Heat Based Descriptors For Multiple 3D View Object Recognition

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Department of Electrical and Computer Engineering

Susana Dias Brandão

B.Sc. Physics Engineering,
Instituto Superior Técnico, University of Lisbon, Portugal

M.Sc. Electrical and Computer Engineering,
Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

August 2015

For Guimas.

Abstract

This thesis contributes to the field of object recognition from observations collected by RGB-D sensors by considering objects with very similar or very complex shapes.

We introduce new descriptors for object 3D partial views, corresponding to their visible surface as observed by the RGB-D sensor. In particular we introduce the Partial View Heat Kernel (PVHK) and the Partial View Stochastic Time (PVST). Both descriptors represent 3D partial views by the distance between the partial view occlusion boundary and a reference point at the surface center. Both descriptors represent distances, by simulating heat diffusing from a source in the reference point to the whole surface. PVHK represents distances by the temperature at the boundary points at a fixed time while PVST considers the time it takes for the boundary points to reach a fixed temperature. We introduce descriptors that also represent the distribution of RGB values over surfaces by associating a diffusion rate with the RGB values, e.g., by simulating a faster diffusion in blue parts than in red ones.

We investigate the natural signature of loose connections in heat diffusion to introduce the concept of complex objects, e.g., chairs. We explore the implications of these signatures in our descriptors, and introduce new, part-aware metrics to compare PVHK descriptors. We also consider very similar objects, that are not always distinguishable by single partial views, but that a mobile robot can circle and collect multiple partial views. Assuming that the robot has a complete representation of each object and that from its odometry can estimate expected changes on observations, we provide an algorithm for the online update on the estimative on the object class. Our algorithm uses a Monte Carlo Sampling-Importance Resampling Filter for combining multiple observations, to which we introduced a similarity based resampling approach for the estimation of a discrete, and constant variable, such as the object class. Our resampling strategy allows to reduce the number of samples required for object classification. Finally, we focus on the importance of the reference point position on the descriptors, and explore the large range of possible descriptors for each partial view. We then introduce an algorithm that searches among all possible descriptors from all partial views of the same object, those that are more closely bundled together and thus improve recognition results in complex objects.

We present recognition results on different sets of objects, both rigid and non-rigid, with and with out color and texture, focusing on same size, same class objects. We also introduce an algorithm for the Joint Alignment and Stitching of Non-Overlapping Meshes (JASNOM), that incidentally allows the construction of complete 3D meshes of objects in the datasets. Finally, we show that the tools presented in this thesis naturally adapt to the representation of more ill-defined shapes. In particular, in response to a challenge from the Veterinary College of The Lisbon University, we applied our methodologies to the identification of very thin goats in an animal farms.

Keywords:RGB-D Sensors, Partial View Representation, Complex Objects, Multiple View Object Recognition, 3D Mesh Construction

Acknowledgments

First and foremost, I would like to thank my advisors, Manuela Veloso and João Paulo Costeira. The past six years have been a wonderful learning experience, and I would like to thank you for your guidance in the incredible world of Robotics and Computer Vision. Also, between your pragmatism and last minute optimism, what I have learned from both goes well beyond the scope of this thesis.

I would also like to thank my family, especially my parents, who always motivated me to continue my education. Without them, I would not have embarked on this journey. I would not have finished the journey without Luís Gumarãis. I cannot thank him enough for the companionship and incredible awesomeness! I also thank my sister and my brother for their friendship and support. Sabina Zejnilovic also deserves to be included in my family as she adopted me into her own in Pittsburgh. Her friendship was (and still is) priceless and I am very thankful for her daily company in the lab.

I would also like to thank Fernando de La Torre, Andrea Vedaldi, Manuel Marques and Rodrigo Ventura, for serving on my thesis committee and for your useful comments. I would also like to thank all the members of the CMU-Portugal structure, who provided me with the wonderful opportunity to participate in a dual Ph.D between CMU and Instituto Superior Tecnico. In particular, I would like to thank Prof. José Moura, and Prof. João Barros made the program possible. I would also like to thank Ana Mateus, who, well beyond her job description, helped me with all the paperwork I never saw, but whose existence I am aware of from other less fortunate Ph.D. students.

Ana Vieira was fundamental for this work, not only by her friendship and enthusiasm, but also by opening the door to our collaboration on animal welfare, and allowing me to leave my lab and visit cool goat farms, while *working*! Telmo Monteiro, George Stiwell and Inês Ajuda, were also important parts of this collaboration, and I am grateful to all.

This work would not have been possible without the help of João Carvalho, Pedro Santos, Tiago Fonseca e Helder Miranda, who kindly served as human models in my work. I would also like to thank Fernando Ribeiro for collaborating with me and investing time in learning and applying my work on his Master thesis. I also want to express my appreciation for the work of Brain Coltin, Joydeep Biswas, Junyun Tay and Somchaya Liemhetcharat whose own research provided me with the infrastructure to use the Cobot and the Nao robots at Manuela's Labs. I thank you for all your help, and I am very pleased to count you among my friends.

Finally, there are all those remarkable people who, albeit not related to my research work, were fundamental for my happiness on both sides of the Atlantic. Those include Sergio Pequito, Ricardo Cabral, Maria Leite and Leid Zejnilovic with whom I had the pleasure of sharing the stress of the first years of Ph.D. Your patience with me was/is remarkable. I would also like to express my gratitude to my close friends Ana Neves, Ana Luísa Pedroso, Ana Luísa Pinho, David Batista, Edgar Felizardo, Luís Ferramacho, Nuno Feliciano, Pinar Ekim Oguz, Paulo Ferreira and Tatiana Cantinho. Your friendship keeps me going! And at last but not the least, I was very happy to make

new friends along the way, both in Portugal and US and I would like to thank Beatriz Ferreira, Çetin Meriçli, Claúdia Soares, Jerónimo Rodrigues, João Saúde, João Mota, José Antunes, Juan Pablo Mendoza, Ricardo Ferreira, Stephanie Rosenthal, Tekin Meriçli, Mehdi Samadi, Vasco Ludovico, and Zita Marinho.

I am deeply appreciative of the support of CMU-Portugal (ICTI) program and Fundação Para a Ciência e Tecnologia (FCT) under the grants SFRH/BD/33780/2009, and FCT UID/EEA/50009/2013.

Contents

List of Figures

Chapter 1

Introduction

We envision robots capable of interacting and collaborating with humans in indoor environments. To fulfill tasks in such environments, robots should be able to recognize and identify objects with different appearance and regular shapes. Furthermore, in recent years, RGB-D sensors have become ubiquitous, and both identification and object recognition from depth and RGB are hot topics in computer vision. In this thesis, we contribute to the effort of having robots recognize objects as perceived by RGB-D sensors. In particular, we introduce new forms of representing both: i) the data retrieved by the sensor, and ii) the a-priori knowledge of the object shape.

The data provided by RGB-D sensors has three main characteristics: i) it corresponds to an image whose pixels have information on the RGB color and depth of the object surface; ii) it corresponds only to partial views of the object, i.e., to the visible surface of the object as observed from a given viewing angle; and iii) it corresponds to a noisy version of the object surface. Figure 1.1 exemplifies the two images provided by the sensor and the partial view of a human in those images. We here address the problem of constructing object representations that allow any future observation by an RGB-D sensor to be compared to previously observed and labeled partial views of different objects, and thus recognized.

We use heat diffusion based descriptors to represent robustly individual partial views. Heat diffusion is known to be resilient to the type of noise present in the RGB-D sensors. Such noise takes the form of both perturbations to the 3D coordinates extracted from the depth information, and to small holes in the object surface. Others have introduced different descriptors based on heat diffusion and have used it to represent complete 3D object surfaces or points in complete objects [13, 14, 18, 48, 56, 58]. Those descriptors depend both on local and global object geometry, and thus do not handle properly large holes in the surface, e.g., the absence of half the object in partial views would change any descriptor previously computed on the complete object. We here contribute to the family of heat diffusion based descriptors with a new approach to representing partial views.

Since a view from the sensor provides incomplete information on object surfaces, we represent

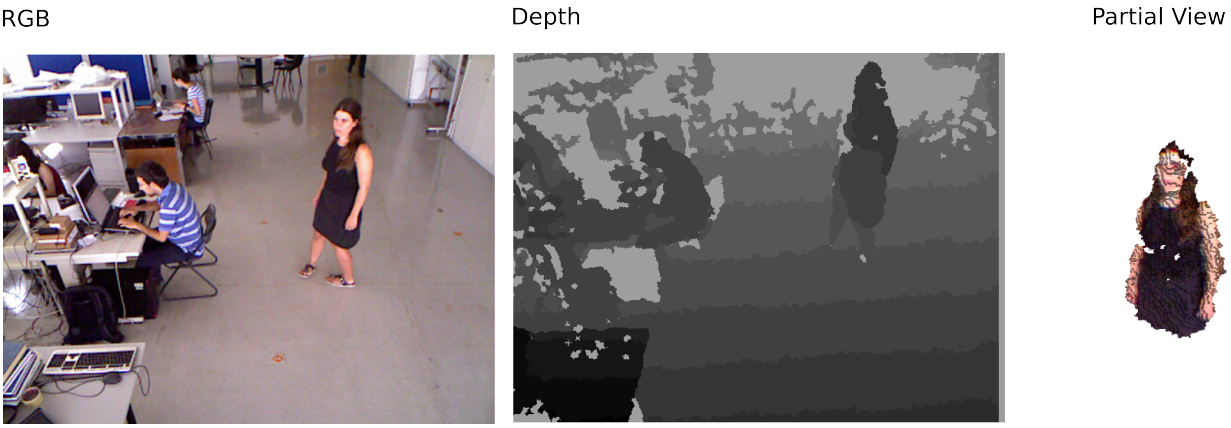


Figure 1.1: Data returned by an RGB-D sensor, comprising an RGB and Depth image. Using both images, we obtain an object partial view, which we use as input to our work.

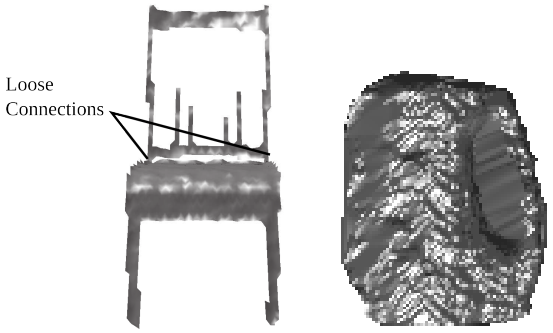
complete objects as sets of partial views, each related to a different viewing angle. We thus organize our a-priori knowledge of all objects of an environment as libraries, where we represent each object as a collection of viewing angles and corresponding descriptors. We are concerned with the collection size required to represent each object. When the collection is large, each new observation of that object will likely be similar to a partial view in the collection, decreasing the probability of miss-classifications. However, by increasing the library size, the effort required to recognize a single partial view also increases.

This thesis focuses on human-made objects, often of the same class, e.g., mugs and kettles. These objects have generic geometric features, such as planes and cylinders, and often share those features with other objects. This lack of distinctive features leads to ambiguous shapes and to objects that are only recognized when a small set of discriminative partial views is observed. Together with libraries that represent only a sparse set of the possible set of partial views, ambiguous partial views are one of the main sources of miss-classifications we faced in our experiments.

Miss-classifications can be detected and corrected when the agent estimating the object class is a mobile robot capable of collecting multiple partial views from different viewing angles. By combining past estimates on the object class while collecting new observations, the robot has constant access to an increasingly accurate classification, and can stop the estimation when it finds a distinctive feature that ensures high confidence. Others have previously introduced methods that combine multiple 3D partial views, usually for the purpose of constructing a complete 3D model, e.g., the kinectFusion algorithm [30]. Such methods could be used as a first step in a 3D object recognition algorithm. However, the robot would first need to go around the complete object before attempting to classify it. This thesis assumes a robot moving around an object, with access to its odometry, and updating continuously the object class, by collecting and representing individual partial views, combining past observations, making predictions on futures ones, and validating its belief on classification. Such robot would not have to go around the complete object.

Heat diffusion based descriptors can seamlessly encode photometric information in the shape descriptor. By indexing color and texture to the object shape, we can further disambiguate similar shapes. Appearance provides discriminative information on the object, especially when we need to identify same class, same geometry objects. Furthermore, indexing appearance to specific points on the object surface allows to further discriminate between objects that share similar visual feature, e.g., a human with a red shirt and blue jeans from another with blue shirt and read jeans.

We further realized that heat diffusion reflects strongly the existence of loosely connected parts, and we introduced the concept of complex objects, e.g, the chair in Figure 1.2(a), as opposed to those objects with compact surfaces, e.g., the kettle in Figure 1.2(b). We formalize the distinction between regular objects and complex objects by exploring the impact of loosed parts on heat based descriptors.



(a) Complex object (b) Regular objects

Figure 1.2: Shapes of regulars and complex objects. The chair is complex, because its back is loosely connected to the seat.

We also note that, as loosely connected parts often self occlude other parts, complex object shapes change significantly between viewing positions, inducing changes in their representations. To accommodated such variability and avoid miss-classification, complex objects require libraries with a large number of partial views. Motivated by the difficulty in representing complex objects by a collection of partial views, we address the construction of compact libraries, where descriptors of the same object bundle together and are as far away as possible of other objects. We expect that compact libraries obtain good recognition results even with a sparse set of partial views.

We address the problem of collecting partial views annotated by the respective sensor viewing angle for inclusion in object libraries. In particular, the viewing angle is difficult to control and to estimate. Thus, when we require precision in the viewing angle estimation, or partial views from positions beyond the allowed by the experimental apparatus, we use existing 3D CAD models. Using openGL libraries, we can generate partial views from these CAD models from all the positions and sensor properties as we need. The CAD models can be retrieved from existing datasets, such as 3D Google Warehouse, or can be constructed by combining partial views into a single model. We introduce an algorithm that allows the creation of 3D+RGB color models.

The tools we developed throughout this thesis are not constrained to the classification of objects, and can be applied in different contexts. We had the opportunity to do so when members of the Veterinary College of the Lisbon University asked us for help in estimating the Body Condition Score (BCS) in goats in animal farms. The BCS conveys information on how fat or thin is an animal, and is of relevance for milk production as both very fat and very thin animals have poor production. Thus, we were invited to devise methods that would allow to automate the estimation of the BCS while animals moved freely in a corridor, as showed in Figure 1.3. Such a premise is in sharp contrast to current methods that require the physical constraint of each animal and a specially trained veterinary. In an initial collaboration, [65], we showed that changes in the rump volume are strongly correlated with BCS, as illustrated in the two rump examples in Figure 1.3 and that humans can be trained to consistently access BCS by visual inspection. In this thesis, we show how our shape representation approach can assess changes in volume and identify very thin animals.

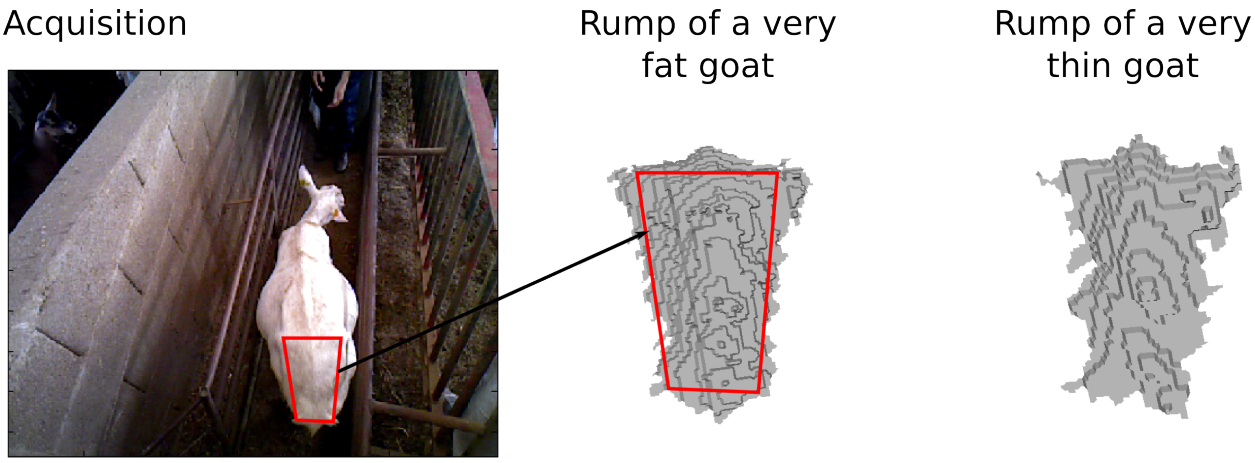


Figure 1.3: Example of the acquisition setup and how goats are different types of surfaces that we need to classify in order to identify extremes of very thin and very fat animals.

1.1 Thesis Question and Approach

This thesis seeks to answer the question:

> How to represent 3D objects within a library, so that they can be identified from an observation of an RGB-D sensor collected from *at least one viewing angle*, considering that objects can have strong *similarities* or very *complex shapes*.

This thesis extends the heat diffusion based family of descriptors so that we can represent partial views. In particular, we introduce the Partial View Heat Kernel (PVHK) that is resilient to noise, is unique, depends on the viewing angle and can be extended to include photometric information. PVHK represents the 3D surfaces corresponding to the visible part of an object in a holistic way, i.e., so that a single descriptor contains information on the whole surface.

The information conveyed by PVHK represents the distance between points in the partial view boundary and a reference point at the center of the surface. To be robust to noise, PVHK represents distances using the solution to a heat diffusion process. Such process is inherently resilient to small perturbations and holes on the surface and allows for the easy integration of photometric information. As illustrated in Figure 1.4, we consider a heat source in the reference point and simulate the heat diffusing through the surface. We then stop the simulation and evaluate the temperature at the boundary. Points closer to the source will be warmer than points further away, and thus temperature effectively represents distances.

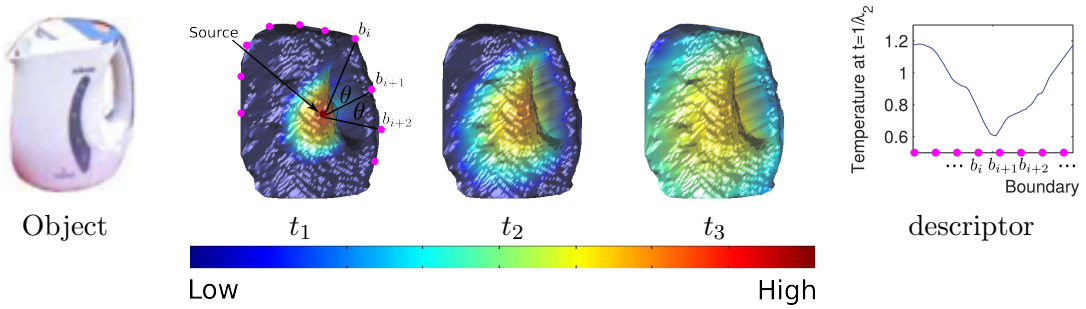


Figure 1.4: Construction of the Partial View Heat Kernel by diffusing heat from a source and evaluating the temperature at the boundary.

The source position is fundamental to the descriptor, as the same partial view has different descriptors if the source changes. We can choose the source position using different approaches so that the resulting descriptor adapts to a specific need. For example, we can obtain descriptors that depend on the observer position by choosing the source based on the relative position between observer and object. Examples are: i) choosing the source as the point in the object surface that is closest to the viewer, or ii) choosing the source as the point in the object surface that is closest to the center of the segmented depth image. The above rules allow to consistently return the same source position without prior knowledge of the object class and observer's viewing angle.

The dependency of the descriptor on the viewing angle is essential when combining multiple observations from different viewing angles to minimize miss-classification errors resulting from similarity between object shapes and sparse libraries. We use a Bayesian setting to sequentially estimate the probability of each object class, updating current estimates with each new observation. To improve the estimate on the object class, the robot can use its odometry to predict new observations and compare them with the actual ones, updating the belief on each classification.

The robot needs to keep estimates not only on the object class, but also in its relative position with respect to the object. We use a Monte Carlo Sampling-Importance Resampling, for the update, often used in tracking or localization algorithms. Common implementations use maps from positions and objects to make predictions on observations and filter wrong hypotheses.

In our implementation of the particle filter, we follow [28] and use a map that relates observations to objects and positions. The map seamlessly combines the notion of similarity between objects and partial views into the filtering process, achieving better estimates of the object class with less computational effort.

We further disambiguate similar shapes by seamlessly introducing color and texture information into the heat diffusion process as a diffusion rate, e.g., we can say that heat diffuses faster in blue than in red. The resulting descriptor represents the photometric information indexed to the 3D shape, so that the descriptor is affected by both the RGB values and their geometric distribution over the object surface.

To handle complex objects, we explored the impact of loose connections on the heat diffusion to identify parts and define proper metrics for complex object's descriptors. We also introduce a second heat based partial view descriptor, the Partial View Stochastic Time (PVST), which naturally handles the presence of parts. As the PVHK, the PVST represents partial views using a robust representation of distances between boundary points and a reference point at the center of the partial view. However, in PVST, distance is conveyed by the time required for the temperature at each boundary point to reach a fixed temperature.

We further used the freedom to choose the source position to construct compact libraries for complex objects, and thus reduce the number of partial views needed in the object library. We take advantage of changes in the descriptor by the source position in the partial view to construct object libraries that follow some desirable property. An example of such property is to have very different descriptors for different objects.

We also introduce an algorithm for the construction of 3D+RGB models of regular objects. The algorithm for the Joint Alignment and Stitching of Non-Overlapping Meshes (JASNOM), allows the construction of an object 3D model by using only two, non-overlapping but complementary partial views. Incidentally, such models can be used in the construction of object libraries as any other CAD model.

We apply the above formalism for the classification of the Body Condition Score (BCS) of goats, and in particular to identify very thin or very fat animals. To evaluate the BCS we assess the rump volume by comparing the heat diffusion in the rump with the diffusion on a rump 2D projection. The thinner the goat, the closest its rump is to a plane and the smallest the difference between the heat diffusion in the two surfaces. The application is a promising example of other 3D images understanding applications that we may tackle with the methodology we introduce in this thesis.

1.2 Thesis Contributions

The key contributions of this thesis are as follows:

- the Partial View Heat Kernel (PVHK) descriptor for the representation of noisy partial views with photometric information;

- a multiple view multiple hypotheses algorithm for estimating objects from multiple observations;
- an analysis of the impact of the loose connections in complex objects to diffusion based descriptors;
- a part-aware metric for the comparison of descriptors of complex objects;
- the Partial View Stochastic Time (PVST) for the representation of partial views of complex objects;
- a source placement algorithm for the offline creation of robust object libraries;
- the Joint Alignment and Stitching of Non-Overlapping Meshes algorithm for the fast construction of textured meshes of complete objects;
- an approach for the automatic identification of very thin goats in an animal farm using 3D sensors.

1.3 Thesis Guide

The thesis is organized in 10 chapters where we present in detail the thesis contributions, and results as we here we outline.

- **Chapter 2 - Partial View Heat Kernel**
  We address the problem of representing the visible surface of an object, i.e., its partial view, as collected by an RGB-D sensor. We review an existing class of 3D descriptors based on heat diffusion, and introduce a partial view descriptor, the Partial View Heat Kernel (PVHK), for the purpose of robustly representing partial views, and combining both the geometric and photometric information into the same descriptor. We provide examples of descriptors in rigid and non-rigid objects; analyze the impact of noise on the descriptor; and address the conditions for which the descriptor is discriminative.

- **Chapter 3 - Partial View Recognition**
  We address the problem of identifying a partial view by comparing its PVHK descriptor with those stored in an object library. We introduce the distance metric we use for the comparison of partial view descriptors, the modified Hausdorff distance. We then show the descriptor and metric effectiveness on the recognition of different object sets. We use real everyday objects, of similar size but distinct shape; same class objects both rigid and non-rigid, with almost exact shape, but with different photometric information. We compare the performance of the PVHK with other partial view descriptors.

- **Chapter 4 - Incremental Object Recognition**
  We address the problem of combining information from multiple observations, captured by a mobile robot that collects multiple partial views. We introduce our Multiple View Multiple Hypotheses algorithm, and show that by using a map from observations to positions and object classes, we can reduce the computational effort and improve recognition. We test our algorithm in i) libraries of objects that are identical from some viewing angles, but have distinctive features; and ii) libraries with a small number of partial views per object.

- **Chapter 5 - Complex Objects and the Partial View Stochastic Time (PVST)**
  We address the problem of representing partial views of complex objects using heat diffusion. We motivate the need to discriminate regular from complex objects, and show how the of loosely connected parts of complex objects impacts heat diffusion. We then introduce a new metric to compare partial view of complex objects, and a new descriptor, the Partial View Stochastic Time, that seamlessly handles object parts. We empirically evaluate the performance of the new approaches on libraries of partial views from 54 chair.

- **Chapter 6 - Source Placement and Compact Libraries**
  We address the problem of defining a source position for a given partial view. We introduce the notion of multiple descriptors for each partial view, by assuming that each point on the surface is a possible heat source. We then choose among the multiple descriptors from several partial views of the same object, those that lead to compact libraries, and to a better recognition of each new partial view. We test our source placement in two libraries of same class complex objects, one with guitars and the other with chairs.

- **Chapter 7 - Construction of 3D Models**
  We address the problem of off-line creating object 3D models. We introduce an algorithm, the Joint Alignment and Stitching of Non-Overlapping Meshes (JASNOM), that constructs complete 3D models of object surfaces by aligning two non-overlapping meshes that cover the complete object shape. By using directly the 3D information retrieved from the sensor, JASNOM allows the creation of textured models. We empirically show that our algorithm can generate complete models of common objects, such as kettles and books, as well as of non-rigid shapes such as humans.

- **Chapter 8 - Application to Automated Animal State Classification**
  We explore the possibility of using the developed approaches for shape representation and understanding in applications beyond object recognition. In particular, we apply the heat diffusion formalism to identify very thin animals in a dairy goat farm. We introduce our approach to evaluating the rump volume by comparing heat diffusion in the rump with the heat diffusion in a plane. We then show our representation results in an annotated set of 30 animals of different species, shapes, and sizes.

- **Chapter 9 - Related Work**
  We provide an overview of the related work and how it relates to the work here presented. In particular, we focus on the three fields to which this thesis contributes, namely: i) 3D+photometric representations; ii) multiple view object recognition; iii) mesh stitching.

- **Chapter 10 - Conclusion**
  We conclude this dissertation with a summary of our contributions along with a discussion of future work.

Chapter 2

Partial View Heat Kernel (PVHK)

In this chapter, we address the problem of representing the visible surface of an object, i.e., its partial view, as retrieved by an RGB-D sensor. As we show in Section 2.1, these surfaces are rich in information, as sensors provide both photometric and geometric information. However, the 3D information provided by common RGB-D sensors is extremely noisy. We here introduce a noise resilient partial view descriptor, the Partial View Heat Kernel (PVHK), [10], that represents both the geometric and photometric information. While we leave to Chapter 3 a detailed analysis of existing 3D partial views representations, in Section 2.2 we briefly overview existing representations resilient to noise whose tools we share. We present PVHK in detail in Section 2.3, and in Section 2.4 we show how the PVHK can accommodate other sources of information, such as the photometric information provided by the sensor. Finally, in Section 2.6 we highlight the main properties of the PVHK.

2.1 Partial Views of RGB-D cameras

We address the representation of 3D partial views, i.e., the the surfaces formed by RGB-D sensors as the object self occludes part of its complete 3D surface, as shown in Figure 2.1.

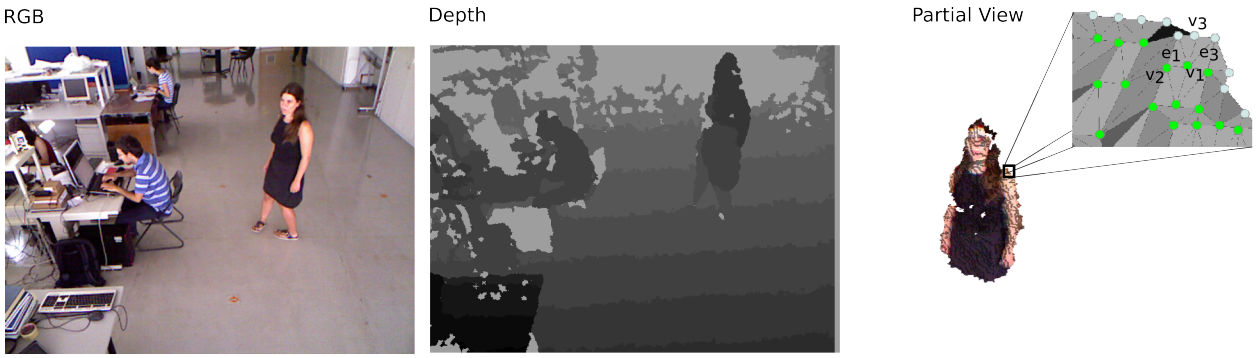


Figure 2.1: Partial view returned by an RGB-D sensor.

Furthermore, we assume a segmentation step, outside the scope of this work, that provides the object partial view separated from the background. The focus of this work are rigid, human made objects, but we empirically show that it can also handle non-rigid object such as those presented in Figure 2.1.

We are interested in common depth sensors that provide partial views as an organized set of vertices, such as the one highlighted in Figure 2.1. The vertices organization defines neighborhoods between vertices and allows to represent partial views as triangular meshes, composed of non-overlapping triangles that cover the visible surface of the object. Additionally, the sensor provides the color of each vertex in the form of an RGB image.

Thus, the returned information from an RGB-D depth sensor is:

- an object mesh, $M = \{V, E, F\}$, composed of
  - a set of vertices, $V = \{v_1, ..., v_{N_V}\}$;
  - a set of edges, connecting neighboring vertices $E = \{e_1 = (v_l, v_k), e_2, ...e_{N_E}\}$;
  - a set of triangles, connecting edges $F = \{f_1 = (e_l, e_k, e_m), f_2, ...f_{N_F}\}$;
- the 3D coordinates of each vertex $v_i : \bar{x}_i \in \mathbb{R}^3$, in the sensor coordinate system;
- the RGB values of each vertex, $v_i : \bar{c}_i \in \mathbb{R}^3$.

However, the sensor is also noisy, inducing uncertainty in the 3D coordinates of surface points and leading to surface holes, as exemplified in the human partial view in Figure 2.1.

To represent the partial views retrieved by such a sensor, we introduce the Partial View Heat Kernel (PVHK) descriptor, which is:

1. Informative, i.e., a single descriptor robustly describes each partial view;
2. Stable, i.e., small perturbations on the surface yield small perturbations on the descriptor;
3. Inclusive, i.e., appearance properties, such as texture, can be seamlessly incorporated into the geometry-based descriptor.

The combination of these three characteristics results in a representation especially suitable for partial views captured from noisy RGB-D sensors during robot navigation or manipulation where the object surfaces are visible with limited, if any, occlusion.

PVHK builds upon distances, over the object surface, between a reference vertex, v_s, and the boundary. The ordered set of all the distances represents surfaces in a unique way, apart from symmetric and isometric transformations. In the example of Figure 2.2, we show the reference point at the center of the partial view, and the different sets of shortest paths between the reference vertex and each boundary point. We order all points in the boundary, $\{l_0, l_a, ..., l_c\}$ as a line, and represent

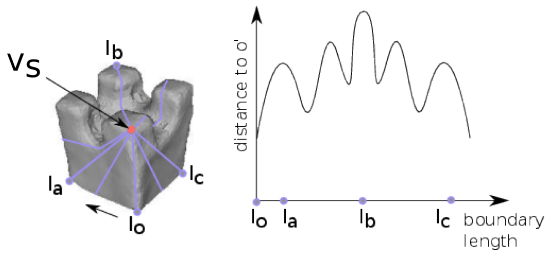


Figure 2.2: Example of the information conveyed by the Partial View Heat Kernel (PVHK) representation.

each point by a function of the shortest distance to v_s. The PVHK represents the complete surface by the organized set of distance, represented in the plot on the right of Figure 2.2.

By representing the shape through its boundary, PVHK allows to easily compare two partial views, without requiring any registration between the two. Furthermore, PVHK leverages on the fact that the boundaries are well defined for each object and correspond to comparable sets of points, i.e., we know that if two partial views were the same, we would have the same distances, and if the distances are different, we can infer changes in partial views.

PVHK relies on diffusive geometry concepts to represent *average* distances ([42], [61]) to ensure that the descriptor is stable with respect to noise and topological artifacts, e.g., holes or small occlusions. Notably, we model the averaging process as the diffusion over the surface of heat transferred to the surface at a single source point. Hence, PVHK represents a partial view as the temperature at the boundary as a result of a heat pulse at the reference point v'_s, as illustrated in Figure 2.3.

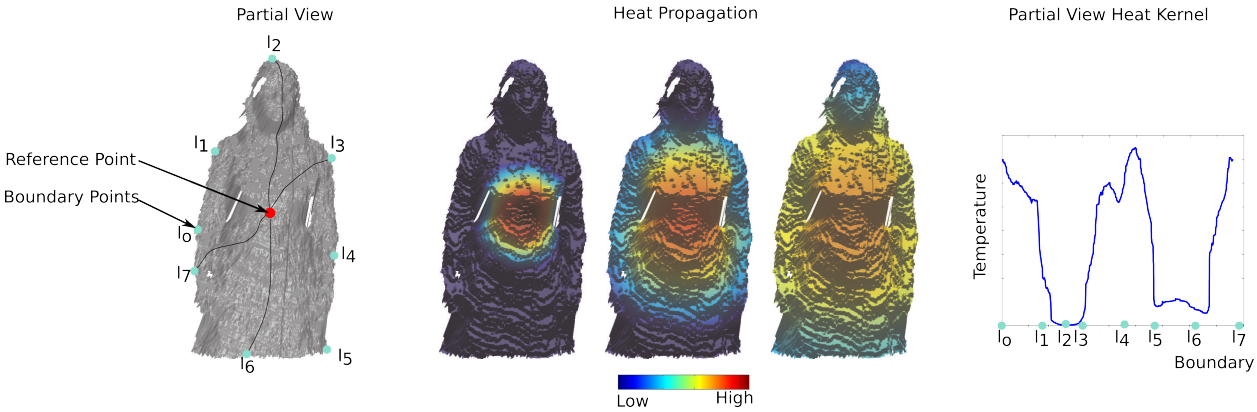


Figure 2.3: Example of the main steps in the computation of the PVHK.

Finally, to ensure seamless integration of heterogeneous information, such as surface color, PVHK treats different visual properties as different heat diffusion rates. As heterogeneous rates lead to different temperature profiles on identical 3D shapes, PVHK uniquely represents objects with the same geometry but different color or texture. By indexing color to the geometry, it

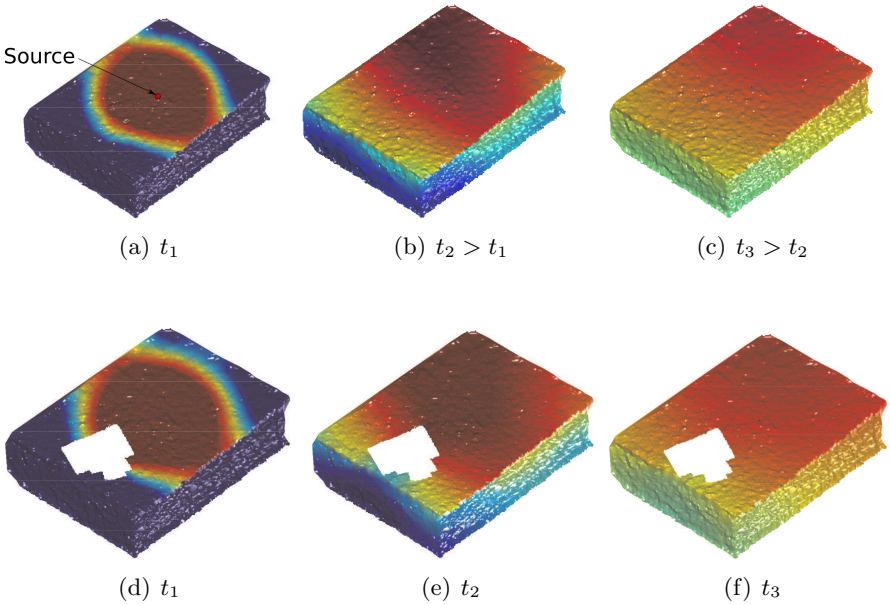
also differently represents objects with the same shape and similar appearance but with different distributions.

2.2 Representations Based on Heat Diffusion

Diffusive distances [42] and diffusive geometry, provide a robust approach to representing shapes subject to noise, and in particular to topological noise resulting from small holes in the surface. Diffusive distances are related to shortest path distances, which are effective at representing a surface topological information. However, are less sensitive to noise, as they are related to diffusive processes occurring over a surface.

Diffusive processes can be interpreted as a sequence of local averaging steps applied to a function representing some quantity, e.g., temperature. The averaging steps dilute local non-homogeneities in the function and effectively transport the quantity from regions of higher values to regions of lower values.

Figure 2.4 shows two examples of diffusive processes taking place on similar surfaces, different only on account of a hole. In the first example, Figures 2.4(a) 2.4(c), the temperature evolves from an initial source to the whole partial view following a concentric pattern, associated with the shortest path between points. In the second example, Figure 2.4(d) to Figure 2.4(f), while the hole affects the shortest path between points, it does not change the temperature significantly. The averaging steps result in that the temperature at a point is defined first by the neighborhood and only implicitly depends on the distance to the source.



(a) t_1 (b) $t_2 > t_1$ (c) $t_3 > t_2$

(d) t_1 (e) t_2 (f) t_3

Figure 2.4: Example of heat diffusion on similar surfaces. Color represents temperature and red regions are warmer than blue.

Diffusive processes can describe local features, such as the Heat Kernel Signature (HKS) [61] and the Scale Invariant Heat Kernel Signature [14]. HKS is a highly robust local descriptor that contains large-scale information. HKS represents a point with the temperature evolution after placing a heat pulse on that point. The time evolution depends on how fast the temperature diffuses to the neighborhood, which in turn depends on the object geometry.

While both descriptors, HKS and SI-HKS, perform well on complete 3D shapes, the same point on an object surface may have different descriptors when parts of the shape are missing [14]. Thus, as we address in Chapter 9, they are not suitable for the representation of partial views, where the we have a large variability of shapes.

In the following, we review the formalism to simulate heat diffusing on surfaces represented by meshes such as those returned by RGB-D sensors.

2.2.1 Heat Kernel

Formally, the temperature diffusion over a surface, as the one in Figure 2.5 defined by a set of vertices $V = \{v_1, v_2, ..., v_{N_V}\}$, with coordinates $\{\bar{x}_1, \bar{x}_2, ..., \bar{x}_{N_V}\}$ together with a set of edges $E = \{e_1 = (v_l, v_k), e_2, ..., e_{N_E}\}$, is described by Eq. 2.1:

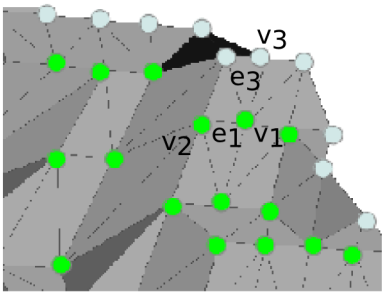


Figure 2.5: Example of the mesh structure, where the dots represent vertices and lines connecting the dots correspond to edges.

$$L\bar{T}(t) = -\partial_t \bar{T}(t), \tag{2.1}$$

where $L \in \mathbb{R}^{N \times N}$, is a discrete Laplace-Beltrami operator, and $\bar{T}(t) \in \mathbb{R}^N$ is a vector containing the temperatures over all vertices in the surface at each instant t.

There are many approaches to representing the discrete Laplace-Beltrami operator [67]. We choose a distance based one, which corresponds to a weighted graph Laplacian where the weight of

each edge is the inverse of its length:

$$L\bar{T}(t) = (D - W)\,\bar{T}(t), \tag{2.2}$$

$$[W]_{v_i,v_j} = \begin{cases} 1/\|\bar{x}_{v_i} - \bar{x}_{v_j}\|^2, & \text{iff} \quad e_l = (v_j, v_i) \in E \\ 0, & \text{otherwise} \end{cases}, \tag{2.3}$$

and where D is a diagonal matrix with entries $D_{ii} = \sum_{j=1}^{N} [W]_{ij}$.

When we apply the Laplace-Beltrami to a vector $\bar{T}$ we obtain, for each vertex v, the averaged difference between the value of $[\bar{T}]_v$ and its neighbors, i.e., if we represent the row v of L as L_v, we can write:

$$L_v\bar{T} = \sum_i \left([\bar{T}]_v - [\bar{T}]_i\right)[W]_{v,i} \tag{2.4}$$

Thus, the Laplace-Beltrami represents differences between the neighboring entries of a vector.

The heat diffusion equation in Eq. 2.1, relates the rate of change of temperature at a single vertex with the difference between the temperature at that vertex and its neighbors. Furthermore, with the Laplace-Beltrami operator defined in Eq. 2.2, closer neighbors have a closer influence in the temperature. This means that sharp gradients in the temperature will be smoothed out very fast. Furthermore, the heat diffusion naturally segments large edges, as they will have very small weights.

The heat kernel, $k(v_j, v_s, t)$ is the solution of Eq. 2.1 at time instant t and vertex v_j when the initial temperature $\bar{T}(0)$, is zero everywhere except at source vertex v_s, i.e.,

$$\bar{T}(0) : [\bar{T}(0)]_i = \begin{cases} 1, & i = v_s \\ 0, & \text{otherwise} \end{cases}. \tag{2.5}$$

The above initial value problem has a closed form solution in terms of the eigenvectors, ϕ_i, and eigenvalues, λ_i, of the Laplace-Beltrami operator, which is given by Eq. 2.6:

$$k(v_j, v_s, t) = \sum_{i=1}^{N} e^{-\lambda_i t}[\bar{\phi}_i]_{v_j}[\bar{\phi}_i]_{v_s}, \tag{2.6}$$

where $[\bar{\phi}_i]_{v_j}$ is the value of $\bar{\phi}_i$ at the vertex v_j.

Eq. 2.6 contains information on the complete surface through the eigenvalues and eigenvectors of L, i.e., even when v_j and v_s are fixed points on the object surface, the descriptors changes if L changes.

2.3 Computing Partial View Descriptors

We define PVHK as the surface boundary temperature measured by stopping the heat diffusion t_s seconds after placing the heat source on a vertex, v_s.

To consistently obtain the same descriptor in independent observations, regardless of object class or viewing angle, we need to define the stopping time, t_s, the source vertex, v_s, and the boundary points where we compute the temperature.

2.3.1 Stopping time

The stopping time t_s must be:

- large enough so that the temperature at the boundary, which is initially zero, raises above some threshold;
- small enough so that the temperature does not become uniform over the whole surface.

In a graph, both events are governed by the diffusion time scale, which is proportional to λ_2^{-1}, the inverse of the first non-zero eigenvalue of the graph Laplace Matrix. For most regular objects, composed of compact surfaces, the diffusion time scale ensures the two above conditions, and guarantees that the temperature at the boundary vertices is representative of the distance to the heat source. Thus, unless stated otherwise, we use $t_s = \lambda_2^{-1}$. In Chapter 5, we will address in more detail the impact of different values of t_s in the descriptor.

2.3.2 Source Position

We choose v_s using simple rules that do not require a-priori knowledge on the object class nor orientation. For example, we use the point closest to the observer, or the point at the center of the segmented depth image. Other approaches could be thought of, and implemented, but the most important aspect is to use consistently the same approach.

The above suggestions have two main properties: i) are easy to find when the vertex coordinates, $\bar{x}_v \in X$ are in the sensor's coordinate system; and ii) depend on the sensor position. When we chose the source as the point closest to the observer, we find the source as:

$$v_s = \underset{\bar{x}_v \in X}{\operatorname{argmin}} \|\bar{x}_v\|^2 \tag{2.7}$$

When we chose the source as the point in the center of the segmented depth image, we note that the projection on the camera plane, corresponds to setting the z-coordinate in each $\bar{x}_v$ to zero, so we estimate the source position as:

$$v_s = \underset{\bar{x}_v \in X}{\operatorname{argmin}} [\bar{x}_v]_1^2 + [\bar{x}_v]_2^2 \tag{2.8}$$

In Chapter 6, we address in detail the impact of different sources in the descriptor and propose new approaches for the source placement.

2.3.3 Boundary Vertices

As the boundary is an oriented closed curve, to represent information on the boundary as a vector, we to consistently define an origin to the curve. We here define such point by assuming that objects have a privileged orientation, i.e., they have a top direction, which defines privileged points over the boundary, e.g., the topmost. While other choices could be made, we define the boundary origin based on a 2D coordinate system defined over the depth image, as illustrated in Figure 2.6. We place the coordinate system origin at the center of the segmented depth image, and align its $\hat{e}_y$ axis with the depth image $\hat{e}_v$. We define the boundary origin as the intersection of the coordinate system $\hat{e}_x$ and the boundary, showed by the white spot in Figure 2.6.

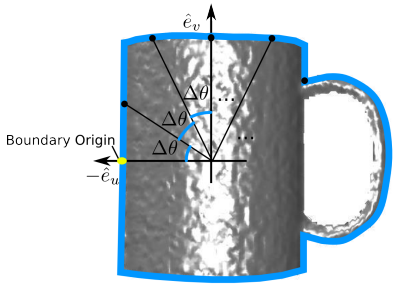


Figure 2.6: Example of the local coordinate system that we use to define the boundary origin and orientation.

However, boundaries of different partial views have a different number of vertices. To have comparable sets of temperatures, we cannot use the temperature over all the vertices. In a first approach, we computed the temperature over the whole boundary, and then interpolate using distances over the boundary. In a second approach, we used vertices over the boundary that, in polar coordinates, were evenly spaced in the angle coordinate, as we show with the black dots in Figure 2.6. The second option is more stable with respect to poor segmentation and noise in the boundary coordinates, while the first is more stable with respect to deformable shapes.

2.3.4 Algorithm for Computing PVHK Descriptors

Given a segmented mesh, M, a set of coordinate vertices, X, expressed in the camera coordinate system, and a set of ordered boundary vertices, B, we determine the source position, v_s, then we compute the Laplace-Beltrami operator, L, and estimate the eigenvectors and eigenvalues. From the first non-zero eigenvalue, we compute the diffusion time scale $t_s = \lambda_2^{-1}$. We finally compute

the temperature at the boundary as:

$$[\bar{z}]_j = k(v_{bj}, v_s, t_s) \approx \sum_{i=1}^{30} e^{-\lambda_i t_s} [\bar{\phi}_i]_{v_{bj}} [\bar{\phi}_i]_{v_s}, \tag{2.9}$$

which differs from 2.6 as we only use the lowest 30 eigenvalues, as $e^{-\lambda_i/\lambda_2} \sim 0$ for $i > 30$. Algorithm 2.1 summarizes the steps required to estimate the PVHK descriptor.

Algorithm 2.1: Computing the PVHK descriptor

Input: Set of vertices X in the camera coordinate system, mesh M, Boundary vertices $B = \{v_{b1}, v_{b2}, ..., v_{bM}\}$

Output: PVHK descriptor, $\bar{z} \in \mathbb{R}^M$.

FIND SOURCE POSITION:

$v_s \leftarrow$ sourcePosition(X) (from Eq. 2.7 or 2.8)

COMPUTE LAPLACE-BELTRAMI OPERATOR:

$L \leftarrow$ computeLaplaceBeltrami(M, X) (from Eq. 2.2)

ESTIMATE EIGENVALUES AND EIGENVECTORS:

$\{\bar{\phi}_i, \lambda_i, i = 1, ..., 30\} \leftarrow$ eigenvectors(L)

COMPUTE DIFFUSION TIME SCALE:

$t_s \leftarrow 1/\lambda_2$

COMPUTE TEMPERATURE AT BOUNDARY:

$\bar{z} : [\bar{z}]_i \leftarrow k(v_{bj}, v_s, t_s)$ (from Eq. 2.9)

Figure 2.7, shows examples of descriptors of objects that share similar sizes and shapes: a cylindrical box, a mug with a handle, a quadrangular box and a toy castle. We note for example as the cylinder and the mug share an almost identical partial view, except that the cylindrical box is taller than the mug. The difference in size is reflected in the descriptor, as the descriptor for the mug has 4 *hills* while the descriptor for the cylinder has only 2. Also the mug handle is reflected in the descriptor by a reduction in the temperature, resulting from the larger distance over the boundary between the handle and the source.

2.4 Color and Texture in PVHK

We introduce color and texture information into PVHK representation by slightly modifying the heat equation. The heat equation represents surfaces with the same diffusion rate on all points. By using different rates at different points, we generate different descriptors for objects with the same geometry and different descriptors for objects with the same color or texture but distributed differently. Thus, to differentiate objects on both appearance and geometry, we relate appearance with diffusion rate. We rewrite the heat equation in as:

$$C^{-1} L \bar{T}(t) = -\partial_t \bar{T}(t) \tag{2.10}$$

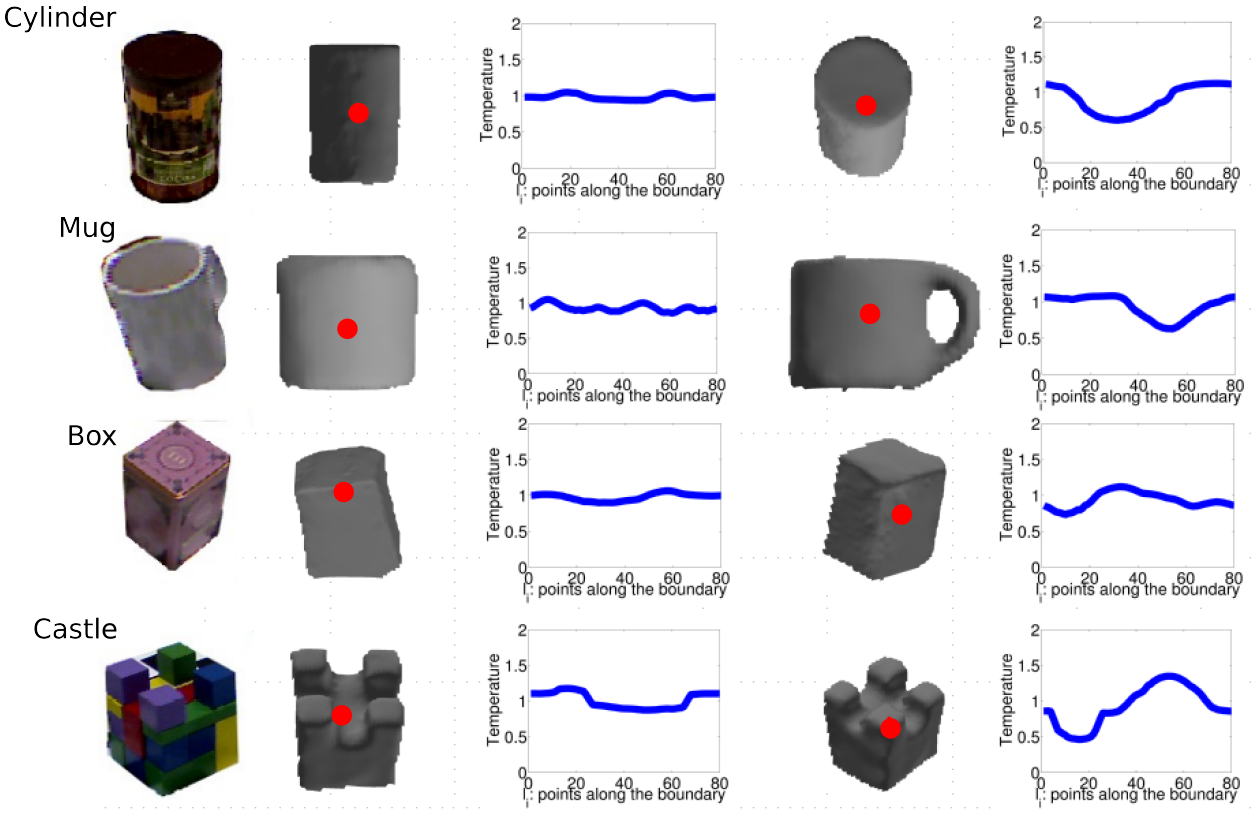


Figure 2.7: Examples of descriptors in different objects that share similar shapes and sizes. The red dot corresponds to the source position.

where C is a diagonal matrix, whose element $[C]_{v,v}$ is any scalar associated with color, or texture, at vertex v.

The solution to the non-homogeneous problem in Eq. 2.10 is identical to the solution to the homogeneous problem in Eq. 2.4, but the eigenvalues, λ_i^c, and eigenvectors $\bar{\phi}_i^c$ are now the solution of the generalized eigenvalue problem $L\bar{\phi}_i^c = C\bar{\phi}_i^c\lambda_i^c$.

With the initial condition of Eq. 2.5, the heat kernel at $t = t_s$ then becomes:

$$k^c(v_j, v_s, t_s) = \sum_{i=1}^{30} [\bar{\phi}_i^c]_{v_j} \exp(-\lambda_i^c t_s)[\bar{\phi}_i^c]_{v_s}[C]_{v_s,v_s}. \tag{2.11}$$

Our proposed approach differs from previous efforts to combine color and geometry, in particular from [34]. Notably, we can extend $[C]_{v,v}$ to represent any scalar quantity and not just color. Examples of useful scalars are the color hue value or cluster indices, e.g., after some clustering preprocessing using any other appearance representation.

Hence, we introduce the C-PVHK, which is computed using Algorithm 2.1, only now we compute the temperature at the boundary using Eq.2.11 instead of Eq. 2.9.

We illustrate the impact of adding appearance information to the descriptor by considering a person in the same position, wearing the same clothes but with different colors, as in Figure 2.8. We assume that $[C]_{v,v}$ corresponds to the color' hue value when scaled to the interval [0.5,1] and present the temperature along the boundary in the graphic on Figure 2.8. The four descriptors present a common behavior associated with shape, e.g., the head, point l_1, introduces the same decrease in the temperature. However, the color modulates the temperature in a very significant way. Notably, the color at the source, which in the example is placed in the blouse, leads to the gap between *Original+Different Skirt* and *Different Blouse + Different Dress*.

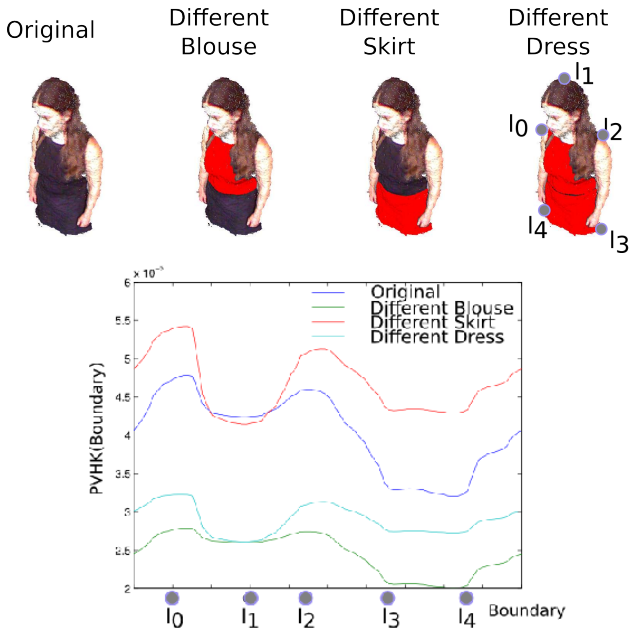


Figure 2.8: Color impact on the descriptor. On the left, we present the mesh and colors. On the right, we present the respective C-PVHK descriptors

2.5 Computational Effort

Given a mesh with a boundary and a source position, the most time consuming step in the computation of a PVHK descriptor is computing the first 30 eigenvectors and eigenvalues of a very sparse matrix. Thus, the effort of computing the PVHK depends on the choice of algorithm to estimate eigenvalues. We use Matlab *eigs* function as it is, to the best of our experience, the fastest implemented algorithm to estimate the first n eigenvalues of sparse matrices.

The graphic in Figure 2.9 shows the time required for to compute the descriptor on a Intel(R) Core(TM) i7-3770 Quad-Core @ 3.40GHz as a function of the number of the number of vertices in the surface mesh.

The number of vertices in a partial view depends on the size of the object and of distance between object and sensor. However, common everyday objects, observed from around 1m of

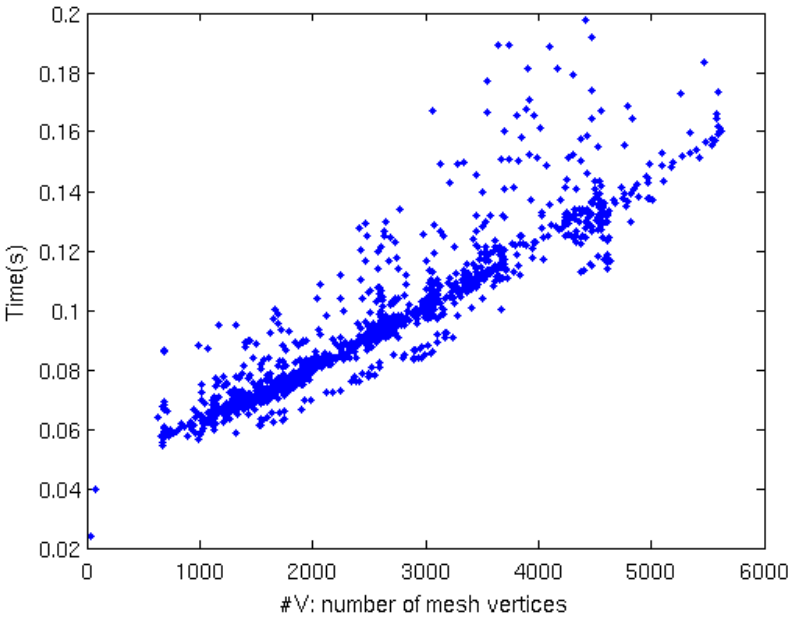


Figure 2.9: Time, in seconds, required to compute a PVHK descriptor as a function of the number of vertices in the mesh.

distance have less than the 5000 vertices here presented, and the descriptor can be computed in less then 0.2s. Larger meshes, e.g., as those from humans, will take a longer time, but still on the order of the second per mesh.

2.6 PVHK Properties

We here address some of the PVHK properties, and the conditions for which the descriptor is informative. Namely, we:

1. provide an estimative and examples of how much do we expect the descriptor of the same partial view to change between observations due to sensor noise;

2. provide an estimative and examples of how much do we expect the descriptor of partial views from similar viewing angles to change;

3. provide an estimative on the condition required for two partial views to have the same descriptor.

2.6.1 Sensor Noise and Perturbations in the Descriptor

We here estimate how much change do we expect in the descriptors of two meshes of the same partial view, M_1 and M_2, identical apart from changes in the coordinates, X_1 and X_2, due to sensor noise.

Our analysis follows the propagation of noise from the sensor to the descriptor:

1. we analyze how the noise in coordinates changes the length of mesh edges and affects the Laplace-Beltrami operator;

2. we use perturbation theory to estimate the relation between the magnitude of the noise in the coordinates and the magnitude of change in the descriptor;

3. we provide the expected value of the descriptor in a noisy situation.

Our main result is that perturbations on the mesh have less and less impact on surface temperature as $t \to +\infty$. Thus, by choosing large stopping times, we ensure that the descriptor is more and more stable. We also estimate that a regular sensor, at 1m from an object, is associated with changes in the temperature on the order of $10^{-3}\bar{T}$, where $\bar{T}$ is the temperature over the boundary, so we have larger perturbations in points where the temperature is higher, near the source, and lower perturbations on the boundary, where we evaluate the temperature.

Impact of sensor noise to the Laplace-Beltrami operator

We assume that each z-coordinate returned by the camera is given by $z = z_0 + z_0^2\varepsilon$, where z_0 is the true distance between a point in the object surface and the camera, and $\varepsilon \sim \mathcal{N}(0, \tau^2)$ is the camera intrinsic random noise variation. This model is described for the Kinect camera in [33] with with $\tau = 1.42 \times 10^{-3} m^{-1}$.

In Appendix A, we show how noise propagates from coordinate z to the distance between vertices and into to the Laplace-Beltrami operator. The main result is that the expected difference between the operator of two meshes, $L^\delta = L_1 - L_2$, is given by: $\left\langle L^\delta \right\rangle = \langle L_1 - L_2 \rangle \propto \eta L_1$, with $\eta \sim \mathcal{O}(z^2\tau^2 f^2)$, where z is the distance to the object and f is the camera focal length. Using typical values for the Kinect camera, with $f \sim 580$ and $z \sim 1m$ we have $\eta \sim 5 \times 10^{-3}$.

Impact of perturbations to the Laplace-Beltrami operator in the temperature

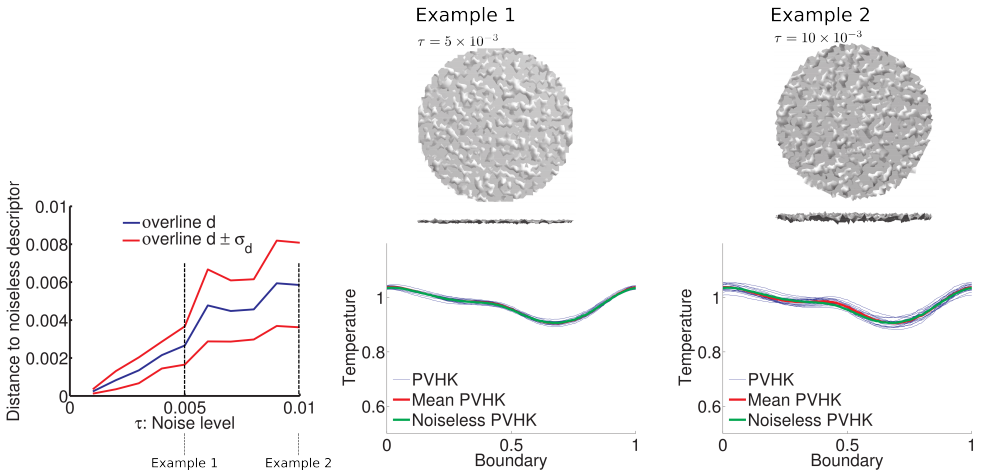
Using first order perturbation theory, [16], we estimate the impact on temperature of perturbations to the Laplace-Beltrami. In Appendix B we show how a small perturbation in the Laplace-Beltrami operator impacts its eigenvectors and eigenvalues and how the perturbation passes on to the temperature.

The main result is that the temperature perturbation, $\bar{T}^\delta(t) = \bar{T}^1(t) - \bar{T}^2(t)$, depends linearly on: (i) L^δ; and on (ii) $\exp(-\Lambda^1 t)$. While the former leads to $\left\langle \bar{T}^\delta \right\rangle \sim 5 \times 10^{-3} T$, the latter implies that:

$$\bar{T}^\delta(t) \xrightarrow[t \to +\infty]{} 0. \tag{2.12}$$

Example

We compare our estimative with the real changes in the descriptor using a synthetic mesh, represented in Figure 2.10. Figure 2.10(a) shows how the descriptor changes with the increase in noise level. The blue line corresponds to the mean distance to the noiseless descriptor, taken over 20 trials. The descriptors were computed at a fixed time instant, $t = 1/\lambda_2$. The red lines correspond to the mean distance $\pm$ the standard deviation and represent the range of change that we can expect in the descriptors. Figure 2.10(b) and Figure 2.10(a) show how noise levels of 5×10^{-3} and 10×10^{-3} in the depth image reflect on the surface mesh and temperature profile at the boundary. In particular, they show how the global shape of the descriptor does not change with noise.



(a) Effect of camera noise level.

(b) Noise level of 5×10^{-3}

(c) Noise level of 10×10^{-3}

Figure 2.10: Impact of noise on the descriptor for a circle at a 1m from the sensor.

2.6.2 Changes in the Viewing Angle

We here estimate how much changes do we expect in the descriptors of two meshes of the same object, M_1 and M_2, obtained from similar viewing angles.

Changes in the viewing angle have a two fold impact on the descriptor:

1. the partial view changes, as occluded parts become visible;

2. the source position changes, as we consider sources that depend on the sensor.

We cannot model the first, as it depends only on the object surface, in the same way that we do not know what happens to the source when the sensor moves. However, we do know that in most regular objects, with compact surfaces, most changes in the viewing angle will result in small changes in the source position over the surface. Furthermore, regular objects, such as a box or a book, have similar partial views when we change the viewing angle, and only occasionally will occluded parts become visible.

Our main result is that under the above conditions, perturbations on the viewing angle lead to perturbations in the temperature at the boundary, which again go to zero when $t \to +\infty$.

We exemplify the impact of changes in the temperature from changes in the source position using the synthetic dataset of four objects from Figure 2.7, with partial views collected from multiple viewing angles.

Changes in the temperature from changes in the source

When the source moves from a vertex v_s to another vertex on its neighborhood, $v_l : (vs, vl) \in E$, the temperature at vertex $v_{b1} \in B$ changes as: $\Delta T_{v_s} = k(v_{b1}, v_s, t_s) - k(v_{b1}, v_l, t_s)$.

If we take the average over all vertices in the neighborhood to where the source can move, and weight with respect to the distance to the neighbor, we arrive at:

$$\tilde{\Delta} T_{v_s} = \sum_{v_l:(v_s,v_l)\in E} (k(v_{b1}, v_s, t_s) - k(v_{b1}, v_l, t_s))/\|\bar{x}_s - \bar{x}_l\|^2 \tag{2.13}$$

$$= L_s T_{v_s} = -\partial_t T_{v_s} \quad \text{(Eq. 2.4 and Eq. 2.1)} \tag{2.14}$$

$$= \sum_{k=2}^{N_s} \lambda_2 [\phi_i]_{v_s} [\phi_i]_{v_{b1}} e^{-t_s \lambda_i} \tag{2.15}$$

where L_s is the row of the Laplace-Beltrami operator L. Again, when time increases, as all λ_i are positive, changes in the descriptor due to changes in the source position go to zero.

Example

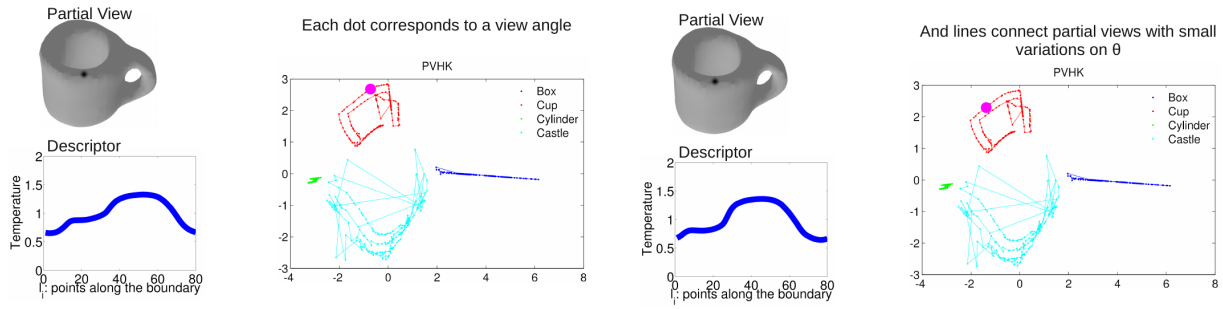
We here provide an example of the set of partial view descriptor of the four objects in Figure 2.7. We show in Figure 2.11 an Isomap projection of the descriptors from a smooth sequence of viewing angles retrieved from the four objects.

We use the mug as an example of the impact of changes in the viewing angle in the descriptor. At each figure, we show the partial view associated with the viewing angle, with the source[1] marked in black. We also present the descriptor of each partial view, and we mark its respective position in the Isomap.

Finally, we note that nodes in the Isomap are connected to neighboring viewing angles, and that there is a relation between similar viewing angles and similar partial view descriptors. This is particularly clear when we look at the sequence of mug descriptors and their position in the Isomap.

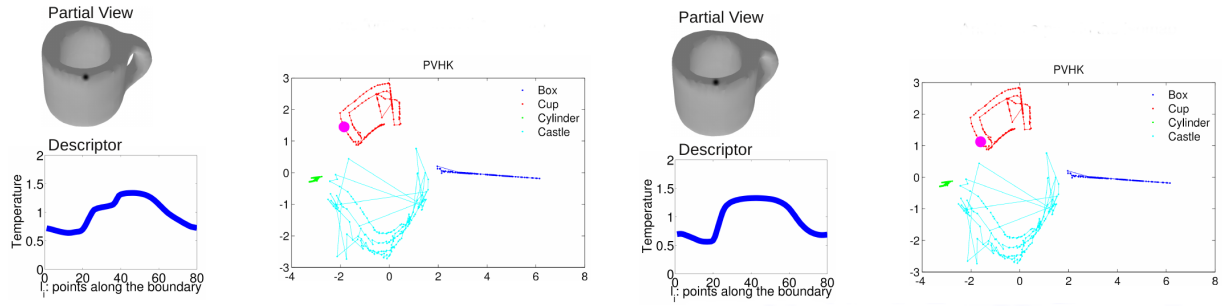
The set of descriptors associated with the cylinder does not display these properties, as its shape does not change.

[1] In this experiment, we chose the source as the point closest to the observer.



(a) t_1

(b) t_2



(a) t_3

(b) t_4

Figure 2.11: 2D Isomap projection applied to the set of objects in Figure 2.7.

2.6.3 Uniqueness of the Boundary Temperature

Let M_1 and M_2 be two generic meshes. We want to know if M_1 and M_2 can be different, even when their temperature profile obtained at a fixed time instant, t_s, over a subset of vertices on the mesh boundary, $v \in B$, is the same.

We know that if temperatures over all points, at all time instants, in M_1 and M_2 are the same, then the two meshes are identical, [48]. However, we only have a subset of vertices at a fixed time instant.

In general it is difficult to define conditions for which two identical temperatures would necessarily imply the same surface. However, we here show sufficient conditions under which the descriptors are the same, regardless of surface geometry. We then provide intuition on why these conditions are unlikely to hold often.

Assuming that we have two different meshes, M_1 and M_2, each with its Laplace-Beltrami operator, L_1, L_2, two surfaces will have the same descriptor if:

$$\bar{z}_1 = \bar{z}_2 \Leftrightarrow \Phi_B^2 \bar{c}_1(t_s) = \Phi_2^B \bar{c}_2(t_s), \tag{2.16}$$

where $\Phi^{1,2} = [\bar{1}, \bar{\phi}_2^{1,2}, \bar{\phi}_3^{1,2}, ...]$ are two collections of orthogonal vectors, resulting from the eigendecomposition of symmetric matrices, the Laplace-Beltrami operators L_1 and L_2. Furthermore, by

how they were defined, both L_1 and L_2 share a common eigenvector $\bar{1}$, associated with the eigenvalue $\lambda_1 = 0$. $\Phi^B_{1,2}$ are subsets of rows of $\Phi_{1,2}$ corresponding to boundary vertices. The space of all possible descriptors, i.e., considering all possible sources and stopping times, t_s, for both M_1 and M_2 are spanned by Φ_1, Φ_2 respectively. When we fix the source and t_s, we define the coordinates of each descriptor in the two spaces: $\bar{c}_{1,2}(t_s) = e^{-\Lambda_{1,2}t_s}\Phi^T_{1,2}\bar{T}(0)$.

The condition in Eq. 2.16 holds at least in two situations. The first is when the two spaces spanned by Φ_1 and Φ_2 intersect at exactly $\bar{c}_1(t_s)$ and $\bar{c}_2(t_s)$.

The intersection is unlikely if the set of possible descriptors, generated by the above vectors $\bar{c}_{1,2}$, is very sparse. So, first their l_0-norm must be large, to ensure that they are spreaded around, but the number of values that they can take has to be reduced.

Given the exponential $e^{-\Lambda_{1,2}t_s}$, the norm of $\bar{c}_{1,2}$ clearly decreases with time, so t_s must be the smallest possible. Here we note that by fixing $t_s = 1/\lambda_2$, and provided that $\lambda_2 \sim \lambda_3, \lambda_4, ..., \lambda_m$ and that $[\bar{\phi}^i_k]_{v_s} \neq 0 \forall_{k=1,...,m}$, $\bar{c}_{1,2}$ have a large enough dimension, reducing the probability of an intersection.

We also ensure that the number of accessible values is reduced by considering that the initial condition in Eq. 2.5 is zero everywhere and N for entry v_s, we further constrain $\bar{c}_{1,2}$.

The second situation is when $c_{1,2}$ becomes orthogonal to $\Phi^{1,2}_b$, which happens, e.g., when $t_s = 0$, which we never consider throughout this work.

In Chapter 5, when analyzing the impact of stopping time in the descriptor, we will revisit this problem in detail. Here we just point out that, while large values of t_s lead to a noise robust descriptor, they also yield a less discriminative one.

2.7 Summary

In this chapter, we introduced the Partial View Heat Kernel (PVHK) descriptor, for representing the visible surface of an object as returned by a depth sensor, such a Kinect camera. While the sensor provides rich information, the 3D information is often noisy. We here showed how to compute the PVHK descriptor, and how to incorporate different information types onto the 3D description. In particular, we showed how to incorporate the RGB information provided by the sensor with the 3D surface.

We have also showed how the descriptor provides a noise resilient descriptor of the partial views, which makes it ideal to represent noisy surfaces.

Chapter 3

Partial View Recognition

In this chapter, we address the problem of identifying a partial view by comparing a PVHK descriptor with those stored in an object library. We first define the distance metric we use to compare partial views descriptors. In Section 3.3 we show the descriptor and metric effectiveness on the recognition of real everyday objects, of similar sizes but distinct shapes. In Section 3.4 we show how using color allows to disambiguate between same class objects, which share similar geometries. In Section3.5 we show how we can use PVHK, and C-PVHK, in non rigid shapes such as humans. Finally, in Section 3.6 we compare the performance of the PVHK with other partial view descriptors.

3.1 Recognizing Objects Using PVHK Descriptors

To recognize the object class of a partial view, we compute the PVHK descriptor of that partial view and then to compare it against previously labeled partial view descriptors, stored in an object library $\mathcal{O}$.

We represent objects in the library as sets of partial views, corresponding to the visible surface of the object, as seen from multiple viewing angles, as represented in Figure 3.1. Its partial view is labeled with respect to the sensor position in the object coordinate system.

It is through the object library that we map each object and viewing angle with descriptors. Formally:

Definition 1. An object library, $\mathcal{O} = \{(s_1, \bar{z}_{s_1}), (s_2, \bar{z}_{s_2}), ..., (s_{s_M}, \bar{z}_{s_{N_\theta}})\}$, is a set of tuples $(\mathcal{S}, \mathbb{R}^M)$ that maps a descriptor $\bar{z} \in \mathrm{R}^M$ to a partial view label $s \in \mathcal{S}$.

As partial views are defined by an object and a sensor viewing angle in the object coordinate frame, we label each partial view as $s = (o, \bar{\theta})$.

In this chapter we use different libraries to highlight different aspects of the PVHK, namely:

- **Library-I** composed of Real Rigid Objects, provides empirical evidence on the accuracy of our representation using sensor data;

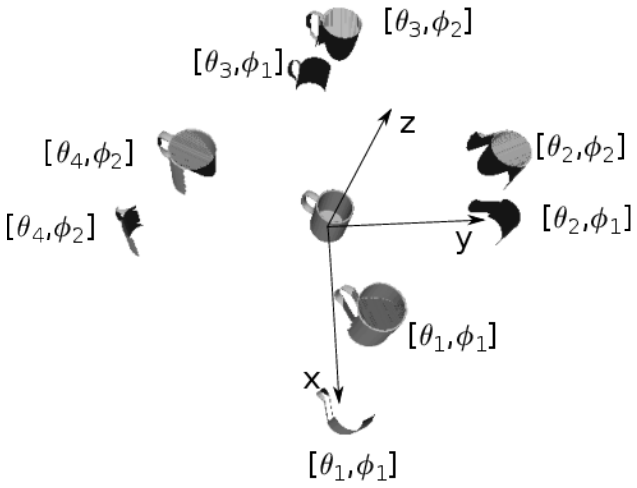


Figure 3.1: Objects in the library are represented by multiple partial views, each associated with the sensor viewing angle in the object coordinate system.

- **Library-II** composed of Real And Colorful Objects, retrieved from [35], illustrates the use both color and 3D information and applicability of the descriptor on rigid objects;
- **Library-III** composed of Non rigid Objects, with and without the respective RGB information, illustrates the applicability of the descriptor on non rigid objects;
- **Library-IV** composed of partial views rendered from CAD models, and previously presented in Figure 2.7.

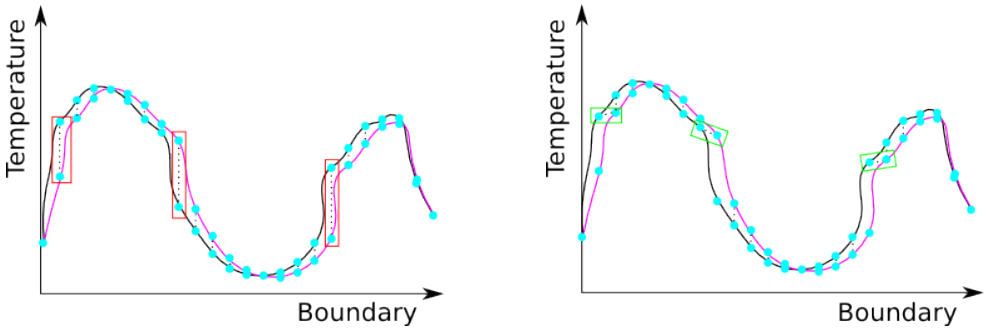
For all recognition tasks, we assume a nearest neighbor classifier. I.e., we search among all partial views in the object library by the closest to the testing partial view and assume they belong to the same object. We define the closest object based on a Modified Hausdorff distance, which provides a relevant distance between descriptors.

3.2 Distance Between Partial Views

We define the distance between two partial views as the distance between their descriptors. However, due to noise, partial views of the same object, seen from the same viewing angle by a noisy sensor, generate necessarily different descriptors.

The sensor noise affects not only the boundary temperature but also the boundary points where we compute the temperature, i.e., the boundary parameterization. If we consider the descriptor as the temperature at $1/M$ intervals of the boundary length, changes in the length caused by noise lead to changes in the vertices where those intervals start and end.

Moreover, changes in boundary parameterization may lead to drastic changes in vector norms, e.g., l_1 or l_2, as illustrated in Figure 3.2(a). In the example, while both descriptors share the same shape, there is a small shift in the boundary. In regions of rapid change in the descriptor, the small shift results in large differences in temperature and thus in large distances between descriptors.



(a) Error only on the temperature (b) Error on both temperature and boundary

Figure 3.2: Two approached for comparing descriptors assuming different sources of error.

Thus, we compare two descriptors using the modified Hausdorff distance, which provides a measure of the difference between the two curves in the graphic. As illustrated in Figure 3.2(b), when computing the Hausdorff distance we compare each point in one curve with its closest on the second curve. Thus small shifts in boundary length will have a small impact on the distance.

To compute the distance between two descriptors, we first represent each as curves in 2D, i.e., the descriptor $\bar{z} \in \mathbb{R}^M$ becomes a set of points $\eta = \{[1/M, [\bar{z}_1]_1], [2/M, [\bar{z}_2]_2], ..., [1, [\bar{z}_M]_M]\}$.

Then, we estimate the distance between two observations using Eq.3.1.

$$d(\bar{z}, \bar{z}') = d_{\text{MH}}(\eta, \eta') = \min \left\{ \sum_{x \in \eta} \inf_{y \in \eta'} \|\bar{x} - \bar{y}\|^2, \sum_{y \in \eta'} \inf_{x \in \eta} \|\bar{x} - \bar{y}\|^2 \right\} \tag{3.1}$$

We summarize the steps required for computing the distance between two partial view descriptors $\bar{z}_1$, $\bar{z}_2$ in Algorithm 3.1.

Algorithm 3.1: Computing distances between PVHK descriptors.

Input: PVHK descriptors, $\bar{z}_1$ and $\bar{z}_2$
Output: $d(\bar{z}_1, \bar{z}_2)$
CONSTRUCT CURVE:
$\eta_{1,2} \leftarrow \{[1/N, [\bar{z}_{1,2}]_1], [2/N, [\bar{z}_{1,2}]_2], ..., [N, [\bar{z}_{1,2}]_N]\}$:
COMPUTE HAUSDORFF DISTANCE:
$d(\bar{z}, \bar{z}') \leftarrow d_{\text{MH}}(\eta_1, \eta_2)$ (from Eq. 3.1)

3.3 Identifying Real Objects

We demonstrate the effectiveness of PVHK on object recognition tasks from 3D partial views using Library-I. The library is composed of 13 regular objects, with compact surfaces, of similar sizes but with different and without RGB values.

3.3.1 Library-I

With a Kinect camera, we collected two sets of partial views, for training and testing respectively, of 13 rigid and similar size objects, represented in Figure 3.3. Moreover, the partial views for each object correspond to a known and dense sampling on the observer orientation, $\theta \in [0^o, 360^o]$.

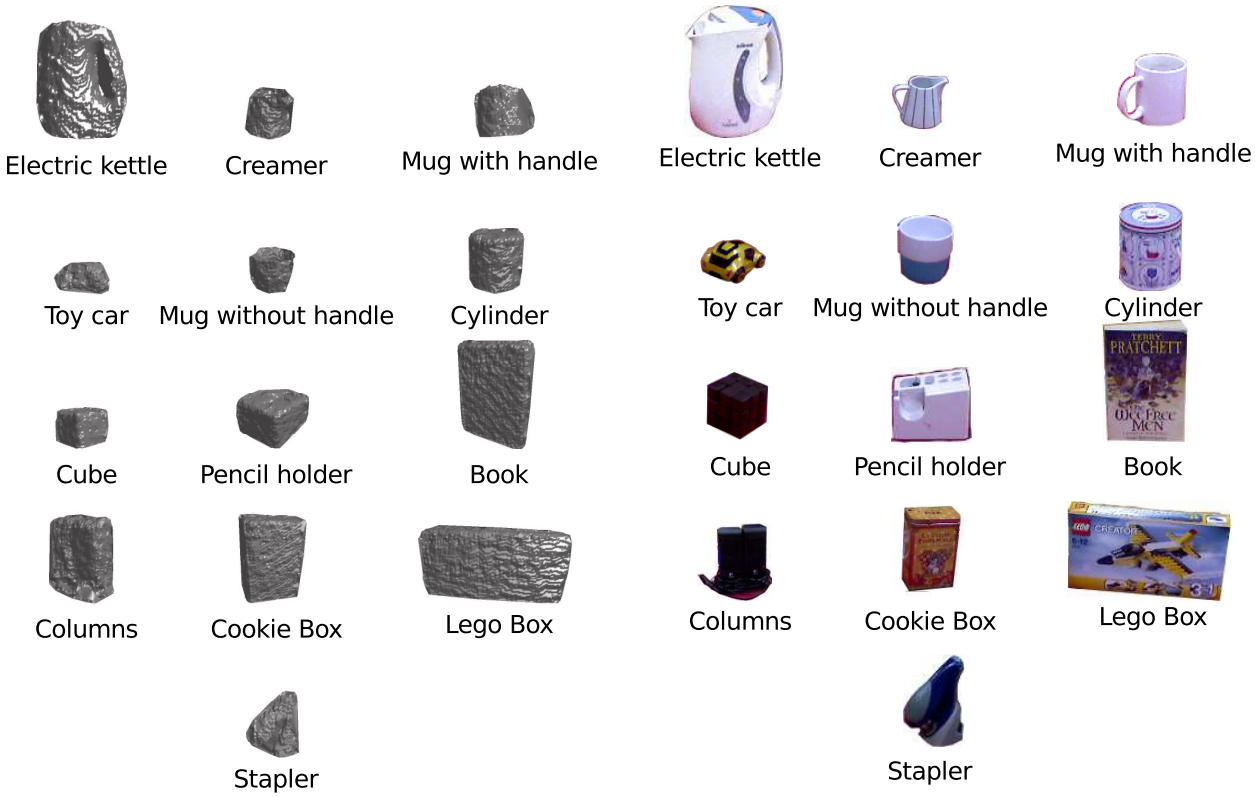


Figure 3.3: Dataset of small objects grasped by a Kinect sensor.

Figure 3.4 represents the acquisition and labeling apparatus. We placed the objects individually in a red cardboard, so that we could easily segment the background. Furthermore we used QR-codes and the Aruco library[25] to define the orientation of the cardboard with respect to the observer.

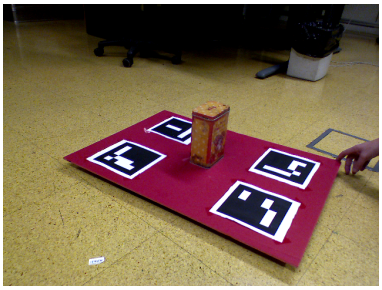


Figure 3.4: Acquisition setup for Library-II. Objects are placed on a red cardboard, for background segmentation, together with QR-codes for orientation estimation with the Aruco library.

Object	Training	Testing	Object	Training	Testing
Electric Kettle	434	306	Creamer	904	471
Mug with handle	777	374	Toy car	935	535
Mug without handle	1041	524	Cylinder	660	269
Cube	1039	448	Pencil holder	681	398
Book	514	292	Columns	675	302
Cookie box	778	401	Lego box	884	430
Stabler	970	459			

3.3.2 Experimental Results

Figure 3.5 highlights the individual partial view results for PVHK using a confusion matrix that relates the true viewing angle of each element on the testing dataset, on the x-axis, to the viewing angle of the closest descriptor from the training dataset, on the y-axis. The confusion matrix shows that a large percentage of miss classifications results from confusion between similar objects, e.g., the cream pitcher and the mug. Besides the miss-classification of object category, the matrix shows also the inner category confusion that we expect in objects with strong symmetries, such as those used in the dataset. The overall accuracy was 95% and the accuracy for each class is represented in the column to the right of the matrix.

We note that the two objects with a larger confusion among them are the creamer and the mug with a handle, which are very similar. Also, we note that in objects such as mug without handle, there is a large confusion within the viewing angles, which is expected as the object is symmetric with respect to changes in viewing angle. A similar effect can also be observed in the Lego box, where we can see that there is a strong confusion between two sets of viewing angles, which correspond to the box symmetry.

3.4 Disambiguation Through Color

When objects have very similar shapes, we can distinguish between them using color or texture. We here show how the color extension of the partial view heat kernel allows to disambiguate different

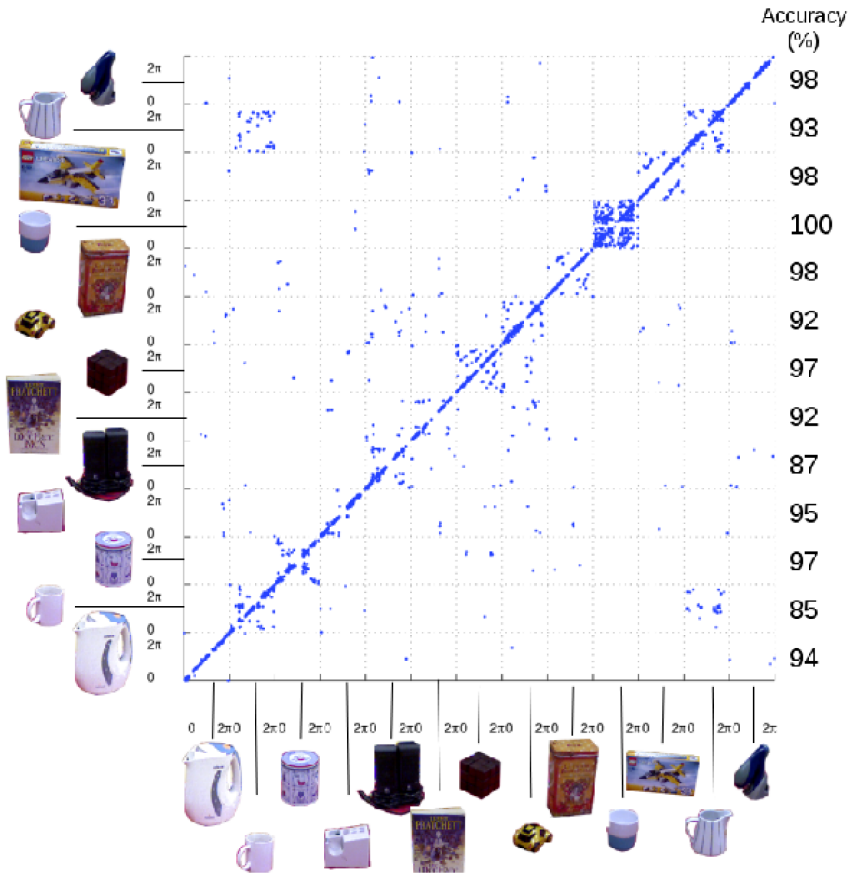


Figure 3.5: Confusion matrix for PVHK testing

instances within four different classes of small real objects.

We recall that C-PVHK is computed as the solution of Eq. 2.10, that we here re-write:

$$C^{-1}L\bar{f}(t) = -\partial_t \bar{f}(t) \tag{3.2}$$

and recall that C is a diagonal matrix, whose entry $[C]_{v,v} = c(v)$ provides a scalar representations of color as different diffusion rates.

We evaluate the use of color by comparing the performance of C-PVHK with that of PVHK. We thus experiment different maps from the RGB values, provided by the sensor, to the scalar $[C]_{v,v}$. This map can take many forms, and we could think of specially tailored maps for any given library. Here we focus on simple experiments, which show how choices on $[C]_{v,v}$ impact the descriptor performance. In particular, we are interested in understanding if smaller values of $[C]_{v,v}$ would have any impact on recognition. We expect that the results will help modeling future maps.

Finally, we also consider the impact of using more or less partial views on the object library.

Our experiments show that, by indexing color to the geometry, we improve recognition results.

We also show that if we map color so that $[C]_{v,v}$ takes small values, the impact on recognition is not significant. Finally, we concluded that the number of partial views in the object library is of utmost importance for recognition.

From the objects used, there was one that showed particularly poor recognition scores, which resulted from a large variability of the object surface, with considerable changes to the boundary. In this situation, the use of color could not improve the recognition results.

3.4.1 Library-II

We used all instances of four different objects from a publicly available RGB-D dataset [35]. We selected objects with different shapes; that presented significant changes in color and texture. In particular, we used: all the food cans, the cereal boxes, the instant noodles packages, and the shampoo packages. Figure 3.6 shows all the different objects we used.

Figure 3.6: Objects in Library-II, composed of 32 objects divided in four classes.

We considered libraries with 35, 20, 15, 10 and 5 partial views per object. These partial views were equally distributed over the angle θ. All the other partial views were used for testing.

3.4.2 Experimental Results

We evaluate the performance of the both descriptors over 16 different experiments, covering four different scalar functions $[C]_{v,v} = c(v) : \mathbb{R}^3 \to \mathbb{R}$, and four different library sizes. The scalar functions we tested were: $c_1(v) = (h(v)+1/2)\times 2$, $c_2(v) = (h(v)+10^{-3})\times 2$, $c_3(v) = (h(v)+5)\times 2$, $c_4(v) = (h(v)+1/2)\times 10$, where $h(v)$ is the hue value of the pixel. Their co-domains differ in the

lower and upper bounds, as well as range.

Figures 3.7 (a)-(d) present the results for each color combination as a function of the size of the testing library. The results are aggregated by class, representing the precision over all the instances of each class.

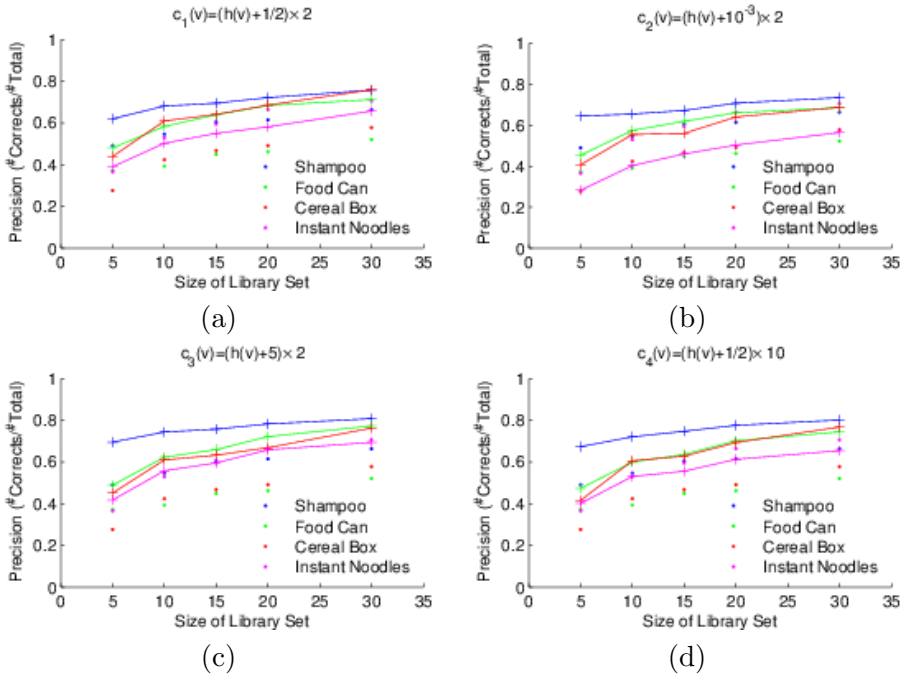


Figure 3.7: Global precision for different scalar functions. Dots correspond to results using PVHK, lines correspond to results using C-PVHK.

In all the experiments, the use of color clearly improved precision results. The results also improved with library size, which is expected considering that we have a better coverage of all possible descriptors associated with each object. Finally, results also hint to no direct relation between the range of values that $c(v)$ can take and precision. However, using small values of $[C]_{v,v}$ clearly affects the results.

Figures 3.8(a)-(d) show precision results for each object in the library using the scalar function $[C]_{v,v} = c_3(v)$.

We see that not all objects are sensitive to the library size, e.g., instances of the shampoo class present similar precisions regardless of library size. Also, some instances of the *Instant Noodle* class clearly present a low precision, in particular, the object with the label 1. In Figure 3.9(a), we show different partial views of this object, separating them between those that were correctly classified and those that were incorrectly classified. What we notice is that the change in shape between viewing angles is considerable. Thus, adding color to the representation just changes the descriptor in a non expected way, yielding it more similar to other objects.

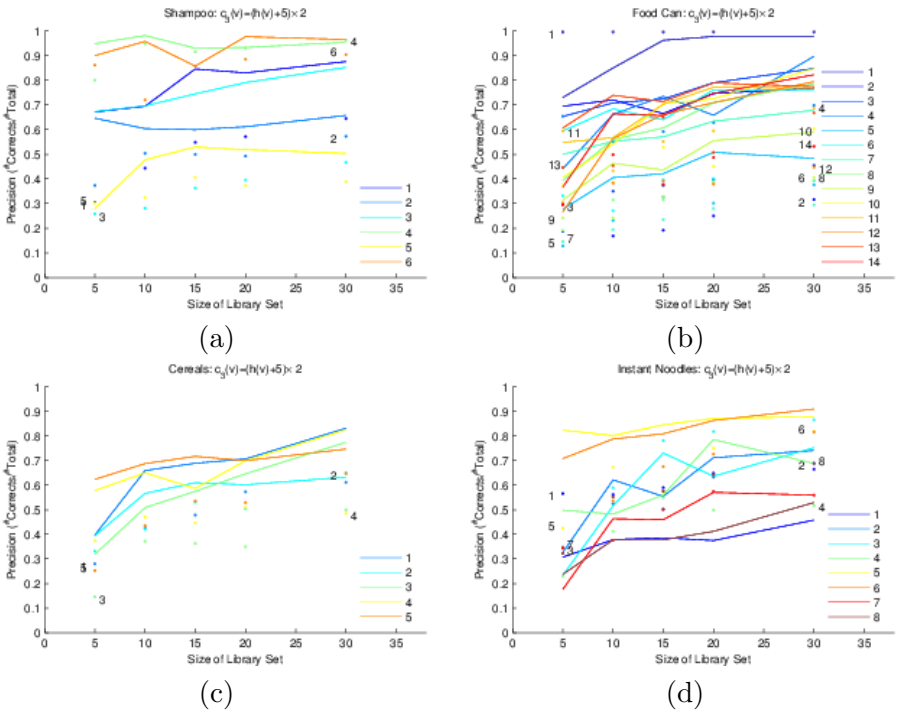
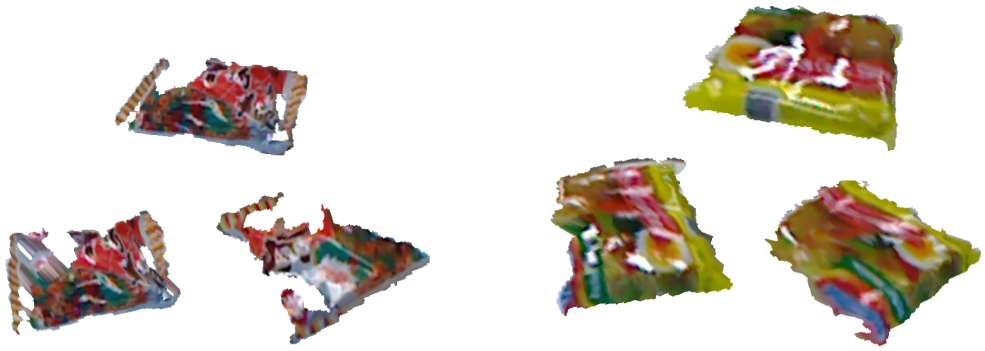


Figure 3.8: Precision per object using c_3. Dots correspond to results using PVHK, and lines of the same color correspond to results using PVHK-C.



(a) Instant Noodles 1 (b) Instant Noodles 6

Figure 3.9: Examples of partial from two objects in the instant noodles library. (a) is the object with label 1 in Figure 3.6 and in Figure 3.7(d), and (b) is the object with label 6.

3.5 Non Rigid Shapes

Deformations in both body and clothes shape raise important challenges for human tracking using 3D descriptors and affect the efficiency of representations aimed for rigid shapes. However, the heat kernel is invariant to surfaces isometric changes, which means that PVHK will also be resilient to

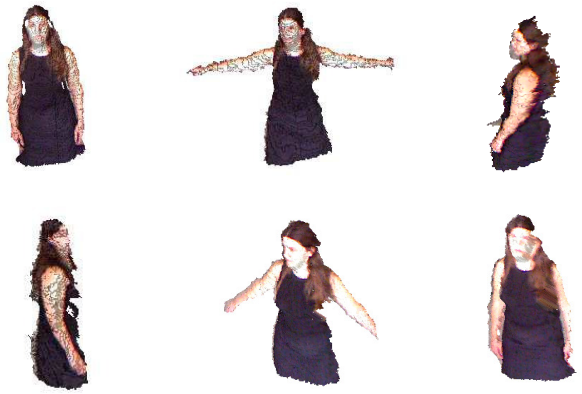
most deformations.

We here show that the PVHK changes mostly when there are considerable changes in the body shape, e.g., when arms move away from the body. These drastic changes in the shape lead to important changes in the descriptor, and are more pronounced than those caused by moving the arms around when they are already away from the body or those caused by walking with the arms next to the body.

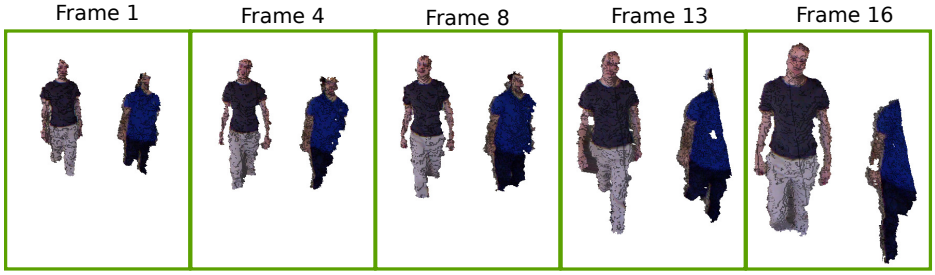
On the other hand, the body shape itself is too similar across individuals to allow recognition using the PVHK. Thus, we again use C-PVHK and show that we can distinguish between them.

3.5.1 Library-III

For the purpose of showing how does the descriptor changes with deformations in body shape, and how we can use color to distinguish between individuals, we introduce two sequences of humans moving around. The first sequence represents a human moving around in a room, and purposefully changing the body shape between the three main positions showed in Figure 3.10(a). The second sequence consists of two humans moving side by side, as showed in Figure 3.10(b).



(a)



(b)

Figure 3.10: Sequences of humans moving freely in a room.

3.5.2 Experimental Results

Results show that the first sequence generated two distinct groups of descriptors, depending on whether arms were close or separated from the body. The groups are visible in Figure 3.11, where we represent a 2D Isomap projection of the descriptors collection and their and respective labels.

This means that we can represent an articulated body by a reduced number of rigid shapes and thus easily perform tracking and recognition tasks.

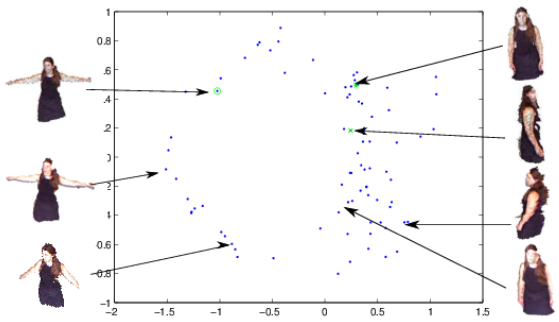


Figure 3.11: 2D Isomap projection for a human moving.

The results on the second sequence show that it is impossible to recognize between two individuals using just the PVHK. But again we distinguish between the two humans using PVHK-C descriptor. Figure 3.12 shows the two distance matrices. Furthermore, by being insensitive to the low level details of face features, PVHK allows for anonymously tracking humans in a contained environment.

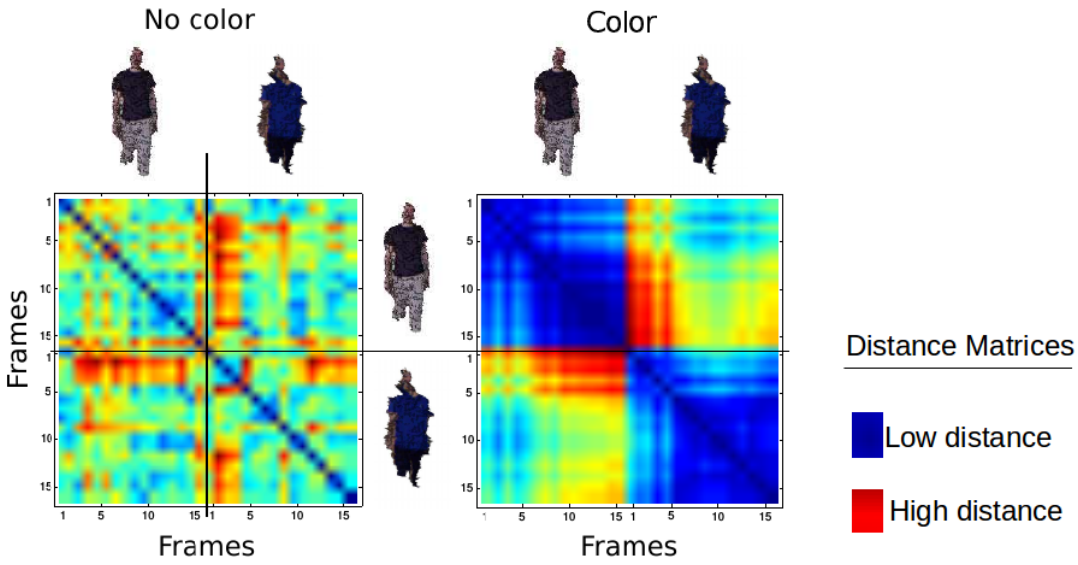


Figure 3.12: Confusion matrix between the humans in the frames with and without color.

3.6 Comparing with Other Descriptors

We evaluate the performance of our descriptor when compared with three existing descriptors:

1. the Scale Invariant Heat Kernel Signature (SI-HKS) [14];
2. the Viewpoint Feature Histogram (VFH) [54];
3. the Ensemble of Shape Features (ESF) [70].

We evaluate the performance of the three descriptors on Library-IV, composed of the four objects presented in Figure 2.7. The results show that the ESF and PVHK represent the four objects in such a way that there is little confusion between descriptors of different objects. We then show that, while ESF performs well in the recognition of multiple objects, the PVHK is more suitable for the representation of objects with large surfaces, such as the cereal boxes in Library-II.

3.6.1 Brief Description of other partial view descriptors

From the three descriptors, the VFH and ESF, are retrieved from the PCL library [55], and were specifically introduced for the representation of partial views. We implemented the SI-HKS descriptor following [14].

Viewpoint Feature Histogram

The VFH is a histogram of changes in surface normals orientation, with respect to a an averaged normal, computed at a central point in the surface.

Ensemble of Shape Functions

The ESF is a set of histograms of shape functions: i) distances between randomly selected vertices; ii) area of triangles formed by randomly selecting three vertices; iii) angles of those triangles, Furthermore, to increase the discriminative power of the descriptor, each of these are separated by whether the path between the two randomly selected vertices is over the surface, outside the surface, or part over and part outside. So, for each of the above three shape functions, there are three histograms, one for each type of path. A fourth function, and respective histogram, further discriminate mixed paths by the ratio between the length outside and inside the object surface.

Scale Invariant Heat Kernel Signature

The Scale Invariant Heat Kernel Signature corresponds to the absolute value of the Heat Kernel Signature time Fourier transform. The Fourier transform represents changes in the object scale as a change in phase, which is then discarded by taking the absolute value. Geometric words are then identified by clustering, using k-means [6], the SI-HKS features extracted from all surface points in object surfaces. Each partial view is then represented by the distribution of visual features present. We note that both the Heat Kernel Signature and the Scale Invariant Signature, depend on the complete shape of the object, and thus the same geometric feature will depend on the partial view.

3.6.2 Results in Library-IV

In Figure 3.13, we represent an Isomap projection for the set of objects and descriptors. As in Figure 2.11, dots correspond to partial views, and connected dots are contiguous view angles.

From the projections we see that ESF and PVHK are more effective at separating objects, since partial views from different objects do not get mixed in a 2D projection. However, ESF does not change as smoothly with the view angle as PVHK, notably in the cup and the castle example. In fact, the ESF depends only on the surface shape, and not on sensor position. Thus, in regular objects, such as boxes, where the shape does not change considerably with variations on the sensor position, the ESF provides no insight on the viewing angle.

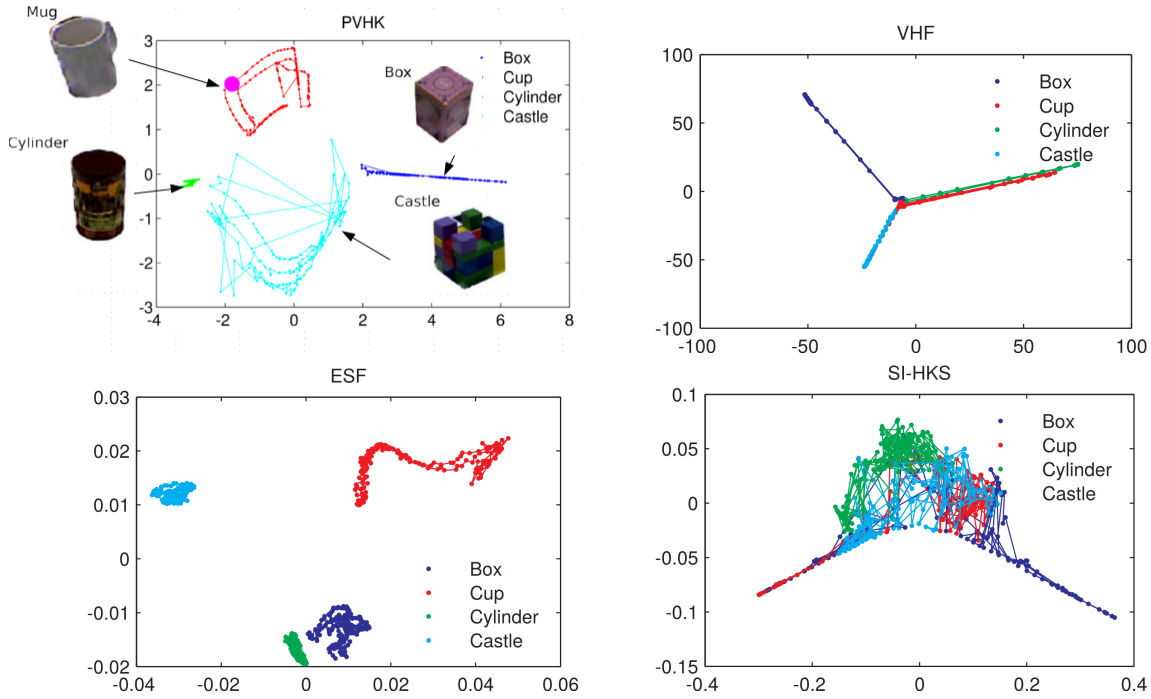


Figure 3.13: 2D Isomap projections of the descriptor from four partial view representations

As acknowledge in [14], the SI-HKS was thought for complete 3D objects, and is strongly affected by missing object parts, as the heat kernel depends on the complete surface shape. So, its performance in this library of partial views is expected.

The PVHK descriptor performs as well as the ESF in the above dataset, however the PVHK is more suitable for representing objects composed of large planar surfaces, such as the Cereals Boxes in Library-II. The results showed in Figure 3.14, show that when we want a higher accuracy, the PVHK performs better than the ESF.

While ESF performs very well on many objects, it is sensitive to changes in the object topology. The ESF separates each shape function in three histograms that depend on whether the path between two points lays over the surface or not. Points collected over a plane will contribute only

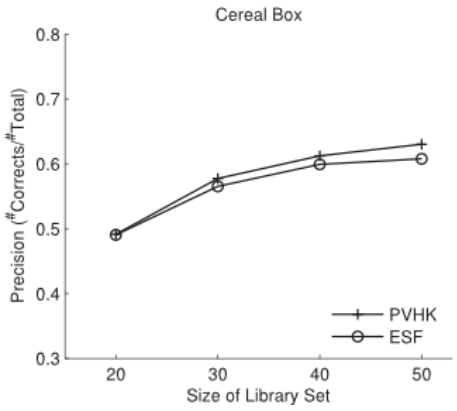


Figure 3.14: Comparison between ESF and PVHK on Library-II objects.

for one of those histograms, as all the paths connecting them lay over the surface. As illustrated in Figure 3.15, when we introduce a hole in the center of the plane, paths will leave the surface, and the shape function for that path counts towards a different histogram. In the example, we note that while the path $V_1 - V_2$ and $V_1 - V_3$ belong to the same plane, they will contribute to different regions of the descriptor. We note that the impact is not felt so strongly on the other shapes, as the histogram for the planes, is less relevant for the shape description.

Figure 3.15: Impact of surface holes on ESF descriptors of planar surfaces.

3.7 Summary

From classification results using datasets of real objects, we show the PVHK potential for (a) discriminating everyday objects of regular sizes and similar shapes and (b) tracking humans.

PVHK is specially suitable in situations with no occlusion from other objects. We thus foresee a large spectrum of applications for PVHK, ranging from robot manipulation, where in front of the robot is only the target object, to robot controlled perception, where the robot can intentionally move to avoid occlusions.

Furthermore, C-PVHK represents color distributions over geometry. This opens the door to many other applications where we need to differentiate objects with the same geometry, from which we highlight the possibility to identify boxes in a supermarket or anonymously tracking humans.

When compared with other descriptors, the PVHK provides a better accuracy than other heat based descriptors, the SI-HKS, and than the VFH. It also performs better than the ESF on objects composed mainly of planar surfaces, where the presence of holes impacts the ESF strongly.

Chapter 4

Incremental Object Recognition

In this chapter, we address the problem of recognition from multiple viewing angles. Often it is not possible to recognize objects from a single partial view with a large certainty. In particular, as showed in [12, 52], recognition from a single partial view is difficult when: i) objects are similar and ambiguous from at least some viewing angles; ii) object libraries are sparse, in the sense that the number and the quality of the partial views kept in the library is not representative of the object. When the agent observing the objects is a mobile robot, it can collect multiple partial views to disambiguate or validate initial guesses. The challenge is to efficiently combine the set of observations into a single classification. We approach the problem with a multiple-hypotheses filter that combines information from a sequence of observations given the robot movement. We further innovate by off-line learning neighborhoods between possible hypotheses based on similarity between observations. Such neighborhoods translate directly the ambiguity between objects and allow to transfer the knowledge of one object to the other. In Section 4.2 we introduce the problem of combining multiple observations, without knowing the viewing angle from where each was retrieved. In Section 4.3 we introduce the appearance models required to estimate the class of each partial views. Finally in Section 4.4 we introduce our Multiple View Multiple Hypotheses algorithm and in 4.5 we evaluate its performance in different datasets.

4.1 Ambiguous Objects

We assume a mobile robot, equipped with an RGB-D sensor, that collects partial views of an object as illustrated in Figure 4.1.

Furthermore, we start by assuming that the robot only expects to find two objects in his environment: a mug with a handle and a mug with no handle. We show in Figure 4.2 the object library for the two objects: the 3D shapes correspond to selected partial views and the colors correspond to the temperature over the surface at $t = t_s$. We recall that an object library, $\mathcal{O} = \{(s_1, \bar{z}_{s_1}), (s_2, \bar{z}_{s_2}), ..., (s_{s_M}, \bar{z}_{s_{N_\theta}})\}$, maps a descriptor $\bar{z} \in \mathrm{R}^M$ to a partial view label $s = (o, \bar{\theta}) \in \mathcal{S}$.

The graphic associated with the 3D shapes corresponds to the PVHK descriptor, $\bar{z}$. In the

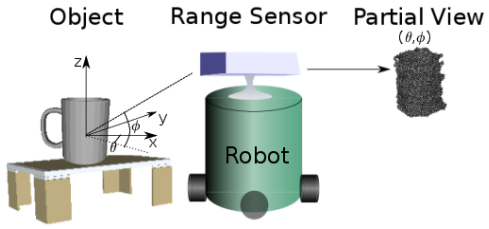


Figure 4.1: A mobile robot capturing a partial view of a mug from the viewing angle $\bar{\theta} = (\theta, \phi)$.

center, we represent the full set of descriptors, each associated with a viewing angle, and use color to represent temperature, so that red corresponds to warmer regions and blue to colder ones.

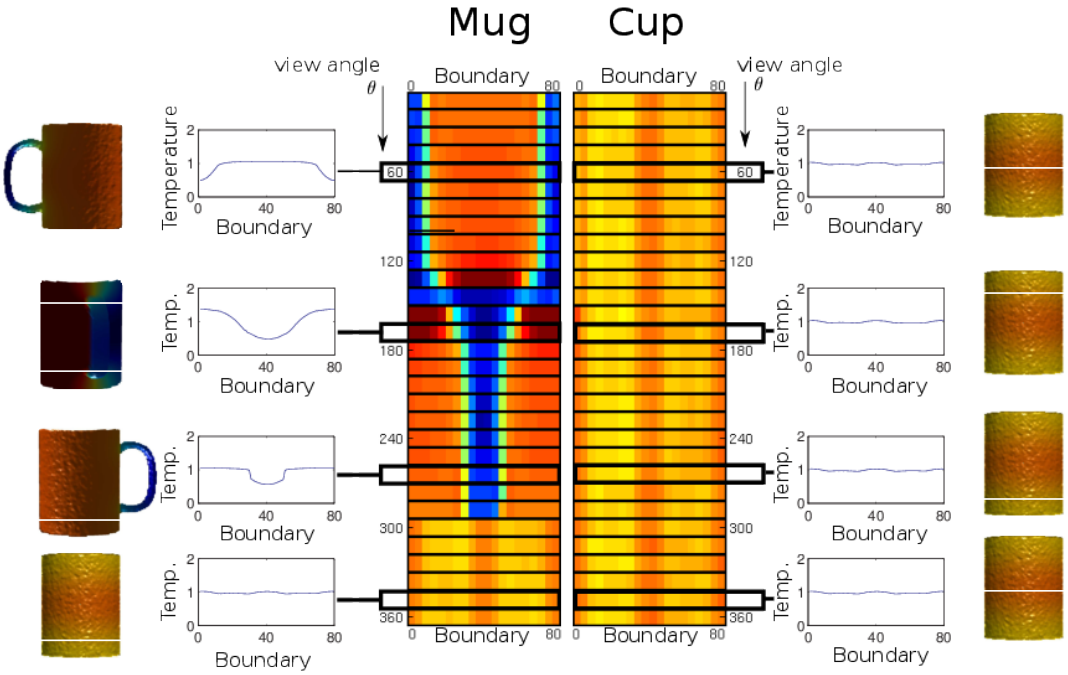


Figure 4.2: Mug and cup library of partial views.

The descriptors on library 4.2 can be separated in four categories. The first corresponds to shapes where the handle is on the left side. The second, associated with shapes where the handle is facing the observer. The third, to shapes where the handle is on the right side. Finally, the fourth represents shapes with no handle, corresponding to the cup and some viewing angles of the mug.

The partial view that the robot observes in Figure 4.1 does not have a handle, and thus the robot cannot distinguish between the two possible objects. In this chapter, we propose to address this problem by having the robot moving around the object while updating at each instance the belief on the object class.

4.2 Recognizing Objects from Multiple Views

We here provide an algorithm to identify an object among similar ones by gathering contiguous observations, assuming that the robot has no previous knowledge on:

- the number of observations;
- the initial viewing angle;
- the sequence of viewing angles.

We propose a probabilistic approach to handle the arbitrary sequence of observations. Formally, given a library of know objects, $\mathcal{O}$, we estimate the object class, $\hat{o}$, from n observations $Z_{1:t} = \{\bar{z}_1, ..., \bar{z}_t\}$, $\bar{z}_i \in \mathbb{R}^M$, of the same object as seen from a sequence of n viewing angles, $\Theta_{1:t} = \{\bar{\theta}_1, ..., \bar{\theta}_t\}$, $\bar{\theta}_i \in [0, 2\pi] \times [0, \pi]$, as the object $o \in O$ maximizing the a-posteriori probability $p(o|Z_{1:t}, \Theta_{1:t})$.

We assume that the robot has access, through odometry measurements, to changes in the viewing angle, $\bar{\Delta}_t$. Thus, while the initial viewing angle $\bar{\theta}_{init}$ is not known, we compute the a-posteriori probability by marginalizing with respect to the initial viewing angle and define our estimator as:

$$\hat{o} = \arg\max_{o} \sum_{\bar{\theta}_{init} \in [0,2\pi]\times[0,\pi]} p(o, \bar{\theta}_{init}|\bar{\Delta}_{1:t-1}, Z_{1:t}). \tag{4.1}$$

Modeling the robot movement and observations as a Markov process, we can simplify the a-posteriori probability in Eq.4.1 by using appearance models, $p(\bar{z}|o, \bar{\theta})$, as building blocks. The appearance models map each partial view defined by an object o and viewing angle $\bar{\theta}$ to possible observations $\bar{z}$. By off-line learning these models, the robot can compute $\hat{o}$ during execution with little cost.

Nevertheless, we would still need to perform a dense search over all the possible initial partial views of all the objects. As there might be possibly infinite partial views, we sample hypothetical initial robot orientations. To propagate these initial hypotheses, we propose a formulation based on the Sequential Importance Resampling Filter, also known as a particle filter, in a Markovian setting, [2]. These filters estimate the a-posteriori by defining a set of hypothesis, called particles. Using the sampling of the search space we can approximate the a-posteriori probability in Eq. 4.1 at each time instant as:

$$p(o, \bar{\theta}_{1:t}|\bar{\Delta}_{1:t-1}, Z_{1:t}) \approx \sum_{i=1}^{N_p} w_t^i \delta\left(s - s_t^i\right) \tag{4.2}$$

where each weight, w_t^i, is associated with a particle $s_t^i = (o^i, \bar{\theta}_t^i)$, here represented by the Dirac delta distribution, δ, defined over $s \in \mathcal{S}$, the space of all possible objects and viewing angles pairs.

Furthermore, the weights correspond to the ratio between the probability of $p(o, \bar{\theta}_{1:t}|Z_{1:t}, \bar{\Delta}_{1:t-1})$ evaluated at the particle center, and the density from which they were sampled, $q\left(s|Z_{1:t}, \bar{\Delta}_{1:t-1}\right)$:

$$w_t^i \propto \frac{p\left(s_t^i|Z_{1:t}, \bar{\Delta}_{1:t-1}\right)}{q\left(s_t^i|Z_{1:t}, \bar{\Delta}_{1:t-1}\right)}. \tag{4.3}$$

In a Markovian setting, we can update the hypothesis probability iteratively by taking into account the probability in the previous time step, a prediction of a new observation based on changes in the robot position and the new observation itself. A general formulation of a particle filter in object recognition would be:

Generate M random initial conditions :

Hypothesize M pairs of possible objects and initial orientations, $s_1^i = (o_i, \bar{\theta}_i)_1, i = 1, ..., M$;

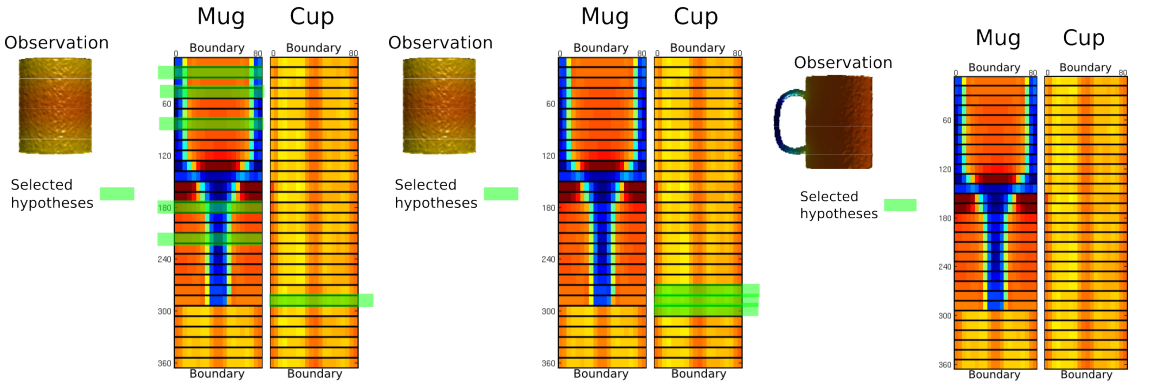
For each time step, j, until Convergence :

1. Estimate a new observation, $\bar{z}_j$;
2. Propagate particles, $s_j^i = s_{j-1}^i + (0, \bar{\Delta}_{j-1})$;
3. Update the probability for each hypothesis;
4. Bootstrap by replacing low by high probability hypothesis;
5. Estimate the object identity;
6. Check convergence.

The inclusion of the object class in the state vector differentiates our problem from more common uses of particle filters, such as, tracking and localization. In particular, the object class separates the search space so that not all the partial views are reachable by a given particle. For example, if a particle is associated with an object o' and viewing angle $\bar{\theta}'$, the above algorithm updates $\bar{\theta}'$ according to the robot movement, but o' will remain constant. As hypotheses can disappear in the bootstrapping step, if, at some iteration, there is no hypothesis associated with a given object, it disappears from the search space. When the removed object was the correct one, we cannot hope to classify correctly the partial view without restarting the estimation.

In the case of very similar objects, this may happen quite often. Consider the example in Figure 4.3, where the robot starts by observing the mug with a handle, but the handle is not in view, i.e., the observation could belong to both objects. The robot draws an initial set of hypotheses, marked with the green rectangles and compares them with the observation. As none of the hypotheses included the mug with the hidden handle, the only hypotheses with considerable weight are from the mug with no handle. In the bootstrapping stage, all hypotheses from the mug with a handle have a small weight and are moved to the mug with no handle. From this step forward, there is nothing in the Sequential Importance Resampling algorithm that would allow to

re-introduce the correct object into the search space and the robot would never be able to recognize the object.



(a) Collect observation and draw set of hypotheses

(b) Compare hypotheses and re-sample

(c) Collect new observation and discard all hypotheses.

Figure 4.3: Sequential Importance Resampling Filter for object estimation.

To ensure that the whole search space is reachable at each stage of the algorithm, we take advantage that our objects are actually similar to each other. We thus contribute a multiple view object identification algorithm that, while leveraging on a Sequential Importance Resampling framework, uses an off-line learned similarity between objects and viewing angles. The similarity is used to find high probability hypothesis during the bootstrap and is based on observations only, i.e., independent of objects and viewing angles.

Our proposed bootstrap method is illustrated in Figure 4.4 with an example with two very similar objects: a cup with no handle and a mug. In the first step, Figure 4.4(a), we map the current hypothesis into the observation space. In the second step, Figure 4.4(b), we search for similar observations. Finally, in Figure 4.4(c), we inverse the map to find all viewing angles that can be associated with those observations.

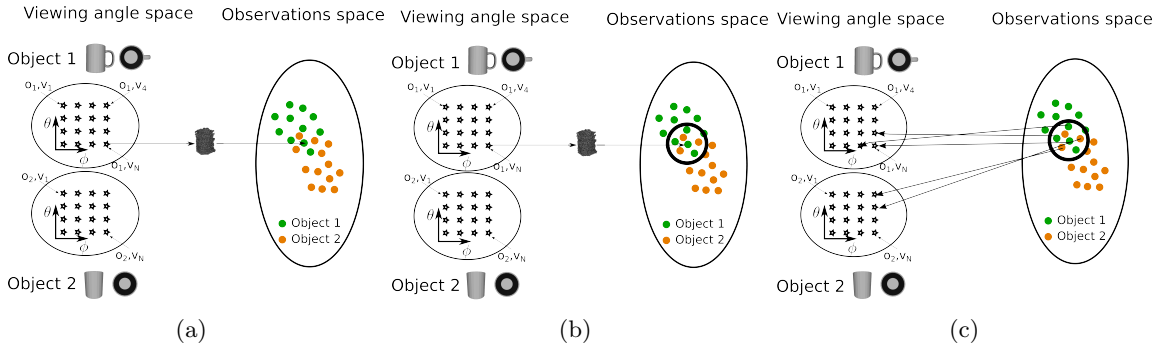


Figure 4.4: Example of the proposed bootstrap method.

4.3 Appearance Model

Each partial view is described using the PVHK, and the distances between partial views are estimated using the Modified Hausdorff distance, as described in Algorithm 3.1, defined in the previous chapter.

However, we here may have more than once observation from the same partial view related to $s = (o, \bar{\theta})$, and when available, we use sets, $Z^{s=(o,\bar{\theta})} = \{z_1, z_2, ...\}$, to represent partial views. To compare sets, we again use the Modified Hausdorff distance:

$$d(Z, Z') = \min \left\{ \sum_{\bar{x} \in Z} \inf_{\bar{y} \in Z'} d(\bar{x}, \bar{y}), \sum_{y \in Z'} \inf_{x \in Z} d(\bar{x}, \bar{y}) \right\}, \tag{4.4}$$

where Z and Z' can have different cardinalities.

We establish the probability $p(Z|s)$ that the set of observations Z corresponds to the partial view s by computing the distance between Z and Z^s. We define the probabilities based on distances using an exponential distribution $p(Z|s) = \exp\left(-d_{\mathrm{H}}(Z, Z^s)/\alpha^{o,\bar{v}}\right)/\alpha^s$. In this context α^s represents the average inner distance between a descriptor of a partial view associated with object o and viewing angle $\bar{\theta}$, and the set of descriptors associated with the same partial view:

$$\alpha^s = \sum_{z' \in Z^s} d_{\mathrm{H}}(\{z'\}, Z^s \backslash \{z'\})/|Z^s| \tag{4.5}$$

We define similarity, μ, between two partial views $s = (o, \bar{\theta})$ and $s' = (o', \bar{\theta}')$, based on the probability that we would identify a set of descriptors from the former as being from as the latter:

$$\mu(s, s') = p(s|s')) = p(Z^s|Z^{s'}) \tag{4.6}$$

4.4 Sequential Importance Resampling for Object Disambiguation

We motivate our Multiple Hypotheses for Multiple Views Object Disambiguation, presented in Algorithm 4.1, by first applying it to the problem initially presented in Figure 4.3. We then address each of the main stages of the filter.

In our example, we start with the robot facing the mug in the viewing angle where it looks like the cup and collects the first observation, represented in Figure 4.5(a) with a star. In the first step, the robot draws six random particles. Then given the first observation, we estimate the probability of each particle, which is represented by the weights w in Figure 4.5(a). While most particles are associated with the mug, they have a reduced probability and a small weight, w. But the particle associated with the mug with no handle explains the observation. So, we collect a new set in the vicinity of the high weight particle.

Figure 4.5(b) represents the new set of particles, and we note that all the new particles are now associated with a descriptor identical to high weight particle, albeit they are associated with both objects.

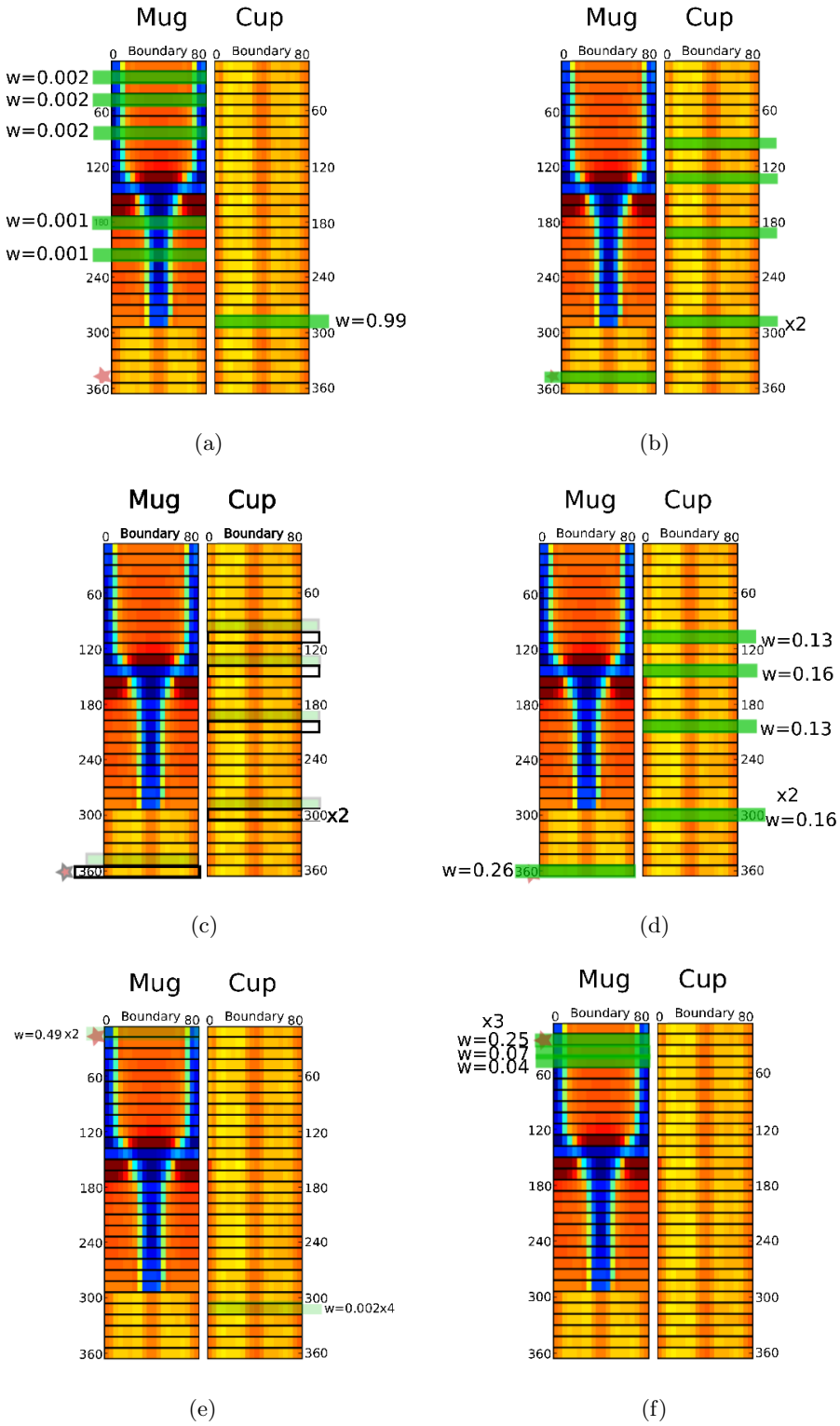
The robot then moves and propagates the particles accordingly, as illustrated in Figure 4.5(c), where we highlight the guesses for the new observation. The weights are then updated by comparing the guess with the observation retrieved, as illustrated in Figure 4.5(d).

In subsequent iterations, the particles coalesce around two main guesses, Figure 4.5(e), but when the handle becomes visible, only one partial view explains the observation and all the remaining partial views vanish, Figure 4.5(f).

The summary of the main steps sequence is provided in Algorithm 4.1. The algorithm receives as input the appearance models that return the probability of each partial view s, and the a-priori knowledge of the similarity between partial views. At each time step, the algorithm, also receives as an input a new observation set Z_t, and an odometry measurement. The output is an estimate of the object class at each time instant.

Algorithm 4.1: Computing Multiple Hypotheses for Multiple View Object Disambiguation.

Input: (i) Appearance models; $p(Z|s = (o, \bar{\theta}))$;
(ii) Similarity $\mu(s|s')$
Output: Object identity: $\hat{o}$
INITIALIZATION
$t \leftarrow 0$
$\mathcal{S}_0 \leftarrow sampleUniformlyAtRandom()$ (see Section 4.4.1)
$w_0 \leftarrow uniformWeights()$
$notConverged \leftarrow true$
while $notConverged$ **do**
  $t \leftarrow t + 1$
  $Z_t \leftarrow getNewObservation()$
  $\Delta_t \leftarrow getDisplacement()$
  for $i \leftarrow 0, i < N, i++$ **do**
    $\mathcal{S}_t \leftarrow propagateParticles(\mathcal{S}_{t-1}, \Delta_{t-1})$ (see Section 4.4.2)
    $\tilde{w}_t \leftarrow estimateAPriori(w_{t-1}, \mathcal{S}_t)$ (see Section 4.4.3)
    $restart \leftarrow checkRestart(\tilde{w}_t)$ (see Section 4.4.4)
    if $restart$ **then**
      $\mathcal{S}_t \leftarrow sampleUniformlyAtRandom()$
    else
      $w_t \leftarrow estimateAPosteriori(\tilde{w}_t)$ (see Section 4.4.5)
      $(notConverged, \hat{o}) \leftarrow checkConvergenceIdentify(\mathcal{S}_t)$ (see Section 4.4.7)
      $\mathcal{S}_t \leftarrow bootstrap(w_t, \mu)$ (see Section 4.4.6)
    end
  end
end



(a) (b) (c) (d) (e) (f)

Figure 4.5: Example the set of iterations of our Multiple Hypotheses for Multiple Views Object Disambiguation algorithm.

4.4.1 Initialize Particles

We start the particle filter by sampling uniformly at random N initial particles, $\mathcal{S}_0 = \{s_0^1, ..., s_0^N\}$, from the set of possible objects and view angles, $\mathcal{S}$. To each particle, we associate a weight $w_0^i = 1/N$ for all $i = 1, ..., N$.

4.4.2 Propagate Particles

At each time step t, we propagate the particles by changing the viewing angle according to the robot movement in the object coordinate system $\bar{\Delta}_{t-1}$.

We thus define the function $f : \mathcal{S} \times [0, 2\pi] \times [0, \pi] \rightarrow \mathcal{S}$ that updates each particle $s^i = (o^i, \bar{\theta}^i)$, associated with the object o^i and the viewing angle v^i, given a robot movement $\bar{\Delta}$:

$$f(s^i, \bar{\Delta}) = (o^i, \bar{\theta}^i + \bar{\Delta}) \tag{4.7}$$

4.4.3 Estimate the a-Priori

From a new set of observations, Z_t, we estimate the a-priori probability distribution by updating each weight as $\tilde{w}_t^i = w_{t-1}^i p\left(Z_t | s_t^i\right)$.

4.4.4 Restarting the Filter

When none of the particles explains the current set of observations, i.e., all weights $\tilde{w}$ are small, we draw a new set of particles and stop the robot movement. We restart the filter until a set of particles explains the current observation, i.e., when the sum of all the weights is higher than some threshold $Th_{restart}$.

4.4.5 Estimate the a-Posteriori

The a-posteriori is given by normalizing across all the a-priori weights, $\tilde{w}$.

$$w_t^i = \tilde{w}_t^i / \sum_{i=1}^{Np} \tilde{w}_t^i. \tag{4.8}$$

4.4.6 Bootstrap

During bootstrap, we eliminate low weight particles and replace them with particles in the neighborhood of those with high weight.

We say that a particle has a low weight by comparing it with the weight of the highest hypothesis, w_h^{max}. The weight of an hypothesis, $h^j = (o^j, \bar{v}^j)$, corresponds to summed weight of all the particles s^i equal to h^j.

Thus, given a threshold $\tau_{boot} \in [0, 1]$, we remove from $\mathcal{S}_t$ all the particles for which $w^i / w_h^{max} < \tau_{boot}$.

We then re-populate $\mathcal{S}_t$ with the partial views more similar to the set of the remaining particles, $\mathcal{S}_t^{remain}$.

We define the similarity $\mu(s, \mathcal{S}_t^{remain})$ between the partial view $s = (o\bar{\theta})$, and a set of particles, $\mathcal{S}_t$, as a weighted sum over the similarity between the partial views and each particle in $\mathcal{S}_t^{remain}$:

$$\mu(s, \mathcal{S}_t^{remain}) = \sum_{i=1}^{|\mathcal{S}_t^{remain}|} w^i \mu\left(s|s^i\right). \tag{4.9}$$

The new particles are then sampled using Stochastic Universal Sampling assuming a probability distribution proportional to the similarity. However, only viewing angles that have a similarity above some threshold σ_{min} are considered.

4.4.7 Test Convergence and Identify Object

The algorithm converges when all the particles agree on the object class. By imposing such a strong consensus, we prevent most false positives as, due to the bootstrap step, we ensure that as long as the observations are consistent with two objects, we have particles from the two objects.

4.5 Performance Evaluation

We evaluate the algorithm performance with respect to both its accuracy at identifying objects, its efficiency and its possible use in different problems.

As baseline for comparison, we use an alternative bootstrap step, where particles are included based on a similarity between viewing angles, not appearance. The re-populate step in Section 4.4.6, becomes just a random sampling over the neighborhood of the remaining particles. We also test for the impact of changes in parameters, e.g., the initial number of particles or the maximum number of particles we replace at each bootstrap step.

We introduce two datasets for testing of our algorithm. The first, similar to the mug example, we use to show that we can disambiguate between real shapes and that there is an improvement in terms of both computational effort and movement around the object. The second, composed of 8 chairs, that we use to show that the approach has more applications than the disambiguation between odd objects.

4.5.1 Datasets

We further test the performance of our algorithm in a similar setup but on a dataset collected with a Kinect sensor. Objects correspond now to human, spinning over himself with and without a bag-pack, as illustrated in Fig. 4.6. In each case we have a total of 24 different orientations, equally distributed around the z-axis. For each orientation, we collected two sets of 25 observations. One set was used for learning the appearance models and the similarity between view angles, the other was used for the algorithm evaluation. The human was segmented in the depth images by background subtraction. This dataset is used to identify whether the human is carrying the bag or not.

Finally, we show the potential for generalization of our algorithm with an example of same-class object identification. Our third dataset contains partial views of the eight chairs represented in

Figure 4.6: Dataset of partial views of a human in different orientations. The dataset corresponds to two generic shapes: Human with no bag, at the top row and Human with a bag, at the bottom row.

Figure 4.7 and retrieved from 3D Google warehouse. While they are similar to each other the chairs are not identical from any view angle. However, due to noise and sparse object libraries, it is not always possible to correctly identify an object. The partial views were obtained from a manner similar to that described for the mug and cup with no handle example. We collected three sets of partial views, one for the construction of the library, one for learning similarities and the third as the testing dataset. The testing dataset contains partial views gather from 127 different view angles per chair, while the object library has only 13 per chair. In this dataset, we used a fixed stopping time for all objects.

Figure 4.7: Dataset of similar chairs.

4.5.2 Accuracy

The accuracy accesses whether the algorithm reaches the correct identification at convergence t_{conv}. We consider two experiments to access the impact of the proposed bootstrap approach on accuracy. In the first we compare with the baseline method of bootstrapping, where only the similarity between viewing angles is accounted for. Second, we evaluate the accuracy as a function of the number of particles replaced at each iteration.

Both experiments run on the human dataset, starting in the same initial state, with the human carrying the bag facing the camera, i.e., in an ambiguous state. Furthermore, to account for the stochastic nature of the algorithm, we repeat each experiment 30 times, and the results we here present are the averages over the trials.

In the first experiment, we fix the convergence criteria and the conditions for restart and resample. The accuracy comparison between algorithms is presented in Figure 4.8(a). The results show that we have a significant increase in accuracy when using the similarity between observations as the criteria for sampling new particles. The impact is more noticeable when the number of particles is kept small.

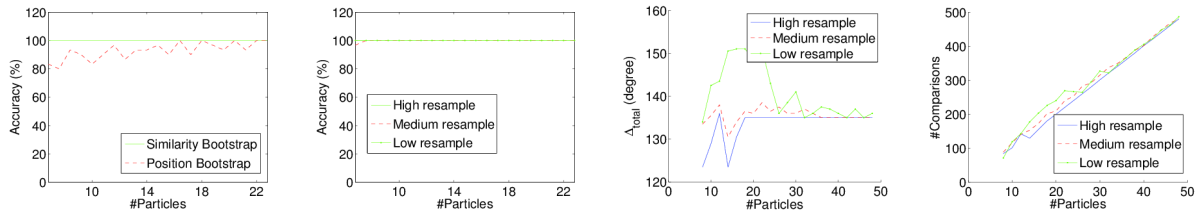
Furthermore, we note that reducing the number of particles replaced at each iteration has little to no effect in terms of recognition, as we show in Figure 4.8(b). The number of replaced particles is controlled by the threshold τ_{boot}, that defines the minimum ratio between a particle weight and the highest hypothesis weight so that the particle is not discarded. By increasing the necessary ratio, we are increasing the number of particles that are discarded and increasing the search of alternative partial views to explain a sequence of observations.

4.5.3 Efficiency

We associate efficiency to the effort required to correctly differentiate between objects. The effort can be either mechanical, evaluated in terms of the distance a robot would have to travel, and computational, evaluated in terms of the total number of comparisons between partial views. Again both were evaluated on the human dataset, using the same setup as the one used to access accuracy.

The distance the robot has to travel is associated with how much of the object surface it needs to cover before identifying it. Our results represented in Figure 4.8(c), show that the robot would have to cover on average 150^o of the human, i.e., it did not had to see the complete object.

The number of comparisons between partial views corresponds to the number of particles used in the experiment times the number of iterations used. Our results represented in Figure 4.8(d), show that for smaller sets of particles the robot would require fewer comparisons using our algorithm than applying exhaustive search. There are 48 known partial views in the dataset. Thus, exhaustive search requires 48 comparisons. As the objects are ambiguous, we need at least two observations, i.e., 96 comparisons, to identify the object. Our results show that we can use more observations and from more viewing angles, and still be competitive in computational terms.



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

Figure 4.8: Evaluating efficiency and accuracy.

4.5.4 Same-class Identification

Both by acquisition, storage and evaluation constraints, we cannot expect that each viewing angle grasped by a robot was previously seen in the object library. In this case, and especially when objects are from the same class, some partial views become misclassified, as we represent in the confusion matrix in Figure 4.9. The figure represents the confusion matrix between the testing dataset, composed of partial views collected from 127 different viewing angles per chair, $\Theta_{test} = \{[45^o, 0^o], [45^o, 2.8^o], ..., [45^o, 360]\}$, and the object libraries composed of partial views from 13 viewing angles per chair, $\Theta_{lib} = \{[45^o, 0^o], [45^o, 28.4^o], ..., [45^o, 360^o]\}$

Using Algorithm 4.1 with particles that could only populate the object library, i.e., that only covered 13 viewing angles of the set of chairs, we were able to recognize all the eight chairs in the viewing angles from the testing dataset. The results we present in Figure 4.10 correspond to the aggregated accuracy over all the chairs and for 10 different initial viewing angles. Given the initial viewing angle, the robot observed the whole object at intervals of 15^o degrees. At each position, the robot collected two observations and at the end of the path the robot identifies the chair. We thus cover all the possible viewing angles in the testing dataset, Θ_{test}.

The partial view observation models assumed an exponential distribution with $\alpha = 0.08$. The similarity μ was learned using an independent dataset.

The results show that, by collecting information from multiple partial views and using our similarity metric, we were able to identify the objects correctly in all the cases. We were also able to do so using a sampling even sparser than the 13 viewing angles per object in the object library, as we obtained a perfect accuracy with only 7 partial views per object.

4.6 Summary

In this chapter, we presented an algorithm for the disambiguation of similar objects by collecting and combining observations from a sequence of viewing angles. The algorithm leverages on a similarity metric between observations to off-line learn neighborhoods between viewing angles. The neighborhoods are used when bootstrapping hypothesis and ensuring that they reflect the objects ambiguity.

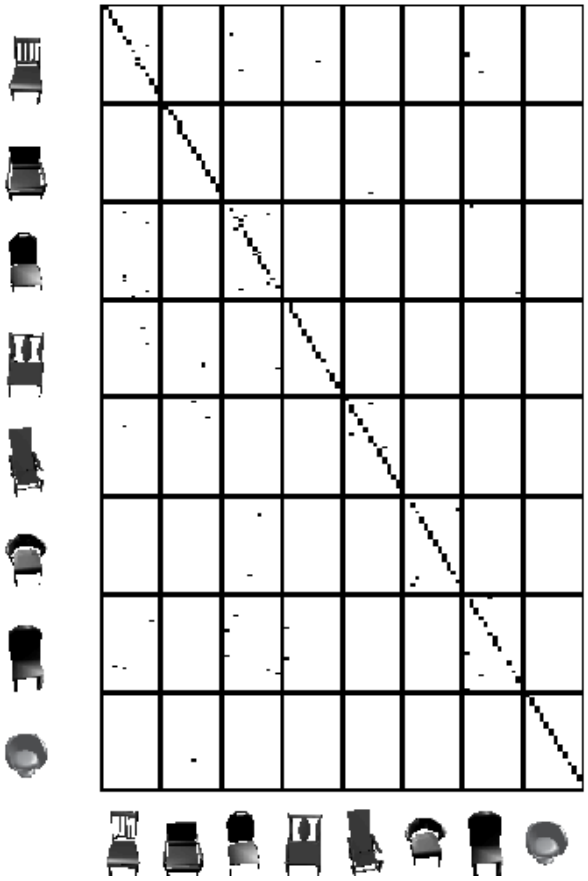


Figure 4.9: Confusion matrix between the testing dataset and the object library.

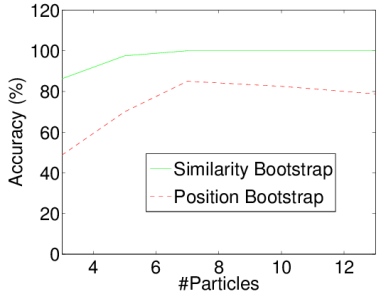


Figure 4.10: Aggregate accuracy as a function of the number of particles per object.

The proposed approach has two main advantages: i) reduces the number of false positives as ambiguous observations lead to an even distribution of particles among the objects; and ii) reduces the number of particles required for estimation, as the particles can cover a much more diverse set of partial views.

Chapter 5

Complex Objects and the Partial View Stochastic Time (PVST)

In this chapter, we address the problem of representing partial views of complex objects using the temperature at the boundary in a single time instant, [9]. In Section 5.1, we motivate the need to discriminate regular objects from complex objects, i.e., objects composed of loosely connected parts. In Section 5.2, we show as the loose connections alter the partial view heat kernel descriptor in complex objects, and in particular hinder its discriminative capabilities. In Section 5.4, we propose two new representations, also based on heat diffusion, but more suitable to handle loose parts. The algorithms and their properties are presented in Section 5.4. We empirically evaluate the performance of the new approaches on a dataset of complex objects in Section 5.5.

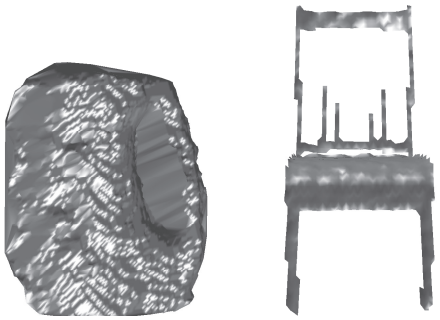
5.1 Regular Objects vs Complex Objects

In previous sections, we used PVHK descriptors to discriminate between several objects. However, most objects were tightly connected and compact, e.g., the kettle in Figure 5.1(a). We call them regular objects.

Additionally, there are less tightly connected objects, to which we call complex objects. For example, the chair in Figure 5.1(b) is a complex object as it has a main and tightly connected part, the seat, and smaller loosely connected parts, the back and the legs.

Complex objects require a re-thinking of the criteria we use to construct PVHK descriptors, particularly on how long do we allow the heat to diffuse before measuring the temperature at the boundary. To ensure descriptiveness, the temperature at the boundary should depend on the distance between boundary and heat source. So, the time instant at which we stop diffusion, t_s, must represent the size or scale of the object.

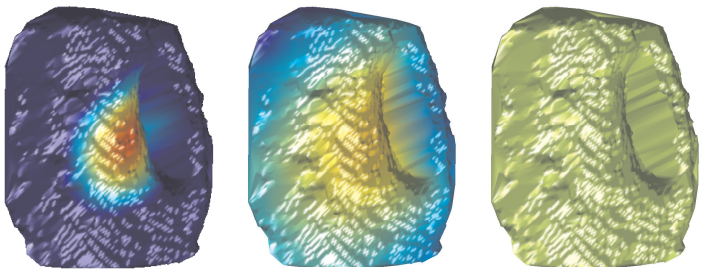
As temperature at the boundary is initially zero, t_s, must be large enough so that heat has time to diffuse from the heat source to the boundary. The resulting temperature should then be in higher temperatures closer to the source, where the heat reaches first. However, t_s should also



(a) Regular objects Complex object

Figure 5.1: Shapes of regulars and complex objects. The chair is complex, because its back is loosely connected to the seat.

be small enough to avoid the equilibrium state, which is characterized by a constant temperature, T_{eq}, over the whole surface. Figure 5.2 illustrates the possible temperature profiles over the surface of a regular object. The profile on the left corresponds to a small t_s, where the temperature at the boundary is very small. The profile on the right corresponds to the equilibrium state, and the profile in the middle corresponds to our desired situation: the temperature at the boundary changes based on the distance to the source.



Too small t_s Good t_s Too large t_s

Figure 5.2: Temperature profiles over a kettle at different time instants.

In previous chapters, we used a stopping time, t_s, associated with the time scale of heat diffusion over the whole surface, i.e., a global time scale of the partial view. This global time scale, $t_{global} = \lambda_2^{-1}$, corresponds to the time required for the temperature on all the points in the object to be above some temperature, T_{th}, defined as a fraction of the equilibrium temperature. Thus, evaluating the temperature at $t_s = t_{global}$ ensures that the temperature at the boundary is at least T_{th}.

The connected surfaces of regular objects reduce in-homogeneities in the surface temperature, which ensures that a large fraction of the object cannot reach the equilibrium temperature, T_{eq}, while some parts are still at a temperature lower than T_{th}.

Thus, linking the time instant t_s to t_{global} also ensures that temperatures at the boundary of regular objects are no longer zero, but different from T_{eq}. In this case, t_{global} depends on the length

of the path the heat has to travel to reach the boundary, and increases with the size of the object.

However, the loose connections of complex objects reduce the heat flux between parts, i.e., when heat diffuses from one part to the other, it is constrained to pass on a small bridge or bottleneck. Just like cars on the road, there is only so much heat that can pass at a given time instant on the bottleneck. Thus, at t_{global}, complex objects, such as the chair in Figure 5.3, will have most of its surface with a temperature close to T_{eq}, while the temperature is still very low within the smaller parts.

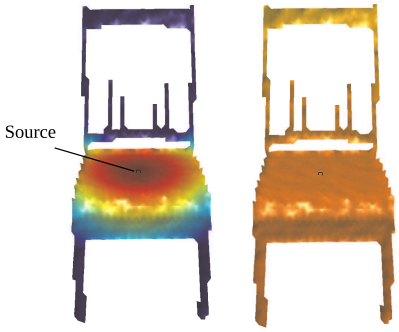


Figure 5.3: Temperature profiles over a chair at different time instants.

In summary, complex objects show:

1. an almost constant temperature at $t_s = t_{global}$,

2. strong in-homogeneities between different object parts.

As at $t = t_{global}$ most of the boundary is also at a temperature equal to T_{eq}, the PVHK is an almost constant descriptor. However, it will have sharp transitions near smaller, cooler parts, leading to large distances between similar shapes.

In this chapter, we address both problems. The first by decreasing t_s in complex objects. The second by introducing a part-aware metric that allows to filter sharp transitions in the descriptor without the need to remove small and loosely connected parts. We also propose an alternative descriptor to the PVHK, the Partial View Stochastic Time, that associates distances to the time it takes to reach a given temperature and thus is impervious to bottlenecks.

5.2 Time Scales in Complex Objects

We define diffusion time scales by establishing bounds on the temperature as a function of time. We then show how the bounds, while globally relevant, do not describe what happens locally, at each part. Finally, we show how loose connections lead to large global time scales and result in poorly informative descriptors.

5.2.1 Global Diffusion Time Scale

As we saw in Chapter 2, heat diffusion is described by the equation:

$$\partial_t \bar{T}(t) = -L\bar{T}(t) \tag{5.1}$$

where L is the Laplace-Beltrami operator. The equation has a closed form solution with respect to the eigenvalues λ_i and eigenvectors $\bar{\phi}_i$ of L expressed as:

$$\bar{T}(t) = \sum_{i=1}^{N_V} \bar{\phi}_i \exp\{-\lambda_i t\} \bar{\phi}_i^T \bar{T}(0). \tag{5.2}$$

If t_s is too large, the temperature at the boundary would be constant everywhere. As by construction L is semi-positive definite, with $\lambda_1 = 0$ and $\lambda_{i>1} > 0$, when $t \to +\infty$, $\exp\{-\lambda_i t\} \to \neq 0$ if and only if $i = 1$. As $\bar{\phi}_1 = \bar{1}/\sqrt{N}$, for large t_s, Eq. 5.2 simplifies to $\bar{T}(t) = \bar{1}$, independently of the source position and object shape.

However, for $t_s = 1/\lambda_2$ as used in [10, 12, 52], there is a lower bound to how different $\bar{T}(t)$ is from the equilibrium temperature:

$$\max_s \|\bar{T}^s(t) - \bar{1}\|_1 \geq \frac{N}{2} \exp\{-\lambda_2 t\}; \tag{5.3}$$

where $\bar{T}^s(t)$ is the temperature at t when we place the heat source at s. We present the proof of the lower bound in Appendix B. The bound ensures that at $t = 1/\lambda_2$, the maximum average distance do the equilibrium temperature is greater than $N/2 \exp\{-1\}$, and thus, the temperature is still not constant everywhere.

If the stopping time t_s is too small, the temperature at the boundary is zero everywhere, and $\bar{T}(t_s)$ does not contain enough information to describe objects. In fact, we can estimate the time required for all vertices to be at a temperature above some threshold T_{th} from the bound in Eq. 5.4, which proof we present in Appendix A.

$$\|\bar{T}(t) - \bar{1}\|_\infty \leq N \exp\{-\lambda_2 t\}, \tag{5.4}$$

The above bound ensures that temperature at all points in the object surface, including those at the boundary and regardless of source position, are above a temperature T_{th} for all $t > \frac{1}{\lambda_2} \log(N/(1 - T_{th}))$.

Both bounds are governed by λ_2, and thus we used $t_0 = t_{global} = 1/\lambda_2$ to compute the descriptor on regular objects. However, in complex object, where parts are small and loosely connected, the two bounds do not reflect what happens in the main part of the object. In particular, bottlenecks introduced by the loose connections decrease λ_2 considerably. In short, λ_2 no longer represents the time scale of diffusion over most of the object, but with the difficulty of heat passing through the

bottleneck.

5.2.2 Impact of Bottlenecks on λ_2

A bottleneck separates the surface in two complementary parts, S_1 and S_2, with $^{\#}S_1$ and $^{\#}S_2$ vertices, connected by a boundary ∂S_1, as illustrated in Figure 5.4.

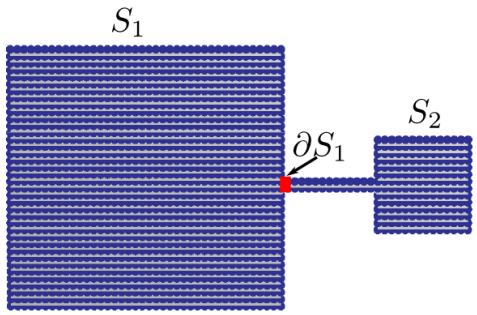


Figure 5.4: Example of a complex object, composed by two squares connected by a bottleneck. At the region of the bottleneck, we separate the surface in two fractions, S_1 and S_2, by means of a boundary ∂S_1.

We here show that the sum of all the weights of ∂S_1, $\mathcal{W}_{S_1} = \sum_{(i,j)\in\partial S_1} w_{i,j}$ imposes an upper bound on λ_2. To show this relation, we write $\mathcal{W}_{S_1}$ as a function of the Laplace-Beltrami L and an indicator vector, $\bar{f}^{S_1} \in \mathbb{R}^N$ defined as:

$$[\bar{f}^{S_1}]_i = \begin{cases} 1/\ ^{\#}S_1 & ,\text{iff} \quad v_i \in S_1 \\ -1/\ ^{\#}S_2 & ,\text{iff} \quad v_i \in S_2 \end{cases} \tag{5.5}$$

As $\bar{f}^{S_1}$ is constant everywhere, and only changes between neighboring vertices in ∂S_1, we can can write $\mathcal{W}_{S_i}$ by recalling that the graph Laplace-Beltrami approximates a second order derivative:

$$[L\bar{f}]_i = \sum_{j\in\mathcal{N}_i} w_{i,j}([\bar{f}^{S_1}]_i - [\bar{f}^{S_1}]_j) \tag{5.6}$$

$$= \left(\frac{1}{^{\#}S_1} + \frac{1}{^{\#}S_2}\right) \begin{cases} \sum_{j\in\mathcal{N}_i} w_{i,j} & \text{iff} \quad i:(i,j)\in\partial S_1 \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

where $\mathcal{N}_i$ is the set of all vertices connected to i by an edge.

Thus, $\mathcal{W}_{S_1}$ is proportional to $\frac{\bar{f}^{T,S_1} L \bar{f}^{S_1}}{\|\bar{f}^{S_1}\|^2} = 2(1/\ ^{\#}S_1 + 1/\ ^{\#}S_2) \sum_{(i,j)\in\partial S_1} w_{i,j}$, which leads to the

bound in Eq. 5.8

$$\lambda_2 \equiv \min_{\bar{f}\in\mathbb{R}^N} \frac{\bar{f}^T L \bar{f}}{\|\bar{f}\|^2} \tag{5.8}$$

$$\leq \min_{S^1} 2\left(\frac{1}{\#S_1} + \frac{1}{\#S_2}\right)\mathcal{W}_{S_1} \tag{5.9}$$

where the latter inequality reflects that all indicator vectors $\bar{f}^{S_1}$ are a subset of all possible $\bar{f} \in \mathbb{R}^N : \quad \bar{f}^T\bar{1} = 0$.

So, when we connected a partial view, S^1, with time scale $1/\lambda_2^1$, to a second surface, S^2, by means of a bottleneck with a small total weight sum, $\mathcal{W}$, we can expect the joint time scale $1/\lambda_2^{total} > 1/\lambda_2^1$ to increase in proportion to $1/\mathcal{W}$.

Numerical Example

Using the example from Figure 5.4, where we represent both the complete and the isolated subgraph S_1. We then estimate the minimum bound from Eq.5.8: $(1/N_1+1/N_2)\mathcal{W}$ for both shapes. Assuming that the square S_1, has a side L, and number of vertices N_l, the partition that minimizes $\mathcal{W}$ cuts S_1 surfaces in two equal rectangles, $S_1 :^{\#} V_1 = 1/2N_l^2$. In this case, the weight of all edges in the cut is given by $\sum_k w_k(e_k) = N_l/l^2 + (N_l-1)/(2l)$, where $l = L/(N_l-1)$ and $\sqrt{2l}$ are the edges lengths. Thus the upper bound to λ_2 is given by $\lambda^{up} = 1/(l^2N_l) + 1/(lN_l) - 1/(lN_l^2)$.

For the second object, the cut passes over the bottleneck and $S_1 :^{\#} V_1 = N_s^2 + N_b$, where N_s is the number of vertices on each side of the small square and N_b is the number of vertices on the bottleneck. The weights associated with this cut correspond to $3/l^2 + 1/l$. Thus the upper bound is given by $\left(1/l^2 + 1/(4l)\right)/(N_s^2 + N_b) \leq \lambda^{up}$.

Computing for our example, we obtain $\lambda_2^1 = 9\times10^{-3}$ for the square S_1, and $\lambda_2^2 = 8.4\times10^{-4}$ for the total shape. On the other hand, we have $\lambda^{up} = 3.90\times10^{-2}$. So, both λ^{up} and λ_2 have decreased by a factor of 10 with the introduction of the bottleneck.

5.2.3 Local Time Scales in Complex Objects

The main consequence of a bottleneck is that it introduces several time scales, destroying the geometric correlation between object size and global time scale. However, in each part, heat propagates as if the bottleneck was not present, and temperature reaches an equilibrium at their original time scale.

Thus, while a regular object has a single time scale, when we attach to it a second object by means of a bottleneck, we end up with three time scales:

1. the time scale associated with the size of the first object, $1/\lambda_2^1$;
2. the time scale associated with the size of the second object, $1/\lambda_2^2$;
3. the time scale associated with the time required by heat to cross the bottleneck $t_{bottleneck}$.

The global time scale is necessarily larger than any of the three.

In the example of Figure 5.5, we simulate the heat diffusion process over the surface of four similar objects, which differ on a bottleneck. The first object is the square S_1 of Figure 5.4, where we place a source at the center. The following three objects corresponds to the joint S_1 and S_2 surface, but with considerable changes to the bottleneck connecting the two squares.

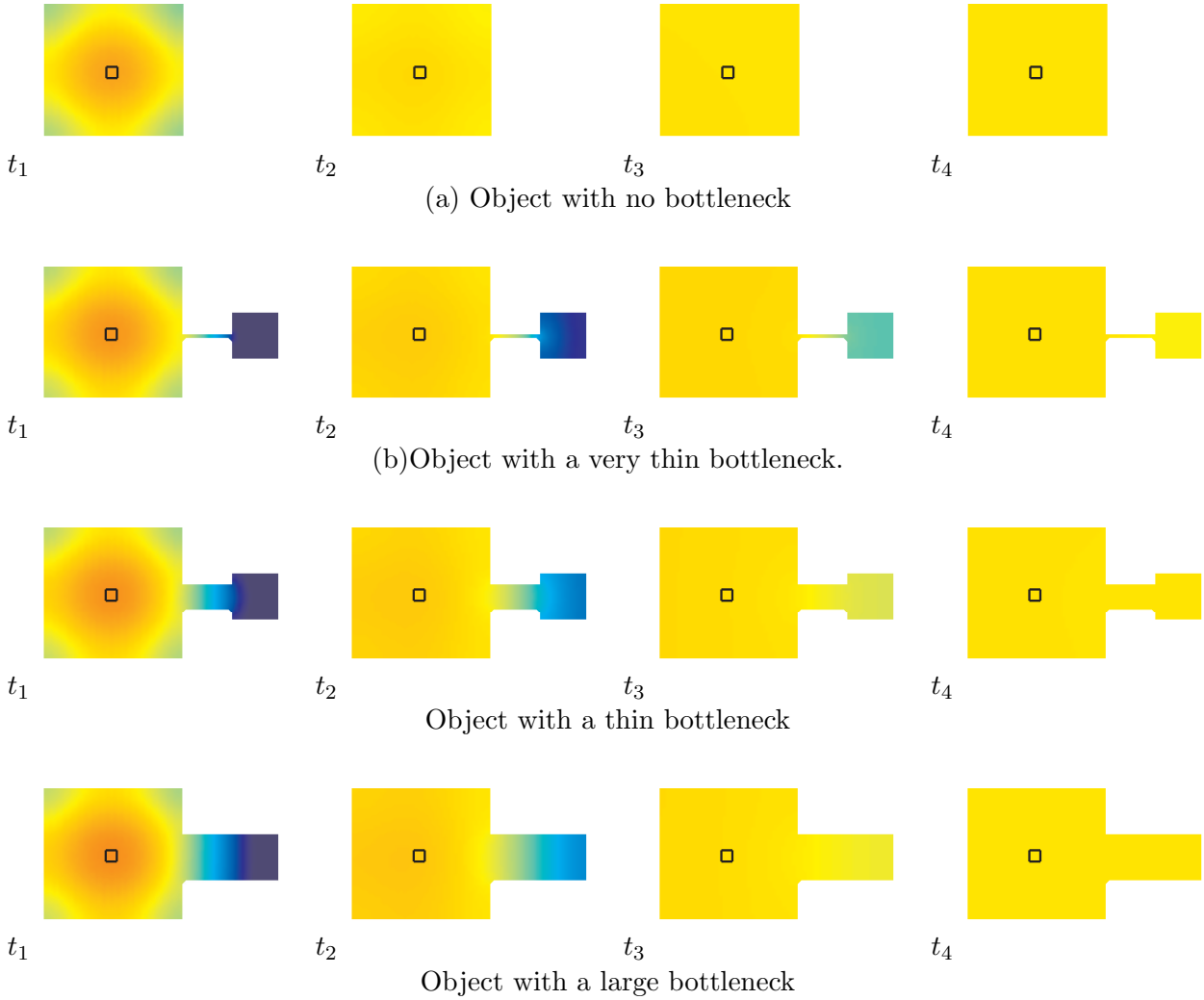


Figure 5.5: Impact of bottlenecks on the global time scale of heat propagation over an object.

The example highlights the impact of bottlenecks on time scales. For the first object, with neither bottleneck nor small part, the time required to ensure that the temperature at all points is above some threshold T_{th}, depends only on the distance between the point to the source. In this case, the global time scale $t^1_{global} = 1/\lambda^1_2$ is associated with the size and diameter of the mesh graph, [38]. Thus, on the second time instant, the square a has a temperature almost constant over

the whole surface. The same happens with the other three objects, i.e., the temperature at a are little affected by the second object. However, in all the remaining three objects, the temperature on the smaller square depends on the bottleneck thickness. In particular, the time it takes for all the parts in the object to be above some threshold temperature T_{th} decreases with the thickness, decreasing the global time scale and increasing λ_2.

5.3 Parts in Complex objects

The presence of bottlenecks also introduces large in-homogeneities in the temperature over the partial view and in the descriptor. In particular, loosely connected parts, show a large contrast in the temperature when compared with larger object parts.

In complex objects, as the global time scale is larger than the time scale of each part, when we use $t_s = 1/\lambda_2^{global}$, the temperature over the largest part is almost constant over time. Thus, regardless of where we place the heat source within that part, there are little changes to its temperature. However, the temperature in the smaller parts has not yet reached equilibrium and thus presents larger changes in the temperature.

In Figure 5.6 we show changes in the temperature over the whole shape at $t_s = 1/\lambda_2^{global}$, when we move the source position in the largest part.

Figure 5.6: Impact of changes in the heat source position in objects with a very thin bottleneck.

We can thus use the impact in the temperature when we change the source position, for soft-identification of small and loosely connected parts in the objects.

5.3.1 Soft Identification of Small and Loosely Connected Parts

As at $t = 1/\lambda_2$ temperature over complex objects' largest part does not change with variations to source position, we identify small parts as those where the temperature does change. To quantify the susceptibility of vertex i to changes in its temperature caused by variations in the source position, we introduce the source position global derivative: $\Delta_s \bar{T}^s(t)$. The global derivative measures how much the temperature changes when the source moves from one vertex to another in the same edge, and accounts for all vertices in the mesh:

$$\Delta_s[\bar{T}^s(t)]_i = \sum_{l=1}^{N} \sum_{j \in \mathcal{N}(l)} ([\bar{T}^j]_i - [\bar{T}^l]_i)^2 w_{l,j}. \tag{5.10}$$

We write $\Delta_s \bar{T}^s(t)$ as a function of the eigenvalues and eigenvectors of L, recalling the relation

between the Laplace-Beltrami operator and the second order derivative. Namely, noting that the temperature at vertex i when the source is placed at j is the same as the temperature at vertex j when the heat source is at i, we have:

$$\Delta_s[\bar{T}^s(t)]_i = \bar{T}^i(t)^T L \bar{T}^i(t)/2 \tag{5.11}$$

$$= \sum_{k=2}^{N} \lambda_k [\bar{\phi}_k]_i^2 \exp\{-2\lambda_k t\}/2 \tag{5.12}$$

We note that the solution is similar to the time derivative of the heat kernel, and thus we expect a similar behavior. Namely, it will be low when the temperature is reaching equilibrium and high when it is still changing. Thus, in complex objects at $t = 1/\lambda_2$, we expect to find larger values of $\Delta_s \bar{T}^s(t)$ only at small parts. By comparing changes in the global derivative in object surfaces, we have a powerful tool for soft identification of object parts.

Figure 5.7 shows the source position global derivative in three chairs. The examples highlight that small parts in complex objects have higher global derivatives.

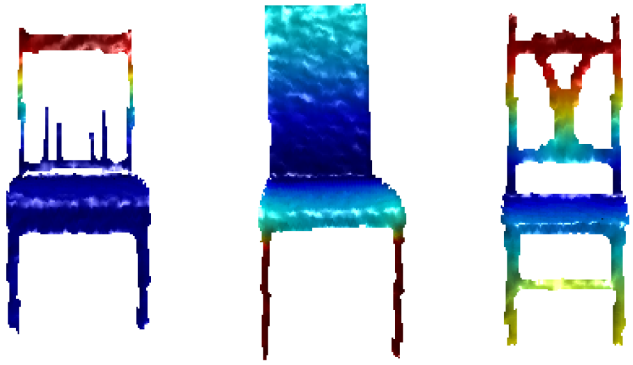


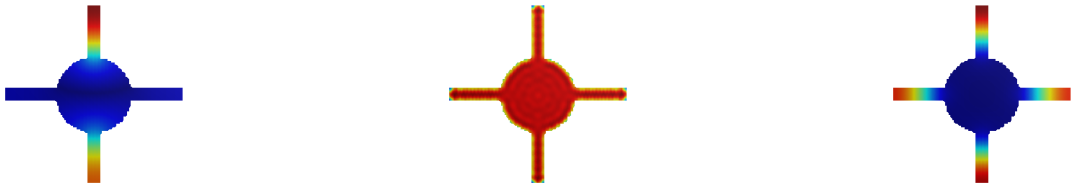
Figure 5.7: Source position global derivative in three different chairs.

5.3.2 Comparison with other Part Identification Approaches

The Laplace-Beltrami operator has been often used in parts identification. Namely, its second eigenvector has been used in spectral clustering[66] and persistence based clustering [51, 59]. Spectral clustering cuts a graph in two parts based on the sign of the second eigenvector in different surface parts. The persistence based approach, uses the eigenvector minima and saddle points to separate parts within a surface. Persistence based clustering was applied using the heat kernel signature [59] at a fixed time instant $t = 0.1$.

We provide an example of a symmetric object in Figure 5.8, where we show how the second eigenvector only identifies two of the four object parts. The heat kernel signature, also performs very poorly as it is constant over most object. On the other hand the source position global

derivative,$\Delta_s \bar{T}^s(t)$, is sensitive to the four object parts and assumes higher values within each parts.



Second eigenvector | Heat Kernel Signature | Source position global derivative

Figure 5.8: Comparison between part identification approaches and the source position global derivative.

5.4 PVHK for Complex Objects

To represent partial views of complex objects, we introduce two new approaches that avoid the use of a global time scale:

1. we fix a-priori the stopping time at which we estimate the temperature;

2. we use time as a surrogate for distance.

The first approach leads to the Fixed Time PVHK (FT-PVHK). FT-PVHK is equivalent to PVHK in all the aspects but requires a-priori estimation of a convenient stopping time, lower than the global time scale $t_{global} = 1/\lambda_2$. Using a stopping time lower than t_{global} no longer ensures that temperatures are significantly higher than zero throughout the whole boundary. Small and loosely connected parts of the partial view that are part of the boundary will show discontinuities in the descriptor, which will exacerbate distances between objects. We thus introduce a new part-aware metric to handle seamlessly the discontinuities when comparing partial views.

The second approach leads to the Partial View Stochastic Time (PVST). This approach represents surfaces by the time it takes for the temperature in the boundary to reach a given value. It does not require the estimation of a single time scale and thus is suitable for both regular and complex objects.

5.4.1 Fixed Time PVHK

In our first approach, we impose a fixed stopping time, t_s, for a given object. I.e., we offline determine the time instant $t_s < t_{global}$, which provides descriptors representative of the distance between most of the boundary and the source.

However, at this lower time scale, the descriptor will show strong discontinuities near loosely connected parts, e.g., a leg of a chair. By themselves discontinuities are desirable as they expose important object features. Withal, small parts are also highly susceptible to poor segmentation due

to sensor noise and, e.g., the chair leg may not appear connected to the main part of the object. The leg may also disappear due to occlusion from other parts of the object. Both descriptors of the chair, with and without the leg, will be very similar except for a strong discontinuity in the leg region, which unreasonably increases the distance between descriptors.

When comparing descriptors, we must consider that some parts of the object are not so relevant. To take the parts into account, we propose changes to our comparison metric, by introducing the probability of a given point in the boundary to be inside a small or large part of the object. The probability is estimated by soft classification of each point in the boundary.

In the following, we focus on the estimation of object time scales, then we introduce the probability of a vertex to be inside a small part of an object, and finally we introduce a new distance metric to compare partial views using the fixed time PVHK.

Complex Object Time Scales

We first highlight that we do not need the exact time scale t_{local}^{main} of the main part. As long as $t_s \sim t_{local}^{L}$, temperatures at the boundary will represent the distance to the source. So, provided it is consistently used to compute all the descriptors for a given object, we have some flexibility in estimating a good t_s.

There are several approaches that we can use to estimate a good value for t_s, such as:(i) offline test of different time scales in a validation dataset; (ii) offline segment the object, and estimate the eigenvalues of the largest part.

Both approaches would provide the required timescale, but would be time consuming. We here propose a natural segmentation of the object by considering the object as seen from multiple view angles.

By self-occlusion, we expect that in at least from a single viewing angle only the main part of the object is visible. We then choose the t_s as the lowest global time scale associated with all the partial views, as any local time scale will always be smaller that the global time scale, as we saw in the previous section.

The main steps for estimating the global time scale of the main parts of complex objects are presented in Algorithm 5.1. The algorithm receives as input a set of $N_{\bar{\theta}}$ meshes and the vertices coordinates from the same object seen from N_θ different viewing angles: $\{M^{s_1} = (V^{s_1}, E^{s_1}, F^{s_1}), M^{s_2}, ..., M^{s_{N_\theta}})\}$ and the respective coordinates $\{X^{s_1}, ..., X^{s_{N_\theta}}\}$. For each partial view, the global time scale is computed, and the algorithm returns the smallest of all time scales as an estimative of t_{local}^{main}.

Algorithm 5.1: Estimating stopping time in complex objects.

Input: Meshes from the same object: $\{M^{s_1} = (V^{s_1}, E^{s_1}, F^{s_1}), M^{s_2}, ..., M^{s_{N_\theta}}\}$

Vertex coordinates $\{X^{s_1}, ..., X^{s_{N_\theta}}\}$

Output: Main part time scale, t_{local}^{Main}

$t_{main} \leftarrow +\infty$

ESTIMATE TIME SCALES OF EACH MESH:

for $j = 1$**to**N_θ **do**

$L \leftarrow computeLaplaceBeltrami(M, X)$ $\lambda_2 \leftarrow computeEigenvalues(V, E)$

$t_{local}^{Main} \leftarrow \min(t_{local}^{Main}, 1/\lambda_2^{s_j})$

end

Comparing Objects with Parts

So far we compare partial views using a modified Hausdorff distance between descriptors. Here, we introduce a weighted modified Hausdorff distance that overlooks small and loosely connected parts.

Weights used for computing this new distance should be low when the boundary is on a small part, and close to 1 on the object main part. Different approaches could be used to map the global temperature derivative in an interval close to $[0, 1]$. We here propose to map the global derivative into this interval by introducing the probability that each boundary vertex is inside or outside a small part. We model the the probability distribution as a Gaussian on the global derivative of temperature with respect to source position.

Thus, to a descriptor $\bar{z}$ computed from the temperature profile $k(v_s, v_b, t_s) : v_b \in B$ defined over the boundary B, we associate a weight vector $\bar{\rho}$, computed as:

$$\bar{\rho} : [\bar{\rho}]_i = \exp\{-(\Delta_{v_s} \bar{T}_B^{v_s}(t_{global}))^2 \alpha\}. \tag{5.13}$$

where α is a normalization constant. As ρ should be in the range of [0,1][1], we fix α as the average of all the possible values of $\Delta_s T$, i.e., $\alpha = 1/\text{mean}\left(\left(\Delta_s \bar{T}_B^{v_s}\right)^2\right)$.

As we introduced in Chapter 3, Eq. 3.1, we compute distances between two observations $\bar{z}$ and $\bar{z}'$ defined over the boundary as: $d(\bar{z}, \bar{z}') = d_{\text{MH}}(\eta, \eta')$, where $\eta = \{[1/L, [\bar{z}_1]_i], [2/L, [\bar{z}]_2], ..., [1, [\bar{z}]_L]\}$ associates a temperature to a position in the boundary.

Thus, the distance between two partial views based on the fixed time PVHK descriptor, be-

[1]We note that we are using the term probability distribution as an analogy, as in truth we normalize $[\bar{\rho}]_i$ so that it has values close to 1, not to ensure that its integral with respect to the global derivative is 1.

comes:

$$d^M(\bar{z},\bar{z}') = d_{\mathrm{H}}^W(\eta,\eta',\bar{w},\bar{w}') \\ = \min\left\{\sum_{i=1}^{N_d}\rho_i \inf_{j=1,\dots,N_d}\|\bar{\eta}_i-\bar{\eta}'_j\|^2, \sum_{j=1}^{N_d}\rho'_j \inf_{i=1,\dots,N_d}\|\bar{\eta}_i-\bar{\eta}'_j\|^2\right\}; \tag{5.14}$$

where N_d is the number of vertices in the boundary and we recall that $\eta : \eta_i = ((i-1)/N_d, [\bar{z}]_i)$ is the curve version of the descriptor.

Algorithm 5.2 summarizes the steps essential for the comparison of partial view meshes using the FT-PVHK.

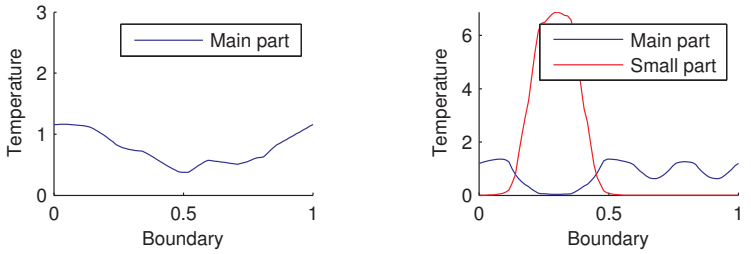
Algorithm 5.2: Comparing FT-PVHK descriptors.

Input: Object Mesh: $M^1 = (V^1, E^1, F^1)$, $M^2 = (V^2, E^2, F^2)$
Vertex coordinates: X_1, X_2
Boundary vertices: $B^1 \in V^1, \quad B^2 \in V^2$
Stopping times: t^1_{main}, t^2_{main}
Expected value of $\Delta_{v_s}\bar{T}^{v_s}(t_{global})$: $\alpha^1, \quad \alpha^2$
Output: Distance between partial views $d(M^1, M^2)$
COMPUTE DESCRIPTORS
$\eta^{1,2} \leftarrow$ computeFixedTimePVHK$(M^{1,2}, \bar{x}^{1,2}, B^{1,2}, t^{1,2}_{main})$
COMPUTE DERIVATIVES USING EQ. 5.11
$\Delta^{1,2}_{v_s}\bar{T}^{v_s}_B(t^{1,2}_{global}) \leftarrow$ softIdentificationOfPart$(M^{1,2}, \bar{x}^{1,2}, B^{1,2})$
COMPUTE WEIGHTS USING EQ. 5.13
$\bar{\rho}^{1,2} \leftarrow \exp\{-(\Delta^{1,2}_{v_s}\bar{T}^{v_s}_B(t^{1,2}_{global}))^2\alpha^{1,2}$
COMPUTE DISTANCE USING EQ. 5.14
$d^k(M_1, M_2) \leftarrow \sum_{i=1}^{N_d} w_i^k \inf_{j=1,\dots,N_d}\|\bar{\eta}_i^k - \bar{\eta}_j^{\{1,2\}\setminus k}\|^2$; $d(M_1, M_2) \leftarrow \min\left\{d^1, d^2\right\}$

Figure 5.9 shows the relation between the descriptors and the weights. In particular, Figure 5.9(a) shows evidence of the discontinuities in the chair descriptors. For the chair, we represent the descriptor both when the source is on the main or small part of the object. We notice that both present strong discontinuities, especially when the source is placed on the small part. On the other hand, Figure 5.9(b) shows how the weights in these discontinuity regions are much smaller than the weights of the object main part. Thus when comparing two chair descriptors, we expect that the discontinuity will have little to no impact in the distance we compute.

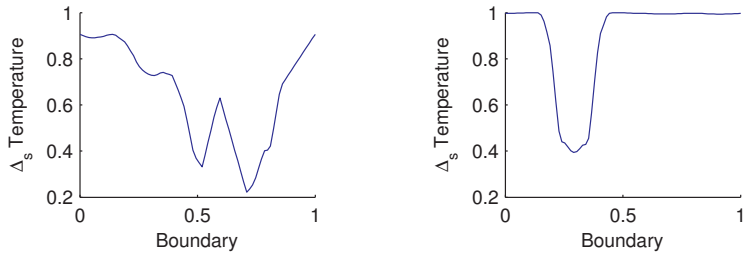
5.4.2 Partial View Stochastic Time

Instead of using the temperature at the boundary at a given time, we use the time it takes for the boundary to reach a given temperature. This is possible, as all the points on the surface have temperatures that, albeit not constrained to, have to pass through the same set of temperatures.



Kettle Chair

(a) Fixed time Partial View Heat Kernel descriptors.



Kettle weights. Chair weights.

(b)Weights used while computing distances between descriptors.

Figure 5.9: Descriptors and weights for three objects: the kettle and the chair.

This fact is captured in the following Lemma:

Lemma 1. Let T_a be a temperature in the interval $[0, 1[$. Then, for each vertex v_i in the partial view boundary B there is a time instant $t(v_i)$ such that $k(v_i, v_s, t(v_i)) = T_a$.

Proof. Proof of Lemma 1: We can show the Lemma by the Intermediate Value Theorem, as:

1. $k(v_i, v_s, t)$ is a continuous function over time for all points in the object surface;
2. since the source is not placed at the boundary, $k(v_i, v_s, 0) = 0$ for all $v_i \in B$.
3. as $k(v_i, v_s, t) \xrightarrow[t \to +\infty]{} 1$

□

Furthermore, we can also relate the resulting time $t(v_i)$ with the distance to the source v_s: points that are closer to the source increase temperature earlier, as seen in Figure 5.10. This figure shows the time it takes for each point in the surface to reach a temperature of $T_a = 0.75$. The blue regions, closer to the source, correspond to smaller time instants $t(v_i)$, while red regions, further away from the source correspond to larger $t(v_i)$.

Computing PVST Descriptors

Algorithm 5.3 describes how to compute the partial view stochastic time (PVST). As with the PVHK, we first need to extract the object mesh, $M = (V, E, F)$, determine the source position,

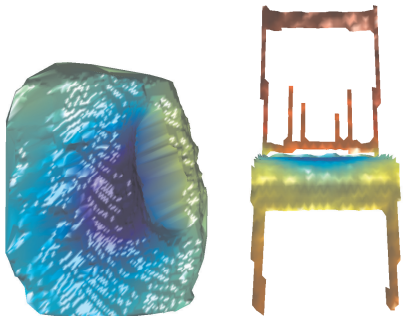


Figure 5.10: Time required for each vertex to reach a temperature of T=0.75.

$v_s \in V$, and determine the boundary vertices, $B \in V$. Then, we choose the time interval $[t_{init}, t_{final}]$ where we search for the correct time instant for each vertex.

Note that the temperature at time instant t is computed as $\bar{T}(t) = \sum_{i=1}^{N_e} \bar{\phi}_i \exp^{-\lambda_i t} \bar{\phi}_i^T \bar{T}(0)$, thus to correctly describe the first time instances we require high order eigenvectors, i.e., we need a large N_e. For example, if the initial temperature is zero everywhere except at a source vertex, v_s, to reconstruct the distribution, we need $N_e = N_V$, where V_v is the number of vertices in mesh M. This may prove to be impractical when we have a large number of vertices in the mesh. We thus fix the number of eigenvectors, and then set the initial time based on the highest order eigenvalue. Still, for objects with 30k vertices, we must compute around 1200 eigenvectors to ensure that the temperature at $t = 1/\lambda_{12}$ is realistic.

Then, for each boundary vertex we compute the time it takes to reach a temperature, T_a. As the temperature at the boundary is not necessarily monotonous, we use an exhaustive search algorithm to find the first time instant for which the temperature reaches a given threshold. The search is on a logarithm scale, i.e., we fix a $\delta\tilde{t}$ and then evaluate the temperature at instances $t_i = \exp\{i\delta\tilde{t}\}$. The interval $\delta\tilde{t}$ is defined as $\delta\tilde{t} \leftarrow (\log(t_{final}) - \log(t_{init}))/N_t$, were we choose $t_{final} = 10/\lambda_2$ as λ_2

is associated with the time required by heat to propagate to the whole object.

Algorithm 5.3: Computing Partial View Stochastic Time.

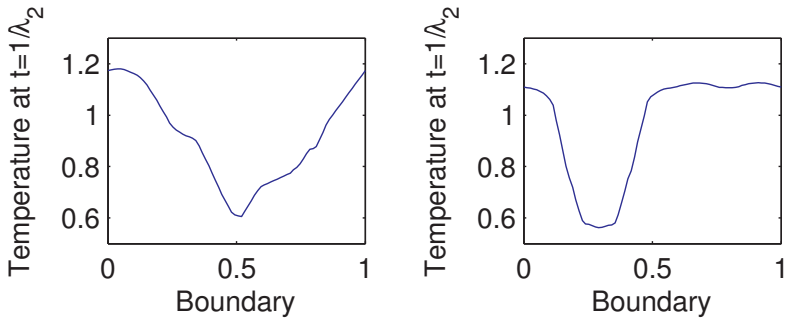
Input: Object Mesh: $M = (V, E, F)$
Vertex coordinates: $\bar{x}_{v_i}$
Boundary vertices: $B \in V$
Source Position: v_s
Number Time Instants: N_t
End temperature: T_{end}
Output: Partial View Stochastic Time, PVST: $\bar{z}_t$
INITIALIZATION:
$\bar{z}_t \leftarrow \bar{0}$
$(\Phi, \Lambda) \leftarrow$ eigValuesVectors(M,$\bar{x}$)
$\Phi_B \leftarrow$ getSubset(Φ, B)
$\Phi_s \leftarrow$ getSubset(Φ, v_s)
$t_{init} \leftarrow 1/\lambda_{12}$
$t_{final} \leftarrow 10/lambda_2$
$\delta\tilde{t} \leftarrow (\log(t_{final}) - \log(t_{init}))/N_t$
COMPUTE TEMPERATURE AT EACH TIME INSTANT:
for $j \leftarrow 1$**to**N_B **do**
  $i \leftarrow 1$
  while $[\bar{T}_B]_j < 0.75$ **do**
    $t \leftarrow \exp\{i\delta\tilde{t}\}$
    $[\bar{T}_B]_j = \Phi_{B_j} \exp\{-\Lambda t\}\Phi_s$
    $i \leftarrow i + 1$
  end
  $[\bar{z}_t]_j = t$
end

In Figure 5.11, we compare the PVHK with the PVST for the same objects. While both descriptors have shape signatures in the same regions of the boundary, there are two main differences.

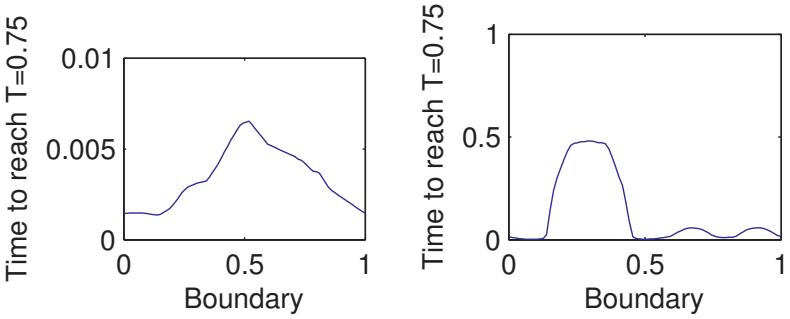
1. Where PVHK decreases, PVST increases and vice-versa: the time it takes for a vertex to reach a given temperature increases with its distance to the source.

2. Shape features leading to small changes in the PVHK are more noticeable in PVST.

The stochastic time partial view is an alternative to the partial view heat kernel: PVST also represents the distance between a point in the source and the boundary points, using a surrogate to shortest distances based on diffusive processes and that is less subjective to noise. However, by requiring the computation of a larger number of eigenvalues, and by requiring the exhaustive



Descriptors using the partial view heat kernel.



Kettle Chair

Descriptors using the partial view stochastic time.

Figure 5.11: Comparison between PVHK and PVST for the three objects.

search for each boundary vertex for a specific temperature, this descriptor takes a considerably longer time to compute.

In the following, we compare in terms of precision the three variants on the partial view heat kernel, namely:

1. the original partial view heat kernel, computed at $t_o = t_{global} = 1/\lambda_2$;
2. the fixed time partial view heat kernel, computes at $t_o \leq t_{global}$;
3. the partial view stochastic time.

5.5 Precision on Complex Objects

To compare the three approaches we introduce a large set of complex objects: the set of chairs represented in Figure 5.12.

We generated a test dataset with 120 partial views per object, by rendering them from different view angles, but at the same distance and height. The rendering followed the Kinect noise model[33]. From this dataset, we chose subsets of 40, 12, 8, and 5 partial views per object and used them as a training set. The selected partial views are rendered from equally spaced view angles.

By computing different descriptors, and weights, we compare each partial view in the testing dataset with all those in the training dataset. The classification follows a nearest neighbor approach.

Figure 5.12: Complex objects, retrieved from 3D Google Warehouse, used in our experiments.

The aggregated precision results are presented in Figure 5.13, while Figure 5.14 shows the confusion between chairs for the larger and the smaller object library. Results show that with the

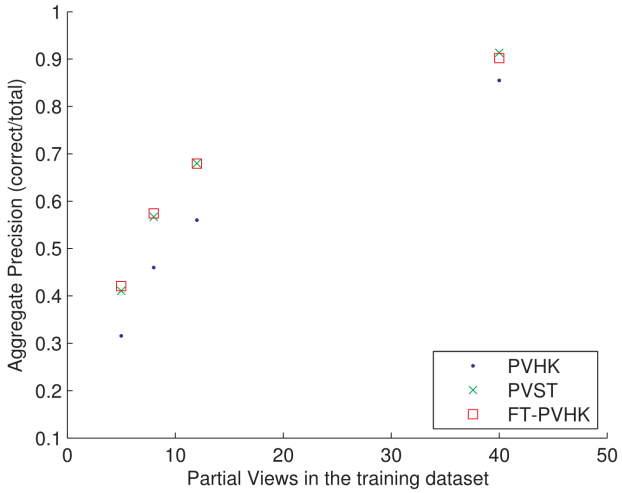


Figure 5.13: Aggregate precision using each of the three methods on the chairs dataset.

proper handling of parts, we considerably improved the recognition of complex objects.

On the other hand, it also shows that we can obtain comparable results using both the PVST descriptor and the FT-PVHK. However, PVST descriptor takes considerably more time to compute, as it requires larger set of eigenvalues and eigenvectors.

The results show that the FT-PVHK improves significantly recognition when compared with the PVHK. The confusion matrices are mostly diagonal for both the FT-PVHK and the PVHK, even when there are only five partial views per object in the training dataset. The FT-PVHK classifies with more accuracy than PVHK, showing the impact of our new weighted modified Hausdorff distance in complex objects.

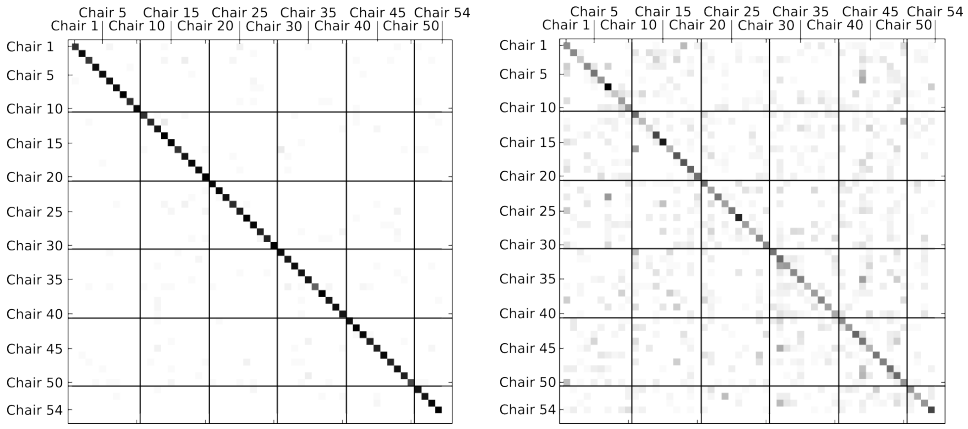
5.6 Summary

In this chapter, we described complex objects and showed that in those the PVHK is not so informative as in regular objects. Moreover, we showed that complex objects have large diffusion time scales and that these time scales, originally used compute the PVHK descriptor, are not adequate to represent all objects.

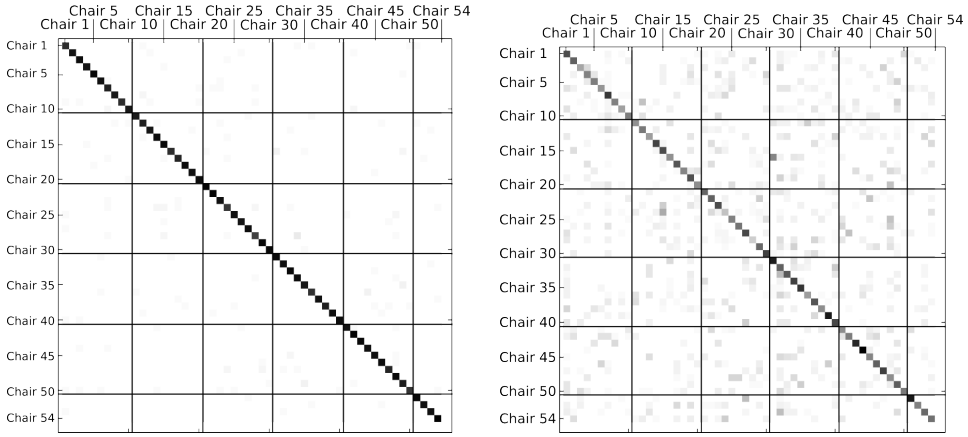
Furthermore, we proposed two other approaches also based on the heat propagation. The first represents the temperature at a time instant smaller than the global time scale. We also introduce a new measure of similarity between objects, that reduces in weight of parts in the distance between descriptors. The second approach relies on the time a vertex requires to reach a predefined temperature at a time scale, but the time it takes a vertex to reach a predefined temperature.

We showed numerical results on a complex object dataset, and concluded that the time based descriptor performs better at representing partial views of complex objects than any of the other proposed descriptors. However, it takes too long to compute. On the other hand, we achieved a good precision using the modified distance to evaluate the fixed time partial view heat kernel. While the overall precision was lower than the one achieved with the PVST, it still performed better than the PVHK and was as fast to compute as the latter.

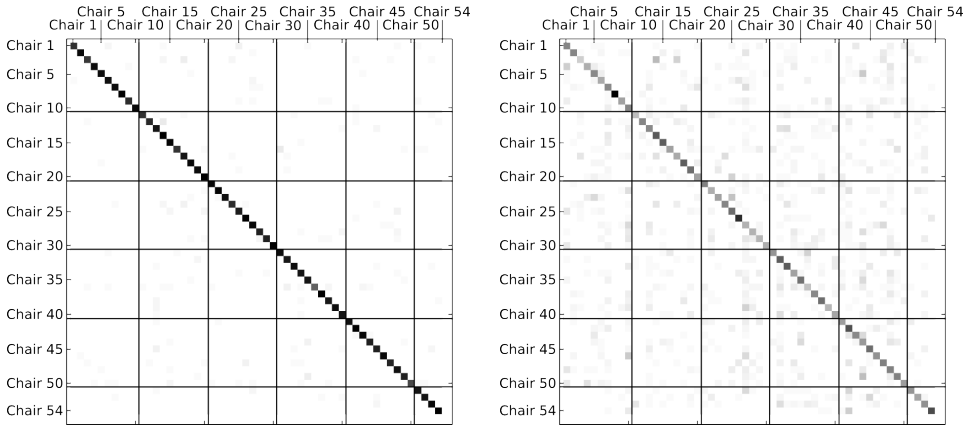
Size of object library	
40 partial views/object	5 partial views/object



(a) PVHK



(b) PVST



(c) FT-PVHK

Figure 5.14: Confusion matrices using object libraries of different sizes.

Chapter 6

Source Placement and Compact Libraries

In this chapter, we address the source placement in a partial view. In Chapters 2 to 5, we used an a-priori set of rules to consistently define a source position for each partial view, ensuring that the same partial view has the same descriptor both when constructing the library and when observed in any other context. However, the source position obtained by those rules is sensitive to noise, and to small changes in the sensor position. Complex objects, with their parts, are especially problematic as, when the source moves between parts, the descriptor may change considerably significantly, resulting in large estimated distances between partial views of similar shape. We here assume that any new observed partial view should have a descriptor similar to those stored in the library. In Section 6.2 we consider multiple possible sources for each new partial view and choose one that best approximate those in the library. In Sections 6.3.1 and 6.3.2 we present our algorithm for selecting sources and descriptors that create libraries that best represent the set of possible observations. We empirically evaluate the performance of our new approaches on two datasets of complex objects in Section 6.4.

6.1 Impact of Noise in the Heat Source Position

In previous chapters, we used simple rules that ensure that the source position is always the same for the same object when we observe it from the same viewing angle in different and independent observations, even when we have no prior knowledge on the object class and viewing angle. The rules we used are: i) the selection of the closest point to the observer; and ii) the selection of the vertex closest to the center of the segmented depth image.

In both cases, the heat source changes with the sensor position even when the partial view shape remains unchanged. And while the PVHK and the PVST descriptors change smoothly with small changes in the source position, the source itself may move considerably over the objects as the observer moves or as noise changes the vertices position.

In Figure 6.1, we show how parts in complex objects can interfere with the source position. In the example, we choose the source as the center of the segmented depth image and consider two partial views obtained by slightly changing the viewing angle. While there was barely no change in the observer position and the partial view shape, the source moved from the chair arm to the chair seat, leading to drastic changes in the partial view descriptor.

Figure 6.1: Impact on the PVHK by changes in the source position due to changes in the observer position when we choose the source as the point closest to the observer.

In Figure 6.2, we show how choosing the source based on the distance between the observer and surface may still lead to changes in the source position, especially on planar surfaces. In planes parallel to the sensor, only sensor noise impacts the distance to observer, affecting which vertices are selected as heat sources.

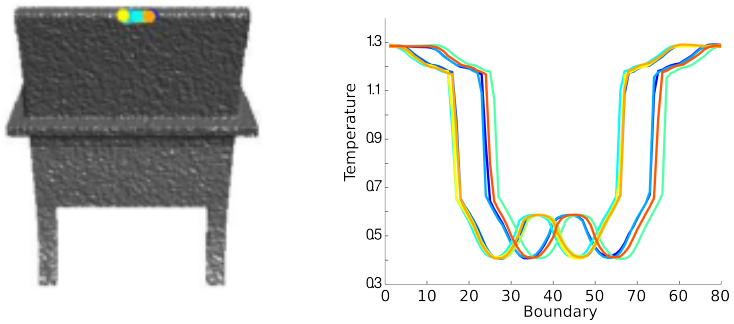


Figure 6.2: Impact on the PVHK by changes in the source position due to noise when we choose the source as the point closest to the observer.

In recognition tasks, where off-the-chart descriptors, resulting from unexpected source positions, are not realistic. Next, we assume that sources should be chosen so that descriptors match the object library and introduce different approaches for selecting the source both at recognition time and at library construction.

6.2 Source Selection for Observed Partial Views

So far, to classify observed partial views, ν[1], we first compute the descriptor that best represents it by: i) estimating a source position; ii) simulating the heat diffusion over the surface; and iii) using the temperature at the boundary as a descriptor. Then, to classify ν, we search over all descriptors in the object library and search for the most similar to the descriptor of ν and assume that they are both from the same object.

Here, we combine the representation and classification steps by: i) assuming that there are more than one possible heat source, and hence descriptors, for ν; and ii) searching over all pairs $(\bar{z}_1^{\nu}, \bar{z}^{\mathcal{O}})$ of descriptors, the first from ν and the second from the object library $\mathcal{O}$, we retrieve the most similar descriptors $\bar{z}_{1,*}^{\nu}$, $\bar{z}_{*}^{\mathcal{O}}$. We represent ν by $\bar{z}_{1,*}^{\nu}$, and classify it based on the label of $\bar{z}_{*}^{\mathcal{O}}$.

6.2.1 Multiple Descriptors from Multiple Heat Sources

We consider as possible sources for an observed partial view, ν, a subset of its vertices, $V_{\nu}^{s} \subseteq V_{\nu}$. For each source in V_{ν}^{s} we compute a descriptor, obtaining a set of possible descriptors, $\tilde{Z}_{\nu} = \{\bar{z}_1^{\nu}, ..., \bar{z}_{N_p}^{\nu}\}$. Figure 6.3 provides an example of possible sources, marked in red, and possible descriptors for the partial view of a chair.

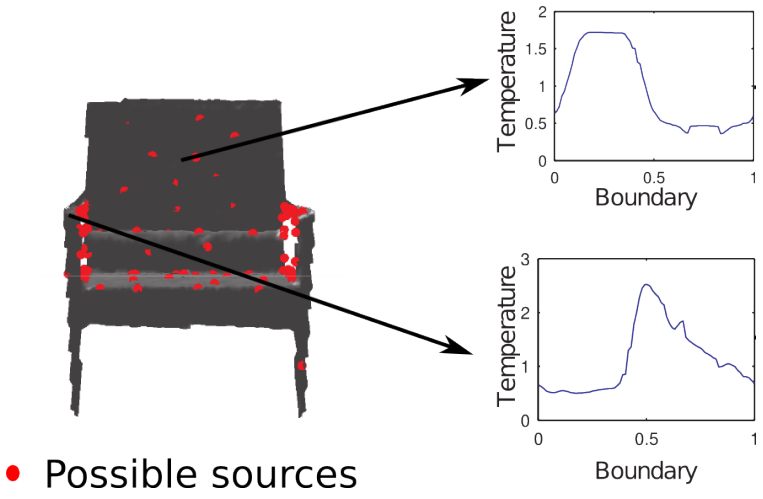


Figure 6.3: Possible sources and descriptors for a chair partial view.

The time required to compute several descriptors does not increase with the number, not change as the most time consuming step is the computation of eigenvectors and eigenvalues, which is done once per partial view. However, the computational effort while classifying a new partial view depends on number of possible descriptors. Ideally, we would use all vertices as possible heat sources, however when the number of resulting possible descriptors would be too large. Thus, we use mesh simplification algorithms, e.g., [24], to extract a sub-sample of possible heat sources at

[1]Partial views in this chapter are represented by Greek letters as ν and μ and s represents the source vertex

positions that represent the partial view shape. We also represent a single partial view in the object library by a single descriptor, to avoid the explosion of computational effort.

We thus introduce a combined approach where, as until now, in the library of known objects we only keep a descriptor per partial view, $\bar{z}^{\mu}_{s_\mu}$, computed using some source s_μ. But, when we observe a new partial view, ν, we use its complete set of possible descriptors, $\tilde{Z}_\nu$, to find the one that best matches any descriptor in $\mathcal{O}$. We thus have two very different sets of descriptors: the first is the set in the object library and the second is the set of possible descriptors from an observed partial view.

6.2.2 Representation and Classification

In our combined approach, we classify a newly observed partial view, ν, by computing the distance, $D^{obs,lib}$, between the set of possible descriptors, $\tilde{Z}_\nu$, and the descriptor of each partial view $\mu in \mathcal{O}$. Thus, we define this distance as the minimum distance between elements of the set of possible descriptors in ν, $\tilde{Z}_\nu$, and the descriptor used to represent the partial view μ in the library, $\bar{z}^{\mu}$.

$$D^{obs,lib}_{\nu,\mu}(\tilde{Z}_\nu, \bar{z}^{\mu}) = \min_{\bar{z} \in \tilde{Z}_\nu} \|\bar{z}^{\mu} - \bar{z}\| \tag{6.1}$$

We again classify a newly observed partial view using a nearest neighbor approach, i.e., based on the object class of its closest partial view in the library. Algorithm 6.1 summarizes the steps for representing and classifying a new observation, assuming multiple descriptors.

Algorithm 6.1: Represent and classify a new partial view from a set of possible sources.

Input: Object library, $\mathcal{O} = \{(\mu_1 = (o_1, \bar{\theta}_1), \bar{z}^{\mu_1}), (\mu_2, \bar{z}^{\mu_2}), ..., (\mu_K, \bar{z}^{\mu_K})\}$, Set of possible descriptors $\tilde{Z}_\nu$
Output: Object Class, o.
INITIALIZATION
$d_{min} \leftarrow 1000$
$o \leftarrow 0$
forall $(\mu' = (o', \bar{\theta}'), \bar{z}^{\mu'}) \in \mathcal{O}$ **do**
  $d \leftarrow D^{obs,lib}_{\nu,\mu'}(\tilde{Z}_\nu, \bar{z}^{\mu'})$
  if $d < d_{min}$ **then**
    $d_{min} \leftarrow d$
    $o \leftarrow o'$
  end
end

Given this classification, we increase the probability of miss-classifications when the observed partial view does not match exactly any partial view in the library. When we have many possible descriptors for $\bar{\nu}$, but no exact match in the library for any of them, the miss-classification may happen if a single descriptor of another object is closer.

However, as we dropped the rules for heat source placement when, we are no longer constrained

by them while creating the library. In the following, we address the problem of creating the library of partial views, so that avoid miss-classifications.

6.3 Source Selection for Object Libraries

To avoid miss-classifications, a library should ensure that all possible descriptors $\tilde{Z}_{\nu^{o_1}}$ of the possible partial views ν^{o_1} of object o^1, are closer to the descriptors of o^1 in the library, $\bar{z}^{\mu^{o_1}}$, than to the descriptors of all possible objects, $\bar{z}^{\omega^{o_2 \neq o_1}}$. This condition translates to:

$$\forall_{\nu^{o_1}=(o_1,\bar{\theta}^1)}, \quad \forall_{\mu^{o_1}=(o_1,\bar{\theta})\in\mathcal{O}} \quad \forall_{\omega^{o_2}=(o^2\neq o^1,\bar{\theta}^2)\in\mathcal{O}}, \quad D^{obs,lib}_{\nu^{o_1},\mu^{o_1}}(\tilde{Z}_{\nu^{o_1}}, \bar{z}^{\mu^{o_1}}) < D^{obs,lib}_{\nu^{o_1},\omega^{o_2}}(\tilde{Z}_{\nu^{o_1}}, \bar{z}^{\omega^{o_2}}). \tag{6.2}$$

The problem of jointly selecting a set of descriptors that ensure the condition in Eq.6.2 is ill posed as there are possibly multiple sets of sources per partial view that we could select. On the other hand, attempting to formulate the problem as an optimization problem, would yield a very large binary problem, of the order of hundred of thousand variables.

In the following, we present two approaches to finding a set of sources that approximate the above condition for a given dataset of objects and partial views. We consider that the set of possible descriptors for all the partial views in the library is a good approximation to the set of all possible descriptors from all possible partial views, even those that are not present in the library. We then select source positions that not only ensure the above condition, as also generalize well to more partial views.

In a first approach, we start by selecting sources from the set of possible descriptors in each partial view in the library that favor large distances between objects. In the second approach, we select sources that favor small distances within the same object.

6.3.1 Rewarding Distances to Other Objects

We want to ensure that possible descriptors of an observed partial view ν from object o are as far as possible from the descriptors in the library of all other objects o'. So, we maximize the distance of descriptors of o' in the library, $\bar{z}^{\mu^{o'}}$, to all possible descriptors of ν, $\tilde{Z}$. For a library of known objects, $\mathcal{O}$, seen by a set of K view angles, we choose the source vertex s_μ for each partial view $\mu \in \mathcal{O}$ based on the distance between the resulting descriptor $\bar{z}^{\mu}_{s_\mu}$ and all the possible descriptors of the other objects:

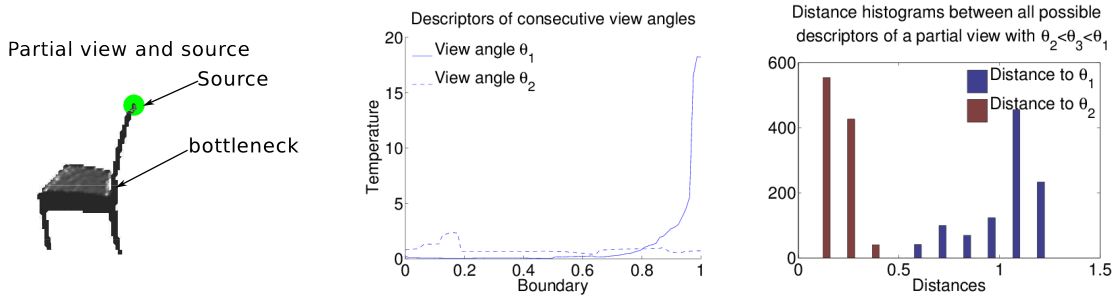
$$s_{\mu=(o,\bar{\theta})} = \underset{v\in V_\mu}{\operatorname{argmax}} \min_{\nu=(o'\neq o,\bar{\theta})\in\mathcal{O}} D^{obs,lib}_{\nu,\mu}(\tilde{Z}_\nu \bar{z}^{\mu}_{v}) \tag{6.3}$$

$$= \underset{v\in V_\mu}{\operatorname{argmax}} \min_{\bar{z}\in\tilde{Z}_\nu} \|\bar{z}^{\mu}_{v} - \bar{z}\|. \tag{6.4}$$

The solution of 6.3 favors sources that lead to descriptors very different from all the possible descriptors of other objects, regardless of descriptors of the same object.

In complex objects, such sources usually end up in small and loosely connected parts. The resulting descriptors are very different from any possible descriptor of other partial views of the same object. Due to segmentation problems or small changes in the viewing angle, small parts may not appear in observed partial view mesh. When the part of the object where we place the source disappears, the descriptor becomes unattainable using the remaining partial view possible sources. Thus, descriptors resulting from sources in the small parts are not usually reproducible, i.e., they are outliers.

We illustrate the impact of outliers with the example in Figure 6.4, where we consider three partial views of the same object, $\mu_1 \in \mathcal{O}$, $\mu_2 \in \mathcal{O}$ and ν, obtained from close viewing angles, with $\theta_{\mu_1} < \theta_\nu < \theta_{\mu_2}$, but that only μ_1 and μ_3 are in the object library $\mathcal{O}$. Furthermore, assume that μ_1 has a source in a small and loosely connected part, represented in Figure 6.4(a), while the third partial view has a source in the object main part. The descriptors $\bar{z}^{\mu_1}$ and $\bar{z}^{\mu_3}$ are represented in Figure 6.4(b).



(a) Partial view and source (b) Resulting descriptors (c) Distance to a partial view.

Figure 6.4: Example of a partial view, collected from view angle θ_1, whose descriptor in the dataset resulted from a source in a small part.

We then assume that ν is an observed partial view, that we need to represent and classify. The distances between its possible descriptors and the descriptors of $\bar{z}^{\mu_1}$ and $\bar{z}^{\mu_2}$ are represented in the histogram of Figure 6.4(c). In the histogram, it is clear that there is a very large distance between all the possible descriptors in ν and $\bar{z}^{\mu_1}$. In fact, the shortest distance between the first partial view is larger than the worst case distance to $\bar{z}^{\mu_2}$. If $\bar{z}^{\mu_2}$ was not present in the library, most likely we would not be able to identify ν based on $\bar{z}^{\mu_1}$. Thus, $\bar{z}^{\mu_1}$ behaves as an outlier.

When the object library has a large number of partial views for each object, an outlier does is not a problem. For each outlier there are other similar enough partial views, i.e., there are many $\bar{z}^{\mu_2}$ for any $\bar{z}^{\mu_1}$ that we include. But when the library is composed of only a few examples, each outlier introduced corresponds to one less example of a given object, and the overall accuracy of recognition is affected.

6.3.2 Penalizing Local Variability

In a second approach, we avoid the inclusion of outliers by imposing that close view angles must have similar descriptors.

The assumption is that if two partial views of the same object, ν_1 and ν_2, retrieved from viewing angles, θ_1 and θ_2, have similar descriptors, $\bar{z}^{\nu_1}$ and $\bar{z}^{\nu_2}$, then, in a new partial view ν_3 with a view angle $\theta_3 \in [\theta_\nu, \theta_\mu]$, there must be a possible descriptor $\bar{z}$ that is also very similar to both $\bar{z}^{\nu_1}$ and to $\bar{z}^{\nu_2}$. We aim at ensuring that our library generalizes well for new partial views.

We impose constraints on the variability of descriptors of any given object. We impose those constraints by rewarding descriptors of partial views that are similar to descriptors of partial views of close viewing angles, i.e., by penalizing local variability.

Let us consider an object library whose partial views ν_i are constrained to a fixed elevation, $\phi_i = \phi_0$, with azimuths evenly sampled around the object $\theta_i = i \times 360/N_\theta$.

Definition 2. The descriptor local variation, $\Delta\bar{z}^{\nu_i} = d(\bar{z}^{\nu_i}, \bar{z}^{\nu_{i+1}})$, measures how much the descriptors in the library change between two consecutive partial views.

Thus, the local variation is a function of the heat source positions on partial views ν_i and ν_{i+1}, respectively s_{ν_i} and $s_{\nu_{i+1}}$.

To ensure that similar partial views have similar descriptors, we aim at decreasing $\Delta\bar{z}^{\nu_i}$ over the complete set of partial views, i.e., we solve the problem in Eq. 6.5, where $\bar{v}_s$ is a vector whose entry $[\bar{v}_s]_i$ is the source vertex on partial view ν_i.

$$\bar{v}_s = \arg \min_{\bar{v}:[\bar{v}]_i \in V^s_{\nu_i}} \sum_{i=1}^{N} \Delta\bar{z}^{\nu_i}_{[\bar{v}]_i}. \tag{6.5}$$

This problem can be formulated as a linear optimization problem, provided a-priori knowledge of the distances between the possible descriptors of consecutive view angles. In particular, we can formulate it as shortest path problem, by representing sources in partial views as nodes in a graph.

The graph, which we depict in Figure 6.5, is a layered graph, created by connecting all the possible sources in partial view ν_i to all the possible sources of the neighboring partial views: ν_{i-1} and ν_{i+1}.

A node n_i in the graph corresponds to a descriptor of object o, view angle θ_l and source position v_k. Edges connect nodes from different, but consecutive view angles. For example, the edge e_{N_1+1} connects the node n_1, on the partial view with $\bar{\theta}_1$, with the node n_{N_1+1}, on partial view $\bar{\theta}_2$. Furthermore, each edge e_i, connecting the node n_k to the node n_l, has an associated cost $[\bar{c}_o]_i$, which reflects the change in the descriptor from placing the source in n_k and in n_l, i.e.,

$$[\bar{c}_o]_i = d(\bar{z}^{\nu_i}_k, \bar{z}^{\nu_{i+1}}_l) \tag{6.6}$$

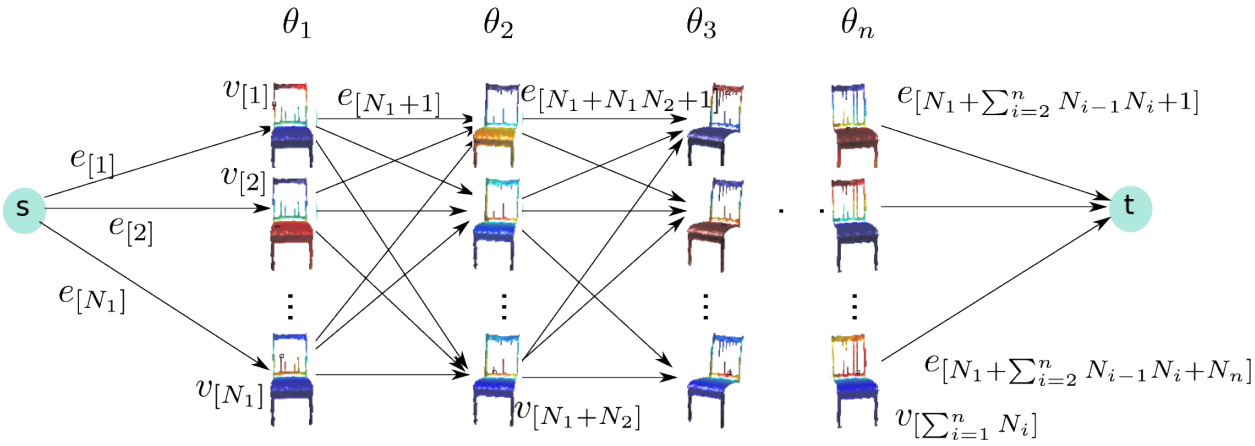


Figure 6.5: Graph representing all possible combinations of descriptors for a single object. Nodes correspond to possibles sources and edges the change in descriptors from consecutive view angles.

The set of descriptors $Z_o \in \mathcal{O}$ for object o, is generated by choosing for each partial view, a source that globally minimizes changes in the descriptor from viewing angle to viewing angle. We minimize $\bar{c}_o^T \tau$, where $\bar{\tau} \in \{0,1\}^{N_l}$ is defined over the edges so that $\tau_i = 1$ if and only if the edge e_i is selected.

Also, the set of edges has to form a closed path, in the sense that the arrival node of one edge has to be the start point of another edge. This constraint can be represented by ensuring that, at each node, the number of selected input edges is the same as the number of output edges, i.e., that $A\bar{\tau} = \bar{b}$, where $A \in \{0,-1,1\}^{N_n \times N_E}$ is the graph incidence matrix, i.e., $[A]_{l,j} = 1$ if and only if e_j arrived to n_l and $[A]_{k,j} = -1$ if and only if it leaves n_l. Furthermore, $\bar{b} \in \{0,1\}^{N_n}$ represents the difference between input and output vertices and thus is equal to zero on all the nodes associated with descriptors. However, to ensure that $\bar{\tau} = \bar{0}$ is not a solution to our problem, we add two extra nodes in the graph, s and t, for which $[\bar{b}]_s = 1$ and $[\bar{b}]_t = -1$.

The problem in Eq.6.5 is then equivalent to:

$$\bar{\tau}^* = \operatorname{argmin} \bar{c}_o^T \bar{\tau} \tag{6.7}$$

$$s.t. A\bar{\tau} = \bar{b} \tag{6.8}$$

$$[\bar{\tau}]_i \in \{0,1\} \tag{6.9}$$

This is a binary linear optimization problem, with an unimodular constraint matrix A. Thus, Eq. 6.9 can be relaxed to the continuum, $[\bar{\tau}]_i \in [0,1]$ and solved with a generic linear programming solver.

Finally, the sources $\bar{v}_s$ that we must use to compute the object library descriptors, correspond to the nodes in the edges for which $\bar{\tau}^*$ is equal to one.

Algorithm 6.2 describes the main steps required to select a set of descriptors that minimize the

variance between consecutive partial views. The algorithm receives as input an extended object library, $\mathcal{O}_e = \{(\nu_1^{o^1} = (o_1, \bar{\theta}_1), \tilde{Z}_{\nu_1^{o_1}}), (\nu_2^{o^1} = (o_1, \bar{\theta}_2), \tilde{Z}_{\nu_2^{o_1}}), ..., (\nu_K^{o_{N_o}} = (o_{N_o}, \bar{\theta}_K), \tilde{Z}_{\nu_K^{N_o}})\}$, which corresponds to the usual object library, but where each partial view is represented by the set of all possible descriptors.

Algorithm 6.2: Select sources by penalizing changes in descriptors of the same object.

Input: Extended object library,
$\mathcal{O}_e = \{(\nu_1^{o^1} = (o_1, \bar{\theta}_1), \tilde{Z}_{\nu_1^{o_1}}), (\nu_2^{o^1} = (o_1, \bar{\theta}_2), \tilde{Z}_{\nu_2^{o_1}}), (\nu_K^{o_{N_o}} = (o_{N_o}, \bar{\theta}_K), \tilde{Z}_{\nu_K^{N_o}})\}$
Number of possible sources per partial view $N_s^{\nu_1^{o_1}}, N_s^{\nu_2^{o_2}}$
Output: Sources $\bar{v}_s^o, \forall o \in \mathcal{O}_e$
Computing edge weights $\bar{c}_0$
forall $o \in \mathcal{O}_e$ **do**
  $j \leftarrow 0$ **forall** $\nu_i^o = (o, \theta_i) \in \mathcal{O}_e$ **do**
    forall $v \in \tilde{V}_{\nu_i^o}$ **do**
      Computing distances to consecutive partial views
      forall $y \in \tilde{V}_{\nu_{i+1}^o}$ **do**
        $j \leftarrow j + 1$
        $[\bar{c}_o]_j \leftarrow d(\bar{z}_v^{\nu_i^o}, \bar{z}_y^{\nu_{i+1}^o})$;
      end
    end
  end
  Constructing the incidence matrix $\bar{A}$ and the continuity vector $\bar{b}$
  $[A, \bar{b}] \leftarrow$ computeAdjancyAndContinuity$(N_s^{\nu_1^o}, ..., N_s^{\nu_K^o})$
  Using a linear solver to find a set of edges
  $\bar{\tau}^* \leftarrow solveLinearProblem(\bar{c}_o, A, \bar{b})$
  Converting edges to sources and computing descriptors
  $s^o \leftarrow sourcesInEdges(\bar{\tau}^*)$
end

6.3.3 Combined Approach

We combine the two previous approaches to obtain descriptors that maximize the distance to other objects and minimize the distance to the same object. Namely, we reduce the cost of edges connected to sources that yield descriptors that are very different from descriptors of other objects.

For each edge e_i, connecting the node n_k to node n_l, we assess how far are the node descriptors to the set of possible descriptors of all the other objects, and penalize those that are close. The penalty takes the form of a cost $[\bar{w}_o]_i$, defined as:

$$[\bar{w}_o]_i = -\min_{\bar{z} \in \tilde{Z}_\mu} \|\bar{z}_k^{\nu_i} - \bar{z}\| - \min_{\bar{z} \in \tilde{Z}_\kappa} \|\bar{z}_v^{\mu} - \bar{z}\|, \forall_{\mu,\kappa \in \mathcal{O}} \tag{6.10}$$

The new cost $[\bar{g}_o]_i$ for edge e_i is then:

$$[\bar{g}_o]_i = \alpha[\bar{c}_o]_i + (1-\alpha)[\bar{w}_o]_i, \tag{6.11}$$

where $[\bar{c}_o]_i$ is the penalty for large variations in the descriptor of consecutive viewing angles and $\alpha \in [0,1]$ is a mixing parameter, which allow us to decide whether we want to benefit more the first or the second method.

The combined approach corresponds to solving:

$$\bar{\tau}^* = \text{argmin}\, \bar{g}_o^T \bar{\tau} \tag{6.12}$$

$$s.t. A\bar{\tau} = \bar{b} \tag{6.13}$$

$$[\bar{\tau}]_i \in [0,1]. \tag{6.14}$$

The Algorithm 6.3 provides the main steps for computing the set of sources, that lead to a compact object library, i.e., a library where the descriptors of the same object are close together and as far away as possible of the descriptors of the other objects.

6.4 Numerical Results

We empirically tested the algorithm on a set of three chairs, represented in Figure 6.6(a) and on another set of 4 guitars represented in Figure 5.12(b). For each object class, we construct 4 object libraries, with 40, 12, 8 and 5 partial views per object.

Using the representation and classification method in Algorithm 6.1, we tested the source selection for the construction of object libraries in Algorithm 6.3, and compared four different values for the mixing parameter $\alpha = \{0, 0.25, 0.75, 1\}$. We also compared with the initial approach for the construction of the object library. The line labeled as *Initial* in plots of Figures 6.7(a)-(b) corresponds to a source placed at the center of the 2D segmented partial view. While creating the object library, we simplified the mesh to 250 vertices. We considered each vertex as a potential source.

The testing dataset corresponds to sets of 120 partial views from each object. We did not perform any type of mesh simplification, and all the vertices in the meshes were used as possible source positions.

Results in Figure 6.7(a)-(b) show that mixtures of the two approaches provide the most reliable libraries. The impact is mostly noticeable when we use small sets of partial views in the object library.

The results obtained in the two datasets are particularly exciting when compared to those obtained using our initial source placement criteria for the objects in the dataset. The careful tailoring of the object library allowed to improve results by almost 10% for the sparsest datasets.

Algorithm 6.3: Selecting sources for constructing a compact object library.

Input: Extended object library,
$\mathcal{O}_e = \{(\nu_1^{o^1} = (o_1, \bar{\theta}_1), \tilde{Z}_{\nu_1^{o_1}}), (\nu_2^{o^1} = (o_1, \bar{\theta}_2), \tilde{Z}_{\nu_2^{o_1}}), (\nu_K^{o_{N_o}} = (o_{N_o}, \bar{\theta}_K), \tilde{Z}_{\nu_K^{N_o}})\}$
Number of possible sources per partial view, $N_s^{\nu_1^{o_1}}, N_s^{\nu_2^{o_2}}$
mixing parameter α
Output: Sources $\bar{v}_s^o$, $\forall o = 1, ..., N_o$
Computing minimum distances to all possible sources of the other objects
forall $o \in \mathcal{O}_e$ **do**
  forall $\nu_i^o = (o, \theta_i) \in \mathcal{O}_e$ **do**
    forall $v \in \tilde{V}_{\nu_i^o}$ **do**
      $\rho_v^{\nu_i^o} \leftarrow \min_{\mu=(o' \neq o, \bar{\theta}) \in \mathcal{O}_e} D_{\mu,\nu_i^o}^{obs,lib}(\tilde{Z}^\mu, \bar{z}_v^{\nu_i^o})$
    end
  end
end
Computing edge weights $\bar{c}_0$
forall $o \in \mathcal{O}_e$ **do**
  $j \leftarrow 0$ **forall** $\nu_i^o = (o, \theta_i) \in \mathcal{O}_e$ **do**
    forall $v \in \tilde{V}_{\nu_i^o}$ **do**
      Computing distances to consecutive partial views
      forall $y \in \tilde{V}_{\nu_{i+1}^o}$ **do**
        $j \leftarrow j + 1$
        $[\bar{c}_o]_j \leftarrow d(\bar{z}_v^{\nu_{i+1}^o}, \bar{z}_y^{\nu_{i+1}^o})$; Computing the cost of each edge
        $[\bar{w}_o]_j \leftarrow -\rho_v^{\nu_i^o} - \rho_y^{\nu_{i+1}^o}$
        $[\bar{g}_o]_j \leftarrow \alpha[\bar{c}_o]_j + (1 - \alpha)[\bar{w}_o]_j$
      end
    end
  end
  Constructing the incidence matrix $\bar{A}$ and the continuity vector $\bar{b}$
  $[A, \bar{b}] \leftarrow$ computeAdjancyAndContinuity$(N_s^{\nu_1^o}, ..., N_s^{\nu_K^o})$
  Using a linear solver to find a set of edges
  $\bar{\tau} \leftarrow solveLinearProblem(\bar{g}_o, A, \bar{b})$
  Converting edges to sources and computing descriptors
  $s^o \leftarrow sourcesInEdges(\tau)$
end

6.5 Summary

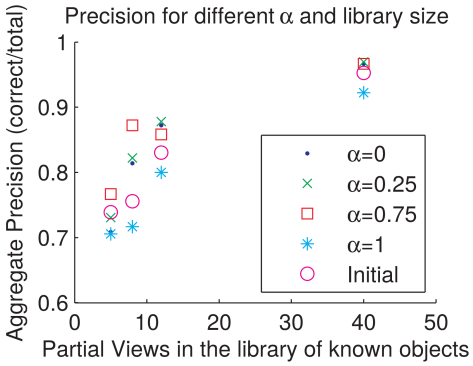
In this chapter, we show how the source position can be affected by sensor noise and position. We provided approaches for defining the source position, which depend on whether we are representing a newly observed partial view or are creating new object libraries.

For the representation of new partial views, the selection of sources aims at reproducing any descriptor in the object library. For the representation of partial views in the library, the source of

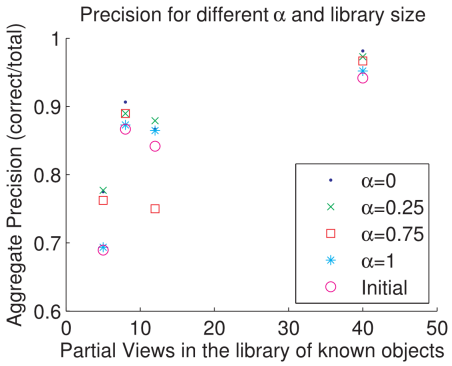
(a) Chairs.

(b) Guitars.

Figure 6.6: Datasets used for testing the accuracy on compact libraries using the PVST.



(a) Chairs



(b) Guitars

Figure 6.7: Aggregated precision for the chair and the guitar datasets using different approaches for source selection.

any partial view depends on the remaining objects in the library and should be chosen as to create compact libraries, which improve the overall accuracy.

We empirically tested of our approach and showed that mixed approaches performed much better than any other approach.

Chapter 7

Construction of 3D Models

In this chapter we present an algorithm, JASNOM, that allows the easy construction of extensive datasets using Joint Alignment and Stitching of Non-Overlapping Meshes [11]. We empirically show that our algorithm is able to create meshes of common objects, such as kettles and books as well as humans. Incidentally this complete 3D meshes can be used for the construction of object libraries, by offline rendering from new view angles.

7.1 Complete 3D Surface From 2 Complementary Meshes

We propose an algorithm, Joint Alignment and Stitching of Non-Overlapping Meshes (JASNOM), that requires little preparation and technical knowledge to create a complete 3D model, which can be used for offline rendering of partial views and dataset construction. JASNOM exploits the underlying manifold structure of range sensors data to recreate the object surface from just two range images.

Obtaining a pair of meshes that comply with these constraints can be easily achieved using active 3D cameras such as the Kinect camera. Since mesh boundaries are typically in regions of strong curvature, e.g., corners and edges, they do not change considerably under small perturbations on the view point. Thus non-overlapping meshes can be obtained by simply flipping objects, as illustrated in Figure 7.1, or roughly positioning two cameras in opposite directions of the object for non-rigid objects.

By not requiring a-priori camera registration nor extra apparatus, JASNOM provides a simplified process for object modeling. Furthermore, by using the boundary geometry for aligning meshes, JASNOM does not depend on geometric nor texture feature matching. In this work we illustrate the potential for fast object modeling using a non rigid object, a Human, and different small and regular objects, with compact surfaces.

Another possible application of JASNOM is to fill holes in a mesh. In the case of interactive object modeling, our algorithm allows a user to select parts from a mesh or library of meshes and use them to fill holes in an incomplete 3D model. The possibility of filling holes from other mesh

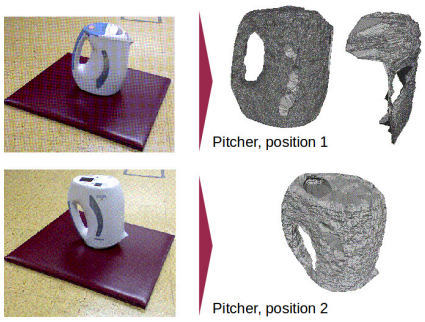


Figure 7.1: Example of a possible, and effortless, procedure for acquisition of two non-overlapping meshes using a Kinect sensor.

parts is of valuable use for modeling objects with self similar surfaces such as planes or cylinders, which are the basic shapes of the man-made objects that populate indoor environments.

JASNOM addresses jointly both the problem of registration and merging of meshes by aligning two meshes by their boundary. As depicted in Figure 7.2, JASNOM aligns two meshes, M_1 and M_2, and glues them to create a single mesh, M. While JASNOM applications can be extended to any problem that can be formulated by boundary alignment, e.g., puzzles, JASNOM was developed with a primary focus on 3D object modeling.

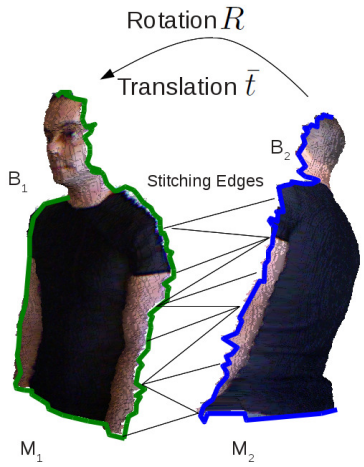


Figure 7.2: Construction of a mesh M from two other meshes, M_1 and M_2, by align both boundaries, B_1 and B_2 through a rotation R and a translation $\bar{t}$.

JASNOM aligns meshes by assuming that their boundaries are the same geometric structure but seen in different coordinate systems, i.e., that each point in one boundary has a corresponding point in the other. Under this assumption, stitching edges should connect corresponding vertices in the two boundaries and should have zero length. The stitching problem can be posed as that of finding correspondences between boundaries and the aligning problem as that of finding the rigid

transformation that minimizes the total edge length.

However, in a realistic scenario, boundaries do not exactly match and there is no a-priori knowledge on the correspondences between the boundaries. In this case, the previous solution would have three main draw backs: i) if the boundaries are strongly irregular, simple minimization of edge lengths may lead to intersections between meshes; ii) in general, finding correspondences between vertices is a combinatorial problem; iii) there is no guarantee that the correspondences by themselves will define a a triangular mesh that allows the completion of the mesh.

Our main contributions address these problems and allow the reconstruction of a triangular mesh between the two boundaries. Namely JASNOM:

- introduces a cost function that penalizes both the edge lengths and the intersection between meshes;
- introduces constraints that simplify the search for the assignments from a combinatorial problem to a discrete linear programming problem, solvable in linear time;
- introduces a stitching algorithm that reconstructs the triangular mesh given a set of assignments.

JASNOM penalizes the intersection between meshes by modeling the intersection as a set of local conditions to be verified by each stitching edge.

To constrain the search space for the assignments, JASNOM uses the fact that the resulting mesh should have the same properties as an object surface. E.g., object surfaces are 2D-manifolds and thus object surface meshes cannot have edges crossing each other except at vertices.

To reconstruct the mesh structure, JASNOM makes use of the assignments from the alignment stage and ensures that properties like mesh *manifoldness* are locally preserved.

7.2 Mesh Alignment

JASNOM addresses the problem of aligning and stitching two meshes M_1 and M_2 by focusing on the boundaries of each mesh, B_1 and B_2 as shown in Figure 7.3. In particular, JASNOM creates a complete mesh by assigning new edges from one boundary to the other and minimizing the total length of these edges by means of a rigid transformation. Furthermore, while minimizing edge length, it must prevent the meshes from intersecting each other. Formally, JASNOM solves an optimization problem whose cost function, J, is composed of two independent terms J_1 and J_2. The first term, J_1, penalizes the total edge length, while J_2 penalizes the intersection. The result of the optimization is the mesh alignment and a initial set of assignments that will later be used for stitching.

7.2.1 Minimizing Edge Lengths

To ensure that edges are as small as possible, JASNOM addresses the aligning of two meshes as a registration problem, where edges represent assignments between vertices in the two meshes. These

assignments are represented by a binary matrix A, whose element $A_{i,j}$ is equal to 1 if and only if vertex v_j in boundary B_1 is connected to vertex v'_i in boundary B_2. Assuming there are K vertices in B_1 and N vertices in B_2, $A \in \{0,1\}^{K \times N}$ and if no additional constraints are added, there are $2^{K \times N}$ different assignment matrices.

Matrix A defines a set of error vectors, $\bar{\xi}_i$, each associated to a stitching edge. The error vector represents the displacement between assigned vertices in the two borders:

$$\bar{\xi}_i = \left(\sum_{j=1}^{K} A_{i,j} \bar{x}_j \right) - \bar{y}'_i, \tag{7.1}$$

where $\bar{x}_j$ and $\bar{y}'_i$ are the coordinates of the vectors in B_1 and B_2 in the same coordinate system. However we only have access to the coordinates in their original coordinate systems, which differ by a rotation R and a translation $\bar{t}$. Therefore, the cost function J_1, responsible for minimizing the length of the stitching edges, is given by Eq.7.2.

$$J_1(A, R, \bar{t}) = \sum_{i=1}^{N} \|\bar{\xi}_i\|^2 = \sum_{i=1}^{N} \left\| \left(\sum_{j=1}^{K} A_{i,j} \bar{x}_j \right) - R\bar{y}_i + \bar{t} \right\|^2 \tag{7.2}$$

7.2.2 Preventing Intersection

To globally ensure that no intersection occurred, JASNOM would have to check for local intersections between each and all the vertices in one mesh versus each and all the faces of the other mesh. JASNOM relaxes the problem by considering only intersections between a vertex $v'_i \in B_2$ and the neighborhood of $v_j \in B_1$ to which it was assigned.

Local intersections can be modeled by keeping track of the position of mesh $M_{1,2}$ with respect to each vertex of the boundary $B_{1,2}$. This relative position is represented for each boundary vertex v by the normal to the boundary $\bar{n}_v$, as shown in the Figure 7.3. Keeping in mind that error vectors $\bar{\xi}$ point from vertices $v' \in B_2$ to vertices $v \in B_1$, if $\bar{\xi}_i$ points in the opposite direction of $\bar{n}_{v'}$, the vertex $v'_i \in B_2$ is on top of mesh M_1.

Ideally, preventing intersections would then result on a set of constraints in the optimization problem. However, since the estimation of the boundary normals is very sensible to noise and irregularities on the boundary, the constraints may yield the problem unsolvable. We thus relax these constraints by introducing them as a second term to the cost function, J_2. The constraints are modeled as a sum of logistic functions that receive as argument the projection of $\bar{\xi}_k$ on $-\bar{n}_{vk}$ and $\bar{n}_{v'k}$ as in Eq.7.3. The logistic function penalizes edges that cross the opposite boundary by

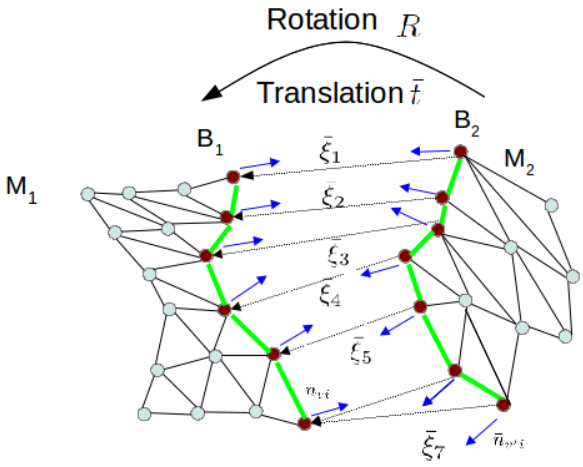


Figure 7.3: Example of two meshes connected by assigning edges from one boundary to the other.

penalizing the negative projections on $\bar{n}_{v'k}$ and the positive projections on $\bar{n}_{vk}$.

$$J_2(A,R,\bar{t},\alpha) = \sum_{k=1}^{N} \frac{1/N}{1+\exp\{\alpha\bar{\xi}_k \cdot \bar{n}_{vk'}/\|\bar{\xi}_k\|\}} + \sum_{k=1}^{N} \frac{1/N}{1+\exp\{-\alpha\bar{\xi}_k \cdot \bar{n}_{vk}/\|\bar{\xi}_k\|\}} \tag{7.3}$$

We introduce a slack variable α to control the steepness of the logistic function. High values of α correspond to steepest transitions on the logistic function and enforce the constraints more strictly. Lower values of lambda relax the constraints. The best value depends on the confidence on the normal estimation.

7.2.3 Minimizing the Cost Function

Formally, JASNOM aligns and stitches the two meshes by finding the matrices A^* and R^*, and the vector $\bar{t}^*$ that minimize the cost function in Eq. 7.4

$$A^*, R^*, \bar{t}^* = \operatorname{argmin} J(A,R,\bar{t};\alpha,\beta) = J_1(A,R,\bar{t}) + \beta J_2(A,R,\bar{t},\alpha) \tag{7.4}$$
$$s.t. \quad A \in \{0,1\}^{N\times K}, \quad R \in \mathcal{O}(3), \quad \bar{t} \in \mathbb{R}^3;$$

where $\beta \in \mathbb{R}^+$ weights the two cost functions and depends on the object or application. E.g., if the task is hole filling and the patch we use is smaller than the hole there will be no intersection and thus β can be set to zero.

Without further constraints, finding the matrix A is a combinatorial problem. However, we note that if the assignments between meshes correspond to edges in the mesh of an object, not all the assignments are valid. For example, no edge can cross the interior of the object. We explore

the physical constraints in the problem to reduce the number of possible assignments between the two meshes. The constraints, which we address in Section 7.3, are independent of the rigid transformation that aligns the two meshes.

JASNOM is then able to tackle separately the discrete problem of finding the assignment matrix A from the problem of finding the rigid transformation, R and $\bar{t}$. The separation and reduced complexity allow the algorithm to address the discrete problem by enumeration, i.e., JASNOM minimizes $J(A, R, \bar{t}; \alpha, \beta)$ by finding the minimum over the set of all valid assignments, $\mathcal{V}_A \in \{0,1\}^{N \times K}$, using exhaustive search.

The problem in Eq. 7.4 can be re-written as:

$$A^*, R^*, \bar{t}^* = \underset{A_\tau, R, \bar{t}}{\operatorname{argmin}} \tilde{J}(R, \bar{t}; A_\tau, \alpha, \beta) \tag{7.5}$$

$$s.t. \quad A \in \mathcal{V}_A \quad \text{(solved by enumerating all possible } A_\tau\text{)}$$

$$\tilde{J}(R, \bar{t}; A, \alpha, \beta) = \min_{R, \bar{t}} J(A, R, \bar{t}) \tag{7.6}$$

The optimization problem expressed in Eq. 7.6 is non-convex. To find a local solution, we use a generic non-linear optimization algorithm, such as BFGS Quasi-Newton method [15]. To initialize the optimization, JASNOM first solves the relaxed problem obtained from Eq. 7.4 by setting $\mu = 0$, which has a closed form solution [57].

7.3 Valid Assignments

Stitching assignments in JASNOM correspond to edges in an object surface and, as shown in Figure 7.4(a.2), these edges have a specific geometric structure. In the following, we address the geometric properties that can be used to constrain possible assignments and then present how JASNOM uses the constraints to efficiently find the best stitching edges.

7.3.1 Assignment Constraints

The complete surface mesh of an object is an orientable 2-manifold mesh, while an isolated part of the surface is an orientable 2-manifold mesh with a boundary. In Figure 7.4(a.2) we exemplify the mesh structure corresponding to an object part. In particular, we note that there are only two types of edges: those that belong to two triangles, and those that belong to only one, i.e., that are in the mesh boundary. Formal definitions of all these concepts can be found in computational geometry books, e.g., [44]. We briefly illustrate them here to allow a better comprehension of the constraints.

Object surfaces are orientable because they have an inside and an outside. Using one of these directions, it is possible to define consistently the normal directions for all points at the surface as shown in Figure 7.4(a). For 2-manifold meshes, the definition of a normal to a triangle is associated with a cyclic order of the triangle vertices. The normal to a triangle with vertices v_1, v_2 and v_3 with

coordinates $\bar{x}_1, \bar{x}_2, \bar{x}_3 \in \mathbb{R}^3$ can be estimated by the outer product $\hat{n}_F = (\bar{x}_2 - \bar{x}_1) \times (\bar{x}_3 - \bar{x}_2)$. If the order of the vertices changes, the direction of the normal vector will be the exact opposite. To ensure consistency on the orientation of two adjacent faces, the two vertices of the common edge must be in opposite order, as shown in Figure 7.4(a.3). Boundary edges have only one possible orientation since they belong to a single triangle. This orientation defines the intrinsic direction of the boundary cycle, as shown in Figure 7.4(b).

The whole surface mesh is orientable if all adjacent faces are consistent. To guarantee that the union of two meshes is orientable, their boundaries cannot have a random orientation with respect to each other. JASNOM stitches two meshes by assigning an edge from one boundary to the other. This situation, illustrated in Figure 7.4(b.2), requires the orientation of the boundaries to oppose each other. This is in consistency with the Gluing theorem.

Since the union of the two meshes is introduced by the assignment matrix A, the matrix must reflect the ordering of the two boundaries. We thus introduce the constraint:

$$A_{i,j} = 1 \Rightarrow A_{i+1,j+k} = 0, \quad \forall k \geq 0. \tag{7.7}$$

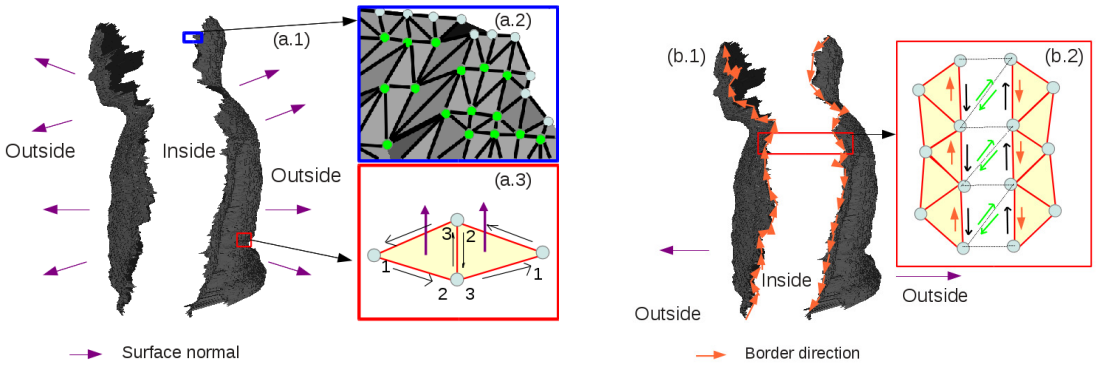


Figure 7.4: Order constraints in the boundary: (a) shows how the orientability of surfaces induces an ordering in the edges; (b) shows how the ordering reflects in the boundary.

7.3.2 Order Preserving Assignments

The space of matrices that satisfy the previous constraint is still very large. To further constrain the valid assignments search space, $\mathcal{V}_A$, we introduce some geometric constraints. In particular, we note that if the two meshes were the exact complementary of each other over the object surface, the two boundaries would correspond to the same vertices and edges. In this case, given a mapping $\varphi : B_2 \rightarrow B_1$ between the two boundaries that returns the point $v_j \in B_1$ equivalent to the point $v'_i \in B_2$, we can define the assignment between the two boundaries as $A_{i,j} = 1 \Leftrightarrow v_j = \varphi(v'_i)$.

To construct this mapping, we define one origin in each boundary, and order the vertices ac-

cording to the boundary orientation. Assuming that the origins correspond to the same point, two points that are at the same distance, i, from the origin, should be equivalent to each other. To account for the opposite boundary orientations, the mapping needs to invert the vertex ordering, e.g., as in $\varphi(v'_i) = v_{N-i}$. This is illustrated in Figure 7.5 where N refers to the total number of vertices in the boundary and i to the order of the vertex v'_i with respect to the boundary of B_2.

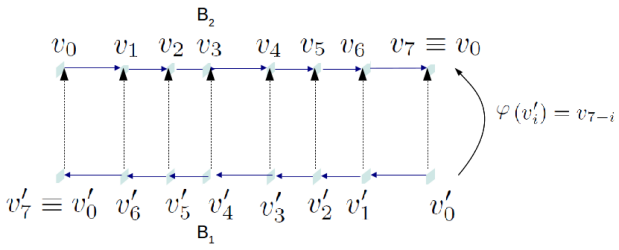


Figure 7.5: Example of construction of an assignment between boundaries in the limit case where the vertices in both boundaries coincide exactly.

For vertices of the two boundaries to map to each other, the sampling in both surfaces has to be exactly the same. Thus, in most cases, mapping the vertices order across boundaries does not preserve the object geometry. It is then more reasonable to map distances over the boundaries. In this work, we use the normalized curve length $l \in [0, 1]$ to account for those cases when the boundaries do not have the exact same length. In this case, the previous map can be rewritten as $\varphi(l') = 1 - l$.

After mapping a point between boundaries based on the normalized length, JASNOM still needs to find the closest vertex to that point. This search can be efficiently implemented by introducing an ordering function $f(l) : [0, 1] \rightarrow [0, N]$, which maps lengths over a specific boundary to a vertex order. For example, if vertex v_k is at a length l_k, $f(l_k) = k$. For values of l that do not correspond to exact vertices length but to points on the boundary edges, $f(l)$ returns the order of the closest vertex.

Using the map $\varphi(l')$ and knowing the ordering function $f(l)$ for B_1, we can find the order j of the vertex $v_j \in B_1$ to which assign $v'_i \in B_2$ by performing three steps. Namely:

i) computing the length $l'_i = l'(v'_i) = l'_{i-1} + \|\bar{y}_i - \bar{y}_{i-1}\|$;

ii) mapping the length l'_i to the length l of the equivalent point in B_1: $l = \varphi(l'(v'_i))$;

iii) finding the vertices in B_1 that have a distance to the boundary closest to l using the ordering function over B_1: $j = f(\varphi(l'(v'_i)))$.

The three steps are illustrated in Figure 7.6.

By repeating for all $v_i \in B_2$, JASNOM defines the assignment matrix A as

$$A_{i,j} = 1 \Leftrightarrow j = round(f(\varphi(l'(v_i)))) \tag{7.8}$$

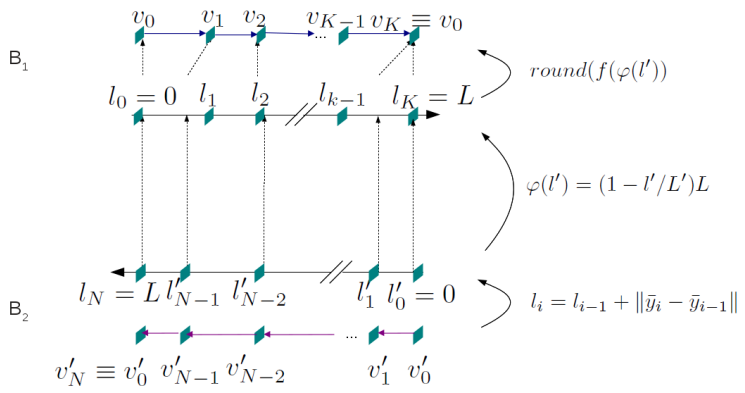


Figure 7.6: Three steps approach to define order preserving assignments between the boundaries.

The previous definition for A depends only on the map φ and the ordering function $f(l)$. However, both functions depend on the vertex defined as an origin on either boundary. If any other vertex $v'_\tau \in B_2$ was assumed to be equivalent to the origin, $v_0 \in B_1$, the mapping could be recovered by shifting l' by l_τ. This origin ambiguity is translated into N different valid maps between boundaries.

JASNOM addresses the ambiguity problem by considering all possible N different shifts τ of the boundary B_2 with respect to the boundary B_1. Each shift gives rise to a new mapping φ_τ and each mapping gives rise to a new assignment matrix A_τ. Thus, the combinatorial problem can be reduced into N independent problems. We note that by changing the shift in B_2 and not in B_1, the ordering function defined in B_1 will be the same in all the shifts in A_τ.

7.4 Final Stitching

After aligning both meshes, JASNOM uses the best assignment to reconstruct the manifold M_c. In particular, the assignment as defined in 7.8 ensures that each vertices in B_2 already has an edge connecting it to a vertex in B_1. However, not all the vertices in B_1 have an edge connecting to B_2 and some vertices in B_1 have more than one edge. Furthermore, just ensuring that there is an edge for all the vertices, does not ensure that the end result is a triangular mesh.

To stitch the meshes together, we use two simple strategies. First, we create a triangular mesh from the assignments already present. Then we assign the missing edges on B_1 so that they do not cross the edges already present.

For the first step, JASNOM adds a second edge to all the vertices $v'_i \in B_2$. As shown in Figure 7.7(b), the target vertex, $v_t \in B_1$ of the second edge of v'_i is the the first target of the next vertex, $v'_{i+1} \in B_2$.

In the second step JASNOM assigns the missing edges in B_1 by running through all the vertices $v_i \in B_1$ by their reverse order. As shown in Figure 7.7(c) each vertex with no edge is assigned the same target vertex $v'_t \in B_2$ as the target of the previous vertex $v_{j-1} \in B_1$.

This strategy locally ensures manifoldness since there are no crossings between neighboring

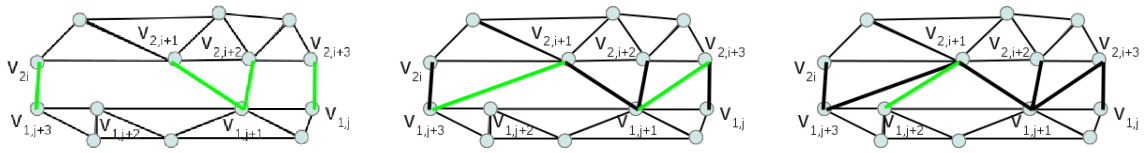


Figure 7.7: Schematic for the stitching between the two meshes given the set of one to one correspondences that result from the alignment stage.

edges. The constraints in the assignments ensure that the initial set of edges do not cross and the new edges always preserve the ordering between boundaries.

In summary, JASNOM creates a complete 3D object surface model from non overlapping meshes by enumerating all valid assignment matrices, $A_\tau \in \mathcal{V}_A$ and, for each matrix, finding the rigid transformation that minimizes the cost function $J(A_\tau, R, \bar{t}; \alpha, \beta)$. JASNOM chooses the best assignment as the one that minimizes the cost function over all the minima, and aligns the meshes accordingly. This assignment serves also as initialization to the stitching algorithm, where the missing triangles are added.

7.5 Proof of Concept

We test our stitching algorithm with three experiments. In the first we illustrate its potential for fast 3D object scanning by modeling two smooth objects. In the second, we illustrate its potential for reconstructing 3D models from articulated objects such as humans. Finally, in the third experiment, we illustrate its potential for hole filling.

For the first experiment, we model two objects. The first is the electric kettle, Figure 7.1, and the second a book, Figure 7.8. To collect both meshes for the example, we retrieve an image with the object in its regular position and then flip it upside down to collect the second image. The complete process is extremely fast from a user perspective and does not require previous registration of multiple cameras. The resulting complete meshes are presented in Figure 7.9.

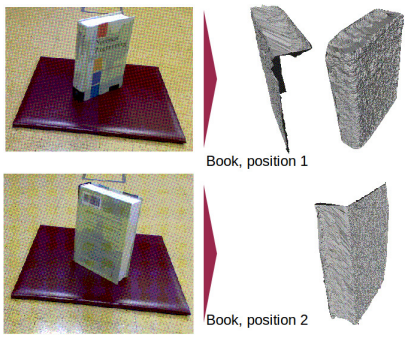


Figure 7.8: Acquisition setup for acquiring two meshes from a book.

For the purpose of accuracy while estimating centroids and other intermediate steps, JASNOM interpolates boundaries to ensure an uniform and dense distribution of points. To deal with the non-compactness of the object, JASNOM selected just the longest boundary. We note that the reconstructed objects show a good match at the boundaries.

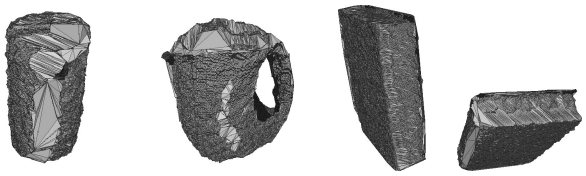


Figure 7.9: Reconstruction of man made objects using JASNOM. The first row presents two different views from the electric kettle and the second from the book.

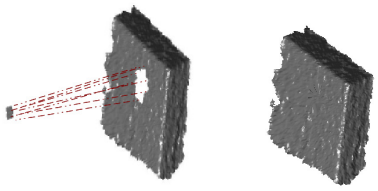
For the second experiment, two range images of the upper body of a human were retrieved simultaneously by two unregistered Kinect cameras. The complete mesh obtained with JASNOM algorithm is shown in Figure 7.10. We note that the two meshes do not cover the complete object and there are several large missing parts across the boundary. However, by preventing intersection, JASNOM was able to keep the overall human structure. In particular, the hole created by the cut at the waist is large enough that by simply attempting to minimize the distance between points, would lead to mesh intersections. Again we note that, with no previous camera registration, JASNOM created a rough shape of a non-rigid object using two Kinect cameras.

For the last experiment, we use a simple range image of an object with a hole and a small patch retrieved from another mesh, Figure 7.11(a). JASNOM covered and stitched the hole, Figure 7.11(b). Since the objective is to insert the patch on the hole in the other mesh, we did not penalize intersections between meshes, i.e., $\mu = 0$. We note that in this case the re-triangulation method left a smooth surface after patching the hole.

Back Front Top

Figure 7.10: Human model completed using JASNOM.

When compared with existing stitching algorithms, JASNOM adds the capability to create



(a) (b)

Figure 7.11: Results for the hole patching experiment using JASNOM. Figure 7.11(a) presents the original mesh with a hole and the patch. Figure 7.11(b) presents the glued mesh.

complete models without previous registration of individual meshes. The registration typically requires overlap between the two meshes, which is not always available or convenient. JASNOM also does not require the calibration of one camera position with respect to the other. The registration and construction of models can be easily achieved with little effort and setup preparation. This allows for the fast creation of extensive 3D (possibly 3D+RGB) models data sets.

JASNOM assumes that the two meshes are complementary over the object surface and, while we showed it could reconstruct objects in more general cases, e.g., the human shape, other objects might not be reconstructed so easily. In particular, we note that the boundaries of the human shape meshes, had a preferential direction, i.e., the elongated shape means that small deviations from the best assignment between boundaries lead to steep increases in the cost functions. More symmetric objects do not benefit from the steepness in the cost function and the alignment is more sensitive to gaps between boundaries. A possible approach, which we will explore in future work, is to reintroduce the asymmetries by penalizing color discontinuities at the boundaries.

7.6 Summary

we have contributed an algorithm, JASNOM, that allows the easy construction of extensive datasets using joint alignment and stitching of non-overlapping meshes. Furthermore, we provided evidence of its potential for fast 3D object scanning through simple experiments with data obtained with a Kinect camera.

From the experiments we here introduced, we conclude that JASNOM successfully constructs 3D models of different object types, including rigid and non rigid. The success of JASNOM is due mostly to the cost function definition. By preventing the intersection between boundaries, JASNOM preserves the object structure even with noisy boundaries. JASNOM is thus able to reconstruct complex shapes with missing parts such as the human we presented.

Chapter 8

Application to Automated Classification of Animals' Body Condition

In this chapter, we show how the tools we developed throughout this thesis are not constrained to object representation and can be pplied in different contexts. The opportunity to explore different uses for our representation arrived as an invitation from fellow colleagues from the Veterinary College of the Lisbon University to help estimating the Body Condition Score (BCS) in dairy farm goats. The BCS conveys information on whether an animal is fat or thin, and both very fat and very thin animals have poor milk production. We were challenged to devise methods that would allow to automate the estimation of the BCS while animals moved freely through a corridor. In an initial collaboration,[65], we showed that changes in the rump volume are strongly correlated with BCS. We here use 3D rump surfaces and a descriptor related to PVHK to classify very thin animals. In Section 8.1 we introduce the body condition score in goats and its possible assessment by visual, and volumetric, cues. In Section 8.2 we introduce all the steps from acquisition and pre-processing. In Section 8.3, we introduce our descriptor, the Heat Based Rump Descriptor (HBRD), and the algorithm to compute it. In Section 8.4 we show examples of the (HBRD), and how we were able to distinguish very thin animals in a group of 32 animals.

8.1 Visual And Volumetric Cues for Assessing the Body Condition Score in Goats

The Body Condition Score (BCS) evaluates an animal fat deposits and is an important indicator of the animal welfare, with implications in terms of milk production. In particular, very low or very high BCS, as those represented in Figure 8.1(a) and (c), are correlated with a decrease in milk production and are not in adherence with consumers expectations on animal's rights.

(a) Very thin (b) Normal (c) Very Fat

Figure 8.1: Examples of very thin, normal and very fat animals.

Also, the European Union recognized farm animals' right to freedom from hunger and thirst and is currently moving towards the introduction of BCS as a key indicator on welfare assessment protocols on goat farms. However, standard techniques for estimating the BCS in goats, e.g.,[27], cannot be used in large scale assessments, as they require restraining and handling of each animal individually by specially trained veterinaries.

During an initial collaboration, [65], we addressed the scalability problem by creating illustrations, the Body Condition Score Pictorial Scale, to allow non-experts to assess the BCS by visual inspection. For the construction of the Pictorial Scale, we identified several visual features in the rump region that are strongly correlated with the animal' BCS. Those features correspond to distances between bones and muscle folds, which are easy to identify visually. We used the features to define a *standard* individual of each class, from which a professional illustrator generated drawings for the scale. The Pictorial Scale can now be used in farms, but still requires trained evaluators.

The features we identified in the initial collaboration [65] worked well for the purpose of creating visually accurate illustrations. However, to retrieve such features, we took photographs taking careful control on conditions such as: i) animals stillness; and ii) rumps alignment with the camera. Both conditions are difficult to ensure without animal handling. We here move towards a scenario where no handling is required by using RGB-D cameras, as 3D information handles better changes in the orientation between camera and animal. Such cameras can be fixed on top of the animals' normal path, and can accurately collect data at roughly 2m from the animal.

RGB-D cameras provide both an RGB image and a depth image, from which we can recover 3D surfaces corresponding to the animal surface. From the whole animal, we extract the rump as showed in Fig. 8.2.

As noted in [65], the main difference between the different BCS categories are the fat reserves

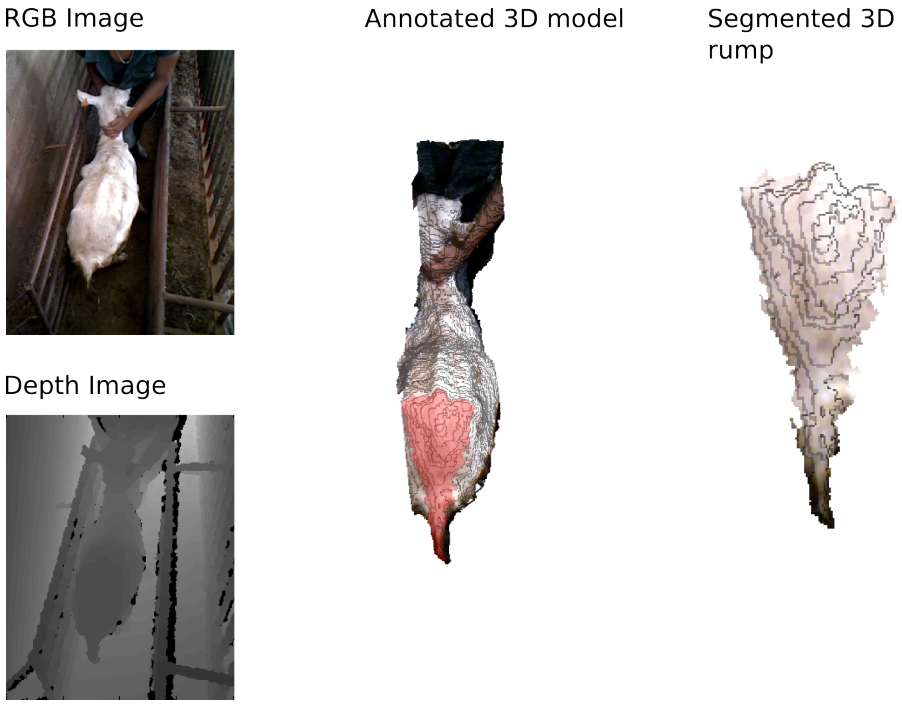


Figure 8.2: Acquiring rump 3D surfaces.

in the rump, which yield a bulkier appearance in fatter animals. To correctly access the animal class we focus on descriptors that represent changes in volume between rumps of different animals. Furthermore, the most noticeable changes in the rump volume concern its upper part, near the hip.

However, the direct comparison of volume between rumps 3D surfaces is very challenging, as: (i) rump shapes vary considerably among animals, regardless of BCS, as showed in Fig.8.1; and (ii) it is difficult to define consistently the rump region in a meaningful and consistent way.

So far, we used heat based descriptors to represent surfaces from 3D objects based on distances between a reference point and the surface boundary. Assuming that boundaries of two surfaces are equivalent, a larger distance means a larger volume and thus different surfaces. However, with changes in rump shape that are not associated with the BCS and with the difficulty in accessing the rump boundary, changes in distances between a reference point and the boundary are not necessarily related to changes in volume.

While we cannot directly apply the PVHK nor the PVST we introduced so far, we are now equipped with a robust set of tools to address this problem. Namely:

- In Chapter 2, we saw that for the differences in temperature across shapes to be significant, we need to compare points equivalence points.
- In Chapter 5, we saw that locally similar shapes have a similar temperature evolution in time, regardless of the surface shape in far parts of the surface regions.

We here show how we can use these tools to introduce a new descriptor to represent rumps of thin animals. We compare the temperature between each rump and their planar projection, as we can easily establish an equivalence relation between the two. Given the time evolution of the temperature over the two, we can access how similar they are. Thin animal rumps, which are similar to their planar projection, will have small differences. Furthermore, we can focus the comparison on the upper part of the rump, without the need to further segment the rump.

8.2 Data Acquisition

While leaving the milking room, animals pass one by one on a narrow corridor. We placed a calibrated RGB-D sensor on a fixed point on top of animals' path. Exceptionally, an expert manually evaluated the animals' BCS to provide ground truth using the simplifies 3 points scale defined in [65].

While we cannot identify the rump region accurately in the different animals, we follow [65] and define the region based on the rump bone structure. In particular, we label in RGB images the tuber sacrale (hip or hook bones) and the tuber ichia (pin bones), as illustrated in Fig. 8.3(a). As seen in Fig. 8.3(b)-(d), those points correspond to features that are easily identifiable in animals of all categories.

From the camera calibration, we can map the annotations in the RGB image, I, to the depth image, D, to obtain the 3D coordinates of the left and right hip bones, $\bar{b}_{l,r}$, and pin bones $\bar{p}_{l,r}$.

When the goat is standing, bone tips approximately define a plane, as the hip and pin bones are connected rigidly. By finding the orientation of the plane defined by the four bone tips with respect to the floor, we rotate the whole surface, so the bone tips lay in the $x-y$ plane. We define the rump as all the points with a positive z. This segmentation is reproducible and consistent, albeit it may lead to the inclusion of other parts of the animal in the rump, e.g., the tail.

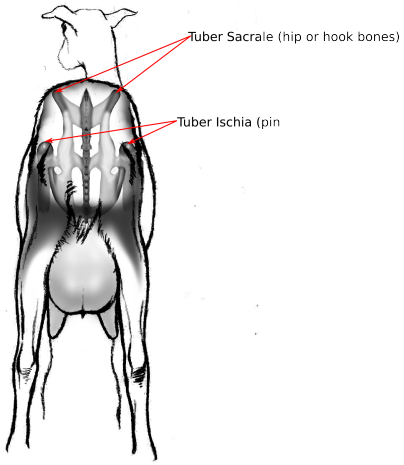
To account for changes in the animal size, we normalize both x and y coordinates of all vertices, so that the bone tips of all the animals are in the same position $\bar{h}'_{l,r}$ $\bar{p}'_{l,r}$ in the $x-y$ plane. To account for possible hip or tip bones miss-alignment, we use a projective transformation for the normalization. The resulting normalized coordinates, $X_{norm} = [\bar{x}_1^{norm} = [x_1^{norm}, y_1^{norm}, z_1], ..., \bar{x}_N^{norm}]$, maintain the same z-coordinate. The edges of the normalized surface connect the same vertices as the edges in the original one.

After segmentation and normalization, we obtain a set of rumps similar to those represented in Fig. 8.4.

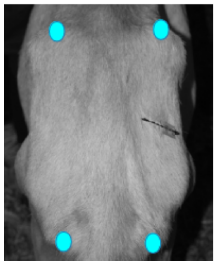
8.3 Rump Description

8.3.1 Representing variable surfaces

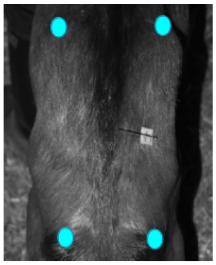
Rumps in Figure 8.4 highlight that the most distinct feature of all surfaces is that thin goats are almost flat. Figure 8.4 also illustrates the intra-class variation. In particular, it shows that goats



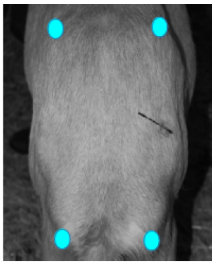
(a) Detail on the bone structure, showing that the hip and the pin bones are part of the same structure, and their distance is fixed.



(b) Very thin

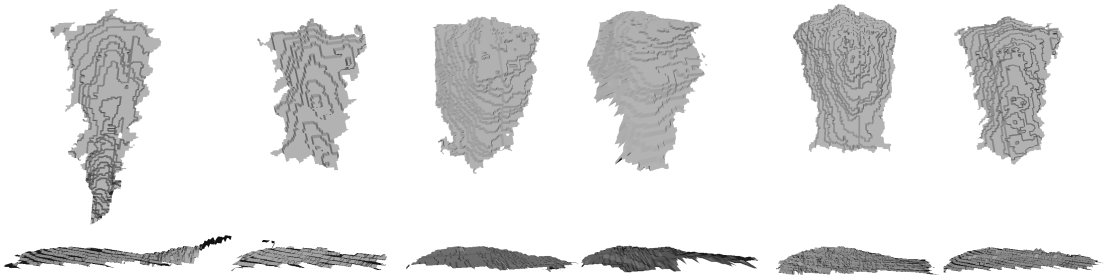


(c) Normal



(d) Very fat

Figure 8.3: Detail on the bone structure of a goat rump and examples of annotated animals.



(a) Thin (b) Thin (c) Normal (d) Normal (e) Fat (f) Fat

Figure 8.4: Example of rumps from different animals. The top image represent a view from the z-axis, while the bottom view from the x-axis.

have different features that do not arise from the BCS. For example, rump boundaries change considerably across animals, and in some animals the tail is included in our estimation of the rump region.

Adding to the natural variation in the shape, we must also account for errors in the segmentation process. Examples are: (i) uncertainty in the identification of hip and pin bones on the animal's rump; (ii) difficulty to ensure that the bone tips are on a plane; and (iii) errors errors in the map between RGB and depth images resulting from poor camera calibration.

We compare the differences in volume by extracting shape information, e.g., distances between points and areas, and compare it with the same information extracted from a planar projection, as showed in Figure 8.5. The planar projection corresponds to the same mesh, but the with z-coordinate set to zero, $X_{plane} = \left[\bar{x}_1^{plane} = [x_i^{plane}, y_i^{plane}, 0], ..., \bar{x}_N^{plane}\right]$.

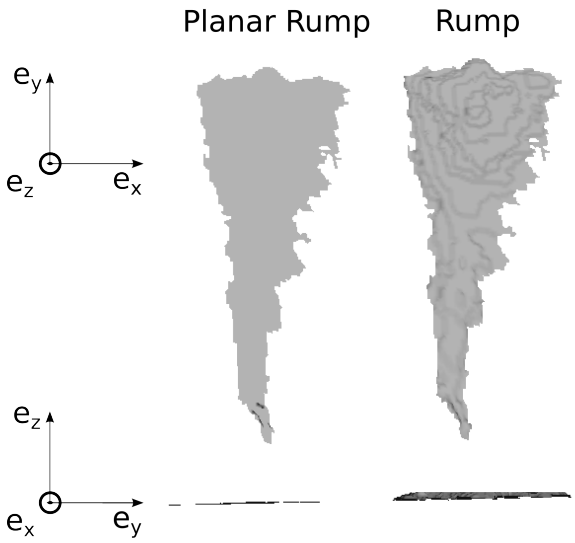


Figure 8.5: Example of a planar rump, on the left, build from the regular rump, on the right.

The comparison between the two surfaces is possible because there is a natural bijection relating the two surfaces, i.e., to each point in the rump corresponds a single point in the planar projection, and for each point in the projection corresponds a single point in the rump. We thus compare the two surfaces, by computing a geometry dependent function in each one. Again, the temperature resulting from a heat diffusion process, as it provides a natural segmentation of the interest region and is a distance proxy for surfaces retrieved by poor resolution sensors. We then access if the geometry in the two surfaces is similar or not by comparing the temperature at equivalent points in both surfaces.

8.3.2 Heat Based Rump Descriptors

We evaluate how much a rump differs from a plane by considering a heat diffusion process starting at its center and the equivalent vertex on its planar projection. Thus, the initial condition for both

the temperature in the normalized surface, $\bar{T}(0)$, and in the plane, $\bar{T}'(0)$, will be the same and different from zero only at some vertex c in the center of the rump, i.e., $[\bar{T}]_c = 1$ and $[\bar{T}]_{i \neq c} = 0$.

The vertices at the center of both rumps, with coordinates $\bar{x}_c$, and $\bar{x}_{plane,c}$, are those closest to the center of the quadrilateral defined by $\bar{h}'_{l,r}$, $\bar{p}'_{l,r}$ in both X_{norm} and X_{plane} respectively.

For each animal, given the set of edges E and the two sets of vertex coordinates X_{norm} and X_{plane}, we compute the Laplace-Beltrami operator, L_{norm} and L_{plane}. From each operator we compute the first 300 eigenvectors and eigenvalues and, given the initial condition, $\bar{T}(0)$, we propagate the temperature at both surfaces using Eq. 5.2. As there is a bijection between the two surfaces, we can compute the difference between the two temperatures, $\bar{T}_{diff}(t) = \bar{T}_{norm}(t) - \bar{T}_{plane}(t)$ at each time instant.

We evaluate the time difference at exponentially large time intervals, as changes in temperature occur faster at the first moments on propagation. In particular, we use time instants $t_k = 0.1e^{k\delta t}$, spanning from 1/700 to 1/10.

We focus on the rump upper part by accessing $\Delta\bar{T}(t)$ at a subset of vertices in the surfaces, $\mathcal{S}$. In particular, we consider those vertices that form the shortest path in the planar mesh between $\bar{x}_c$ and $\bar{h}'_l$.

Finally, we construct the descriptor, $\bar{z}$ by considering, for each time instant t_k, the maximum of $\Delta\bar{T}(t_k)$ over the subset of vertices $\mathcal{S}$, i.e.,

$$\bar{z} : [\bar{z}]_k = \max_{x \in \mathcal{S}} [T_{diff}(t_k)]_x \tag{8.1}$$

The main steps for computing HBRD are highlighted in Algorithm 8.1. The algorithm requires as input an RGB image, I, a Depth image, D, which we here assume that is already mapped into the RGB image. The algorithm further requires as input the time instants at which we compute the temperature, $\bar{t}$, and the coordinates of the left and right hip and pin bones in the normalized rump, $\bar{h}'_{l,r}$, $\bar{p}'_{l,r}$.

8.4 Results

We used the algorithm in Algorithm 8.1 to describe different animals.

Figure 8.6 shows that thinner animals converge faster to the temperature of a planar temperature. The figure represents four rumps, two very thin and two normal. The colors represent the absolute difference from the rump to the planar rump. The shortest path $\mathcal{S}$, where we evaluate the temperature, is marked in black.

Figure 8.7 shows the descriptors for the animals in Figure 8.6. There is a clear difference over the maximum of the difference between normal and thin animals. Furthermore, we note that by looking only into what happens on the top part of the rump, the animals tail has little impact on the temperature on the top part of the rump.

Finally, we show that our rump descriptor can differentiate between a dataset of 32 animals, 9

Algorithm 8.1: Computing the Heat Based Rump Descriptor (HBRD).

Input: RGB image: I; Depth image: D; Time instants: $\bar{t}$; bone tips in the normalized rump: $\bar{h}'_{l,r}$, $\bar{p}'_{l,r}$

Output: Rump descriptor, $\bar{z}_r$.

ANNOTATE HIP AND PIN BONES IN THE RGB IMAGE:

$[\bar{h}_{l,r}, \bar{p}_{l,r}] \leftarrow \text{annotate}(I)$

SEGMENT AND NORMALIZE DEPTH IMAGE:

$[X_{norm}, E] \leftarrow \text{segmentNormalize}(D, \bar{h}_{l,r}, \bar{p}_{l,r}, \bar{h}'_{l,r}, \bar{p}'_{l,r})$

$X_{plane} \leftarrow \text{project}(X_{norm})$

FIND PATH BETWEEN CENTER AND LEFT HIP BONE:

$\bar{x}_c \leftarrow \text{centroid}(\bar{h}'_l, \bar{h}'_r, \bar{p}'_l, \bar{p}'_r)$ $\mathcal{S} \leftarrow \text{shortestPath}(mesh, X_{plane}, \bar{h}'_l, \bar{x}_c)$

for $i = 1; i < size(\bar{t}); i++$ **do**

  ESTIMATE BOTH TEMPERATURES DISTRIBUTIONS, FROM EQ. 5.2:

  $\bar{T}^{\mathcal{S}}_{norm} \leftarrow \text{propagateHeat}(X_{norm}, E, \mathcal{S}, [\bar{t}]_i)$

  $\bar{T}^{\mathcal{S}}_{plane} \leftarrow \text{propagateHeat}(X_{plane}, E, \mathcal{S}, [\bar{t}]_i)$

  $\Delta T([\bar{t}]_i) = \bar{T}_{norm} - \bar{T}_{plane}$

  GET DESCRIPTOR, FROM EQ. 8.1:

  $[\bar{z}_r]_i \leftarrow \max(\Delta T([\bar{t}]_i))$

end

thin, 17 normal and 6 fat. Figure 8.8 shows the 3D-Isomap projection of the set of descriptors.

Results show that very thin animals are well clustered, i.e., that the Heat Based Rump Descriptor captures a very elusive characteristics. We further note that, by introducing a comparison surface, i.e., the rump planar projection, we naturally remove most of the dependency from changes in the rump that are not intrinsic to the class. Finally, as the result of heat diffusion is naturally comparable between surfaces, we were able to compare one rump to its planar version, and to compare differences in temperature across surfaces.

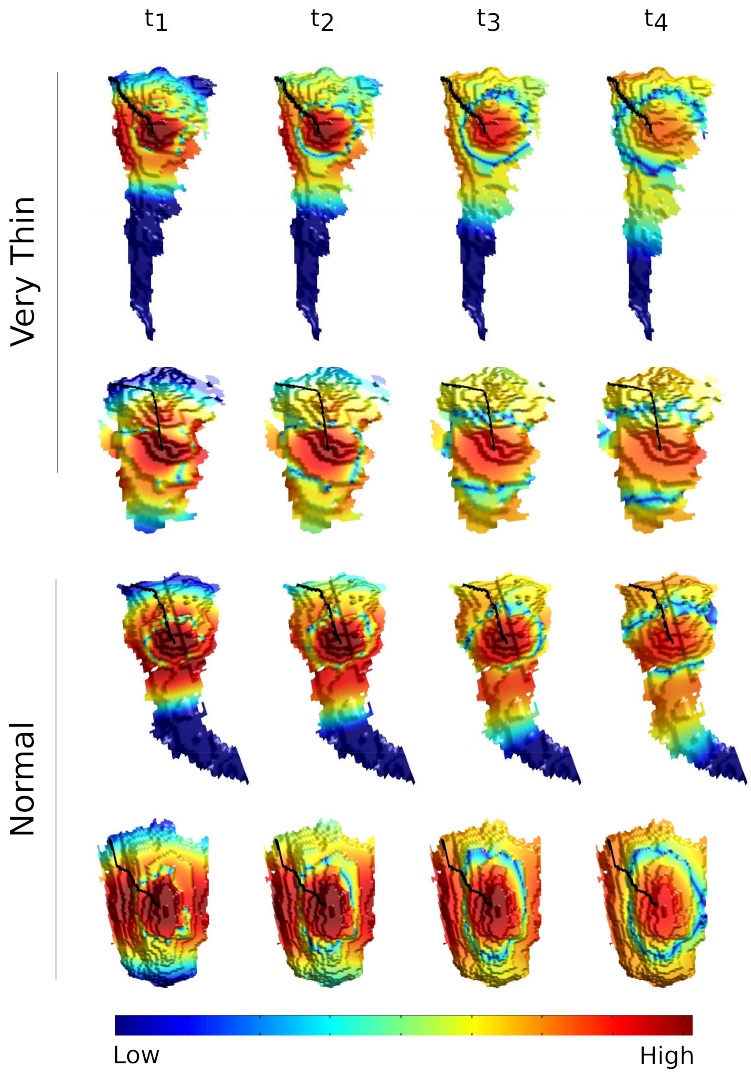


Figure 8.6: Difference over time between the temperature over the rump and the planar rump.

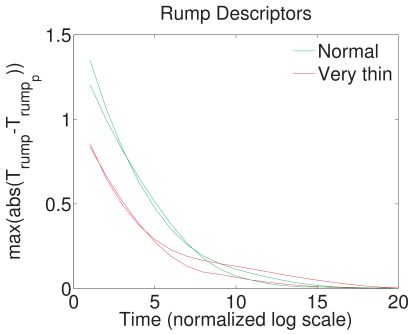


Figure 8.7: Maximum difference over time and over the path marked in Figure 8.6.

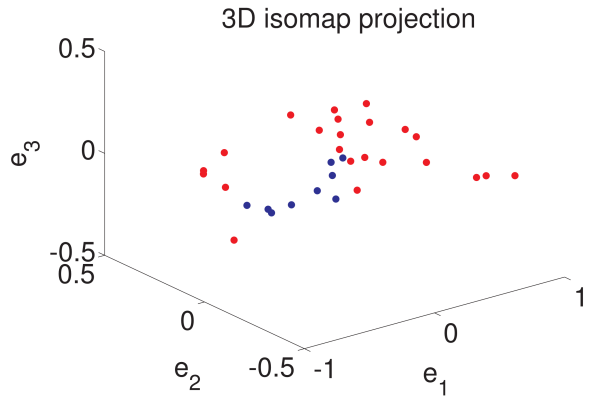


Figure 8.8: 3D Isomap projection of the rump descriptors on a dataset of 32 animals. The blue points correspond to thin animals while red correspond to normal and very fat.

8.5 Conclusion

We introduced the Heat Based Rump Descriptor (HBRD) for the identification of very thin goats in dairy farms. The identification of such animals is of utmost relevance not only by the economic implications of the decrease in the milk production associated with a low BCS, as it is in direct violation of the animal's rights.

The HBRD assesses the BCS by assessing the rump volume. To handle the large variability of animals shapes and the difficulty of defining exactly which part of the rump is relevant, HBRD uses heat diffusion to represent distances between points in two equivalent surfaces. The volume is assessed by having the surfaces differ only on the characteristic that we want to measure, i.e., the volume. The use of heat diffusion allows to soft segment the region of interest. The difference in temperature on both surfaces will be more significant in initial time instants, where only the regions close to the source have a significant impact on the temperature.

Using a dataset of 32 animals, we showed that HBRD provides a good representation for the problem, as all the very thin animals in the dataset were clustered together.

By the introduction of relevant descriptors, the work here presented is an important step towards the automation of BCS assessment in dairy goats. Future work should then focus on the automatic identification of the hip and pin bones in the RGB images.

In this chapter, we achieved two goals. The first was to show the potential of the methodologies we used in this thesis to address different problems: the classification of goats based on their body condition score. The second was to show that the intuitive interpretation of the temperature profile allows to easily adapt the descriptor to other contexts, emphasizing different parts of shapes and constructing descriptors suitable for each task.

Chapter 9

Related Work

In this chapter, we provide an overview of the related work pertaining to this thesis and how it relates to our work. This thesis provides contributions in three fields that we can enumerate by order of relevance: (i) 3D+photometric representation, which we address in Section 9.1; (ii) multiple view object recognition, which we address in Section 9.2; (iii) mesh stitching, which we address in Section 9.3.

9.1 Shape Representation

We view two ways to represent individual partial views, namely (i) as a set of local features and (ii) as a single holistic feature. We present a brief overview of both alternatives, with emphasis on the holistic features as they relate closely to PVHK.

9.1.1 Local Features

Local features are common to represent partial views since a small set of features can represent complex objects. For example, Fig. 9.1 shows the five different features required to represent the box and castle we saw in Chapter 2: three types of corners (P2, P4 and P5), an edge (P2), and a plane (P1).

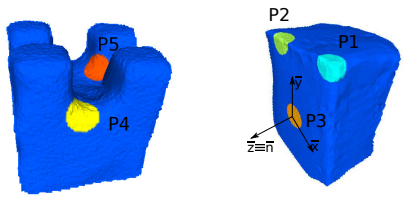


Figure 9.1: Example of shapes that can be described using only 5 local features.

Since local representations describe only a small portion of an object, recognition algorithms either solve first a registration problem or combine features into bags of features, similar to bags of

words. Consequently, descriptors need to be invariant to changes in pose. Several representations achieve invariance by describing the feature on a tangent space to the object surface at each point, since this space is not only invariant to changes in a pose as is easy to reproduce. Examples of such representations are the Fast Point Feature Histogram (FPFH) [53], Signatures of Histograms of OrienTations(SHOT) [63], Local Surface Patches(LSP) [17], Spin Images (SI) [31], and Intrinsic Shape Signatures (ISS) [71]

However, methods for estimating the tangent space are sensitive to noise because they rely on normal estimation. As we illustrate in Figure 9.2, this negatively reflects on the descriptors. In the figure, we show the variance of different representations as the distance, d, between object and sensor increases, increasing the noise. We estimate the variance by computing the descriptor of the same point over 40 point clouds generated for each value of d. As descriptors have high dimensionality, we represent the variance as ratio between the maximum eigenvalue of the covariant matrix and the mean descriptor. The point used for comparison is $P1$ from Figure 9.1 and the descriptors correspond to SHOT, FPFH, and a holistic partial view representation, View Point Histogram, that we include for comparison purposes.

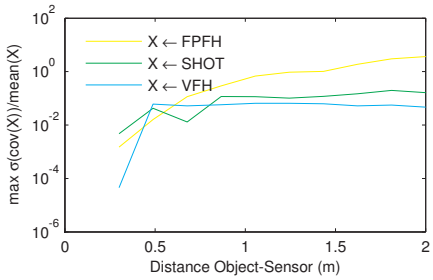


Figure 9.2: Noise impact on point like descriptors.

9.1.2 Holistic Partial View Features

By describing a larger surface, holistic partial view representations are more stable to noise, even when defined on a tangent space. E.g., the Viewpoint Feature Histogram (VFH) [54] is an extension of FPFH to the whole partial view, but has a lower variance, as shown in Figure 9.2.

To altogether avoid tangent space estimation, other representations build upon distances between points on the object surface. E.g., representations for complete objects can be build from the distribution of Euclidean distances between points [47]. Extensions to account also for topological information, e.g., [29], are constructed by classifying whether lines connecting pairs of points lay inside the object surface or not. The later was also extended to partial views as Ensemble of Shape Functions, (ESF) [70].

The discriminative power resulting from topological the information comes at the cost of increased sensitivity to holes in the surface due to sensor noise. A more robust approach relies on the use of diffusive distances [42] as a noise resilient surrogate to shortest path distances on object

surface.

Diffusive processes can describe local features, such as the Heat Kernel Signature (HKS) [61] and the Scale Invariant Heat Kernel Signature (SI-HKS) [14]. HKS is a highly robust local descriptor that contains large scale information. HKS represents a point with the temperature evolution after placing a heat pulse source on that point. The evolution depends on how fast the temperature propagates to the neighborhood, which in turn depends on the object geometry. While both descriptors, HKS and SI-HKS, perform well on complete 3D shapes, the same point on an object surface may have different descriptors depending on the partial view. Accordingly, matching features across partial views using HKS or SI-HKS is not feasible.

9.1.3 Shape and Appearance

To jointly combine the appearance and shape, some approaches, e.g., [7, 36], resort to extending ad-hoc the descriptor dimension to include some color/texture descriptor on the extra dimensions. However, the joint descriptors do not effectively associate appearance features with positions in the object.

On the other hand, the photometric heat kernel [34], directly associates appearance to 3D coordinates by changing the space where the object is defined. I.e., each point in the surface lays in a 6D space with physical coordinates plus RGB values. The formalism used for diffusive process extends naturally to this new space, however, it takes into account only color gradients. Color gradients may improve segmentation as intended by authors, but hinders recognition as a white wall becomes equivalent to a blue wall.

More recently, an approach that extends the photometric heat kernel to different types of texture features were introduced (textMesh - our reference[1]), [68]. However, this approach does not rely on diffusion, but on Local Binary Pattern, which is closely related to a binary version of the Laplace-Beltrami operator. Also, a new method was proposed to introduce photometric information as scalars over a mesh (w-HKS) [1]. In particular, the heat diffusion in a weighted manifold was also used to represent non-rigid shapes, [1], however, it was used in the computation of local features and ah-doc holistic of complete shapes on representations based on bag-of-features.

Finally, information of different sources can be fused by considering covariance matrices (cov-RGBD - our reference)[62], over vectors describing different types of features, e.g., distances between points, volumes, HI-HKS, or color values and other texture features.

9.1.4 Observer Position

The potential for 3D poses estimation through partial view descriptors as been the focus of different representations, [54, 69]. The use of partial view descriptors as the advantage that it does not require the registration of different point clouds. Besides the use of normal estimation in the Viewpoint Feature Histogram[54], others have introduced and approach for Learning Descriptors

[1]Authors do not use a clear name for their algorithm. This name is our responsibility alone.

for Object Recognition and 3D Pose Estimation (learn3DPose - our reference)[69]. Learn3DPose uses Convolutional Neural Networks that allow the embedding of descriptors on a manifold. The position in the manifold encodes both information on both 3D pose and object class, so that distances in the manifold are related to changes in the object pose.

9.1.5 Part Aware Representation

The identification of object parts, by their semantic value, is a well studied topic in computer vision in both 2D and 3D. The field is extensive and very active throughout decades. A very relevant contribution in terms of 2D images is the work developed by Felzenszwalb, P. F. et al., for Object Detection with Discriminatively Trained Part-Based Models, [23], however, it is the result of learning on large collections of 2D images, and not a geometry based representation in which we focus next.

Most common approaches, e.g., [40, 50, 51, 59, 66], focus on the segmentation of shapes in polygons or skeletons. Approaches can be separated in those that try to model the shape of the object [40, 50], e.g., by finding regions of concavity in the shape, or those that resort to methods similar to spectral clustering and also related to the eigenvectors and eigenvalues of the Laplace-Beltrami operator, e.g., the Hierarchical Shape Segmentation and Registration via Topological Features of Laplace-Beltrami Eigenfunctions [51]. In common, and as far as we are aware, all the approaches focus on the segmentation/breaking of the object, and do not account for smooth transitions between the parts.

The use of part-aware metrics, instead of object segmentation has also been proposed by Liu, R. et al.[41] (PartAware - our reference), for the purpose of improving matching between points across two objects represented as watertight CAD models. The definition of part for the construction of such metric could not be extended to the context or partial views.

9.1.6 How our Work Fits

We represent partial views by a set of distances between boundaries and a reference point. Assuming an equivalence between boundaries and the reference point across objects:

- distances uniquely define the partial view,
- changes in distances can be easily interpreted in terms of changes in the shape.

Furthermore, by using the boundary to represent partial views, we obtain a signature that can be easily compared across shapes, without the need of registration.

We use a heat kernel approach to providing a noise resilient representation of distances that has already proved to be easily expandable to include color. The heat diffusion over a graph is a well-studied problem and thus allowed us to improve further our representation with the introduction of new part aware metrics.

Finally, our proposed representation can be made either pose dependent or independent. The view-dependent naturally lead to the distribution of descriptors over a manifold and allowed its use in object identification and disambiguation from multiple views.

By representing complete partial views, we need a large number of partial views per object, and thus there is the potential of our approach not to scale so well for vast datasets.

Descriptor	Handles partial Views	Scales well to large datasets	Extends to texture	Is Robust to noise	Depends on pose
PVHK	yes	no	yes	yes	yes
Local Features					
FPFH	yes	yes	no	no	no
SHOT	yes	yes	no	no	no
SI	no	yes	no	no	no
Holistic Partial View Features					
HKS-SI	no	yes	yes	yes	no
ESF	yes	no	no	yes	no
VFH	yes	no	no	no	yes
covRGBD	yes	no	yes	-[2]	no
learn3DPose	yes	no	no	-	yes

9.2 Multiple View Multiple Hypotheses Object Identification

There are several approaches for merging information from multiple consecutive observations. We here highlight those that also use 3D partial views as input data or that sequentially improve the object class estimation using a Bayesian setting and a Bayesian setting.

The information from consecutive 3D partial views can be used to construct complete 3D models, e.g., with the KinectFusion algorithm, [30] or with any method described in the following section.

However, constructing a model does not solve the classification problem. Even a complete 3D object surface would still need to be represented, e.g., as bag of HKS-SI features, and go through a classifier. Furthermore, the use of KinectFusion would require an observer to see fully the object before attempting to recognize it. Our algorithm can provide at each moment an estimative of the object class.

Multiple-hypothesis approaches have also been extensively used for object tracking in 2D color videos, e.g., in the Boosted Particle Filter algorithm (BPF) [46], or localization of real robots actuating on the environment [19]. However, in both applications, hypotheses do not include the object class and the localization or tracking algorithms assume that an independent algorithm provides the object class.

Notwithstanding, some tracking algorithms and localization algorithms, such as Multiple Cue 3D object Recognition (MC3DOR) [45], Detection and Tracking (DT) [20] and Global Localization by Soft Object Recognition from 3D Partial Views (GL)[52], also using the PVHK, have been

extended to include object classification. However, in none of above examples the similarity between partial views of multiple objects are used.

The current work differs significantly from the previous examples in the sense that we use an a-priori known map between the view angle and appearance to improve our recognition, in a manner similar to what can be seen in Active Monte Carlo Recognition (AMCR) [28]. The latter introduces an algorithm for object recognition based on multiple-hypothesis, as well as the notion that when dealing with sequential class estimation there are two spaces: one associated with the object appearance and another associated with the observer dynamics. The authors also propose a mapping between the two, which reflects the notion of similarity between different state-vectors based on the similarity between objects. However, AMCR uses the mapping to establish a relation between two sets of particles, one that moves in the object appearance space, and the other that moves on the observers space.

There is also a rich literature on hypotheses testing for active object recognition, e.g., [3] and references therein. In the active context, object recognition is also formulated in a Bayesian framework, where the belief on a set of hypotheses is propagated over a sequence of actions. However, there is not a sampling approach as we here present. Instead there is an hypothesis associated with each point in the search space. Our current work is complementary to these in the sense that it provides a way to handle large search spaces.

9.2.1 How our Work fits

We use a Monte Carlo Sequential-Resampling Filter to estimate online an object class sequentially collecting multiple partial views. The use of particle filters for improving an estimative from multiple observations has been used in different scenarios, including object recognition. However, objects and positions are usually estimated using dedicated sets of observations, while we here leverage on the PVHK sensitivity to the observer viewing angle to determine the orientation between observer and object through a sequence of movements.

To handle the coupling between position and object class we rely on the concept of two spaces connected by an off-line mapping introduced in [28]. However, we only require a set of particles on the observers space, as we use the mapping to infer distances from the appearance space. Furthermore, we propose more complex appearance models and similarities than those used in [28].

Approach	Estimates object class and pose	Uses similarity	Uses 3D partial views
PVS-Resampling	yes	yes	yes
MC3DOR	yes	no	yes
DT	yes	no	yes
AMCR	yes	yes	no
BPF	no	no	no
MCOR	no	no	no
GL	yes	no	yes

9.3 Mesh Stitching

The use of range images for 3D object modeling motivates the use of mesh stitching to construct complete models. Due to their planar topology, range images induce an intrinsic mesh in point clouds, but do not represent the whole object. Thus, to recover the complete object surface, several meshes from range images can be stitched together, instead of using point cloud filtering approaches, such as Poisson Reconstruction (PR) [32], Moving Least Squares (MLS) [37], Algebraic Point Sets (APS) [26] or KinectFusion [30], or using approximations to the convex-hull, such as Alpha Shapes (AS) [22] or ball pivoting (AB) [4].

In terms of applications, we note that using the original mesh and vertices instead of using pos-processing approaches introduces several advantages, e.g., adding color to the models is immediate. As such, we here focus on other works that preserve the original mesh.

In [64], authors present a three step algorithm for stitching range images that make use of the overlap between images to both align and stitch them. The algorithm first step is to align meshes by means of an Iterative Closest Point (ICP) algorithm, [5]. The second step removes overlapping regions between two adjacent meshes, by deleting triangles. This step leaves only the triangles that do not overlap or that overlap only partially. The final step stitches meshes by the points where the partially overlapped triangles intersect. The stitching procedure adds vertices at the intersection and new triangles are built on top of the original ones. When there is no overlap, meshes cannot be aligned using ICP and the stitching cannot be built on top of existing triangles.

More recently, different authors, e.g. in [43] and [49], used the technique described in [64] for mesh stitching with the purpose of filling holes in a model. In both algorithms an initial step for mesh alignment was required. However, while [43] used parts of the same object from different meshes to fill in the holes, [49] used parts of other objects. Because the objects are different, instead of aligning the meshes with an ICP type of algorithm, [49] resorts to non-rigid deformations. Both algorithms used the stitching algorithm proposed in [64] to combine different meshes.

The Progressive Gap Closing (PGC) [8] and Integration of Sets of Range Views (ISRV) [60] focus on the stitching part, and assume that meshes are already registered. The former algorithm stitches by introducing new edges and minimizes their length by creating and deleting vertices in the boundaries. In our algorithm, JASNOM, we also focus on minimizing the edge length, but we do it for the purpose of finding a rigid transformation that aligns the two meshes. The algorithm in [60] uses a Delaunay triangulation on a re-projection of non-overlapping meshes. However, the complete algorithm assumes that there is a very fine alignment between meshes, which JASNOM does not require.

Finally, we call the attention to recent work that, as JASNOM, aims at simplifying the acquisition setup. Namely, the work of Dou, M. et al. for 3D Scanning Deformable Objects with a Single RGBD Sensor (Scan1)[21]. Scan1 reconstructs deformable objects by combining multiple partial views without rigidity constrains. Again by filtering across multiple partial views, they arrive at a

very good 3D model, but they have lost the RGB information of each point in the surface.

9.3.1 How our work fits

JASNOM adds to the capabilities of the previous algorithms, the possibility of aligning meshes with no overlap and connecting meshes without resorting to existing triangles to ensure manifoldness. Furthermore it does not require a-priori alignment and, as it preserves the original 3D mesh, allows for the creation of color 3D meshes without the need to further register color into the mesh.

Approach	Independent of previous alignment	Preserves original RGB information	Returns a manifold mesh
JASNOM	yes	yes	yes
Poisson Reconstruction	no	no	yes
MLS	no	no	yes
APS	no	no	yes
KinectFusion	no	no	yes
Scan1	no	no	yes
ICP	no	yes	no

Chapter 10

Conclusions

We review the major scientific contributions of this thesis before discussing promising directions for future research.

10.1 Contributions

Heat diffusion for the representation of partial views

We introduce a heat diffusion approach for holistically represent surfaces, namely partial views of objects. Using relevant characteristics of the heat diffusion, we developed an approach for partial view representation that relies on the distance between a reference point , where we place a heat source, and the boundary points where we access the temperature. By representing the partial view by the boundary points, we allow for a natural mapping between partial views, not requiring any registration between shapes, while preserving geometric information on the descriptor. Furthermore, we introduce a method for encoding geometric distributions of other relevant information by changing the diffusion rate point by point. Currently, existing heat diffusion methods only represented points on a surface and, as diffusion processes depend on the complete partial view, the descriptor of a single point would change with changes in the partial view.

Partial views descriptors

We contribute to three novel approaches to representing partial views.

- The Partial View Heat Kernel (PVHK), which captures distance by the temperature at the boundary at a time instant that depends on the partial view geometry.
- The Partial View Stochastic Time (PVST), which captures shape by the time it takes the boundary points to reach a fixed temperature.
- The Color Partial View Heat Kernel (C-PVHK), which, by associating color and texture to the diffusion rate, captures both distance and photometric information by the temperature at the boundary.

An analysis of regular vs. complex objects

We introduced the concept of complex objects as those with loosely connected parts. The concept followed naturally on properties of heat diffusion, which we analyzed in detail. From the analysis resulted:

- the introduction of a soft classification of points in the partial view as parts or not parts;
- the introduction of a novel metric for objects based on the soft classification of parts;
- the introduction of a new stopping time for the representation of complex objects with the PVHK;
- the introduction of PVST.

Examples on how to adapt the proposed descriptors to different applications

We explored our liberty to choose the source position in each partial view to tailors the descriptor to different applications. We created observer dependent descriptors by associating the source to the relative position between the object and the observer. Such descriptors are useful in applications where we want to combine multiple observations from multiple viewing angles, e.g. for disambiguating similar objects or for localization tasks. We created observer independent descriptors, by conveniently placing sources in the object library partial views so that the descriptors in the library were more discriminative for a given set of objects and partial views. We further showed how to extended the PVHK to a new and challenging context: the assessment of very thin animals in a goat farm.

Multiple view multiple hypotheses object recognition algorithm

We introduced a multiple view multiple hypotheses object recognition algorithm, for the purpose of disambiguating between similar objects and to validate recognition results. We introduced a similarity based resampling approach to reducing the number of hypotheses required to ensure a good coverage of the set of possible objects and viewing angles.

An algorithm for the creation of compact libraries

We introduced a source placement algorithm that takes into account the set of objects in the library and their partial views, to create compact libraries. The sources are placed so that the descriptors of different objects are as far away as possible from one another, and close to descriptors of partial views of the same object, especially to those of similar view angles.

Analysis of the discriminative nature of introduced descriptors in different datasets and applicatio

We demonstrated the effectiveness of the introduced descriptors in several contexts.

- We classified an object library of small regular objects, with the PVHK and a using nearest neighbors approach. The PVHK achieved an average recognition rate of 95%,

with most of the confusion occurring between objects that are clearly identical from some view angles.

- We compared the PVHK with state of the art descriptors in a dataset of 4 objects. The PVHK not only performed on par in terms of accuracy, but also had the advantage that it changed smoothly with the viewing angle, allowing for observed position dependent applications.
- We classified several regular and same class objects using C-PVHK and showed that C-PVHK can effectively index color to geometry.
- We classified partial views of an object library of 54 chairs using both PVST and the FT-PVHK with part-metrics. Both approaches can distinguish between all the 54 chairs with an average accuracy of 85% using just eight partial views per object.
- We represented several non-rigid shapes using PVHK and showed that, as heat diffusion is invariant the isometric deformations, PVHK does not change considerably between changes in pose. We also showed that C-PVHK differentiates different humans, with similar attire, while they walk and go through different changes in their shape.
- We showed that the we can disambiguate between similar shapes using multiple observations from different viewing angles, and that our multiple view multiple hypotheses approaches, which relied on similarity to recognize objects can differentiate between partial views of multiple objects.

JASNOM for the construction of complete 3D meshes

We contributed an algorithm for the Joint Alignment and Stitching of Non-Overlapping Meshes (JASNOM), for the creation of complete 3D meshes representing object surfaces constructed from 2 non-overlapping but complementary meshes, with not previous alignment. We showed how it could be used to reconstruct 3D meshes of a human from 2 meshes acquired simultaneously from opposite sides of the human RGB-D sensors. We also showed how to reconstruct regular objects using a 2-step approach.

10.2 Future Work

Color mapping

Currently, C-PVHK encodes photometric information by a means of a scalar function, the diffusion rate, and we considered only very simple functions, such as the hue of each pixel. How could we learn an optimal mapping that would improve recognition over a set of partial views? Could such mapping receive as input other information, such as SIFT features? Are we constrained by scalar function, or are there other approaches to introducing multivariate functions?

Different graphs

Currently we use the PVHK and PVST to represent partial view meshes, which correspond to a planar graph. It would be interesting to see how any of the above descriptors could handle other sorts of graphs. For example, how could we describe a graph representing a building, with nodes centered on doors, windows or other architectonic features of relevance?

Generating initial hypotheses

We introduced a resampling approach that handles similarity between objects for the purpose of disambiguating between object. However, similar approaches could be used for the initial hypotheses generation. How can we further reduce the number of particles by using good criteria on the initial sampling approach?

Modeling sequences of observations

We introduced a Bayesian approach for combining multiple observations for the same object, which was based on a map between annotated viewing angles and previously observed descriptors. However, it would be interesting to model the set of possible descriptors, so that we could have guesses to viewing angles not present in the object library. Could we use manifold learning to model the set of possible observations? And could we use such manifolds to recognize an object from multiple observations without the use of a Bayesian approach?

Recognizing very fat goats

We used the very thin goats as an example of the versatility of the methodologies we here developed. Can we use similar approaches to recognizing very fat animals as well.

10.3 Concluding Remarks

This thesis contributes with a bottom-up approach to 3D partial views representation. We have introduced a methodology to represent distances within partial views. We have showed its properties and explored its behavior in different types of objects. Equipped with the understanding on the properties, we have introduced adaptations on the representation and showed how the representation can be adapted to answer different types of problems.

Appendix A

Impact of sensor noise on the Laplace-Beltrami operator

When estimating the impact of the sensor noise in the vertices position, we follow the noise model introduced in [33]. We thus assume that the depth information retrieved by the sensor is perturbated by Gaussian noise, i.e., $z_i = z_i + \varepsilon z_i^2$, $\varepsilon_i \sim \mathcal{N}(0, \tau)$, with $\tau = 1.42 \times 10^{-3} m^{-1}$.

This error on the depth impacts also the x and y coordinates, as those are computed from z the focal length, f and the distance to the center of the image. Thus, the coordinates of vertex v_i, whose coordinates would be $\bar{x}_{0,i} = (x_0, y_0, z_0)$ in the absence of noise, become $\bar{x} \simeq (x_0, y_0, z_0)\,(1 + z_0\varepsilon)$.

The square of the distance between two vertices becomes $d_{i,j}^2 = \|\bar{x}_j - \bar{x}_i\| = \rho_{0,i}^2 z_0^2 (\mathbf{e}_i - \mathbf{e}_j).^2 + d_{0,i,j}(1 + z^2\mathbf{e}_j^2 + 2z\mathbf{e}_j) + 2\tilde{\rho}_{0,i,j} z(\mathbf{e}_i - \mathbf{e}_j)(1 + z\mathbf{e}_j)$, where $\rho_{0,i} = \|\bar{x}_{0,i}\|$ and $\tilde{\rho} = \bar{x}_i \cdot (\bar{x}_j - \bar{x}_i)$.

The Laplace-Beltrami depends on the inverse of the square of the distance, which in second order expantion on $\mathbf{e}$ results in:

$$\frac{1}{d_{i,j}^2} = \frac{1}{d_{0,i,j}^2}\left(1 - 2z_i\mathbf{e}_j + 3z_i^2\mathbf{e}_j^2 - \frac{1}{d_{0,i,j}^2}\left[\rho_i^2 z_i^2(\mathbf{e}_i - \mathbf{e}_j)^2 - 2\tilde{\rho} z_i(\mathbf{e}_i - \mathbf{e}_j)(1 - 3z_i\mathbf{e}_j)\right] + \frac{4}{d_{0,i,j}^4}\tilde{\rho}^2 z_i^2(\mathbf{e}_i - \mathbf{e}_j)^2\right) \tag{A.1}$$

On average, this means that

$$\left\langle {d_{i,j}}^{-2} \right\rangle = {d_{0,i,j}}^{-2}\left(1 + 3z_i^2\tau^2 - {d_{0,i,j}}^{-2}\left(\rho_i^2 z_i^2 2\tau^2 + 6\tilde{\rho} z_j \tau^2\right) + 4{d_{0,i,j}}^{-4}\tilde{\rho}^2 z_i^2 2\tau^2\right) \tag{A.2}$$

$$\simeq {d_{0,i,j}}^{-2}\left(1 - {d_{0,i,j}}^{-2}\rho_i z_j^2 \tau^2 + {d_{0,i,j}}^{-4}\tilde{\rho}^2 z_i^2 2\tau^2\right) \tag{A.3}$$

For typical values of the sensor distance, $z = 1$, and resolution, a focal length of 580 for a 460×680 image, the expected value is of ther order of : $\left\langle {d_{i,j}}^{-2} \right\rangle = {d_{0,i,j}}^{-2}\left(1 + 5 \times 10^{-3}\right)$. Thus the trace of the Laplace-Beltrami operator will be affected by something of the order of

$5 \times 10^{-3} Tr(L_0)$, where $Tr(L_0)$ is the trace of the Laplace-Beltrami operator in the absence of noise and corresponds to the sum over all $d_{0,i,j}^{-2}$ in the object surface, i.e., is proportional to $\left\langle d_{0,i,j}^{-2} \right\rangle$.

Appendix B

Impact of perturbations on the Laplace-Beltrami to the temperature

Given a Laplace-Beltrami operator L^1 and a perturbation to that operator L^η, where $\|L^\eta\| \ll \|L^1\|$ we can approximate the eigenvalues and eigenvectors of the operator $L^2 = L^1 + L^\eta$ using perturbation theory.

Provided that L^1 does not have eigenvalues with geometric multiplicity greater than 1 and using firts order expantion on the perturbations, we can write:

$$\lambda_i^2 \approx \lambda_i^1 + \lambda_i^\eta, \quad \lambda_i^\eta = \bar{\phi}_i^{1,T} L^\eta \bar{\phi}_i^1 \tag{B.1}$$

$$\bar{\phi}_i^2 \approx \bar{\phi}_i^1 + \bar{\phi}_i^\eta, \quad \bar{\phi}_i^\eta = \sum_{j \neq i} \frac{\bar{\phi}_i^{1,T} L^\eta (\sqrt{2}\tau) \bar{\phi}_j^1}{\lambda_i^1 - \lambda_j^1} \bar{\phi}_j^1. \tag{B.2}$$

We note that $\bar{\phi}_1 = \bar{0}$ as all the Laplace-Beltrami operators have $\lambda_1 = 0$ and $\bar{\phi}_1 = \bar{1}$.

The temperature associated with the operator L^2 can be estimated as:
$\bar{T}^2(t_2) = \bar{T}^1 + \bar{T}^\eta(t_2) \approx (\Phi^1 + \Phi^\eta) \exp\{-\Lambda^1 t - \Lambda^\eta t\} (\phi_s^1 + \phi_s^\eta)$, where $t_2 = (\lambda_2^1 + \lambda_2^\eta)^{-1}$. Retaining again only first order terms yields:

$$\begin{aligned}\bar{T}^\eta(t_2) \approx & \Phi^\eta \exp\{-\Lambda^1 t_2\} \Phi^{1,T} \bar{T}(0) + \Phi^1 \exp\{-\Lambda^1 t_2\} \Phi^{\eta,T} \bar{T}(0) - \\ & \Phi^1 \exp\{-\Lambda^1 t_2\} (\Lambda^\eta t_2) \Phi^{1,T} \bar{T}(0).\end{aligned} \tag{B.3}$$

Appendix C

Distance to equilibrium, upper and lower bounds

C.1 Proof of Eq. 5.4

Eq. 5.3 is a particular case of Theorem 20.6 from[39], and we here present its proof. We first introduce a bound for the norm of the temperature vector $\bar{T}(t)$ for each time instant t and regardless of the source position. And then, we show the bound for each vector entry $[\bar{T}(t)]_i$. A more general proof for continuous diffusion processes in both directed and undirected graphs can be found in [39].

Let $\bar{T}(0)$ be any initial temperature distribution over an undirected graph with a Laplacian L. The temperature at each time instant is given by $\bar{T}(t) = \exp\{-Lt\}\bar{T}(0)$, and when $t \to +\infty$, the temperature reaches equilibrium at $T_{eq}\bar{1}$ with $T_{eq} = \|\bar{T}(0)\|/N$.

Let $u(t) = \|e^{-Lt}(\bar{T}(0) - T_{eq}\bar{1})\|_2^2$ represent the norm of the difference between the temperature at each time instant t and the equilibrium. The norm changes with time as:

$$u'(t) = -2(\bar{T}(0) - \bar{1}T_{eq})^T \exp\{-Lt\}L\exp\{-Lt\}(\bar{T}(0) - \bar{1}T_{eq}). \tag{C.1}$$

Reminding the bound on λ_2 for any function $\bar{f}$ with zero mean:

$$\lambda_2 \leq \frac{\bar{f}^T \exp\{-Lt\}L\exp\{-Lt\}\bar{f}}{\|\exp\{-Lt\}\bar{f}\|_2}, \tag{C.2}$$

we introduce an upper bound for $u'(t)$: $u'(t) \leq -2\lambda_2 u(t)$.

Given the initial condition $u(0) = \|\bar{T}(0) - T_{eq}\|_2^2$, we define an upper bound on $u(t)$ based on λ_2: $u(t) = \|\exp\{-Lt\}\bar{f}\|_2^2 \leq \|\bar{f}\|^2 e^{-2\lambda_2 t}$. Furthermore, as $\exp\{-Lt\}\bar{1} = \bar{1}$, we bound the norm of

distances to equilibrium temperature by:

$$\| \exp\{-Lt\}\bar{T}(0) - \bar{1}\|_2^2 \leq \|\bar{T}(0) - \bar{1}\|_2^2 \exp\{-2\lambda_2 t\} \tag{C.3}$$

$$\tag{C.4}$$

Assuming an initial temperature $\bar{T}(0) = \bar{\delta}_i$ defined as $[\bar{\delta}_i]_j = N$ for $j = 1$ and 0 otherwise, we obtain the bound on the norm of the temperature distribution:

$$\| \exp\{-Lt\}\bar{\delta}_i - \bar{1}\|_2^2 \leq N^2 \exp\{-2\lambda_2 t\}. \tag{C.5}$$

$$\tag{C.6}$$

From the bound on the norm, we obtain a bound for the temperature at each vertex, when the source is placed at vertex j, $\bar{T}(t) = e^{-Lt}\bar{\delta}_j$:

$$[\bar{T}(t)]_i = [\exp\{-Lt\}\bar{\delta}_j]_i = \bar{\delta}_i^T \exp\{-Lt\}\bar{\delta}_j/N \tag{C.7}$$

$$\left|[\bar{T}(t)]_i - 1\right| = \left|\bar{\delta}_i^T \exp\{-Lt\}\bar{\delta}_j/N - 1\right| \tag{C.8}$$

$$= \left|\bar{\delta}_i^T e^{-Lt}\bar{\delta}_j - N\right|/N \tag{C.9}$$

$$= \left|(\bar{\delta}_i - \bar{1})^T \exp\{-L/2t\} \exp\{-L/2t\}(\bar{\delta}_j - \bar{1})\right|/N \tag{C.10}$$

$$\leq \| \exp\{-Lt/2\}\{\bar{1} - \bar{\delta}_i\}\|/N \tag{C.11}$$

$$\leq N \exp\{-\lambda_2 t\} \tag{C.12}$$

C.2 Proof of Eq. 5.3

We here present the proof for the lower bound from Eq. 5.4, which is a particular case of the Lemma 20.11 from [39]. A more general proof for continuous diffusion processes in both directed and undirected graphs can be found in [39].

Let L be the Laplacian of an undirected graph, with eigenvalues λ_i $i = 1, N$, and respective eigenvectors $\bar{\phi}_i$, so that $\lambda_1 = 0$ and $\bar{\phi}_1 = \bar{1}/N$. Thus, $\exp\{-Lt\}\bar{\phi}_i = \left(\exp\{-Lt\} - \bar{1}\bar{1}^T/N\right)\bar{\phi}_i^T$ and we have the identity:

$$\bar{\delta}_j^T \exp\{-Lt\}\bar{\phi}_i = \exp\{-t\lambda_i\}\bar{\delta}_j^T\bar{\phi}_i \tag{C.13}$$

$$\left|[\exp\{-Lt\}\bar{\phi}_i]_j\right| = \left|\bar{\phi}_i\right|_j \exp\{-t\lambda_i\} \tag{C.14}$$

$$= \left|[(\exp\{-Lt\} - \bar{1}\bar{1}^T/N)\bar{\phi}_i^T]_j\right|. \tag{C.15}$$

It follows that

$$\exp\{-t\lambda_i\}\|[\bar{\phi}_i]_j\| \leq \max_j \left|\bar{\delta}_j^T(e^{-Lt} - \bar{1}\bar{1}^T/N)\right| /N \left|\bar{\phi}_i^T\right|_\infty \tag{C.16}$$

$$\leq \max_j \left|e^{-Lt}\bar{\delta}_j - \bar{1})\right| \left|\bar{\phi}_i^T\right|_\infty /N \tag{C.17}$$

$$\tag{C.18}$$

Choosing $i = 2$, as $\exp\{-\lambda_2 t\} \geq \exp\{-\lambda_{i>2} t\}$, and j so that $\| \, [bar\phi_i]_j \, \| = \left|\bar{\phi}_i^T\right|_\infty$, we arrive at the bound in Eq. 5.4: $\exp\{-t\lambda_i\}N \leq \max_j \left|e^{-Lt}\bar{\delta}_j - \bar{1})\right|$

Bibliography

[1] M. Abdelrahman, A. Farag, D. Swanson, and M.T. El-Melegy. Heat diffusion over weighted manifolds: A new descriptor for textured 3d non-rigid shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[2] M.S. Arulampalam, S. Gordon N. Maskell, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *In IEEE Trans. on Sig. Processing*, 2002.

[3] N. Atanasov, B. Sankaran, J. Le Ny, T. Koletschka, G. J. Pappas, and K. Daniilidis. Hypothesis testing framework for active object detection. In *ICRA*, 2013.

[4] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[5] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, February 1992.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[7] M. Blum, J. T. Springenberg, J. Wülfing, and R. Riedmiller. A learned feature descriptor for object recognition in rgb-d data. In *ICRA*, 2012.

[8] P. Borodin, M. Novotni, and R. Klein. Progressive gap closing for mesh repairing. In J. Vince and R. Earnshaw, editors, *Advances in Modelling, Animation and Rendering*, pages 201–213. Springer Verlag, July 2002.

[9] S. Brandão, J.P. Costeira, and Veloso M. Multi-part complex objects heat-based 3d partial view descriptors. In *Submitted Computer Vision and Image Understanding*, 2015.

[10] S. Brandão, J.P. Costeira, and Veloso M.V. The partial view heat kernel descriptor for 3d object representation. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, 2014.

[11] S. Brandão, J.P. Costeira, and M. Veloso. Effortless scanning of 3d object models by boundary aligning and stitching. In *Proceedings of the 9th International Conference on Computer Vision Theory and Applications*, 2014.

[12] S. Brandão, M. Veloso, and J.P. Costeira. Multiple hypotheses for object class disambiguation from multiple observations. In *2nd International Conference on 3D Vision, 3DV 2014, Tokyo, Japan, December 8-11, 2014*, 2014.

[13] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein, and R. Kimmel. Partial similarity of objects, or how to compare a centaur to a horse. *Int. J. Comput. Vision*, 84(2):163–183, August 2009.

[14] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *CVPR*, pages 1704–1711, 2010.

[15] C. G. Broyden. The convergence of a class of double-rank minimization algorithms. *JIMA*, 1970.

[16] F. W. Byron and R. W. Fuller. *The Mathematics of Classical and Quantum Physics*, chapter 9. Dover, 1992.

[17] H. Chen and B. Bhanu. 3d free-form object recognition in range images using local surface patches. *Pattern Recognition Letters*, 28(10):1252–1262, 2007.

[18] R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[19] B. Coltin and M. Veloso. Multi-observation sensor resetting localization with ambiguous landmarks. In *AAAI*, 2011.

[20] J. Czyz, B. Ristic, and B. Macq. A particle filter for joint detection and tracking of color objects. *IVC*, 2007.

[21] M. Dou, J. Taylor, H. Fuchs, A. Fitzgibbon, and S. Izadi. 3d scanning deformable objects with a single rgbd sensor. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[22] H. Edelsbrunner and E.P. Mücke. Three-dimensional alpha shapes. *Trans. Graph.*, 1994.

[23] P. F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, September 2010.

[24] M. Garland and P. Heckbert. Surface simplification using quadric error metrics.

[25] S. Garrido-Jurado, R. Muñoz Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 2014.

[26] G. Guennebaud and M. Gross. Algebraic point set surfaces. *ACM Trans. Graph.*, 2007.

[27] J. Hervieu and P. Morand-Fehr. Comment noter l'état corporel des chèvres. pages 26—-32, 1999.

[28] F.V. Hundelshausen and M. Veloso. Active monte carlo recognition. In *GCAI*, 2007.

[29] C. Y. Ip, D. Lapadat, L. Sieger, and W. C. Regli. Using shape distributions to compare solid models. In *SMA*, 2002.

[30] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *UIST*, 2011.

[31] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *PAMI*, 1999.

[32] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics*, 2006.

[33] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 2012.

[34] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, and R. Kimmel. Photometric heat kernel signatures. In *Proceedings of the Third international conference on Scale Space and Variational Methods in Computer Vision*, SSVM'11, pages 616–627, Berlin, Heidelberg, 2012. Springer-Verlag.

[35] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *In IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.

[36] K Lai, L. Bo, X. Ren, and D. Fox. Sparse distance learning for object recognition combining rgb and depth information. In *ICRA*, 2011.

[37] D. Levin. Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization*, 3, 2003.

[38] D. A. Levin, Y. Peres, and Wilmer E. L. *Markov Chains and Mixing Times*, chapter 13. American Mathematical Society, 2006.

[39] D. A. Levin, Y. Peres, and Wilmer E. L. *Markov Chains and Mixing Times*, chapter 20. American Mathematical Society, 2006.

[40] G. Liu, Z. Xi, and J. Lien. Dual-space decomposition of 2d complex shapes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[41] R. Liu, H. Zhang, A. Shamir, and D. Cohen-Or. A part-aware surface metric for shape analysis. *Computer Graphics Forum*, 28(2):397–406, 2009.

[42] M. Mahmoudi and G. Sapiro. Three-dimensional point cloud recognition via distributions of geometric distances. *Graphical Models*, 71(1):22 – 31, 2009.

[43] S. Marras, F. Ganovelli, P. Cignoni, R. Scateni, and R. Scopigno. Controlled and adaptive mesh zippering. *GRAPP International Conference on Computer Graphics Theory and Applications*, pages 104–109, 2010.

[44] J.R. Munkres. *Elements of Algebraic Topology*. Westview Press, 1984.

[45] K. Okada, M. Kojima, S. Tokutsu, T. Maki, Y. Mori, and M. Inaba. Multi-cue 3d object recognition in knowledge-based vision-guided humanoid robot system. In *IROS*, 2007.

[46] K. Okuma, A. Taleghani, N. De Freitas, O. De Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, 2004.

[47] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM Trans. Graph.*, 2002.

[48] M. Ovsjanikov, Q. Mérigot, F. Mémoli, and L. J. Guibas. One point isometric matching with the heat kernel. *Comput. Graph. Forum*, 29(5):1555–1564, 2010.

[49] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. Guibas. Example-based 3d scan completion. In *Symposium on Geometry Processing*, pages 23–32, 2005.

[50] Z. Ren, J. Yuan, C. Li, and W. Liu. Minimum near-convex decomposition for robust shape representation. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 303–310, Nov 2011.

[51] M. Reuter. Hierarchical Shape Segmentation and Registration via Topological Features of Laplace-Beltrami Eigenfunctions. *International Journal of Computer Vision*, 89(2):287–308, September 2010.

[52] Fernando Ribeiro, Susana Brandão, João P. Costeira, and Manuela Veloso. Global localization by soft object recognition from 3d partial views. In *IROS*, 2015.

[53] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009.

[54] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. In *IROS*, October 18-22 2010.

[55] R. B. Rusu and D. Cousins. 3D is here: Point Cloud Library (PCL). In *ICRA*, May 9-13 2011.

[56] Y. Sahillioglu and Y. Yemez. 3d shape correspondence by isometry-driven greedy optimization. In *CVPR*, pages 453–458. IEEE, 2010.

[57] P. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, March 1966.

[58] A. Sharma, R. Horaud, J. Cech, and E. Boyer. Topologically-robust 3d shape matching based on diffusion geometry and seed growing. In *CVPR*, pages 2481–2488, 2011.

[59] P. Skraba, M. Ovsjanikov, F. Chazal, and L. Guibas. Persistence-based segmentation of deformable shapes. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, page 2146, 2010.

[60] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *PAMI*, 17(4):344–358, 1995.

[61] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *SGP*, 2009.

[62] H. Tabia, H. Laga, D. Picard, and P. Gosselin. Covariance descriptors for 3d shape matching and retrieval. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[63] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. ECCV'10, pages 356–369, 2010.

[64] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 311–318, New York, NY, USA, 1994. ACM.

[65] A. Vieira, S. Brandão, A. Monteiro, I. Ajuda, and G. Stilwell. Development and validation of a visual body condition scoring system for dairy goats. *Journal of Dairy Science*, 2015.

[66] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[67] M. Wardetzky, S. Mathur, F. Kälberer, and E. Grinspun. Discrete laplace operators: no free lunch. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 33–37. Eurographics Association, 2007.

[68] N. Werghi, C. Tortorici, S. Berretti, and A. Del Bimbo. Representing 3d texture on mesh manifolds for retrieval and recognition applications. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2015.

[69] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[70] W. Wohlkinger and M. Vincze. Ensemble of shape functions for 3d object classification. In *ROBIO*, 2011.

[71] Y. Zhong. Intrinsic shape signatures: A shape descriptor for 3D object recognition. In *ICCV Workshops*, pages 689–696, September 2009.