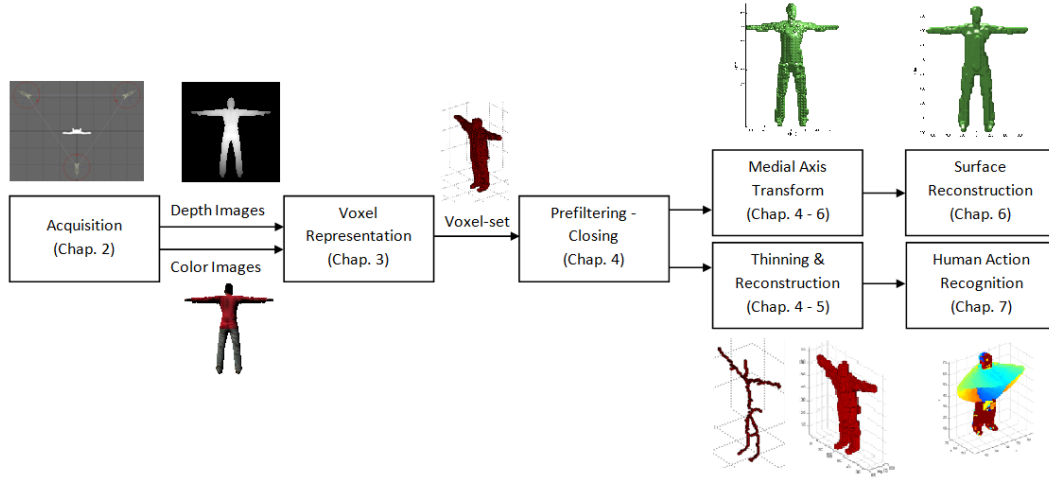


Figure 1.1: Block diagram of the proposed method for actor representation and classification of human actions captured with a multi-Kinect system.

Chapter 4 describes the developed 3D morphological operators and algorithms. After an introduction on digital topology, some useful morphology processes are presented, mainly focusing on 3D thinning algorithms.

In Chapter 5 the new 3D thinning algorithm designed to extract curve skeleton (in a fully automatic process without any parameter selection) is described. Moreover, the properties of a desirable curve skeleton are discussed, demonstrating that our thinning algorithm performs well if compared with the state of the art thinning algorithms designed to extract curve skeleton [93].

In Chapter 6 the representation of the body surface as union of balls (Medial Axis Transform) or as zero level of a 3D function is discussed, starting from a 3D extension of the morphological skeleton extraction algorithm. This result is compared with one of the most used approaches based on Voronoi-diagram [2].

Chapter 7 describes in detail our human action classification system, reporting the results of simulations and experiments conducted considering the application scenario introduced in Chapters 2 and 3, focusing on the improvement obtained using some morphological operators described in Chapters 4 and 5.

Finally, Chapter 8 draws the conclusions and outlines possible directions of future researches.

1.7 List of Publications

The most important publications relative to the work presented in the thesis are listed in the following:

- S. Milani, E. Frigerio, M. Marcon, S. Tubaro, “*Denoising Infrared Structured Light DBR Signals Using 3D Morphological Operator*”, in proceedings of IEEE 3DTV Conference 2012: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), October 2012;
- M. Marcon, E. Frigerio, A. Sarti, S. Tubaro, “*3D Correspondences in textured Depth-maps through Planar Similarity Transform*”, 2012 IEEE International Conference on Emerging Signal Processing Applications (ESPA), January 2012;
- M. Marcon, E. Frigerio, A. Sarti, S. Tubaro, “*3D Wide Baseline Correspondences using Depth-maps*”, Signal Processing: Image Communication Journal, Elsevier, Vol. 27, Issue 8, Page(s) 849-855, September 2012;
- M. Marcon, E. Frigerio, A. Sarti, S. Tubaro, “*3D Wide Baseline Correspondences using Depth-maps*”, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Springer, Vol. 79, Page(s) 194-203, 2012.

2 Acquisition System

In this Chapter, the acquisition system for capturing the actor performance from different viewpoints is described together with the procedure for the estimation of the calibration parameters used in the following reconstruction phase.

The system is developed in the Image and Sound Processing Group (ISPG) laboratory of the Politecnico di Milano. It is thought to obtain a 3D reconstruction of the action performed. The choice between a monocular or a multiple view system can affect the system complexity both in design and in realization. A monocular system has less data to be processed but needs complex algorithms and models in order to retrieve all the information lost in a 2D projection of a 3D scene. A multiple view setting allows the reconstruction of the body using a larger amount of data. Considering the lowering of the cost of 3D acquisition devices and the more complete representation of the scene, multiple view systems can be considered nowadays the natural choice for vision-based gesture recognizers design. The use of a set of cameras can greatly improve the performance of the whole system [21], [26], [49]. In fact, a multi-viewpoint input device makes the system inherently able to solve viewpoint dependence, motion ambiguities and self-occlusions. The price to pay for this type of system is the increased complexity in input management and a larger amount of data to be stored and processed. Multi-camera systems, which constituted the standard for dynamic 3D scenes reconstruction for many years [95], [111], [117] are being replaced with multi-DIBR (Depth Image Based Rendering) device systems. Many low cost realtime depth map reconstruction devices, recently appeared on the market, have opened new opportunities for the Computer Vision community to integrate these information in many research areas. In particular, a multi-Kinect acquisition system is developed in the ISPG laboratory.

Before being able to reconstruct the 3D scene, the system must be calibrated: both intrinsic and extrinsic parameters must be known. The block diagram of the implemented calibration procedure is represented in Fig. 2.1. For the intrinsic cal-

2 Acquisition System

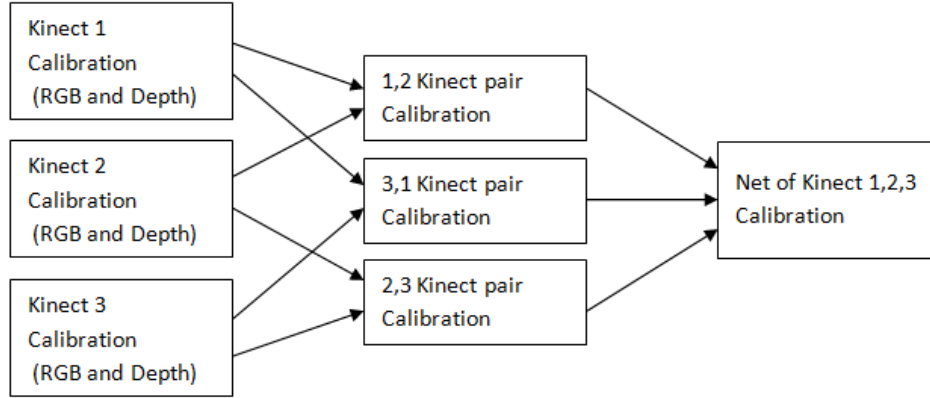


Figure 2.1: Block diagram of the implemented calibration procedure developed for a net of 3 Kinect sensors.

ibration we follow the approach proposed by Herrera *et al.* [48] that requires only a planar surface on which it is printed a chessboard to be acquired from various poses. For the extrinsic calibration the most commonly adopted method is to find stereo-correspondences between images, to extract the Fundamental or the Essential matrix between a couple of cameras, and to estimate the canonical camera calibration matrices [42]. We apply this method to the 3 couples of Kinect (1and2, 2and3, and 3and1) and refine the solution using a Bundle Adjustment algorithm [113]. The innovative contribution in the process developed is to find robust stereo correspondences for wide baseline acquisitions. Robust stereo-correspondences between images acquired with wide baseline are estimated using the information provided by the corresponding depth maps. The knowledge of the underlying depth map together with a visual snapshot of the scene can greatly improve the robustness of points matching between different views even for wide baseline acquisitions. In this Chapter we are presenting how visual correspondences from different views can be identified by robust Similarity Invariant Descriptors (SIDs) once their laying plane is known (extrinsic calibration process). Depth maps, providing a rough geometrical description of the underlying scene, allow to select only feature points belonging to almost planar regions, skipping geometrical corners or edges that undergo non-linear distortions for viewpoint changes. The proposed SIDs keep much more information of the original area with respect to commonly used affine invariant descriptors, like Scale Invariant Feature Transform (SIFT), making the proposed approach much less prone to false matches even for wide viewpoint changes.

This Chapter is organized as follows. Section 2.1 introduces the acquisition setup

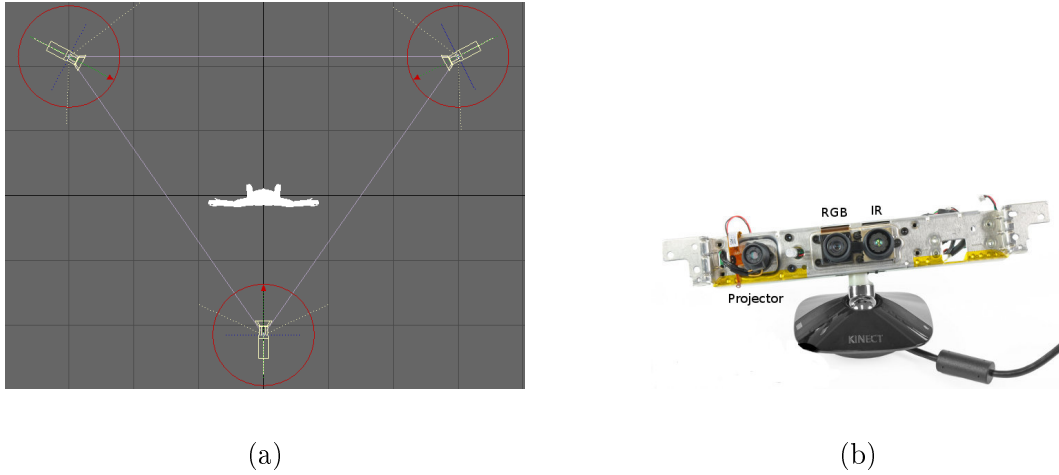


Figure 2.2: Smart Space. (a) A graphic scheme of the acquisition setup with 3 devices and an actor in the scene. (b) A Kinect device.

developed in the Image and Sound Processing Group (ISPG) laboratory of the Politecnico di Milano. Then the complete procedure for the system calibration is described: calibration of each Kinect device in Section 2.2, calibration of a pair of Kinect devices in Section 2.3 and the complete calibration of the 3 devices in Section 2.4. Section 2.5 shows the discriminative power of the proposed SID with respect to the state of the art SIFT.

2.1 Acquisition Setup

Inside the Image and Sound Processing Group laboratory there is a room called “Smart Space” built for realizing synchronous video acquisitions of objects and action scenes. The acquisition set up is composed by a metal structure of size $6 \times 4.3 \times 2.5$ m. The walls have blue curtains for chroma keying segmentation and the floor has blue moquette for easier background subtraction operation. The illumination is uniform through a set of diffused light sources. These precautions help to reduce the shadows impact during the image segmentation. A multi-Kinect structure with three viewpoints is used for data acquisition. All the devices are mutually calibrated and placed at the same height (1.5 m) at the vertexes of an isosceles triangle with base equal to 6 m and height equal to 4.3 m (Fig. 2.2(a)). The triangle is almost equilateral because the angles are equal to 55° , 55° and 70° . Each device looks directly to

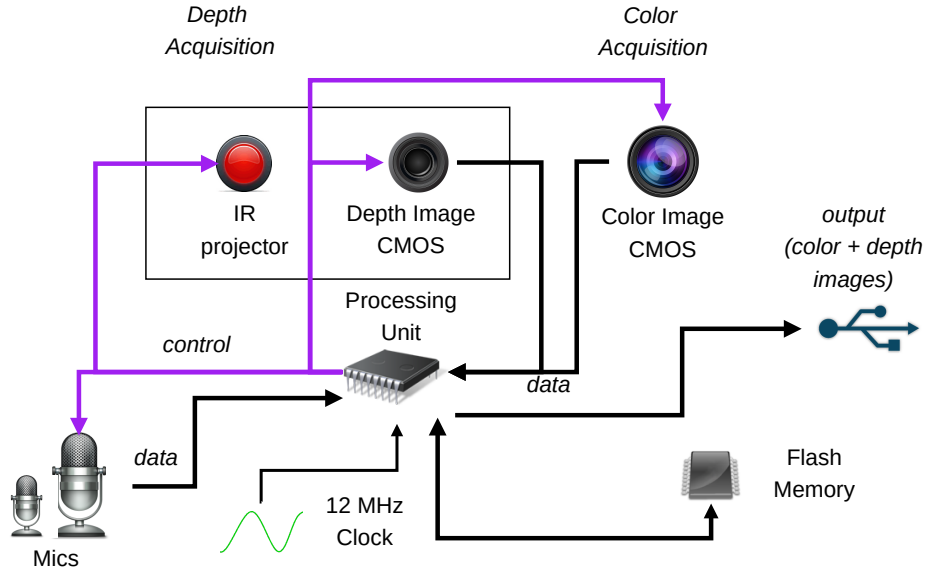


Figure 2.3: Block diagram of the MS Kinect sensor.

the center of the room and sees the whole actor. This structure is completely new respect to the one we used in past years that was made of multiple cameras [95].

2.1.1 The Microsoft Xbox 360 Kinect Device

The MS Xbox Kinect device (Fig. 2.2(b)) is a game controller realized for the Xbox 360 console. Thanks to its low cost and good performance, this device is having success in many other fields, not only related to video-games and entertainment. Fig. 2.3 shows a simplified block diagram of this device. Color information is available, since an RGB CMOS camera permits to obtain a standard picture (480×640) of the acquired scene (Fig. 2.4(a)). Depth information of the acquired scene is available, as well, since an IR depth sensor permits to obtain an *elevation* depth map (480×640), where each pixel has a gray tone proportional to the distance of the acquired 3D point from the IR camera image plane (Fig. 2.4(b)). The implemented IR depth sensor consists in an IR projector, an IR CMOS camera, and a processing unit that controls them and elaborates the acquired signal.

The algorithm used to obtain depth information is based on the triangulation between the ray emitted by the IR projector and its reflection from the scene to the IR camera image plane [101]. In particular, a constant IR pattern of dots is projected by the IR projector on the scene, and the IR CMOS camera acquires the reflected



Figure 2.4: Examples of (a) an RGB image and (b) the corresponding depth map acquired with a Kinect.

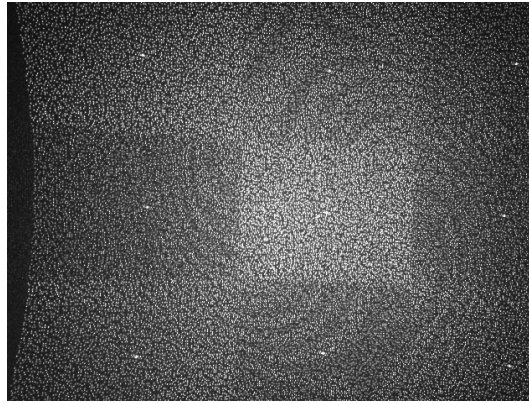


Figure 2.5: Kinect IR pattern used for the depth map estimation (available at <http://www.ros.org/cgi-bin/mt-4.3/mt-tb.cgi/279>).

pattern, which will be distorted according to the geometry of the objects (see Fig. 2.5 for a typical example). The central processing unit estimates the disparity between the observed pattern and a pre-recorded image at a constant depth. The output consists of an image of scaled disparity values.

In order to better explain the depth estimation method, the single-point triangulation principle can be considered [101]. Fig. 2.6 shows the configuration for a single point. O_p and O_c are the IR projector and camera optical centers respectively, d represents the baseline, and z_p and z_c are the projector and camera optical axes. The projected dot reaches the object surface at point S and is back scattered at

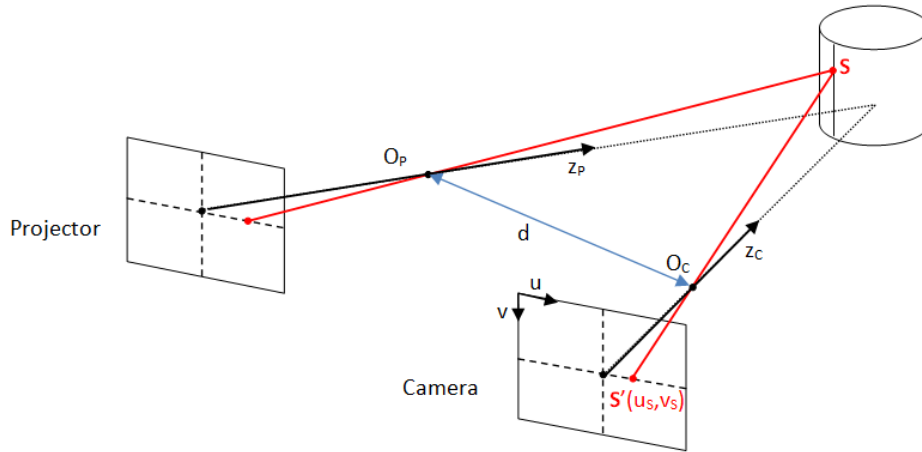


Figure 2.6: Scheme of the triangulation principle. O_p and O_c are the IR projector and camera optical centers respectively, d represents the baseline, and z_p and z_c are the projector and camera optical axes.

point S' on the IR camera image plane. The measurement of the location (u_s, v_s) of the image point S' defines the line of sight $S'O_c$. Since the ray $S'O_p$ is known, the intersection between the two rays is the point S . Knowing the distance of S from the IR camera image plane, the gray tone of S' is assigned.

Working with the triangulation principle and using an IR pattern, the Kinect has some weaknesses: the sunlight blinds the IR depth sensor (see Fig. 2.7 (top row)), and the projected pattern should not be significantly changed by the reflection of the object surface (it changes depending on the incident angle on the surface or on the kind of surface). Transparent or shiny surfaces are not seen by the IR sensor and appear white in the depth map (as the bottle of water in Fig. 2.7 (middle row), or the table surface in Fig. 2.7 (bottom row)). There are also missing data in correspondence of occlusions and shadows (some examples are available at Milani's web page [81]).

2.1.2 Time Synchronization

Video sequences, both color images and depth maps, coming from different devices provide a scene description from different viewpoints. In order to obtain the 3D reconstruction of the scene, it is necessary to know position and orientation of all the devices respect to each other. An important aspect in the reconstruction of 3D scenes through multiple devices is the time synchronization between them. Working with Kinect devices, we are able to record synchronized depth and color streams

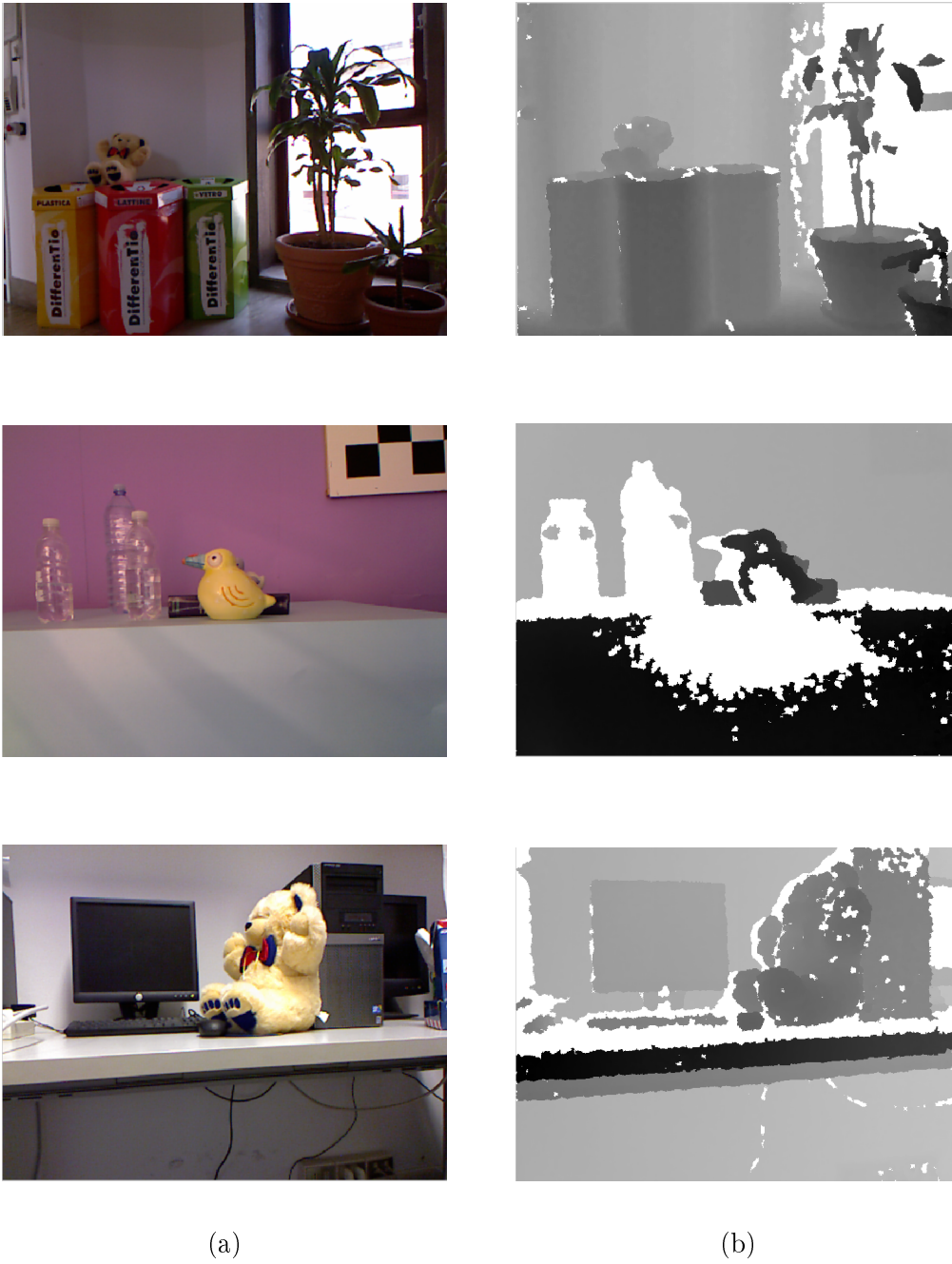


Figure 2.7: Examples of (a) RGB images and (b) the corresponding depth maps acquired with a Kinect.

2 Acquisition System

from each Kinect sensor. The device provides RGB and depth data that are not hardware synchronized. However, the OpenNI framework SDK, with its *Point Cloud Library*, allows to synchronize the frames on the client side finding the best possible combination of frames, resulting in a lag up to 16 ms between depth and RGB frames. This is sufficiently accurate considering the speed of the recorded movements.

The implemented system for acquiring synchronized multi-view color and depth data using multiple Kinect devices follows the idea proposed by Ahmed [1]. First of all each Kinect is connected to a dedicated machine. The hardware setup is identical for each Kinect, which assures that the data transfer and processing will take place at the same speed. Each machine is also synchronized to a web-server (<http://www.time.is>), so that all the internal clocks are synchronized (within a precision of 20 ms), all reporting the exact time. For data synchronization all the machines are programmed to start recording at the exact same time and also the number of frames to record is fixed. Since the Kinect frame rate, the total number of frames to record, and the recording start time are known, exact clock ticks for each frame (both RGB and depth) are calculated in advance. Each machine then uses the synchronous interface at the precalculated time to query for depth and RGB data. In order to minimize the I/O overhead, both RGB and depth are stored in a buffer and, once the recording is finished, are written in the disk. Since all the machines are querying the data at the exact time, the acquisition drift is kept under check.

2.2 Kinect Calibration

For the Kinect calibration we follow the method proposed by Herrera *et al.* [48] that requires only a planar surface on which is printed a chessboard to be acquired from various poses. The complete model (internal and external calibration parameters) includes $20 + 6N$ parameters, where N is the number of calibrated images.

Color camera calibration has been studied extensively [46] [133]. For the color camera intrinsics, we use a similar intrinsic model as Heikkilä and Silven [46], which consists of a pinhole model with radial and tangential distortion corrections. The projection of a point from color camera coordinates $\mathbf{x}_c = [x_c, y_c, z_c]^T$ to color image coordinates $\mathbf{p}_c = [u_c, v_c]^T$ is obtained through the following equations. The point is

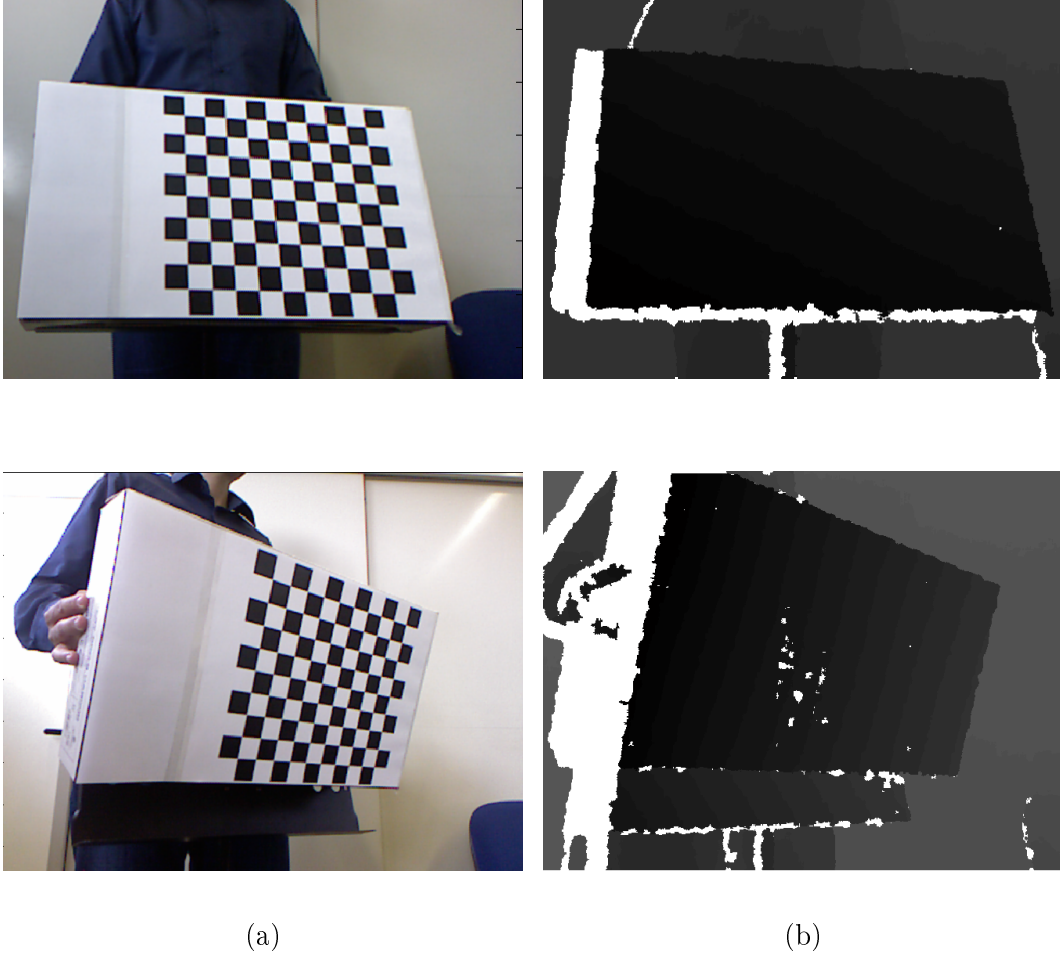


Figure 2.8: Examples of chessboard acquired from different viewpoints for the Kinect calibration: (a) RGB images, (b) depth images.

first normalized by $\mathbf{x}_n = [x_n, y_n]^T = [x_c/z_c, y_c/z_c]^T$. Distortion is then performed:

$$\mathbf{x}_g = \begin{bmatrix} 2k_3x_ny_n + k_4(r^2 + 2x_n^2) \\ k_3(r^2 + 2y_n^2) + 2k_4x_ny_n \end{bmatrix} \quad (2.1)$$

$$\mathbf{x}_k = (1 + k_1r^2 + k_2r^4)\mathbf{x}_n + \mathbf{x}_g, \quad (2.2)$$

where k_1, k_2, k_3, k_4 are coefficients for radial distortion, and $r^2 = x_n^2 + y_n^2$. Finally

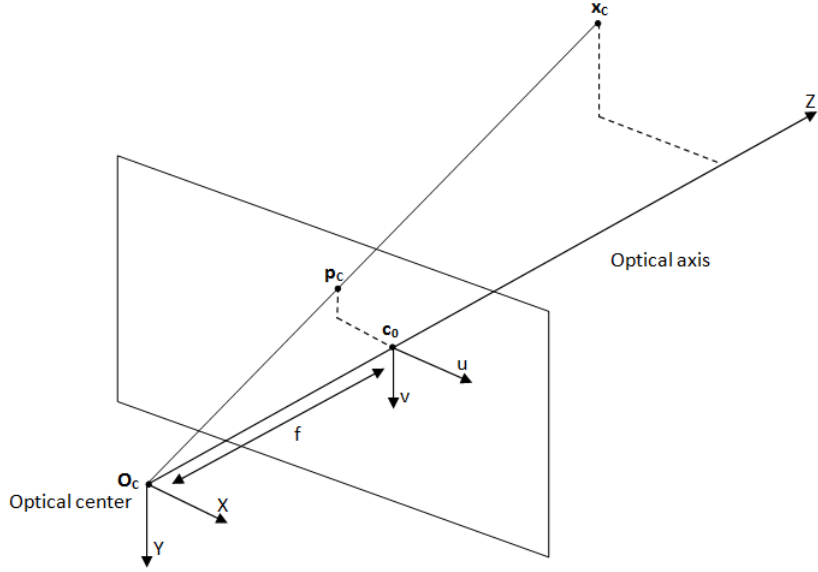


Figure 2.9: Representation of the camera pin-hole model avoiding distortions. $\mathbf{x}_c$ is the 3D point in camera coordinates (z axis corresponds to the camera optical axis), $\mathbf{p}_c$ is its projection on the image plane, $\mathbf{c}_0$ is the principal point, and O_c is the optical center.

the image coordinates are obtained:

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} = \begin{bmatrix} f_{cx} & 0 & u_{c0} \\ 0 & f_{cy} & v_{c0} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix}, \quad (2.3)$$

where (f_{cx}, f_{cy}) is the effective focal length expressed in pixel size (the camera focal length is unique but it is usually assumed that pixels height is different from their width), and (u_{c0}, v_{c0}) is the image center, also called the principal point. The classic pin-hole camera model is represented in Fig. 2.9. The complete color camera model is described by 8 parameters $L_c = \{f_{cx}, f_{cy}, u_{c0}, v_{c0}, k_1, k_2, k_3, k_4\}$.

The transformation between depth camera coordinates $\mathbf{x}_d = [x_d, y_d, z_d]^T$ and depth image coordinates $\mathbf{p}_d = [u_d, v_d]^T$ follows the same model used for the color camera. Since the distortion coefficients are estimated with very high uncertainty and the distortion correction does not improve the reprojection, Herrera *et al.* [46] do not use distortion correction for the depth image. The relation between the disparity

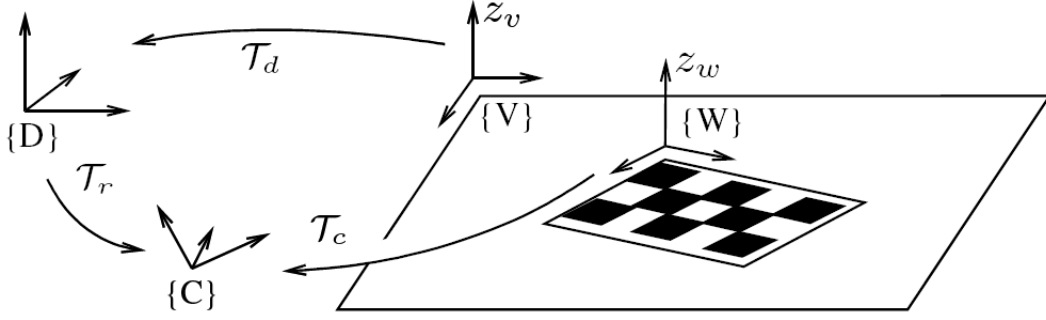


Figure 2.10: Reference frames and transformations present on a scene. $\{C\}$ and $\{D\}$ are the color and depth cameras' reference frames respectively. $\{V\}$ is the reference frame anchored to the calibration plane and $\{W\}$ is the world reference frame anchored to the calibration pattern.

value d and the depth z_d is modeled using the equation:

$$z_d = \frac{1}{\alpha(d - \beta)}, \quad (2.4)$$

where α and β are part of the the depth camera intrinsic parameters to be calibrated. The model for the depth camera is described by 6 parameters $L_d = \{f_{dx}, f_{dy}, u_{d0}, v_{d0}, \alpha, \beta\}$.

About the extrinsic parameters calibration and the relative pose estimation, the goal is to find the relationship between all the different reference frames present in a scene as shown in Fig. 2.10. Points from one reference frame can be transformed to another one using a rigid transformation denoted by $T = \{R, \mathbf{t}\}$ where R is a rotation and $\mathbf{t}$ a translation. The relative pose T_r is constant, while each image has its own pose T_c , resulting in $6 + 6N$ pose parameters.

Herrera *et al.* [46] use the corners of a planar checkerboard pattern for calibration. The checkerboard corners provide suitable constraints for the color image, while the planarity of the points provides constraints on the depth image.

2.2.1 RGB Camera Calibration

Briefly they use Zhang's method [133] to initialize the camera parameters: the checkerboard corners are extracted from the intensity image. A homography is then computed for each image using the known corner positions in world coordinates $\{W\}$ and the measured positions in the image. Calling K the camera matrix in Eq. (2.3) and considering that the model plane lays on the plane $z_w = 0$ in the world

2 Acquisition System

coordinate system, it follows that:

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} = K[R_w | \mathbf{t}_w] \begin{bmatrix} x_w \\ y_w \\ 0 \\ 1 \end{bmatrix}. \quad (2.5)$$

Denoting the columns of the rotation matrix as $\mathbf{r}_{w,i}$:

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} = K[\mathbf{r}_{w,1}, \mathbf{r}_{w,2}, \mathbf{t}_w] \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix}. \quad (2.6)$$

The matrix $H = K[\mathbf{r}_{w,1}, \mathbf{r}_{w,2}, \mathbf{t}_w]$ represents the homography that relates a model point $\mathbf{x}_w$ (a corner on the real chessboard) with its image point $\mathbf{p}_c$ (the correspondent corner on the image). H is a 3×3 matrix, defined up to a scalar factor, and it can be easily estimated from the correspondences $\mathbf{p}_c, \mathbf{x}_w$. Each correspondence imposes two linear constraints on the elements $h_{i,j}$ of H [42]:

$$-x_w h_{2,1} - y_w h_{2,2} - h_{2,3} + v_c x_w h_{3,1} + v_c y_w h_{3,2} + v_c h_{3,3} = 0, \quad (2.7)$$

$$x_w h_{1,1} + y_w h_{1,2} + h_{1,3} - u_c x_w h_{3,1} - u_c y_w h_{3,2} - u_c h_{3,3} = 0. \quad (2.8)$$

The homography is then estimated solving a linear system of equations. Each homography imposes constraints on the camera parameters which are then solved with a linear system of equations considering that $\mathbf{r}_{w,i}$ and $\mathbf{r}_{w,j}$ are orthonormal [42].

2.2.2 Depth Camera Calibration

The same method is used to initialize the depth camera parameters selecting the four corners of the calibration plane (the whole dark box in Fig. 2.8 (b)). These corners are very noisy and are only used here to obtain an initial guess. The homography is thus computed between $\{V\}$ and $\{D\}$. This initializes the focal lengths, the principal point, and the transformation T_d . Using these initial parameters we obtain a guess for the depth of each selected corner. With this depth and the inverse of the measured disparity, an overdetermined system of linear equations is built using Eq. (2.4), which gives an initial guess for the depth parameters (α and β) [47].

2.2.3 RGB and Depth Cameras Calibration

In order to estimate T_r , having T_c and T_d , the coplanarity between $\{W\}$ and $\{V\}$ is used. We can use this information by extracting the plane equation in each reference frame and using it as a constraint. A plane is defined using its unit normal vector $\mathbf{n}$ and the distance to the origin δ :

$$\mathbf{n}^T \mathbf{x} = \delta. \quad (2.9)$$

Assuming that the reference frame $\{W\}$ is anchored to the calibration plane, the parameters of the plane in each frames are $\mathbf{n} = [0, 0, 1]^T$ and $\delta = 0$. The transformation T_c , obtained at the previous calibration step for each observation in the camera routine, gives the orientation ($\mathbf{n}_{ci}$) and the distance (δ_{ci}) of the plane $\mathbf{n}_{ci}^T \mathbf{x} - \delta_{ci} = 0$ with respect to the camera origin, for each image in the camera frame of reference. If we divide a rotation matrix into its columns $R_i = [\mathbf{r}_{i,1}, \mathbf{r}_{i,2}, \mathbf{r}_{i,3}]$, the plane parameters in camera coordinates are:

$$\mathbf{n}_{ci} = \mathbf{r}_{i,3} \quad \text{and} \quad \delta_{ci} = \mathbf{r}_{i,3}^T \mathbf{t}_i. \quad (2.10)$$

The same considerations are made for depth camera frames. As mentioned by Unnikrishnan and Hebert [116], the relative pose can be obtained in closed form from several images concatenating the plane parameters for each image in matrices of the form: $M_c = [\mathbf{n}_{c1}, \mathbf{n}_{c2}, \dots, \mathbf{n}_{cn}]$, $\mathbf{b}_c = [\delta_{c1}, \delta_{c2}, \dots, \delta_{cn}]$, and likewise for the depth camera to form M_d and $\mathbf{b}_d$. n is the number of depth-image observation pairs, and the primary subscript c (or d) denotes the camera (or depth) frame of reference respectively. Imposing the coplanarity between $\{W\}$ and $\{V\}$ for each color and depth camera acquisition, first the translation that minimizes the difference in distance from the color camera origin to each plane, represented both in the camera coordinate system and in the depth camera coordinate system, is found [116]:

$$\mathbf{t}_r = (M_c M_c^T)^{-1} M_c (\mathbf{b}_c - \mathbf{b}_d)^T. \quad (2.11)$$

Also the rotation between the reference frames that minimizes the difference between the normal from the origin to the corresponding planes in the two frames is found [116]:

$$R'_r = M_d M_c^T. \quad (2.12)$$

Due to noise, R'_r may not be orthonormal. We obtain a valid rotation matrix through Singular Value Decomposition (SVD) using: $R_r = UV^T$, where USV^T is the SVD of R'_r .

2.2.4 Optimization

The optimization stage of the calibration method aims to minimize the weighted sum of squares of the measurement reprojection errors [116]. The error for the color camera is the Euclidean distance between the measured corner position $\mathbf{p}_c$ and its reprojected position $\mathbf{p}'_c$. Whereas for the depth camera it is the difference between the measured disparity d and the predicted disparity d' obtained by inverting Eq. (2.4). Since the errors have different units, they are weighted using the inverse of the corresponding measurement variance, σ_c^2 and σ_d^2 . The resulting cost function is:

$$c = \frac{\sum [(u_c - u'_c)^2 + (v_c - v'_c)^2]}{\sigma_c^2} + \frac{\sum (d - d')}{\sigma_d^2}. \quad (2.13)$$

Note that Eq. (2.13) is highly non-linear. The Levenberg-Marquardt algorithm is used to minimize it with respect to the calibration parameters [116]. The initialization gives a very rough guess of the depth camera parameters and relative pose, whereas the color camera parameters have fairly good initial values. To account for this, the non-linear minimization is split into two phases. The first phase uses fixed parameters for the color camera L_c and external pose T_c and optimizes the depth camera parameters L_d and the relative pose T_r . A second minimization is performed over all the parameters to obtain an optimal estimation.

The whole procedure is implemented by Herrera *et al.* in a Matlab Toolbox (<http://www.ee.oulu.fi/~dherrera/kinect/>) [47].

2.3 Kinect Pair Calibration

Assuming that a set of putative correspondences between each pair of RGB images is available (see Section 2.5 for a detailed explanation of the proposed algorithm used to find robust stereo correspondences between two Kinect devices), it is possible to extract the extrinsic parameters between pairs of devices following the standard calibration method that uses stereo correspondences to extract first the Fundamental matrix, and then the rigid transformation $\{R_{ij}, \mathbf{t}_j\}$, where i denotes the device

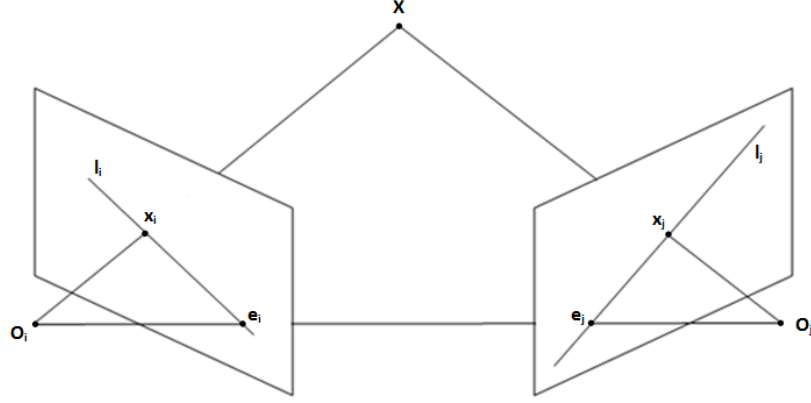


Figure 2.11: Epipolar geometry. $\mathbf{x}_j$ and $\mathbf{x}_i$ are two corresponding points, projection of $\mathbf{X}$ on the two cameras image planes. O_i and O_j are the two optical centers. l_i and l_j are the epipolar lines: intersection of the epipolar plane (plane defined by O_i and O_j and $\mathbf{X}$) with the respective camera image plane. e_i and e_j are the epipoles: intersection of the baseline with the image plane.

assumed as canonical [42].

For each couple of devices, we compute the Fundamental matrix $F_{i,j}$ using a robust estimator (RANdom Sample Consensus [112]) on the classical eight points algorithm proposed by Hartley in 1997 [41]. The Fundamental matrix is defined by the equation:

$$\mathbf{x}_j^T F \mathbf{x}_i = 0, \quad (2.14)$$

where $\mathbf{x}_j$ and $\mathbf{x}_i$ are corresponding points on the image plane of the device j and i respectively (see Fig. 2.11 for the considered geometry). Given sufficiently many point matches (at least eight), this equation can be used to compute the unknown matrix F . In particular, writing $\mathbf{x}_j = [x_j, y_j, 1]^T$ and $\mathbf{x}_i = [x_i, y_i, 1]^T$, each point match gives rise to one linear equation in the unknown entries of F :

$$x_i x_j f_{11} + x_i y_j f_{21} + x_i f_{31} + y_i x_j f_{12} + y_i y_j f_{22} + y_i f_{32} + x_j f_{31} + y_j f_{32} + f_{33} = 0. \quad (2.15)$$

From all the point matches, we obtain a set of linear equations of the form:

$$A \mathbf{f} = \mathbf{0}, \quad (2.16)$$

where $\mathbf{f}$ is a nine-vector containing the entries of the matrix F , and A is the equation

2 Acquisition System

matrix composed by vectors row coming from stereo-correspondences:

$$[x_i x_j, x_i y_j, x_i, y_i x_j, y_i y_j, y_i, x_j, y_j, 1]. \quad (2.17)$$

The Fundamental matrix F , and hence the solution vector $\mathbf{f}$, are defined only up to an unknown scale. For this reason, and to avoid the trivial solution $\mathbf{f}$, we make the additional constraint:

$$\|\mathbf{f}\| = 1, \quad (2.18)$$

where $\|\mathbf{f}\|$ is the norm of $\mathbf{f}$. We then enforce the rank 2 condition using Singular Value Decomposition (SVD).

Once the Fundamental matrix is estimated for each of the three possible couples of Kinect, the Essential matrix can be estimated (using intrinsic calibration parameters for each camera) [42]:

$$E = K_j^T F K_i, \quad (2.19)$$

where K_j and K_i are the intrinsic camera calibration matrices for the 2 devices. The Essential matrix can be decomposed using SVD and used to extract the relative position of the second device respect to the device assumed as canonical ($P_i = K_i[I|\mathbf{0}]$). Considering the SVD of E : $SVD(E) = UWV^T$, since the matrix E is an essential matrix, W must be a diagonal matrix having $w_{11} = w_{22} = w$ and $w_{33} = 0$. Collecting w , E can be rewritten as: $E = U \text{diag}\{1, 1, 0\} V^T$.

Given the matrix E , four possible configurations for the cameras are mathematically possible. E can be expressed as:

$$E = SR = [\mathbf{t}]_x R, \quad (2.20)$$

where R is the rotation matrix and $S = [\mathbf{t}]_x$ is the emisymmetric matrix from which the translation vector is estimated:

$$[\mathbf{t}]_x = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_e & 0 & -t_1 \\ -t_2 & t_1 & 1 \end{bmatrix}. \quad (2.21)$$

The camera calibration matrix will be: $P_j = K_j[R|\mathbf{t}]$. Defining $R_{zp}(\frac{\pi}{2})$ and $R_{zn}(\frac{-\pi}{2})$

two orthonormal matrices representing a rotation of $\pm \frac{\pi}{2}$ around the z axis:

$$R_{zp}\left(\frac{\pi}{2}\right) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.22)$$

$$R_{zn}\left(-\frac{\pi}{2}\right) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.23)$$

the possible solutions satisfying Eq. (2.20) are:

$$S_1 = UR_{zp}diag\{1, 1, 0\}U^T \quad R_1 = UR_{zp}^T V^T \quad (2.24)$$

$$S_2 = UR_{zn}diag\{1, 1, 0\}U^T \quad R_2 = UR_{zn}^T V^T. \quad (2.25)$$

Computing S_1 and S_2 and considering that the columns of U , expressed as $[\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3]$, are orthonormal, the two possible solutions for $\mathbf{t}$ are:

$$\mathbf{t}_1 = \mathbf{u}_3 \quad (2.26)$$

$$\mathbf{t}_2 = -\mathbf{u}_3. \quad (2.27)$$

Therefore, for a given Essential matrix E and a first camera matrix $P_i = K_i[I|\mathbf{0}]$, there are four possible choices for the second camera matrix P_j :

$$P_{j,1} = K_j[R_1|\mathbf{u}_3] \quad (2.28)$$

$$P_{j,2} = K_j[R_1|-\mathbf{u}_3] \quad (2.29)$$

$$P_{j,3} = K_j[R_2|\mathbf{u}_3] \quad (2.30)$$

$$P_{j,4} = K_j[R_2|-\mathbf{u}_3]. \quad (2.31)$$

From the four solutions proposed above, only one ($[\hat{R}|\hat{\mathbf{t}}]$) allows to reconstruct geometrically the 3D scene such as it will be in front of the image planes of both cameras (see Fig. 2.12).

In order to improve the parameters estimation, a minimization process is then

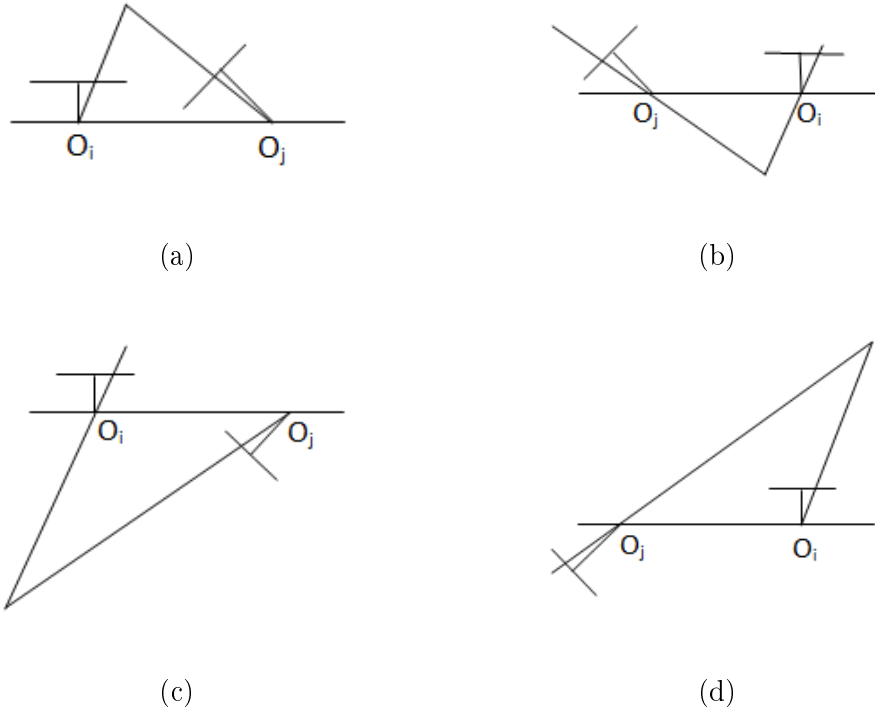


Figure 2.12: Geometric representation of the four solutions obtainable with the Kinect pair calibration algorithms. Only the solution (a) is geometrically coherent because the reconstructed point is in front of both the cameras.

implemented that minimizes the cost function [42]:

$$e = \sum_{k=1}^N \{d(\mathbf{x}_j^{(k)}, \hat{F} \mathbf{x}_i^{(k)})^2 + d(\mathbf{x}_i^{(k)}, \hat{F}^T \mathbf{x}_j^{(k)})^2\}, \quad (2.32)$$

where $\hat{F}$ is the estimated Fundamental matrix $\hat{F} = K_i^{-T} \hat{E} K_j^{-1}$.

2.4 3 Kinect Net Calibration

In order to obtain a complete system calibration, we implement a Bundle Adjustment algorithm [42] that allows to adjust the rays that link the optical center of each device with each 3D reconstructed point. In particular the matrices $\hat{P}_i$ of each device and

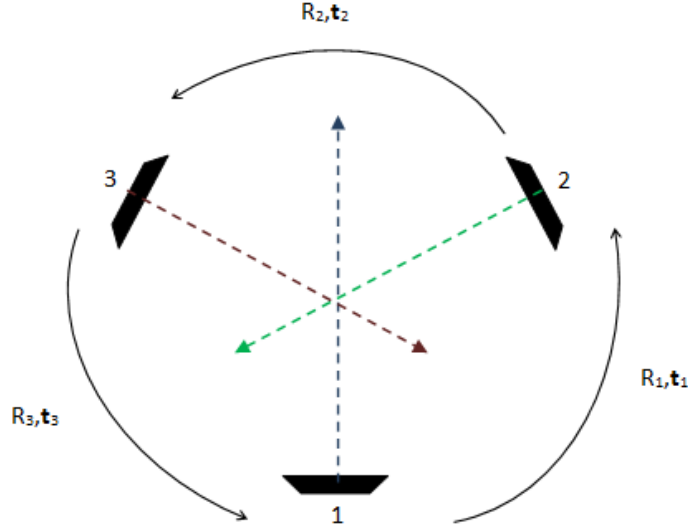


Figure 2.13: Closed motion configuration. Each couple of devices is considered in counterclockwise order, considering always the first device as canonical.

the points $\hat{\mathbf{X}}^j$ are chosen so that the cost function:

$$c = \sum_{i,j} d(\hat{P}_i \hat{\mathbf{X}}^j, \mathbf{x}_i^j)^2 \quad (2.33)$$

is minimum. We work in a *closed motion* configuration: looking at the three devices of Fig. 2.13, each couple is considered in counterclockwise order, taking always the first device as canonical. At the end of the initialization procedure, to each branch of the calibration path is associated a pair of extrinsic matrices, in canonical form, where $[R_m | \mathbf{t}_m]$ represents the m -th branch. Considering a generic point $\mathbf{X}_m$ represented in the m -th Kinect reference system, it can be represented in the $m+1$ -th Kinect reference system as:

$$\mathbf{X}_{m+1} = R_m \mathbf{X}_m + \mathbf{t}_m. \quad (2.34)$$

Considering our net of 3 Kinect devices and one point $\mathbf{X}$, $\mathbf{X}_1$ is the point expressed in the Kinect 1 system reference, $\mathbf{X}_2$ is the same point expressed in the Kinect 2 system reference, and $\mathbf{X}_3$ is the same point expressed in the Kinect 3 system reference. $\mathbf{X}_1$ in relationship with the second device is:

$$\mathbf{X}_2 = R_1 \mathbf{X}_1 + \mathbf{t}_1. \quad (2.35)$$

2 Acquisition System

Computed $\mathbf{X}_2$, we can represent it in relationship with the third device as:

$$\mathbf{X}_3 = R_2\mathbf{X}_2 + \mathbf{t}_2. \quad (2.36)$$

Combining the above equations:

$$\mathbf{X}_3 = R_2R_1\mathbf{X}_1 + R_2\mathbf{t}_1 + \mathbf{t}_2, \quad (2.37)$$

where the extrinsic parameters of the third device, respect to the first as canonical, are:

$$R_{13} = R_2R_1, \quad (2.38)$$

$$\mathbf{t}_{13} = R_2\mathbf{t}_1 + \mathbf{t}_2. \quad (2.39)$$

The notation R_{ij} means that the i -th device is considered as canonical (meaning that during the Kinect pair calibration procedure we have estimated $R_3 = R_{31}$, $R_2 = R_{23}$, and $R_1 = R_{12}$).

Considering only one device (the device on the bottom of Fig. 2.2(a)) as canonical for the system, all the extrinsic matrices and all the reprojected points can be expressed in its reference system. Moreover, we can express the canonical device as a composition of all the devices as:

$$R_c = \prod_{i=3}^1 R_i = I \quad (2.40)$$

$$\mathbf{t}_c = \sum_{i=1}^2 \left(\prod_{l=3}^{i+1} R_l \right) \mathbf{t}_i + \mathbf{t}_3 = \mathbf{0}. \quad (2.41)$$

Due to calibration errors $R_c \neq I$ and $\mathbf{t}_c \neq \mathbf{0}$. Moreover, considering the 3D point in reference coordinate system $\mathbf{X} = [X, Y, Z]^T$, a *closing error* will occur:

$$e_c = \|\mathbf{X} - \hat{\mathbf{X}}\|^2, \quad (2.42)$$

where $\hat{\mathbf{X}} = R_c\mathbf{X} + \mathbf{t}_c$. The estimation of extrinsic parameters can be improved minimizing the cost function:

$$\sum_{j=1}^K \|\mathbf{X}^j - \hat{\mathbf{X}}^j\|^2, \quad (2.43)$$

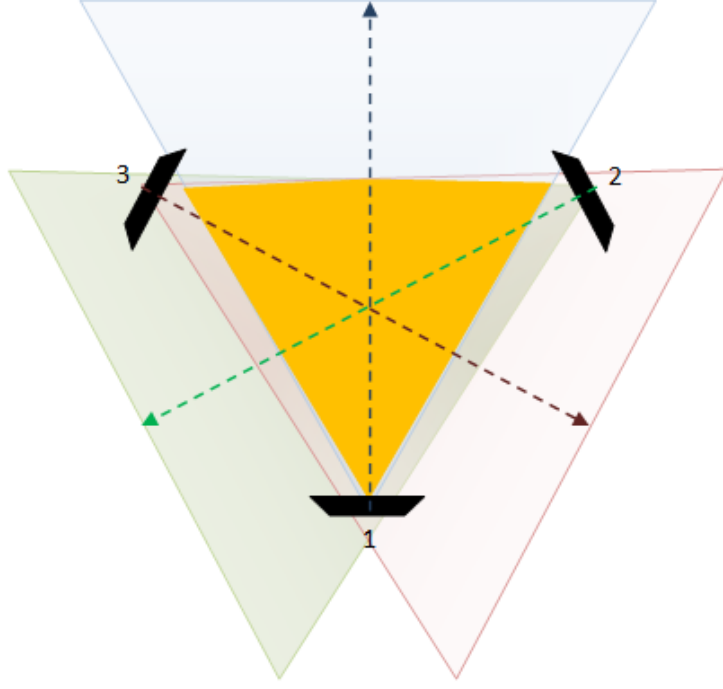


Figure 2.14: Representation of the scene acquired by the proposed net of 3 calibrated Kinect devices. Each device points at the center of the overlapping area. Horizontal FOV and optical axes are represented for each device. Kinect labeled as 1 is the canonical device.

where K indicates the number of 3D points reconstructed. In order to insert in the minimization process constraints related to the calibration data acquired too, the cost function to be minimized (respect to $[R_m | \mathbf{t}_m]$) becomes [113]:

$$\alpha \sum_{m=1}^M e_{m,m+1} + \beta \sum_{j=1}^K \|\mathbf{X}^j - \hat{\mathbf{X}}^j\|^2, \quad (2.44)$$

where $M = 3$ is the number of devices to calibrate, $e_{m,m+1}$ is the error of the epipolar relationship between the devices m and $m + 1$ ($m + 1 = 4$ corresponds to the device 1) as defined in Eq. (2.32), α and β are multiplicative factors that empathize the closing motion error or the epipolar error computed on the data used for the calibration process. This cost function is minimized using the Nelder and Mead algorithm [88]. At the end of the minimization process the 3 depth camera matrices as $P_m = K_m[R_m | \mathbf{t}_m]$ are known and we are able to reconstruct the captured scene as represented in Fig. 2.14.

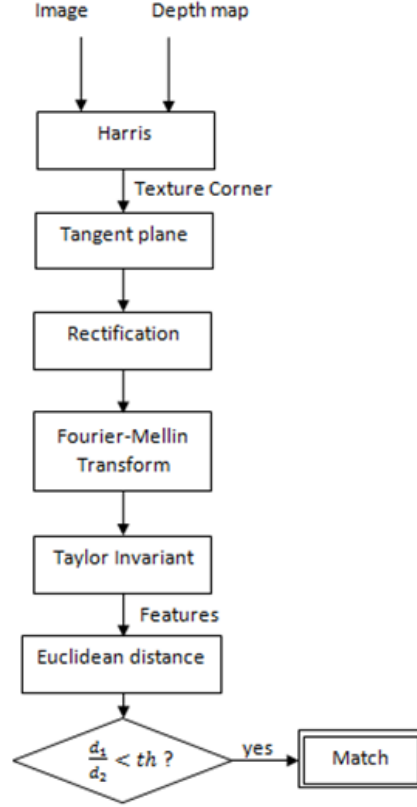


Figure 2.15: Block diagram of the proposed matching algorithm, where depth maps are integrated with RGB images to provide robust stereo-correspondences with wide baseline.

2.5 Wide Baseline Correspondences using Depth Maps

Since our main contributions in the calibration process are the definition and implementation of a novel matching method, where depth maps are integrated with RGB images to provide robust descriptors, we dedicate an entire section to describe the algorithm. The block diagram of the proposed algorithm for finding putative stereo correspondences is represented in Fig. 2.15. Feature points matching between two or more shots of a scene acquired from different viewpoints is one of the basic and most tackled computer vision problems. This is the crucial step to define epipolar geometry between views. In many common applications, like objects tracking in video sequences, the baseline is relatively small and features matching can be easily obtained using well known feature descriptors [34]. However many other applications require feature matching in much more challenging contexts. Many in-

interesting approaches based on two RGB images have been proposed in the literature. Starting from the pioneering work of Schmid and Mohr [104], Matas *et al.* [78] introduced the maximally stable extremal regions (MSER), where affine-invariant stable subsets of extremal regions are used to find corresponding *Distinguished Regions* between images. Van Gool *et al.* [83] introduced moment descriptors for uniform regions. Other approaches are based on clearly distinguishable points (like corners) and affine-invariant descriptors of their neighborhoods. Thanks to its outperforming capabilities, as shown by Mikolajczyk and Schmid [80], one of the most popular algorithms is the Scale Invariant Feature Transform (SIFT), proposed by Lowe [70]. The SIFT algorithm is based on a local histogram of oriented gradient around an interest point and its success is mainly due to a good compromise between accuracy and speed (it has also been integrated in a Virtex II Xilinx Field Programmable Gate Array, FPGA [105]). Actually some other algorithms, always based on affine invariant descriptors, got growing interest like the Gradient Location and Orientation Histogram (GLOH) [80], which is quite close to the SIFT approach, but requires a Principal Component Analysis (PCA) for data compression, or the Speeded-Up Robust Features (SURF) [8], a powerful descriptor derived from an accurate integration and simplification of previous descriptors. All of the aforementioned approaches assume that, even if nothing is known about the underlying geometry of the scene, the defined features, since they are describing a very small portion of the object, will undergo a simple planar transformation that can be approximated with an affine homography. This simplification has two main drawbacks: first of all the extracted features are very general and weak, since wide affine/projective transformations must provide very similar results; moreover, whenever the acquired object presents abrupt geometrical discontinuities (*e.g.* geometrical edges or corners) the affine approximation is not valid anymore. A possible solution to the latter problem can be obtained having a rough description of the underlying 3D geometry. In particular, within the Astute Artemis project, we investigate how depth maps can be used to estimate almost planar regions within the image where the considered interest points are laying.

Many recent approaches integrate range data or depth maps together with geometrical descriptors to improve matching rate in wide baseline or scene changing contexts. In particular, we point out two quite similar and significant approaches that tackle 3D distortion geometrical corrections [60] [124]. Anyway both of them use SIFT as final descriptor after the geometrical corrections. On the contrary, our

aim is to explore, after the rectification of the planar region around the interest point, some less flexible and less generic but more robust descriptors. In particular our descriptors are just *similarity invariant*, bringing with them more information about the analyzed region.

The rough geometrical description of the scene obtained using the depth map, together with Similarity Invariant Descriptor applied on the corresponding visual snapshot, allows to find robust stereo correspondences in images acquired with wide baseline.

2.5.1 Surface vs. Texture Relevant Points

We apply the Harris corner detector [40] on each color image and, having the corresponding depth map, we are able to locate the extracted corners on the corresponding depth map (as shown in Fig. 2.16). Moreover we are able to distinguish between:

- Texture corners: corners due to textural variations but belonging to a flat surface;
- Geometrical corners: corners corresponding to abrupt variations in the surface.

The capability to characterize different Harris features as geometrical or not (*i.e.* if they are also present or not on the depth map) is particularly important for the definition of robust invariant descriptors. To find if a corner is a texture corner or not we apply the Principal Component Analysis (PCA) on the depth map region surrounding each interest point. In particular, according to Jolliffe [54], we evaluate the covariance matrix of a 15×15 neighborhood around the corner (we use a 15×15 neighborhood window centered at the considered point, but it can be adapted accordingly to the surface roughness or curvature) and then we perform the eigenvector decomposition. A neighborhood of N pixels around an interest point on a calibrated depth map can be represented as a 3D cloud of points $\mathbf{p}_j$. The covariance matrix S_p is estimated as:

$$S_p = \frac{1}{N-1} \sum_{j=1}^N (\mathbf{p}_j - \bar{\mathbf{p}})(\mathbf{p}_j - \bar{\mathbf{p}})^T, \quad (2.45)$$

where $\bar{\mathbf{p}}$ is the center of mass of the point cloud. S_p is a 3×3 matrix and applying the PCA:

$$S_p = W \Lambda W^T, \quad (2.46)$$

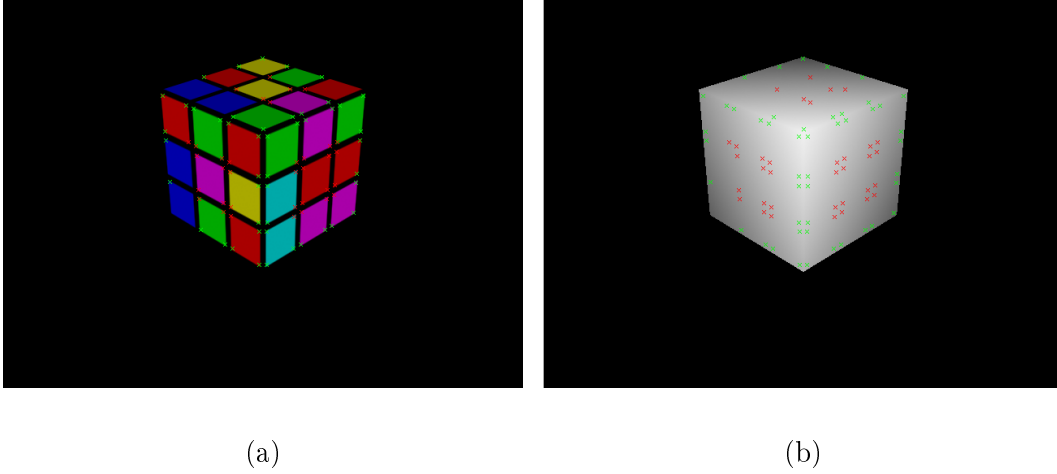


Figure 2.16: Texture (in red) and surface (in green) corners extracted on (a) the color image and classified using (b) the corresponding depth map.

where $\Lambda = \text{diag}\{\lambda_1, \lambda_2, \lambda_3\}$. Since the lower eigenvalue λ_3 indicates the dispersion (variance) orthogonally to the two major directions, we account that value as the local “flatness” index of the point cloud and we accept only values lower than a threshold. In other words we are imposing to consider only texture corners, which neighborhood belongs to a flat surface (represented in red in Fig. 2.16).

2.5.2 Local Plane Extraction

The opportunity to recover univocally the plane where the neighborhood of the interest point lays allows us, applying the proper homography, to obtain a frontal view of the neighborhood of the considered point, independently from the viewpoint. Once the planarity of a region around an interest point is verified, the parameters of the plane passing through that region can be estimated. Many techniques have been developed to find flat surfaces in depth maps [129]. Also surface curvature from points cloud has been deeply investigated [130]. To find the tangent plane, we consider the eigenvector $\mathbf{w} = [w_x, w_y, w_z]^T$ associated to the lower eigenvalue. It represents the tangent plane direction cosines. Accordingly to the equation for a generic point of plane $\mathbf{p}$:

$$\rho = \mathbf{p} \cdot \mathbf{w} = p_x w_x + p_y w_y + p_z w_z, \quad (2.47)$$

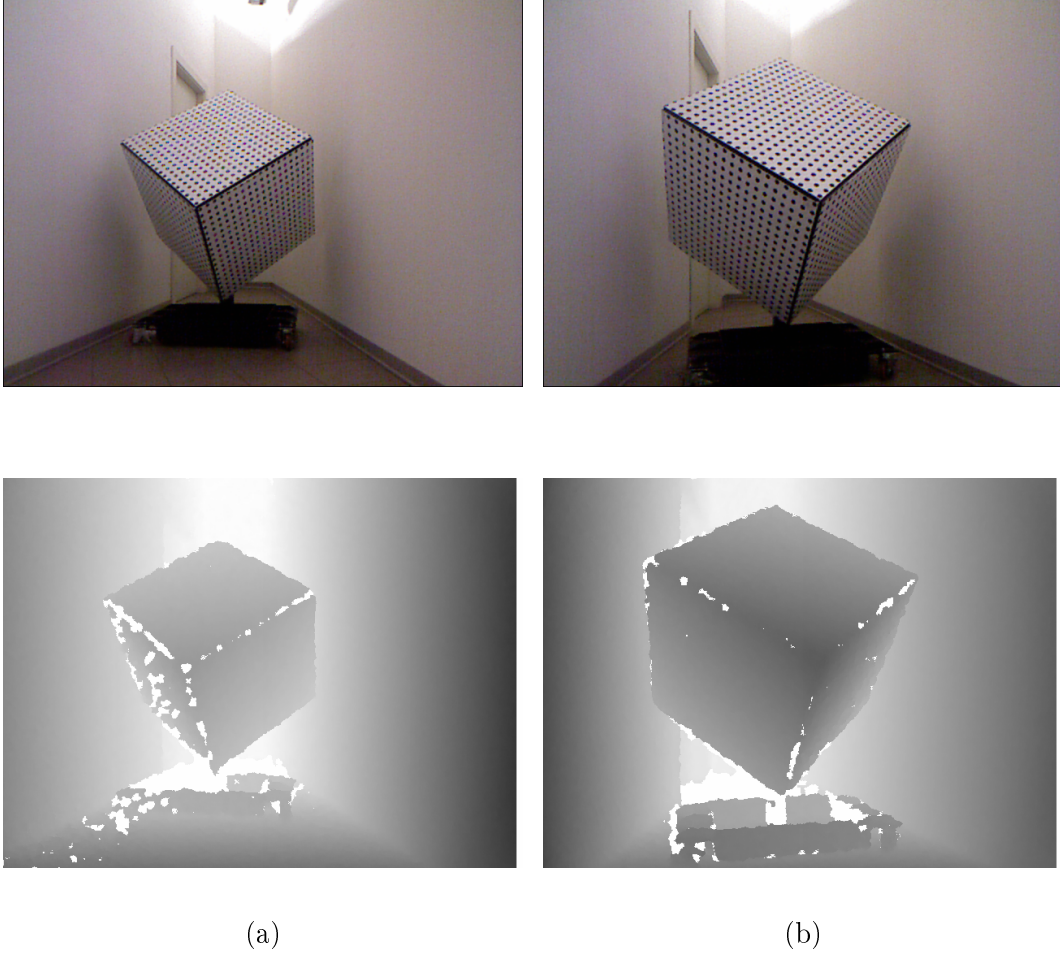


Figure 2.17: Two color images (top row) and their corresponding depth maps (bottom row) acquired with the Kinect device.

where the parameter ρ represents the distance between the plane and the camera reference system. This distance can be recovered with metrical exactness only if the depth map is intrinsically calibrated (pixel size and focal length are known).

To improve the estimation of the plane parameters and to group interest points belonging to the same plane, we adopt a uniform quantization for planes parameters with different orientations and distances from the origin in a spherical coordinate system. In particular, given a point $\mathbf{p}$ on the plane with Eq. (2.47) and defining θ as the azimuth angle and φ as the elevation angle, we can rewrite Eq. (2.47) as:

$$p_x \cdot \cos\theta \cdot \sin\varphi + p_y \cdot \sin\theta \cdot \sin\varphi + p_z \cdot \cos\varphi = \rho. \quad (2.48)$$

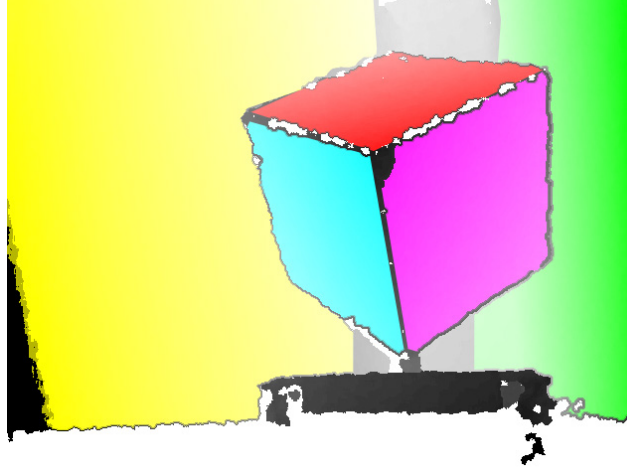


Figure 2.18: Clustering of different planes on a depth map.

We store the number of interest points belonging to a particular plane, defined by its three parameters (ρ, θ, φ) , in the corresponding cell of the 3D Radon Transform $C(\rho, \theta, \varphi)$ [27]. The regions of cells in the Radon Transform (RT) with the main number of elements represent the planes present in the acquired scene. For the same quantization of the three parameters ρ, θ, φ , we have experimentally verified that 3D Hough [15] and 3D Radon Transforms obtain very similar results for the 3D discrete function C .

To calibrate pairs of Kinect devices, we use a half-cube (3 contiguous faces) with sides length of 0.8 m. Fig. 2.17 shows two different shots of the calibration pattern with their respective depth maps. Since we work with 3 couple of Kinect devices, the only constraint is that, for each useful acquisition, at least a couple of devices sees at least one common plane of the half-cube. Using the RT, different plane surfaces from the depth map can be localized in space (as illustrated in Fig. 2.18).

2.5.3 Plane Rectification

Once we find planes in the depth map, we can easily define the homography to rotate each plane in order to obtain a frontal view [42]: the vector normal to the plane has to align to the optical axis and the image projected on the depth map is then transformed into a frontal-view image. A schematic representation of the plane rotation procedure is presented in Fig. 2.19.

Not only the vector $\mathbf{w}$ orthogonal to the plane is necessary, but also the pivot

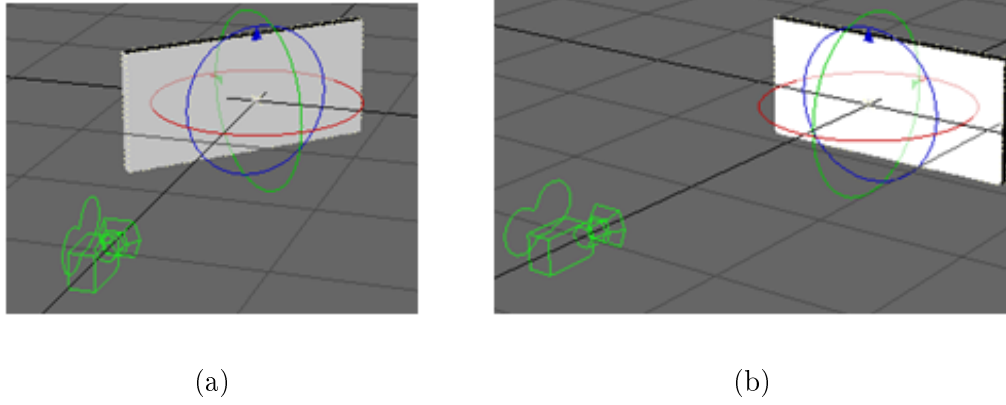


Figure 2.19: Representation of the estimated plane (a) before and (b) after the rotation procedure in order to align it with the camera image plane.

point for the plane rotation. It is computed as the center of mass of the acquired points belonging to the estimated plane. A 3D rotation, once the rotation axis and angle are estimated, can be computed following these steps:

- the plane is translated so that the central point corresponds to the origin of the system reference;
- the plane is rotated so that the orthogonal axis corresponds to the z axis of the reference system. In particular, considering a sinistrorse tern, first a rotation around the y axis is performed so that $\mathbf{w}$ belongs to the plane $x = 0$, second a rotation around the x axis is performed till $\mathbf{w}'$ corresponds to the z axis.

Considering the vector $\mathbf{w}$ orthogonal to the plane, the rotation angle around the y axis is:

$$\theta = \arctan\left(\frac{w_2}{w_1}\right). \quad (2.49)$$

When the new orthogonal vector $\mathbf{w}' = [w'_1, w'_2, w'_3]$ is computed, the rotation angle around the x axis is:

$$\varphi = \arctan\left(\frac{w'_1}{w'_3}\right). \quad (2.50)$$

Calling the pivot point $\mathbf{p} = (x, y, z)$, the transformation is summarized by the ho-

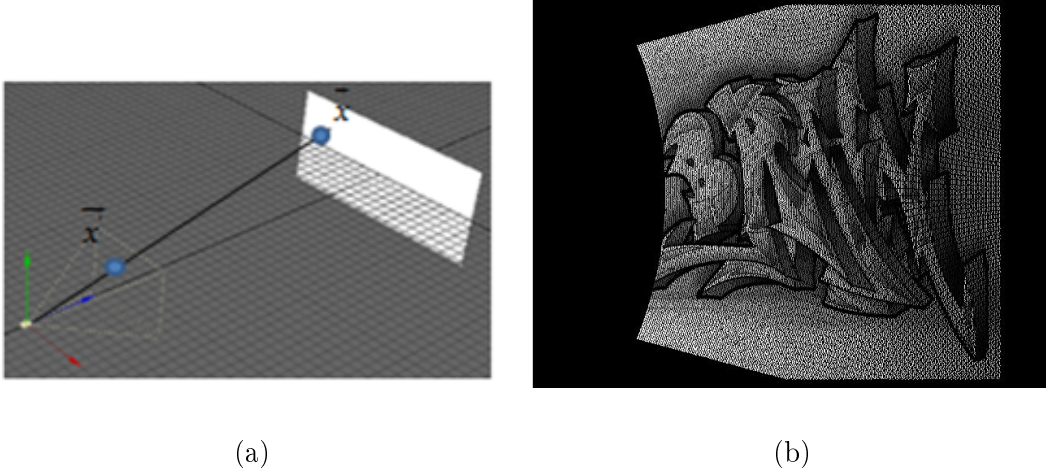


Figure 2.20: (a) Representation of the homography between the rotated 3D plane and the virtual image plane. (b) Frontal view image obtained after the projection of the rotated plane on the camera image plane.

mography:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \rho \\ 0 & 0 & 0 & 1 \end{bmatrix} R_x(\varphi) R_y(\theta) \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2.51)$$

where $R_x(\varphi)$ is the rotation matrix representing a rotation of an angle φ around the x axis, $R_y(\theta)$ is the rotation matrix representing a rotation of an angle θ around the y axis, and ρ is a scaling factor that can be used in order to place the rotated plane at a fixed distance ρ from the original plane.

The homography between the rotated 3D plane and the virtual image plane is computed to project the points from the rotated plane into the camera image plane (the camera parameters are known) as shown in Fig. 2.20(a). The frontal view image is therefore obtained (Fig. 2.20(b)). The frontal view is not yet useful because it is not regularly sampled. We apply bi-cubic interpolation on the rectified images in order to minimize artifacts due to the quantization. Thanks to this process we are able to obtain the so called *rectified images*, as shown in Fig. 2.21(c) for a set of synthetic examples.

The direct effect of this transformation is that the misalignment between two images of the same plane, acquired under different viewpoints, after the rectification can be modeled by a 2D similarity transformation, completely defined by translation,

2 Acquisition System

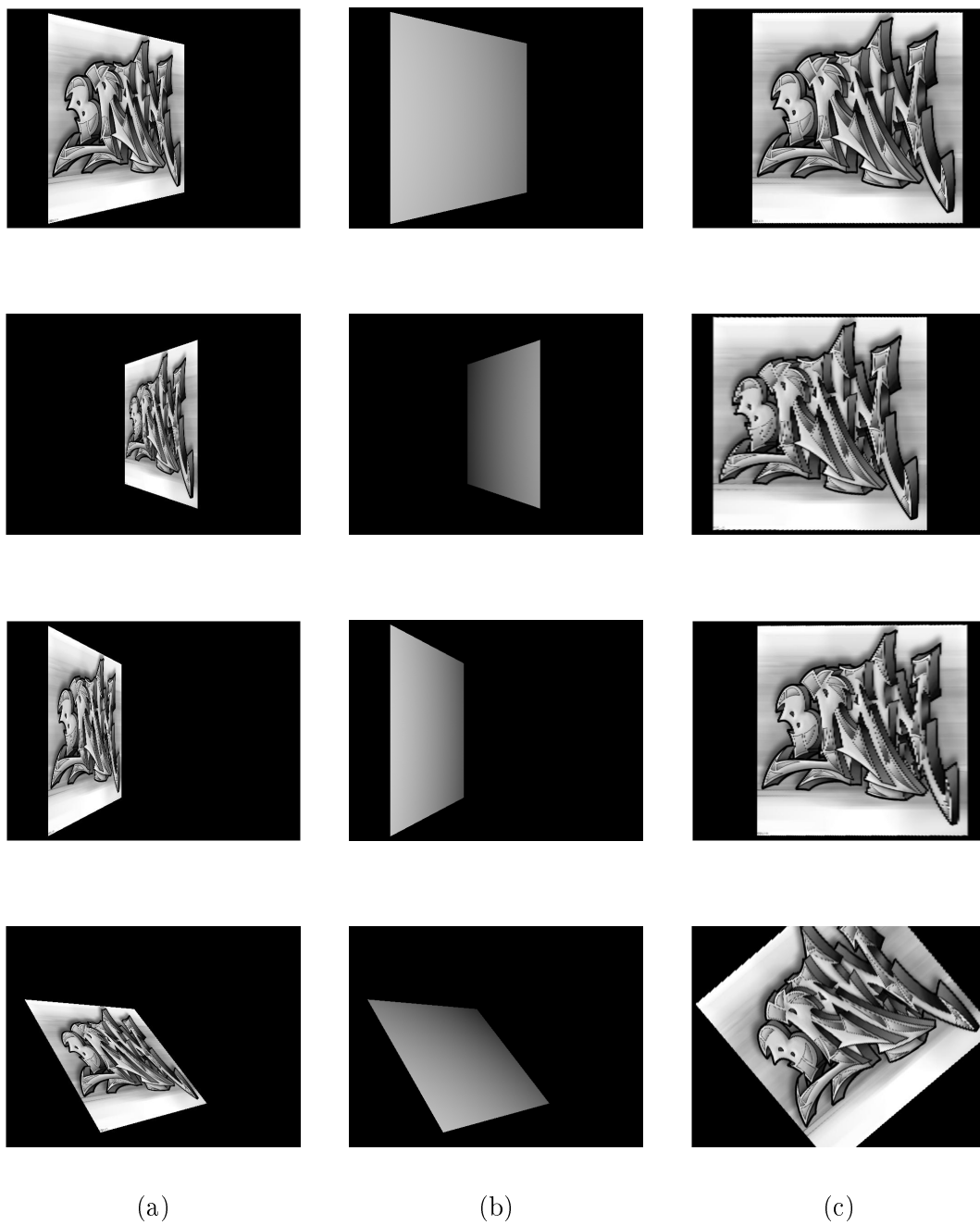


Figure 2.21: Examples of (a) synthetic images obtained by a ray-tracing software and (b) the relative depth maps. (c) Images obtained after the rectification process.

rotation and scaling (Fig. 2.21(c)). Considering only rectangular neighborhood (15×15) centered around an interest point, we are automatically avoiding the translation transform between regions of interest and, as a consequence, the feature descriptor is no longer required to be invariant to translation, but only to rotation and scaling. Furthermore if the acquisition device is also metrically calibrated (it provides us the metric distance of each point from the camera), we are able to compare all the features as if they are placed at the same distance from the camera and the only remaining degree of freedom is due to the rotation.

2.5.4 Similarity Invariant Transform

Accordingly to the aforementioned steps, we are able to obtain a 2D representation of the same 3D object part, whose misalignment can be modeled by a four-parameter geometric transformation that maps each point (x_f, y_f) in F to a corresponding point (x_g, y_g) in G according to the matrix equation (in homogeneous coordinates):

$$\begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix} = \begin{bmatrix} s \cos \vartheta & s \sin \vartheta & -\Delta x \\ -s \sin \vartheta & s \cos \vartheta & -\Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix}. \quad (2.52)$$

Equivalently, defining the two images as two functions denoted by f and g , representing a gray-level image defined over a compact set of R^2 , for any pixel (x, y) is true that:

$$f(x, y) = g(s(x \cos \vartheta + y \sin \vartheta) - \Delta x, s(-x \sin \vartheta + y \cos \vartheta) - \Delta y), \quad (2.53)$$

where Δx and Δy are translations, s is the uniform scale factor, and ϑ is the rotation angle. In other words, when we speak about similarity transformation, we refer to the operations in this order:

$$RST = RS_{s, \vartheta} \cdot T_{\Delta x, \Delta y}. \quad (2.54)$$

Since we are comparing image regions centered around interest points, the translation invariance has no relevance in our case and the similarity invariance can be limited to rotation and scaling.

Many approaches are present in the literature to tackle this problem [87], anyway most of them, like geometric moments and complex moments, discards more infor-

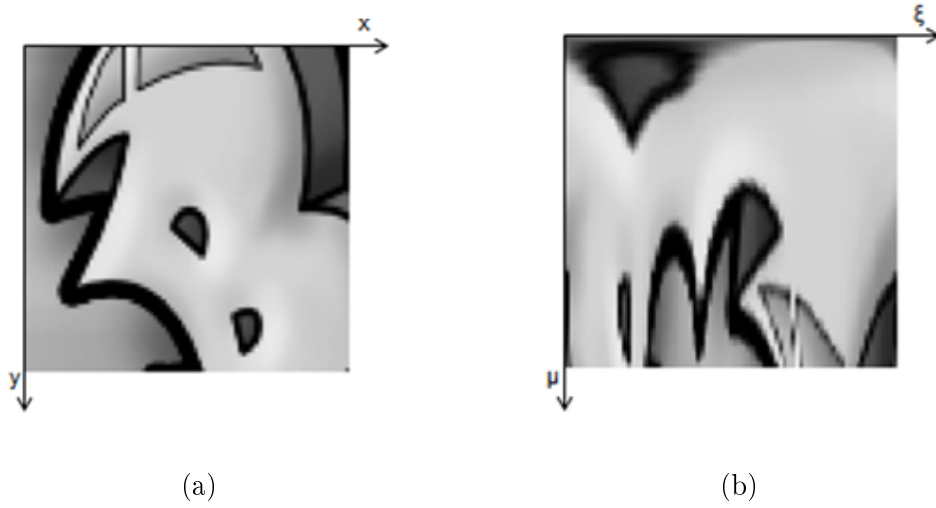


Figure 2.22: Example of neighborhood of the interest point (a) in cartesian coordinates and (b) in log-polar coordinates.

mation that is strictly necessary in order to obtain a descriptor invariant to rotation, scaling, and translation. As a result, the extracted features are more general than required. We orient our research towards complete descriptors, that means that only representations retaining all the information of an image, except for orientation and scale, are considered.

In particular, we use the Fourier-Mellin transform (FMT) [20] that is the Fourier Transform of the image $f(x, y)$ mapped in its corresponding Log-Polar coordinates $f_{LP}(\mu, \xi)$ (Fig. 2.22):

$$f_{LP}(\mu, \xi) = \begin{cases} f(e^\mu \cos \xi, e^\mu \sin \xi) & \xi \in [0, 2\pi) \\ 0 & otherwise \end{cases}, \quad (2.55)$$

where μ is the logarithm of the distance to the origin and ξ is the angle between a line of reference (the x-axis) and the line through the origin and the point (x, y) :

$$\begin{cases} \mu &= \log(\sqrt{x^2 + y^2}) \\ \xi &= \arctg\left(\frac{y}{x}\right) \end{cases}. \quad (2.56)$$

The FMT is defined as:

$$F_m(w, k) = \int_0^\infty \int_0^{2\pi} f_{LP}(\mu, \xi) e^{-j(w\mu + k\xi)} d\xi d\mu. \quad (2.57)$$

Then we explore two possible invariants for orientation and scale: the Taylor Invariant and the Hessian Invariant. In particular, we recall that, after a Log-polar transformation, a rotation corresponds to a circular shift along the axis representing the angles, while a scaling corresponds to a shift along the logarithmic radial axis. Applying the 2D Fourier transform to the Log-polar transform, the aforementioned shifts are reflected in a linear phase contribution, while the amplitude remains unchanged.

Taylor Invariant Descriptor

The Taylor invariant descriptor [16] is focused on eliminating the linear phase contribution that is present in the image spectrum by subtracting the linear phase part from the spectrum. Let $F(u, v)$ be the Fourier transform of an image $f(x, y)$, and $\phi(u, v)$ be the phase of the Fourier transform. The following complex function is called the Taylor invariant:

$$F_{TI}(u, v) = e^{-j(au + bv)} F(u, v), \quad (2.58)$$

where a and b are respectively the derivatives with respect to u and v of $\phi(u, v)$ at the origin $(0, 0)$, i.e.:

$$\begin{aligned} a &= \varphi_u(0, 0) \\ b &= \varphi_v(0, 0) \end{aligned} \quad (2.59)$$

The effect is then the registration of the input features in such a way that the phase spectrum is flat in the origin, *i.e.* if we should take the inverse transforms, all of them will be translated to accomplish to this constraint.

Hessian Invariant Descriptor

The idea behind the Hessian Invariant Descriptor [16] is to differentiate the phase spectrum twice to eliminate the linear phase terms. The invariant parts are then the modulus of the spectrum and the three, second order, partial derivatives of the

phase spectrum:

$$F_H(u, v) = [|F(u, v)|, \varphi_{uu}(u, v), \varphi_{uv}(u, v), \varphi_{vv}(u, v)]. \quad (2.60)$$

Considering the image $f(x, y)$ expressed in log-polar coordinates $f(\mu, \xi)$, the two aforementioned descriptors are invariant to rotation and scaling. We would like to point out that a descriptor invariant to rotation and scaling is achieved also considering only the modulus of the spectrum. Preserving the phase relationships allows to keep more information and to build a more detailed and robust descriptor.

We evaluate both descriptors obtaining very similar results in terms of correct match rate. Anyway the Taylor descriptor has half elements respect to the Hessian descriptor, a fact that must be taken into account when the computational load is considered.

2.5.5 Proposed Algorithm for Stereo-matching

The proposed algorithm can be summarized as follow:

- for each shot of the scene, interest points are extracted using Harris corner detector applied on the picture;
- the PCA is applied on the neighborhood 15×15 of the corresponding point of each detected point on the depth map and it is used to check the planarity of the region around each interest point. Only texture corners are kept.
- the 3D Radon Transform is applied to all the texture corners and the planes present in the acquired scene are estimated.
- the PCA is applied to depth map points belonging to each plane and the homography that transforms that plane into a frontal plane is estimated and applied, followed by a bicubic interpolation in order to avoid artifacts (rectification process);
- the 2D Fourier-Mellin Transform is applied on the reprojected region around each interest point:
 - if the depth map is metrically calibrated, the previous homography places the region around our interest point at a constant distance from the image

plane. In this case the 2D Fourier Transform is applied to the reprojected region expressed in polar coordinates;

- else the scale of the final region is not known and the 2D Fourier-Mellin Transform is applied to the reprojected region (2D Fourier Transform applied on the reprojected region expressed in log-polar coordinates);
- the Taylor invariant is applied on $F_m(w, k)$;
- the resulted vector (SID) is used as feature descriptor of the interest point and correct matches from different images are selected as those for which the Euclidean distance is minimum (only if the ratio between the first and the second putative match is less than a threshold).

The resulted set of putative correspondences is used to calibrate pairs of Kinect devices as described in Section 2.3.

2.5.6 Results and Discussion

We perform some experiments and demonstrate the accuracy of the extracted matches [75], [74]. We check the discriminative power of the proposed SIDs with respect to state of the art SIFT by applying them on the rectified versions of the original images acquired with different viewpoints. Obviously SIFT is built to find matches in the more general case of affine/perspective transforms, but our aim is to demonstrate that, when depth maps are available together with their texture images, more detailed descriptors, invariant only to rotation and eventually scaling, can be adopted providing us better matching score, even when a wide baseline is present.

We perform some experiments using snapshots similar to those visible in Fig. 2.23. We also analyze results on synthetic images, in order to insulate acquisition noise contributions and errors due to a wrong estimation of the homography (see Fig. 2.21 for some examples).

In Fig. 2.25 it is possible to see correspondences found by the Taylor Invariant Descriptor (TID) and the SIFT applied on rectified images. The results proposed are obtained using the Taylor Invariant Descriptor, but they are similar to those obtained using the Hessian Invariant Descriptor (HID).

As far as we know, no database of pictures and associated depth maps with a wide baseline is yet available, so we decide to test our algorithm taking 10 pictures of the box illustrated in Fig. 2.23 acquired from different viewpoints. We use a Kinect

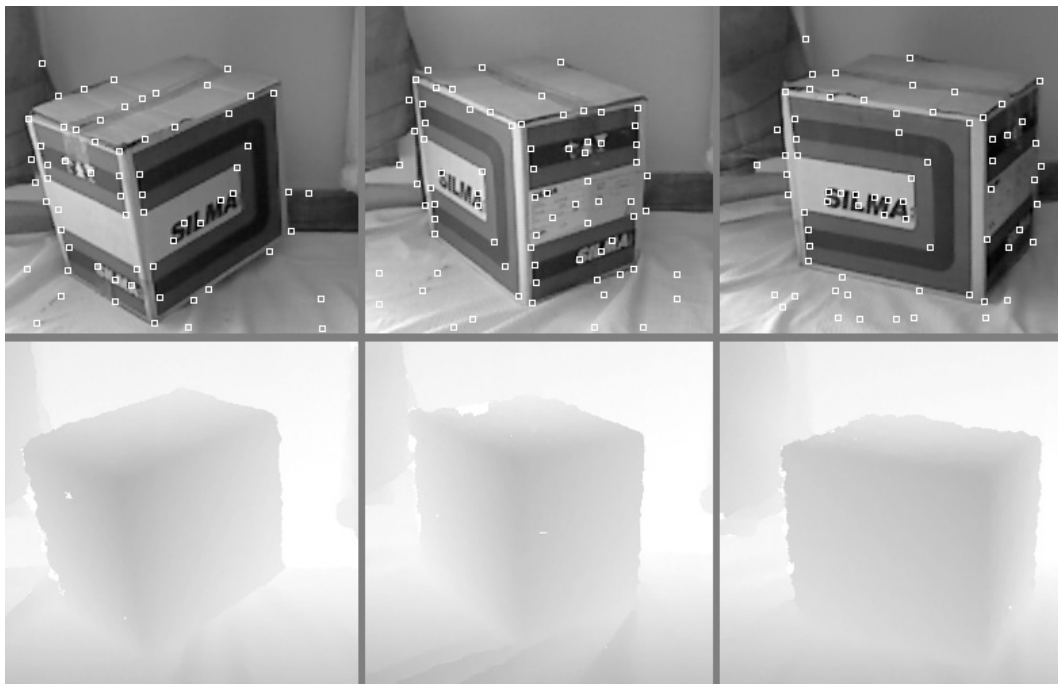


Figure 2.23: Gray scale images of a box acquired from different viewpoints (top row) and corresponding depth maps (bottom row). Texture corners are indicated by empty squares.



Figure 2.24: Images of the rectified version of the same plane estimated from the depth map.

device for the acquisitions in an indoor environment and without any restriction except avoiding the sun light directly on the IR device's camera. In Fig. 2.24 we show how the planes, where the interest points lay, are reprojected in frontal views. The homographies have been defined accordingly to the PCA analysis on the underlying depth map. Matching results are reported in Table 2.1.

The resulting approach demonstrates the profitable integration of depth maps with acquired images to strengthen matching capabilities and that the proposed

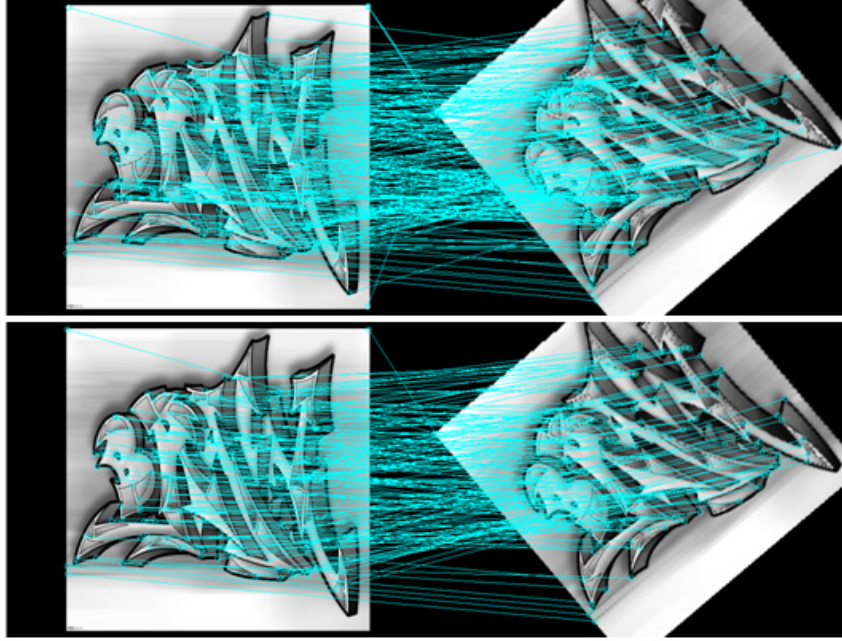


Figure 2.25: Point correspondences found by the SID (top) and the SIFT (bottom) descriptors applied after the rectification process performed on the global image.

Table 2.1: Matching results with different descriptors.

Matching results			
		Synthetic Depth-map	Real box
SIFT	correct Match (%)	81%	75%
	1 st over 2 nd best match	0.79	0.90
Fourier-polar 2D (calibrated) with Taylor Invariant	correct Match (%)	88%	83%
	1 st over 2 nd best match	0.61	0.79
Fourier-Mellin 2D with Taylor Invariant	correct Match (%)	84%	80%
	1 st over 2 nd best match	0.69	0.78
Fourier-polar 2D (calibrated) with Hessian Invariant	correct Match (%)	88%	85%
	1 st over 2 nd best match	0.65	0.73
Fourier-Mellin 2D with Hessian Invariant	correct Match (%)	85%	80%
	1 st over 2 nd best match	0.69	0.79

descriptors hold much more information of the original image patch respect to more general descriptors. Furthermore, the computational complexity of the proposed descriptors is also very close to the state of the art feature descriptors.

3 Volumetric Human Body Data

In this Chapter we introduce the steps for the realization of the database of actions. Once the system developed in the ISPG laboratory is completely calibrated, three synchronized sequences of RGB images and corresponding depth maps are available. We extract the human silhouette from each image and we use it to segment the corresponding depth map and to reproject depth-pixels into the scene. The 3D information coming from different depth maps taken from different viewpoints are then merged, in order to obtain a complete description of the 3D actor body represented as a voxel set.

We extract the human silhouettes on the color images in order to obtain a segmentation actor-scene finer than the one achievable by imposing a threshold on the corresponding depth map. Two connected pixels, one belonging to the subject and the other one to the background, have very different depth values. A simple segmentation could be performed setting that if two connected pixels show a difference in depth levels greater than a predefined threshold, one will belong to the foreground and the other will belong to the background. Two are the main drawbacks of this kind of approach: (i) depth maps are usually very noisy and a segmentation based on their boundaries will easily introduce reconstruction errors and artifacts; (ii) this simple threshold based approach is not able to distinguish between the actor and the floor, and the volumetric rendering will suffer from this miss-segmentation. Color images are less noisy and the scenario developed in the ISPG laboratory allows to obtain an extremely accurate segmentation of the actor silhouette.

The silhouette mask extracted on the RGB image is replicated on the corresponding depth map. For the 3D reconstruction, the *Visual Hull* approach is used [62]. The Visual Hull is a finite region of the space that contains the object, it is a sort of “pod”. The main advantage using depth maps is that the information achievable allows to reconstruct a sampled version of the subject surface that is in front of the device, also in its concave parts. If only the silhouettes extracted from the RGB

3 Volumetric Human Body Data

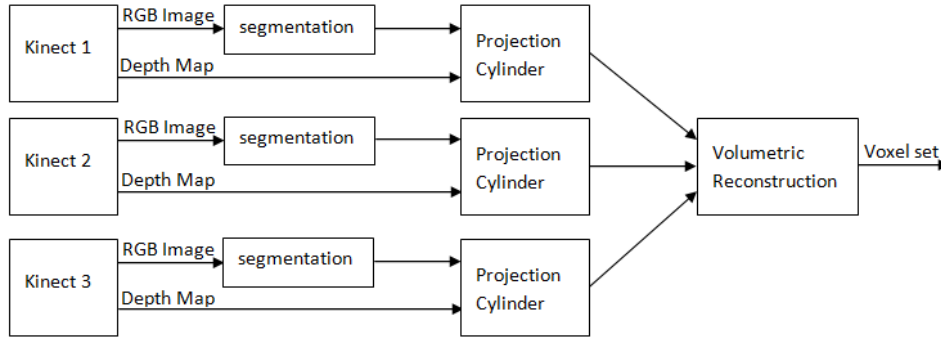


Figure 3.1: Block diagram of the acquisition process from the synchronized video sequences of both color images and depth maps, till the 3D reconstruction of the actor pose in each frame.

images are used, the volumetric description of the body will be obtained intersecting the solid regions of space in which the object is constrained to lie by each silhouette. These regions are cones (occlusion cones) obtained by back projecting from a viewpoint (each camera optical center) the corresponding silhouette for perspective projections. As the number of cones increases, the object is reconstructed with higher precision, but only the Convex Hull is achievable. Using the information coming from depth maps, the volumetric reconstruction is obtained intersecting no more cones, but pseudo-cylinders having the sampled subject surface facing the device as one base (see Fig. 3.4 as example). These concave surfaces represent a great advantage in order to obtain a volumetric reconstruction that is closer to the real actor body.

Therefore, using Kinect devices allows to combine two advantages: the RGB images allow an accurate segmentation; and the depth maps allow to reconstruct the human body also in its concave parts (using a Visual Hull approach). Moreover, obtaining a 3D body reconstruction before analyzing the object surface or classifying the actions has two main advantages:

- Both viewpoint problems and body postures tracking are clearly separated.
- Volumetric reconstruction greatly reduces problems with viewpoint occlusions, simplifying the object description for the recognition task.

This Chapter describes the acquisition process of the calibrated net of 3 Kinect devices, from the synchronized video sequences of both color images and depth maps, till the 3D reconstruction of the actor pose in each frame. The output is, for each acquired sequence, a sequence of 3D reconstructed voxel sets representing the space

occupancy of the actor body at each acquired time. This process, called *Voxel representation* in Fig. 1.1, involves different steps, shown in Fig. 3.1 and listed below:

- Video acquisition: a Kinect net composed by three calibrated devices acquires the scene in which the actor is moving.
- Background subtraction: each RGB video acquired is segmented and the actor is isolated respect to the background. The result is a binary mask (silhouette) for each frame of each camera.
- Volumetric reconstruction: the extracted silhouettes are used in order to segment the corresponding depth maps and to compute the volumetric reconstruction merging different views.

We now analyze each element that composes the acquisition system.

This Chapter is organized as follows. Section 3.1 describes the color image segmentation process in order to extract the actor silhouette from each acquired frame. Section 3.2 introduces the procedure used to reproject the segmented depth map in front of the device (IR-camera) and Section 3.3 explains the merging of the voxel reconstructions using the extrinsic camera calibration matrices. In Section 3.4, the database of gestures acquired is described more in details.

3.1 Image Segmentation

Image segmentation is the first process that has to be performed on the acquired sequences. This type of technique is used in order to distinguish the actor silhouette from the rest of the image (for each frame and for each color image acquired). The controlled environment and the presence of a single actor in the scene allow to extract the silhouette with an easy segmentation algorithm. The goal is to obtain a precise actor silhouette, minimizing the possibility of false positives (pixels assigned to the silhouette but do not belonging to the actor, mostly due to shadows projected on the floor) and false negatives (hole in the silhouette due to reflexes or to clothes with color similar to the background).

To perform segmentation is used the same algorithm developed in the ISPG laboratory for the multi-camera system [95]. It is a variation of the *Running Gaussian Average* algorithm [123] applied to the color space *Hue-Saturation-Value* (HVS) in

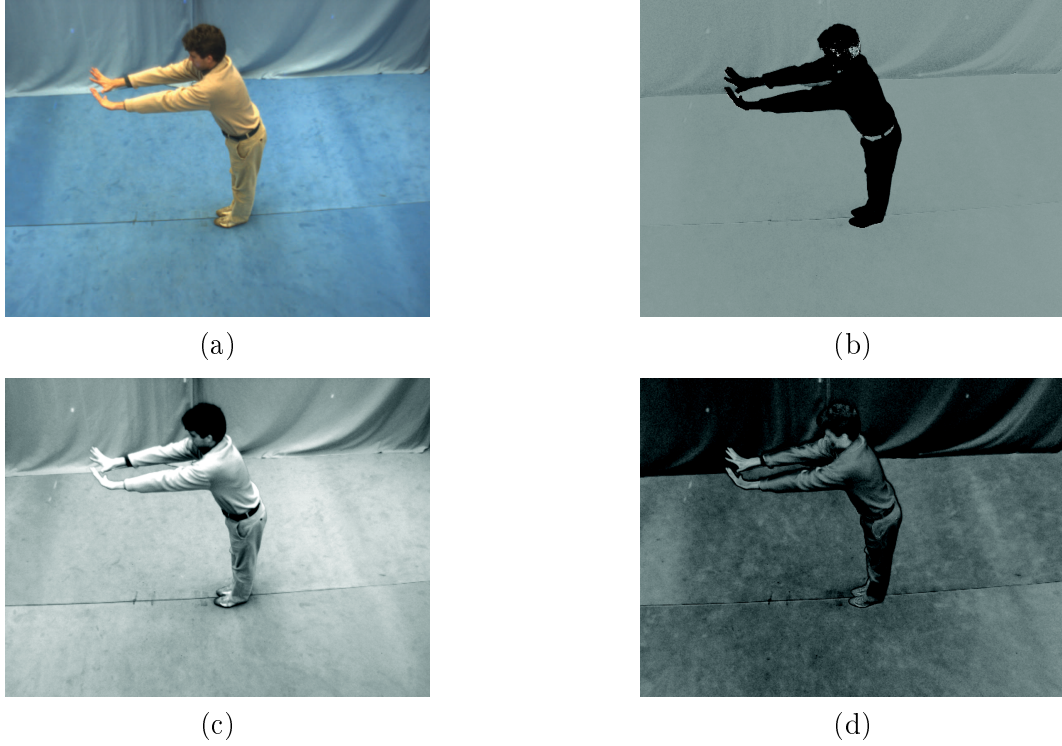


Figure 3.2: Example of (a) acquired image and its visualization in the (b) Hue, (c) Value, (d) Saturation channels. This image is acquired with the set of 8 cameras originally used in the Smart Space [95] and on which the proposed segmentation algorithm was tested at first.

order to take into account the blue color of the room. For each viewpoint, the background model is composed by a set of Gaussian probability density functions, one for each pixel, learned with a sequence of 20 frames acquired with the empty room (only the Hue channel is considered). In order to reduce the memory occupancy, sampled mean and standard deviation of the Hue level of each pixel are computed incrementally, requiring the storage of only two matrices having the same size of the acquired frame:

$$\mu_n = \mu_{n-1} + \frac{I_n - \mu_{n-1}}{n}, \quad (3.1)$$

$$\sigma_n^2 = \frac{(n-2)\sigma_{n-1}^2 + (I_n - \mu_{n-1})(I_n - \mu_n)}{n-1}, \quad (3.2)$$

where μ_n and σ_n^2 respectively indicate the mean value and the variance updated

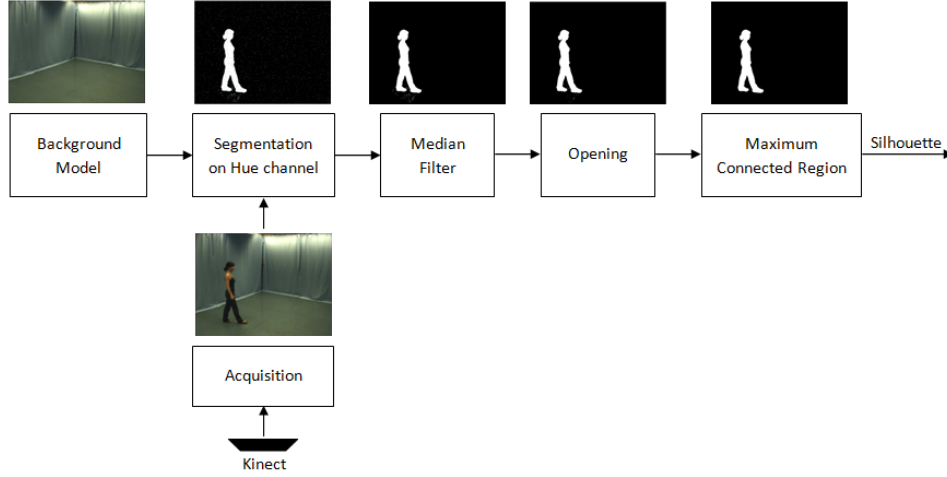


Figure 3.3: Block diagram of the segmentation process. The background model is used for an estimation of the human silhouette in the Hue channel. A median filter, a morphological opening and the selection of the maximum connected region are then used to refine the extracted silhouette.

till the frame n ; n indicates the number of considered frames; and I_n indicates the Hue intensity value of the considered pixel in frame n . Only the Hue channel is historically used because the blue color of the room makes this channel the most informative for the segmentation (as demonstrated with an example in Fig. 3.2).

Using a controlled and stationary environment, the estimated background model is kept fixed for the entire sequence. The segmentation is made by implementing a binary test: a pixel is classified as belonging to the foreground if the difference between its Hue value and the estimated mean is greater than the estimated variance multiplied by a scale factor of 4.0 (this value has been heuristically found from our sequences to minimize the aforementioned problems):

$$|I_n - \mu_n| > 4\sigma_n. \quad (3.3)$$

This Gaussian model is able to produce a first good segmentation solving problems due to shadows, but some noise and some spurious regions still remain in the result. Three post refinements are applied:

- a *median filter* of 5×5 pixels size is applied in order to remove “salt and pepper” noise (preserving at the same time the mask boundaries);
- a morphological operator of *opening* is applied in order to fill remaining holes in the extracted silhouette;

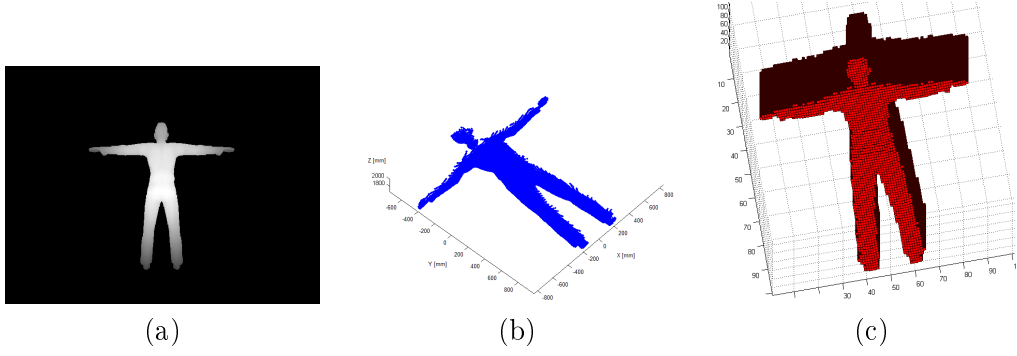


Figure 3.4: (a) Segmented depth map. (b) 3D surface points reconstruction. (c) Volumetric rendering.

- the *maximum connected region* is selected, using morphological operators, and identified as the actor silhouette.

Fig. 3.3 represents the block diagram of the proposed algorithm together with an example of the extracted mask after each step.

3.2 Projection Cylinders

The Visual Hull, concept for silhouette-based image understanding, was first introduced by Laurentini in the 1994 [62]. Considering a generic 3D object S and the set of the acquired views R , for each view r , the corresponding silhouette s_r exists. s_r distinguishes, on the image plane, between points belonging or not to the object. For each view the 3D visual cone vh_r is then computed. vh_r is the set of optical rays that projects each point belonging to s_r from the optical center to the 3D space. The 3D space intersection of all the possible cones, $vh_R = \cap_{r \in R} vh_r$, is an approximation of the object S . The Visual Hull is the intersection when the number of views goes to infinity vh_∞ . The Visual Hull is not exactly the object S because the algorithm is not able to reconstruct the concave parts of the objects. Other techniques can be used to solve this issue [76], [95], [121]. Using Kinect devices, we can apply the Visual Hull algorithm on the segmented depth maps, obtaining a 3D reconstruction of the object also in its concave parts, without implementing other processing.

Once the actor silhouette is extracted from each of the available frames, this mask is converted on the IR camera image plane. This produces a better depth map segmentation than the one achievable putting a threshold on the depth map pixels. Moreover, the interference between two or more Kinect must be taken into account.

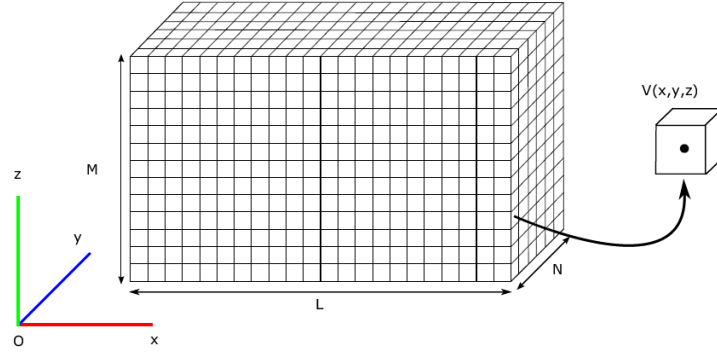


Figure 3.5: Voxel sampling of the scene.

The interference, due to the IR projected pattern, occurs as sporadic holes (group of black pixels) on the depth maps. We are studying different algorithms for denoising depth maps [82], but, in this particular implementation, a gaussian median filter is applied on the masked depth maps in order to avoid both holes and spikes on the estimated surface.

Having the segmented depth map (Fig. 3.4(a)) and all the IR-camera parameters, we can reconstruct a sampled version of the object surface standing in front of the device (Fig. 3.4(b)). Each pixel belonging to the silhouette on the depth map is then projected in the 3D space, from the distance corresponding to its gray level till the maximum extension of the considered space, on a line orthogonal to the IR camera image plane (using Kinect we are working with elevation depth maps where the gray tone corresponds to the distance from the image plane [102]). The obtained volume is called *projection cylinder* (Fig. 3.4(c)) and it is obtained for each device at each acquired frame.

The volumetric rendering is achieved working with *voxels* (volume elements), *i.e.*, discretizing the three-dimensional space (in this case each voxel is a box of 2 cm side as shown in Fig. 3.5). We impose 4 m as the maximum distance from the depth camera reference plane. The projection cylinder is built for all the available views by assuming a regular grid for each of them (canonical space for each view).

3.3 Volumetric Reconstruction

Once the projection cylinders are computed for each of the 3 different views, they are combined in order to reconstruct the 3D volume occupied by the human body. Since the 3 Kinect net is calibrated, the position of each device respect to the others is known. The extrinsic calibration parameters, once the voxel space centered on each device is obtained, allow us to apply the roto-translation matrix that brings each view to a common system reference. After this process, it is possible to merge the different reconstructions by mean of an easy *AND* logical operator between voxels. Using the *AND* operator may generate holes in the reconstructed voxel set. Experimentally, we often avoid this kind of errors by applying a morphological *closure* (see Chapter 4 for its definition) on the volumetric reconstruction obtained from each device before compute the intersection. Fig. 3.6 summarizes the process starting from the masked depth maps:

- each masked depth map is reprojected in the canonical space of the corresponding IR camera, obtaining the projection cylinder for each view;
- a morphological closing is performed in order to avoid holes in the reconstructed volume;
- each projection cylinder is roto-translated in the Kinect 1 reference system;
- the 3 transformed projection cylinders are merged with an *AND* logical operator, obtaining the 3D reconstruction of the subject volume into the scene.

The choice of first reconstructing the scene seen by each device and then merging all the achieved volumes is due to constrain the system neither to the number of devices, nor to their relative positions. As much as the number of devices increases, the object is reconstructed with higher precision. Other approaches could be used: for example the calibrated alignment of the three sampled surfaces. We prefer the Visual Hull approach in order to directly obtain a volumetric reconstruction of the human body and not a sampled version of its surface. Moreover, if parts of the actor are not seen by any device, the proposed algorithm will produce an inaccuracy in the reconstructed volume: that volume still remains a complete solid body. Aligning the depth maps, instead, a miss-acquired portion becomes a hole in the estimated surface and this is a great problem for both the action classification and the surface representation applications. An artifact in the reconstructed voxel set reduces the performances

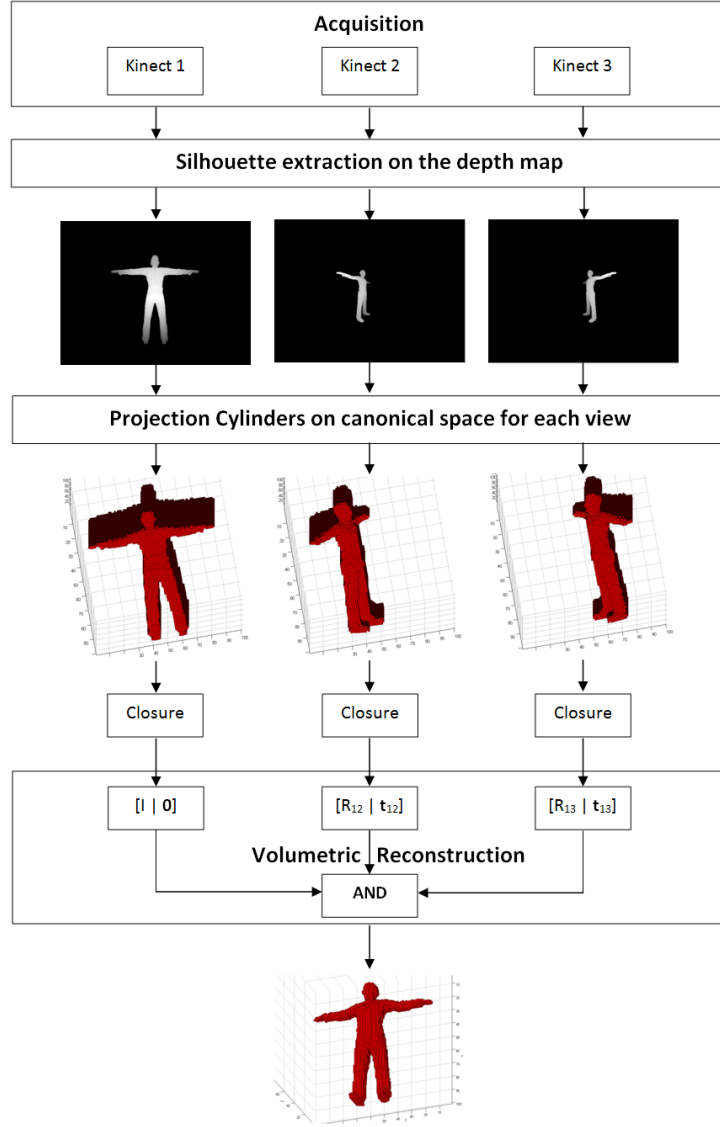


Figure 3.6: Block diagram of the 3D volume reconstruction process. The masked depth map pixels are projected in the canonical space of each device. After a roto-translation process, the projection cylinders are then merged obtaining the 3D reconstruction of the subject volume into the scene.

but does not limit the applicability of the proposed algorithms. Moreover the action classification system (proposed in Chapter 7) is robust to local reconstruction errors. The resulting voxel sets always compactly represent all the available information about the actor body.

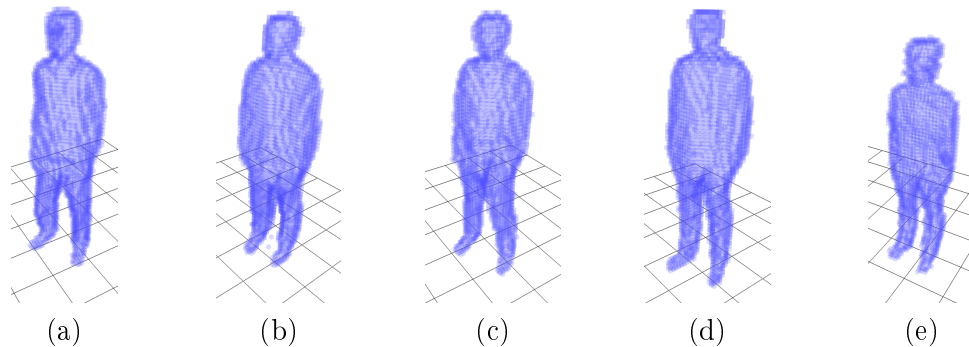


Figure 3.7: Actors in the default pose. Voxel sets at the beginning of the sequence **Open-002** for the actors: (a) Fede, (b) Gabbo, (c) Massi, (d) Teo and (e) Vale.

From the picture at the bottom of Fig. 3.6, it can be noted that, with these *ad-hoc* three views, we obtain a voxel structure that well represents the human body for that particular pose. This model is well suited for subsequent processing to extract one or more descriptors of the object structure. The so called *volumetric reconstruction* of the scene is computed before any other analysis procedure. As the number of views increases, the object is reconstructed with higher precision. All these procedures, performed without any a priori information about the shape of the object, produce a raw 3-D representation of the volumetric data in terms of voxels. The amount of information associated to a voxel-set is, in this case, only the one referred to the position of the ON-voxels in space.

3.4 Database

Data consist of a series of ten actions performed by five actors in five different repetitions, for a total of 250 distinct sequences (all the voxel sets are available at [77]). All the sequences begin and end with the actor still, standing with arms at his sides. Of the five actors, four are male (**Fede**, **Gabbo**, **Massi** and **Teo**) and one is a female (**Vale**). The need to have different performers is to develop a dataset that is able to validate our action classification system in terms of invariance under body structure, gender and personal action execution. Fig. 3.7 shows the initial voxel set of each actor in the sequence **Open-002**, in order to appreciate differences in size. The ten actions are:

Crouch: the actor performs the movement to curl up on the ground, keeps the squat

position for a while and then performs the inverse movement, returning to a standing position;

Grasp: the actor mimics the action of picking an object from the ground with his right hand;

Kick: the actor performs a wide kick with his right leg;

March: the actor mimics a typical military march, with large movements of legs and arms;

Move: the actor mimics a removal of an object with the right arm;

Open: the actor, standing up, raises his arms outstretched till forming an angle of 90° , opens them till the level of his shoulders, and then downs them along his body;

PointAt: the actor indicates with a finger a point in the sky, using his right arm;

Pull: the actor folds his arms on his chest and then back into position, keeping them close to his body;

Push: the actor raises his arms close to his body, then away, as if he is making space for himself in a confined space, and finally brings them back along the body with an upward movement;

Walk: the actor begins to walk, makes two or three steps and then stops.

This database of actions is not thought for a particular application of motion classification. We choose actions that involve both part (upper or bottom) or the whole human body in order to verify the ability of the acquisition setup and system to reconstruct different limbs of the human body in different postures and to check if this impacts the classification system performances. Moreover we choose action similar between each other, as **Walk** and **March**, **Move** and **PointAt**, **Open** and **Push**, in order to analyze the discriminative power of the proposed algorithm. Examples of volumetric reconstructions of some frames for each action class of our database, are shown in Figs 3.8-3.17.

3 Volumetric Human Body Data

3.4.1 Crouch

The actor performs the movement to curl up on the ground, keeps the squat position for a while and then performs the inverse movement, returning to a standing position.

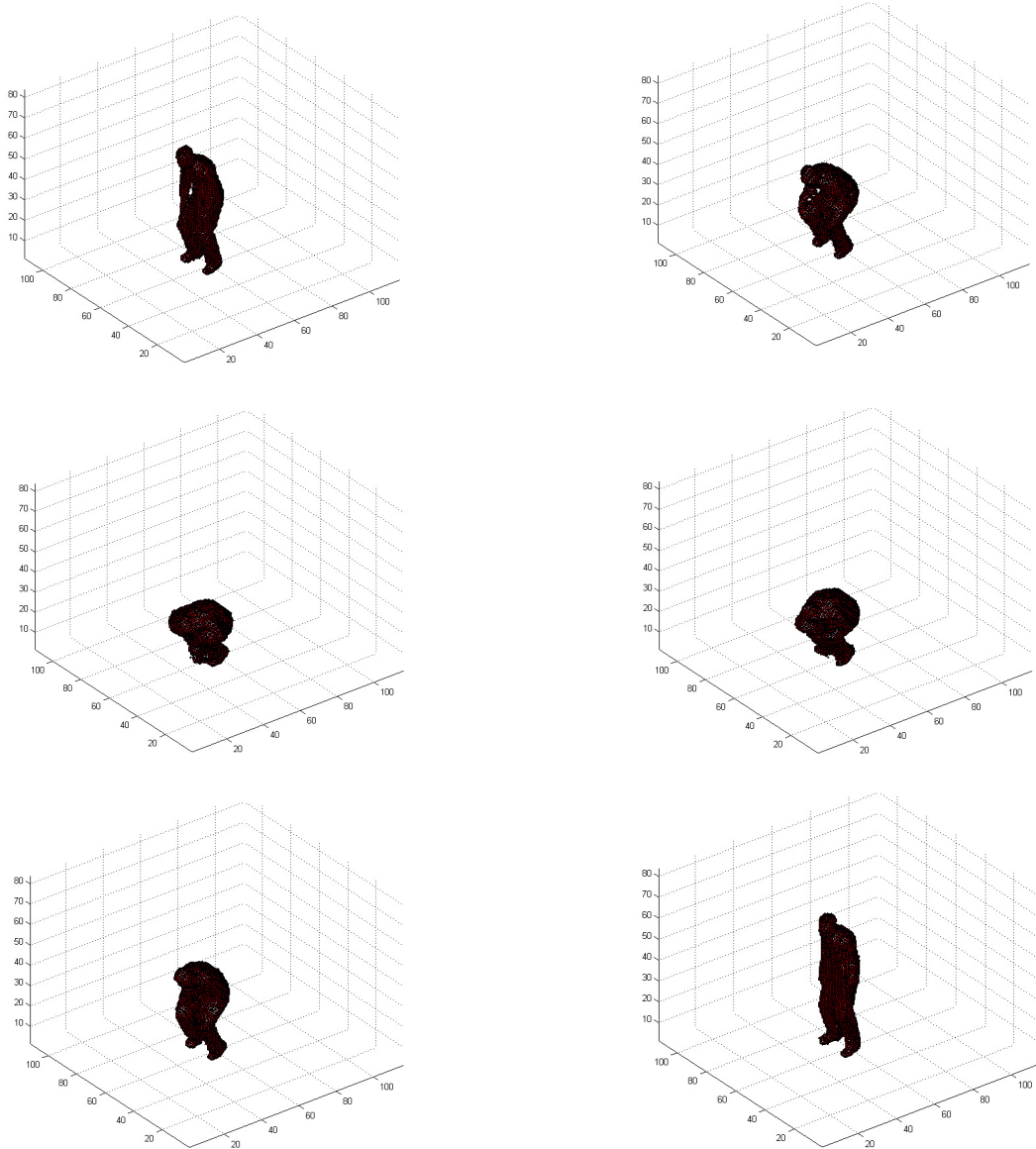


Figure 3.8: Crouch.

3.4.2 Grasp

The actor mimics the action of picking an object from the ground with his right hand.

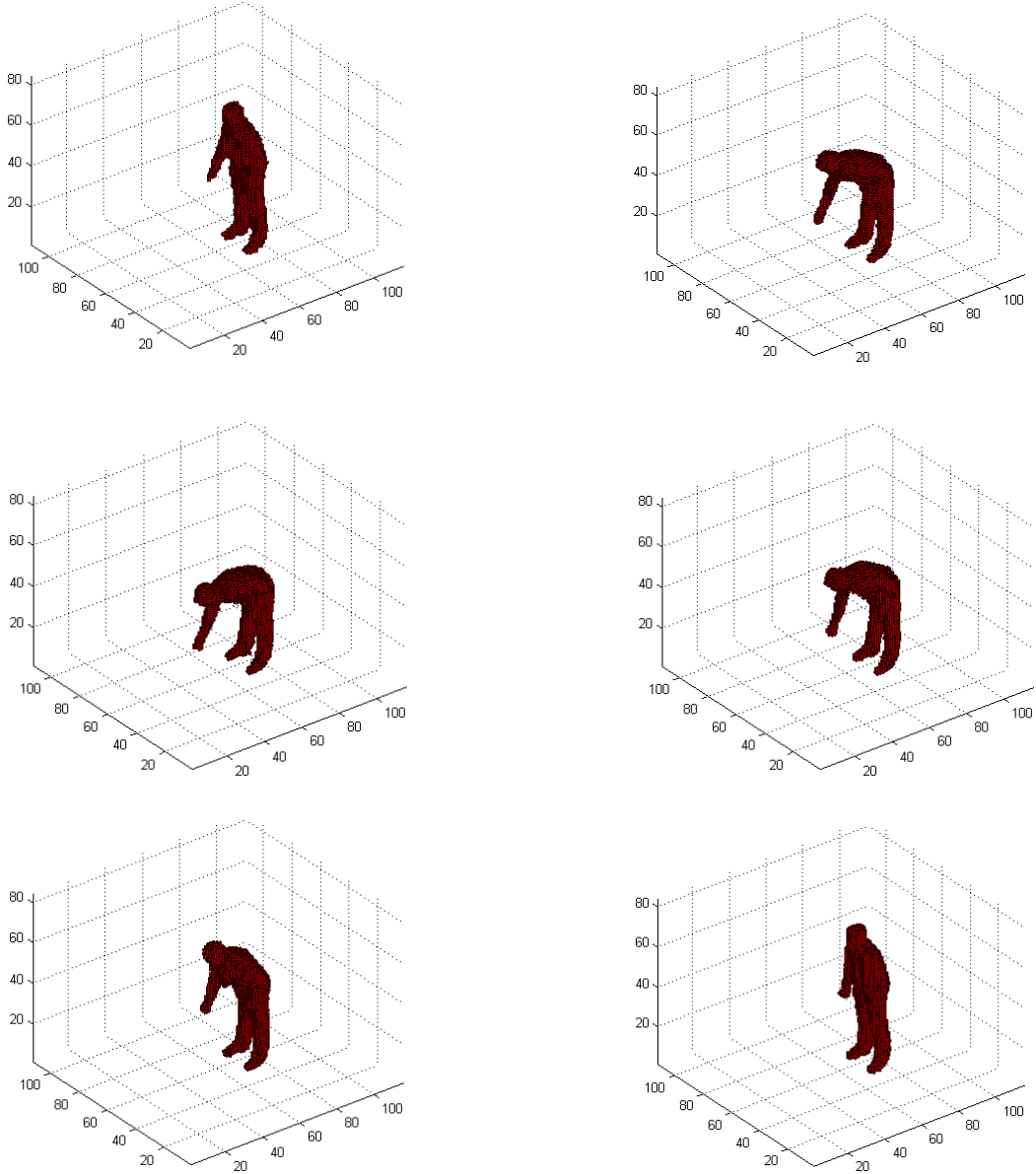


Figure 3.9: **Grasp**.

3.4.3 Kick

The actor performs a wide kick with his right leg.

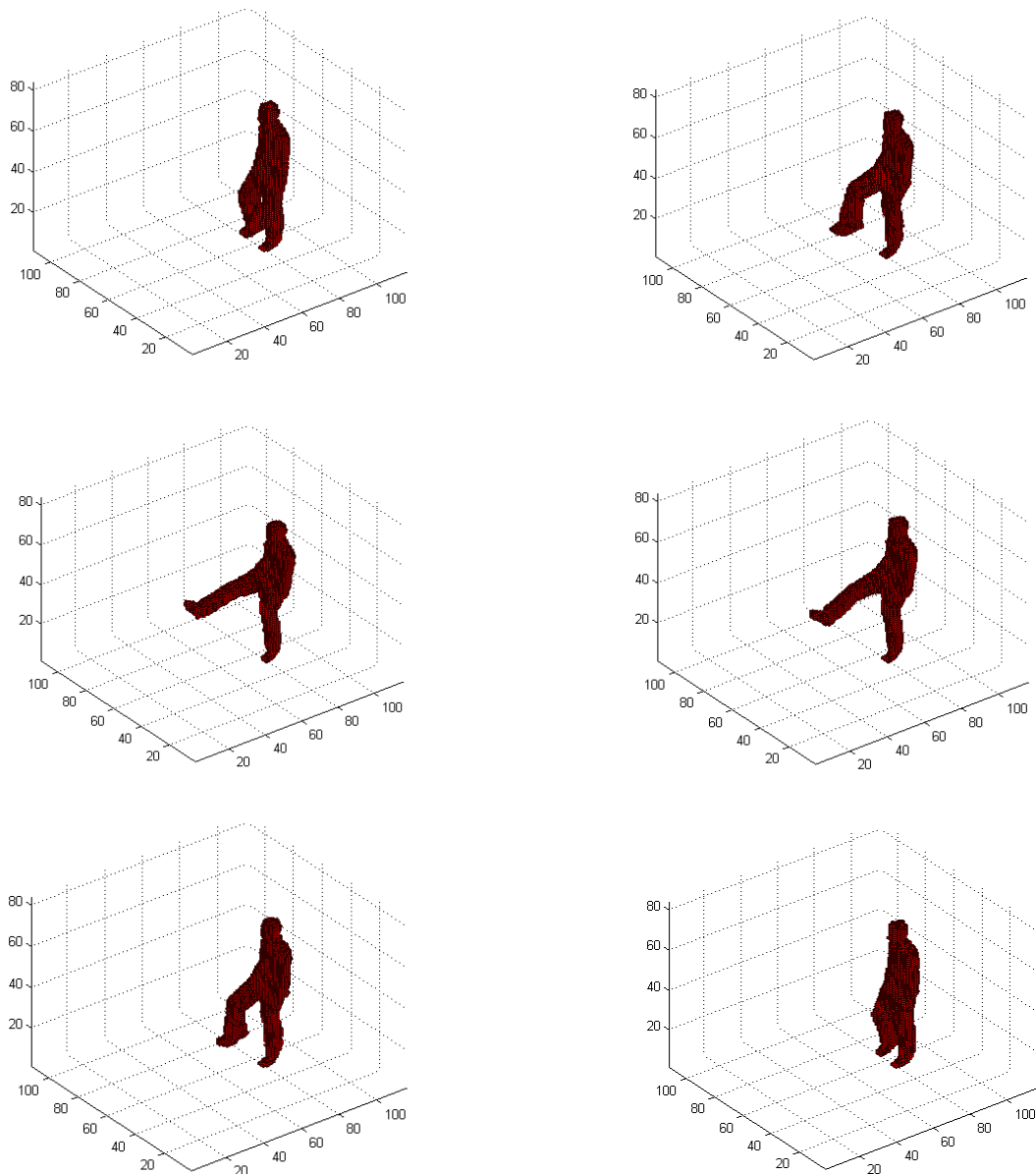


Figure 3.10: Kick.

3.4.4 March

The actor mimics a typical military march, with large movements of legs and arms.

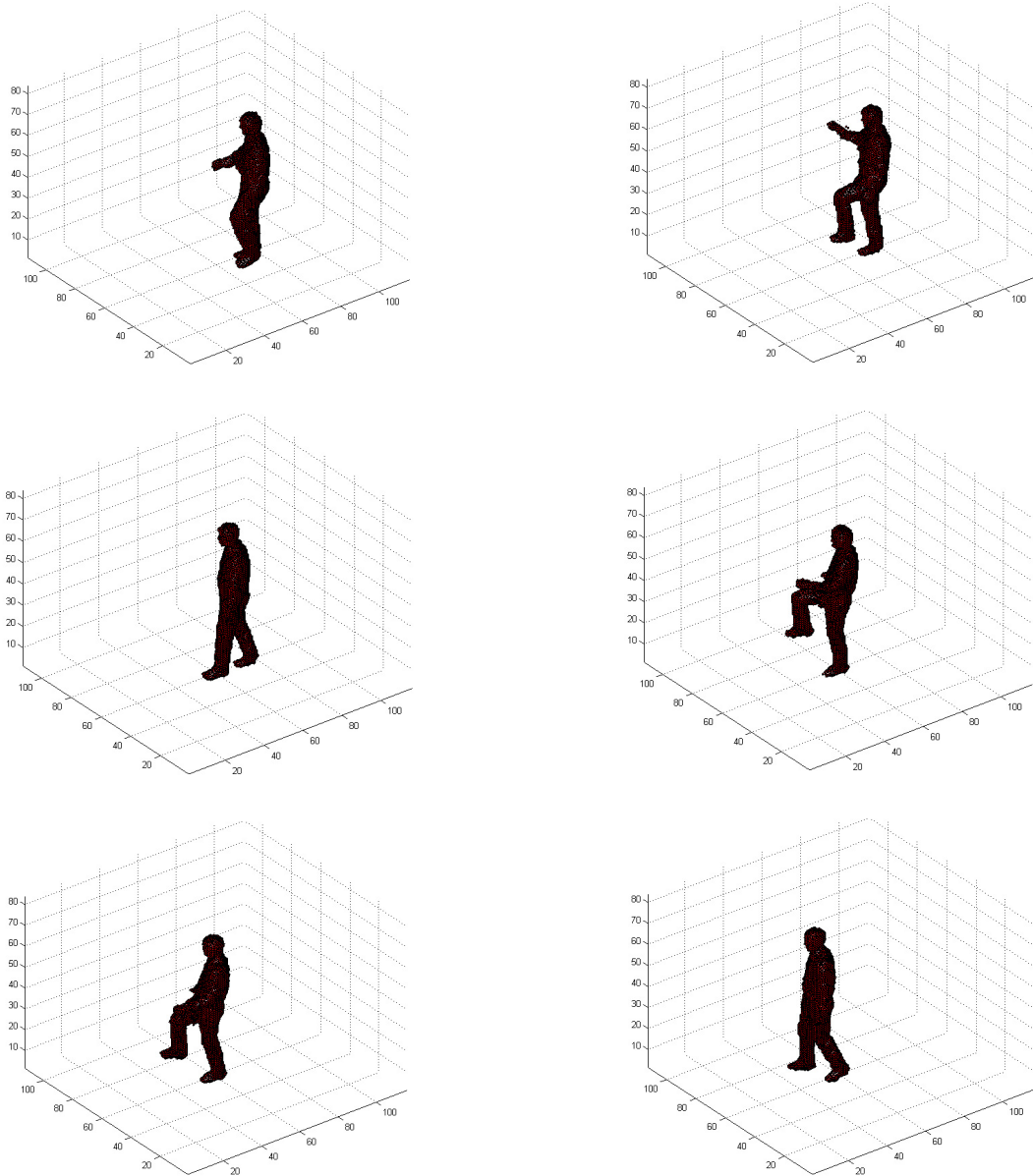


Figure 3.11: March.

3.4.5 Move

The actor mimics a removal of an object with the right arm.

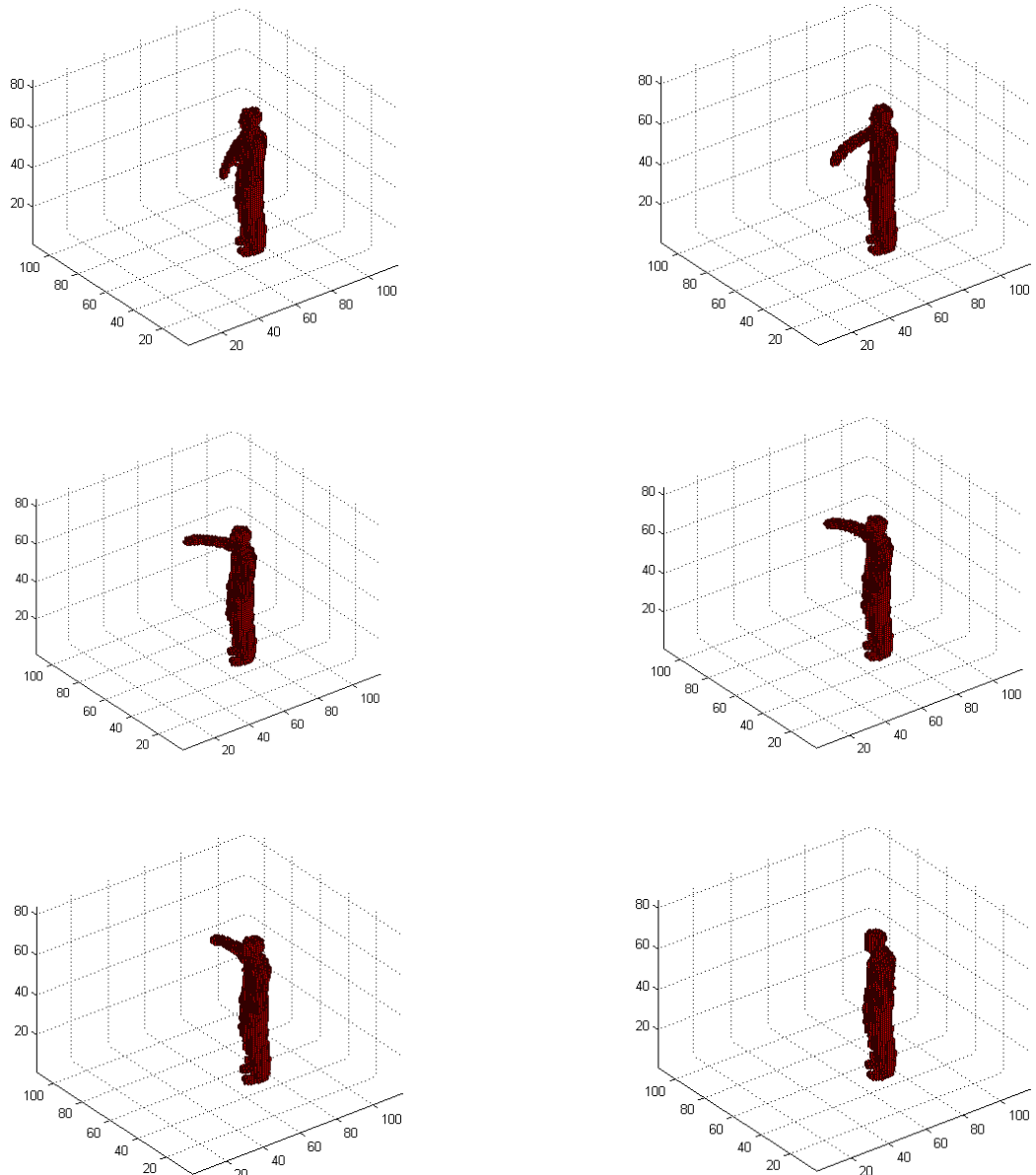


Figure 3.12: Move.

3.4.6 Open

The actor, standing up, raises his arms outstretched till forming an angle of 90° , opens them till the level of his shoulders, and then downs them along his body.

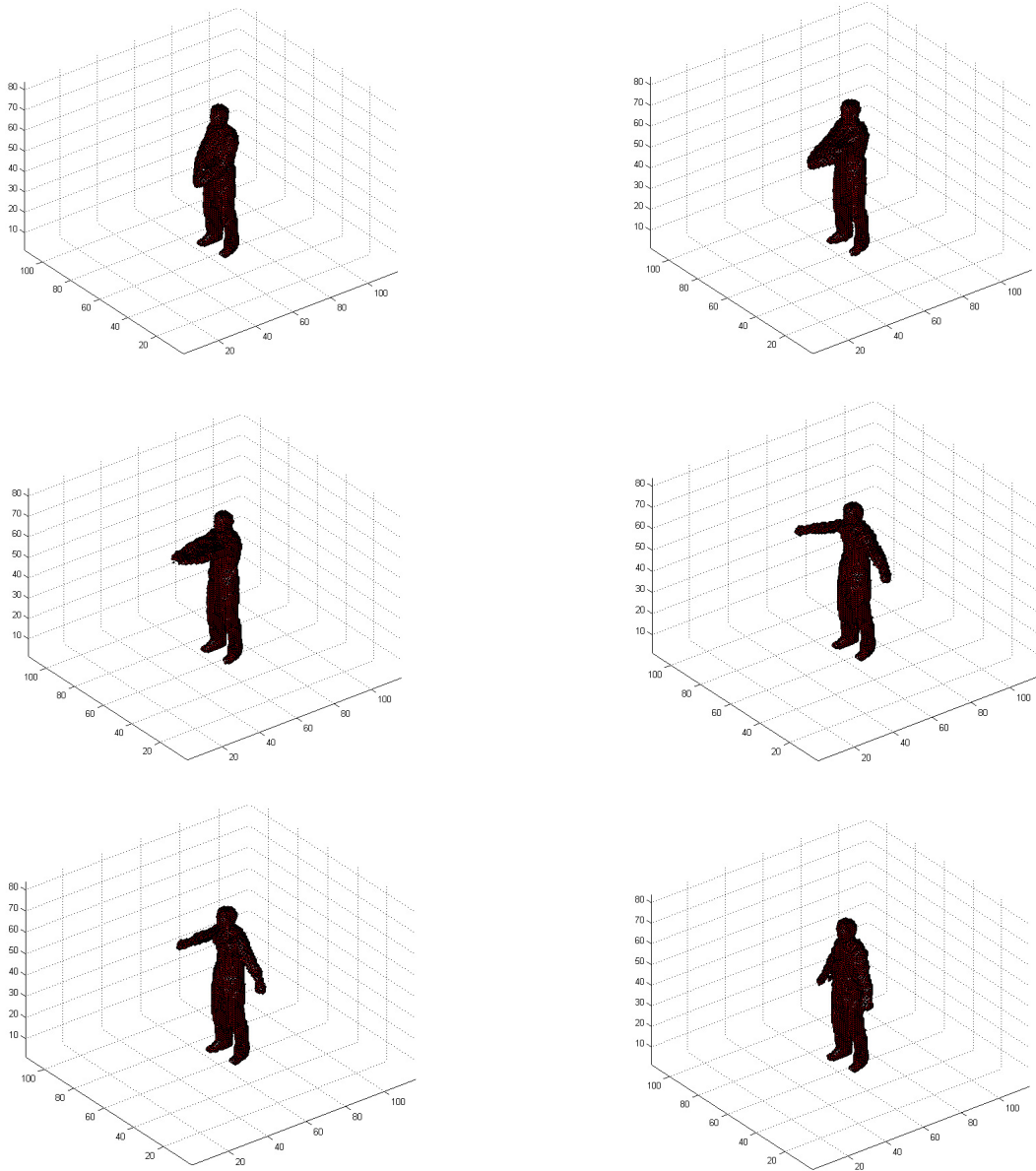


Figure 3.13: Open.

3.4.7 PointAt

The actor indicates with a finger a point in the sky, using his right arm.

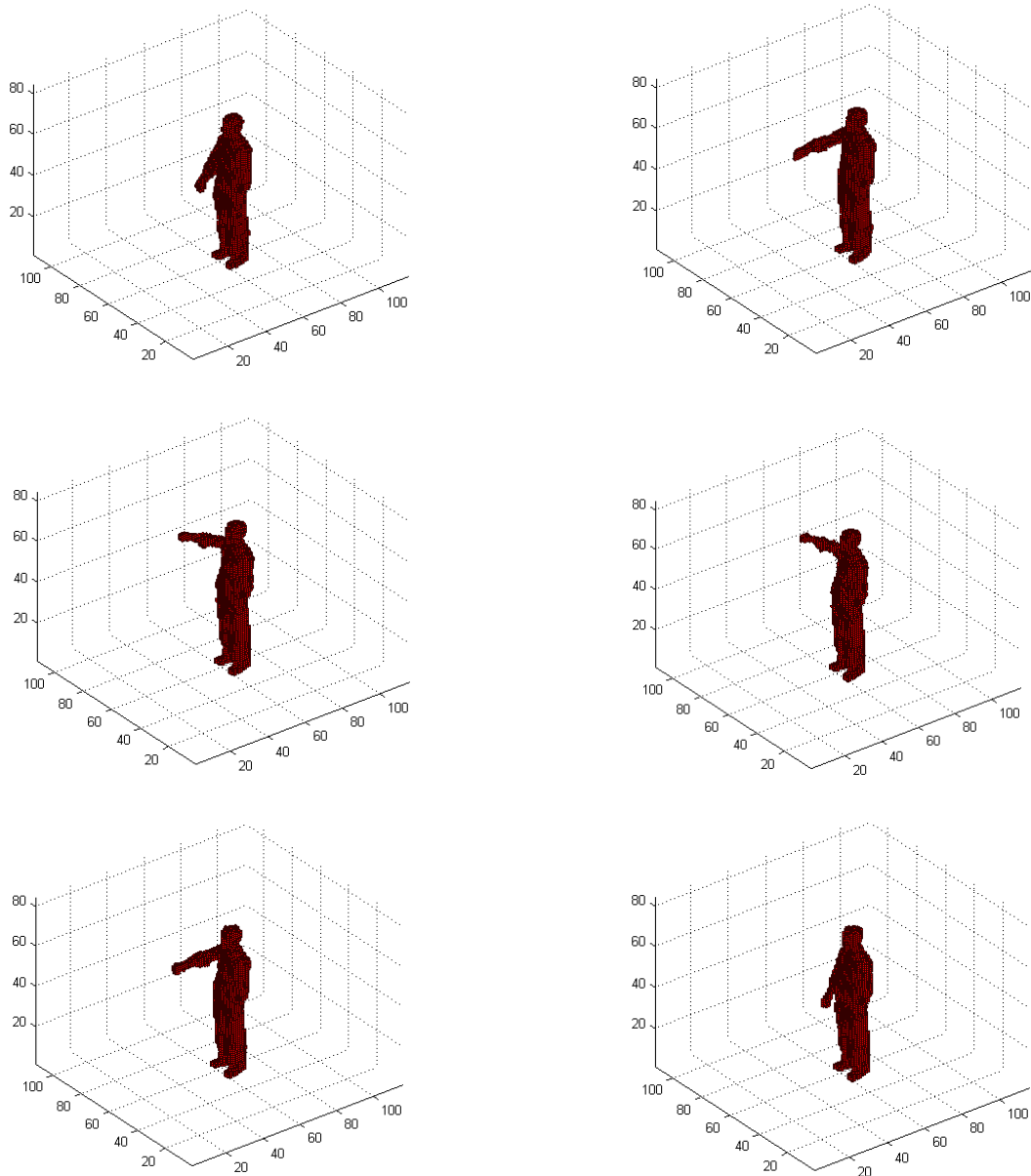


Figure 3.14: PointAt.

3.4.8 Pull

The actor folds his arms on his chest and then back into position, keeping them close to his body.

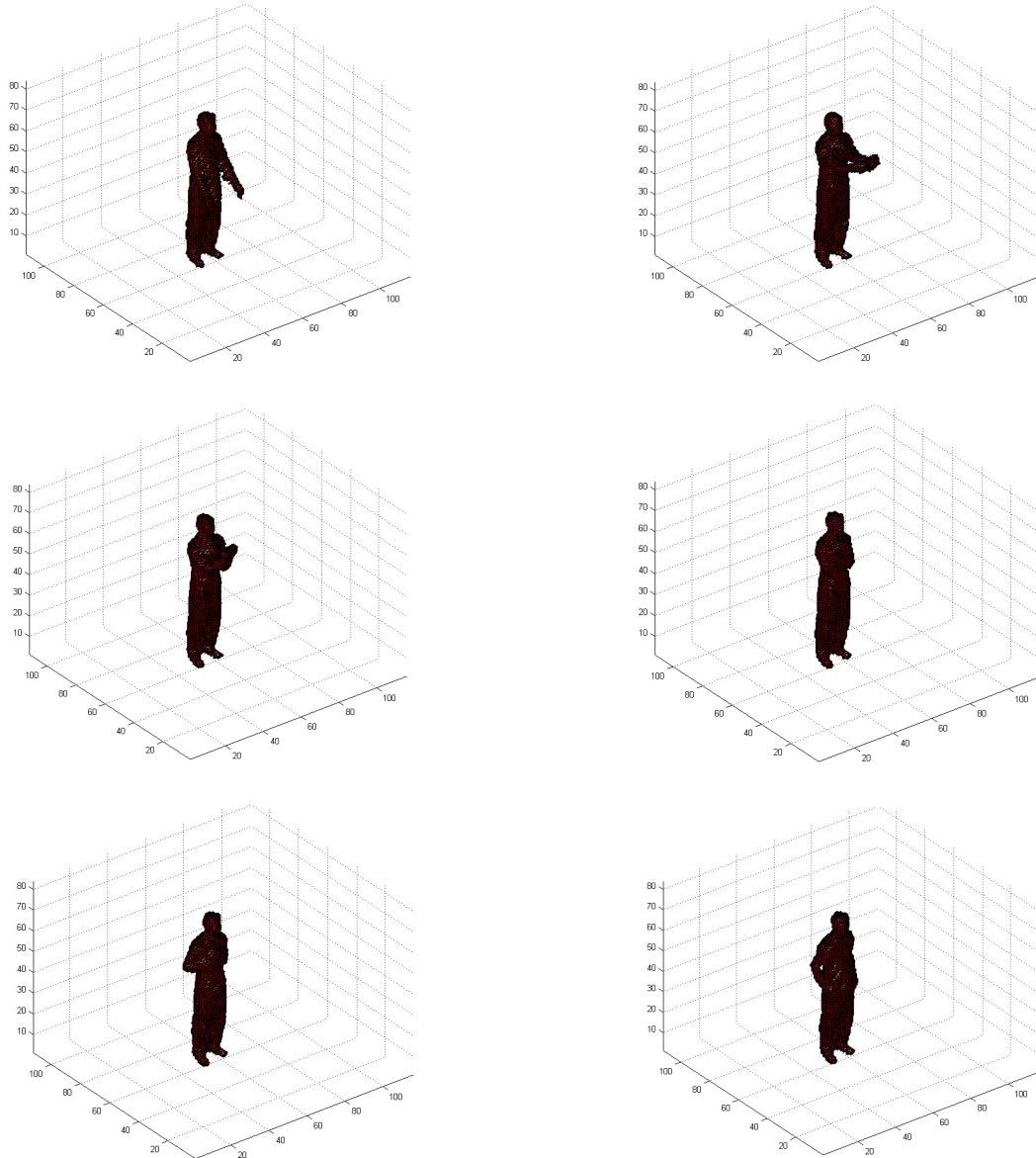


Figure 3.15: Pull.

3.4.9 Push

The actor raises his arms close to his body, then away, and finally brings them back along the body with an upward movement.

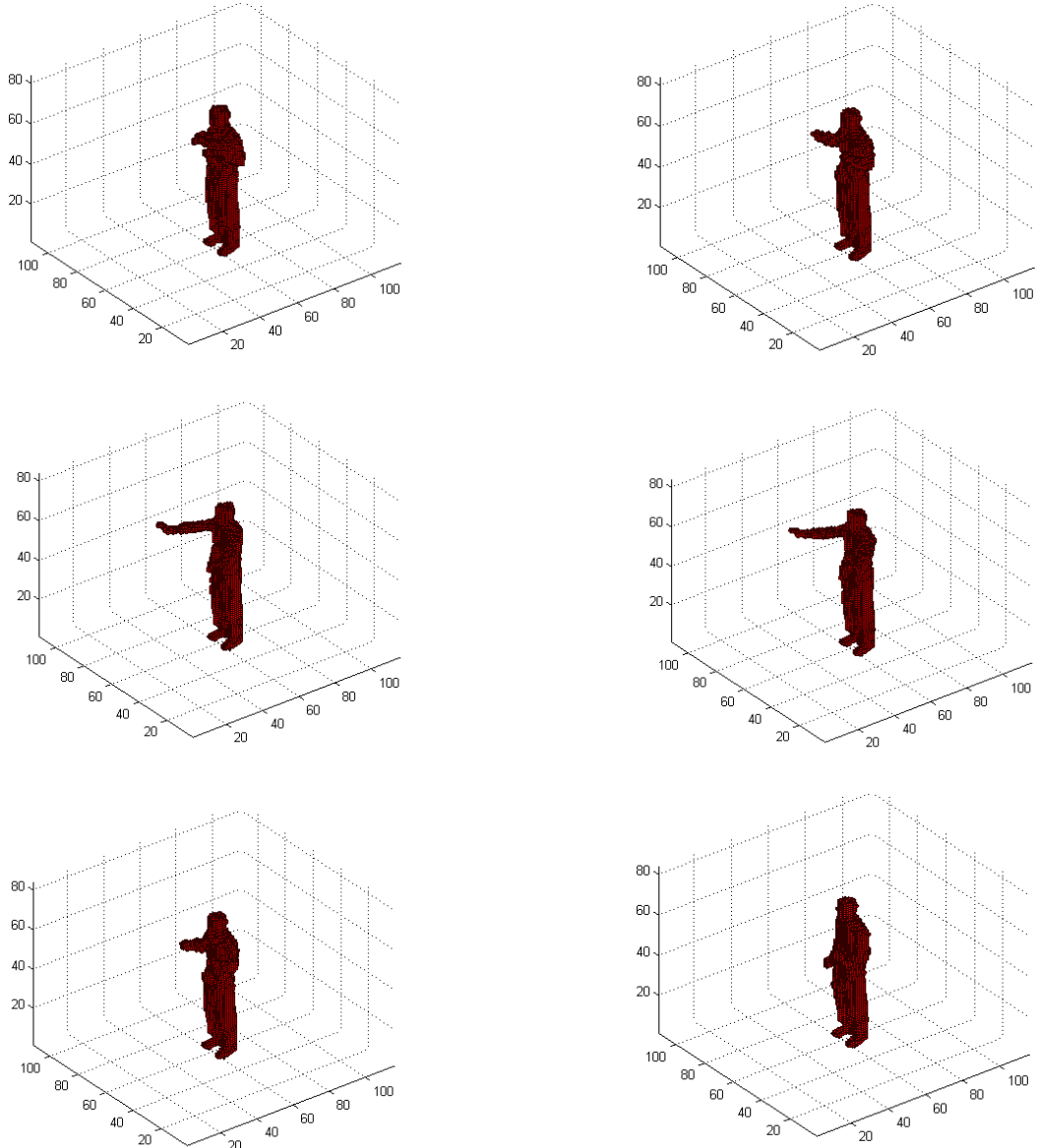


Figure 3.16: Push.

3.4.10 Walk

The actor begins to walk, makes two or three steps and then stops.

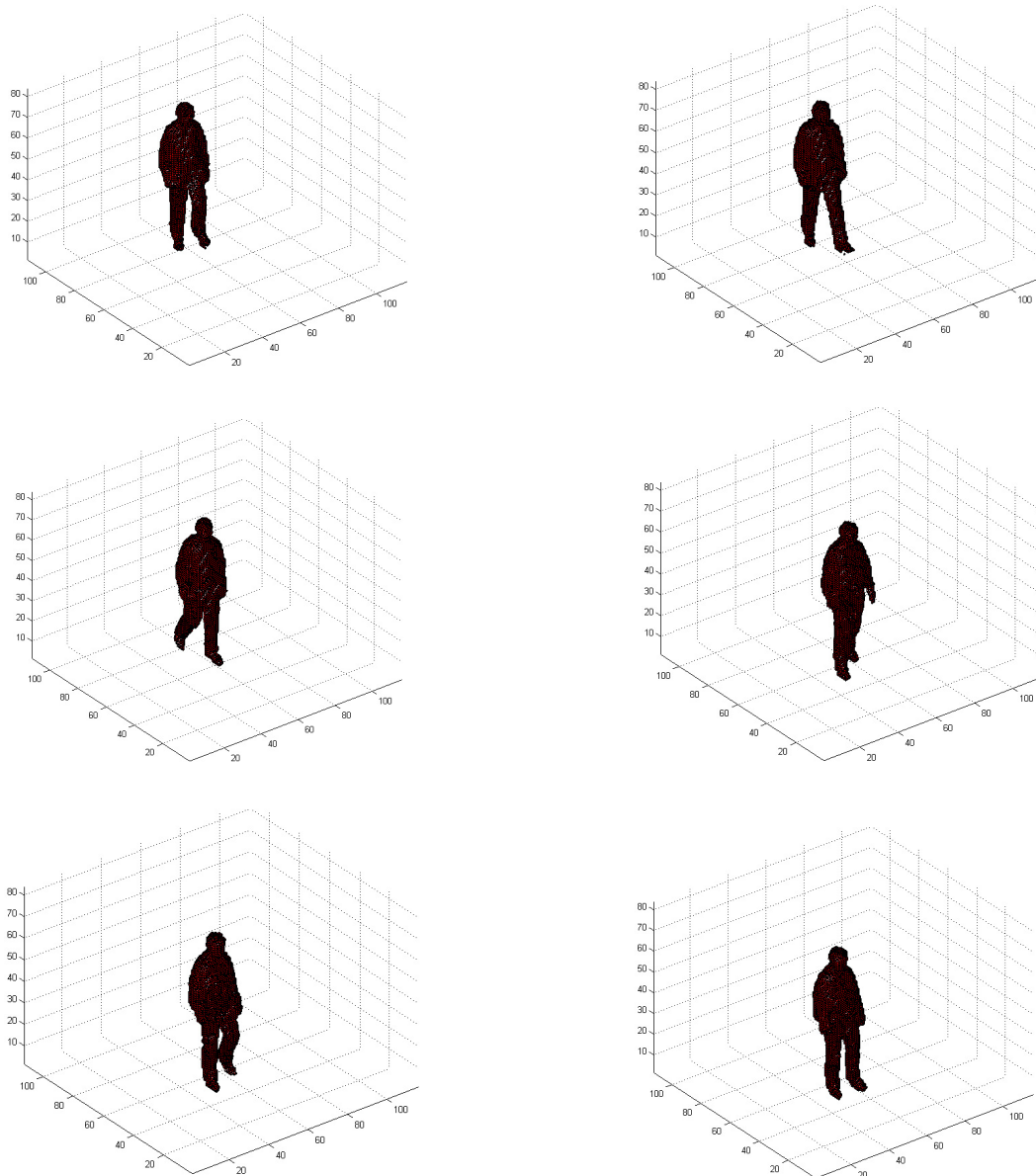


Figure 3.17: Walk.

4 3D Morphological Operators

This Chapter is devoted to introduce the 3D extensions of some of the most useful and used morphological operators, originally proposed for 2D binary images. These tools are developed in order to extract a compact and expressive representation of three dimensional binary digital objects. Mathematical morphology is the collection of theories and techniques for analysis and processing of geometrical structures. It was originally developed for binary images and was later extended to gray-scale functions and images [38], [106]. Starting from the simple concepts of *dilation* and *erosion*, more complicated morphology algorithms (usually representable as combinations of the two basic operators) were developed during years to aid different kind of solutions related to geometrical forms (filtering, pattern recognition, denoising, object representation). The goal of this Chapter is to build a general framework that extends the 2D morphological operators to 3D. This 3D extension is challenging for the thinning process, in which the definition of the 3D structuring elements is not a direct extension of the 2D ones and, moreover, it is not easy to figure out the result of the combination of different structuring elements. Inside the wide choice of possible structuring elements and combinations of them, the main challenging task is to select a set of structuring elements that produces the desired output (see Chapter 5 for a deeper analysis of the desired properties and a description of the proposed algorithm).

This Chapter is organized as follows. In Section 4.1 we introduce a few concepts from digital topology (please refer to [59] and [90] for a complete review on digital topology). In Section 4.2 the basic morphological operators for 3D images are described: erosion, dilation, opening, closing, and hit-or-miss transform. In Section 4.3 some of the most used morphological algorithms are described and extended for 3D objects: surface extraction, extraction of 26-connected components, morphological skeleton extraction, and thinning. Inside this latter Section 4.3, only a general introduction on the thinning process is presented. The proposed algorithm and the

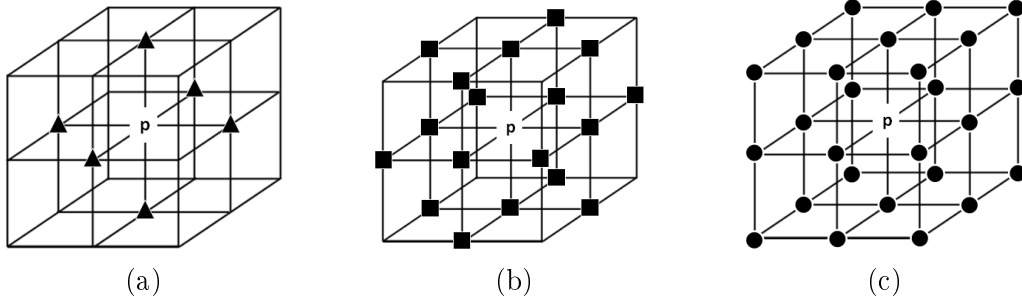


Figure 4.1: (a) The 6-neighbors of p represented by black triangles, (b) the 18-neighbors of p represented by black squares, and (c) the 26-neighbors of p represented as black circles.

desired output are described in Chapter 5. In the following, we refer to 3D object or 3D images as synonyms: a binary voxel representation of the volume occupied by the foreground object.

4.1 Review on Digital Topology

Let us consider the discrete space $\mathbb{Z}^3$. Each point $\mathbf{p}$ in this space is called *voxel* (from volume element) and it is defined by its three integer coordinates (p_x, p_y, p_z) . A voxel can be viewed as a cube, having 6 faces, 12 edges, and 8 corners.

Two voxels $\mathbf{p}$ and $\mathbf{q} \in \mathbb{Z}^3$ are 6-adjacent if they have a common face; they are 18-adjacent if they have a common face or edge and 26-adjacent if they have a common face, edge or corner. The set of 6-adjacent voxels to a voxel $\mathbf{p}$ is also known as the 6-neighborhood of $\mathbf{p}$ denoted by $N_6(\mathbf{p})$. Similarly, $N_{18}(\mathbf{p})$ and $N_{26}(\mathbf{p})$ are the 18 and 26-neighborhoods of $\mathbf{p} \in \mathbb{Z}^3$ (represented in Fig. 4.1(a)-(c)). More formally, these three neighborhoods are considered as:

$$N_{26}(\mathbf{p}) = \{\mathbf{p}' \in \mathbb{Z}^3 : \max\{|p_x - p'_x|, |p_y - p'_y|, |p_z - p'_z|\} \leq 1\}, \quad (4.1)$$

$$N_{18}(\mathbf{p}) = \{\mathbf{p}' \in \mathbb{Z}^3 : \{|p_x - p'_x| + |p_y - p'_y| + |p_z - p'_z|\} \leq 2\} \cap N_{26}(\mathbf{p}), \quad (4.2)$$

$$N_6(\mathbf{p}) = \{\mathbf{p}' \in \mathbb{Z}^3 : \{|p_x - p'_x| + |p_y - p'_y| + |p_z - p'_z|\} \leq 1\}. \quad (4.3)$$

The 6-neighbors of $\mathbf{p}$ determine six major directions (Fig. 4.2): Up, Down, North,

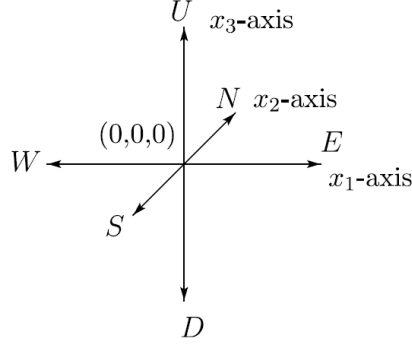


Figure 4.2: The six major direction in the $\mathbb{Z}^3$ space: Up (U), Down (D), North (N), South (S), West (W), East (E).

South, West, East; respectively denoted by U , D , N , S , W and E .

An n -path is a sequence of voxels $\mathbf{p}_1, \dots, \mathbf{p}_k$ with $\mathbf{p}_i$ n -adjacent to $\mathbf{p}_{i+1}$, where n could be 6, 18, or 26. An n -connected component is then a set of voxels such that any two such voxels are connected by a n -path included in that component.

A *3D binary picture* [59] is a mapping that assigns a value of 0 or 1 to each point with integer coordinates in the 3D digital space denoted by $\mathbb{Z}^3$. Points having the value of 1 are called black points and those with the value of 0 are called white points. Black points form the object of a picture (foreground), while white points form the background. It is assumed that any picture contains finitely many black points. More formally a 3D binary digital picture is described by the quadruple $P = (\mathbb{Z}^3, m, n, B)$, where $B \subseteq \mathbb{Z}^3$ is the set of black voxels representing the object in the picture (also known as object voxels), while $\mathbb{Z}^3 \setminus B$ represents the background (white) voxels. The pair (m, n) specifies respectively the object and background connectivity. In order to avoid topological paradoxes such as objects being both connected and disconnected [59], different values must be chosen for m and n . Common choices are (26,6) and (18,6). We consider (26,6)-pictures.

In 3D pictures a *cavity* is a background connected component surrounded by an object component (an empty space inside the object). More formally a white component C is called a *cavity* if all the voxels in its 6-envelope $N_e^6(C)$ are black [59].

There is an additional concept, called *hole*, in 3D pictures. A *hole* or *tunnel* is formed from 0's, but it is not a cavity. While a mathematical definition of a tunnel is difficult [59], [98], [109], we can turn to intuitive examples of tunnels in well-known objects. The center hole of a donut or the empty space between the handle of a

4 3D Morphological Operators



Figure 4.3: (a) A cavity in a black component, and (b) a tunnel in a black component. White voxels are not shown [128].

teapot and its main body are examples of tunnels. For a mathematical definition we follow Xie *et al.* [128] that expand the definition introduced by Bertrand [9]. Let L be a closed 26-path in a black component S . L' is another closed 26-path in $N^{26}(L) \cap S$. We call the mapping from L to L' a *transformation* of L , if all the voxels in L' are 26-adjacent to L and all the voxels in L are also 26-adjacent to L' . A black voxel set S contains a *tunnel* or *hole*, if there exists a closed 26-path in S that cannot be transformed into a single black voxel.

Fig. 4.3 illustrates cavity (a) and tunnel (b) typical configurations. Algorithms to detect and characterize cavities and tunnels in a 3D object are described in [109].

4.2 Morphological Operators

4.2.1 Erosion

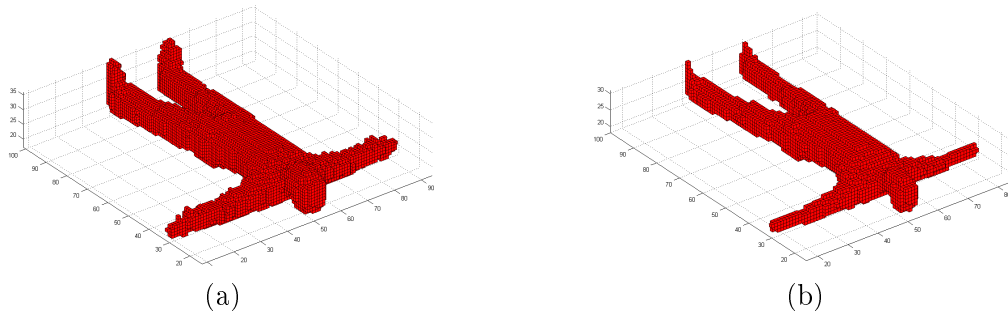


Figure 4.4: Morphological erosion using a structuring element E equal to a $3 \times 3 \times 3$ block of voxels set as 1. (a) Original binary 3D object B . (b) Erosion of B by E .

When an erosion is performed, the center of the structuring element E is placed

on each object voxel $\in B$. If, for each voxel of the structuring element having value equal to 1, the corresponding voxel of the image is a foreground voxel, then the original voxel remains unchanged, otherwise it is set to 0. The erosion of B by E can be understood as the locus of voxels reached by the center of E , when E moves completely inside B . It is usually indicated as $B \ominus E$. Its effect is illustrated in Fig. 4.4(b).

4.2.2 Dilation

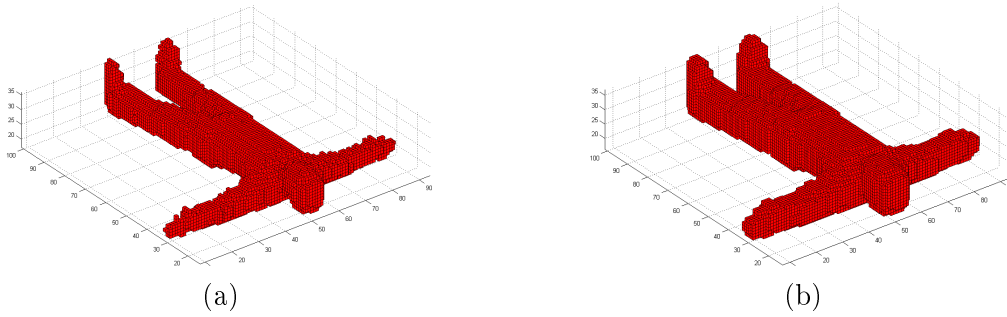


Figure 4.5: Morphological dilation using a structuring element E equal to a $3 \times 3 \times 3$ block of voxels set as 1. (a) Original binary 3D object B . (b) Dilation of B by E .

When a dilation is performed, the center of the structuring element E is placed on each object voxel $\in B$ and all the image voxels corresponding to the foreground voxels of the structuring element are set equal to 1. The dilation of B by E can be understood as the locus of voxels covered by E , when the center of E moves inside B . It is usually indicated as $B \oplus E$. Its effect is illustrated in Fig. 4.5(b).

4.2.3 Opening

The opening of B by E is obtained by the erosion of B by E followed by the dilation of the resulted image by E . The structuring element moves inside the image: all the foreground voxels, that can be covered by the structuring element so that it is completely contained in the foreground region, remain unchanged, while all the foreground voxels that could not be reached by the structuring element without covering also background voxels are set to 0. The opening of B by E ($B \circ E = (B \ominus E) \oplus E$) is the locus of translations of the structuring element E inside the image B . Its effect is illustrated in Fig. 4.6(b).

4 3D Morphological Operators

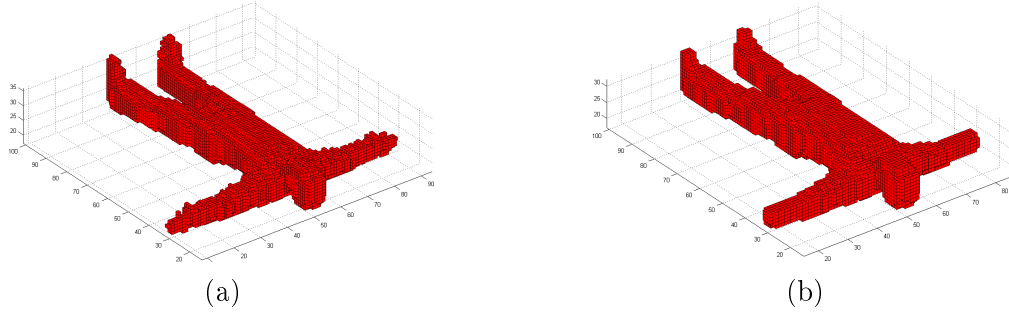


Figure 4.6: Morphological opening using a structuring element E equal to a $3 \times 3 \times 3$ block of voxels set as 1. (a) Original binary 3D object B . (b) Opening of B by E .

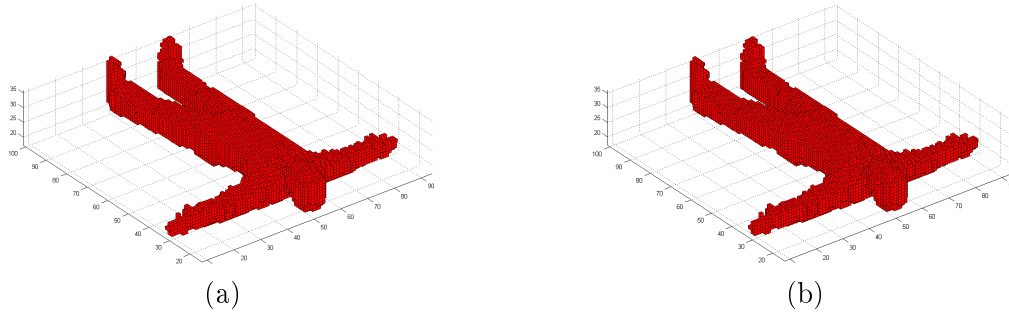


Figure 4.7: Morphological closing using a structuring element E equal to a $3 \times 3 \times 3$ block of voxels set as 1. (a) Original binary 3D object B . (b) Closing of B by E .

4.2.4 Closing

The closing of B by E is obtained by the dilation of B by E followed by the erosion of the resulted image by E . The structuring element moves around each object in the image: for each background voxel, if the structuring element can touch it without intersecting any foreground voxel, the original voxel remains unchanged, otherwise it is set as foreground. The closing of B by E ($B \bullet E = (B \oplus E) \ominus E$) is the complement of the locus of the translations of the symmetric of the structuring element outside the image B . It produces an output similar to dilation, but more close to the original object. Its effect is illustrated in Fig. 4.7(b).

4.2.5 Hit-or-Miss Transform

The Hit-or-Miss (HoM) transform of B uses the morphological erosion and a pair of *disjoint* structuring elements $\{E, F\}$. Formally two sets are disjoint if their intersection is the empty set. Considering structuring elements defined on a local neighbor-

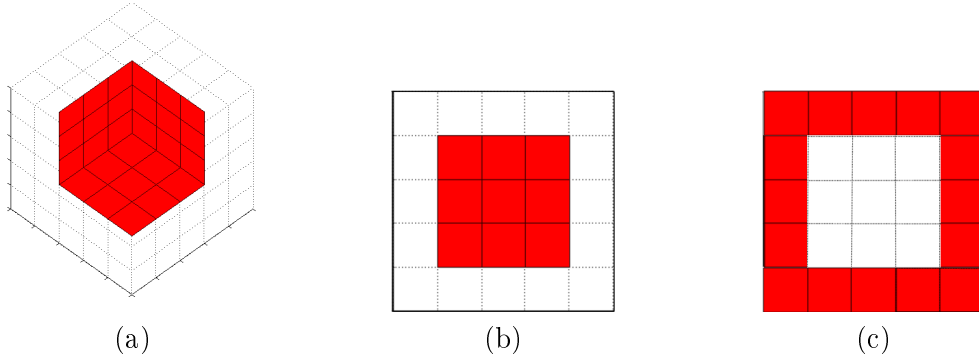


Figure 4.8: (a) The structuring element E is a block of $3 \times 3 \times 3$ black voxels in the center of W defined as a space of $5 \times 5 \times 5$ voxels. Top view of (b) the structuring element E and (c) its local background F . Foreground voxels are represented in red.

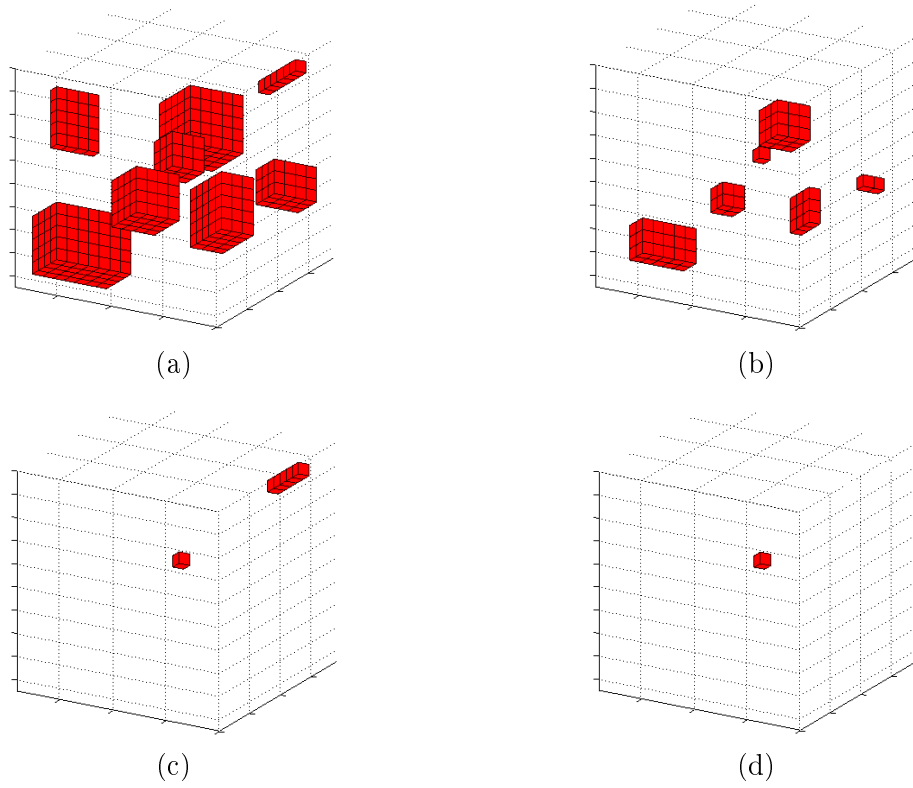


Figure 4.9: Hit-or-Miss transform using a structuring element E equal to a $3 \times 3 \times 3$ block of voxels set as 1 in the center of a local neighborhood W of $5 \times 5 \times 5$ voxels. (a) Original binary 3D object B . (b) Erosion of B by E . (c) Erosion of B^C by $F = W - E$. (d) Hit-or-Miss transform of B by E .

hood W ($E \subset W \subset \mathbb{Z}^3$), two structuring elements are disjoint if $F = W - E$. $W - E$ is called also *local background*.

4 3D Morphological Operators

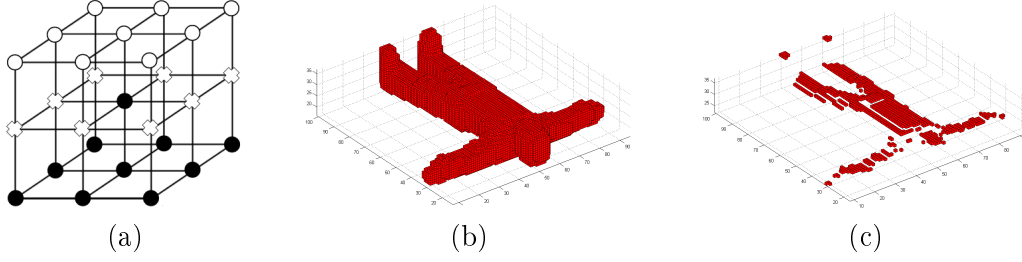


Figure 4.10: Simplified Hit-or-Miss transform using a structuring element E. (a) Structuring element E represented as a lattice structure where each point represents a voxel and its value is “0” if the point is white, “1” if the point is black, or “do not care” if the point is a cross. (b) Original binary 3D object B. (c) Simplified Hit-or-Miss transform of B by E.

Let consider, for example, as local neighborhood W a space of $5 \times 5 \times 5$ voxels and as structuring element E a block of $3 \times 3 \times 3$ black voxels in the center of W . The disjoint structuring element F is therefore a $5 \times 5 \times 5$ set of black voxels with the central $3 \times 3 \times 3$ voxels equal to 0. The structuring element E and its local neighborhood are represented in Fig. 4.8(a) (the top views of E and F are represented in Fig. 4.8(b) and (c) respectively).

The Hit-or-Miss transform of B by $\{E, F\}$ is obtained by the difference between the erosion of B by E and the erosion of B^C by F (the local background). B^C is the complement of B (defined in $\mathbb{Z}^3$). It means that a point $\mathbf{p}$ in B belongs to the HoM transform, if E translated in $\mathbf{p}$ fits in B and F translated in $\mathbf{p}$ fits the background of B . The HoM transform of B by E (defined on W) is therefore:

$$B \circledast E = (B \ominus E) \cap (B^C \ominus (W - E)) \quad (4.4)$$

and is the set of voxels in B which neighborhood exactly matches the structuring element E . Fig. 4.9(d) shows the effect of the HoM transform considering the structuring element shown in Fig. 4.8.

In many morphological algorithms the simplified version of the HoM (SHoM) transform is implemented, where the structuring element has usually the values of “0”, “1”, or “do not care”. The idea is still to find the exact match between the neighborhood of $\mathbf{p}$ (voxel of B) and the structuring element E , with the possibility that the voxels marked in E as ‘do not care’ can match both with foreground or background voxels of B . Practically the structuring element moves around each object in the image and the two local neighborhood are tested: a value of “0” on E must match a background voxel, a value of “1” on E must match an object voxel, while a value of “do not

care” on E can match either a background or an object voxel. If this condition is verified, the examined voxel on B remains unchanged, otherwise it becomes a background voxel. Fig. 4.10 shows the effect of the SHoM transform (c) considering the structuring element shown in Fig. 4.10(a).

4.3 Morphological Algorithms

4.3.1 Surface Extraction

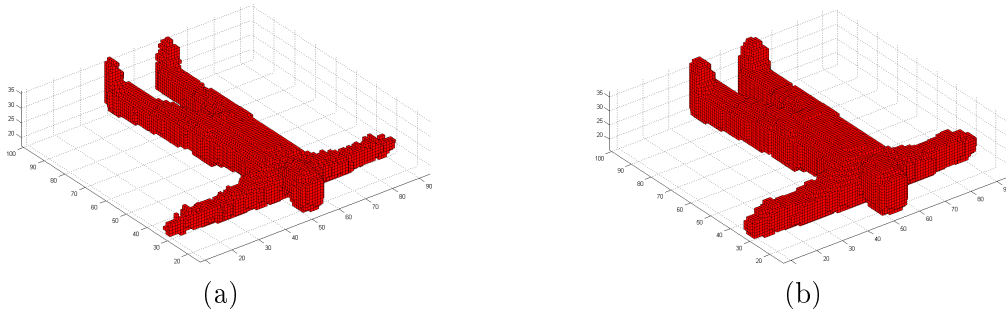


Figure 4.11: Surface Extraction. (a) Original binary 3D object O . (b) Surface.

There are different possible methods to extract the object surface having its volume voxel occupancy. For some applications (see Chapters 6 and 7 for some examples) we find that the best method is to operate a dilation of B by E and to define the surface as the difference between the result and the original object:

$$\beta(B) = (B \oplus E) - B. \quad (4.5)$$

A 3D object B and its extracted surface are shown in Fig. 4.11.

4.3.2 Extraction of Connected Elements

This algorithm is useful, for example, for counting how many n -connected blocks compose a 3D image and also for counting by how many voxels each block is made. Starting from a foreground voxel belonging to B ($\mathbf{p}_0$), the goal is to find all the connected components to that voxel by following the iterative procedure:

$$\mathbf{p}_k = (\mathbf{p}_{k-1} \oplus E) \cap B \quad k = 1, 2, 3, \dots \quad (4.6)$$

4 3D Morphological Operators

The procedure terminates when $\mathbf{p}_k = \mathbf{p}_{k-1}$ and produces the connected component B_i . Then the same iterative procedure starts again on another foreground voxel of $B - \{\cup_{j=1}^i B_i\}$ till no more foreground voxels remain.

4.3.3 Morphological Skeleton Extraction

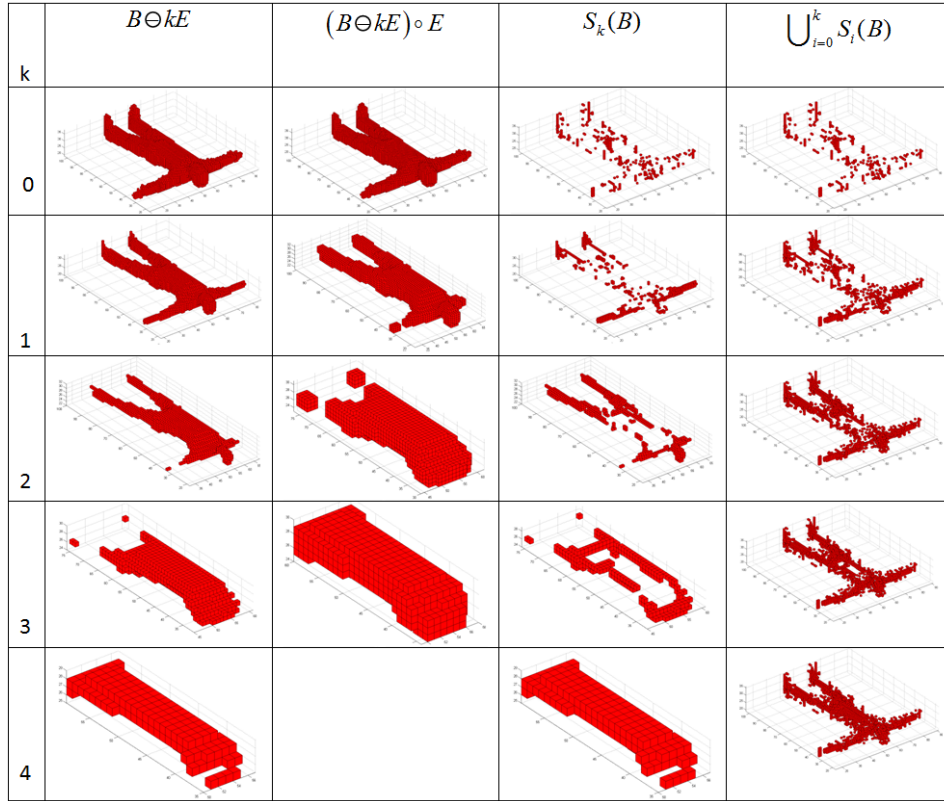


Figure 4.12: Step by step representation of the morphological skeleton extraction algorithm from the original object (top-left image) till its morphological skeleton (bottom-right image).

The morphological skeleton operator for the voxel set B can be expressed in terms of sequence of morphological operations of erosion and opening [38]. Given the structuring element E and indicating with $\ominus k$ a sequence of k morphological erosions and with $\circ$ a morphological opening, the morphological skeleton ($S(B)$) can be expressed as:

$$S(B) = \bigcup_{k=0}^K S_k(B) \quad (4.7)$$

where:

$$S_k(B) = (B \ominus kE) - ((B \ominus kE) \circ E). \quad (4.8)$$

This procedure is iterated until $B \ominus KE$ is an empty set. This algorithm, as well as its version in 2D, does not produce a connected skeleton even if the original object is connected. Fig. 4.12 is a schematic representation of the morphological skeleton extraction algorithm from the original object (top-left image) till its morphological skeleton (bottom-right image).

4.3.4 Thinning

Thinning is a morphological operation that is used to remove selected foreground voxels. This tool is useful for many Computer Vision applications because it provides a local lower dimensional characterization of the solid. In this Section, we illustrate how the thinning algorithm works in general, describing the algorithm proposed by Palágyi and Kuba [93]. This is one of the most known and used algorithm for 3D thinning of binary (26,6) pictures. Since the proposed thinning algorithm (described in the next Chapter) is inspired to their method, it seems appropriate to first describe the Palágyi and Kuba algorithm [93] in order to better appreciate the innovation introduced by our changes.

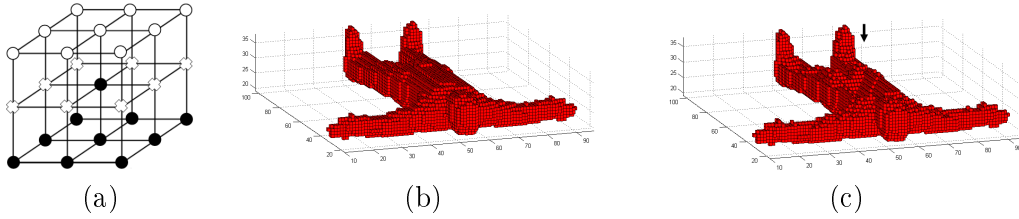


Figure 4.13: Thinning using a structuring element E. (a) Structuring element E represented as a lattice structure where each point represents a voxel and its value is “0” if the point is white, “1” if the point is black, or “do not care” if the point is a cross. (b) Original binary 3D object B. (c) Thinning of B by E.

A thinning algorithm is a *reduction operator* because it changes some 1’s (black or object voxels) to 0’s (white or background voxels), but does not alter 0’s. The new value of each object voxel depends on its neighborhood (only $3 \times 3 \times 3$ neighborhoods are considered in this thesis, but extensions to any dimensions of neighborhoods is straightforward). The thinning of B by structuring element E , denoted by $B \otimes E$, can be defined in terms of the Hit-or-Miss Transform [106]:

$$B \otimes E = B - (B \circledast E) = B \cap (B \circledast E)^c, \quad (4.9)$$

4 3D Morphological Operators

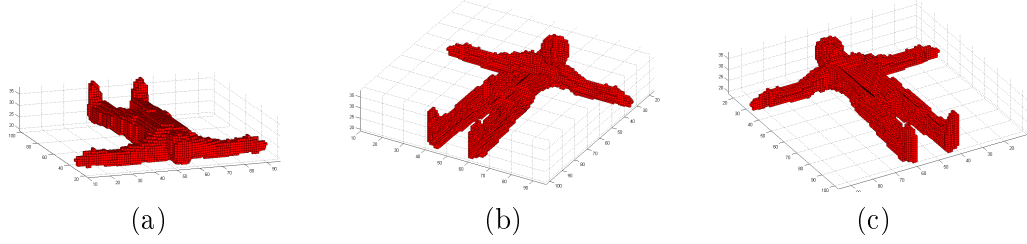


Figure 4.14: Thinning using structuring element $\{E\}$. (a) Original binary 3D object B . (b)-(c) Thinning of B by $\{E\}$ from two different point of views.

where $\otimes$ denotes the Simplified Hit-or-Miss transform. The SHoM transform preserves voxels whose neighborhoods match the elements of E . The structuring element origin scans each voxel of B and, if there is a perfect overlap between the neighborhood voxels with those of the structuring element, the voxel on which the structuring element lays is set to 0 (background), otherwise it is left at 1. The use of the SHoM transform allows to define structuring elements with values of “0”, “1”, or “do not care”, as described in Section 4.2.5.

Actually equation (4.9) represents only one *thinning cycle*. The thinning process is iterative: from the thinned image $B_1 = B \otimes E$, the process is iterated until no more black voxels can be deleted:

Begin

$$Y = B;$$

repeat

$$X = Y \otimes E;$$

$$Y = X;$$

until no points are deleted;

end.

Fig. 4.13(c) shows the effect of thinning B (b) with the structuring element E (a).

Moreover the usual process is to thin B using a sequence of structuring elements $\{E\} = \{E^1, E^2, \dots, E^n\}$. A thinning iteration is therefore computed as:

$$B \otimes \{E\} = (((B \otimes E^1) \otimes E^2) \dots) \otimes E^n. \quad (4.10)$$

We call a *thinning cycle* an iteration. Let $P = (\mathbb{Z}, 26, 6, B, W)$ be the finite 3D picture to be thinned. We use $P_m = (B_m, W_m)(m = 0, \dots, n)$ to denote the transient

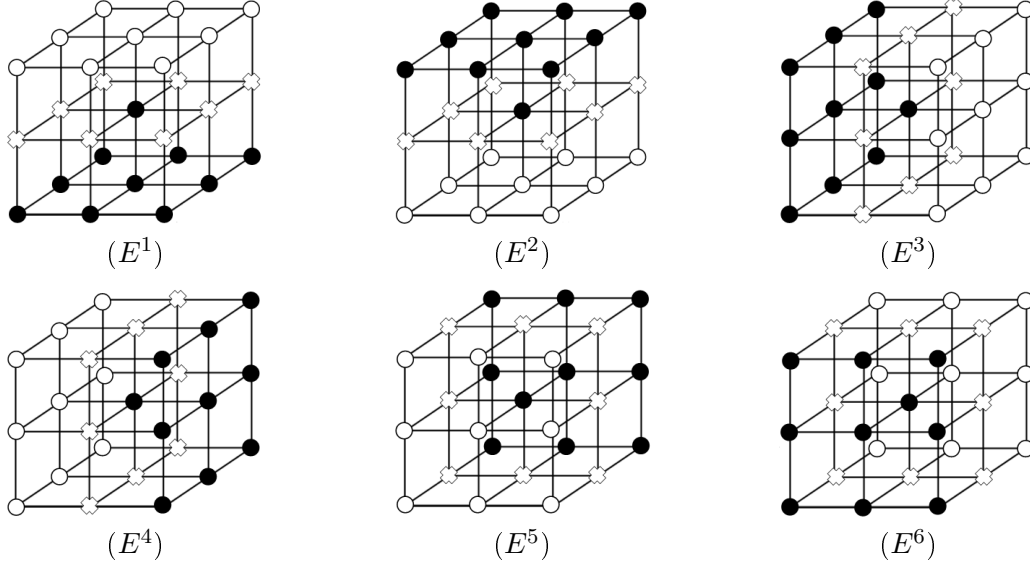


Figure 4.15: Example of structuring elements for thinning. Each structuring element E^i is represented as a lattice structure where each point represents a voxel and its value is “0” if the point is white, “1” if the point is black, or “do not care” if the point is a cross.

3D image produced after each iteration, where $B_{m+1} = B_m \otimes \{E\}$, $B_{m+1} \subset B_m$, $B_0 = B$, $W_{m+1} \supset W_m$, and $W_0 = W$. The iteration operating on B_m is called the m -th iteration. An iteration is called an (i, j) iteration, if it starts with image (P, B_i, W_i) and ends with (P, B_j, W_j) . Basing on the fact that no additional voxel can be deleted from B_n , we call B_n the final or thinned set. Similarly, we call a black voxel in B_n a final voxel [128]. Fig. 4.14(b)-(c) show the effect of thinning B (a) with the structuring elements $\{E\}$ (Fig. 4.15). Considering the effect of a single structuring element, as shown in Fig. 4.13, we can conclude that these structuring elements are designed to peel the object B in one of the 6 major direction each, as shown in Fig. 4.16.

Palágyi and Kuba Thinning Algorithm [93]

Palágyi and Kuba propose a 6 sub-iteration 3D thinning algorithm [93]. Each thinning cycle is divided in 6 sub-iterations where different sets of structuring elements are used. The 6 sub-iterations are constructed according to the six deletion directions: Up, Down, North, South, West, and East. Considering the canonical space, right handed oriented, we call U and D the extremes (negative and positive respectively) of the y axis, W and E the extremes (negative and positive respectively) of

4 3D Morphological Operators

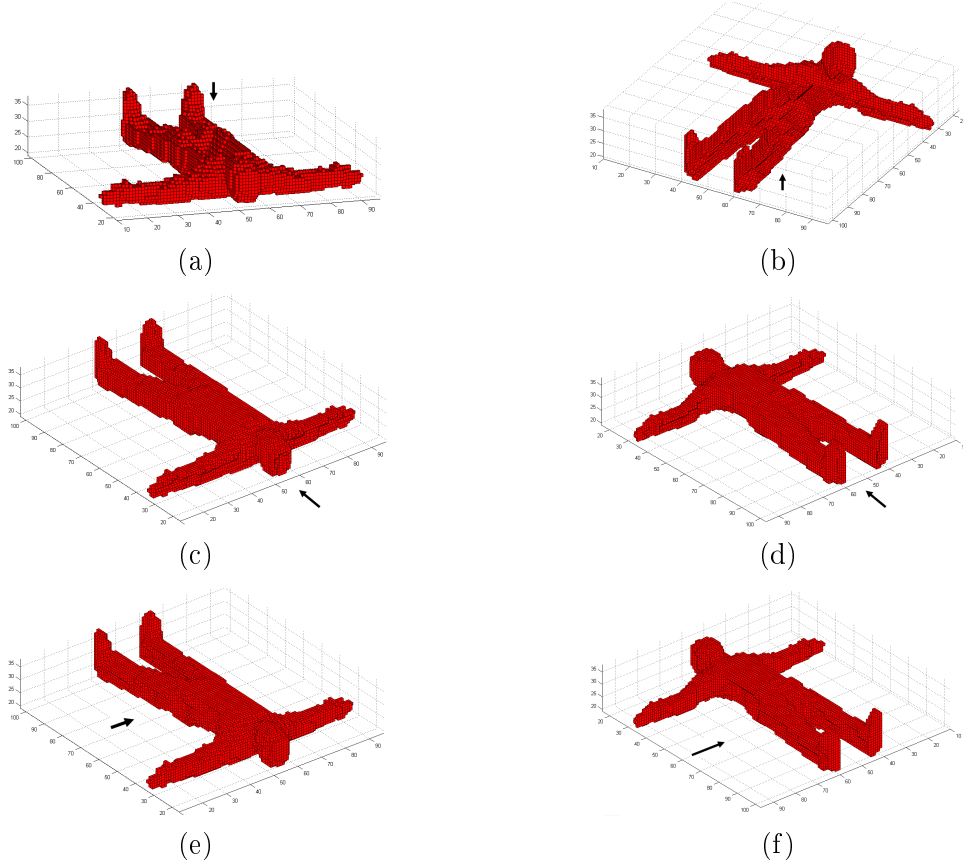


Figure 4.16: Thinning using a structuring element E designed for peeling in a specific direction. (a) Thinning of B by E^1 . (b) Thinning of B by E^2 . (c) Thinning of B by E^3 . (d) Thinning of B by E^4 . (e) Thinning of B by E^5 . (f) Thinning of B by E^6 .

the x axis, and S and N the extremes (negative and positive respectively) of the z axis (as shown in Fig. 4.2).

Let $\{E_U\}$ denote the set of structuring elements assigned to the deletion direction U . The structuring elements assigned to the direction U give the conditions to delete certain U -border points. A border voxel is an object voxel with a face-neighbor in the background. Therefore, a U -border voxel has as above voxel a white voxel, a D -border voxel has as below voxel a white voxel, and the same for other directions. The six basic structuring elements (templates) $E_U^1 - E_U^6$ assigned to the direction U are given in Fig. 4.17. They are described by four kinds of values: “1” means full, “0” means empty, “x” means “do not care” and “.” means that at least one point marked “.” is black. The template themselves are not all the structuring elements.

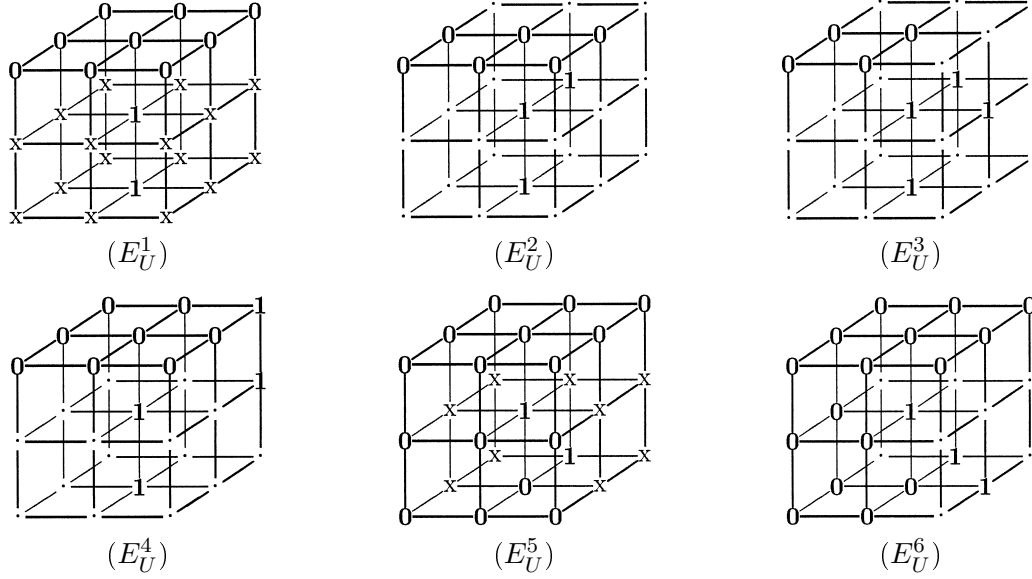


Figure 4.17: The six basic structuring elements $E_U^1 - E_U^6$ assigned to the direction U of the thinning algorithm proposed by Palágyi and Kuba [93]. Each structuring element E_U^i is represented as a lattice structure and it is described by four kinds of values: “1” means full, “0” means empty, “x” means “do not care” and “.” means that at least one point marked “.” is black.

Additionally, all the rotations of 90° , 180° and 270° around the U - D axis of the basic structuring elements $E_U^1 - E_U^6$ are structuring elements too, for a total of 21 structuring elements for each sub-iteration. Deletion conditions of the other five subiterations assigned to the directions D, N, S, W, and E, can be derived from the appropriate rotations and reflections of the structuring elements in $\{E_U\}$. In total we obtain $21 \cdot 6 = 126$ structuring elements.

Defining Y the binary array representing the thinned voxel set resulting from each sub-iteration, the 6-subiteration thinning algorithm can be summarized as follows:

X is the binary array representing the picture P;

Begin

$Y = X;$

repeat

$Y = \text{U-thinning}\{Y, \{E_U\}\};$

$Y = \text{W-thinning}\{Y, \{E_W\}\};$

$Y = \text{S-thinning}\{Y, \{E_S\}\};$

4 3D Morphological Operators

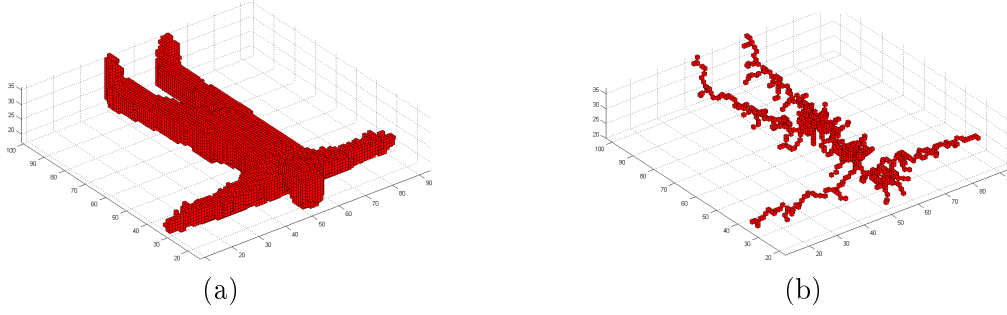


Figure 4.18: Thinning using the Palágyi and Kuba algorithm [93]. (a) Original binary 3D object B. (b) Thinning of B by $\{E\}$.

$$Y = \text{D-thinning}\{Y, \{E_D\}\};$$

$$Y = \text{E-thinning}\{Y, \{E_E\}\};$$

$$Y = \text{N-thinning}\{Y, \{E_N\}\};$$

until no points are deleted;

end.

U-thinning represents the thinning in the U direction, computed following Eq. (4.10) with structuring elements $\{E_U\}$. This means that each sub-iteration is divided into 21 sequential steps. As example, the U -thinning $\{Y, \{E_U\}\}$ is computed as follows:

$$U\text{-thinning}\{Y, \{E_U\}\} = ((\dots((Y \otimes E_U^1) \otimes E_U^2)\dots) \otimes E_U^{21}), \quad (4.11)$$

where each deleting condition is applied to the result of the previous hit-or miss application.

This thinning algorithm examines a $3 \times 3 \times 3$ neighborhood of each border voxel in each subiteration. Only the border points having the prescribed $3 \times 3 \times 3$ neighborhood can be deleted during a subiteration. Each of the 6 sub-iterations is computed sequentially. Fig. 4.18(b) shows the effect of the Palágyi and Kuba thinning algorithm [93].

From Fig. 4.19, it can be observed that the thinned set B_n is not the combination of $B_{ni} = \{B \otimes \{E\}_i\}_n$, where $\{E\}_i$ is the set of structuring elements obtained considering all the possible rotated versions of the basic structuring element E_U^i and B_{ni} is the output of the thinning operator using the set of structuring elements $\{E\}_i$. For this reason, it is difficult to figure out the output of the thinning algorithm from a given set of structuring elements $\{E\}$ and moreover it is more difficult to select the

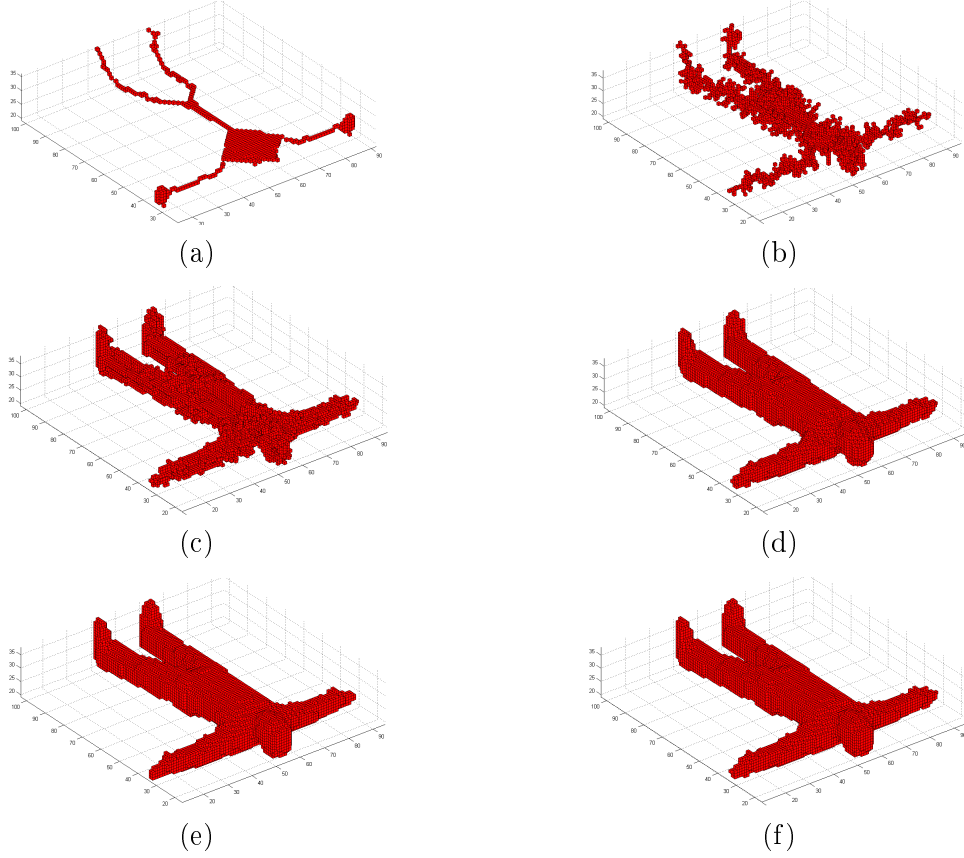


Figure 4.19: Thinning using one type of structuring elements of the Palágyi and Kuba algorithm [93]. (a) Thinning of B by $\{E\}_1$. (b) Thinning of B by $\{E\}_2$. (c) Thinning of B by $\{E\}_3$. (d) Thinning of B by $\{E\}_4$. (e) Thinning of B by $\{E\}_5$. (f) Thinning of B by $\{E\}_6$.

set of structuring elements to be used to achieve a desired thinned set. This is the main challenging task developing a thinning algorithm.

In the next Chapter we describe the proposed thinning algorithm, verifying that the output shows desirable properties and comparing it with the Palágyi and Kuba thinning algorithm [93].

5 Curve Skeleton

This Chapter is devoted to the definition of the *curve skeleton* and to the development of a novel algorithm for extracting it. A curve skeleton is a one-voxel wide representation of a 3D object. This tool is useful for many Computer Vision applications (such as virtual navigation, recognition, classification, surface approximation, shape analysis and metamorphosis, and animation) because it provides a compact and expressive characterization of the solid. Thinning is a frequently used method for extracting skeletons in discrete spaces. There are many algorithms in the literature describing extracting methodologies for different applications [28], [108]. In this Chapter we provide an overview of thinning advantages and we propose a new 3D thinning algorithm that directly extracts curve skeleton. The curve skeleton is computed directly in one phase, without pre or post processing and without any parameter setting. It is aimed to obtain a subset of the 3D object consisting only of curves, centered within the object, and topologically equivalent to it. Our algorithm provides good results, preserve topology, is easy to implement, and shows better noise robustness respect to one of the most known method [93]. In the following two Chapters 6, 7, we also give some examples of possible applications in which our algorithm is implemented successfully.

This Chapter is organized as follows. Section 5.1 provides an overview of the most used skeleton extraction methods, underlying their advantages and limitations. Section 5.2 presents the most desirable features requested for a curve skeleton representation. Section 5.3 describes the proposed thinning algorithm underlying our innovative contribution respect to the Palágyi and Kuba algorithm [93], one of the most well known curve skeleton extraction thinning algorithm (introduced in Section 4.3.4). Section 5.4 shows in detail our skeleton properties and compares it with the skeleton obtained using the Palágyi and Kuba algorithm [93].

5.1 Skeletonization

The process that consists in peeling iteratively a black object on a white background while preserving its topological properties has given rise to intensive international research activities for the last fifty years with hundreds of published papers [73]. The importance of such an investigation is due to the interest in topological *skeletons*, as a fundamental pre-processing step in pattern recognition and image analysis. Its duration is due to the fact that no definitely satisfying solution has been found. In recent years several innovative applications, that use skeleton representations, (*e.g.*, tracking, identification [64], and human computer interactions [122]) have been proposed in the literature, due also to the development of devices, such as IR-based sensors, that permit a real-time acquisition of a 3D scene with different accuracies and resolutions.

Skeletons are region based shape descriptors which summarize the general form of an object/shape B [30]. A skeleton corresponds to the points locally symmetric to a shape, in the sense of being a “tick” or “central” structure. More formally, the skeleton is defined as the set of centers of all the maximal balls included in B . The process of obtaining a skeleton is called *skeletonization*.

In 2D it is a set of curves, while in the 3D case the corresponding object is called *surface skeleton* because it contains non only curves, but also surface patches [23] (as shown in Fig. 5.1(c)). The surface skeleton consists of the union of surfaces and curves, symmetrically placed within the object and characterized by the same object topology. For this reason, we refer to the surface skeleton also as *medial surface*. In many applications (*e.g.*, animation and virtual navigation), however, a concise representation of 3D objects, with curve arcs or straight lines, is desirable. This line-like representation of a 3D object is called *curve skeleton* and it is a simplified 1D representation of the original 3D object, consisting only of curves [30], [4] (as shown in Fig. 5.1(b)).

Both theoretical and applicative issues are present in the vast literature concerning skeletonization [28], [108]. A common approach to skeletonization is based on an iterated topology preserving removal. Thinning is not the only method for extracting 3D skeletons from objects. A commonly used classification scheme proposed in the literature divides the skeletonization algorithms into the following classes [28]: *topological thinning* (grassfire propagation), *distance transform*-based (ridge detection), *geometric* (Voronoi diagram-based) and *general field function*-based. However, many

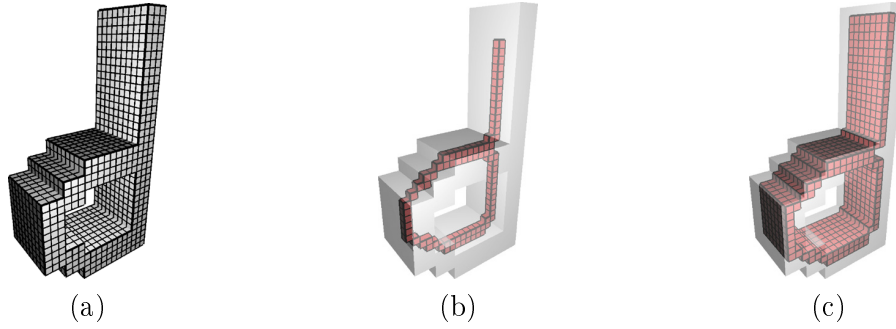


Figure 5.1: (a) Original object, (b) curve skeleton, and (c) surface skeleton obtained with the algorithm proposed in [69].

of the surveyed methods that produce curve skeletons use pieces from several classes listed above to obtain a curve skeleton. For example, there are thinning algorithms that use the distance field information to determine the thinning order, or some distance field methods which use thinning to prune the skeleton [23].

Thinning is an iterative object reduction technique for producing a reasonable approximation of the skeleton in a topology preserving way [68]. During the 3D thinning process, border points of a binary object, that satisfy certain topological and geometric constraints, are deleted in an iterative procedure. Border detection and removal are iterated and the process terminates when the object entirely consists of border elements and these are all necessary to topology preservation or to account for shape properties. There are several subclasses of thinning methods based on how detachable points are detected and considered for removal. *Sequential* algorithms can only delete one point in each iteration, while *parallel* algorithms delete all the points satisfying a condition simultaneously.

Distance Field methods attempt to find candidate voxels, for the curve skeleton points, that are locally centered within the object. These voxels correspond to the ridges of a distance field. Various distance functions can be used such as the Euclidean distance or its approximations [14]. Most of the distance field-based algorithms implement three different steps [25], [43], [110]: to find candidate voxels (*i.e.* to find ridge points, local maxima, saddles), to prune down the set of candidate voxels (that is usually a fairly large set) to a manageable size, and to reconnect the remaining voxels (that are usually disconnected after the pruning step) in order to produce a set of 1D curves.

Geometric methods are applied to objects represented by polygonal meshes or

scattered point sets. A popular approach is to use the Voronoi diagram generated by the vertices of the 3D polygonal representation or directly by a set of unorganized points [2], [6], [86]. The internal edges and faces of the Voronoi diagram can be used to extract an approximation of the surface skeleton of the shape. A curve skeleton can be extracted from this medial surface approximation by pruning it to a 1D structure. Different algorithms based on this approach use different methods for selecting the desired subset of the Voronoi diagram.

There are other geometric methods that produce a 1D skeleton directly. Reeb graph-based shape descriptors are 1D structures encoding the topology and geometry of the original shape. A Reeb graph is not a curve skeleton. It is not even defined in the same space of the original object. However, an embedding of the Reeb graph into 3D space can be attempted by mapping each edge into a sequence of 3D points defined, for example, as the centers of the successive level-set contours associated with the respective edge. This defines a curve skeleton for the original object [10], [17], [127].

In **General-Field Functions**-based methods, the curve skeleton is extracted by detecting the local extremes of the considered field and connecting them. A force following algorithm can be used for connectivity [24]. Another possibility is the use of an active contour to detect the final location of the curves connecting the extremes in the field [71], [125]. Various types of fields can be used to extract curve skeletons: generalized potential field functions [24], where the potential at a point interior to the object is determined as a sum of potentials generated by the boundary of the object, are included in this class.

The main disadvantage of the geometric methods is their sensitivity to noise. Furthermore, these methods are more computationally intensive than the thinning/distance field-based methods and produce a surface skeleton, not a curve skeleton. Nevertheless, reducing a surface skeleton to a curve skeleton is possible using the curve thinning algorithms. The main advantage of general-field functions over the distance field is that they can produce nice curves on medial sheets where the distance field is constant. This is because they take into account larger boundary areas, not just the distance to the closest point on the boundary. This also creates an averaging effect that makes these algorithms less sensitive to boundary noise. However, they are much more expensive to compute. Resolution of the voxel grid also affects the field functions, which tend to be more sensitive to noise in thin regions of the object, because significant contributions towards the final field value at an interior point come

from fewer boundary voxels. Another disadvantage of the general field functions is their numerical instability, since the computations usually involve first or even second order derivatives. The distance field-based methods can accurately extract the surface skeleton; however, they cannot extract a curve skeleton from arbitrary objects without employing additional techniques to prune the medial surface. The main advantage of these methods is that the computation of the distance field is very fast and it is usually needed by the application afterward. The thinning process is usually very fast and does not depend on the resolution of the surface sampling. Moreover it can produce directly the curve skeleton without implementing any pruning post-processing and the skeleton is topology equivalent to the object without implementing any linking algorithm. The main disadvantage is its sensitivity to noise.

In spite of its simplicity, there is no rigorous definition of a curve skeleton. We do not attempt to give a precise definition of the curve skeleton. Our goals are to provide an analysis of different aspects of this structure and to propose a new method to extract one-voxel wide skeleton that tries to satisfy most of the desirable properties, first of all that preserves topological information. For the purpose of this work, the curve skeleton is defined as a simplified 1D representation of a 3D object without losing relevant information. In the following, the topological curve skeleton is referred simply as curve skeleton and the output of the thinning process as thinned set.

5.2 Desirable Properties

In this Section, we describe a set of desirable curve skeleton properties, compiled from analyzing the literature on the subject [23] and we discuss the aforementioned curve skeletonization methods in terms of the properties presented.

For the following discussion, we use $Sk(B)$ to denote the curve skeleton of a 3D object B and we discuss about the following curve skeleton properties: homotopy, connectivity, invariance under isometric transformations, reconstruction, thinness, centeredness, robustness, and efficiency.

5.2.1 Homotopy (topology preservation)

The curve skeleton should be topologically equivalent to the original object. Two objects have the same topology if they have the same number of connected components, tunnels, and cavities. The above formulation applied to an object B_2 derived from an object B_1 : “object B_2 preserves the topology of object B_1 is meaningful only if an additional constraint is added: object B_2 is obtained from B_1 by only removing object voxels (no adding)” [59]. Otherwise, object B_2 could end up having a completely new configuration, but still have the same topology. For example, by adding object voxels to B_2 , B_2 may grow limbs where B_1 do not have them, but still have the same number of connected components, tunnels, and cavities. With this observation, the above definition of topology preserving is meaningful in the context of skeletonization, where the skeleton $Sk(B)$ is a subset of the original object B . Of course, we cannot have cavities in a 1D curve, so, in a strict sense, a curve skeleton cannot preserve the topology of an object with cavities. Fortunately, the curve skeleton definition and extraction for objects with cavities are not issues that concern our database of reconstructed poses.

Homotopy is ensured by the thinning methods because only voxels that do not change the object topology (the simple points) are removed. Since distance field and geometric methods do not produce a curve skeleton directly, topology preservation depends on the subsequent pruning and connectivity steps. The Reeb graph is an accurate representation of the object topology. Curve skeletons produced by algorithms employing generalized fields could not correspond to the topology of the original object. Due to numerical errors and to the resolution of the voxel grid, potential field algorithms can create loops in the curve skeleton that do not correspond to a tunnel or hole in the original object [22].

5.2.2 Connectivity

The output skeleton should have the same connectivity as the original object, which means that any object in the input image cannot be split (or disconnected) or completely deleted; any cavity within the object cannot be merged with background or another cavity; creation of new cavities should be avoided; the skeletons should not contain any background element. This is a consequence of homotopy. If the curve skeleton corresponds to a single connected object, then, by maintaining the topology of this object, the curve skeleton would have to consist of a single connected

component itself.

Connectivity is usually checked by all the algorithms. Some algorithms (*e.g.*, thinning, level-set based geometric methods) explicitly maintain connectivity during computation, while other methods check and enforce connectivity in a postprocessing step.

5.2.3 Invariance under Isometric Transformations

Given an isometric transformation T (a transformation in which the distances between points are preserved), the curve skeleton of the transformed object $T(B)$, denoted by $Sk(T(B))$, should be the same as the transformed curve skeleton of the original object. Formally, the invariance criterion is given by: $T(Sk(B)) = Sk(T(B))$. This property is important for matching applications where the curve skeleton is used as a shape descriptor. In such applications, it is common to have similar objects in different orientations, that nevertheless still need to be matched. For this reason, the shape descriptor must be insensitive to object orientation.

Directional thinning methods are sensitive to the object orientation. The final result (end points, number of branches, and their location) depends on the order in which the different directions are processed. Distance field, Voronoi-based, and general field methods do not depend on object orientation. Reeb graph-based methods can be sensitive to object orientation depending on the function chosen to extract the level sets.

In all the cases involving discrete representations of objects, the finite resolution of the voxel grid produces small errors when objects are transformed. As a result, even though curve skeletonization algorithms themselves are not sensitive to object orientation, the input data itself is already adversely affected by the transformation. Such small discretization errors show up on the boundary of the transformed object and their effect on the resulting curve skeleton is similar to that of surface noise. A relaxed criterion can be imposed: skeletonization and object rotation must commute, at least for rotation angles multiple of $\pi/2$ [73].

5.2.4 Reconstruction

Reconstruction refers to the ability to recover the original object from the curve skeleton. Given the definition of the surface skeleton as the set of centers of maximal inscribed balls, an obvious choice as reconstruction method is to compute the union of

maximal inscribed balls centered at each curve skeleton point (ball growing approach) [3]. The radius of each ball is given by the distance to the closest point on the boundary of the object. If we denote the reconstruction operation by $Rec(Sk(B))$, then accurate reconstruction means that $Rec(Sk(B)) = B$.

Regardless of the method used to compute it, a complete and accurate reconstruction of the original object is not possible from the information retained in a curve skeleton alone when using the simple ball growing approach, since it is only a subset of the surface skeleton. That is, in general, $Rec(Sk(B)) \neq B$. The difference volume $B - Rec(Sk(B))$ will provide a quantifiable measure of the ability to reconstruct the object. If the degree of reconstruction is very low, it will mean that the curve skeleton does not capture much of the original object, in terms of volumetric occupancy. Reconstruction can be improved by storing more information in each curve skeleton point. For instance, one could store the three radii of a maximal inscribed ellipsoid and replace ball-growing with an ellipsoid-growing algorithm. Alternatively, one could simply extract a curve skeleton with many more branches that should reconstruct more of the original object using the classic ball-growing approach.

Reconstruction using the ball-growing approach [35] needs distance field information in order to determine the radius of the ball that will be grown from each curve skeleton point. In this respect, the distance field-based methods have an advantage over the other methods, because this information is already available.

5.2.5 Thinness (1D)

Curve skeletons should be one-dimensional, that is, at most one voxel thick in all the directions, except at joints where the curve skeleton might become thicker to ensure connectivity between the different branches. We can distinguish three types of curve skeleton points [12]: *regular points* on a 1D curve that have exactly two neighbors, *end-points* of a curve that have exactly one neighbor, and *junction points* (where curves meet) which can have three or more neighbors.

Thinning algorithms can either directly produce a curve skeleton or further thin a surface skeleton to a 1D representation. Parallel thinning algorithms, which remove all the simple points at once, may not be able to achieve 1D skeletons due to homotopy constraints. Directional thinning methods do not have this disadvantage. Distance field methods and Voronoi-based geometric methods do not produce a 1D skeleton directly. Both require significant postprocessing to reduce the can-

didate voxels (distance field) or the medial surface (Voronoi-based methods) to a curve skeleton. The Reeb graph-based methods directly produce 1D straight-line skeleton segments, although the representation is no longer voxel-based. Thinness is an implicit property of the general-field methods that use force-following or active contours to generate 1D skeleton branches.

5.2.6 Centeredness

An important characteristic of a curve skeleton is its centeredness within the object. To achieve perfect centeredness, it is required for the curve skeleton to lie on the medial surface, since the medial surface is centered within the object. This means that the skeleton must contain the centers of maximal balls. This criterion alone is not enough and, in addition, we require the curves to be centered within the medial surface patches they belong to [30].

Given the well known sensitivity of the medial surface to small perturbations on the boundary of the object [5], constraining the curve skeleton to lie on the medial surface may make it sensitive to such changes as well. Instead of exact centeredness, an approximate centeredness is probably enough for many applications, such as virtual navigation or animation. For example, in a virtual colonoscopy application, we still want the navigation path to be close to the center of the object, but being one or two voxels away from the exact center is not a big concern. Moreover, resolution affects any centeredness measurement in the discrete domain.

Thinning and general field methods do not guarantee centeredness. In the case of directional thinning, this would depend on the order in which the different directions are applied. In the case of general field methods, since they are taking into account a larger surface area than the two closest points, centeredness is usually compromised. Methods using a distance field can better achieve centeredness because of the centeredness information included in the distance field. Geometric methods, directly computing points where the maximal inscribed spheres touch the surface [63], can also achieve centeredness. They are dependent on the sampling density of the object's surface: a denser sampling produces a more centered curve skeleton [2], but the running time increases. Centeredness of level-set based [56] curve skeletons is poor, because centroids are directly connected with straight-line segments, regardless of the configuration of the object between these points.

5.2.7 Robustness

Medial surface is very sensitive to small changes in the boundary. A desirable property of the curve skeleton is to exhibit weak sensitivity to noise on the boundary of the object, that is, the curve skeleton of a noise-free object and the curve skeleton of the same object with noise should be similar. A robust curve skeleton cannot be perfectly centered. Exact centeredness would constrain the curve skeleton to the medial surface, which is extremely sensitive to boundary perturbations.

Thinning, distance field, and Voronoi-based geometric methods are sensitive to noise, generating unnecessary branches in the skeleton as a result. Several methods have been proposed to filter the resulting skeletons [6], [25]. General field approaches are less susceptible to noise because of the large amount of averaging included in the underlying computation. These methods are more sensitive to resolution.

5.2.8 Efficiency

The skeletonization algorithm should be efficient: many applications need real-time computations.

Thinning is practically a linear process in the number of object voxels. Most of the voxels of the input object are removed when they are first processed (they are simple points). The non-simple points are processed again at every subsequent thinning step until the algorithm terminates. An exact complexity analysis of such algorithms is difficult, since they are data dependent. The Euclidean distance field of a 3D object can be computed in linear time using the algorithm of Saito and Toriwaki [99]. The subsequent steps of filtering and reconnecting the curve skeleton may, however, have a higher complexity (but, they usually operate on a greatly reduced set of voxels). Computation of the Voronoi diagram of a set of n points in 3D is $O(n^2)$ in the worst case, although, in practice, it is almost linear [2]. As in the distance field case, additional processing is necessary to get a curve skeleton, but the number of input elements is reduced. The computational complexity of the level-set-based geometric methods depends on the kind of function computed over the object. The following steps, detecting connected components in each level set and computing their centroids, are linear in n . The complexity of the potential field computation is $O(n^2)$ [24], where n is the number of object voxels. These methods are computationally more intensive than distance field methods, because they take into consideration larger boundary areas, not just the closest point.

5.2.9 Discussion

Not all the properties described above are essential to all the types of applications. Furthermore, it is well-known that these requirements cannot be satisfied simultaneously as some of them are contradictory. In particular, noise immunity and one-pixel thickness are incompatible with centeredness. Also thinness and reconstruction are conflicting. Even for objects whose medial surface actually contains only curves (like tubular objects), a one-voxel thick curve skeleton may not contain all the necessary maximal balls to accurately reconstruct the object. As a result, various algorithms that extract curve skeletons usually satisfy only a subset of these properties, depending on the application.

For the distance field and the Voronoi diagram-based geometric methods, a curve skeleton is more difficult to obtain directly. The first step of these methods generates a structure closer to the medial surface, so additional processing is required to obtain a 1D curve skeleton. The distance field-based algorithms can obtain an accurate skeleton, with a large number of branches. The obtained skeleton reflects the essential structure of objects, and it is invariant under translation, rotation, and affine transformations, but it cannot maintain the connectivity. Moreover the centroids of the clusters are connected using straight lines, so the centeredness property is not always satisfied. In fact, some curve skeleton segments can go outside the object, even if the centroids are well centered because of the centeredness information included in the distance field. Also in the Reeb graph-based geometric algorithm some segments are outside the object. This problem is difficult to fix regardless of the function used to define the level sets. Reeb graph based methods can get accurate representation of the object topology. They are robust to surface noise, mesh sampling and model pose variations. They can be sensitive to object orientation depending on the function chosen to extract the level sets. They can explicitly maintain connectivity during computation. The potential field methods yield the cleanest and smoothest curve skeleton at the initial stage. This is due to the “global averaging” effect of the potential field. However, for objects with thin, flat regions, the algorithm identifies too many critical points, creating a large number of curve skeleton branches in that area, very similar to a medial surface. This is because the algorithm is resolution dependent and this is a disadvantage. The thinning algorithm performs well in all the cases, but the resulting curve skeletons are not smooth since all the surface irregularities propagate to the curve skeleton through the peeling process. This also

makes it more sensitive to noise. Thinning methods can preserve the topology of objects well, because only voxels that do not change the topology are removed, and this is their most excellence property. Another advantage is that there are no parameters for the thinning implementation. Moreover, the 3D thinned curve skeleton of a binary object is fast to compute and, in theory, no postprocessing steps are required.

For all these reasons, we focus on thinning algorithms theory and properties. Some important notions and a new topology preserving 3D thinning algorithm to extract one-voxel wide skeleton preserving topological information are illustrated in the following Section 5.3. After that, in Section 5.4, we show that our procedure is easy to implement, fast, robust to the noise on the object boundary and allows to recover a 3D object that is close to the original object and that reflects the salient aspects of the original shape (overcoming classical limits of thinning methods).

5.3 3D Thinning Algorithm to extract Curve Skeleton

5.3.1 Important Notions

We focus on the development of a 3D thinning algorithm to extract the curve skeleton of a 3D binary object. Ideally, 3D skeleton is 1-voxel wide line representation that both contains shape features and retains topological structure of the original object. Thinning is an iterative layer by layer peeling technique that deletes, at each iteration step, only object voxels that satisfy certain geometrically and topological constraints (defined by the structuring elements).

The first and most important requirement, for a thinning algorithm, is to design structuring elements that allow to preserve topology. A reduction operator does not preserve topology [92] if:

- any object in the input picture is split (into two or more parts) or it is completely deleted,
- any cavity in the input picture is merged with the background or another cavity,
- a cavity is created where there is no cavity in the input picture,
- any hole in the input picture is merged with the background or with another hole,
- a hole is created where there is no hole in the input picture.

In particular, the thinning algorithms rely on the concept of simple point. A *simple point*, a concept introduced by Morgenthaler in 1981 [91], is an object point (voxel) that can be removed without changing the object topology [59]. Most thinning algorithms are designed and proven correct for a specific (6, 8, or 26) connectivity. We use the following result for (26,6) pictures:

Theorem 1 [72], [92], [97]. *Black point p is simple in picture $(\mathbb{Z}^3, 26, 6, B)$ if and only if all of the following conditions hold:*

1. *the set $(B \setminus \{\mathbf{p}\}) \cap N_{26}(\mathbf{p})$ contains exactly one 26-component;*
2. *the set $(\mathbb{Z}^3 \setminus B) \cap N_6(\mathbf{p})$ is not empty and it is 6-connected in the set $(\mathbb{Z}^3 \setminus B) \cap N_{18}(\mathbf{p})$.*

Theorem 1 shows that the simplicity in (26,6) pictures is a local property; it can be decided in the view of the $3 \times 3 \times 3$ neighborhood of a given point. The local characterization of simple points makes thinning algorithms much more efficient [65]. The *support* of a reduction operator, applied to a black point $\mathbf{p}$, is the minimal set of points whose values determine whether $\mathbf{p}$ is deleted by the operator [39]. Since we focus on (26,6) images, the support of our thinning operator is $3 \times 3 \times 3$. It means that the local neighborhood of the implemented structuring elements is $3 \times 3 \times 3$.

Identifying simple points is not enough for a thinning algorithm: removing all the simple points from the object produces excessive shortening of the curve skeleton branches. This is because all the *end-points* of curve skeleton curves are simple points themselves (*i.e.*, removing them will not change the topology of the object). Additional conditions are used to prevent removal of curve end-points in order to preserve important geometrical information relative to the shape of the objects. For the 3D case, we can define two different kinds of end points: curve-end points and surface-end points. A thinning process which preserves curve-end points (resp. surface-end points) is called a *curve-thinning* algorithm (resp. a *surface-thinning* algorithm). The result obtained by a curve thinning algorithm (resp. a surface thinning algorithm) is a topological curve skeleton (resp. a surface skeleton). Therefore, we focus on the design of a 3D thinning algorithm that preserves only curve-end points.

5.3.2 Proposed 3D Thinning Algorithm

We implement a 6-subiteration 3D thinning algorithm. The 6 subiterations are constructed according to the six major directions: Up, Down, North, South, West, and East. Each subiteration is based on a set of structuring elements based on

a Boolean characterization of simple points which are not curve-end points. The deletion conditions are defined on a $3 \times 3 \times 3$ local neighborhood.

The idea is close to the method proposed by Palágyi and Kuba [93], described in details in Section 4.3.4. This choice is motivated considering that most of the existing 3D thinning algorithms are difficult to implement due either to complex checking conditions or to lengthy matching templates involved [128]. On the other hand, the Palágyi and Kuba algorithm still remains one of the most quoted and used thinning algorithm to extract curve skeleton [23]. Moreover this is designed on the six major directions that, taking into account the difficulty to figure out the output given a set of structuring elements, can help to intuitively solve this challenging task. However, as can be noted from Fig. 4.18(b), the thinned set obtained with their method presents several secondary branches that are not desirable in a curve skeleton.

The proposed thinning algorithm, instead, is designed to satisfy all the aforementioned eight requirements: connectivity and topological preservation *in primis* (this is one of the reasons why we focalize on thinning algorithms), but also thinness, robustness and centeredness, that are the main classical drawbacks of the curve skeleton obtained with thinning, as the one obtainable with Palágyi and Kuba is. The reconstruction property is particularly important for the developed applications and, even if it is briefly discussed in this Chapter, it is better described in Chapter 6.

Our main contribution, respect to the Palágyi and Kuba implementation, is the definition of new and different structuring elements, in order to obtain a more desirable curve skeleton: more robust to noise on the boundary and with less peripheral branches without applying any post-pruning (another strength of the proposed algorithm respect to the state of the art of the extracting skeleton methodologies, not only thinning).

We design four basic structuring elements $E_U^1 - E_U^4$ for the direction U, shown in Fig. 5.2. As for the structuring elements designed by Palágyi and Kuba [93], our structuring elements are described by four kinds of values: “1” means full, “0” means empty, “x” means “do not care” and “.” in E^1 means that at least one point marked “.” is black. All the rotations of 90° , 180° and 270° around the U - D axis of the base elements $E_U^2 - E_U^4$ are structuring elements too, for a total of 13 structuring elements for each sub-iteration. Considering also the other five directions D, N, S, W, and E, the set of structuring elements $\{E\}$ is, in total, composed by $13 \cdot 6 = 78$

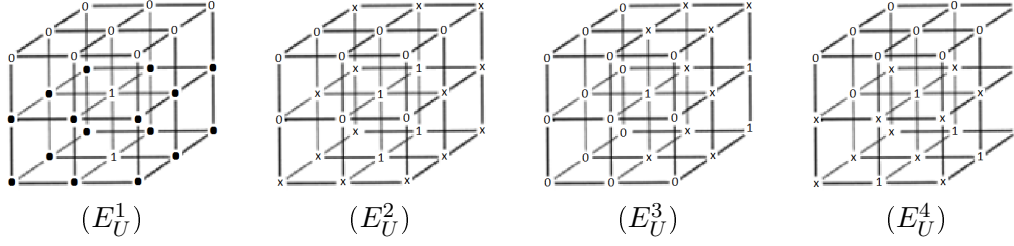


Figure 5.2: The four basic structuring elements $E_U^1 - E_U^4$ assigned to the direction U of the proposed thinning algorithm. Each structuring element E_U^i is represented as a lattice structure and it is described by four kinds of values: “1” means full, “0” means empty, “x” means “do not care” and “.” means that at least one point marked “.” is black.

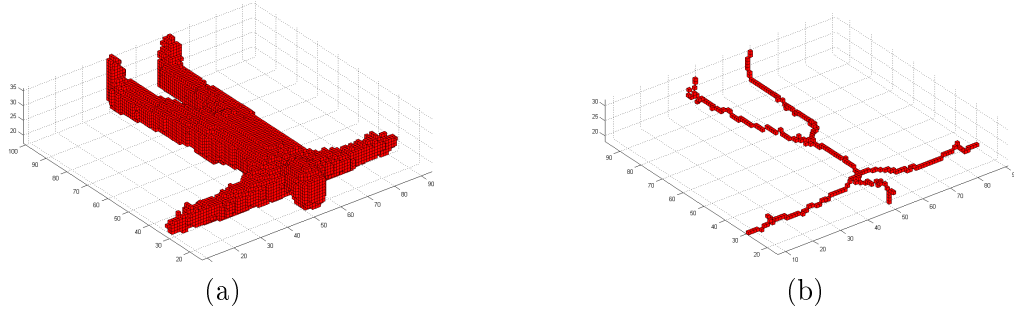


Figure 5.3: Thinning using the proposed thinning algorithm. (a) Original binary 3D object B. (b) Thinning of B by $\{E\}$.

elements.

The effect of the proposed thinning algorithm, on the object used in Chapter 4, is shown in Fig. 5.3(b). It can be seen that the extracted curve skeleton presents more less branches than that obtained with the Palágyi and Kuba algorithm (Fig. 4.18(b)). Fig. 5.4 summarizes also the effect of each single structuring element. Even if the thinned set B_n is not the combination of the partial thinned sets, the contribution of each single structuring element $\{E\}_i$ can be intuitively understood.

The main strength of the developed algorithm for curve skeleton extraction is the ability to extract a compact representation of the object without imposing any a-priori knowledge of the object topology, without requiring any parameter setting, and without needing any postprocessing pruning or linking procedure. Examples of thinned sets obtained applying the proposed algorithm, to some of the reconstructed frames for each action class of our database, are shown in Figs 5.5-5.14.

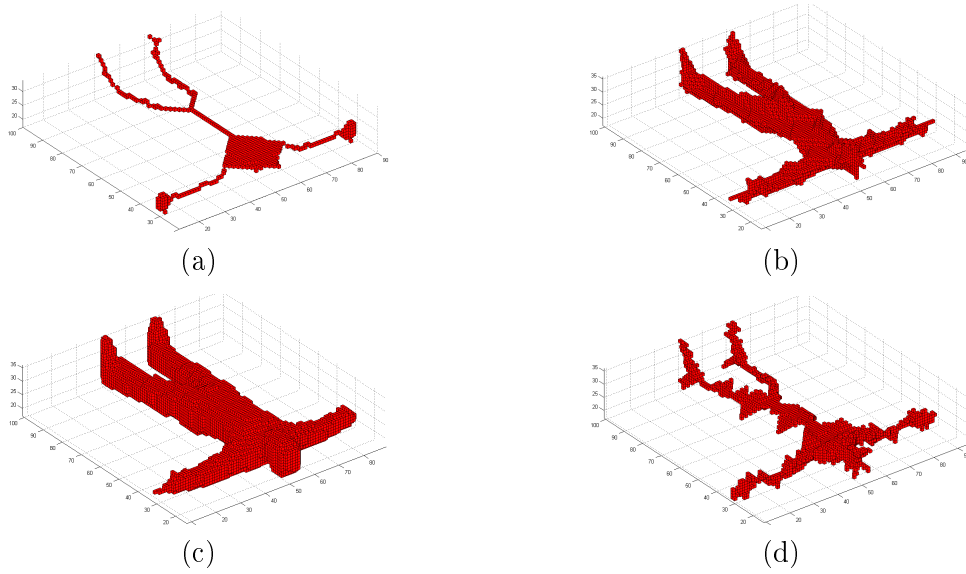


Figure 5.4: Thinning using one type of structuring elements of the proposed thinning algorithm. (a) Thinning of B by $\{E\}_1$. (b) Thinning of B by $\{E\}_2$. (c) Thinning of B by $\{E\}_3$. (d) Thinning of B by $\{E\}_4$.

5.4 Results

We evaluate the performance of the proposed algorithm (in terms of the eight requirements illustrated in the previous Section) comparing the results of our algorithm respect to the curve skeleton obtained using the Palágyi and Kuba algorithm [93] for our database of human 3D shape reconstructed from the multi-Kinect system implemented in our laboratory. We choose the Palágyi and Kuba algorithm because it is one of the most quoted thinning algorithms for skeleton extraction [23]. Moreover they design structuring elements for thinning iteratively in the six major directions. Even if some variations are presented during years (12 subiterations for example [94]), considering the difficulty to design structuring elements for a 3D thinning algorithm, we prefer to peel the objects following the major directions. We focalize our attention on thinning algorithms because we look for a method to extract a compact and topology preserving representation of the object in the scene without needing any a priori information on the object to represent, any parameter setting, any sampling requirement, or any pruning or linking post/pre-processing.

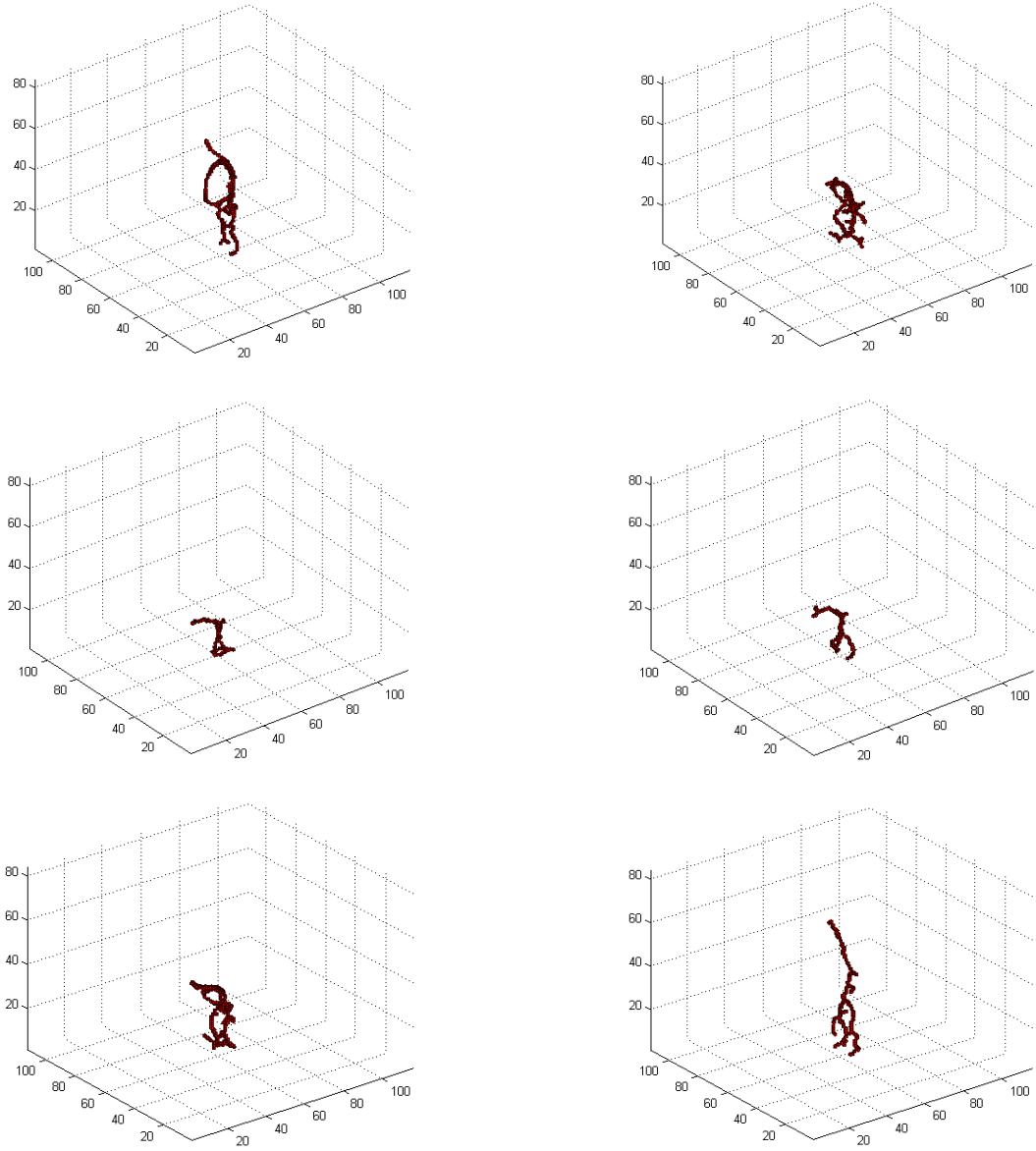


Figure 5.5: Thinning using the proposed algorithm on six reconstructed frames from the action: **Crouch**.

5 Curve Skeleton

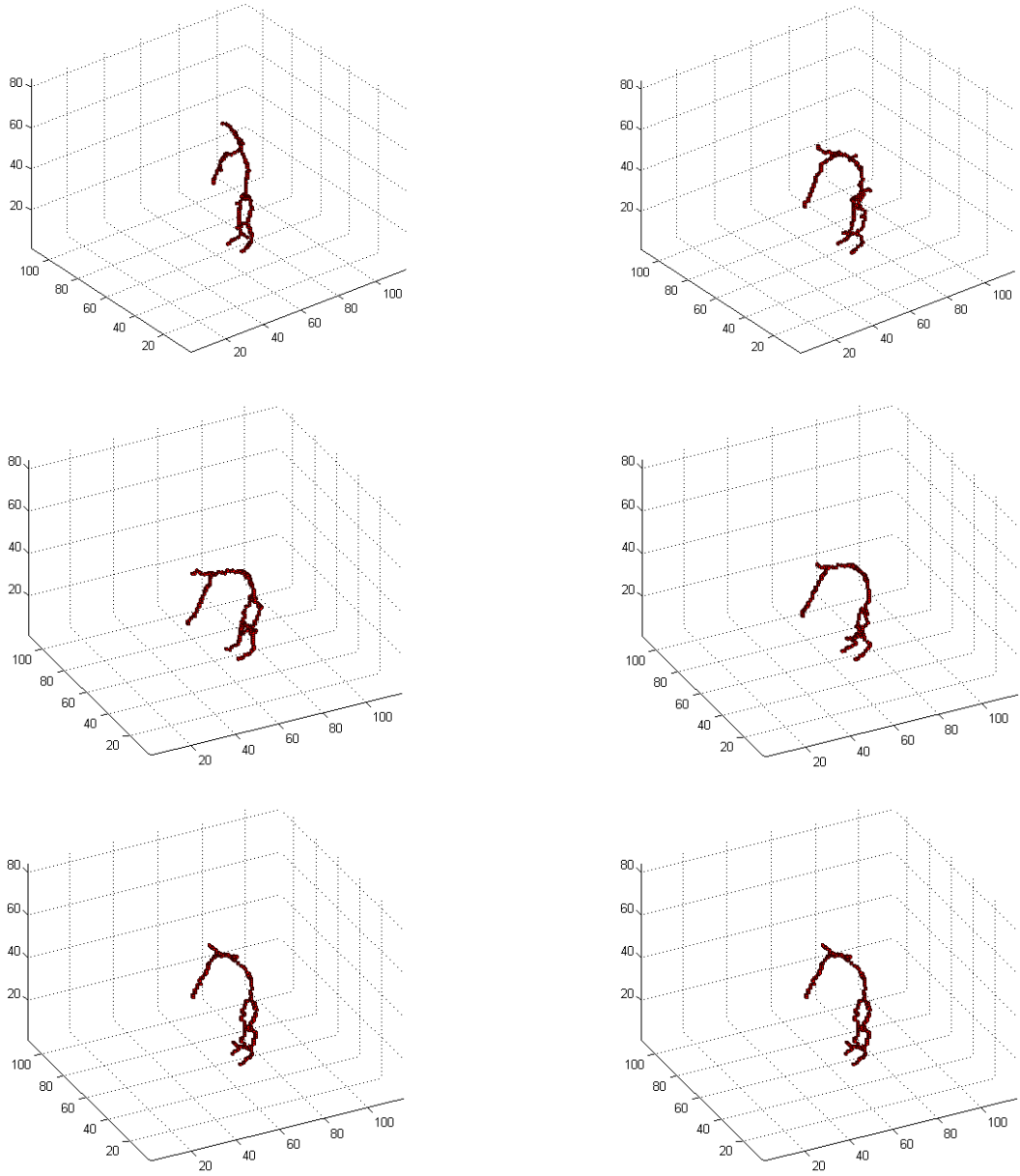


Figure 5.6: Thinning using the proposed algorithm on six reconstructed frames from the action: **Grasp**.

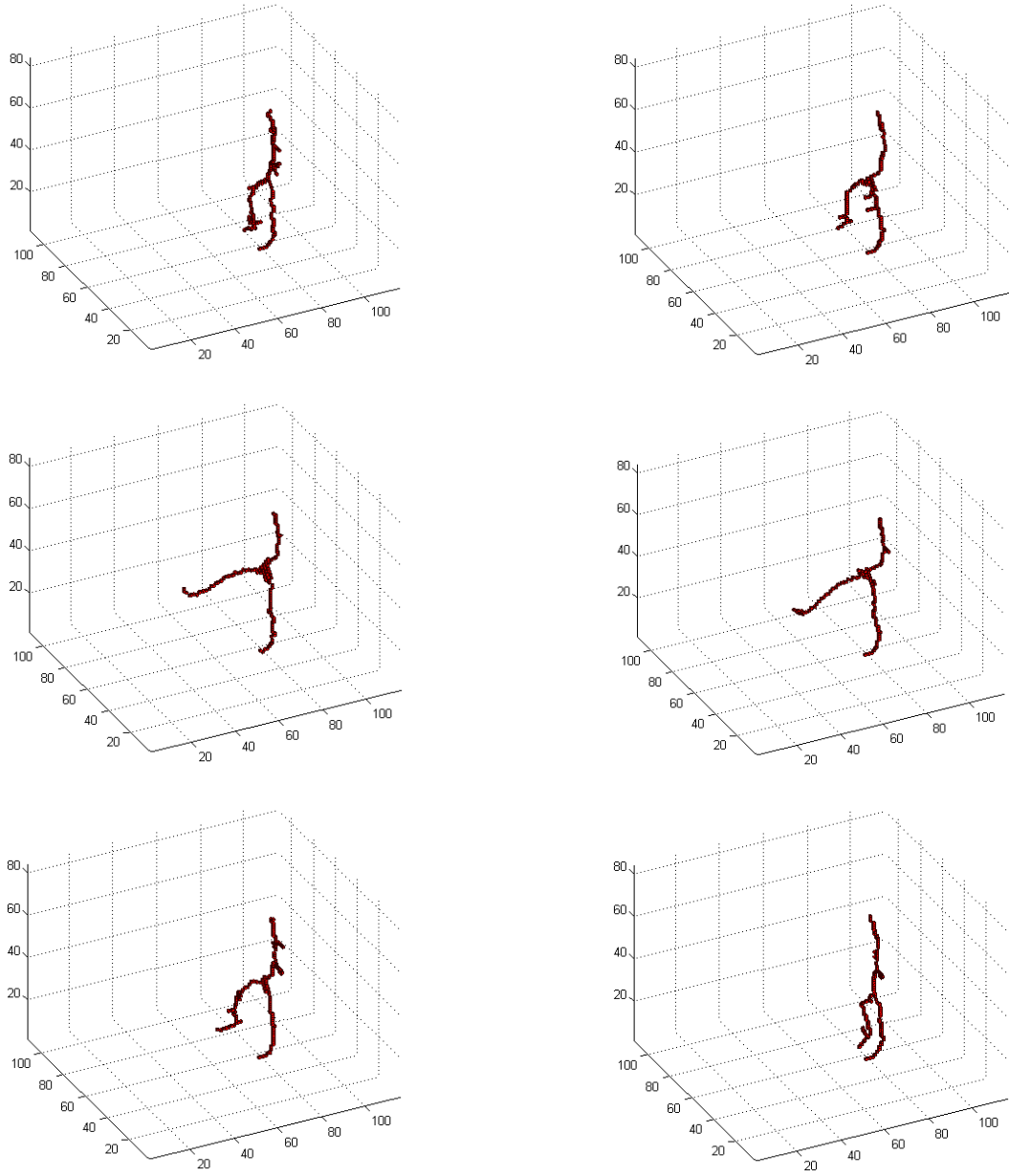


Figure 5.7: Thinning using the proposed algorithm on six reconstructed frames from the action: Kick.

5 Curve Skeleton

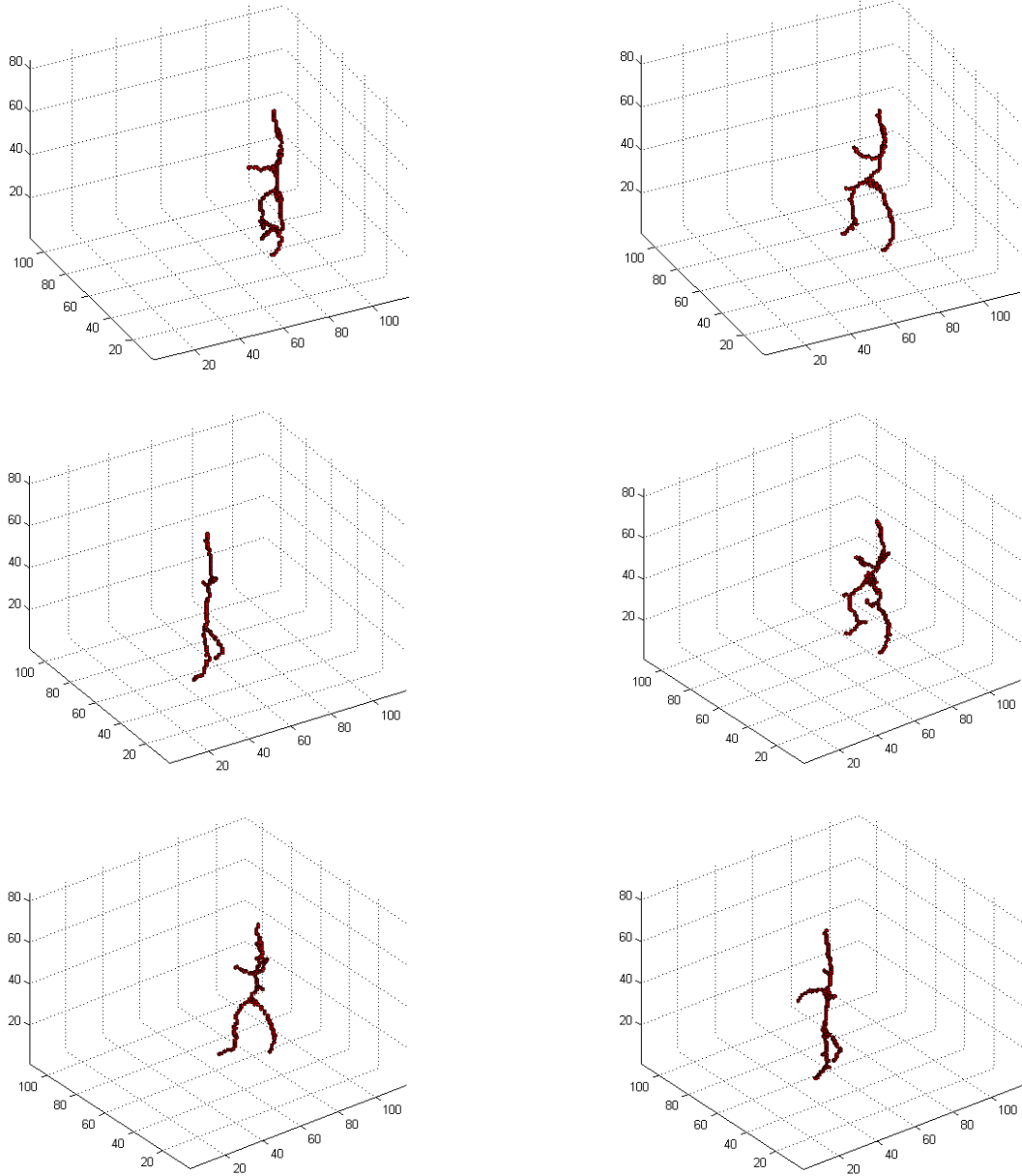


Figure 5.8: Thinning using the proposed algorithm on six reconstructed frames from the action: **March**.

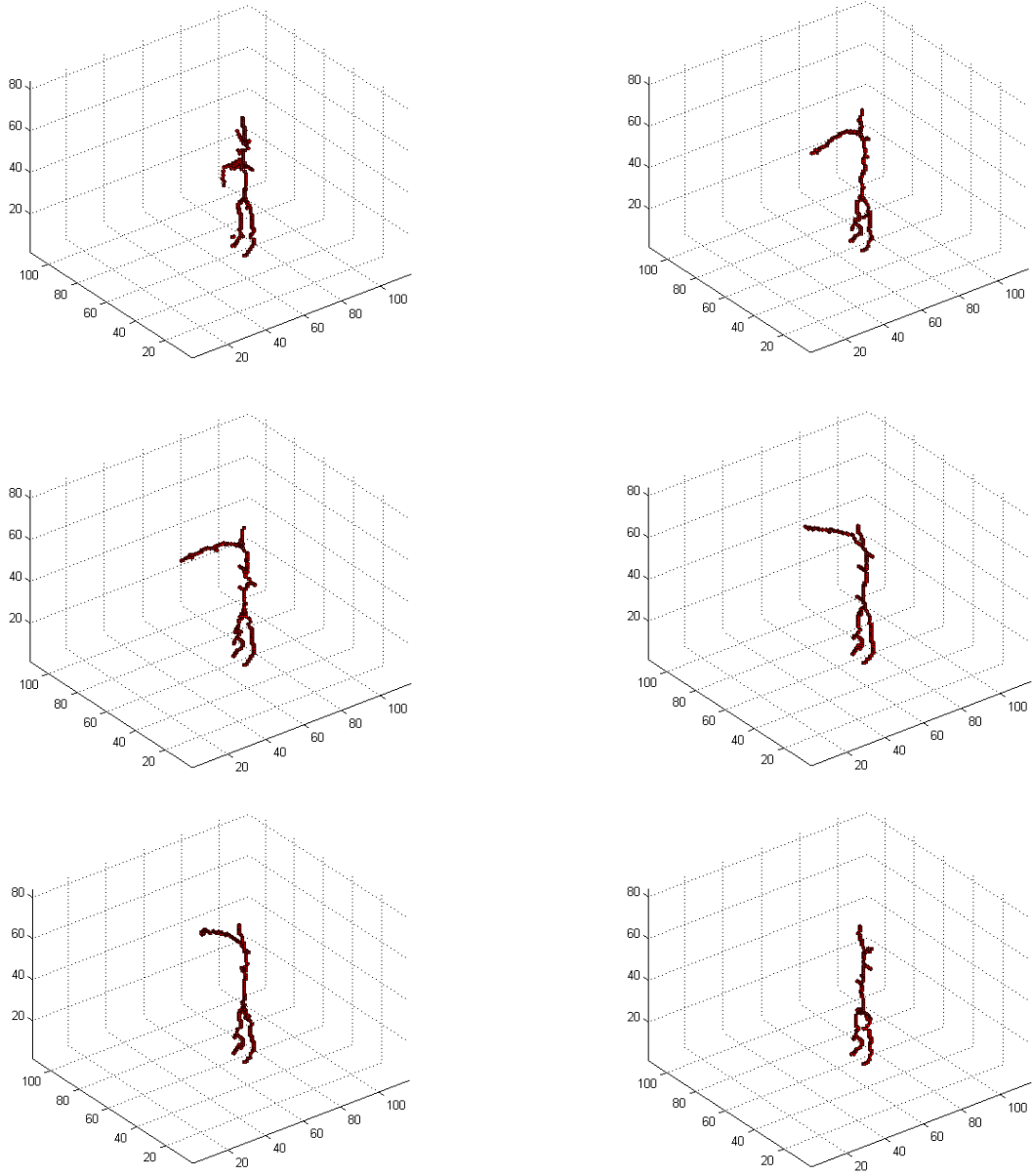


Figure 5.9: Thinning using the proposed algorithm on six reconstructed frames from the action: Move.

5 Curve Skeleton

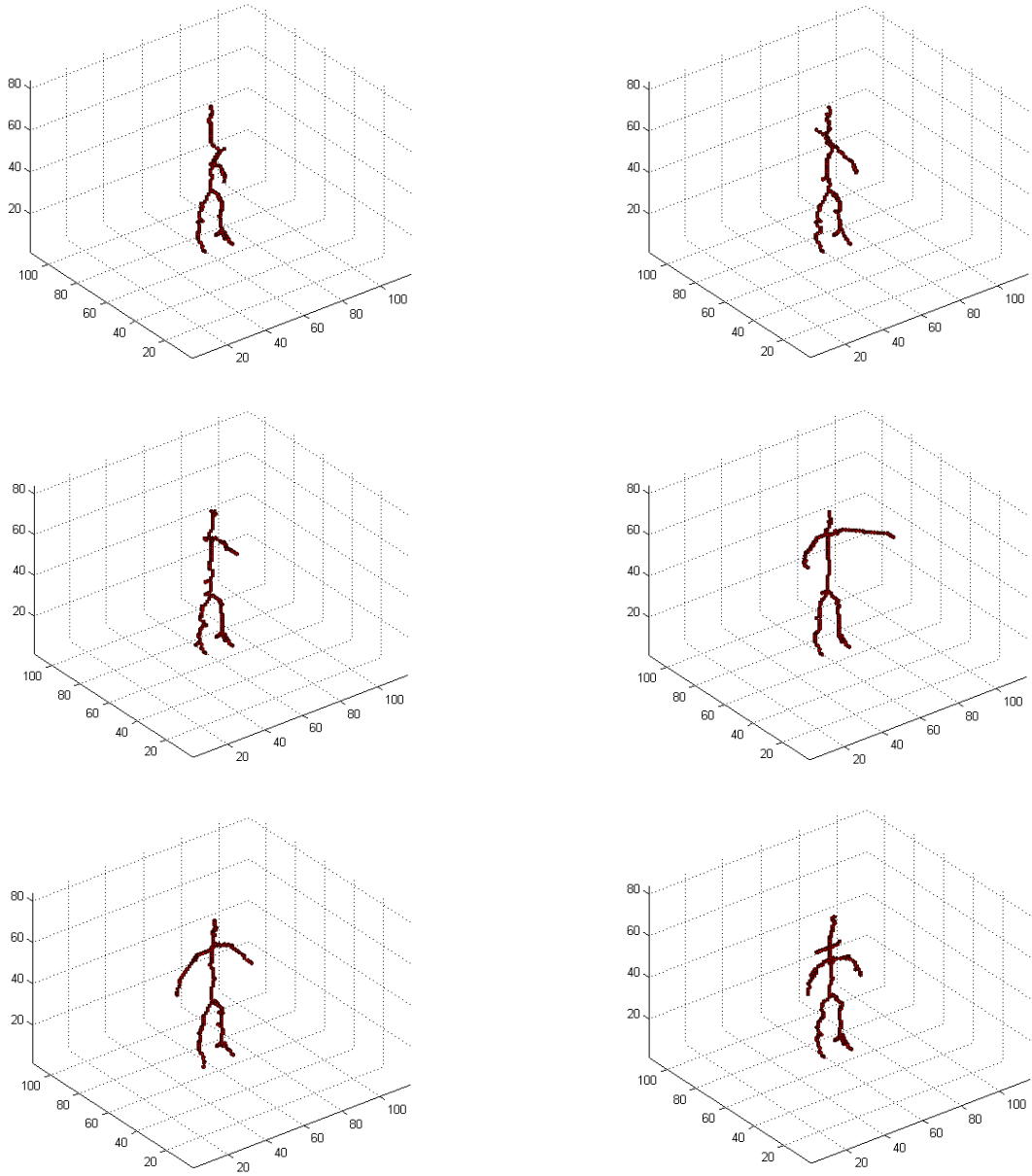


Figure 5.10: Thinning using the proposed algorithm on six reconstructed frames from the action: **Open**.

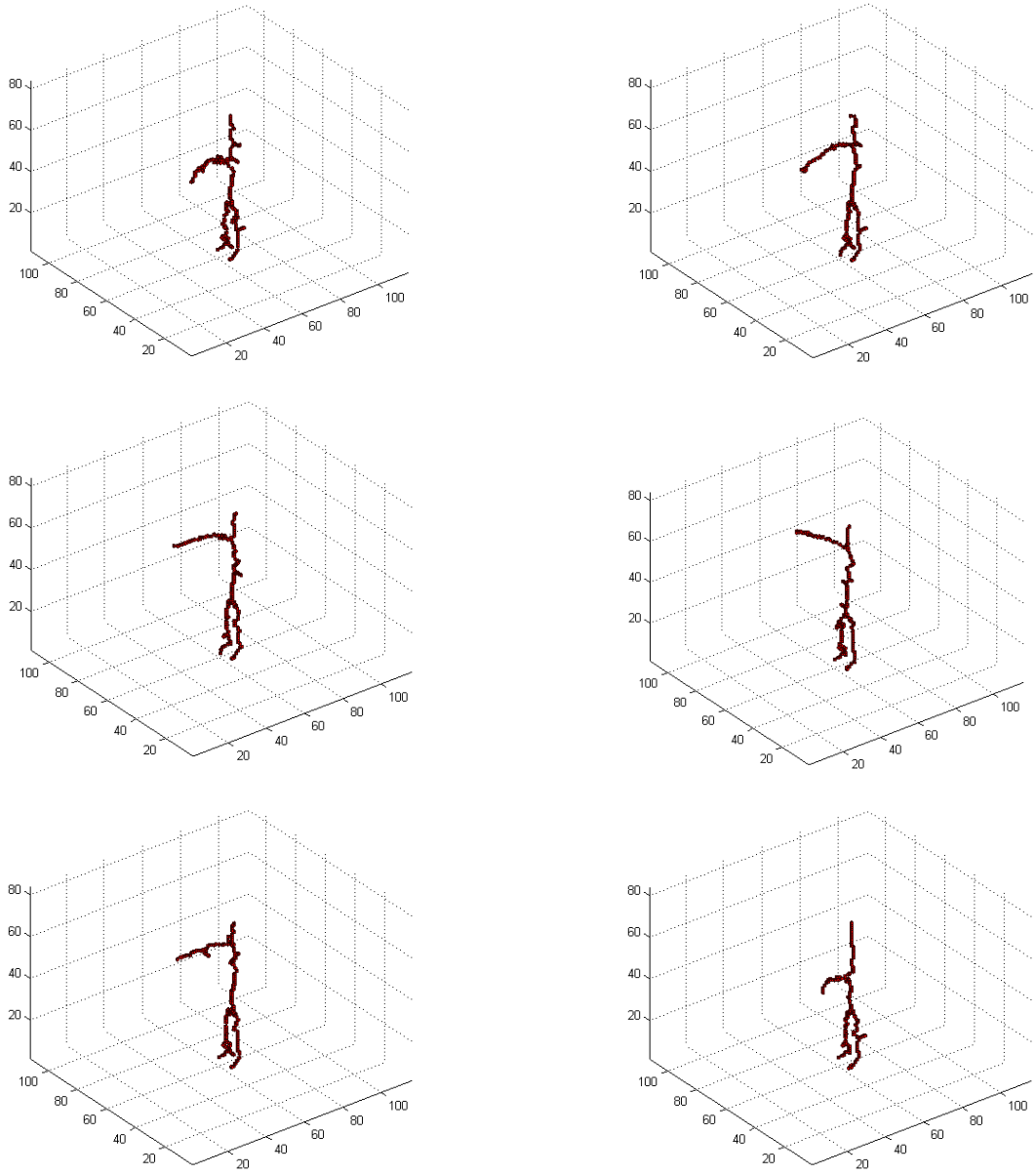


Figure 5.11: Thinning using the proposed algorithm on six reconstructed frames from the action: **PointAt**.

5 Curve Skeleton

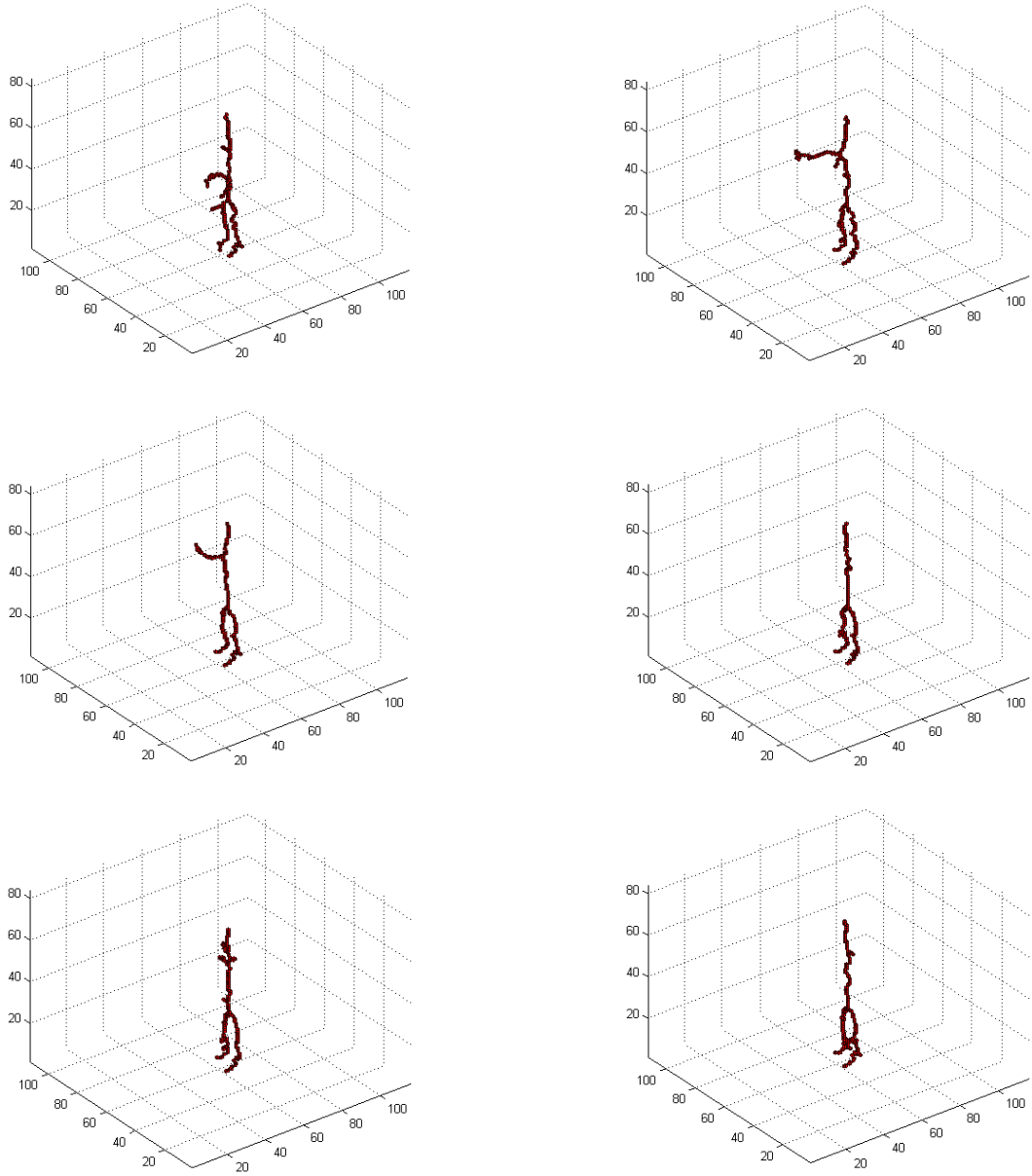


Figure 5.12: Thinning using the proposed algorithm on six reconstructed frames from the action: Pull.

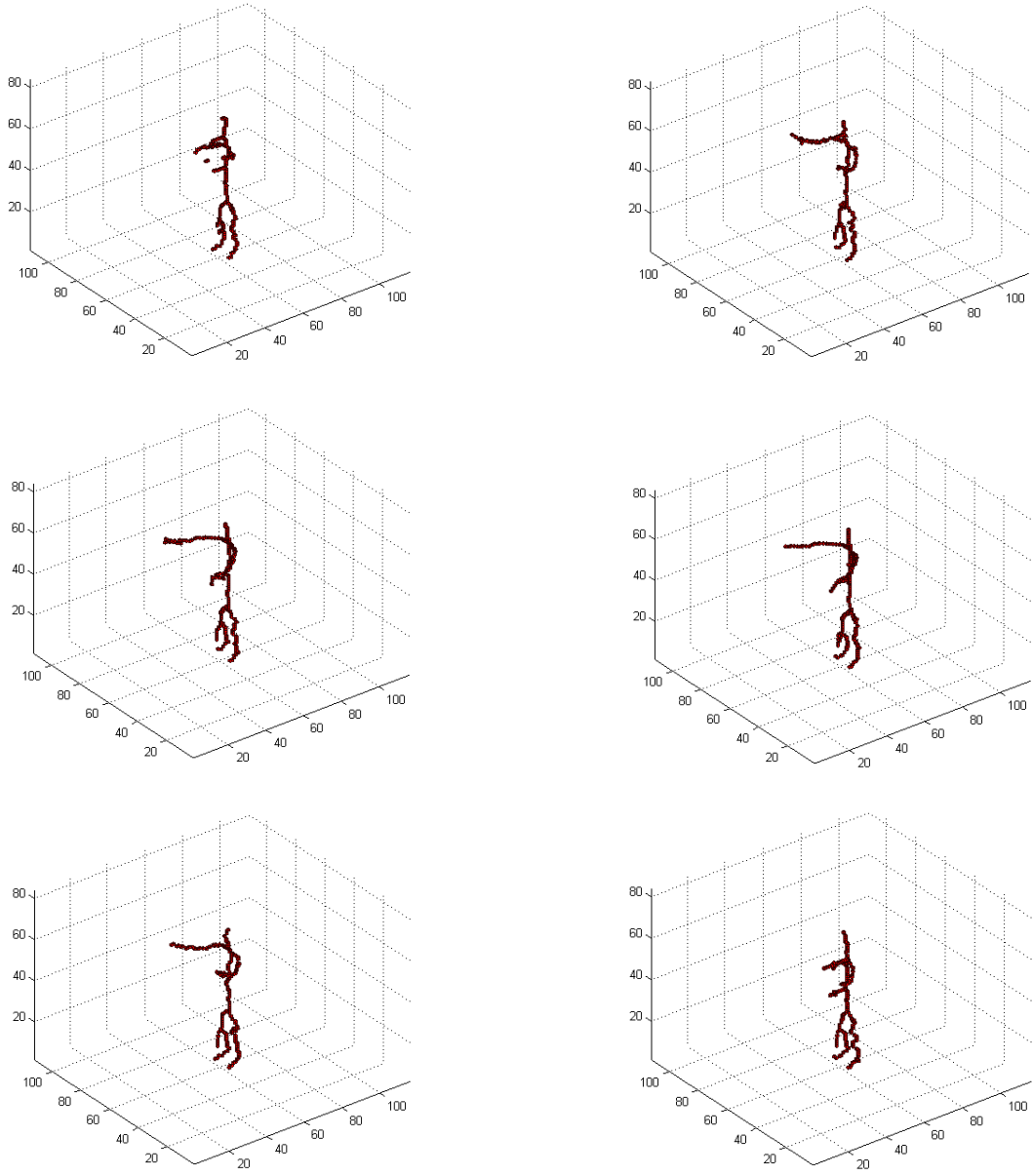


Figure 5.13: Thinning using the proposed algorithm on six reconstructed frames from the action: **Push**.

5 Curve Skeleton

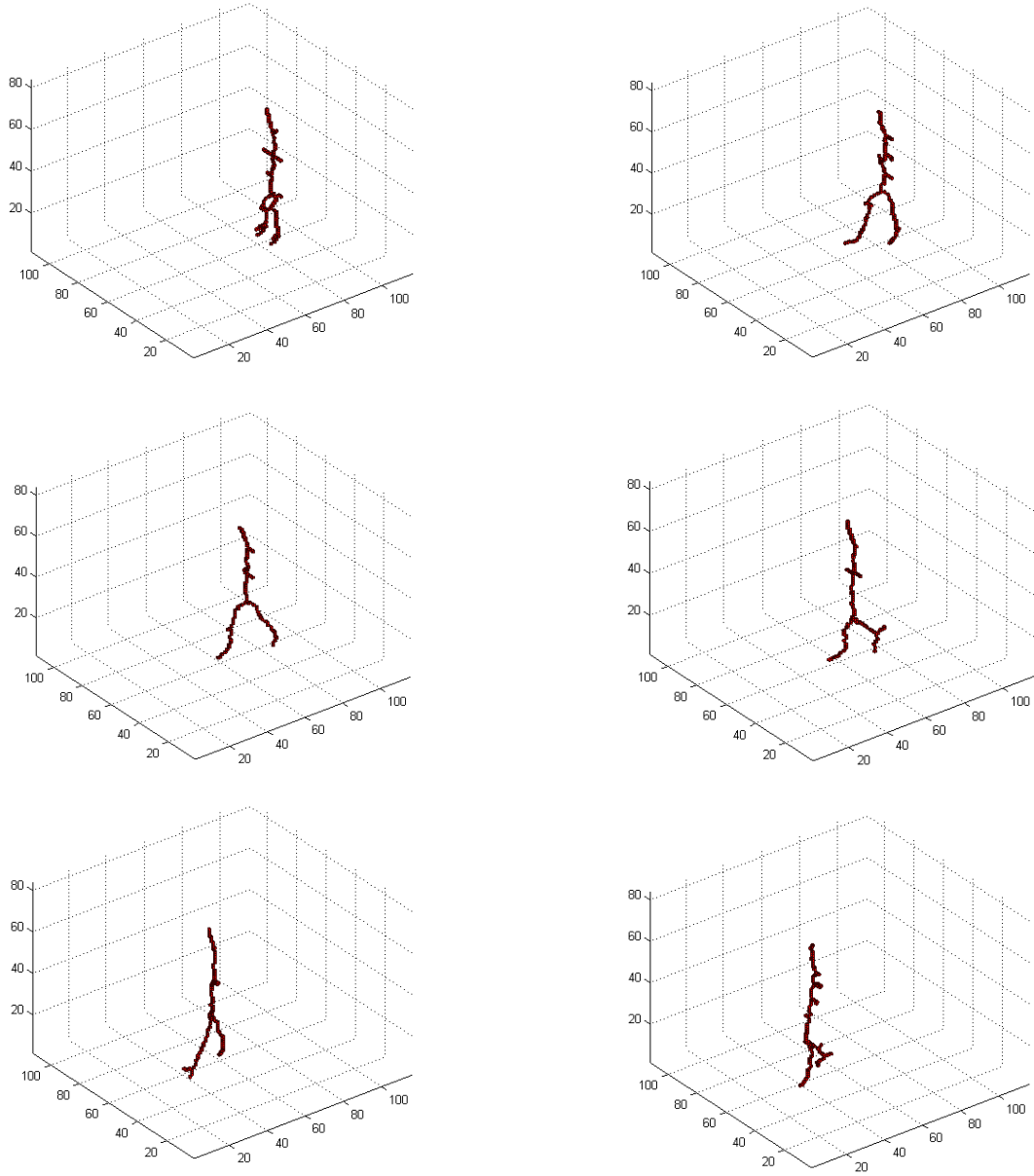


Figure 5.14: Thinning using the proposed algorithm on six reconstructed frames from the action: **Walk**.

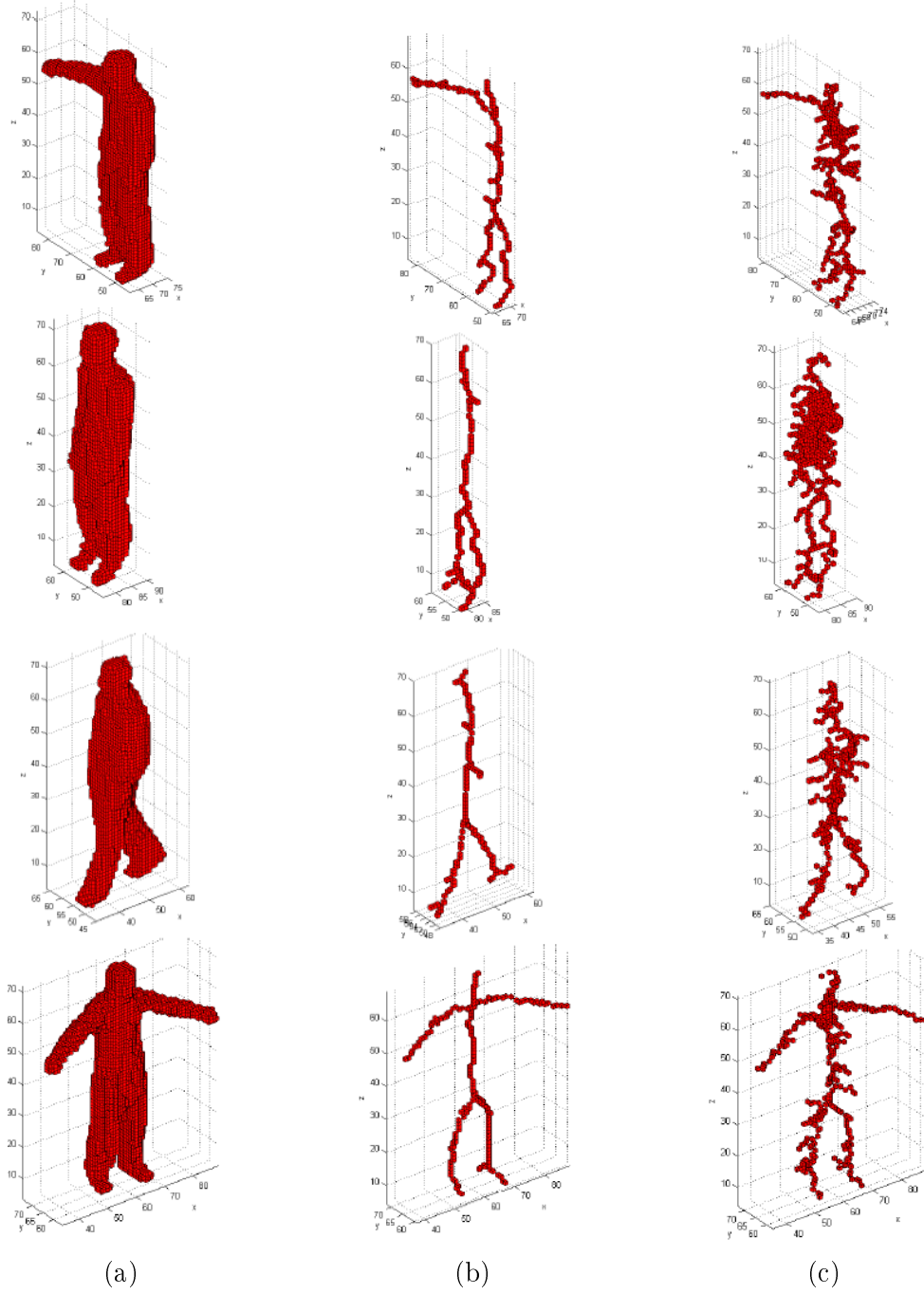


Figure 5.15: (a) Original 3D objects. (b) Curve skeletons obtained using our algorithm. (c) Curve skeleton obtained using Palágyi and Kuba algorithm [93].

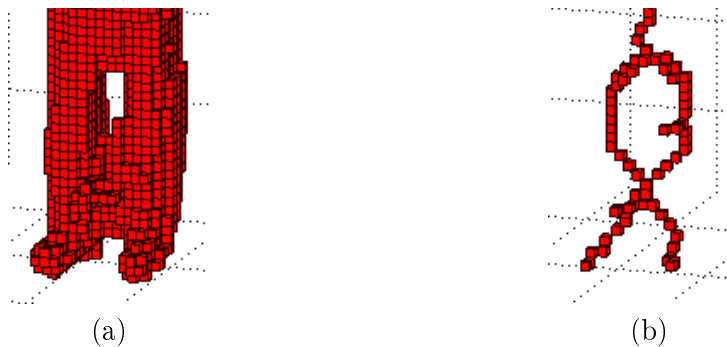


Figure 5.16: Example that shows topology preservation. (a) Original object. (b) Curve skeleton.

We use this database of real data because our first purpose is to analyze the extracted curve skeletons in a real situation. The desired thinning algorithm must be able to fast extract the body main posture for each pose, without being too sensitive to noise or shape details. Topology and connectivity preservations are essential constraints that can be easily achieved paying attention on the definition of the structuring elements. Centeredness and thinness are soft constraints in our implementation: the thinned voxel set must be as geometrically centered as possible within the object boundary, as thin as possible, and must look similar to the object shape. The main reasons are related to computation load and to the ability to recover the original object from the extracted skeleton. As it is described in details in Chapter 6, the implemented method to represent the object surface from the extracted skeleton requires that the thinned voxels are far from the surface itself. Finally the invariance under rotation is related to the action classification implementation (see Chapter 7), but it is useful in many applications: the characterization of the object must remain constant when the object is moved.

Some examples of input objects (reconstructed body poses) are shown in Fig. 5.15(a), together with the corresponding curve skeletons extracted using our algorithm (b) or the Palágyi and Kuba algorithm (c). As it can be noticed, the original 3D reconstructed object boundaries are not always smooth and it can be considered as noise on the boundary. It affects the result, but we focus on testing our algorithm for real situations and not for regular geometric shapes.

It can be proved that if a point $\mathbf{p} (\in B)$ can be deleted by at least one structuring element in $\{E_U\}$, then $\mathbf{p}$ will be simple. It must be proved only for the four basic structuring elements since the simplicity is a rotation invariant property. Topology

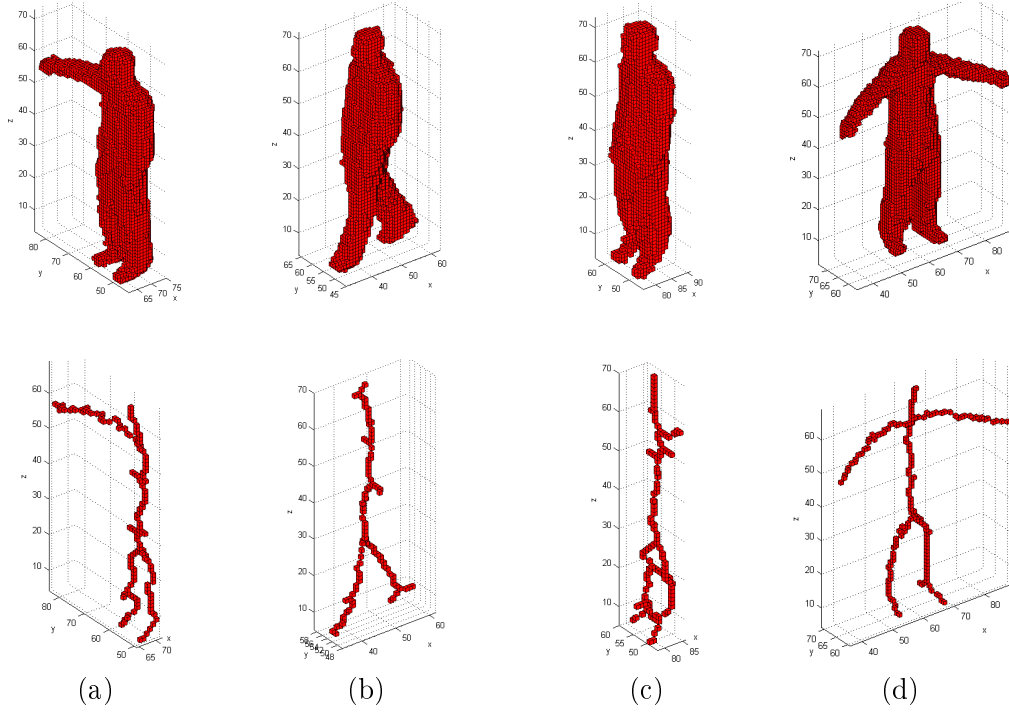


Figure 5.17: Original object of the examples proposed in Fig. 5.15 corrupted with additive Gaussian noise with standard deviation equal to 0.2 (top row). Curve skeletons (bottom row).

preservation is satisfied by our curve skeleton, since removal is done sequentially by using the notion of simple point and no pruning post-processing is done. Since each single deleting criterion is applied alone, we need to demonstrate that each structuring element preserves topology even when the central voxel is removed. For $E_U^2 - E_U^4$ structuring elements, it is easy to demonstrate, because the remaining 2 or 3 black voxels are 26-connected and, moreover, at least one of those is 26-connected with each of the “x” points. For E_U^1 we can observe that the remaining black voxel is 26-connected with each of the “.” points. Since at least one point marked “.” is black, the 26-connection is guarantee (and, as consequence, connectivity is preserved). A visual demonstration of this property is shown in Fig. 5.16, where the tunnel created by the wrong 3D reconstruction obtained from the multi-Kinect system is exactly reported in the curve skeleton.

From the examples proposed in Fig. 5.15(b), we can see that our curve skeleton is sufficiently thin even if it is not always 1-voxel thin (cross points are present in correspondence of both joints and particularly noisy points). The reason is that our algorithm, as the Palágyi’s method [93], is not able to delete all the simple points.

5 Curve Skeleton

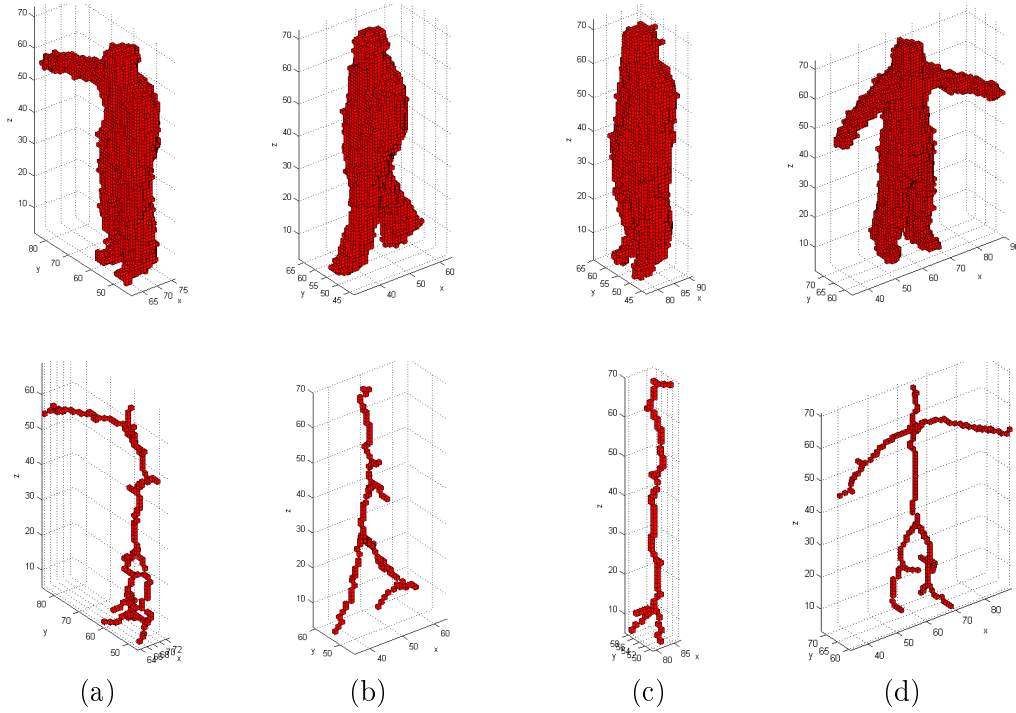


Figure 5.18: Original object of the examples proposed in Fig. 5.15 corrupted with additive Gaussian noise with standard deviation equal to 0.6 (top row). Curve skeletons (bottom row).

Note that the “goodness” of a thinning algorithm is not directly proportional to the number of the simple points that are recognized as points to be deleted [93]. As a matter of fact, looking at the examples in Fig. 5.15, we can state that our method is able to extract the main 3D object axis of symmetry, even if the extracted skeleton is not completely 1-voxel thin. Moreover, from the examples proposed in Fig. 5.5, it seems that, when the input voxel set is particularly noisy (the acquisition system has problems to reconstruct the 3D space where the actor body is crouching, for example), the extracted skeleton loses its thin property.

The thinning process may produce undesired branches due to noisy boundary. In order to evaluate the robustness of the curve skeletons to small changes in the surface, we add a Gaussian noise to the surface voxels and we compare the extracted skeletons. We propose the results for two values of standard deviation 0.2 and 0.6. In the first case 99.7% of the noised voxels are inside ± 0.6 respect to the center of the original voxels, *i.e.*, a voxel could be added to each of the original ones. In the second case 99.7% of the noised voxels are inside ± 1.8 respect to the center of the original voxels, *i.e.*, two voxels could be added to each of the original ones. This appears as

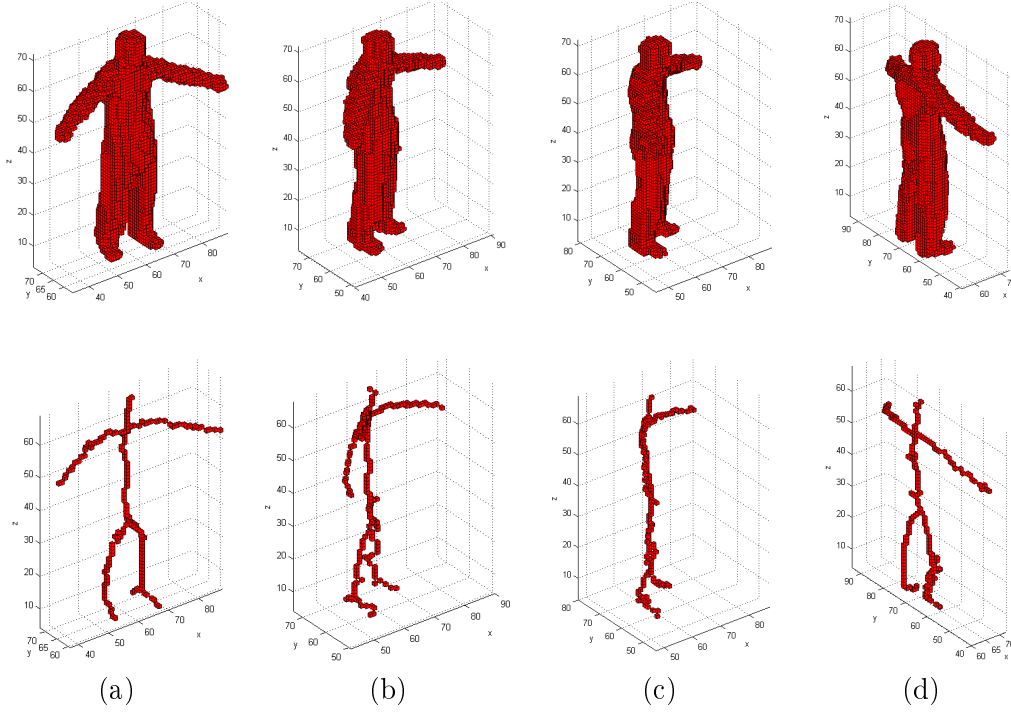


Figure 5.19: Object rotated around z axis of (a) 0° , (b) 30° , (c) 45° , (d) 90° (top row) and the extracted curve skeletons (bottom row).

a great change of the original surface. This could also change the topology of the 3D object (Fig. 5.17(c) and Fig. 5.18(a) and (c)), and, consequently, of the extracted curve skeleton. Experimental results shown in Fig. 5.17 and Fig. 5.18 demonstrate the behavior of our algorithm under additive Gaussian noise with standard deviation equal to 0.2 and 0.6 respectively. In Fig. 5.17 it can be appreciated that the curve skeletons are very similar to those of the noise-free voxel set, except for some small branches due to the irregularities in the surface introduced by the noise. In Fig. 5.18 the amount of noise is very high and often changes the topology of the object changing the curve skeleton in those parts as well. Omitting these parts, however, it can be seen that more small branches appear, due to the merge of many noisy voxels in a particular part of the object. Anyway these new branches are surprisingly few and the extracted curve skeleton remains quite smooth also for very noisy objects. From these experiments, we can conjecture that the compact crowd of thinned voxels that appears in Fig. 5.5 is due to the presence of many secondary branches clustered in particularly noisy regions.

Thinness is often in contrast with the invariance under isometric transformations

and the ability to recover the original object. Involving discrete representations of objects, the finite resolution of the voxel grid produces small errors when objects are transformed. Leaving out these errors, Fig. 5.19 demonstrates (only for the last example in Fig. 5.15, but we have verified the results for all the database) that our method is robust under isometric transformations. In particular, we focalize our attention on rotations around the vertical axes, the most important for an action classification scenario. It can be appreciated that, even if the skeleton of the rotated object $Sk(T(B))$ is not exactly the rotated original curve skeleton $T(Sk(B))$, they are very similar and moreover they both capture the main parts of the original shape.

Thinning does not guarantee centeredness and, in the case of directional thinning, this would depend on the order in which the different directions are applied. We choose to follow the directions U, W, S, D, E, and N in order to be as isotropous as possible. We cannot mathematically demonstrate that if the order of directions changes, we will obtain the same curve skeleton, but experimentally this is what happens. One possible way to quantify the centeredness of a curve skeleton is to seed a number of uniformly distributed radial rays in a plane normal to the direction of the curve skeleton at each one of its points, and to measure the distance to the boundary along each of these rays. Centered points should have the same distance to the boundary along each pair of opposite rays. In order to evaluate the centredness of the extracted curve skeleton, we consider only its line points (a black point $\mathbf{p}$ is a line point if there are exactly two black points m-adjacent to $\mathbf{p}$). We trace eight uniformly distributed radial rays in a plane orthogonal to the direction of the curve skeleton in that point and we compute the distance to the boundary along each of these rays. We compute the mean difference of these distances and we obtain a value of 1.0883 voxels that indicates that the extracted curve skeleton is well centered.

In order to increment the efficiency of the method, we apply the thinning procedure at each iteration only on the border voxels: the deleting conditions are tested only on object voxels with a face-neighbor in the background. We obtain the same result and we reduce the computation time of 31% in mean. We compare the efficiency of our algorithm respect to Palágyi and Kuba algorithm [93], considering the computation time for the same voxel sets and the number of iterations needed to obtain the curve skeleton. Table 5.1 summarizes the mean values in terms of number of thinned voxels and iteration steps of the thinning process applied on each of the reconstructed frame for the 25 (5 actors and 5 performances) sequences of each action in the database. We are able to obtain curve skeletons with less voxels in less number of iterations

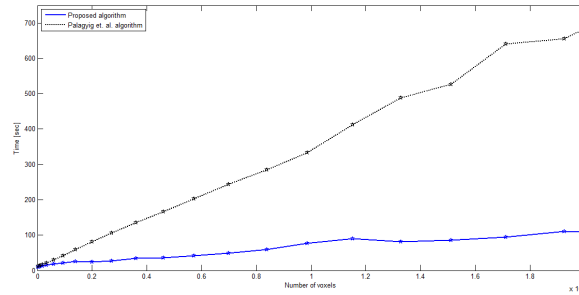


Figure 5.20: Efficiency of our algorithm compared to Palágyi and Kuba algorithm [93] in terms of computation time respect to the number of voxels of the original object. The algorithms are run on a PC under Windows 7 (64 bit), using a 1.60 + 1.60 GHz Intel Dual Core CPU. Note that just the thinning itself is considered here; reading the input volume and writing the output image are not taken into account, but the processing involved is not excessive.

(we achieve curve skeletons with 61% less voxels in mean: our skeletons have less branches but do not loose important information). Fig. 5.20 shows the computation time respect to the number of voxels composing the original object. It can be inferred that, starting from the same voxel set, our algorithm is faster than Palágyi and Kuba algorithm [93]. Moreover, the computational time reduction highly increases as the number of voxels increases.

Table 5.1: Efficiency of our algorithm compared to Palágyi and Kuba algorithm [93] in terms of voxels of the extracted curve skeletons and number of iterations needed.

Action	Voxels	Our algorithm		Palágyi <i>et al.</i>	
		Sk-voxels	It.	Sk-voxels	It.
Crouch	6775	117	10	367	13
Grasp	7315	145	9	291	11
Kick	6873	118	13	365	15
March	6710	105	7	458	17
Move	7361	134	8	311	13
Open	7199	162	5	311	7
PointAt	6949	136	9	288	11
Pull	7300	137	9	300	10
Push	7110	179	9	362	11
Walk	7384	105	8	289	14

Reconstruction

The examples in Fig. 5.15(b) show that the extracted curve skeletons have a number of curves corresponding to the main parts perceived as composing the objects. To give some less subjective evaluation, we compute the representative power of the thinned voxel set in terms of reconstruction ability. The precise reconstruction of the original object from its representation is a delicate and high challenging process. For this reason the literature devoted to this goal is awfully wide and covers very different techniques (the power crust of Amenta *et al.* [2] is just an example to which we will refer in Chapter 6).

Since a high level of reconstruction accuracy is not required, neither for this evaluation nor for the developed applications, we implement a simple method that can be considered a voxel approximation of the ball growing approach. Each thinned voxel is substituted by a solid ball centered in the voxel and with radius equal to the distance from the thinned voxel to the nearest point on the surface.

In order to compute the object surface, we use the algorithm proposed in Section 4.3.1: the surface is computed by a 3D morphological dilation with a $3 \times 3 \times 3$ structuring element and then subtracting the original 3D object to the result. This surface extraction method over-estimate the object space occupancy, as Fig. 5.21 (top row) shows. It is a very fast method and, moreover, it has the great advantage of permitting the approximation of the real volume as union of spheres. Over-estimating the space occupancy, there is not the possibility that a voxel on the surface is also a thinned voxel: the two voxel sets have no voxel in common, neither the border voxels.

In Fig. 5.21 we compare the voxel sets reconstructed starting from the curve skeleton obtained with the proposed thinning algorithm (middle row) and the Palágyi and Kuba thinning algorithm [93] (bottom row) for the examples proposed in Fig. 5.15. We can appreciate that the pose is fully understandable from our reconstructed volumes, even if some unimportant details are lost. Looking at the bottom row, we can say that many thinned voxels obtained with the Palágyi and Kuba algorithm have distance close or equal to 1 to almost one of the surface voxels. This produces a reconstructed volume with many superficial solid balls that do not add any finer detail or any necessary information. These balls are due to the many secondaries branches obtained in the voxel set.

In order to make a more general evaluation, we compute and compare the reduction

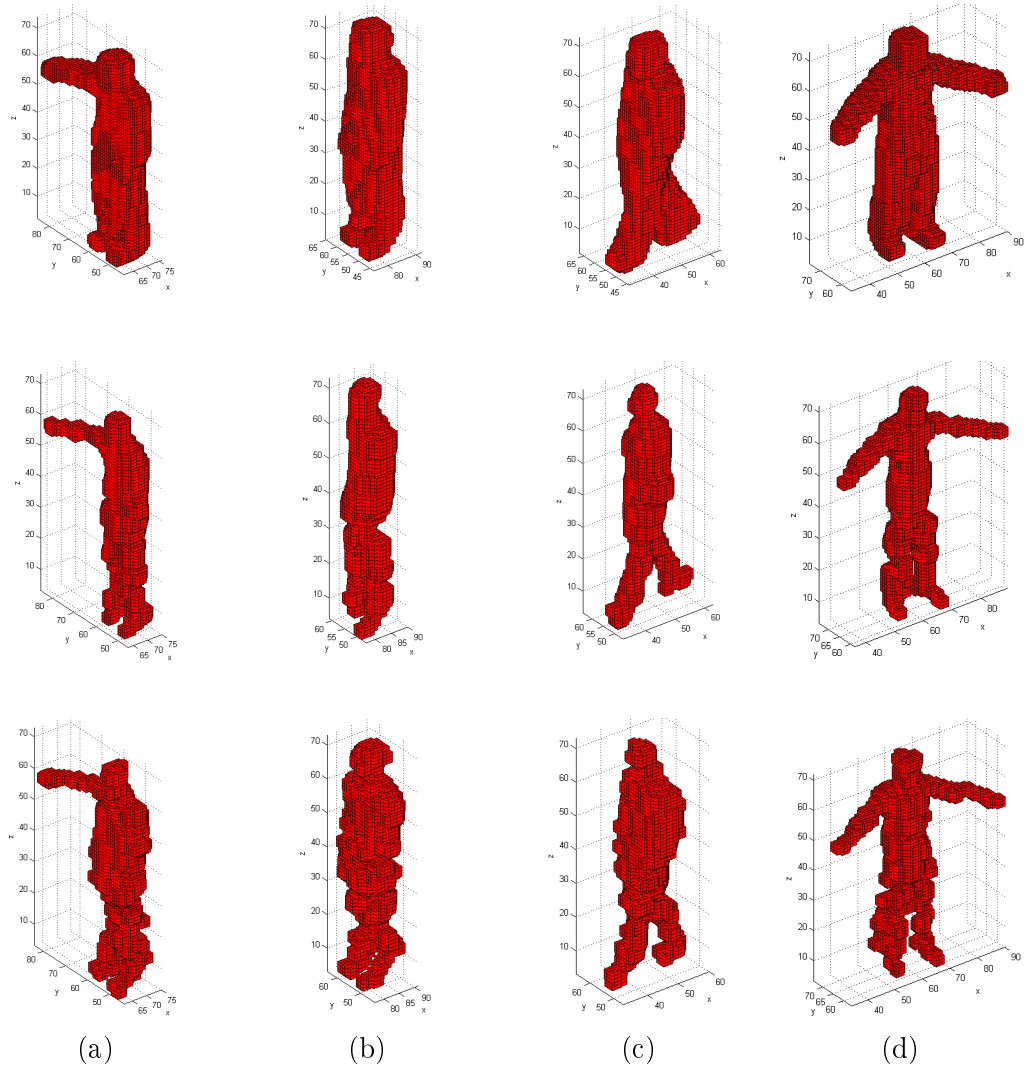


Figure 5.21: (Top row) Surface extracted from the volumes shown in Fig. 5.15(a). (Middle row) Volumes recovered from the curve skeletons extracted with the proposed algorithm (shown in Fig. 5.15(b)). (Bottom row) Volumes recovered from the curve skeletons extracted with the Palágyi and Kuba algorithm [93] (shown in Fig. 5.15 (c)).

and representative power of both the tested algorithms (Table 5.2). The reduction power is defined as the percentage of voxels in the curve skeleton with respect to the number of voxels in the original object. The representative power is the percentage of reconstructed voxels (obtained applying the volume approximation as union of voxel-balls centered on the extracted skeleton) respect to the number of voxels in the original object. For a given radius, each solid ball is approximated considering all

5 Curve Skeleton

the voxels that are inside or touch the continuous surface. The implemented method deliberately under-estimate the original volume, in order to obtain a reconstructed volume that still maintains the object topology. The results are estimated for each reconstructed frame and mediated on the class of the action they belong to. We obtain a reduction power, in mean, of 1.8855%, while the reduction power achieved with the Palágyi and Kuba algorithm is only 4.7306%, less than our as it can be intuitively deduced from the presence of the many secondary branches on the Palágyi and Kuba extracted curve skeleton. The representative power is 81.3251% in mean for the proposed algorithm, compared with the 86.9024% of Palágyi and Kuba. The secondary branches that produces the superficial balls, that can be seen in Fig. 5.21 (bottom row), are the main cause of this grater value but, as we have already observed, it is not translated in a more detailed reconstruction.

In the next Chapter we illustrate in detail a surface reconstruction method that takes inspiration from the proposed volume reconstruction algorithm, but it is devoted to a representation of the surface alone and not of the entire volume.

Table 5.2: Data reduction and representative power for our and the Palágyi and Kuba [93] thinning algorithms.

Action	Our algorithm		Palágyi <i>et al.</i>	
	Reduction	Reconstruction	Reduction	Reconstruction
Crouch	1.7269	79.3873	5.3875	88.3271
Grasp	1.9822	78.9077	3.9781	84.9225
Kick	1.7168	73.8251	5.3106	82.4924
March	1.5648	74.8080	6.8256	83.0415
Move	1.8204	79.9437	4.2250	85.2532
Open	2.2503	84.8467	4.3200	88.7396
PointAt	1.9571	84.0553	4.1445	93.1357
Pull	1.8767	85.1918	4.1095	85.7671
Push	2.5175	93.8115	5.0914	91.2940
Walk	1.4420	78.4736	3.9139	86.0510

6 Surface Representation

In this Chapter we describe two surface reconstruction methods that are used to represent the human body, acquired with our system and represented as a voxel set, as union of balls or as an implicit surface. The goal of a surface reconstruction algorithm is to construct a digital model of a physical surface. The methods compute a surface S' that approximates an unknown physical surface S using a sampled set $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ of points on it. Eventually, some additional information about the sampling process or the geometry of the manifold S can be available as well.

Recent improvements in automated shape acquisition have stimulated a profusion of surface reconstruction techniques for computer graphics and reverse engineering applications. A typical surface acquisition application consists of the following steps:

- Data point acquisition: measurement of 3D coordinates for points on the surface, producing range images as depth maps.
- Registering and integration: if multiple views are used, a registering process is performed in order to find the transformations that put the data in a common coordinate system, followed by the integration of all the data in a unified representation.
- Surface fitting: construction of a concise and accurate approximation of the physical surface.

In Chapter 2, the developed acquisition system is described. Chapter 3 is devoted to the illustration of the proposed procedure for obtaining the 3D voxel set representing the actor in the scene, while in Section 4.3.1 a morphological algorithm to extract the voxel surface starting from the volumetric occupancy is proposed. The centers of the voxels surface are the available sampled set X for the surface representation methods described in this Chapter.

6 Surface Representation

Surface reconstruction methods can be roughly classified as *Voronoi-based* and *mesh-free*. Voronoi-based reconstruction algorithms compute the Delaunay triangulation of the sample points, the dual of the Voronoi diagram. A subcomplex, interpolating the sampled surface, is then extracted from the Delaunay triangulation [2], [18], [58]. In the mesh-free approaches, the surface is approximated or interpolated using explicit methods (such as deformable models), parametric methods (such as B-Spline), or implicit methods (such as Radial Basis Functions (RBF), Poisson (PDE) or Moving Least Squares (MLS)) [57], [115], [120]. Among the many techniques developed for surface reconstruction, we can observe that even if the results are very accurate, these methods are time consuming, computationally heavy, and, above all, based on data acquired in a very accurate manner, such as with a laser scanner. The demand for domestic applications has led to the development of hardware that quickly extracts advanced information of the scene, as Microsoft Xbox Kinect sensor [52]. In this case, however, the surface sampling accuracy is not as high as required by the aforementioned techniques. Basing on all these considerations, we decide to build a system for the reconstruction of the object surface acquired with low cost accuracy DIBR devices. This representation is robust to noise acquisition, fast and easy to implement, even if it is not as accurate as the state of the art methods.

The proposed solutions for surface reconstruction (in this Chapter we use surface representation and reconstruction as synonyms) take inspiration from two well known algorithms: the surface representation as union of balls, proposed by Amenta and Kolluri [3], and the surface representation as an iso-surface of a function defined as linear combination of radial basis functions, proposed by Samozino *et al.* [100]. Both methods require high surface sampling density. Therefore, in order to compare the results of the proposed solutions for surface reconstruction with the two techniques taken as inspiration, we develop a simulated environment. This simulation allows to obtain a volumetric reconstruction with a space discretization finer than that achievable with the multi-Kinect acquisition setup developed in the ISPG laboratory.

This Chapter is organized as follows: in Section 6.1, the method applied for building the depth maps database used to test the proposed surface reconstruction approaches is described. Section 6.2 talks about the surface representation as union of Maximal Balls [3] and Section 6.3 is about the surface representation as an iso-surface of a function defined as a linear combination of Radial Basis Functions (RBF) [100]. Section 6.4 revisits the described methods introducing new surface reconstruction

methods that use the information achievable from the volumetric reconstruction of the scene. Section 6.5 extends the proposed methods to the use of curve skeleton voxels (extracted using the proposed thinning algorithm described in Chapter 5) for the surface representation.

6.1 Simulated Acquisition Set-up

Taking into account the high surface sampling density required by the surface algorithms presented in the literature, we cannot use our database of reconstructed volumes in order to test the implemented methods. We simulate a similar acquisition setup using LightWave 3D in order to achieve a 3D reconstruction of the object in the scene with a sampling density sufficient for applying both the aforementioned techniques for surface reconstruction. In this Section we describe the implemented process for simulating a 3D acquisition set-up similar to our multi-Kinect system, but able to obtain a finer sampling of the 3D space.

LightWave 3D is a high end computer graphics program developed by NewTek [44]. It is a software package used for rendering 3D images, both animated and static. It includes a rendering engine that supports advanced features as realistic reflection and refraction. The 3D modeling component supports both polygon modeling and subdivision surfaces. The animation component has features such as reverse and forward kinematics for character animation, particle systems and dynamics.

6.1.1 Building Synthetic Depth Maps using LightWave

The great potential of LightWave [7] is here exploited to build perspective depth maps, where the gray tone of each pixel represents the distance of the corresponding 3D point to the optical center of the depth-camera taken as reference (the IR-camera of an hypothetical Kinect device in our case). This result is achieved by imposing a specific texture on the illuminated surface of the subject. In details, given any synthetic object, as shown in Fig. 6.1(a), a perspective depth map that reflects the configuration of Fig. 6.1(b) can be achieved by imposing as texture a gradient that changes the gray tone of the object linearly as a function of the distance from the IR-camera optical center. We use a model of a human hand in order to achieve the sampling accuracy required by the surface reconstruction algorithms implemented. The camera in Fig. 6.1(b) reflects the position of the IR-camera of the simulated

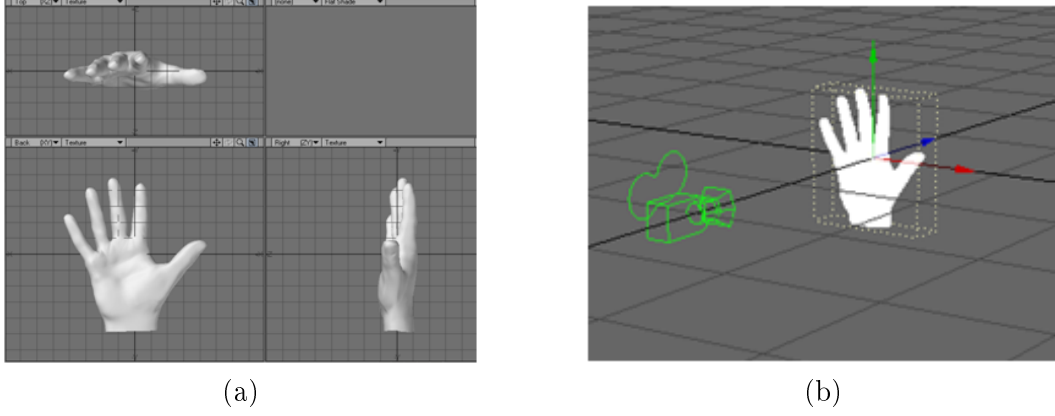


Figure 6.1: (a) Synthetic Right Hand (20 mm grid) and (b) configuration for the depth map acquisition.

Kinect device. The gray tone I_{min} ($= 0$, *i.e.*, black) is associated to all the background pixels. They represent points at a distance greater than or equal to d_{max} from the IR-camera (corresponding to 435 mm for our example). The gray tone I_{max} ($= 255$, *i.e.*, white) is associated to all the pixels representing points at a distance less than or equal to d_{min} (corresponding to 375 mm for our case). All the pixels that represent points with distance between d_{min} and d_{max} have a gray tone that is linearly decreasing, while the distance of the corresponding point to the optical center of the device is increasing. Thanks to the linear gradient and imposing no light except the ambient light, we can achieve any depth map simulating a generic viewpoint, as the one shown in Fig. 6.2(a), which reflects the acquisition conditions of Fig. 6.1(b). Using LightWave, we are able to simulate perspective depth maps for any acquisition set up and with any resolution needed.

6.1.2 Canonical Volumetric Reconstruction

Having the depth map and all the IR-camera parameters, we can reconstruct a sampled version of the object surface standing in front of the simulated Kinect device (as shown in Fig. 6.2(b)). Knowing the depth map and the linear gradient equation, the distance of any 3D point which corresponds to any pixel in the image can be estimated as:

$$d = \frac{d_{min} - d_{max}}{I_{max} - I_{min}}(I - I_{max}) + d_{min}. \quad (6.1)$$

Assuming that the system is calibrated, both the intrinsic and extrinsic parameters

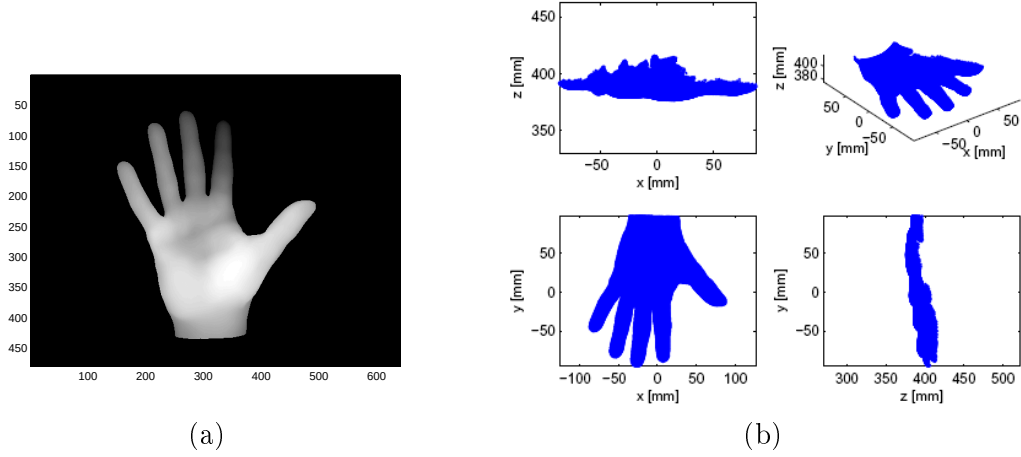


Figure 6.2: (a) Depth map achieved using LightWave and (b) 3D surface points reconstruction.

are known. Moreover, knowing the distance of each point from the IR-camera optical center, the 3D coordinates (p_x, p_y, p_z) of the point are easily evaluated:

$$p_z = d \frac{f}{\sqrt{(u - u_0)^2 + (v - v_0)^2 + f^2}}, \quad (6.2)$$

$$p_x = p_z \frac{u - u_0}{f}, \quad (6.3)$$

$$p_y = p_z \frac{v - v_0}{f}, \quad (6.4)$$

where (u, v) are the point coordinates on the image plane, (u_0, v_0) are the principal point coordinates, and f is the focal length.

Before doing the 3D reconstruction, the image is pre-processed in order to separate the background from the foreground and to remove any outlier (pixels with gray tone greatly different from those in the neighborhood). Moreover, using LightWave, we obtain perspective depth maps, that are then transformed in elevation depth maps in order to simulate a system that is as close as possible to our real system. In an elevation depth map the gray tone of each pixel represents the distance of the corresponding 3D point to the IR camera image plane, while in a perspective depth map the gray tone of each pixel represents the distance of the corresponding 3D point to the IR camera optical center. Knowing the 3D coordinates of each point on the acquired surface, it is straightforward to compute the gray tone of each pixel

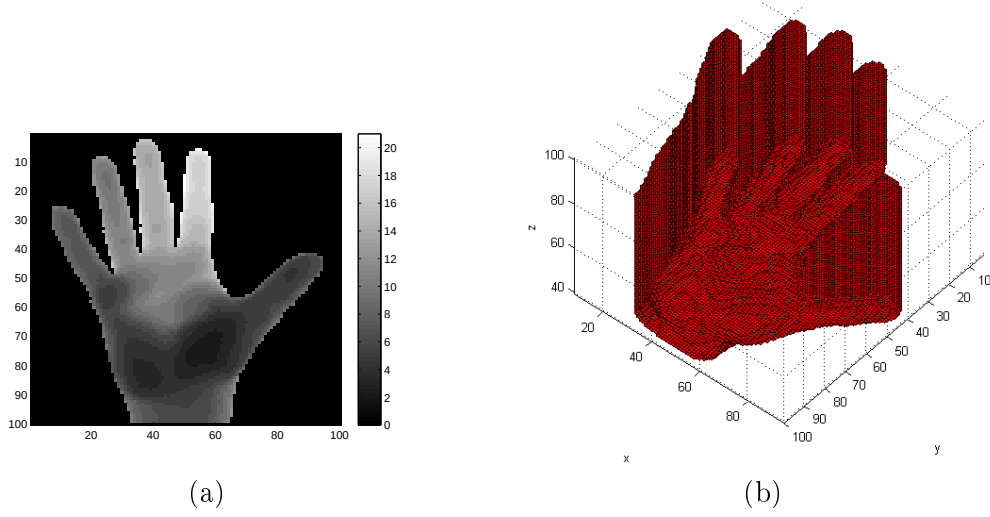


Figure 6.3: (a) Elevation depth map and (b) volumetric rendering obtained from the depth map.

in function of the distance from the image plane, as shown in Fig. 6.3(a): each 3D estimated surface point is projected again on the image plane with a gray tone level (d_{el}) equal to:

$$d_{el} = p_z - f. \quad (6.5)$$

A bicubic interpolation is then applied in order to minimize artifacts due to quantization. Having this new calibrated depth map, we are able to compute the 3D coordinates of each pixel and to project each surface sample till the the maximum extension of the considered space (the projection cylinder shown in Fig. 6.3(b)). The volumetric rendering is achieved working with voxels (in this case using box of 2 mm side).

6.1.3 Volumetric Integration

As for our real acquisition system, we implement a multi-view acquisition setup to obtain a complete description of the 3D object. Therefore, it is necessary to merge the 3D information coming from different depth maps taken from different viewpoints. As in the real case, we first reconstruct the scene seen by each device and then merge the volumetric rendering achieved from any view. This volumetric reconstruction is performed for all the available views by assuming a regular grid for each of them (canonical space for each view). Once the voxel space centered on each device is obtained, applying the roto-translation matrix that brings each view to the reference

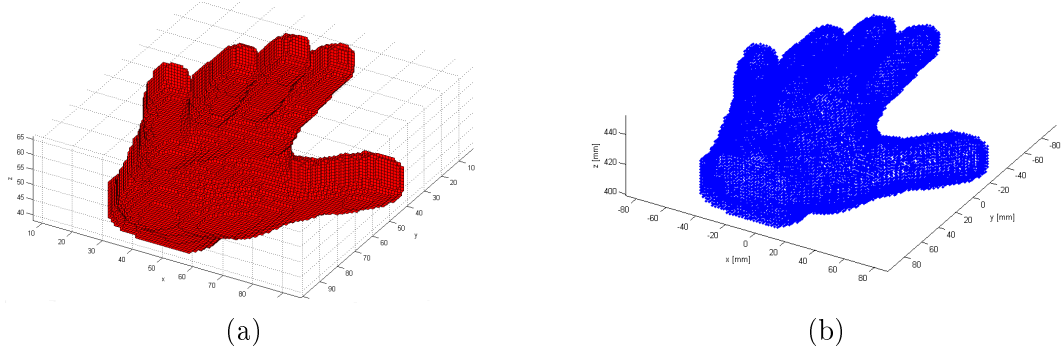


Figure 6.4: (a) Voxel surface and (b) sampled surface of the volumetric rendering obtained from the depth map.

view, it is possible to intersect the different projection cylinders applying a logical *AND* between voxels, as shown in details in Chapter 3. The result is a unique voxel set that gradually incorporates information coming from all the devices. In Table 6.1 the depth maps achieved with three other configurations, in which the devices are turned by $H = +45^\circ$, $H = -45^\circ$ and $H = 180^\circ$ respectively (using the LightWave definition for angles), the volumetric rendering obtained with each of them, and the voxel set obtained by increasing the number of views are shown. The resulting voxel set compactly represents all the available information about the object surface and its volume occupancy.

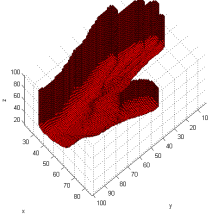
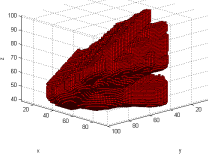
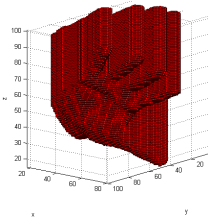
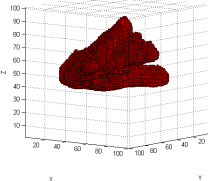
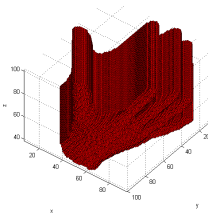
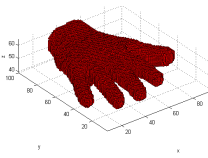
The object surface is computed using the morphological algorithm described in Section 4.3.1. In Fig. 6.4 the voxel surface (a) and their centers (b) are represented. The sampled set X is therefore composed by the 56168 centers of the voxels obtained with the surface extraction algorithm. The achieved sampling accuracy is sufficient for the considered surface reconstruction algorithms.

6.2 Surface Representation as Union of Maximal Balls

In this Section we describe the idea behind the surface representation as union of balls, following the method proposed by Amenta and Kolluri [3]. Starting from a dense point cloud of sample points, presumed to be on or close to the object boundary, first the Voronoi diagram of the scattered data points is computed and then the *medial axis* is approximated by a subset of Voronoi vertices, called *poles*. The medial axis (MA) and the medial axis transform (MAT), introduced by Blum [11], are fundamental concepts for surface reconstruction and they deserve to be explained

6 Surface Representation

Table 6.1: Volumetric Reconstruction using LightWave. Depth maps obtained with LightWave by rotating the acquisition system of H degrees. Volumetric reconstruction obtained from a single depth map. Volumetric reconstruction obtained downstream the merging.

Angle (H)	Depth Map	Projection Cylinder	Volumetric Reconstruction
45°			
-45°			
180°			

in detail.

6.2.1 Medial Axis and Medial Axis Transform

In 2D, the medial axis of a shape is a set of curves defined as the locus of points that have at least two closest points on the boundary of the shape [66]. In the 3D case, the corresponding object is called the *medial surface* because, in addition to curves, it can also contain surface patches [23]. Considering a 3D object $B \subset R^3$ and its boundary $\beta(B)$, the medial surface (MA from medial axis in the 2D case) of B is defined as the locus of centers of all the maximal balls of B which touch β in, at least, two points. A ball of radius r centered at $\mathbf{x} \in B$ is defined as:

$$Sr(\mathbf{x}) = \{\mathbf{y} \in R^3, d(\mathbf{x}, \mathbf{y}) \leq r\}, \quad (6.6)$$

where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between two points $\mathbf{x}$ and $\mathbf{y}$ in R^3 . A ball $Sr(\mathbf{x}) \subset B$ is maximal if it is not completely included in any other ball included in B [59]. The *medial axis transform* MAT is defined as the function that assigns to each medial axis point the radius value of its medial ball. It is shown that the object B could be extracted from $MAT(B)$ as the union of all the medial balls [3]. The boundary of B can be calculated as the boundary of the union of all the medial balls (inverse MAT). MA has another great advantage: an object and its inner MA have the same topological structure and can continuously deform to each other. Exact computation of both the MA and the MAT is extremely difficult in a three-dimensional space and, most importantly, it is extremely slow [2], [31], [33].

6.2.2 Medial Axis Approximation by a Set of Poles

The internal edges and faces of the Voronoi diagram of the scattered data is an approximation of the medial surface of the shape [3]. Each Voronoi vertex is the center of a circum-sphere of the Delaunay triangulation and the connection of all the centers of the circum-spheres produces the Voronoi diagram. Considering all the Voronoi vertexes produces many balls with centers close to the surface, corresponding to flat “sliver” tetrahedra in the Delaunay triangulation of X [3]. These balls may be (and almost always are) present even when the data are completely noise-free, at any finite sampling density. Their centers must be removed with a heuristic clean-up step. Different algorithms based on this approach use different methods for selecting the desired subset of the Voronoi diagram. In particular, after the computation of the Voronoi diagram of the scattered data points, Amenta *et al.* retrieve the poles [2]. The poles of a sampled surface point are the farthest vertex of its Voronoi cell in the interior of β and the farthest vertex of its Voronoi cell on the exterior of β . Connecting these interior poles P_I forms a good approximation to the MA and the union of the inner polar balls forms a good approximation of the object surface. Given a “good sample” from a smooth surface, they demonstrate that the output is guaranteed to be topologically correct and convergent to the original surface as the sampling density increases. The required sampling density varies locally, rigorously capturing the intuitive notion that featureless area can be reconstructed from fewer samples. Under the assumption that X is a sufficiently dense sample from β , they prove that:

- The boundary of the union of the set B_I of Voronoi balls centered at points P_I

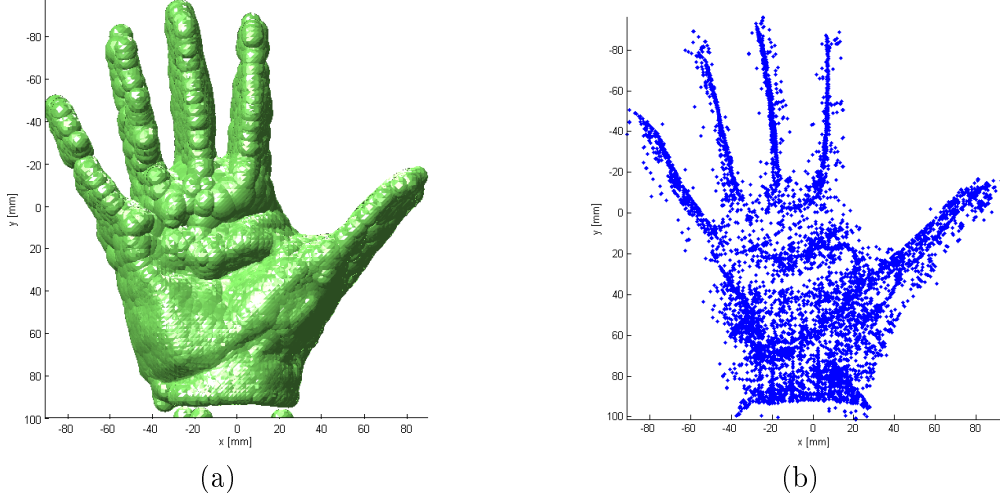


Figure 6.5: (a) Surface representation as Union of balls [3]. (b) Selected poles.

is close to B boundary.

- Its isosurface normals are close to nearby normals on β .
- It is homeomorphic to B. This implies a topological relationship between the true medial axis of the object and the medial axis of the union of balls.
- The set of balls centers in their construction converges to the true medial axis as the sampling density increases.

We implement the Amenta *et al.* algorithm [3] on the synthetic object shown in Fig. 6.1(a) and the result is shown in Fig. 6.5(a). Starting from the sampled 56168 points on the surface, we compute the Voronoi diagram and we select 8338 poles (Fig. 6.5(b)).

6.3 Surface Representation as Implicit Surface

In this Section we describe the idea behind the surface representation as an isosurface of a function defined as a linear combination of Radial Basis Functions (RBF), algorithm proposed by Samozino *et al.* [100].

Generally, implicit reconstruction methods attempt to find a smooth function $f: \mathbb{R}^3 \rightarrow \mathbb{R}$ such that $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ is close to the zero set $Z(f)$. Among the mesh-free approaches, Samozino *et al.* algorithm grabs our attention because it approx-

imates the surface as a zero level set of a function defined as a linear combination of compactly supported radial basis functions, each one associated with a polar ball computed using Amenta *et al.* approach [3].

The approximation problem is formulated as follows. Given $\{\mathbf{p}_i\}_{i=1,\dots,N}$ a set of N points (called *constraints*) and N scalar numbers $F = \{f_i\}_{i=1,\dots,N}$, find a function $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ satisfying the approximation condition:

$$f^* = \operatorname{argmin}_f E(f), \quad (6.7)$$

where E is the least square error:

$$E(f) = \sum_{i=1}^N (f_i - f(\mathbf{p}_i))^2. \quad (6.8)$$

In the RBF approach, the function f is defined from a class of basis functions $\Phi_j : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}$, as a linear combination:

$$f(x) = \sum_{j=1}^m \alpha_j \Phi_j(\mathbf{x}, \mathbf{c}_j), \quad (6.9)$$

where $\{\mathbf{c}_j\}_{j=1,\dots,m}$ denotes a set of m center points and $\{\alpha_j\}_{j=1,\dots,m}$ denotes a set of unknown weights to be solved for.

The reconstruction problem boils down to determine the vector $\alpha = [\alpha_1, \dots, \alpha_m]^T$, by computing the best least square approximation of the function f with respect to the constraints $\{\mathbf{p}_i\}$ (Eq. (6.8)). The minimization of E (Eq. (6.8)) implies solving the linear system of equations [100]:

$$G_{P,\Phi}^T G_{P,\Phi} \alpha = G_{P,\Phi}^T \mathbf{f}, \quad (6.10)$$

where $G_{P,\Phi}$ is a $N \times m$ matrix defined as: $G_{P,\Phi} = [\Phi(\mathbf{p}_i, \mathbf{c}_j)]_{i=1,\dots,N, j=1,\dots,m}$ and $\mathbf{f}$ is a N -elements vector defined as $\mathbf{f} = [f_1, \dots, f_N]^T$. The size of the matrix to be inverted is $m \times m$, where m is the number of centers.

The set of points P , where the function value is specified a priori (called *constraints*), includes the surface points where the function f should vanish by definition, *i.e.*, where all the f_i should be zero. To avoid the trivial solution $\alpha = \mathbf{0}$, during the minimization of E in Eq. (6.8), several interior constraints are added where the

function f does not vanish. For each additional constraint point $\mathbf{p}_k$, we assign to f a signed value f_k . This value is the approximated signed distance between $\mathbf{p}_k$ and the sampled surface. The N constraints $\{\mathbf{p}_i\}$, with $\{i = 1, \dots, N\}$ are, therefore, composed by the n input points and by the m additional off-surface constraints, where the function f is specified.

In particular, in the Samozino *et al.* algorithm [100], the additional m constraints are called *centers* and they are a subset of the poles computed using the Amenta and Kolluri approach [3]. The algorithm of Samozino *et al.* [100] proceeds as follows: given a 3D point set scattered on a surface, first its Delaunay triangulation and the dual Voronoi diagram are computed. Then the authors repeatedly refine a subset of the Voronoi vertices. In the first stage, poles are extracted from the Voronoi vertices and they are classified as inside or outside (following the Amenta *et al.* approach [3]). In the second stage, a user-defined number of m centers is selected, among the set of poles. The selection is performed by filtering and then clustering the set of poles. Poles are filtered in order to adjust the level of detail to the budget of centers and they are clustered in order to achieve a center distribution nicely spread on the medial axis.

The set of N constraints P imposes the function f to be zero at the n surface points $\{\mathbf{p}_i\}_{i=1, \dots, n}$:

$$f_i = f(\mathbf{p}_i) = 0, \quad i = 1, \dots, n, \quad (6.11)$$

and to be equal to the signed distance between the point and the sampled surface at the m centers $\{\mathbf{c}_j\}_{j=1, \dots, m}$. Therefore $N = n + m$. The distance is positive for points inside the shape and negative for points outside the shape. Since each center $\mathbf{c}_j$ is an inside pole, it has as corresponding scalar value (f_j) the radius of its corresponding circum-sphere of the Delaunay triangulation (r_i).

The reconstructed surface is required to be independent from Euclidean transformations. The function Φ is thus restricted to the set of radial functions:

$$\Phi(\mathbf{x}, \mathbf{c}_i) = \phi(\|\mathbf{x} - \mathbf{c}_i\|), \quad (6.12)$$

where $\|\cdot\|$ denotes the Euclidean distance and $\phi : \mathbb{R}^+ \rightarrow \mathbb{R}$. In order to obtain a sparse matrix G , making the computational load tractable also for a great amount of sampled surface points, compactly supported RBF are introduced by Wendland [119]. Other compactly supported RBFs (CSRBF) can be used for reconstruction [103], [126]. The support size s_i for the function ϕ_i centered on $\mathbf{c}_i$ is computed using r_i .

6.3 Surface Representation as Implicit Surface

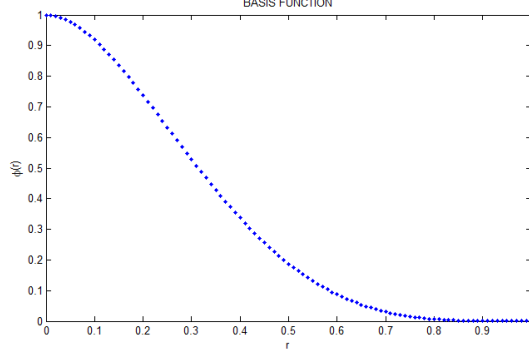


Figure 6.6: Wendland's Radial Basis Function [119].

The ϕ function, defined in Eq. (6.12), centered on $\mathbf{c}_i$, is scaled according to the local support s_i :

$$\phi_i(\|\mathbf{x} - \mathbf{c}_i\|) = \phi\left(\frac{\|\mathbf{x} - \mathbf{c}_i\|}{s_i}\right)s_i = \phi_i(r_i)s_i. \quad (6.13)$$

The basis function chosen in our implementation is (see Fig. 6.6):

$$\phi(r) = (1 - r)_+^4(1 + 4r), \quad (6.14)$$

where the symbol $+$ means:

$$\begin{cases} (x)_+ = x & \text{if } x > 0 \\ (x)_+ = 0 & \text{otherwise} \end{cases}. \quad (6.15)$$

Therefore the $N \times m$ matrix G is defined as:

$$G = \begin{bmatrix} \phi_1(\|\mathbf{p}_1 - \mathbf{c}_1\|) & \dots & \phi_m(\|\mathbf{p}_1 - \mathbf{c}_m\|) \\ \vdots & \ddots & \vdots \\ \phi_1(\|\mathbf{p}_N - \mathbf{c}_1\|) & \dots & \phi_m(\|\mathbf{p}_N - \mathbf{c}_m\|) \end{bmatrix}. \quad (6.16)$$

The use of compactly supported functions f_i leads to a sparse matrix with about 90% zero elements. Each term $e_{i,j}$ of the matrix $G^T G$ is computed as a sum:

$$e_{i,j} = \sum_{k=1}^N \phi_i(\|\mathbf{p}_k - \mathbf{c}_i\|)\phi_j(\|\mathbf{p}_k - \mathbf{c}_j\|). \quad (6.17)$$

Therefore, for each constraint $\mathbf{p}$, we need to find the list l_p of centers which contains $\mathbf{p}$ in their support. For each pair of centers $(\mathbf{c}_i, \mathbf{c}_j)$ contained in the list l_p , we add a

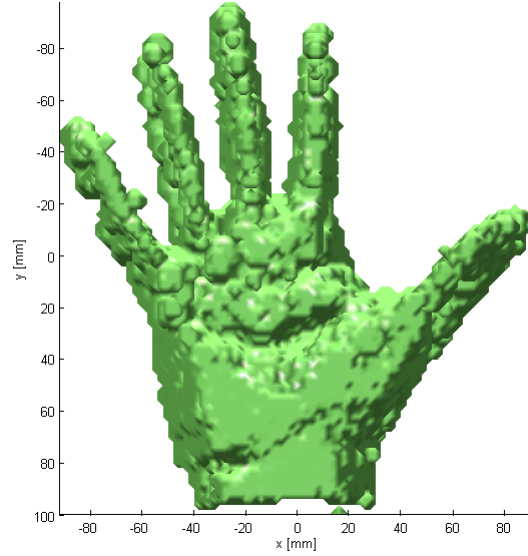


Figure 6.7: Surface representation as iso-surface of a function defined as a linear combination of Radial Basis Functions (RBF) [100].

term for $\mathbf{p}$ in the computation of $e_{i,j}$.

We implement the algorithm in Matlab and, for the purposed example ($n=56168$ surface points and $m=8338$ centers), the computation of G takes about 3 minutes, avoiding any efficient research but making it exhaustively. The obtained matrix is sparse: only 6% of its entries are different from zero. The zero level of the obtained function $f(\mathbf{x})$, computed from Eq. (6.7), is presented in Fig. 6.7 and the computation time needed for $100 \times 100 \times 100$ points is 0.8 seconds, greatly less than the time needed to represent the balls in Fig. 6.5 (several minutes) because only the iso-surface $f(\mathbf{x}) = 0$ is represented and no more all the maximal balls.

6.4 Implemented Algorithms for Surface Representation

Thinking on applications that do not require high level of accuracy or details in the reconstructed surface (think on an avatar representation and not on a reconstruction of a digital model for a mechanical artifact for example), we revisit the two described algorithms to approximate the object surface.

The Voronoi-based methods do not work well if the surface resolution is low, as in the 3D reconstructed poses with the developed multi-Kinect system. If the surface sampling is too sparse, the Voronoi cells are neither narrow, nor orthogonally

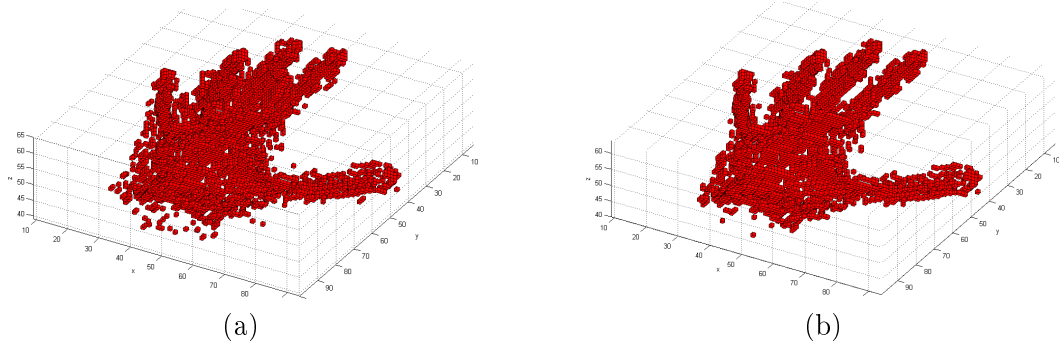


Figure 6.8: (a) Medial surface approximation with the morphological skeleton. (b) Medial surface approximation with the morphological skeleton except the voxels $\in S_0(B)$.

oriented respect to the surface [2]. Therefore, the poles selection is useless and the reconstruction fails. Considering this limitation, we still focus on the medial surface concept, but we look for a morphological approximation of it (instead of that based on the Voronoi diagram).

We approximate successfully the medial surface (MA) implementing the morphological skeletonization algorithm described in Section 4.3.3. The obtained morphological skeleton (shown in Fig. 6.8(a) for the reconstructed object at the right-bottom of Table 6.1) does not preserve the topology of the original object, as well as it is not maintained approximating the MA as a set of poles of the Voronoi diagram. The main important property is that the reconstructed surface maintains the original object connectivity and this is true both for the proposed and the Voronoi-based methods. Moreover the proposed MA approximation is applicable independently from the sampling accuracy of the available volumetric data, while Voronoi based MAT approximation methods work well only for a required sampling density [2]. Another great advantage of our method is its low computational load that makes it greatly faster than any Voronoi based method. The computation of the 4043 morphological skeleton voxels extracted with the proposed algorithm takes less than a second, greatly less than the hour taken for extracting the poles from the computed Voronoi diagram implementing the Amenta *et al.* approach [3]. Both algorithms run on a PC under Windows 7 (64 bit), using a 1.60 + 1.60 GHz Intel Dual Core CPU.

6.4.1 Union of Maximal Balls

Starting from our very fast MA approximation as the morphological skeleton, it is possible to represent the object surface as a union of balls (Maximal Balls). Each

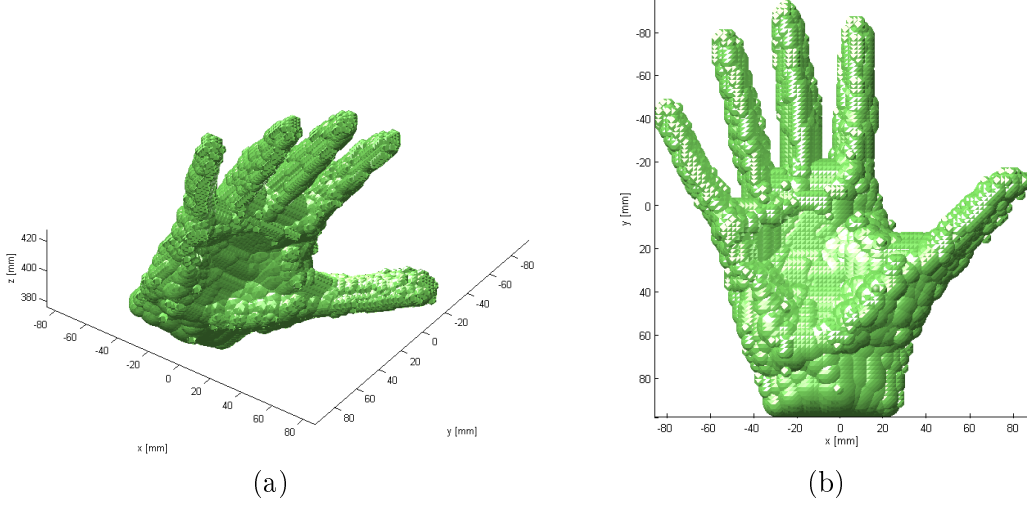


Figure 6.9: Surface approximation as Union of balls associated to the morphological skeleton (neglecting the voxels of $S_0(B)$). (a) Representation in the cartesian coordinates and (b) its projection in the xy plane.

of the $S_k(B)$ voxels can be considered as the center of a ball, with a radius that depends on the step at which it is generated. Since we use as structuring element a $3 \times 3 \times 3$ voxels block with origin at its center, to each skeleton-voxel corresponds a sphere with center located in the center of the voxel itself and with radius r equal to:

$$r = (k - \frac{1}{2})step, \quad (6.18)$$

where k indicates the step of the iterative algorithm in which the skeleton voxel is generated, and “*step*” is the 3D space sampling interval size expressed in millimeters. Fig. 6.9 shows the balls associated to the skeleton of Fig. 6.8(b) and it can be observed that the obtained surface is homotopic with the original object. In particular, we avoid the $S_0(B)$ morphological skeleton voxels because they are voxels belonging to the surface of B and do not produce any sphere. Therefore the medial surface approximation is $MA = S(B) - S_0(B)$ and it is shown in Fig. 6.8(b).

Each of these balls can be also easily interpreted in statistical terms as a Gaussian probability distribution, with mean (μ) equal to the center of the sphere and standard deviation (σ) for each direction equal to one third of the ball radius. As for a Gaussian distribution, the 99.7% of possible values are inside the interval $\mu \pm 3\sigma$, the union of the spheres can be seen as the union of the iso-surfaces at 0.997 probability of each Gaussian distribution.

Comparing the results obtained with the Amenta *et al.* [3] and the proposed rep-

representation as union of balls (Fig. 6.5(a) and Fig. 6.9(b)) we can observe that both representations are quite noisy: many balls with small radius and near to the surface are present, even if we work with noise free object in this case. More refinement in the poles choice must be done in order to obtain a more smooth and appreciable result, incrementing the computational time and the number of machine operations. Some refinement procedures can be implemented. A simple method to reduce the number of balls representing the object surface is to perform a k-means clustering over the set of all the possible centers. After the convergence of the clustering procedure, the centroid of each cluster is replaced by the closest voxel within its cluster. It guarantees that the final centers are located on the original morphological skeleton (or Voronoi diagram).

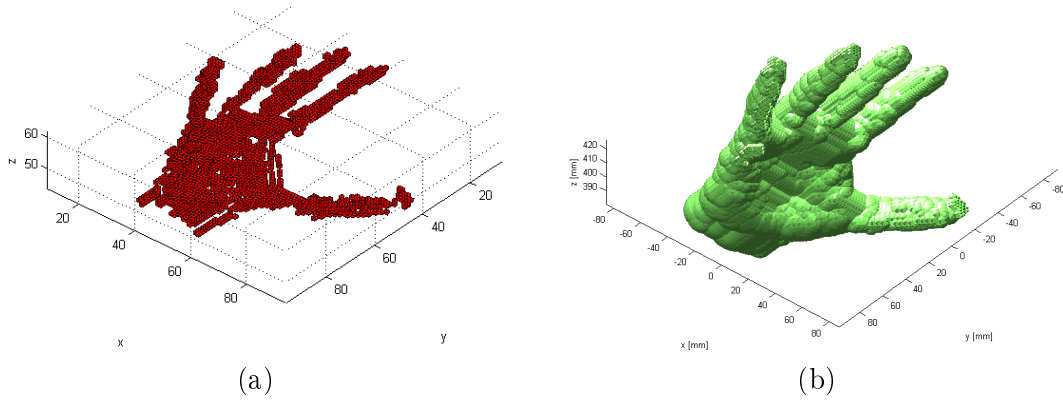


Figure 6.10: (a) Morphological skeleton filtered imposing that each block has at least 20 element 26-connected and (b) the 3074 (76% of the original) corresponding balls.

Other approaches seem more interesting in our case. In order to filter the resulting balls, we can compute the number of 26-connected voxels that form a morphological skeleton block (using the morphological algorithm described in Section 4.3.2) and cut the block with less elements than a threshold. Fig. 6.10 shows the result imposing that each block must contain at least 20 voxels that are 26-connected.

Moreover, this representation allows a hierarchical interpretation: setting the minimum acceptable radius, we can range from the most massive elements of the object representation till a more detailed description, as summarized in Fig. 6.11. In this case we have 4043 spheres with minimum radius equal to 3 mm and maximum radius equal to 17 mm, that is likely equal to the thickness of the hand palm. Fig. 6.11 shows that by increasing the minimum acceptable radius, first the fingertips are missed and then only the palm remains. Considering all the spheres with ra-

6 Surface Representation

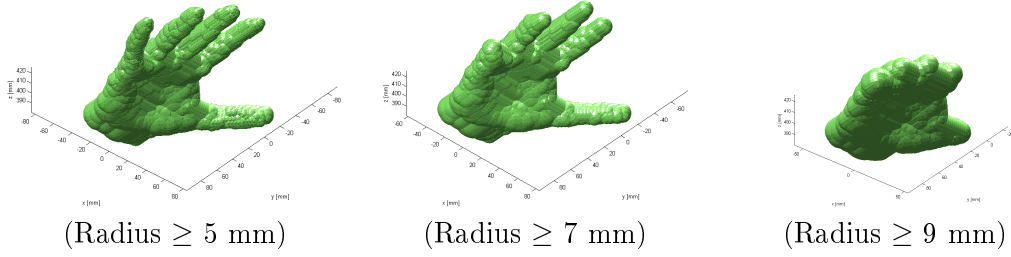


Figure 6.11: Balls filtered according to the minimum allowable radius.

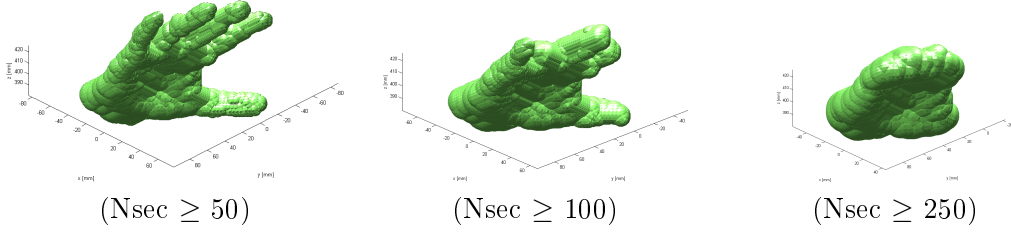


Figure 6.12: Balls filtered according to the minimum allowable number of intersecting balls.

dius greater than 5 mm, 3633 spheres remain, with radius greater than 7 mm, 2301 spheres remain, and, with radius greater than 9 mm, only 1673 spheres are left.

Another interesting way to study the hierarchy of the object parts is to consider the number of balls that intersect each skeleton ball: Fig. 6.12 represents the results setting the minimum number of intersecting spheres allowed. The spheres with more than 50 intersecting spheres are 2825, with more than 100 intersecting spheres are 2208 and with more than 250 spheres are 1455.

In order to obtain a MA that maintains the topology of the original object, a postprocessing procedure can be applied: first the number of 26-connected elements of the skeleton can be estimated and, eventually, filtrated; then they can be linked to each other following the path indicated by the linked spheres. We do not implement this algorithm because we have already obtained a surface that is homotopic with the object surface. Moreover, in Section 6.5, we illustrate the surface representation using the curve skeleton, that is homotopic with the original object.

6.4.2 Implicit Surface

Starting from the proposed approximation of the medial surface using the morphological skeleton ($MA = S(B) - S_0(B)$), also the Samozino *et al.* algorithm [100] can be reviewed and it is possible to represent the object surface as the zero level of a

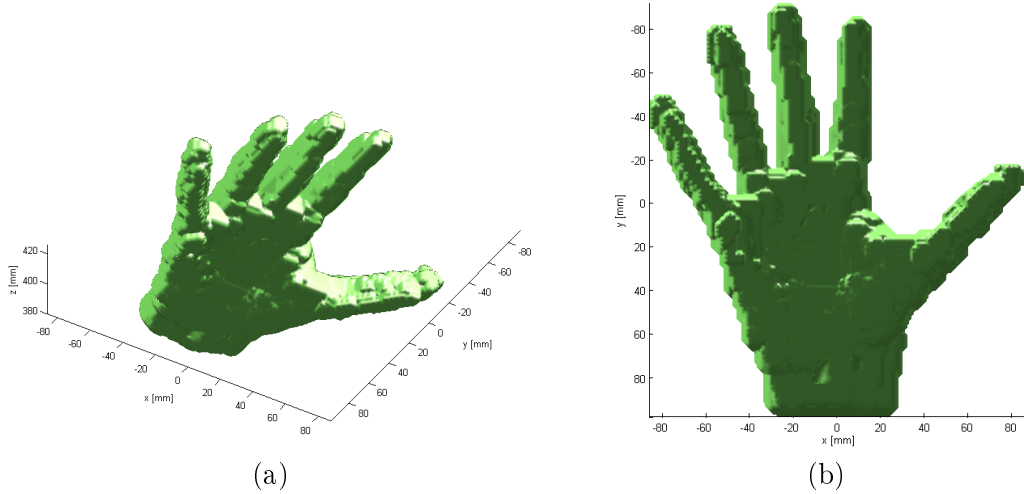


Figure 6.13: Surface representation as zero level of a function defined as linear combination of Wendland's Radial Basis Functions associated to the morphological skeleton (neglecting the voxels of $S_0(B)$). (a) Representation in the cartesian coordinates and (b) its projection in the xy plane.

function defined as linear combination of radial basis functions.

Data points are the centers of surface voxels and the m centers are the centers of the morphological skeleton voxels (neglecting the voxels resulting from the first iteration). Therefore the N constraints are the $n = 56168$ surface points plus the $m = 4043$ voxels on the approximated medial surface. We impose the function f to be zero at the surface points and to be equal to the radius of the corresponding maximal ball (r_i) at the morphological skeleton voxels.

We implement the algorithm in Matlab and, for our example, the computation of G takes 31 s, avoiding any efficient research but making it exhaustively, and the obtained matrix is greatly sparse: only 1% of its entries are different from zero. The zero level of the obtained function $f(x)$, computed from Eq. (6.7) is represented in Fig. 6.13 and the computation time needed for $100 \times 100 \times 100$ points is 1 minute, greatly less than the time needed to represent all the balls as in Fig. 6.9(b) (more than 5 minutes). In comparison to the algorithm proposed by Samozino *et al.* [100], fewer centers are used. This leads to a faster computation of the weights and a faster evaluation of the functions. The effects of some noisy balls can be seen on the tip of the thumb, for example. These balls can be removed by applying one of the previously proposed filtering methods.

The main difference with the Samozino *et al.* method is that they compute the Delaunay triangulation and the dual Voronoi diagram of the given 3D surface points.

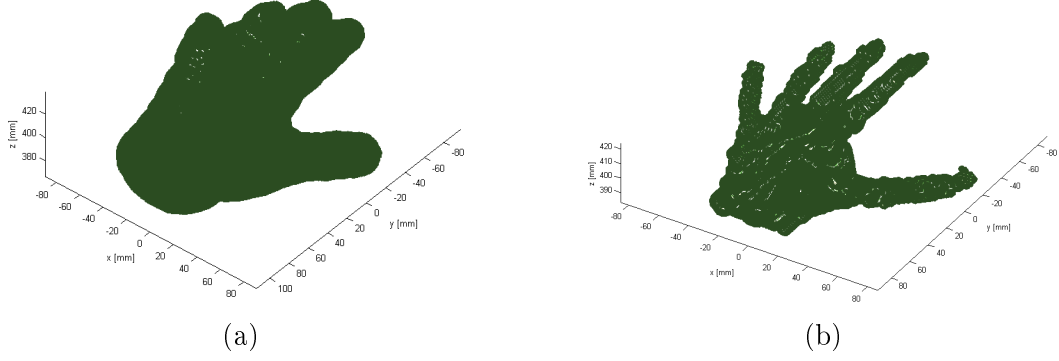


Figure 6.14: Surface representation as an isosurface of a function defined as linear combination of Radial Basis Functions with Gaussian support and standard deviation equal to (a) 0.05; (b) 0.02.

Then they repeatedly refine a subset of the Voronoi vertices. In our algorithm we apply the 3D morphological skeleton extraction algorithm and, eventually, we filter the extracted voxels.

The choice of the basis function influences the surface representation. We implement also a Gaussian RBF:

$$\phi(r) = \exp\left\{\frac{-r^2}{2\sigma^2}\right\}. \quad (6.19)$$

Considering small variances, the Gaussian RBF can be well approximated as having limited support. Fig. 6.14 represents the surface extracted using as RBF the function in Eq. (6.19) with $\sigma = \frac{1}{20} = 0.05$ (a) and $\sigma = \frac{1}{30} = 0.02$ (b). It can be seen that reducing the variance brings to a thinned surface representation because the Gaussian support is reduced as well.

6.4.3 Discussion

In this Section we develop a technique, based on morphological operators, to approximate the Medial Axis Transform in terms of surface representations as the union of Maximal Balls (Fig.6.15(c)). This method can be applied independently from the sampling accuracy of the available volumetric data, while Voronoi based MAT approximation methods work well only for a required sampling density [2]. This important feature suggests its use with the 3D pose reconstructed in our database of actions obtained with the multi-Kinect system implemented in the ISPG laboratory. We obtain a really fast and reliable method to represent human body surface for each frame. Another great advantage of our method is its low computational load that makes it greatly faster than any Voronoi based method. Moreover, applying an

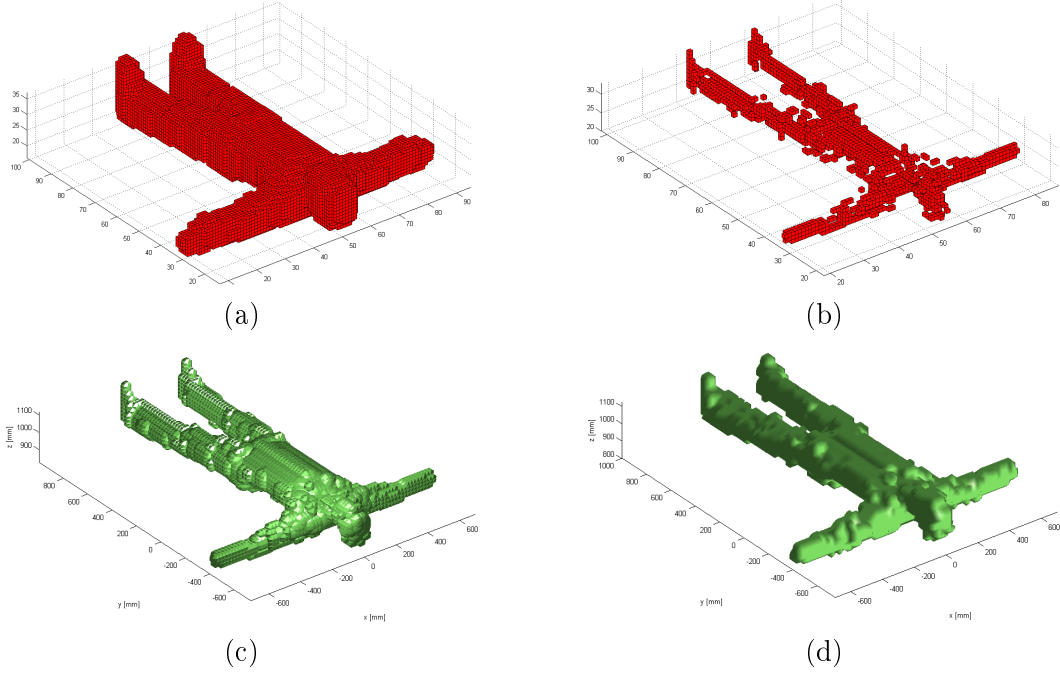


Figure 6.15: Example of actor surface representation as (c) union of balls and as (d) zero level of a function defined as linear combination of Wendland's Radial Basis Functions. Balls or RBF are associated to the (b) morphological skeleton (neglecting the voxels of $S_0(B)$, $m=1835$ voxels). (a) Representation of the voxel surface ($n=9822$ voxels).

easy postprocessing (filtering considering the number of 26-connected elements for each block, or the radius, or the number of intersecting spheres, or a combination of the previous), we are able to reduce significantly the number of balls used to represent the object and to extract its most important components (in term of the metric used). This approach allows us also to obtain a surface representation as an iso-surface of a three-dimensional function (Fig. 6.15(d)). The matrix G , for the example proposed in Fig. 6.15, has only 1.49% of non null elements. We believe that this light computational and fast algorithm is attractive for the development of domestic and user friendly applications.

In the following Section we illustrate some results obtained applying these two algorithms using the curve skeleton voxels instead of the morphological skeleton voxels.

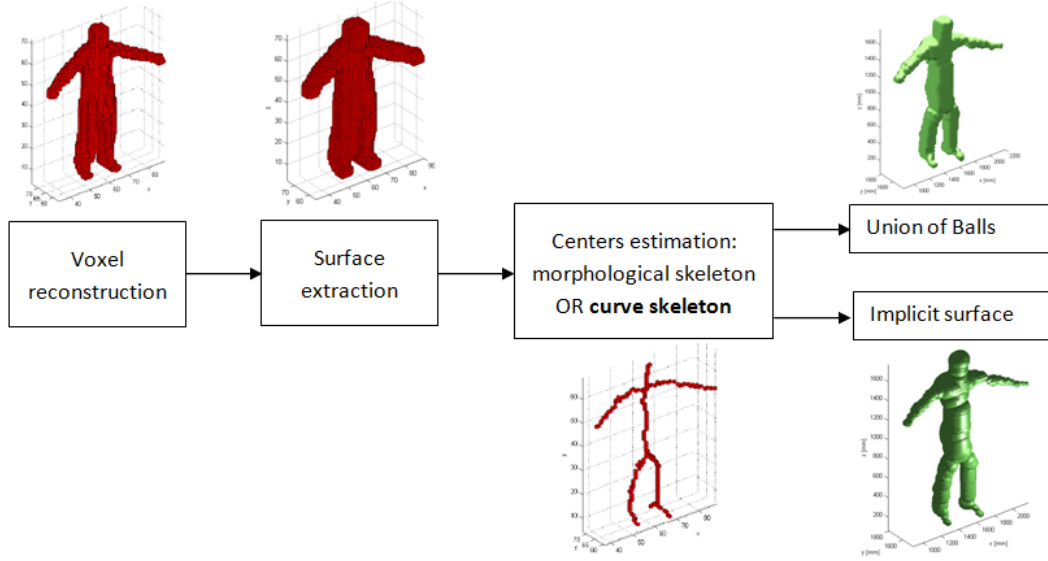


Figure 6.16: Block diagram of the implemented surface reconstruction procedure. The example shows resulting steps using the curve skeleton voxels.

6.5 Surface Reconstruction from Curve Skeleton

In this Section we show some results obtained applying the surface reconstruction algorithms, described in previous Sections, on the curve skeleton voxels. Curve skeleton provides an intrinsic structural shape description, which is useful for further applications dealing with shape understanding (like shape recognition or retrieval). Its advantage, respect to the medial shape approximation by means of the morphological skeleton, is its topological equivalence with the original object. Its disadvantage is that it approximates the curve skeleton and not the medial surface. Therefore flat parts of the original object cannot be well reconstructed from the information extracted from the curve skeleton.

The idea is to extract the object surface with the morphological surface extraction (described in Section 4.3.1), compute the curve skeleton with the proposed thinning algorithm (described in Section 5.3) and consider the centers of the skeleton voxels as the m centers of the maximal balls or of the RBFs in the implicit surface approach. The only information, that must be recovered before applying the surface reconstruction methods, is the sphere radius or the signed distance to the surface (r_i). In order to associate a radius to each medial ball, it is necessary to estimate the distance between each skeleton voxel and its nearest voxel on the surface. The

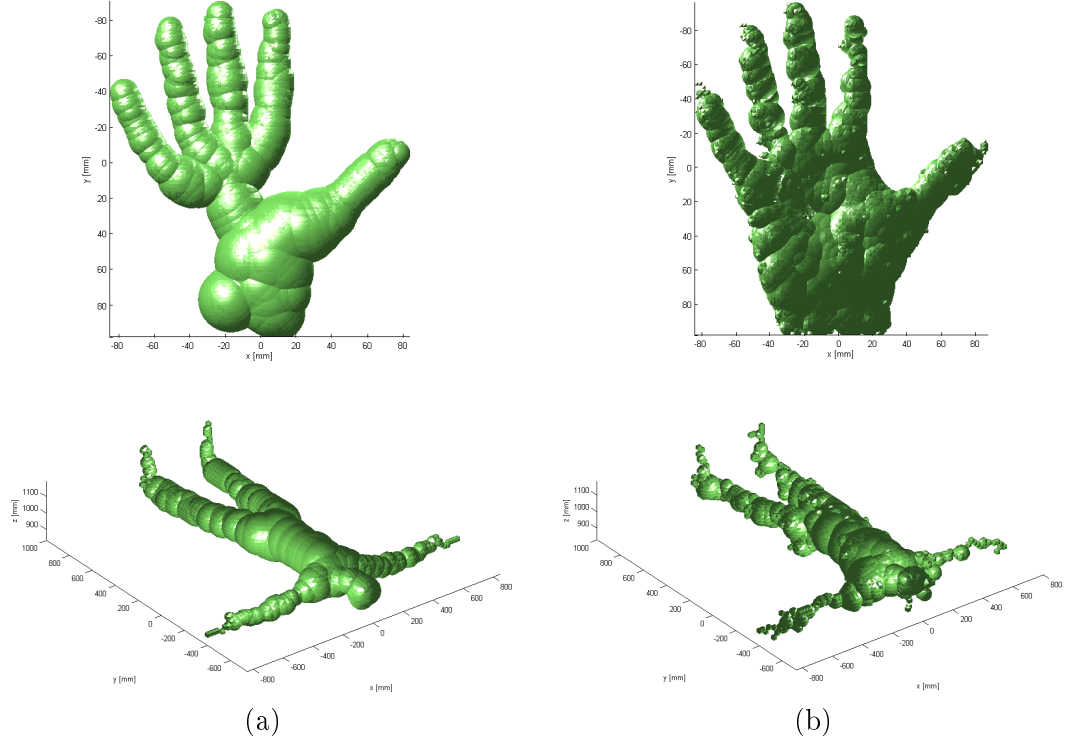


Figure 6.17: Two examples of surfaces reconstructed as union of balls from curve skeletons extracted using (a) our algorithm and (b) Palágyi *et al.* algorithm [93].

block diagram of the implemented procedure is represented in Fig. 6.16.

Fig. 6.17 shows the reconstructed surface as the union of Maximal Balls obtained starting from our curve skeleton (a) compared to the reconstructed surface obtained from the Palágyi *et al.* [93] curve skeleton (b), for the reconstructed hand shown in Table 6.1 (top row), and for the reconstructed pose shown in Fig. 4.11 (bottom row). It can be noted, above all from the example of the hand, that the Palágyi *et al.* algorithm [93], even if designed to extract the curve skeleton, having many secondary branches, looks more like an approximation of the medial surface (see Fig. 6.18). Moreover it can be noted, above all from the example of the human body, that many skeleton voxels, obtained with Palágyi *et al.* method, are really close to the surface and the reconstructed model is too thin in some parts (the skeleton is not centered in many branches as can be seen in the man arms in Fig. 6.17(b)).

Using our 3D thinning algorithm, the size of the object reconstructed is not as thick as that of the input (Fig. 6.17(a)), but shape information is mostly retained by the skeleton, since all the regions perceived as significant in the input can still

6 Surface Representation

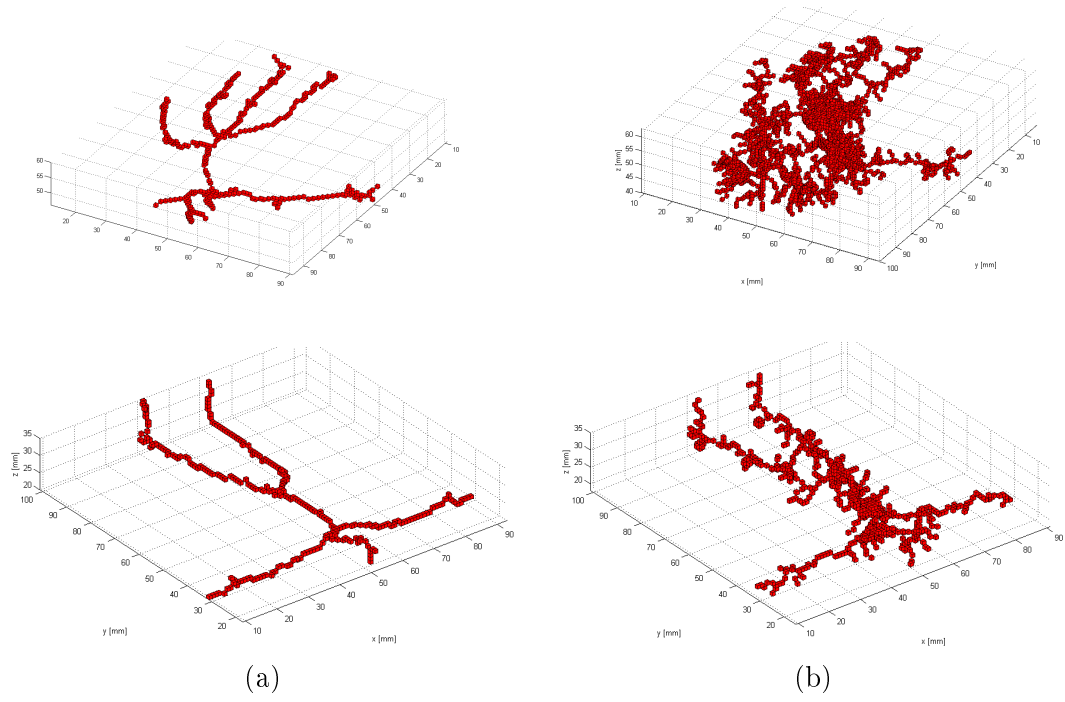


Figure 6.18: Two examples of curve skeletons extracted using (a) our algorithm and (b) Palágyi *et al.* algorithm [93].

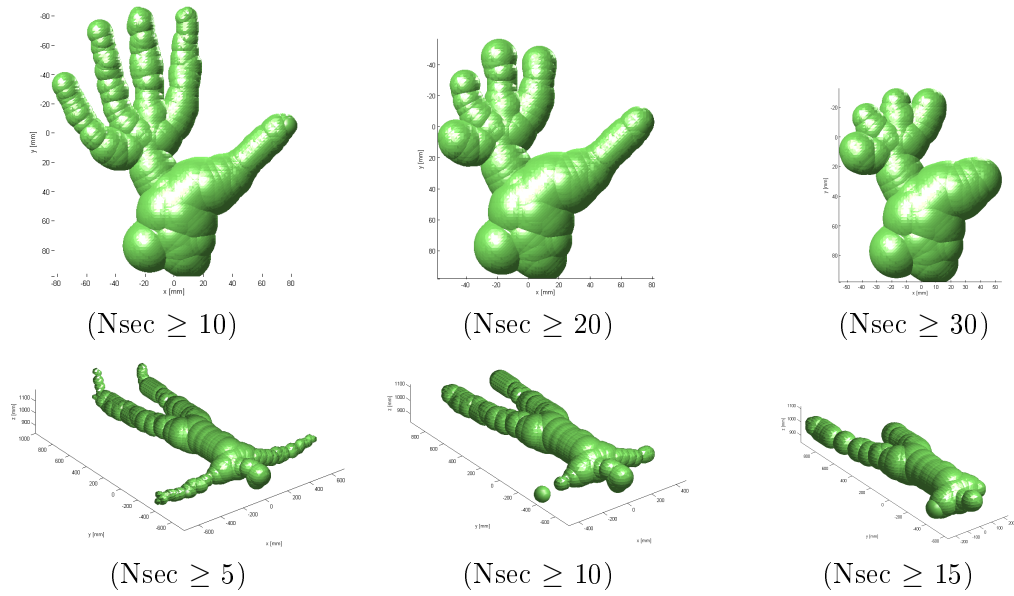


Figure 6.19: Skeleton balls filtered according to the minimum allowable number of intersecting balls.

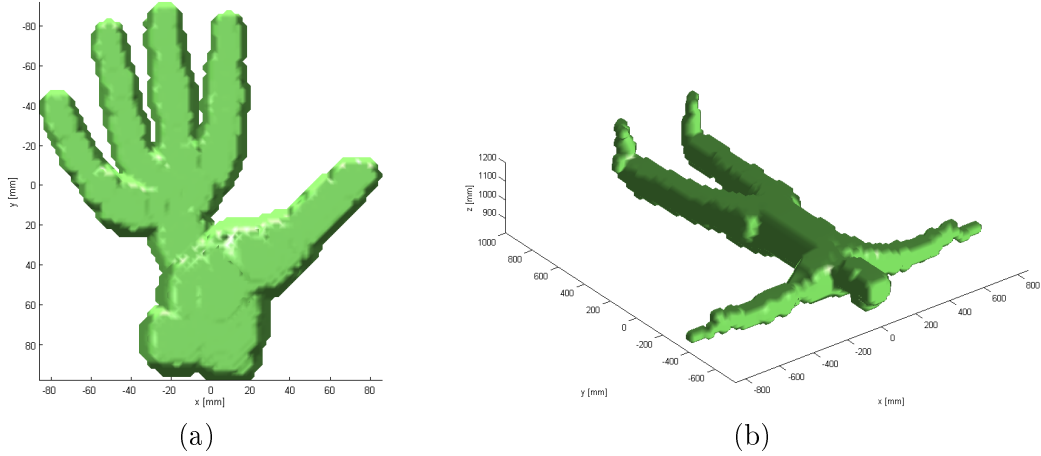


Figure 6.20: Examples of surface representation as an iso-surface of a function defined, using the curve skeleton voxels, as a linear combination of Wendland's Radial Basis Function.

be found in the reconstructed object. Using the proposed thinning algorithm, we can appreciate the smoothness of the curve skeleton (and consequently the reconstruction) obtained. Moreover we can apply the same filtering process illustrated in Section 6.4, considering the minimum allowable radius or the number of intersecting spheres, as shown in Fig. 6.19. We can range from the most massive elements till a more detailed description of the object representation, obtaining a hierarchical interpretation of the object. However, this filtering could loose the topology preserving feature because it cuts some skeleton voxels that are not simple. Lastly, the surface reconstructed as an iso-surface of a function defined as linear combination of Radial Basis Functions is shown in Fig. 6.20, using Wendland's Radial Basis Function. In particular, for the example representing a pose, considering the $n = 9822$ surface points and the $m = 252$ curve skeleton centers, the computation of the matrix G takes less than a second (only the 0.073% of its elements are different from zero), while the representation as iso-surface takes less than a minute.

7 Human Action Classification

This Chapter deals with the definition of a framework for interpreting, modeling and classifying sequences of body movements into a pre-defined vocabulary of actions. Even if in the literature there is not a clear distinction between action and gesture, we usually define a *gesture* an elementary action, “raise the right leg” or “bend the left arm” for example, regarding only one or two body parts, and an *action* a more complex movement regarding the whole body or at least its upper or lower part and composed by several gestures. Moreover we indicate as *posture* (or *pose*) a static body position, assumed by the actor in a single frame. In particular our vocabulary of actions can be considered as composed by both actions and gestures and we refer to them equally as actions or gestures.

Human motor activity and its interpretation are intensively studied by psychologists. The interpretation of a gesture and the analysis of its associated characteristics, such as the location, the dynamic path, the symbolic meaning and its emotional quality [51] are essential for those applications concerned with the recognition of what a gesturing man is doing and how. Action recognition is an important and challenging topic in Computer Vision, with many fundamental applications including video surveillance, video indexing and social sciences.

According to Neumann *et al.* [89] and from a computational perspective, actions are best defined as four-dimensional patterns in space and in time. However, many current works in gesture or action interpretation remain rooted in view-dependent representations. Video recordings of actions can similarly be defined as three-dimensional patterns in image-space and in time, resulting from the perspective projection of the world into the image plane at each time instant. Recognizing actions from a single video is however plagued with the unavoidable fact that parts of the action are hidden from the camera because of self-occlusions. That the human brain is able to recognize actions from a single viewpoint should not hide the fact that actions are firmly four-dimensional, and, furthermore, that the mental models

7 Human Action Classification

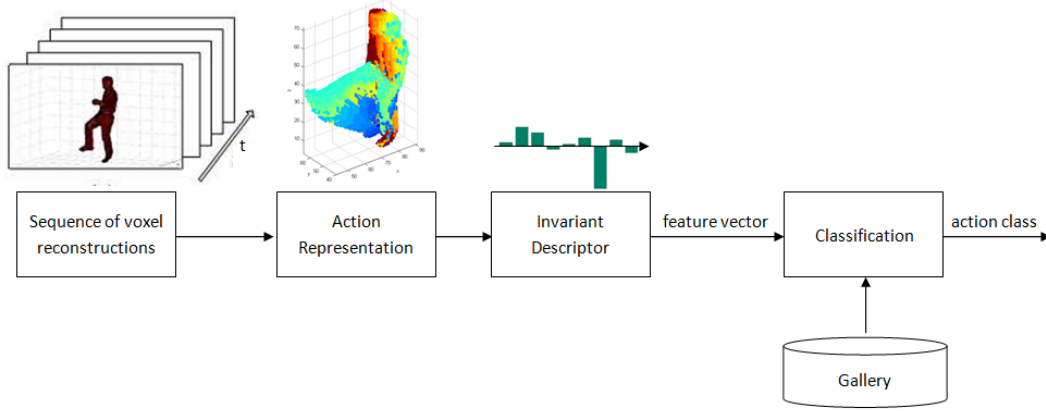


Figure 7.1: Block diagram of the classification procedure from the 3D volumetric reconstruction of the actor pose in each frame.

of actions supporting recognition may also be four-dimensional.

The main assumptions we consider in our framework, taking the Efros *et al.* table as a reference [84], are: the subject remains inside the workspace during the whole capturing process; only one subject is present in the workspace at a time; gestures can possibly involve the whole body; no occlusion occurs in the acquired scene except self occlusions; the Kinect intrinsic and extrinsic parameters are known and fixed (calibrated set); and the workspace shows a constant lighting and a static background.

In this Chapter, we investigate how to build spatio-temporal models of human actions that can support categorization and recognition of simple action classes, independently from viewpoint, actor gender and body sizes. We use our multi-Kinect acquisition system and 3D reconstructed human body occupancy. We split action recognition in three separate tasks, as summarized in Fig. 7.1. The first task is the representation of the four-dimensional patterns reconstructed from each sequence, the second task is the extraction of motion descriptors, and the third task is the classification of the descriptors into action classes.

The idea of a multi-view acquisition system for human body tracking can be found in early works, where apparent contours extracted from three orthogonal views are separately used to fit 2D deformable models, that are then applied to constrain a 3D body model [55]. Performing a volumetric reconstruction of the scene prior to the feature extraction and classification stages is in contrast with solutions that extract motion descriptors directly from multi-viewpoint sequences. Such descriptors

are not invariant to viewpoint, which can be partially solved by multiplying the number of action classes by the number of possible viewpoints [13], relative motion directions [32], and point correspondences [131]. This leads to a poorer categorization and to an increased complexity.

Using the volumetric reconstruction, problems such as viewpoint dependence and motion ambiguities are inherently solved. A voxel-based volumetric reconstruction of the human body from multi-viewpoint silhouettes extracted from the acquired videos is used for the purpose of fitting a human body ellipsoid model [114]. Even if a gesture can be defined as a time series of body joint configurations (postures), the pose estimation is not strictly necessary in order to classify actions. In this research, we follow the idea proposed by Weinland *et al.* [117]. They investigate the alternative possibility of building free-viewpoint class models from view-invariant motion descriptors. The key to their approach is the assumption that we need only to consider variations in viewpoints around the central vertical axis of the human body. Within this assumption, they propose a representation based on Fourier analysis of Motion History Volumes (MHV) in cylindrical coordinates. Our innovative contribution is to build MHV not on the original 3D volume, but on the reconstructed volume obtained after a filtering process realized with 3D morphological operators. In particular, we compute the MHV on the reconstructed volumes after the curve skeleton extraction process. This procedure has the advantage of underlying the human limbs that effectively are the central parts of the movement.

The Chapter is organized as follows. First, we recall Weinland *et al.* [117] definition of Motion History Volume as a three dimensional extension of Motion History Images proposed by Bobick and Davis [13] (Section 7.1). In Section 7.2 we present efficient descriptors for matching and aligning MHVs. Section 7.3 describes the classification procedure. Section 7.4 summarizes the proposed action classification method and discusses the obtained results testing our database of human actions. We compare the proposed method with the Weinland *et al.* classification method [117].

7.1 Action Representation

In this Section, we first recall 2D motion templates as introduced by Davis and Bobick [13] to describe temporal actions. We then propose their generalization to 3D, as defined by Weinland *et al.* [117], in order to remove the viewpoint dependence using calibrated devices.

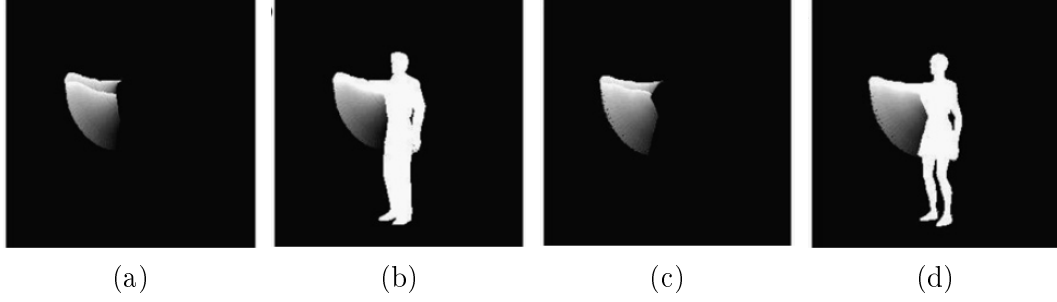


Figure 7.2: Motion versus occupancy [13]. Using motion only, in image (a) and (c), we can roughly gather that someone is lifting one arm. Using the whole silhouette instead, in (b) and (d), makes it clear that the right arm is lifted. However the same movement executed by a woman, in (c), compares favorably with the man's action in (a), whereas the whole bodies comparisons between (b) and (d) is less evident.

7.1.1 Motion History Images

Motion Energy Images (MEI) and Motion History Images (MHI) [13] are introduced to capture motion information in images. They encode, respectively, where motion occurred and the history of motion occurrences in the image. Pixel values are therefore binary values (MEI) encoding motion occurrence at a pixel, or multiple-values (MHI) encoding how recently motion occurred at a pixel. More formally, consider the binary-valued function $D(x, y, t)$, where $D = 1$ indicates that, at time t and location (x, y) , the pixel belongs to the body silhouette, then the MHI function is defined by:

$$h_{\tau}(x, y, t) = \begin{cases} \tau & \text{if } D(x, y, t) = 1 \\ \max\{0, h_{\tau}(x, y, t-1) - 1\} & \text{otherwise} \end{cases}, \quad (7.1)$$

where τ is the maximum duration stored for a motion. The associated MEI can easily be computed by thresholding $h > 0$. The above motion templates are based on occupancy because $D(x, y, t)$ is the silhouette occupancy function. Bobick and Davis also suggest to compute templates based on motion [13], *i.e.* with $D(x, y, t)$ a motion indicating function. They argue that including the complete body makes templates more robust to incidental motions that occur during an action. Occupancy provides robust cues for recognition, even if occupancy encodes not only motion but also shapes which may add difficulties when comparing movements, as illustrated in Fig. 7.2.

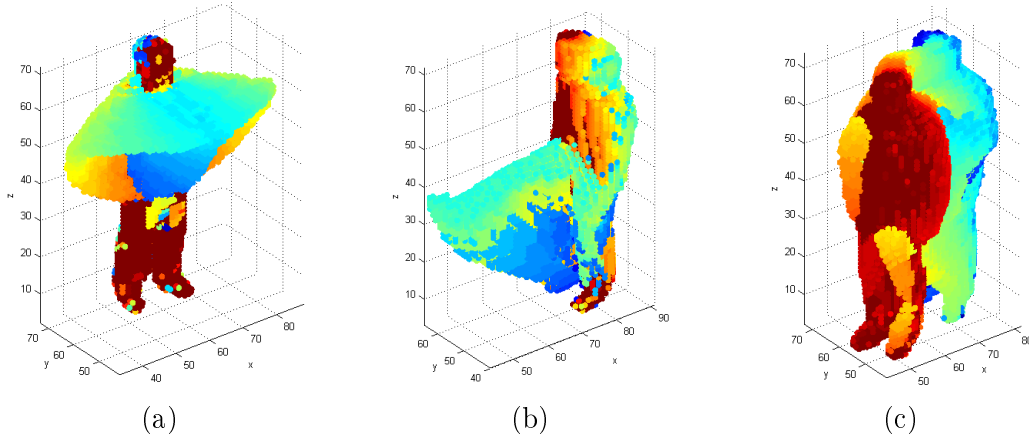


Figure 7.3: Motion History Volumes examples: (a) **Open**, (b) **Kick**, (c) **March**. Color values: red = current; ...; blue = maximum duration encode voxels occupied at the beginning of the action.

7.1.2 Motion History Volumes

The choice of a 3D representation has several advantages over a single, or multiple, 2D representation:

- A 3D representation is a natural way to fuse multiple images information. Such a representation is more informative than simple sets of 2D images, since additional calibration information is taken into account.
- A 3D representation is more robust to the object position relative to the devices, as it replaces a possibly complex matching between learned views and the actual observations by a 3D alignment (see Section 7.2).
- A 3D representation allows different device configurations.

Motion templates (MEI or MHI) extend easily to 3D by considering the occupancy function $D(x, y, z, t)$ in 3D, where $D = 1$ if (x, y, z) is occupied at time t and $D = 0$ otherwise, and by considering voxels instead of pixels [117]:

$$v_{\tau}(x, y, z, t) = \begin{cases} \tau & \text{if } D(x, y, z, t) = 1 \\ \max\{0, v_{\tau}(x, y, z, t-1) - 1\} & \text{otherwise} \end{cases} . \quad (7.2)$$

7 Human Action Classification

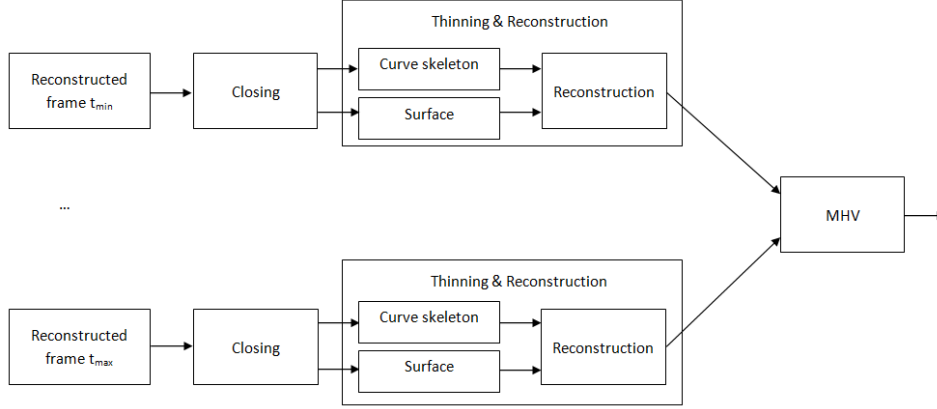


Figure 7.4: Block diagram of the action representation process from the 3D volumetric reconstruction of the actor pose in each frame to the Motion History Volume.

In the rest of the paper, we will assume templates to be normalized respect to the duration of an action:

$$v(x, y, z) = \frac{v_{\tau=t_{max}-t_{min}}(x, y, z, t_{max})}{t_{max} - t_{min}}, \quad (7.3)$$

where t_{min} and t_{max} are start and end time of an action. Hence, motion templates loose dependencies on the absolute execution speed and they result all with the same length. In this particular implementation, t_{min} and t_{max} are easily determinable as the first and last frame of each recorded sequence, while, in a general case, a motion segmentation needs to be conducted [117]. The input occupancy function $D(x, y, z, t)$ represents the reconstructed voxel set for each frame. Note however that, as for 2D motion templates, different body proportions may still result in very different templates. Fig. 7.3 shows examples for motion history volumes for the reconstructed volumes.

Actually we do not consider the original reconstructed volumes, but we apply some morphological processing on the reconstructed volume (for each frame) in order to eliminate artifacts due to reconstruction and to better highlight the body parts involved during the movement. In order to reduce artifacts in the 3D reconstruction, a closing operator is first applied (described in Section 4.2.4). It produces an output similar to dilation, but more close to the original object. After this pre-filtering, the curve skeleton is extracted and then the original volume is approximated using the voxel approximation of the ball growing approach (described in Section 5.4):

- The curve skeleton is extracted using the proposed thinning algorithm (described in Section 5.3).
- The surface is extracted with a 3D morphological dilation with a $3 \times 3 \times 3$ structuring element and then by subtracting the original 3D object to the result (as shown in Section 4.3.1).
- The distance of each skeleton voxel to the nearest point on the surface is computed and the skeleton voxel is substituted with a discretized ball with center equal to the skeleton voxel and with radius equal to the estimated distance to the surface (Section 5.4). Some examples of reconstructed 3D frames are shown in Fig. 5.21.

We refer to this procedure as *thinning&reconstruction* (T&R) and it is summarized, inside the action representation diagram, in Fig. 7.4. The MHV is computed after this *thinning&reconstruction* procedure applied to each volumetric reconstructed pose. Fig. 7.5 shows two MHVs computed without and with our procedure. It can be noted that the 3D thinning better highlights the movement (the left arm that is not involved in the movement but remains always close to the body is not reconstructed in Fig. 7.5(b)) and reduces the body shape dependence (the physical stage is less represented in Fig. 7.5(b) even if the height information is still present) incrementing the similarity between movements, even if performed by actors with different gender or different body structure.

7.2 Motion Descriptors

Our purpose is to compare body motions that are free in locations, orientations and sizes. This is not the case of motion templates, as defined in the previous Section, since they encode space occupancy. Even if 3D thinning decreases the dependence on gender and dresses, it is not sufficient to completely solve body sizes. The location and scale dependencies can be removed by centering, with respect to the center of the mass, and scale normalizing, with respect to a unit variance, motion templates, as it is usual in shape matching [117]. For the rotation, the most used tools are Hu moments (we can consider their simple 3D extensions by Sadjadi and Hall [96]) or Fourier based features. However, several works tend to show that moments are inappropriate feature descriptors, especially in the presence of noise [107]. In contrast, several works

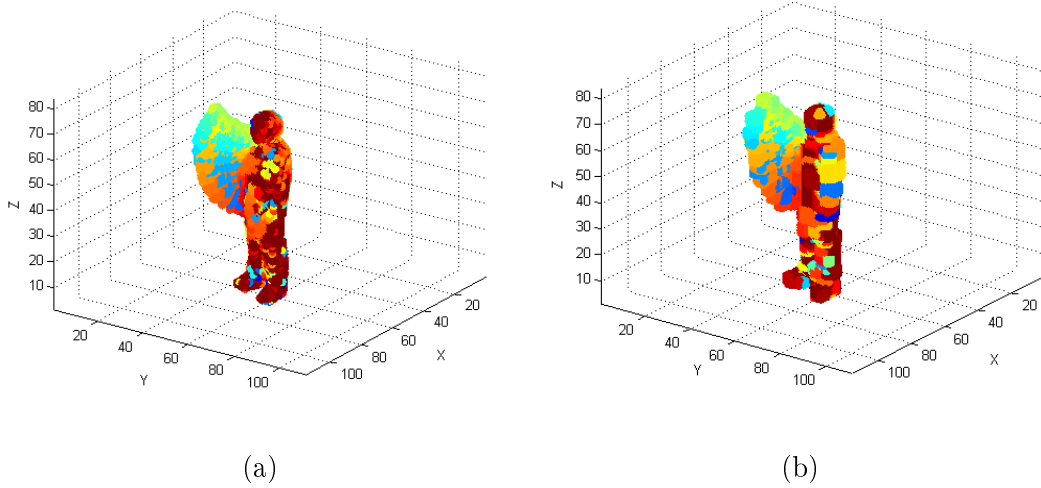


Figure 7.5: Motion History Volumes of an action `PointAt`: (a) computed directly on the voxel set, (b) computed on the reconstructed voxel set after the thinning procedure.

demonstrate better results using Fourier based features [45]. Fourier based features are robust to noise and irregularities, and present the nice property to separate coarse global and fine local features in low and high frequency components. Moreover, they can be efficiently computed using the Fast Fourier Transform (FFT). Our approach is therefore based on these features, as suggested by Weinland *et al.* [117]. Invariance of the Fourier transform follows from the Fourier *shift theorem*: a function $f_0(x)$ and its translated counterpart $f_t(x) = f_0(x - x_0)$ only differ by a phase modulation after Fourier transformation:

$$F_t(k) = F_0(k)e^{-j2\pi kx_0}. \quad (7.4)$$

The invariance property translates easily onto rotation by choosing coordinate systems that map rotations onto translations. A popular example is the Fourier-Mellin transform [20], that uses log-polar coordinates for translation, scale, and rotation invariant image registration (as we have discussed in Section 2.5.4). In a similar way, we use Fourier transform and cylindrical coordinates, centered on bodies, to express motion templates in a way invariant to locations and rotations around the z-axis. The overall choice is motivated by the assumption that similar actions only differ by rigid transformations composed of scale, translation, and rotation around the z-axis. Of course, this does not account for all the similar actions of any body, but it appears to be reasonable in most situations. Although it is possible to precisely recover

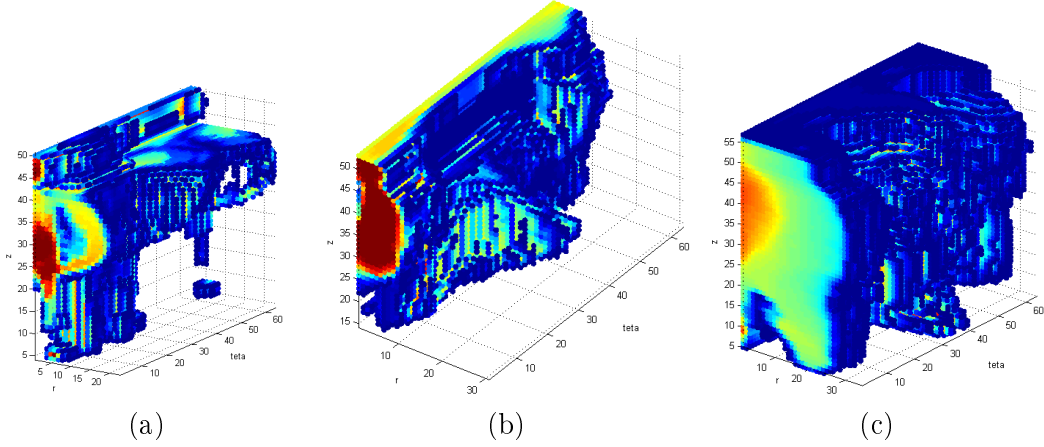


Figure 7.6: Motion History Volumes examples represented in cylindrical coordinates: (a) **Open**, (b) **Kick**, (c) **March**. Color values: red = current; ...; blue = maximum duration encode voxels occupied at the beginning of the action.

the relative orientations between history volumes using the phase or normalized correlation in Fourier space and to compare the aligned volumes directly, this almost never improves the classification results [61]. Using invariant motion descriptors is advantageous, because we do not need to align training examples for learning a class model, or to align test examples with all the class prototypes for recognition.

7.2.1 Invariant Representation

We express the motion templates in a cylindrical coordinate system (r, θ, z) :

$$v\left(\sqrt{x^2 + y^2}, \tan^{-1}\left(\frac{y}{x}\right), z\right) \rightarrow v(r, \theta, z), \quad (7.5)$$

where (x, y, z) are the coordinates of a point in the cartesian coordinate system, and (r, θ, z) are the coordinates of the same point in the cylindrical coordinate system representing respectively radius, azimuth angle, and height.

Thus, a rotation around the z -axis (of an angle equal to θ_0) results in a cyclical translation shifts:

$$v(x \cos \theta_0 + y \sin \theta_0, -x \sin \theta_0 + y \cos \theta_0, z) \rightarrow v(r, \theta + \theta_0, z). \quad (7.6)$$

We center and scale-normalize the templates. In detail, if v is the volumetric cylindrical representation of a motion template, we assume all the voxels that represent a time step, *i.e.*, for which $v(r, \theta, z) > 0$, to be part of a point cloud. We compute

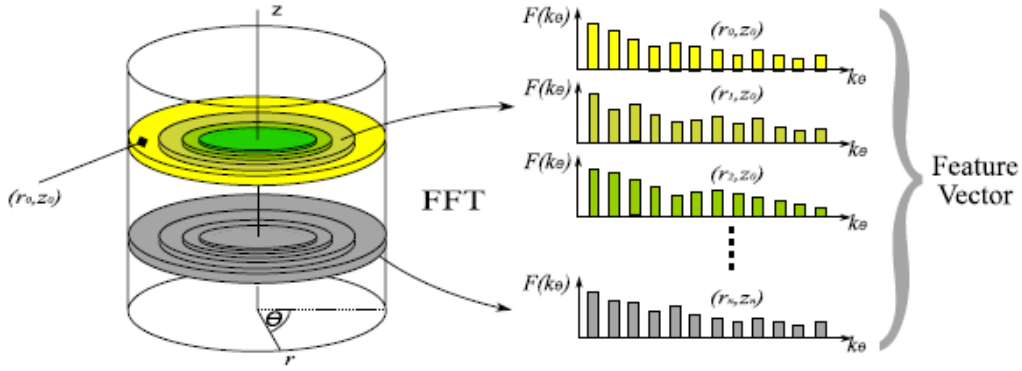


Figure 7.7: 1D-Fourier transform in cylindrical coordinates. Fourier transforms over θ are computed for couples of values (r, z) . Concatenation of the Fourier magnitudes for all the r and z forms the final feature vector.

the mean μ and standard deviations σ_r and σ_z in r - and z -directions. The template is then shifted, so that $\mu = \mathbf{0}$, and scale normalized, so that $\sigma_z = \sigma_r = 10$. Some examples of the normalized MHV represented in cylindrical coordinates are shown in Fig. 7.6. We choose to normalize in z - and r -directions, instead of a principal component based normalization, focusing on the main directions actions differ on. This method gives good results for people performing similar actions.

The absolute values $|V(r, k_\theta, z)|$ of the 1D Fourier transform:

$$V(r, k_\theta, z) = \int_{-\pi}^{\pi} v(r, \theta, z) e^{-j2\pi k_\theta \theta} d\theta, \quad (7.7)$$

for each value of r and z , are invariant to rotation along θ .

See Fig. 7.7 for an illustration of the 1D-Fourier transform. For the 1D Fourier-transform the spatial order along z and r remains unaffected. An important property of the 1D-Fourier magnitudes is its trivial ambiguity with respect to the reversal of the signal. Consequently, motions that are symmetric to the z -axis (*e.g.*, move left arm-move right arm) result in the same motion descriptors. This can be considered as a loss in information. In cases where it is important to resolve left/right ambiguities a slightly different descriptor can be used. One such descriptor is the magnitude $|V(k_r, k_\theta, k_z)|$ of the 3D-Fourier transform [118]:

$$V(k_r, k_\theta, k_z) = \int_{-\infty}^{\infty} \int_{-\pi}^{\pi} \int_{-\infty}^{\infty} v(r, \theta, z) e^{-j2\pi(k_r r + k_\theta \theta + k_z z)} dr d\theta dz, \quad (7.8)$$

applied to the motion template v . This descriptor is only symmetric with respect

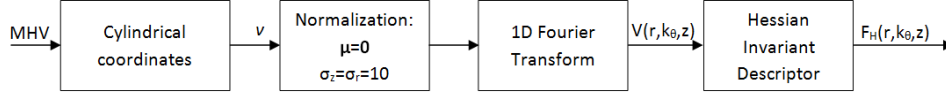


Figure 7.8: Block diagram of the computation of the motion descriptor of each sequence from the Motion History Volume till the computation of the feature vector.

to an inversion of all the variables, *i.e.*, humans standing upside-down, which does not happen very often in practice. Weinland *et al.* use both descriptors in Eq. (7.7) and Eq. (7.8) with success [118], [117]. The results are anyway inferior for those obtained with Eq. (7.8) to those obtained with Eq. (7.7).

In order to construct a feature descriptor that does not avoid completely the phase information, but performs well as Eq. (7.7), we implement a Hessian Invariant Descriptor. The idea behind the Hessian Invariant Descriptor [16] (as described in Chapter 2) is to differentiate the phase spectrum twice to eliminate the linear phase terms. The invariant parts are then the modulus of the spectrum and the second order partial derivative respect to θ of the phase spectrum:

$$F_H(r, k_\theta, z) = [|V(r, k_\theta, z)|, \varphi_{k_\theta k_\theta}(r, k_\theta, z)]. \quad (7.9)$$

The procedure implemented to obtain the motion feature vectors is summarized in Fig. 7.8.

7.3 Action Classification

In this Section we illustrate classification results using Linear Discriminant Analysis (LDA) for dimensional reduction [50], [79], combined with a minimum distance classifier using Mahalanobis distance [29]. In order to test the classification system, a gallery of N samples for training is considered. Each sample is a feature vector, output of the aforementioned procedure, representing an action sequence recorded in our database (described in Section 3.4 and available at [77]). More details about the gallery dimension and the tests performed are described in Section 7.4.

7.3.1 Dimensionality Reduction

In order to reduce the vectors dimensionality, we use the Linear Discriminant Analysis (LDA), a supervised linear feature extraction technique, that projects data

through a linear mapping that maximizes the between-class variance while minimizes the within-class variance [79]. Calling:

- C the number of classes, *i.e.* the number of actions in the database,
- N_i the number of samples analyzed for the i -th class, *i.e.* the number of sequences for each action in the training set,
- N the number of samples for training ($N = N_i C$),
- D and d the feature vectors sizes respectively before and after the reduction,
- S_W the within-class covariance matrix (which has at most rank $N - C$),
- S_B the between-class covariance matrix (which has at most rank $C - 1$),
- S_t the total covariance matrix (which has at most rank $N - 1$);

LDA seeks the vectors $\mathbf{w}$ such that maximize the Fisher criterion [79]:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmax}} \left\{ \frac{|\mathbf{w}^T S_B \mathbf{w}|}{|\mathbf{w}^T S_W \mathbf{w}|} \right\}. \quad (7.10)$$

These vectors $\mathbf{w}$ can be found by solving the eigenvalue problem:

$$S_W^{-1} S_B \mathbf{w}_i = \mathbf{w}_i \lambda_i. \quad (7.11)$$

When the size of the original feature space D is much larger than the number N of vectors available for training, the within-class scatter matrix is singular, so not invertible. To address this problem, we follow the approach proposed by Huang *et al.* [50] which belongs to the class of approaches in the null-space. The basic idea is to reduce the dimensionality working only in the null-space of the within-class scatter matrix and to diagonalize the between-class scatter matrix in that space. Specifically, the basic idea is to maximize the extended Fisher criterion [67]:

$$\hat{F}(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_B \mathbf{w} + \mathbf{w}^T S_W \mathbf{w}}. \quad (7.12)$$

Each $\mathbf{w}$ such that $\mathbf{w}^T S_W \mathbf{w} = 0$ while $\mathbf{w}^T S_B \mathbf{w} \neq 0$ maximizes the function $\hat{F}(\mathbf{w})$ [19]. These vectors belong to the null-space of S_W . However, not all the null-space of S_W is

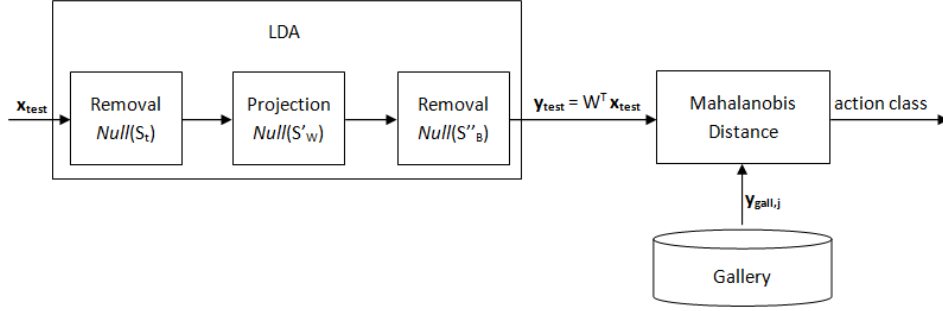


Figure 7.9: Block diagram of the classification process from the computed feature vector to the definition of the action label.

needed [67]. It is necessary to eliminate the null-space of S_t , which is the intersection between the null-space of S_W and the null-space of S_B :

$$\begin{cases} \text{if } \mathbf{w}^T S_W \mathbf{w} = 0 \quad \& \quad \mathbf{w}^T S_B \mathbf{w} \neq 0 \quad \text{keep} \\ \text{if } \mathbf{w}^T S_W \mathbf{w} = 0 \quad \& \quad \mathbf{w}^T S_B \mathbf{w} = 0 \quad \text{discard} \end{cases} \quad (7.13)$$

Then, in the complementary subspace, the null-space of the new within-class scatter matrix is computed. In this subspace the vectors that maximize the distance between different classes are estimated.

All the steps, performed in order to remove the null-space of the total scatter matrix (that is equivalent to project the vectors in the column space of S_t) and then, working in the null-space of S_W , to diagonalize the matrix S_B , are summarized below (and represented in Fig. 7.9):

- First of all the null-space of S_t must be calculated and removed solving the eigenvalue problem:

$$S_t = V S V^T \quad (7.14)$$

and taking the eigenvectors $\mathbf{v}$ corresponding to the non-zero eigenvalues:

$$U = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{N-1}]. \quad (7.15)$$

- S_W and S_B are then projected in the null space of S_t :

$$\begin{aligned} S'_W &= U^T S_W U \\ S'_B &= U^T S_B U \end{aligned} \quad (7.16)$$

7 Human Action Classification

- An orthonormal basis for the null-space of S'_W is calculated, obtaining the matrix Q , which consists of the $C - 1$ vectors such that:

$$S'_W \mathbf{q} = \mathbf{0}. \quad (7.17)$$

- Then S'_W and S'_B are projected in this subspace:

$$\begin{aligned} S''_W &= Q^T S'_W Q = 0 \\ S''_B &= Q^T S'_B Q = (UQ)^T S_B (UQ) \end{aligned} \quad (7.18)$$

- Eigenvalues and eigenvectors of S''_B are computed:

$$S''_B = Z \Delta Z^T. \quad (7.19)$$

Therefore the projection matrix is:

$$W_{D \times d} = UQZ \quad (7.20)$$

and the reduced feature vectors are:

$$\mathbf{y} = W^T \mathbf{x}. \quad (7.21)$$

This algorithm is theoretically very efficient because it works in the null-space of S_W after the discharge of the null-space of S_t , which contains no useful directions. It considerably decreases the computational cost. In addition, the total covariance matrix of the projected vectors is a diagonal matrix. This means that the transformed components are uncorrelated and, therefore, the redundant information contained in the original feature is eliminated.

7.3.2 Classification

Classification is realized using a Nearest Neighbor Classifier. Given a feature vector $\mathbf{x}_{test}$ to be classified, it is first projected in the LDA sub-space, following Eq. (7.21):

$$\mathbf{y}_{test} = W^T \mathbf{x}_{test}. \quad (7.22)$$

It is then compared with each feature vector memorized in the gallery ($\mathbf{y}_{test,j}$) and assigned to the class c of the gallery vector from which the Mahalanobis distance is minimal:

$$c = \underset{j=1,\dots,N}{\operatorname{argmin}} \{dist_j\} = \underset{j=1,\dots,N}{\operatorname{argmin}} \left\{ \sqrt{\sum_{z=1}^d \left(\frac{\mathbf{y}_{test,z}}{\sqrt{\delta_z}} - \frac{\mathbf{y}_{gall,j,z}}{\sqrt{\delta_z}} \right)^2} \right\}. \quad (7.23)$$

The Mahalanobis distance [29] is the sum of the Euclidean distance between each component, normalized by the variance δ_z . $\{\delta_z\}_{z=1,\dots,D}$ are the singular values of S_B'' . The goal is to simulate as covariance matrix an Identity matrix, resulting in an isotropic distribution of the elements. Therefore all the components are equivalent in the distance computation. Without the division for the variance, it would be most likely to find great distances in the directions in which the vectors have great variance, and small distances where the variance is lower. Considering components with equal Euclidean distance, the one with lower variance should worth proportionally more, because in that direction the data tend to be more compact. Normalizing instead, all the components tend to be equally distributed and all the directions worth equivalently.

7.4 Results

In this Section, we summarize the proposed algorithm and our implementation of the algorithm proposed by Weinland *et al.* [117] and we compare the results obtained using the database of actions developed in the ISPG laboratory (described in Section 3.4 and available at [77]). It is composed by 10 different actions, each performed 5 times by 5 actors (showing different orientations and different body structures).

The proposed algorithm can be summarized as follows:

- for each frame of the action sequence, the 3D actor volume (B) is reconstructed using the procedure described in Chapter 3;
- a closing operator (with a $3 \times 3 \times 3$ structuring element) is applied on B;
- the thinning&reconstruction procedure, as described in Section 7.1, is applied on the volumetric rendering;

7 Human Action Classification

- the normalized Motion History Volume is computed for each volumetric reconstructed action sequence (Eq. (7.3));
- the Hessian Invariant Descriptor is applied on the 1D Fourier Transform of the MHV expressed in cylindrical coordinates (Eq. (7.9));
- the resulted feature vector is projected in the subspace determined by LDA (Eq. (7.21));
- the classification is performed using a Nearest Neighbor Classifier, based on the Mahalanobis distance (Eq. (7.23)).

Our implementation of the algorithm proposed by Weinland *et al.* [117] can be summarized as follows:

- for each frame of the action sequence, the 3D actor volume (B) is reconstructed using the procedure described in Chapter 3;
- a closing operator (with a $3 \times 3 \times 3$ structuring element) is applied on B (this pre-filtering is not proposed by Weinland *et al.* [117], but we apply it in order to filter some artifacts due to the voxel reconstruction process);
- the normalized Motion History Volume is computed for each volumetric reconstructed action sequence (Eq. (7.3));
- the modulus of the 1D Fourier Transform (Eq. (7.7)) of the MHV expressed in cylindrical coordinates is computed;
- the resulted feature vector is projected in the subspace determined by Principal Component Analysis (PCA) [85];
- the classification is performed using a Nearest Neighbor Classifier, based on the Mahalanobis distance (Eq. (7.23)).

A leave-one-out cross validation is implemented, where we successively use 4 actors (1 sequence for each performer and action) to learn the actions (40 sequences in total) and the other sequences (210 in total) for testing. We would like to underly that one actor is not present in the sequences used for learning.

We compare our method to the method proposed by Weinland *et al.* [117] that uses MHV and PCA+Mahalanobis for classification. Moreover we implement a classification process that follows the proposed method neglecting the T&R procedure

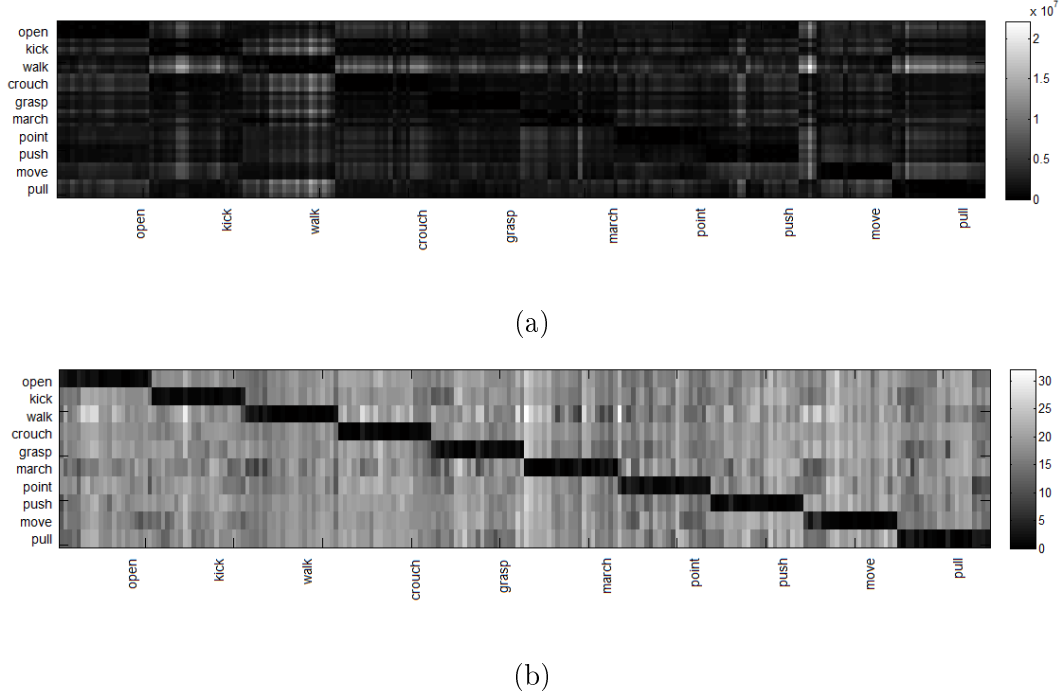


Figure 7.10: Similarity matrices: (a) using Weinland *et al.* algorithm (PCA) [117]; (b) using our algorithm (T&R+LDA).

(called briefly the LDA method): it uses LDA+Mahalanobis classifier applied on the MHV computed on the original sequence of voxel representations (extracted as described in Chapter 3). We obtain an average classification rate of 0.9190 with the Weinland *et al.* method [117] (briefly called PCA method), respect to a 0.9571 using LDA method, and to 1 obtained using the proposed method (called T&R+LDA). Detailed results are given in Table 7.1 and Fig. 7.10. Moreover with LDA the original descriptors are reduced from $D = 262144$ to $d = 9$ components (while with PCA the projected vectors have $d = 39$ components).

Even if the proposed database can be improved considering more difficult and similar motion patterns, some of the proposed actions are challenging for the classification system: **PointAt**, for example, is easily misclassified as **Move** and viceversa, because they both concern the movement only of the right arm. **Walk** is often misunderstood as **March** if the actor performs movements wider than usual. At the same time **Push** can be misclassified as **Open**, and **Crouch** as **Grasp**, but this happens only implementing the PCA method. Using a better classifier the two misunderstand-

ings are easily solved. This comparison shows, first of all, that LDA outperforms PCA, because not only reduces more the feature dimensions, but also produces a greater classification rate with less elements. In fact, Principal Component Analysis can cut some useful information. Moreover the thinning&reconstruction procedure increments the classification rate because it reduces the body shape dependence, incrementing the similarity between movements even if performed by actors with different gender or different body structure.

This increment of the performances is obtained at the cost of an increment of computational cost, mostly due to the thinning procedure and the computation of the distance between each thinned voxel and its nearest point on the surface (necessary for the reconstruction step). We do not directly use the extracted curve skeletons because the MHVs computed on them are not as useful as the corresponding MHVs computed on the reconstructed volumes. The MHVs extracted on the curve skeletons are too “thin” along the body and too “noisy” (as shown in Fig. 7.11). Therefore there are problems both to align the MHVs (often imposing $\mu = \mathbf{0}$ does not correspond to a correct alignment) and to compute the FFT because there is not the reconstruction process that smooths the volume on which the FFT is applied. Both these problems

Table 7.1: Action classification results. Results applying the PCA, LDA, and T&R + LDA methods are presented.

Action	PCA (%)	LDA (%)	T&R + LDA (%)
Open	100	100	100
Kick	100	100	100
Walk	76.19	90.48	100
Crouch	90.48	100	100
Grasp	100	100	100
March	100	90.48	100
PointAt	85.71	95.24	100
Push	95.24	100	100
MoveRt	71.43	80.95	100
Pull	100	100	100
Average Rate	91.9	95.71	100

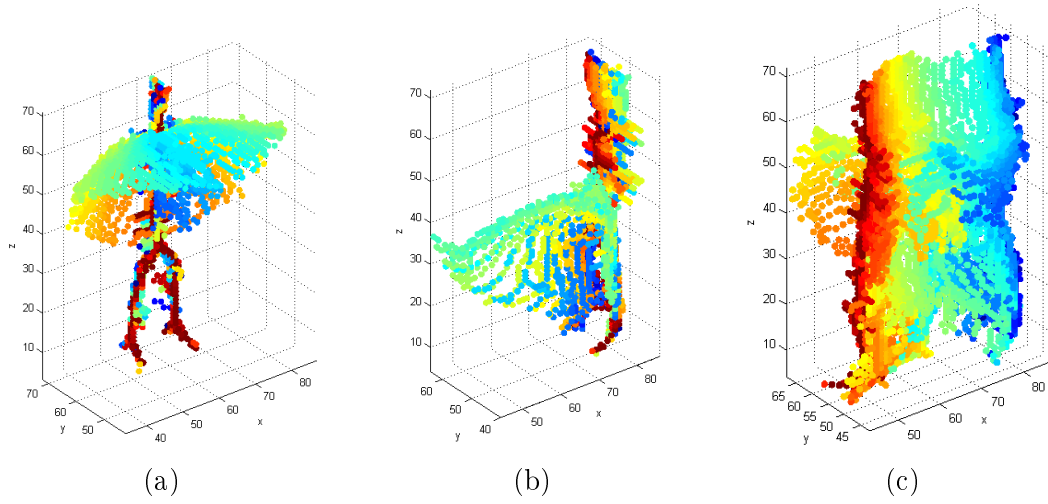


Figure 7.11: Examples of Motion History Volumes computed on the extracted curve skeletons: (a) **Open**, (b) **Kick**, (c) **March**. Color values: red = current; ...; blue = maximum duration encode voxels occupied at the beginning of the action.

conduct to feature vectors that are not distinctive for the corresponding action (the classification rate falls to the 0.82%).

8 Conclusions and future research directions

8.1 Conclusions

In this thesis we present novel solutions for the analysis, representation, and classification of human body and actions. All the proposed methodologies are developed following a morphological approach, which allows us to deal with 3D reconstructed objects without the use of an *a priori* model or any requirement on the sampling accuracy of the surface.

Starting from the 3D reconstruction of the human body as a voxel set representation, we propose some morphological techniques that make use of digital topology for reformulating standard Pattern Recognition problems, such as surface reconstruction and Medial Axis approximation. Moreover, thanks to these morphological techniques, we face the problem of action classification using a novel and promising approach. Furthermore we design an automatic human action recognition system, considering both the problems of acquisition setup (calibration and volumetric reconstruction) and of feature extraction. In the following, we recall briefly the main issues tackled and the solutions proposed.

8.1.1 Calibration

Thinking on an innovative and user friendly acquisition setup, we decide to use a net of 3 Kinect devices to capture the scene. Multiple view systems are nowadays the natural choice for indoor vision-based gesture recognizers design. Our system is viewpoint independent and motion ambiguities are avoided. Moreover, we decide to use Kinect devices because of the opportunity to combine RGB images with the information coming from depth maps.

From the analysis of many different solutions proposed in the literature, we understand that the common approaches, designed to find putative stereo-correspondences in images acquired with wide baseline, extract features that are very general and weak, since nothing is known of the underlying geometry of the acquired scene. For this reason, they are not able to distinguish if abrupt geometrical discontinuity is or not present in the point neighborhood. We propose a novel approach to define putative correspondences between images, where the information from corresponding depth maps are fruitfully integrated to reduce variability in the neighborhood around interest points. In particular, we are able to recognize corners due to variations only in the texture and not in the surface. Considering only texture corners, projective or affine distortions are reduced to similarity transforms, making available more robust and complete descriptors like Taylor or Hessian Invariant Descriptors applied to the Fourier-Mellin Transform (*i.e.* to the Fourier Transform applied on the rectified neighborhood mapped in log-polar coordinates). We also show how these descriptors, that are robust only to rotation, or rotation and scaling, provide better results with respect to state of the art descriptors like SIFT [75]. This indicates that the auxiliary information from depth map improves features matching even for wide baseline views. The resulting approach demonstrates the profitable integration between depth maps and color images to strengthen matching capabilities. The proposed descriptors hold much more information of the original image patch with respect to more general descriptors. Furthermore the computational complexity of the proposed descriptors is also very close to state of the art feature descriptors.

Starting from these results and considering that less devices can be used respect to a wide-baseline multi camera system, we implement a novel acquisition system based on color and depth images (using low cost Kinect devices).

8.1.2 Volumetric Reconstruction

Implementing a simplified acquisition setup, composed by a blue room with only a performer in the scene and no occlusions (except self-occlusions), we develop a system for the volumetric reconstruction of the actor body in the scene. The background subtraction is performed on the color images, because depth maps are too noisy to extract good silhouettes. Each silhouette is then converted in the IR camera image plane, because depth maps allow us to reconstruct the 3D scene also in its concave parts implementing a Visual Hull approach. Silhouettes extracted from color images

allow to obtain only the *Convex Hull* implementing a Visual Hull approach. Finally the projected cylinders are merged using a low cost *AND* operator.

8.1.3 Morphology

We propose a unified theory for dealing with different types of analysis problems based on a 3D reconstruction of the scene. In particular, we extend the well known morphology operators and algorithms designed for binary images to the voxel world. They can be applied independently from the tools used to obtain the 3D reconstruction, from the surface sampling accuracy, and from the object represented. The most manifold problem to deal with is the definition of the structuring elements of the 3D thinning algorithm, developed in order to obtain an approximation of the curve skeleton that is: homotopic with the original object; geometrically centered within the object boundary; as close as possible to the object shape; smooth; robust to noise on the surface; as thin as possible; and efficient. Our algorithm shows all these properties, does not depend from any parameter, and does not require any pre- or post-processing. We discuss all these aforementioned desirable features, comparing our results with one of the most cited thinning algorithm [93], and we show the improvements achieved.

8.1.4 Surface Reconstruction

3D morphological operators can be used to obtain a fast and accurate surface representation of the object surface, using a ball-growing as well as a mesh-free approach. Both the developed surface reconstruction methods are applicable using the information extracted from the morphological or the curve skeletons. Comparing our results with the Voronoi-based approach [3], the computational gain is indisputable, while the results precision remains comparable. We have also the advantage that no requirement on the reconstructed object or on its surface is imposed, but only the voxel representation is needed. Furthermore our approach is suitable for a filtering procedure that allows to extract the most important components of the object, moving towards a hierarchical representation.

8.1.5 Action Classification

Regarding the action recognizer device, having the 3D human body reconstruction available for each frame, two are the main issues to deal with: the representation of

the action pattern and the feature extraction. For the classification we implement a well known Linear Discriminant Analysis + Nearest Neighbor Classifier, based on the Mahalanobis distance, showing that the extracted features are greatly characterizing for the action performed. Moreover considering the actions globally, no time warping or frame alinement are needed (no semantic meaning are extracted).

The developed thinning&reconstruction morphological procedure allows to obtain a better representation of the human body, highlighting the movement and reducing the body shape dependence. Its application increments the similarity between movements, even if performed by actors with different gender or different body structure.

For the representation of the action pattern, we choose to use the Motion History Volume (considering the occupancy function). In order to obtain invariance under translation and body size, we consider each volume centered on the center of the mass and with standard deviation in r and z directions fixed for all the subjects. In order to obtain the invariance under rotation around the z axis, we employ the Hessian Invariant Descriptor: considering the voxel set in cylindrical coordinates, a rotation around the z axis corresponds to a shift in the θ axis. Computing the 1D Fourier Transform (only respect to the θ axis), the linear phase contribution introduced by the aforementioned rotation can be tackled considering the module and the phase second order derivative respect to that axis. The extracted feature vectors show good results. We demonstrate that, using the designed morphology operators, the performance greatly improves till 100% success in action classification.

8.2 Future Works

To conclude, we mention a set of possible evolutions of the work presented in this thesis. Some of them are already under evaluation, but they need additional experiments before being applied.

8.2.1 Segmentation in a Real Environment

In Chapter 3, we describe a silhouette extraction procedure based on the Hue channel of the color image. It generates an accurate image segmentation because we implement a simplified scenario: a room with blue curtains, blue floor and uniform illumination. In order to generalize the application field (*e. g.* domestic application, monitoring for dangerous working sites, surveillance), more advanced color

camera based background subtraction methods must be studied and implemented, considering eventually also a dynamic (and not only static) background.

8.2.2 Medial Surface Approximation

In Chapter 6, we point out that the algorithm based on the morphological skeleton does not maintain the topology of the original object, even if the surface representation is of the same homotopy type as the input surface. This is achieved using the ball growing approach. We design an algorithm to compute the elements 26-connected inside an object and to compute how many and which spheres are intersecting each skeleton ball. These two pieces of information can be combined and used to link separated object belonging to the morphological skeleton, obtaining a medial surface approximation that reflects the same topological structure of the input object and reflects better its volume occupancy respect to how it is estimable using the curve skeleton approximation. Also some a priori knowledge of the topological structure of the human body can be used to recover a medial surface approximation that preserves the topological structure and to produce a more accurate and detailed surface reconstruction.

8.2.3 Exploring a Large Data Set

In Chapter 7, we show that the action classification system obtains 100% recognition rate with our database (outperforming other implemented methods). Only with a greater database, composed by more actions performed by more actors, it is possible to build a real statistical model for action recognition system training and performance rating. We use 4 sequences for each action for training and 21 for testing, with an actor not present in the training phase. If more actors will be available, the classification performance, in terms of independence from the performer, could be tested. Experimentally, it is verified that similar actions (*e.g.* **March** and **Walk** or **PointAt** and **Move**) can be misclassified and confused. A more extended vocabulary of actions, composed by more similar actions, could better prove our algorithm, that looks foolproof for the presented database.

8.2.4 Segmenting Continuous Gesture Sequences

The sequences used to carry out experiments contain only one action, start with a standard pose, and end with the same pose. Even if the initial and final poses do

not seem a problem for the classification procedure, one of the next frontiers of this project is to make the system able to deal with sequences containing a continuous stream of different gestures. Different temporal segmentation approaches are proposed in the literature. *Temporal segmentation* consists in splitting a continuous sequence of motions into elementary segments. In particular Weinland *et al.* [117] propose an automatic procedure that uses the MHVs computed over small time windows (typically 2 – 10 frames) and that relies on the definition of motion boundaries as minima in motion energy.

8.2.5 Gesture Recognition with Additional Information

The system developed produces only a label corresponding to the action performed. No semantic information is extracted. For example, the speed of an action can be a useful information to distinguish walking from running, but it could be more interesting to conduct a behavioral analysis of the action: to recover if the movement is performed impetuously. The execution speed is a variable easily computable, the interpretation of a gesture, instead, is more difficult. Some researches deal with *Fuzzy Logic* [132] in order to achieve this issue, but a deeper analysis of the human behaviors and how to represent them is still necessary.

8.2.6 Dealing with Multiple Actors and/or Occlusions

The system implemented is able to deal only with sequences involving only one actor at a time and no other object in the scene. A more realistic extension of this system will be the possibility of recognizing multiple actors in a real scene (with any kind of furniture present) and classifying the action performed by each of them separately. If the actors act separately and no object occlusions are present, the problem will be easily solved, because each actor is represented by its own voxel set. Problems could arise when the actors interact touching each other and touching other objects. In this case a segmentation is necessary and when occlusion happens, probably the use of a body model for the estimation of each actor pose in each frame would be required. On our database we have already worked on a tracking system that uses a body model in order to recover the actor pose in each frame and it can be useful in such kind of situations to merge both the approaches. However, also the aforementioned tracking system is tested only on sequences with a single actor in the scene. A new database of actions involving multiple actors and objects needs to be developed.

Bibliography

- [1] N. Ahmed. A system for 360: Acquisition and 3d animation reconstruction using multiple rgb-d cameras. 2012.
- [2] N. Amenta, S. Choi, and R.K. Kolluri. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266, 2001.
- [3] N. Amenta and R.K. Kolluri. Accurate and efficient unions of balls. In *Proceedings of the sixteenth annual symposium on Computational geometry*, pages 119–128, 2000.
- [4] C. Arcelli, G. Sanniti di Baja, and L. Serino. A parallel algorithm to skeletonize the distance transform of 3d objects. *Image and Vision Computing*, 27(6):666–672, 2009.
- [5] D. Attali, J.D. Boissonnat, and H. Edelsbrunner. Stability and computation of medial axes-a state-of-the-art report. *Mathematical foundations of scientific visualization, computer graphics, and massive data exploration*, pages 109–125, 2009.
- [6] D. Attali and J.O. Lachaud. Delaunay conforming iso-surface, skeleton extraction and noise removal. *Computational Geometry*, 19(2-3):175–189, 2001.
- [7] G. Avgerakis. *Digital animation bible*, volume 1. McGraw-Hill/TAB Electronics, 2003.
- [8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
- [9] G. Bertrand. Simple points, topological numbers and geodesic neighborhoods in cubic grids. *Pattern recognition letters*, 15(10):1003–1011, 1994.

- [10] S. Biasotti, S. Marini, M. Mortara, G. Patane, M. Spagnuolo, and B. Falcidieno. 3d shape matching through topological structures. In *Discrete Geometry for Computer Imagery*, pages 194–203, 2003.
- [11] H. Blum et al. A transformation for extracting new descriptors of shape. *Models for the perception of speech and visual form*, 19(5):362–380, 1967.
- [12] H. Blum and R.N. Nagel. Shape description using weighted symmetric axis features. *Pattern recognition*, 10(3):167–180, 1978.
- [13] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, 2001.
- [14] G. Borgefors. On digital distance transforms in three dimensions. *Computer Vision and Image Understanding*, 64(3):368–376, 1996.
- [15] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter. The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2):1–13, 2011.
- [16] R.D. Brandt and F. Lin. Representations that uniquely characterize images modulo translation, rotation, and scaling. *Pattern Recognition Letters*, 17(9):1001–1015, 1996.
- [17] G. Brostow, I. Essa, D. Steedly, and V. Kwatra. Novel skeletal representation for articulated creatures. *Computer Vision-ECCV 2004*, pages 66–78, 2004.
- [18] F. Cazals and J. Giesen. Delaunay triangulation based surface reconstruction. *Effective Computational Geometry for Curves and Surfaces*, pages 231–276, 2006.
- [19] L.F. Chen, H.Y.M. Liao, M.T. Ko, J.C. Lin, and G.J. Yu. A new lda-based face recognition system which can solve the small sample size problem. *Pattern recognition*, 33(10):1713–1726, 2000.
- [20] Q. Chen, M. Defrise, and F. Deconinck. Symmetric phase-only matched filtering of fourier-mellin transforms for image registration and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(12):1156–1168, 1994.

- [21] I. Cohen and H. Li. Inference of human postures by classification of 3d human body shape. In *Analysis and Modeling of Faces and Gestures, 2003. AMFG 2003. IEEE International Workshop on*, pages 74–81, 2003.
- [22] N.D. Cornea, D. Silver, and P. Min. Curve-skeleton applications. In *Visualization, 2005. VIS 05. IEEE*, pages 95–102, 2005.
- [23] N.D. Cornea, D. Silver, and P. Min. Curve-skeleton properties, applications, and algorithms. *Visualization and Computer Graphics, IEEE Transactions on*, 13(3):530–548, 2007.
- [24] N.D. Cornea, D. Silver, X. Yuan, and R. Balasubramanian. Computing hierarchical curve-skeletons of 3d objects. *The Visual Computer*, 21(11):945–955, 2005.
- [25] M. Couprie and R. Zour. Discrete bisector function and euclidean skeleton. In *Discrete Geometry for Computer Imagery*, pages 216–227, 2005.
- [26] F. Cuzzolin, A. Sarti, and S. Tubaro. Invariant action classification with volumetric data. In *Multimedia Signal Processing, 2004 IEEE 6th Workshop on*, pages 395–398, 2004.
- [27] P. Daras, D. Zarpalas, D. Tzovaras, and M.G. Strintzis. Shape matching using the 3d radon transform. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 953–960, 2004.
- [28] L. De Floriani and M. Spagnuolo. *Shape analysis and structuring*. Springer Verlag, 2008.
- [29] R. De Maesschalck, D. Jouan-Rimbaud, and DL Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18, 2000.
- [30] T.K. Dey and J. Sun. Defining and computing curve-skeletons with medial geodesic function. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 143–152, 2006.
- [31] T.K. Dey and W. Zhao. Approximate medial axis as a voronoi subcomplex. *Computer-Aided Design*, 36(2):195–202, 2004.

- [32] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 726–733, 2003.
- [33] M. Foskey, M.C. Lin, and D. Manocha. Efficient computation of a simplified medial axis. *Journal of Computing and Information Science in Engineering*, 3:274, 2003.
- [34] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto. Improving features tracking with robust statistics. *Pattern Analysis and Applications*, 2:312–320, 1999.
- [35] N. Gagvani and D. Silver. Parameter-controlled volume thinning. *Graphical Models and Image Processing*, 61(3):149–164, 1999.
- [36] S.B. Gokturk, H. Yalcin, and C. Bamji. A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions. In *Proc. of CVPRW 2004*, volume 3, page 35, June 27 – July 2, 2004.
- [37] P. Golland, W. Eric, and L. Grimson. Fixed topology skeletons. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 10–17, 2000.
- [38] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice Hall, second edition, 2002.
- [39] R.W. Hall. Parallel connectivity-preserving thinning algorithms. *Machine Intelligence and Pattern Recognition*, 19:145–179, 1996.
- [40] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 50–60. Manchester, UK, 1988.
- [41] R.I. Hartley. In defense of the eight-point algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(6):580–593, 1997.
- [42] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [43] M.S. Hassouna and A.A. Farag. Robust centerline extraction framework using level sets. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 458–465, 2005.

- [44] J.M. Hebert. Lightwave 3d user guide. *NewTek*, 2011.
- [45] D. Heesch and S. Rüger. Combining features for content-based sketch retrieval - a comparative evaluation of retrieval performance. *Advances in Information Retrieval*, pages 357–360, 2002.
- [46] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112, 1997.
- [47] C. Herrera, J. Kannala, et al. Joint depth and color camera calibration with distortion correction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(10):2058–2064, 2012.
- [48] C. Herrera, J. Kannala, and J. Heikkilä. Accurate and practical calibration of a depth and color camera pair. In *Computer Analysis of Images and Patterns*, pages 437–445, 2011.
- [49] A. Hilton and J. Starck. Multiple view reconstruction of people. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 357–364, 2004.
- [50] R. Huang, Q. Liu, H. Lu, and S. Ma. Solving the small sample size problem of lda. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 29–32, 2002.
- [51] C. Hummels and P.J. Stappers. Meaningful gestures for human computer interaction: beyond hand postures. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 591–596, 1998.
- [52] IHS iSuppli. The teardown: The kinect for xbox 360. *IET Engineering Technology*, 6(3):94–95, April 2011.
- [53] B. Javidi. *Image recognition and classification: algorithms, systems, and applications*, volume 78. CRC Press, 2002.
- [54] I.T. Jolliffe. *Principal Component Analysis*, volume XXIX. Springer, NY, 2nd ed. edition, 2002.

- [55] I.A. Kakadiaris and D. Metaxas. 3d human body model acquisition from multiple views. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 618–623, 1995.
- [56] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 954–961, 2003.
- [57] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, 2006.
- [58] R. Kolluri, J.R. Shewchuk, and J.F. O’Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 11–21, 2004.
- [59] T.Y. Kong and A. Rosenfeld. Digital topology: introduction and survey. *Computer Vision, Graphics, and Image Processing*, 48(3):357–393, 1989.
- [60] K. Koser and R. Koch. Perspectively invariant normal features. In *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.
- [61] C.D. Kuglin and D.C. Hines. The phase correlation image alignment method. *IEEE International Conference on Cybernetics and Society*, pages 163–165, 2006.
- [62] A. Laurentini. The visual hull concept for silhouette-based image understanding. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(2):150–162, 1994.
- [63] F.F. Leymarie. Three-dimensional shape representation via shock flows. In *BROWN UNIVERSITY*, 2003.
- [64] T. Leyvand, C. Meekhof, Y.C. Wei, J. Sun, and B. Guo. Kinect identity: Technology and experience. *Computer*, 44(4):94–96, April 2011.
- [65] J. Li and Q. Tian. 3d modeling skeleton extraction: A review. In *Computer Application and System Modeling (ICCA SM), 2010 International Conference on*, volume 8, pages V8–434, 2010.
- [66] A. Lieutier. Any open bounded subset of $\mathbb{R}^n$ has the same homotopy type as its medial axis. *Computer-Aided Design*, 36(11):1029–1046, 2004.

- [67] K. Liu, Y.Q. Cheng, J.Y. Yang, and X. Liu. An efficient algorithm for foley-sammon optimal set of discriminant vectors by algebraic method. *IJPRAI*, 6(5):817–829, 1992.
- [68] C. Lohou. Detection of the non-topology preservation of ma’s 3d surface-thinning algorithm, by the use of simple points. *Pattern Recognition Letters*, 29(6):822–827, 2008.
- [69] C. Lohou and G. Bertrand. Two symmetrical thinning algorithms for 3d binary images, based on p-simple points. *Pattern recognition*, 40(8):2301–2314, 2007.
- [70] D. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [71] W.C. Ma, F.C. Wu, and M. Ouhyoung. Skeleton extraction of 3d objects with radial basis functions. In *Shape Modeling International, 2003*, pages 207–215, 2003.
- [72] G. Malandain and G. Bertrand. Fast characterization of 3d simple points. In *Pattern Recognition, 1992. Vol. III. Conference C: Image, Speech and Signal Analysis, Proceedings., 11th IAPR International Conference on*, pages 232–235, 1992.
- [73] A. Manzanera, T. Bernard, F. Prêteux, and B. Longuet. Ultra-fast skeleton based on an isotropic fully parallel algorithm. In *Discrete Geometry for Computer Imagery*, pages 313–324, 1999.
- [74] M. Marcon, E. Frigerio, A. Sarti, and S. Tubaro. 3d correspondences in textured depth-maps through planar similarity transform. In *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on*, pages 17–20, 2012.
- [75] M. Marcon, E. Frigerio, A. Sarti, and S. Tubaro. 3d wide baseline correspondences using depth-maps. *Signal Processing: Image Communication*, 2012.
- [76] M. Marcon, L. Picarreta, A. Sarti, and S. Tubaro. A fast level-set approach to surface modeling from unorganized sample points. In *Proceedings of 2nd IEEE Workshop on Variational and Level Set Methods in Computer Vision (VLSM’03), vol. a*, pages 191–196, 2003.

- [77] M. Marcon, A. Sarti, and S. Tubaro. <http://www-dsp.elet.polimi.it/ispg/index.php/description.html>, 2010.
- [78] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767, 2004. British Machine Vision Computing 2002.
- [79] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop.*, pages 41–48, 1999.
- [80] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, pages 1615–1630, 2005.
- [81] S. Milani. MS Kinect Denoising Test images - <http://www.dei.unipd.it/~sim1mil/materiale/kinect>, 2011.
- [82] S. Milani, E. Frigerio, M. Marcon, and S. Tubaro. Denoising infrared structured light dibr signals using 3d morphological operators. In *3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2012*, pages 1–4, 2012.
- [83] F. Mindru, T. Tuytelaars, L. Van Gool, and T. Moons. Moment invariants for recognition under changing viewpoint and illumination. *Computer Vision and Image Understanding*, 94(1):3–27, 2004.
- [84] T.B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.
- [85] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *Automatic Control, IEEE Transactions on*, 26(1):17–32, 1981.
- [86] M. Mortara and G. Patane. Shape-covering for skeleton extraction. *International Journal of Shape Modeling*, 8(2):139–158, 2002.

- [87] R. Mukundan and K.R. Ramakrishnan. *Moment functions in image analysis: theory and applications*. World Scientific Pub Co Inc, 1998.
- [88] J.A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.
- [89] J. Neumann, C. Fermüller, and Y. Aloimonos. Animated heads: From 3d motion fields to action descriptions. In *Proceedings of the IFIP TC5/WG5*, volume 10, pages 1–11, 2001.
- [90] N. Nikolaidis and I. Pitas. *3-D image processing algorithms*. John Wiley & Sons, Inc., 2000.
- [91] University of Maryland at College Park. Computer Science Center and DG Morgenthaler. *Three-dimensional simple points: serial erosion, parallel thinning, and skeletonization*. 1981.
- [92] K. Palágyi. A 3-subiteration 3d thinning algorithm for extracting medial surfaces. *Pattern Recognition Letters*, 23(6):663–675, 2002.
- [93] K. Palágyi and A. Kuba. A 3d 6-subiteration thinning algorithm for extracting medial lines. *Pattern Recognition Letters*, 19(7):613–627, 1998.
- [94] K. Palágyi and A. Kuba. A parallel 3d 12-subiteration thinning algorithm. *Graphical Models and Image Processing*, 61(4):199–221, 1999.
- [95] M. Pierobon, M. Marcon, A. Sarti, and S. Tubaro. Classification of human body actions by invariant body shape descriptor. *Citeseer*, 2008.
- [96] F.A. Sadjadi and E.L. Hall. Three-dimensional moment invariants. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):127–136, 1980.
- [97] P.K. Saha and B.B. Chaudhuri. Detection of 3-d simple points for topology preserving transformations with application to thinning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(10):1028–1032, 1994.
- [98] P.K. Saha and B.B. Chaudhuri. 3d digital topology under binary transformation with applications. *Computer vision and image understanding*, 63(3):418–429, 1996.

- [99] T. Saito and J.I. Toriwaki. New algorithms for euclidean distance transformation of an n-dimensional digitized picture with applications. *Pattern recognition*, 27(11):1551–1565, 1994.
- [100] M. Samozino, M. Alexa, P. Alliez, and M. Yvinec. Reconstruction with voronoi centered radial basis functions. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 51–60, 2006.
- [101] G. Sansoni, M. Trebeschi, and F. Docchio. State-of-the-art and applications of 3d imaging sensors in industry, cultural heritage, medicine, and criminal investigation. *Sensors*, 9(1):568–601, 2009.
- [102] A. Sarti and S. Tubaro. Image-based surface modeling: a multi-resolution approach. *Signal processing*, 82(9):1215–1232, 2002.
- [103] R. Schaback. Creating surfaces from scattered data using radial basis functions. *Mathematical methods for curves and surfaces*, pages 477–496, 1995.
- [104] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.
- [105] S. Se, H.K. Ng, P. Jasiobedzki, and T.J. Moyung. Vision based modeling and localization for planetary exploration rovers. In *Proceedings of International Astronautical Congress*, 2004.
- [106] J. Serra. *Image analysis and mathematical morphology*. London.: Academic Press.[Review by Fensen, EB in: J. Microsc. 131 (1983) 258.] Cell size, Staining Microscopy Technique, Mathematics, General article Review article (PMBD, 185707888), 1982.
- [107] D. Shen and H.H.S. Ip. Discriminative wavelet shape descriptors for recognition of 2-d patterns. *Pattern recognition*, 32(2):151–166, 1999.
- [108] K. Siddiqi and S.M. Pizer. *Medial representations: mathematics, algorithms and applications*, volume 37. Springer Verlag, 2008.
- [109] S. Svensson, C. Arcelli, and G. Sanniti di Baja. Finding cavities and tunnels in 3d complex objects. In *Image Analysis and Processing, 2003. Proceedings. 12th International Conference on*, pages 342–347, 2003.

- [110] A. Telea and A. Vilanova. A robust level-set algorithm for centerline extraction. In *Proceedings of the symposium on Data visualisation 2003*, pages 185–194, 2003.
- [111] C. Theobalt, J. Carranza, M.A. Magnor, and H.P. Seidel. Combining 3d flow fields with silhouette-based human motion capture for immersive video. *Graphical Models*, 66(6):333–351, 2004.
- [112] P.H.S. Torr and D.W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24(3):271–300, 1997.
- [113] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment: a modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000.
- [114] M.M. Trivedi, K.S. Huang, and I. Mikic. Dynamic context capture and distributed video arrays for intelligent spaces. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):145–163, 2005.
- [115] G. Turk and J.F. O’Brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics (TOG)*, 21(4):855–873, 2002.
- [116] R. Unnikrishnan and M. Hebert. Fast extrinsic calibration of a laser rangefinder to a camera. *Carnegie Mellon University*, 2005.
- [117] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006.
- [118] D. Weinland, R. Ronfard, E. Boyer, et al. Motion history volumes for free viewpoint action recognition. In *Workshop on modeling People and Human Interaction (PHI’05)*, 2005.
- [119] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, 4(1):389–396, 1995.
- [120] H. Wendland. *Scattered data approximation*, volume 17. Cambridge University Press, 2005.

- [121] R.T. Whitaker. A level-set approach to 3d reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231, 1998.
- [122] A.D. Wilson. Using a depth camera as a touch sensor. In *Proc. of ACM ITS 2010*, pages 69–72, Germany, November 2010.
- [123] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfunder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785, 1997.
- [124] C. Wu, B. Clipp, X. Li, J. M. Frahm, and M. Pollefeys. 3d model matching with viewpoint-invariant patches (vip). In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.
- [125] F.C. Wu, W.C. Ma, R.H. Liang, B.Y. Chen, and M. Ouhyoung. Domain connected graph: the skeleton of a closed 3d shape for animation. *The Visual Computer*, 22(2):117–135, 2006.
- [126] Z. Wu. Compactly supported positive definite radial functions. *Advances in Computational Mathematics*, 4(1):283–292, 1995.
- [127] Y. Xiao, P. Siebert, and N. Werghi. A discrete reeb graph approach for the segmentation of human body scans. In *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, pages 378–385, 2003.
- [128] W. Xie, R.P. Thompson, and R. Perucchio. A topology-preserving parallel 3d thinning algorithm for extracting the curve skeleton. *Pattern Recognition*, 36(7):1529–1544, 2003.
- [129] M.Y. Yang and W. Foerstner. Plane detection in point cloud data. *Technical Report TR-IGG-P-2010-01*, 2010. Department of Photogrammetry Institute of Geodesy and Geoinformation University of Bonn.
- [130] P. Yang and X. Qian. Direct computing of surface curvatures for point-set surfaces. In *Proceedings of 2007 IEEE/Eurographics Symposium on Point-based Graphics(PBG)*, 2007.

- [131] A. Yilmaz and M. Shah. Actions sketch: A novel action representation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 984–989, 2005.
- [132] L.A. Zadeh. Soft computing and fuzzy logic. *Software, IEEE*, 11(6):48–56, 1994.
- [133] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 666–673, 1999.