Roberto Toldo

Towards automatic acquisition of high-level 3D models from images

Ph.D. Thesis

May 16, 2013

Università degli Studi di Verona
Dipartimento di Informatica

Advisor:
prof. Andrea Fusiello

Series N°: **TD-07-13**

Università di Verona
Dipartimento di Informatica
Strada le Grazie 15, 37134 Verona
Italy

Abstract

In recent years there has been a surge of interest in automatic modeling from images. While the current state of the art in three-dimensional reconstruction has focused on the recovery of dense and accurate representations of objects imaged through pictures or video, the sustained interest in accessible modeling software is a strong evidence of an untapped general need for compact, abstract representations of objects. In this thesis, the problem of producing high level models starting from images is discussed in details. In the first part, an automatic uncalibrated Structure from Motion pipeline is presented. Starting from the output of the pipeline, two different approaches of generating high-level renditions are studied. The first approach employs a novel Multiple view Stereo algorithm to produce a dense and accurate point cloud. A retrieval system for meshes, based on segmentation and Bag of Words, is then introduced. In the latter approach, the sparse Structure from Motion point cloud is fitted by planes and planar patches. Planar patches are a compact, intermediate representation of the scene. Both branches of the thesis aim to narrow the gap between scene acquisition and interpretation, through the definition of high level renditions produced by very different strategies.

Preface

During my PhD study at the University of Verona, I have produced twelve conference papers and two journal papers (one of which is currently under review). This thesis is based on the content presented in those papers, as listed in the following:

- R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *European Conference on Computer Vision*, pages 537-547, 2008.
- M. Farenzena, A. Fusiello, R. Gherardi, and R. Toldo. Towards unsupervised reconstruction of architectural models. In *Vision, Modeling, and Visualization*, pages 41-50, 2008.
- M. Farenzena, A. Fusiello, R. Gherardi, and R. Toldo. Automatic structure recovery and visualization. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, volume 18, 2009.
- R. Toldo and A. Fusiello. Automatic estimation of the inlier threshold in robust multiple structures fitting. In *International Conference on Image Analysis and Processing*, pages 123-131, 2009.
- R. Toldo, U. Castellani, and A. Fusiello. A bag of words approach for 3d object categorization. In *Computer Vision/Computer Graphics Collaboration Techniques*, pages 116-127, 2009.
- R. Toldo, U. Castellani, and A. Fusiello. Visual vocabulary signature for 3d object retrieval and partial matching. In *Eurographics Workshop on 3D Object Retrieval*, pages 21-28, 2009.
- R. Toldo and A. Fusiello. Real-time incremental j-linkage for robust multiple structures estimation. In *International Symposium on 3D Data Processing, Visualization and Transmission*, page 6, 2010.
- R. Gherardi, R. Toldo, M. Farenzena, and A. Fusiello. Samantha: towards automatic image-based model acquisition. In *Visual Media Production (CVMP)*, 2010 Conference on, pages 161-170, 2010.
- R. Toldo and A. Fusiello. Photo-consistent planar patches from unstructured cloud of points. In *European Conference on Computer Vision*, pages 589-602, 2010.
- R. Toldo, A. Beinat, and F. Crosilla. Global registration of multiple point clouds embedding the generalized procrustes analysis into an icp framework. In *3D Data Processing Visualization and Transmission Conference*, 2010.

- R. Toldo, U. Castellani, and A. Fusiello. The bag of words approach for retrieval and categorization of 3d objects. *The Visual Computer*, 26(10):1257-1268, 2010.
- R. Gherardi, R. Toldo, V. Garro, and A. Fusiello. Automatic camera orientation and structure recovery with samantha. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, pages 38-5, 2011.
- R. Toldo, F. Fantini, L. Giona, S. Fantoni, and A Fusiello. Accurate multiview stereo reconstruction with fast visibility integration and tight disparity bounding. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, 2013, In press.
- R. Toldo and A. Fusiello. Image-consistent patches from unstructured points with j-linkage. Submitted to *IMAVIS* journal.

Contents

List of Algorithms

1

Introduction

Nowadays digital cameras are everywhere. The overwhelming diffusion and the recent technological advances of cellular phones, tablets, PDAs, laptops and the increasing accessibility to professional photographic equipment have caused a surge of interest in automatic modeling from images.

In Computer Vision, image-based modeling is the process of automatically extracting geometric objects and scene models from images. In recent years, the progress in this research field has been, and still is, remarkable. The research is proceeding arm in arm with commercial applications that benefit from this technology, ranging from the reconstruction of realistic 3D models for movies and video game industries to the everyday applications. For example, in Microsoft Photosynth [111] a new way of organizing photos has been introduced letting the user to seamlessly switch between images of a landmark site. The very recent Autodesk 123d Catch [3] allows to extract detailed 3D models from 2D images with no user intervention and by taking advantage of cloud computing. Three-dimensional models are becoming more and more important; for example in the well-known Google Earth project [71], one can now import its own 3D models to the existing database. These are just a few of the many examples.

Although the recent developments in three-dimensional reconstruction from images addressed the recovery of dense and accurate models, there is still an unfulfilled need for compact, abstract representations of objects. What separates dense triangulated reconstructions from higher-level renditions is a *semantic gap*, which still remains one of the most challenging topic in Computer Vision.

In this thesis, the problem of producing high-level models from images is discussed in details. The aim is to explore new methods and techniques to extract both detailed and high-level, abstract 3D models from images, towards the narrowing of the semantic gap. In Figure 1.1 the general outline of the thesis is shown. The proposed innovations are manifold and cover different branches of image-based automatic modeling.

In Chapter 2, an automatic Structure from Motion (SfM) pipeline is firstly introduced. The output of the pipeline will be later used by the methods presented in the next chapters of this dissertation. The system presents many innovations and improvements with respect to the current state of the art: it is very robust, thanks to its hierarchical processing and it is completely autocalibrated, thus not

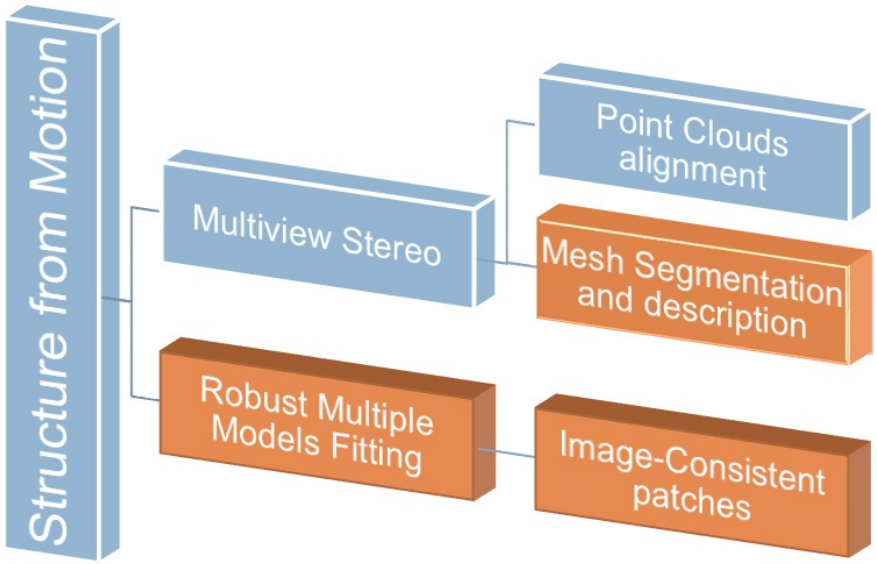


Figure 1.1. Thesis outline. Low-Level stages are colored in blue. Stages producing High-Level information are colored in orange.

requiring any user input or meta-data in the images. Starting from the output of the pipeline, two very different approaches of generating high-level models are studied.

In the first approach a novel Multiview Stereo (MVS) algorithm, presented in Chapter 3, is employed to produce a dense point cloud and a detailed triangulated surface. The algorithm wisely makes use of all the information coming from the SfM pipeline for a preliminary tight depth bounding estimation. In addition, a fast visibility-based technique is introduced to remove rogue points. The method can produce very accurate three-dimensional models and has been engineered with some parts taking advantage of the Graphics Processing Unit (GPU). In Chapter 4 a method for registering multiple point clouds, like the one produced by a MVS pipeline or by a laser scanner, is introduced. The algorithm makes use of the Generalized Procrustes Analysis (GPA) and simultaneously register all the point clouds at each iteration, increasing the convergence speed and robustness with respect to pairwise approaches. In Chapter 5 a retrieval system for meshes, based on segmentation and a Bag of Words framework, is introduced. The retrieval pipeline is very general and can be applied to any triangular mesh. Mesh segmentation and description represent a step forward in the direction of higher level renditions.

In the last part of the thesis, a very different approach for generating high-level, abstract models is followed. SfM point clouds are directly fitted by planes and planar patches. In Chapter 6 a novel framework, called J-Linkage, to fit multiple instances of the same geometric model to noisy data is introduced. J-Linkage has several advantages over existing RANSAC-based methods as it extracts models

by specifying a signature to each point and by performing a specific agglomerative clustering on them. Improvements to knock down the computing times and automatically estimate the inlier threshold are also presented in the same chapter. Finally, in Chapter 7, the J-Linkage framework is adapted and employed to extract planar patches. We will see how planar patches are a compact, abstract and intermediate representation of the scene. The final goal of both branches of the thesis is to produce high level renditions with very different strategies.

2

Structure from Motion

Three dimensional reconstruction is the process of recovering the properties of the environment and optionally of the sensing instrument from a series of measures. This generic definition is wide enough to accommodate very diverse methodologies, such as time-of-flight laser scanning, photometric stereo or satellite triangulation. The Structure from Motion (SfM) field of research is concerned with the recovery of the three dimensional geometry of the scene (the structure) when observed through a moving camera (the motion). Sensor data is either a video or a set of pictures; additional information, such as the calibration parameters, can be used if available. In this chapter our contributions to the problem of uncalibrated Structure from Motion from pictures i.e, the problem of recovering a three dimensional model of a scene given a set of images, will be described. The sought result is generally an unstructured 3D point cloud, consisting of the keypoints which were identified and tracked in the scene and a set of camera matrices, identifying position and direction of each picture with respect to an arbitrary reference frame.

Current state of the art follows typically a sequential pattern, incrementally growing a seed reconstruction adding one view at a time. Instead, in this chapter it will be proposed to describe the entire reconstruction process as a binary tree, constructed by agglomerative clustering over the set of views. Each leaf corresponds to a single image, while internal nodes represent partial reconstructions obtained by merging the left and right sub-nodes. The reconstruction proceed from bottom to top, starting from several seed couple and eventually reaching the root node, corresponding to the final, complete result. It will be demonstrated that this paradigm has several advantages over the sequential one, both in terms of computational performance (which improves by one order of magnitude on average) and overall error containment. Such a system provides true scalability, since it is inherently parallelizable.

The pipeline makes use of auto-calibration of self-calibration, which is the process of automatic estimation from images of the internal parameters of the cameras that captured them. Current Structure from Motion research has partly sidestepped the issue using ancillary data such as EXIF tags embedded in recent image formats. Their presence or consistency however is not guaranteed, and poses a problem especially when the number of images is not big enough to provide sufficient information to jump start the reconstruction process.

The rest of the Chapter is organized as follows. In the next three sections the basic stages of SAMANTHA, our reconstruction pipeline will be described. The pipeline departs from the sequential paradigm prevalent in current literature and embraces instead a hierarchical approach. In particular the three stages presented are the matching stage (Section 2.2), the view clustering (Section 2.3) and the actual reconstruction (Section 2.4).

Next, a new clustering strategy will be introduced, derived from the simple linkage, that makes the dendrogram more balanced (Section 2.5), thereby reducing the actual complexity of the method, and endows SAMANTHA with the capability of dealing with uncalibrated images (Section 2.6) by including the self-calibration approach described in Section 2.7. Being conscious that "the devil is in the detail" extreme care will be used in describing every detail of the pipeline that makes it reproducible. Experiments reported in Section 2.8 corroborate our claims. Finally, conclusions are drawn in Section 2.9

The proposed method has several advantages, like a provably lower computational complexity which is necessary to achieve true scalability and better error containment leading to more stability and less drift.

2.1 Related works

In recent years there has been a surge of interest in automatic architectural/urban modeling from images. Literature covers several approaches for solving this problem. These can be categorized in two main branches: A first one is composed of methods specifically tailored for urban environments and engineered to run in real-time [32,127]. These systems usually rely on a host of additional information, such as GPS/INS navigation systems and camera calibration. The second category, where our contributions are situated, comprises Structure from Motion pipelines that process images in batch and handle the reconstruction process making no assumptions on the imaged scene and on the acquisition rig [12,85,89,155,192].

The main challenges to be solved are computational efficiency (in order to be able to deal with more and more images) and generality. As for the first issue, several different solutions has been explored: the most successful have been those aimed at reducing the impact of the bundle adjustment phase, which with feature extraction dominates the computational complexity. A class of solutions that have been proposed are the so-called partitioning methods [51]. They reduce the reconstruction problem into smaller and better conditioned subproblems which can be effectively optimized. Two main strategies can be distinguished.

The first one is to tackle directly the bundle adjustment algorithm, exploiting its properties and regularities. The idea is to split the optimization problem into smaller, more tractable components. The subproblems can be selected analytically as in [156], where spectral partitioning has been applied to SaM, or they can emerge from the underlying 3D structure of the problem, as described in [118]. The computational gain of such methods is obtained by limiting the combinatorial explosion of the algorithm complexity as the number of images and feature points increases.

The second strategy is to select a subset of the input images and feature points that subsumes the entire solution. Hierarchical sub-sampling was pioneered by [51],

using a balanced tree of trifocal tensors over a video sequence. The approach was subsequently refined by [120], adding heuristics for redundant frames suppression and tensor triplet selection. In [150] the sequence is divided into segments, which are resolved locally. They are subsequently merged hierarchically, eventually using a representative subset of the segment frames. A similar approach is followed in [68], focusing on obtaining a well behaved segment subdivision and on the robustness of the following merging step. The advantage of these methods over their sequential counterparts lays in the fact that they improve error distribution on the entire dataset and bridge over degenerate configurations. Anyhow, they work for video sequences, so they cannot be applied to unordered, sparse images. A recent work [154] that works with sparse dataset describes a way to select a subset of images whose reconstruction provably approximates the one obtained using the entire set. This considerably lowers the computational requirements by controllably removing redundancy from the dataset. Even in this case, however, the images selected are processed incrementally. Moreover, this method does not avoid computing the epipolar geometry between all pairs of images.

There is actually a third solution covered in literature, orthogonal to the aforementioned approaches. In [1], the computational complexity of the reconstruction is tackled by throwing additional computational power to the problem. Within such framework, the former algorithmical challenges are substituted by load balancing and subdivision of reconstruction tasks. Such direction of research strongly suggests that the current monolithical pipelines should be modified to accommodate ways to parallelize and optimally split the workflow of reconstruction tasks.

The generality issue refers to the assumption that are made on the input images, or, equivalently on the amount of ancillary information that is required in addition to pixels values. Existing pipelines either assumes known internal parameters [12,85], or constant internal parameters [89,192], or relies on EXIF data plus external information (camera CCD dimensions) [155].

To the best of our knowledge, despite auto calibration with varying parameters have been introduced several years ago [126], no working SaM pipeline have been demonstrated yet with varying parameters and no ancillary information.

A new hierarchical and parallelizable scheme for SaM will be presented. The images are organized into a hierarchical cluster tree, as in Figure 2.4, and the reconstruction proceeds hierarchically along this tree from the leaves to the root. Partial reconstructions correspond to internal nodes, whereas images are stored in the leaves. This scheme provably cuts the computational complexity by one order of magnitude (provided that the dendrogram is well balanced), and it is less sensible to typical problems of sequential approaches, namely sensitivity to initialization [170] and drift [32]. It is also scalable and efficient, since it partitions the problem into smaller instances and combines them hierarchically, making it inherently parallelizable.

This approach has some analogy with [139], where a spanning tree is built to establish in which order the images must be processed. After that, however, the images are processed in a standard incremental way.

Recently a new approach have been presented in [33] that solve for a global set of camera poses given local estimates of geometry using a MRF. However, they incorporated geotags and other prior information (like known intrinsics) into the

problem. These assumptions are reasonable since they address a very large scale SaM problem. On the other hand, we cope with smaller areas but we do not exploit any other information besides the images themselves.

Our aim is to push the "pure" SaM technique problem as far as possible, to see what can be achieved without any including ancillary information.

Autocalibration (a.k.a. self-calibration) has generated a lot of theoretical interest since its introduction in the seminal paper by Maybank and Faugeras [110]. The attention spawned by the problem however is inherently practical, since it eliminates the need for off-line calibration and enables the use of content acquired in an uncontrolled environment. Modern computer vision has partly sidestepped the issue using ancillary information, such as EXIF tags embedded in some image formats. Such data unfortunately is not always guaranteed to be present or consistent with its medium, and does not extinguish the need for reliable autocalibration procedures.

Lots of published methods rely on Equations involving the dual image of the absolute quadric (DIAQ), introduced by Triggs in [186]. Earliest approaches for variable focal lengths were based on linear, weighted systems [126, 128], solved directly or iteratively [145]. Their reliability were improved by more recent algorithms, such as [19], solving super-linear systems while forcing directly the positive definiteness of the DIAQ. Such enhancements were necessary because of the structural non-linearity of the task: for this reason the problem has also been approached using branch and bound schemes, based either on the Kruppa Equations [59], dual linear autocalibration [8] or the modulus constraint [20].

The algorithm described in [76] shares with the branch and bound approaches the guarantee of convergence; the non-linear part, corresponding to the localization of the plane at infinity, is solved exhaustively after having used the cheiral inequalities to compute explicit bounds on its location.

The technique described in this chapter is closely related to the latter: first, we derive the location of the plane at infinity given two perspective projection matrices and a guess on their intrinsic parameters, and subsequently use this procedure to iterate through the space of camera intrinsic parameters looking for the best collineation that makes the reconstruction Euclidean. The search space is inherently bounded by the finiteness of the acquisition devices; each sample and the corresponding plane at infinity define a collineation of space whose likelihood can be computed evaluating skew, aspect ratio, principal point and related constraints for each transformed camera. The best solution is eventually refined via non-linear least squares.

Such approach has several advantages: it's fast, easy to implement and reliable, since a reasonable solution can always be found in non-degenerate configurations, even in extreme cases such as when autocalibrating just two cameras.

2.2 Keypoint Matching

In this section we describe the stage of Samantha that is devoted to the automatic extraction and matching of keypoints among all the n available images. Its output is to be fed into the geometric stage, that will perform the actual structure

and motion recovery. Good and reliable matches are the key for any geometric computation.

Although the building blocks of this stage are fairly standard techniques, we carefully assembled a procedure that is fully automatic, robust (matches are pruned to discard as much outliers as possible) and computationally efficient.

First of all, keyponts are extracted in all n images. Departing from SIFT, we tried several detector/descriptor. A novel scale-space feature extractor based on Difference of Gaussian, with a radial and symmetric descriptor has been used in the next. This solution was preferred to the SIFT because of patents issue, but it goes without saying that SIFT – in the excellent implementation provided by VLfeat [190] – can be used instead.

As the images are unordered, the first objective is to recover the *image graph*, i.e., the graph that tells which image overlaps (or can be matched) with which other.

This must be done in a computationally efficient way, without trying to match keypoints between every image pair. As a matter of fact, keypoint matching is one the most expensive stages, so one would like to reduce the number of images to be matched from $O(n^2)$ to $O(n)$.

In this phase we consider only a constant number of descriptors in each image (we used 500, where a typical image contains thousands of keypoints). Then, each keypoint description is matched to its ℓ (approximate) nearest neighbors in feature space (we use $\ell = 6$), using the ANN library [113] (with $\epsilon = 0.5$). A 2D histogram is then built that registers in each bin the number of matches between the corresponding views.

In the original approach [11] (which we followed also in [45]), every image is connected (in the image network) to the m images that have the greatest number of keypoints matches with it ($m = 8$). Hence, the number of images to match is $O(n)$, being m constant.

Consider the complete weighted graph $G = (V, E)$ where V are views and the weighted adjacency matrix is the 2D histogram. This graph is – in general – dense, have $|V| = O(n^2)$. The objective is to extract a subgraph G' with a number of edges that is linear in n. The strategy of [11] creates a graph with mn edges, where each node has degree m. When the number of images is large, it tends to create cliques of very similar images with weak (or no) inter-cliques connections. On the other hand, one would like to get an image graph that is strongly connected, to avoid over-fragmentation in the subsequent clustering phase. This idea is captured by the notion of *k-edge-connectedness*: In graph theory, a graph is k-edge-connected if it remains connected whenever fewer than k edges are removed. So, the graph produced by the original approach has a low k, while one would like to have k as high as possible (ideally $k = m$).

We devised a strategy that builds a subgraph G' of G which is "almost" m-edge-connected by construction.

1. Build the maximum spanning tree of G: the tree is composed by $n - 1$ edges;
2. remove them from G and add them to G';
3. repeat m times.

Figure 2.1. An example planar embedding of the image graph G' after the first iteration.

In the hypothesis that m spanning trees can be extracted from G, the algorithm produces a subgrapgh G' with $(n-1)m$ edges that is the union of m disjoint 1-edge-connected subgraph, hence it is m-edge-connected. Please note that by taking the *maximum* spanning tree we favor edges with high weight. So this strategy can be seen as a compromise between picking pairs with the highest score in the histogram and creating a strongly connected image graph.

If the hypothesis on G is not verified, a spanning forest will be obtained at a certain iteration, and G' will not be m-edge-connected. However, when $m \ll |E|$ one could expect that "almost" m spanning trees can be extracted from G without disconnecting it.

Matching follows a nearest neighbor approach [106], with rejection of those keypoints for which the ratio of the nearest neighbor distance to the second nearest neighbor distance is greater than a threshold (set to 1.5 in our experiments). The ANN built previously is obviously exploited here. We also discard matches that are not injective or symmetric (in other word, if A matches B, B must also match A).

Homographies and fundamental matrices between pairs of matching images are then computed using M-estimator SAmple Consensus (MSAC) [182], a variation of RANSAC that gives outliers a fixed penalty but scores inliers on how well they fit the data. This renders the output less sensitive to a higher inlier threshold, thereby rendering less critical the choice of the threshold, at no extra computational cost with respect to RANSAC. The random sampling is done with the bucketing technique [204], which produces samples of points well distributed over the image. This helps to reduce the number of iterations and provide more stable estimates. Since

LMEDS, RANSAC and its variants (like MSAC) have a low statistical efficiency, the model must be eventually re-estimated on a refined set of inliers[1].

Let e_i be the residuals of *all* the N points after MSAC, and let S^* be the sample that attained the best score; following [157], a robust estimator of the scale is:

$$\sigma^* = 1.4826 \left(1 + \frac{5}{N - |S^*|}\right) \sqrt{\text{med}_{i \notin S^*} e_i^2}. \tag{2.1}$$

The final set of inliers are those points such that

$$|e_i| < \theta \sigma^*, \tag{2.2}$$

where θ is a constant (we used 1.5).

The model parameters are eventually re-estimated on this set of inliers via least-squares minimization of the (first-order approximation of the) geometric error [26, 107].

The more likely model (homography or fundamental matrix) is selected according to the Geometric Robust Information Criterion (GRIC) [184]:

$$\begin{aligned} \text{GRIC} &= \sum \rho(e_i^2) + nd \log(r) + k \log(rn) \\ \rho(x) &= \min\left(x/\sigma^2, 2(r-d)\right) \end{aligned} \tag{2.3}$$

where σ is the standard deviation of the measurement error, k is number of parameters of the model, d is dimension of the fitted manifold, and r is the dimension of the measurements. In our case, $k = 7, d = 3, r = 4$ for fundamental matrices and $k = 8, d = 2, r = 4$ for homographies. The model with the lower GRIC is the more likely.

Finally, if the number of remaining matches between two images is less than a threshold (computed basing on a statistical test as in [11]) then they are discarded.

After that, keypoints matching in multiple images are connected into *tracks*: consider the undirected graph $G(V, E)$ where V are the keypoints and E represents matches; a track is a connected component of G. Vertices are labeled with the view the keypoints belong to: an inconsistency arise when in track a label occurs more than once. Inconsistent tracks and those shorter than three frames are discarded.

Figure 2.2. Tracks over a 12-views set. For the sake of readability only a sample of 50 tracks over 2646 have been plotted.

[1] It is understood that when we will refer to MSAC in the following, this procedure is always carried out.

Figure 2.3. An example of one image (top left) from "Piazza Bra" and its six closest neighbors according to the affinity defined in Equation 2.4.

2.3 Views Clustering

The images are organized into a tree with agglomerative clustering, using a measure of overlap as the distance. The reconstruction then follows this tree from the leaves to the root. As a result, the problems is broken into smaller instances, which are then separately solved and combined.

algorithms for image views clustering have been proposed in literature in the context reconstruction [139], panoramas [11], image mining [131] and scene summarization [151]. The distance being used and the clustering algorithm are application-specific.

In our case we deploy an image affinity measure that befits the structure-and-motion reconstruction task. It is computed by taking into account the number of common keypoints and how well they are spread over the images.In formulae, let S_i and S_j be the set of matching keypoints in image I_i and I_j respectively:

$$a_{i,j} = \frac{1}{2}\frac{|S_i \cap S_j|}{|S_i \cup S_j|} + \frac{1}{2}\frac{CH(S_i) + CH(S_j)}{A_i + A_j} \tag{2.4}$$

where $CH(\cdot)$ is the area of the convex hull of a set of points and A_i (A_j) is the total area of image I_i (I_j). The first term is an affinity index between sets, also known as Jaccard index. The distance is $(1 - a_{i,j})$, as $a_{i,j}$ ranges in $[0, 1]$.

Views are grouped together by agglomerative clustering, which produces a hierarchical, binary cluster tree, called *dendrogram*. The general agglomerative clustering algorithm proceeds in a bottom-up manner: starting from all singletons, each sweep of the algorithm merges the two clusters with the smallest distance. The way the distance between clusters is computed produces different flavors of the algorithm, namely the simple linkage, complete linkage and average linkage [41]. We selected the *simple linkage* rule: The distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters.

Simple linkage clustering is appropriate to our case because: i) the clustering problem *per se* is fairly easy, ii) nearest neighbors information is readily available with ANN and iii) it produces "elongated" or "stringy" clusters which fits very well with the typical spatial arrangement of images sweeping a certain area or building.

Figure 2.4. An example of dendrogram for a 12-views set.

2.4 Hierarchical Reconstruction

The dendrogram produced by the clustering stage imposes a hierarchical organization of the views that will be followed by SAMANTHA. At every node in the dendrogram an action must be taken, that augment the reconstruction (cameras + 3D points). Three operations are possible: When two views are merged a two-views reconstruction must be performed. When a view is added to a cluster a resection-intersection step must be taken (as in the standard sequential pipeline). When two non-trivial clusters are merged an absolute orientation problem must be solved. Each of these steps is detailed in the following.

While it is useful to conceptually separate the clustering from the reconstruction, actually the two phases occurs simultaneously: during the simple linkage iteration, every time a merge is attempted the corrisponding reconstruction step is taken. If it fails, the merge is discarded and the next possible merge is considered.

Compared to the standard sequential approach, this framework has a lower computational complexity, is independent from the initial pair of views, and copes better with drift problems, typical of sequential schemes.

2.4.1 Two-views reconstruction

Let us assume *pro tempore* that the intrinsic parameters are known; this constraint will be removed in Section 2.6.

The extrinsic parameters of two given views are obtained by factorizing the essential matrix (obtained from F), as in [78]. Then 3D points are reconstructed

by *intersection* (or triangulation) and the reconstruction is refined with bundle adjstement [187].

Triangulation (or intersection) is performed by the iterated linear LS method [79]. Points are pruned by analyzing the condition number of the linear system and the reprojection error. The first test discards ill-conditioned 3D points, using a threshold on the condition number of the linear system (10^4, in our experiments). The second test applies the reprojection error of the points.

In order for this reconstriction to be successfull the two views must satisfy two conflicting requirements: have both a large number of keypoints in common and a baseline sufficiently large so as to allow a well-conditioned reconstruction. The first requirement is automatically verified by affinity defined in (2.4). The second requisite is tantamount to say that the fundamental matrix must explain the data far better than other models (namely, the homography), and this can be implemented by considering the GRIC, as in [128].

We therefore allow two views i and view j to merge in a cluster only if:

$$\text{gric}(F_{i,j}) < \alpha \, \text{gric}(H_{i,j}) \quad \text{with } \alpha \geq 1, \tag{2.5}$$

where $\text{gric}(F_{i,j})$ and $\text{gric}(H_{i,j})$ are the GRIC scores obtained by the fundamental matrix and the homography matrix respectively (we used $\alpha = 1.02$).

2.4.2 One-view addition

The reconstructed 3D points that are visible in the view to be added provides a set of 3D-2D correspondences, that are exploited to solve an exterior orientation problem. We used the P3P algorithm described in [96] inside MSAC, followed by non-linear minimization of the reprojection error at the end. The 3D structure is then updated with tracks that are visible in the last view. Triangulation is carried out as described before.

2.4.3 Fusion

Two partial reconstructions are to be conflated into one. They live in two different reference systems, therefore one has to be registered onto the other with a similarity transformation. They have, by construction, some 3D points in common, that are used to solve an absolute orientation problem with MSAC.

When a tentative model is evaluated inside MSAC, the residuals are computed as the lenght of 2D segments projected in the images, instead than the natural choice of considering 3D segments. This counterintuitive decision allows us to set a meaningful inlier threshold in pixels, whereas a threshold in 3D space would have been pointless, given the scale ambiguity. The final refinement, computed with the Procrustean method [90], minimizes the proper geometric residual, i.e. sum of squared distances of 3D points.

Once the cameras are registered, the common 3D points are re-computed by intersection, with the same cautions as before, namely the reprojection error threshold and test of the conditioning number. Intersection is also performed on any track that becomes visible after the merging. The new reconstruction is finally refined with bundle adjustment.

In general, the triangulation obeys the following strategy. As soon as one track reaches lenght two (restricted to the reconstructed views only), the point is triangulated. If the operation fails (because of one of the sanity checks described above) the 3D point is provisionally discarded but the track is kept. An attempt to reconstruct the 3D point is undertaken everytime the lenght of track increses.

2.4.4 Complexity analysis

The hierarchical approach that have been outlined above allows to decrease the computational complexity with respect to the sequential SaM pipeline. Indeed, if the number of views is n and every view adds a constant number of points ℓ to the reconstruction, the computational complexity[2] in time of sequential SaM is $O(n^5)$, whereas the complexity of SAMANTHA (in the best case) is $O(n^4)$.

The cost of bundle adjustment with m points and n views is $O(mn(m+2n)^2)$ [150], hence it is $O(n^4)$ if $m = \ell n$.

In the sequential SaM, adding view i requires a constant number of bundle adjustments (typically one or two) with i views, hence the complexity is

$$\sum_{i=1}^{n} O(i^4) = O(n^5). \tag{2.6}$$

In the case of the hierarchical approach, consider a node of the dendrogram where two clusters are merged into a cluster of size n. The cost $T(n)$ of adjusting that cluster is given by $O(n^4)$ plus the cost of doing the same onto the left and right subtrees. In the hypothesis that the dendrogram is well balanced, i.e., the two clusters have the same size, this cost is given by $2T(n/2)$. Hence the asymptotic time complexity T in the best case is given by the solution of the following recurrence:

$$T(n) = 2T(n/2) + O(n^4) \tag{2.7}$$

that is $T(n) = O(n^4)$ by the third branch of the Master's theorem [31].

The worst case is when a single cluster is grown by adding one view at a time. In this case, which corresponds to the sequential case, the dendrogram is extremely unbalanced and the complexity drops to $O(n^5)$.

2.5 Dendrogram balancing

As demonstrated in precedence, the hierarchical framework can provide a provable computational gain, provided that the resulting tree is well-balanced. The worst case complexity, corresponding to a sequence of single view additions, is no better than the standard sequential approach. It is therefore crucial to ensure a good balance during the clustering phase. Our solution is to employ a novel clustering procedure, which instead of following a completely greedy strategy promotes the

[2] We are considering here only the cost of bundle adjustment, which clearly dominates the other operations.

creation of better balanced dendrograms. This technique has been introduced in [66].

The view clustering procedure proposed in the previous Section allows us to organize the available views into a hierarchical cluster structure (a tree) that will guide the reconstruction process. This approach decreases the computational complexity with respect to sequential SaM pipelines, from $O(n^5)$ to $O(n^4)$ in the best case, i.e. when the tree is well balanced (n is the number of views). If the tree is unbalanced this computational gains vanishes. It is therefore crucial to enforce the balancing of the tree.

The preceding solution, which used the simple rule, specified that the distance between two clusters is to be determined by the distance of the two closest objects (nearest neighbors) in the different clusters. In order to produce better balanced trees, we modified the agglomerative clustering strategy as follows: starting from all singletons, each sweep of the algorithm merges the pair with the smallest cardinality among the closest pair of clusters. The distance is computed according to the simple linkage rule. The cardinality of a pair is the sum of the cardinality of the two clusters. In this way we are softening the closest first agglomerative criterion by introducing a competing smallest first principle that tends to produce better balanced dendrograms.

The amount of balancing is regulated by the parameter ℓ : when $\ell = 1$ this is the standard agglomerative clustering with no balancing; when $\ell n/2$ (n is the number of views) a perfect balanced tree is obtained, but the clustering is poor, since distance is largely disregarded. We found in our experiments that a good compromise is $\ell = 5$, but this value is not critical.

Figure 2.5 shows an example of balancing achieved by our technique. The height of the tree is reduced from 14 to 9 and more initial pairs are present in the dendrogram on the right.

2.6 Uncalibrated Hierarchical Reconstruction

The main difference from the previous hierarchical approach is that now leaf nodes do not have proper calibration right from the start of the reconstruction process. The reconstruction starts projective, and as soon as a cluster reaches a given dimension m, the Euclidean upgrade procedure is triggered (in principle autocalibration with known skew and aspect ratio requires a minimum of $m = 4$ views to work; for good measure we used $m = 10$).

However, we strive to maintain an "almost" euclidean reference frame even for clusters smaller than m, to better condition reconstruction. Hence autocalibration is triggered for clusters of cardinality $\leq m$, starting from two views. The result is an approximate Euclidean upgrade; in fact these clusters are still regardes as projective, until they reach cardinality m. After that point autocalibration is not performed any more and the internal parameters of each camera are refined further only with bundle adjustment, as the reconstruction proceeds. In order to not to hamper the computation too much, the internal parameters of a camera becomes fixed as soon as they have been bundle-adjusted together with at least $k > m$ cameras (we used $k = 25$).

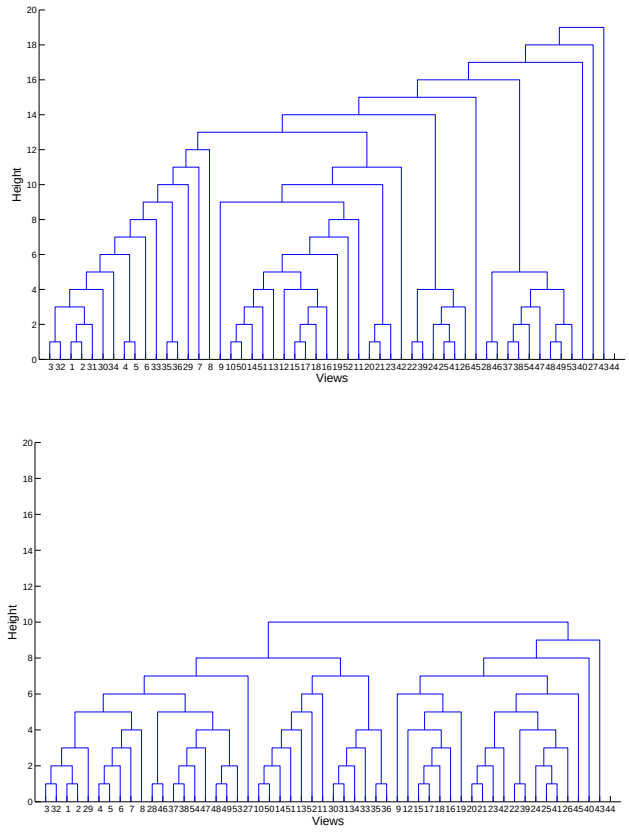


Figure 2.5. An example of the dendrogram produced by [45] (top) and the more balanced dendrogram produced by our technique (bottom) on a 52-views set, with $\ell = 5$.

Each step of hierarchical reconstruction must therefore be modified to accommodate for projective clusters.

2.6.1 Two-views reconstruction

The reconstruction from two views is always projective. The following two camera matrices are used:

$$P_1 = [I \mid \mathbf{0}] \quad \text{and} \quad P_2 = [[\mathbf{e}_2]_\times F \mid \mathbf{e}_2], \tag{2.8}$$

This canonical pair yields a projective reconstruction with the plane at infinity passing through the center of the second. This reconstruction is related to the Euclidean one by a projective of the 3D space, H.

Section 2.7 will describe the used method for guessing the best plane at infinity – equivalently H – compatible with rough focal estimates, obtained from the magnitude of the image diagonal. Even when the true focal lengths are far from

the estimates, this procedure will provide a useful, well conditioned starting point for the subsequent reconstruction steps.

Cheirality is then tested and enforced on the reconstruction. In practice only a reflection (switches all points from in front of the cameras to behind) may be necessary, as the twisted pair case never occur. In fact, the twisting corresponds to the infinity plane crossing the baseline, which would imply that tour guess for the infinity plane is very poor.

The position in space of the points is then obtained by triangulation (a.k.a. intersection) as before. Eventually bundle adjustment is run to improve the reconstruction.

2.6.2 One-view addition

The reconstructed 3D points that are visible in the view to be added provides a set of 3D-2D correspondences, that are exploited to glue the view to the partial reconstruction. This can be done by exterior orientation (as before) or by resection with DLT [77], depending on whether the partial reconstruction is Euclidean or projective (a single view is always uncalibrated). MSAC [182] is used in both cases in order to cope with outliers. The non-linear refinement minimizes the reprojection error. The view that has been glued might have brought in some new tracks, that are triangulated as described before. Finally, bundle adjustment is run on the current reconstruction (either Euclidean or projective).

2.6.3 Fusion

Partial reconstructions live in two different reference systems, that are related by a similarity if both are Euclidean or by a projectivity if one is uncalibrated. The points that they have in common are the tie points that serve to the purpose of computing the unknown transformation, using MSAC to discard wrong matches. In this case the projectivity that brings the projective reconstruction onto the Euclidean one is sought, thereby obtaining the correct Euclidean basis. Inside MSAC the projectivity is computed with DLT on 5 correspondences, scored with the image reprojection error and finally refined using non-linear minimization of the geometric error (distance of 3D points).

The new reconstruction is refined with bundle adjustment (either Euclidean or projective) and upgraded to a Euclidean frame when the conditions stated beforehand are met.

2.7 Autocalibration

As customary, we assume being given a projective reconstruction $\{P_i; X_j\}$ $i = i \dots n;\ j = 1 \dots m$. The purpose of autocalibration is therefore to find the collineation of space H such that $\left\{P_i H; H^{-1} X_j\right\}$ is a *Euclidean* reconstruction, i.e., it differs from the true one by a similarity.

The set of camera matrices can always be transformed to the following canonical form by post-multiplying each P_i by the matrix $[P_1;\ 0\ 0\ 0\ 1]^{-1}$:

$$P_1 = [I \mid \mathbf{0}] \qquad P_i = [Q_i \mid \mathbf{q}_i]\,. \tag{2.9}$$

In this situation, the collineation of space H performing the Euclidean upgrade has the following structure:

$$H = \begin{bmatrix} K_1 & \mathbf{0} \\ \mathbf{v}^\top & \lambda \end{bmatrix} \tag{2.10}$$

where K_1 is the calibration matrix of the first camera, $\mathbf{v}$ a vector which determines the location of the plane at infinity and λ a scalar fixating the overall scale of the reconstruction.

The method used for autocalibration is the one presented in [67]. It works by enumerating through the inherently bounded space of internal camera parameters in order to find the best rectifying homography. In this section it will be summarized, for further details please refer to the original paper.

The used technique is based on two stages:

1. Given a guess on the intrinsic parameters of two cameras compute a consistent upgrading collineation. This yields an estimate of all cameras but the first.
2. Score the intrinsic parameters of these $n-1$ cameras based on the likelihood of skew, aspect ratio and principal point.

The space of the intrinsic parameters of the two cameras is enumerated and the best solution is eventually refined via non-linear least squares.

To score each sampled point (f_1, f_2), we consider the aspect ratio, skew and principal point location of the resulting transformed camera matrices and aggregate their respective value into a single cost function:

$$\{f_1, f_2\} = \arg\min_{f_1, f_2} \sum_{\ell=2}^{n} \mathcal{C}^2(K_\ell) \tag{2.11}$$

where K_ℓ is the intrinsic parameters matrix of the ℓ-th camera after the Euclidean upgrade determined by (f_1, f_2), and

$$\mathcal{C}(K) = \overbrace{w_{sk}|k_{1,2}|}^{\text{skew}} + \overbrace{w_{ar}|k_{1,1} - k_{2,2}|}^{\text{aspect ratio}} + \overbrace{w_{u_o}|k_{1,3}| + w_{v_o}|k_{2,3}|}^{\text{principal point}} \tag{2.12}$$

where $k_{i,j}$ denotes the entry (i, j) of K and w are suitable weights, computed as in [128].

The entire procedure is presented as pseudo-code in Algorithm 1. With the perspective projection matrices the code presented takes as input also the viewport matrices of the cameras, defined as:

$$V = \frac{1}{2} \begin{bmatrix} \sqrt{w^2 + h^2} & 0 & w \\ 0 & \sqrt{w^2 + h^2} & h \\ 0 & 0 & 2 \end{bmatrix} \tag{2.13}$$

where w and h are respectively the width and height of each image.

Algorithm 1 Autocalibration

Input: a set of PPMs P and their viewports V.
Output: their upgraded, euclidean counterparts.

1. Normalize each P: $P \leftarrow V^{-1}P/\|P_{3,1:3}\|$
2. iterate over focal pairs K_1, K_2
   a) Compute Π_∞
   b) Build H from (2.10)
   c) Iterate over PPMs P
      i. $P_E \leftarrow PH$
      ii. $K \leftarrow$ *intrinsics of* P_E
      iii. Compute $\mathcal{C}(K)$ from (2.12)
3. Aggregate cost and select minimum
4. Refine non-linearly
5. De-normalize P and perform euclidean upgrade: $P \leftarrow VPH$

2.8 Experiments

In order to test the effectiveness of the proposed SfM pipeline, we run it on several real-cases, challenging datasets. Samantha has been entirely written in c++ by the Author and it can be downloaded from the web [3]. The results in this section were produced with version 1.30. We compared our hierarchical Structure from Motion solution with VisualSfM [199, 200], a recent sequential pipeline. While the superiority of Samantha over Bundler has been already demonstrated in [66], VisualSfm represents an improved sequential solution which makes use of a novel single parameter radial distortion model to improve robustness and efficiently performs the computation on the GPU. The reported results were obtained with version 0.5.22 of VisualSfM.

In all the experiments of this Section, the intrinsic camera parameters were considered to be unknown for both pipelines. However, while Samantha uses a completely autocalibrated approach, VisualSfm extracts an initial guess of the intrinsic parameters from the image exif and a database of known sensor sizes for common cameras. It must be noticed that if this ancillary information was not available, VisualSfM could not run.

We considered four experiments. The first one, "Piazza Navona", contains 92 photos of the famous square in Rome, taken with a Samsung ST45 Camera at a fixed focal length of 35mm. The dataset is publicly available for download [4]. Results are shown in Figure 2.6. In this case, Samantha produced a complete and correct reconstruction of the square while VisualSfm failed to converge.

The second dataset, "Piazza Bra", is composed by 331 photos of the main square in Verona. Photos were taken with a Nikon D50 camera and a fixed focal length of 18mm. Also this dataset is publicly available for download [5]. Results are

[3] http://samantha.3dflow.net
[4] http://www.icet-rilevamento.lecco.polimi.it/index.php?view=article&id=322
[5] http://www.diegm.uniud.it/fusiello/demo/samantha/

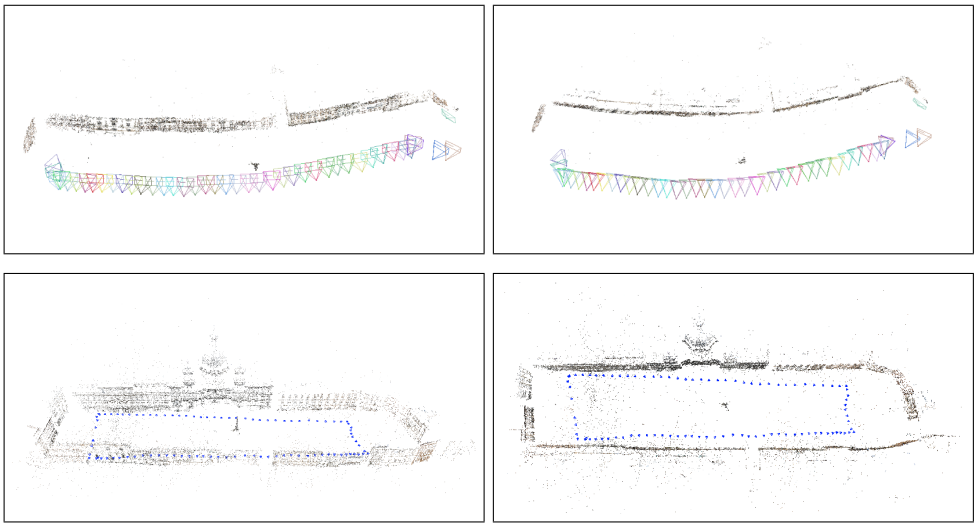


Figure 2.6. Comparative Structure from Motion Results from "Piazza Navona" dataset. Top: VisualSfm results, Bottom: Samantha results.

shown in Figure 2.7. Both VisualSfM and Samantha produced a correct result in this case. It can be also noticed that Bundler was failing with the same dataset [66].

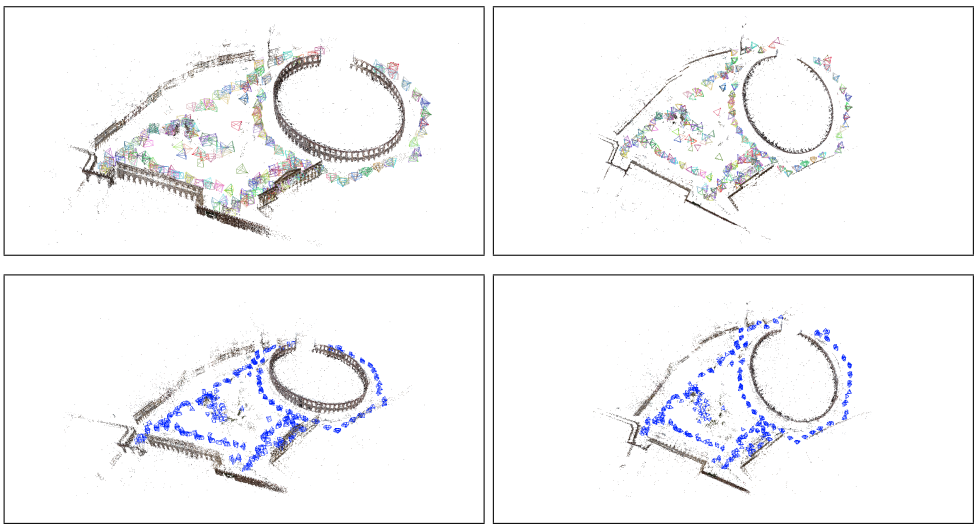


Figure 2.7. Comparative Structure from Motion Results from "Piazza Bra" dataset. Top: VisualSfm results, Bottom: Samantha results.

The third test, "Duomo di Pisa", is composed by 309 photos of the Duomo of Pisa in the famous Piazza dei Miracoli square. The dataset is composed of three sets of photos taken with a Nikon D40X camera at different focal lengths (13mm,

20mm and 58mm). Results are shown in Figure 2.8. Also in this case both the sequential and the hierarchical pipelines converged to an optimal solution. The dataset is probably the less challenging one as there are many photos covering a relatively small and limited scene.

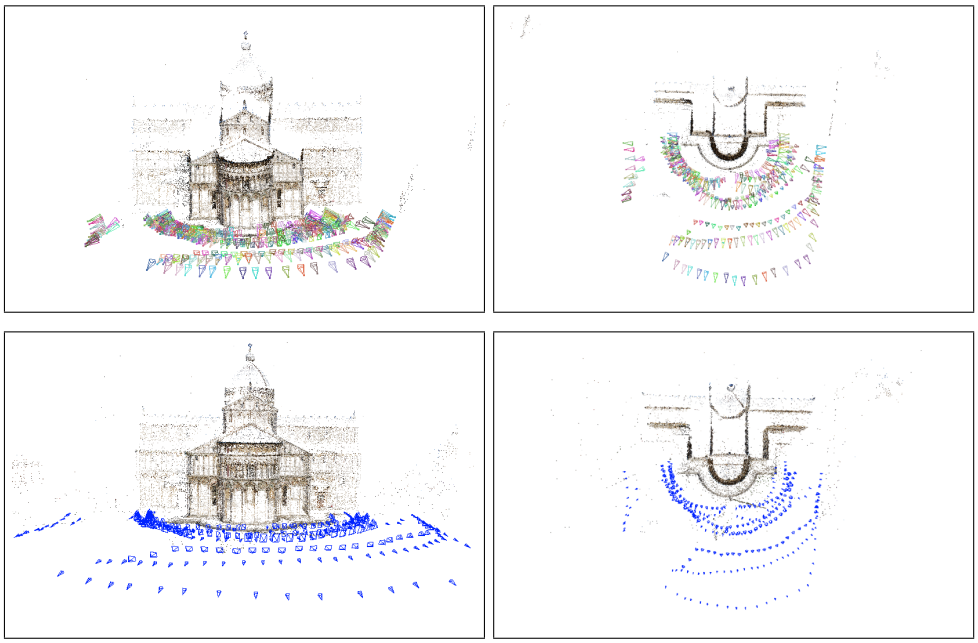


Figure 2.8. Comparative Structure from Motion Results from "Duomo di Pisa" dataset. Top: VisualSfm results, Bottom: Samantha results.

The last tested dataset, "Piazza S. Giacomo", is composed by 270 photos of one of the main square in Udine. The dataset is composed by two set of photos, the first one taken with a Sony DSC-H10 camera with a fixed focal length of 6mm and the second one taken with a Pentax OptioE20 camera with a fixed focal length of 6 mm. Results are shown in Figure 2.9. In this case, Samantha outperforms VisualSfm as the second one produced an incorrect reconstruction. In fact, while Samantha produced an optimal result, with VisualSfm some cameras appear out of the square and some walls were reconstructed in an incorrect way.

2.9 Final Remarks

In this chapter a complete structure from Motion pipeline has been presented. The hierarchical framework has been demonstrated to best the sequential approach both in computational complexity and with respect to the overall containment of error. In addition, an automatic self-calibration procedure has been seamlessly embedded in the pipeline. This constitutes the first example of uncalibrated Structure from Motion for generic datasets not using external, ancillary information.

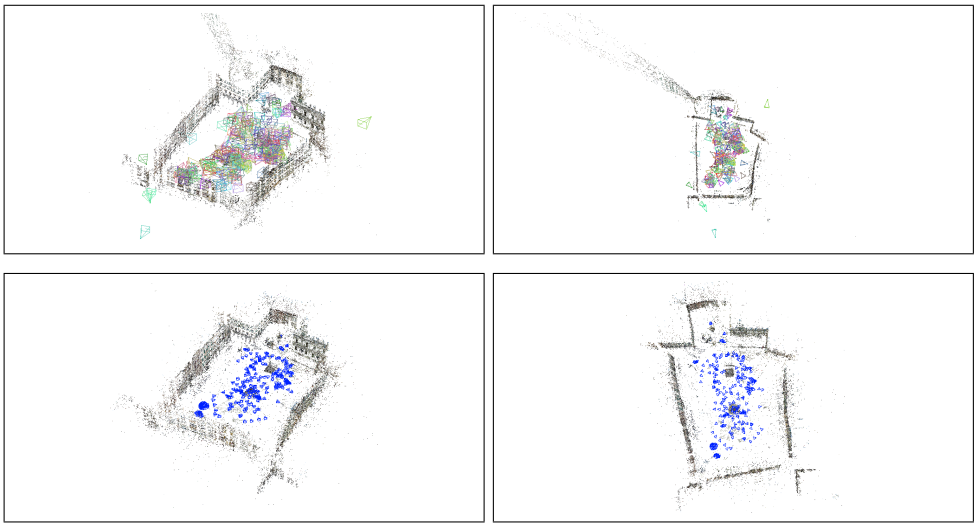


Figure 2.9. Comparative Structure from Motion Results from "Piazza S. Giacomo" dataset. Top: VisualSfm results, Bottom: Samantha results.

The robustness of our approach has been demonstrated on several 3D reconstruction datasets.

The pipeline has been entirely written in C++ language by the Author, with some parts taking advantage of the GPU, and can be downloaded from the web. Future research will continue trying to further improve the computational complexity of our Structure from Motion pipeline. In the next chapters, the output of the SfM pipeline will be used as a base point to extract high level three dimensional models.

3

Multiple View Stereo

The goal of Multi-view Stereo (MVS) is to extract a dense 3D surface reconstruction from multiple images taken from known camera viewpoints. This is a well studied problem with many practical and industrial applications. Laser scanners yield to very accurate and detailed 3D reconstructions. However, they are based on expensive hardware, difficult to carry and rather complex to set, especially for large-scale outdoor reconstructions. In all these cases, MVS can be applied successfully.

A novel method to perform multi-view stereo will be presented in this chapter. The algorithm is focused on accuracy, speed and scalability. In addition, it can be seamlessly integrated as a back-end of any Structure and Motion pipeline (like the one presented in Chapter 2), since it takes advantage, when available, of the sparse SfM point cloud and its visibility information. The Chapter is structured as follows: in Section 3.1 the most relevant multiview stereo techniques are reviewed, in Section 3.2 the method is presented and in Section 3.3 results are shown on challenging and real datasets. Finally in Section 3.4 conclusions are drawn.

3.1 Related Works

The problem of reconstructing a 3D scene from multiple views, have been tackled by many researchers. In [143] several multiview stereo algorithms are presented and a taxonomy is drawn. According to the authors, six fundamental properties differentiate MVS algorithms: reconstruction algorithm, scene representation, photo-consistency measure, visibility model, shape prior and initialization requirements. We will follow this general taxonomy to present the evolution of multiview stereo algorithms.

According to [143], there are mainly four classes of multiview stereo techniques. In the first one, a surface is generated by the definition of a cost function directly on a 3D volume [144, 185]. Several heuristics can be carried out to extract the surface. Some approaches extract an optimal surface by defining a volumetric MRF [56, 99, 134, 152, 194]. A second class of algorithms is composed by methods that iteratively find an optimal surface by minimizing a cost function. Space carving is a popular technique that falls into this category. An initial conservative surface

is defined to contain the entire scene volume and it is iteratively modelled by carving away portion of volumes considering visibility constraints [102, 153]. The third category is composed by methods that compute a depth map for each view [63, 99, 161]. These depth maps can be merged as a post process stage [117]. The fourth class is composed by algorithms that, instead of performing dense matching for each pixel, extract and match a subset of feature points for each image and then fit a surface to the reconstructed features [108, 167].

The scene can be represented in many ways in a reconstruction pipeline. Many times it is represented by a discrete occupancy function (e.g. voxels) [52, 194] or a function encoding distance to the closest surface (e.g. level sets) [48, 129]. Some algorithms represent the scene as a depth map for each input view [39, 161]. The different depth maps can be merged into a common 3D space at a later stage. Other algorithms make use of polygon meshes to represent the surface as a set of planar polygons. This representation can be used in the central part of a reconstruction pipeline since it is well-suited for visibility computation [53] or it can be computed in the final part, starting from a dense point cloud [81, 93, 195]. Many modern algorithms employ different representation over their reconstruction pipeline. The proposed algorithm, for example, falls into this category.

According to [143], photoconsistency measures may be defined in scene space or image space. When photoconsistency is defined in *scene space*, points or planar polygons are moved in 3D and the mutual agreement between their projections on images is evaluated. This can be done by using the variance [102, 144] or a window matching metric [80, 88]. This is the prevalent case and also the proposed algorithm implicitly works in scene space. Normalized cross correlation is the most commonly used window based matching metric. In contrast, *image space* methods use the scene geometry to create a warping between images at different viewpoints. The photoconsistency measure is given by the residual between the synthetic and original views [129, 162]. Some algorithms use also silhouettes [53, 80, 152] or shadows [138] to enhance the reconstruction. However, this techniques are very sensitive to light changes.

A visibility model is used to specify which views to consider when matching regions or pixels using photo consistency. A common approach is to use a predicted estimation of the geometry to determine the visibility model [52, 80, 101, 102, 152, 194]. Other methods simply use clusters of nearby cameras [80, 138], or employ different heuristics to detect outliers views and do not consider them during reconstruction [39, 63, 80, 92]. Instead, the proposed method exploits the robust visibility information coming from a structure and motion pipeline.

Shape priors are often implicitly or explicitly imposed to the generated surface in order to bias the reconstruction to have desired characteristics. Some methods search for a *minimal surface* either by imposing to start from a gross initial shape, by smoothing points with high-curvatures [38, 152, 166] or by imposing planarity [53, 57]. Other methods implicitly search for a *maximal surface* since they does not impose any surface smoothness [52, 101, 102, 136, 144, 185]. Finally some approaches optimize an image-based smoothness terms [16, 63, 92, 99, 161]. This kind of prior fits nicely into 2D MRF solvers.

In addition to a set of calibrated images, many algorithm require additional information on the scene to bound the reconstruction. Usually this is done by

defining a rough bounding box [16, 101, 102, 194] or by simply limiting the range of disparity values in *image space* methods [63, 99, 161].

In recent years, many algorithms have shifted the focus on large scale reconstructions. This is a challenging problem, since dealing more data leads to computational and robustness problems. In [81] the large scale reconstruction problem is solved by defining a minimum s-t cut based global optimization that transforms a dense point cloud into a visibility consistent mesh followed by a mesh-based variational refinement that captures small details, smartly handling photoconsistency regularization and adaptive resolution. The computation can be carried on the graphics processing unit (GPU) [195], knocking down the computing times. Multi-view stereo algorithms are well-suited for general purpose GPU programming (GPGPU), since many of their tasks can be executed in parallel. In [58] the large scale problem is solved with a *divide et impera* solution. The collection of photos are decomposed into overlapping sets that can be processed in parallel, and finally merged. The algorithm have been successfully tested on datasets with over ten thousand images, yielding a 3D reconstruction with nearly thirty million points.

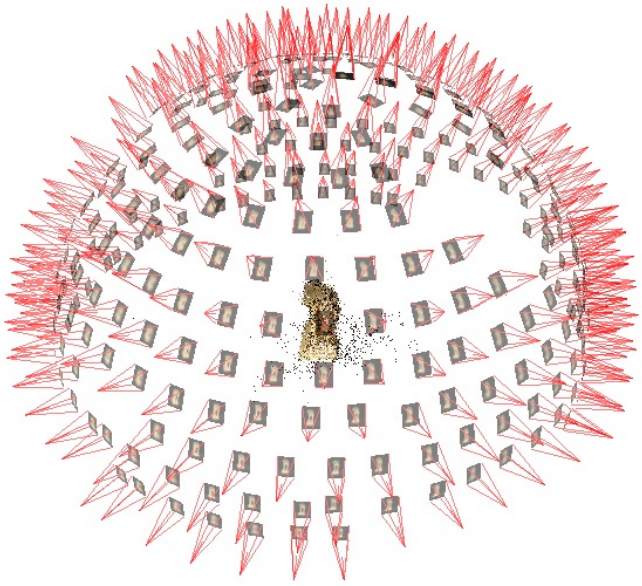


Figure 3.1. An example of structure and motion (cameras) produced by Samantha.

3.2 Method

In this section a novel multiview stereo method will be presented. The algorithm resembles [16] in some parts since it uses a similar MRF depth map optimization at its core. Some parts have been implemented to run on GPU.

The focus of the method is on accuracy and on tight integration with the structure and motion pipeline "Samantha" introduced in Chapter 2, which produces a sparse cloud of 3D *keypoints* (the "structure") and the internal and external parameters of the cameras (the "motion"); see Figure 3.1. The method is completely automatic, as no user input is required.

3.2.1 Extraction of depth hypothesis

The goal of this phase is to extract a number of *candidates depths* for each pixel $\mathbf{m}$ and for each image I_i. These hypothesis will be later used as labels in a MRF that extracts the final depth map $\delta_i(\mathbf{m})$. Similarly to many multiview stereo algorithms, a pixel-level matching along epipolar lines is used, with Normalized Cross Correlation (NCC) as the matching metric, which gives a good tradeoff between speed and robustness to photometric nuisances.

Every depth map is created independently from the others. The extraction of candidate depths is performed by considering the reference image I_i and a number (we used three) of neighboring views $\mathcal{N}(I_i)$. The choice of the near views can be critical. To obtain as much information as possible one should be assured that the neighbour views are viewing the very same part of the scene. This is nearly impossible to estimate without an a-priori knowledge of the scene.

To solve this problem, we leverage on the sparse structure and the visibility information provided by Samantha, with a simple "overlap" measure based on the Jaccard index:

$$d_{\mathrm{J}}(I_1, I_2) = \frac{|\mathcal{V}(I_1) \cap \mathcal{V}(I_2)|}{|\mathcal{V}(I_1) \cup \mathcal{V}(I_2)|} \tag{3.1}$$

where I_1 and I_2 are two images and $\mathcal{V}(I)$ is the set of 3D keypoints visible in image I. By choosing the three views with the highest overlap measure with I_i we are implicitly guaranteed that they are close to I_i and looking at the same part of the scene.

The candidate depths for each pixel are searched along the optical ray, or equivalently, along the epipolar line of each neighbouring image using block matching and NCC. In this way, a correlation profile $C_j(\zeta)$, parametrized with the depth ζ, is computed for every pixel $\mathbf{m}$ and every neighbour image $I_j \in \mathcal{N}(I_i)$.

As suggested by [16] candidates depth correspond to local peaks of the correlation (peaks with a NCC value lower than 0.6 are discarded). In principle:

$$\delta_i(\mathbf{m}) = \arg \operatorname{localmax}_\zeta C_j(\zeta) \quad j \in \mathcal{N}(i) \tag{3.2}$$

where localmax is an operator that returns a fixed number of local maxima. In practice, each of the local peaks of $C_j(\zeta)$ casts a vote (weighted by its score value) on a discrete histogram along the optical ray. At the end of the process, the k bins of the histograms with the highest score are retained (we used $k = 5$) as the

candidate depths for the pixel. These k candidate depths for a point $\mathbf{m}$ are stored in the map δ and the corresponding correlation values in the map γ.

The number of histogram bins can be critical to the accuracy of the algorithm. In order to avoid any loss of fine details, we keep track of the depth inside each bins using a moving average approach, where the weight is given by the match score itself.

Depth Range estimation

The search range of each pixel depth can heavily impact the performance of the algorithm: an effective heuristic to delimit the search range improves both the running times and the candidate depths estimates.

Some algorithms assume the depth range to be known, but this assumption does not hold in many real cases. The search range can be limited by approximating a global surface or independently for each pixel.

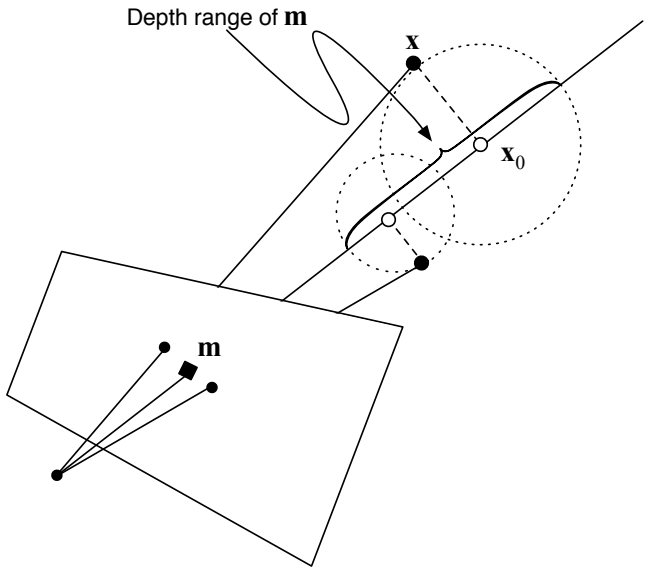


Figure 3.2. The depth range of a pixel $\mathbf{m}$ (black square) is estimated using the closest keypoints (black circles). Every keypoint have a corresponding 3D point $\mathbf{x}$, which projects onto the optical ray of $\mathbf{m}$ at $\mathbf{x}_0$ and produces a depth interval of radius $||\mathbf{x}-\mathbf{x}_0||$ centered at $\mathbf{x}_0$. The union of these intervals is the depth range of $\mathbf{m}$.

A first order approximation is represented by a bounding volume, which can be readily extracted from the SaM point cloud. The intersection of the optical ray with the volume is easy to compute, but the resulting search range on the optical ray can be still too wide.

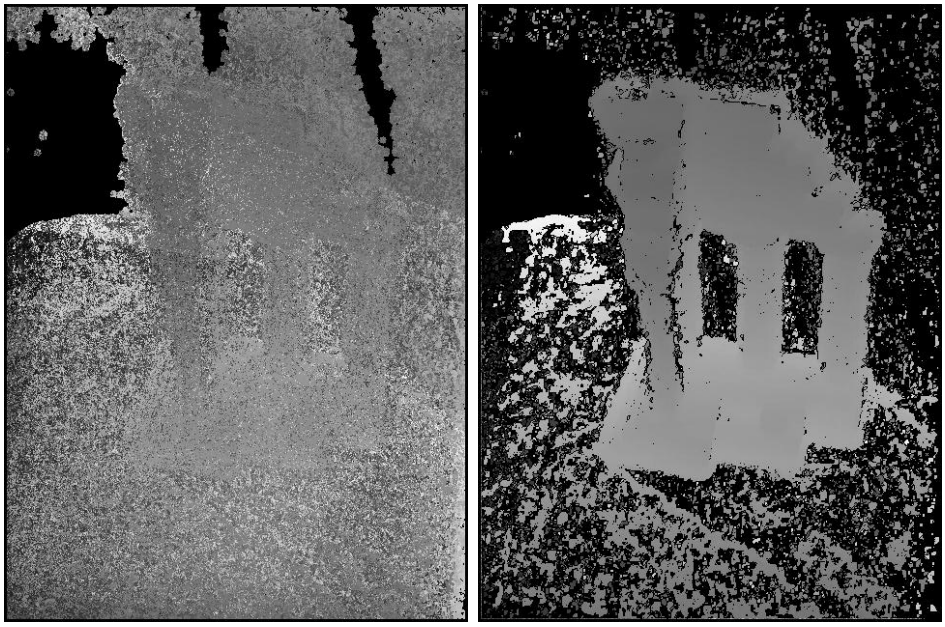
In order to limit further the search range of the candidate depth, we compute the boundary independently for each pixel by using the information coming from

the structure and motion. For each image pixel we consider the five closest keypoints that have been reconstructed by Samantha. Let $\mathbf{x}$ denote the corresponding 3D keypoint position and let $\mathbf{x}_o$ be its projection on the optical ray of the pixel. The search range along the optical ray is defined as the union of the (five) intervals with center in $\mathbf{x}_o$ and radius $||\mathbf{x} - \mathbf{x}_o||$. An example is shown in Figure 3.2.

Rectification

Rectification is a widely used technique in stereo analysis, however it is not very common in the multiple view framework. Given two views, rectification forces the epipolar line to be parallel and horizontal [60]. The idea behind rectification is to define two new camera matrices which preserve the optical centers but with image planes parallel to the baseline. The computation of the correlation window on horizontal lines avoid the need of bilinear interpolation, and lends itself easily to GPU implementation.

The images in the rectified space are linked to the original image space by a 2D homography. As a consequence, any point on the epipolar line is linked by a 1D homography to the same epipolar line expressed in the original space. The general multiview stereo framework is thus unchanged; matching is performed in rectified space and transformed back in original space by means of an homography.



(a) (b)

Figure 3.3. An example of depth map before (a) and after (b) the MRF optimization. The map (a) shows the the candidate depth with the best score.

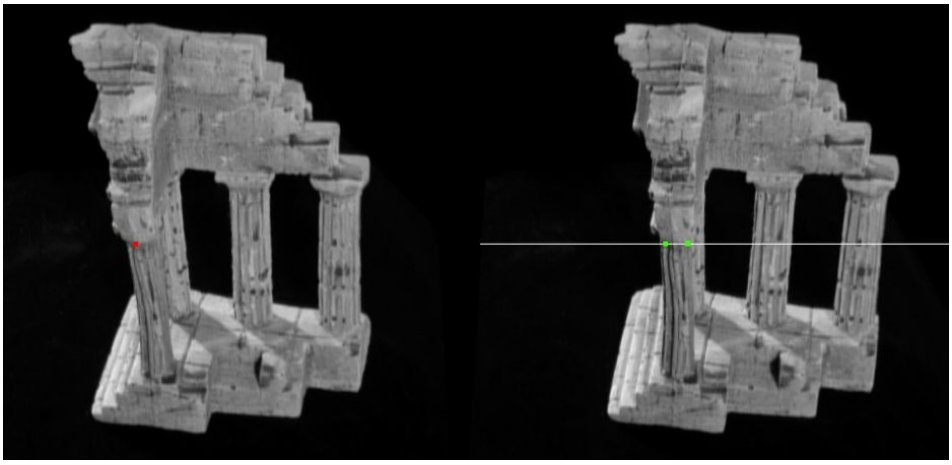


Figure 3.4. An example of rectified images. Match candidates for the red point in the left image are searched along the segment on the horizontal epipolar line bounded by the two green points in the right image.

3.2.2 Depth Map generation

The final depth map is generated from the depth hypothesis using a discrete MRF optimization technique over the (regular) image grid. The MRF assigns a label $l \in \{l_1 \dots l_k, l_{k+1}\}$ to each pixel $\mathbf{m}$, where the first k labels correspond to the candidate depths and l_{k+1} is the *undetermined* state. The cost function to be minimized consist – as customary – of an unary function E_{data} that depend on the value at the pixel and a smoothness term E_{smooth} that depends on pairwise interaction.

The smoothness term is modelled as described in [16], whereas the data term is based on [62]:

$$E_{\text{data}}(\mathbf{m}, l) = 1 - \frac{{}^{l}\gamma_i(\mathbf{m})^2}{{}^{l}\gamma_i(\mathbf{m})^2 + |\mathcal{N}(i)|} \tag{3.3}$$

where ${}^{l}\gamma(\mathbf{m})$ is the NCC peak score of the l^{th} candidate depth for pixel $\mathbf{m}$ — as explained in Section 3.2.1 – and $|\mathcal{N}(i)|$ is the number of neighbouring views of I_i. The undetermined label is given a fixed score of 0.4. With respect to the original formulation, the Geman-McClure score function improve further the robustness against spurious matches.

The MRF is solved with a sequential tree re-weighted message passing optimization [98] that we implemented in CUDA. An example of depth map optimization is shown in Figure 3.3.

3.2.3 Visibility accounting

Depth maps are lifted in 3D space to produce a *photoconsistency* volume φ, represented by an octree that accumulates the scores coming from each depth map δ_i.

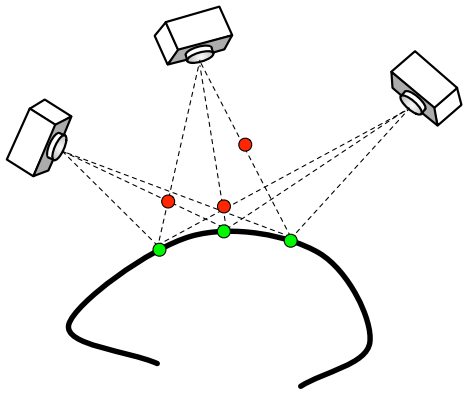


Figure 3.5. Rogue points (red) can be identified as occlusors of actual surface points (green).

In order to avoid any loss of accuracy, a moving average approach have been used inside each bin. At the end of the lifting process, each cell $\mathbf{x}$ contains a 3D point position $\text{pos}(\mathbf{x})$ - which can be shifted with respect to the cell center - and a photoconsistency value $\varphi(\mathbf{x})$ given by the sum of the correlation scores of the points that fall in that bin.

(a) (b)

Figure 3.6. An example of photoconsistency volume (with color) before (a) and after (b) the spurious points removal.

The photoconsistency volume at this stage contains a lot of spurious points, which do not belong to a real surface (see Figure 3.6.a for an example). They are characterized by two features: i) their photoconsistency is generally lower than actual surface points, and ii) they usually occludes actual surface points (Figure 3.5).

This observation leads to an iterative strategy where the photoconsistency of an occlusor is decreased by a fraction of the photoconsistency of the occluded point. Points with negative photoconsistency are eventually removed. The procedure is summarized in Algorithm 2. An example of rogue points removal is shown in Figure 3.6.

Algorithm 2 Visibility Spurious Points Removal

Input: photoconsistency map $\varphi(\mathbf{x})$
Output: photoconsistency map $\varphi(\mathbf{x})$

1. For each image I_i:
   a) Project each point $\mathbf{x}$ s.t. $\varphi(\mathbf{x}) > 0$ on image I_i.
   b) Group projected points in pixel cells and order them by depth.
   c) For each cell:
      i. let $\mathbf{x}_k$ be the point with the highest $\varphi(\mathbf{x})$ visible by image I_i.
      ii. for each point $\mathbf{x}$ occluding $\mathbf{x}_k$:
      $\varphi(\mathbf{x}) \leftarrow \varphi(\mathbf{x}) - \varphi(\mathbf{x}_k)/|\mathcal{V}(\mathbf{x}_k)|$
2. Remove points $\mathbf{x}$ s.t. $\varphi(\mathbf{x}) < 0$.
3. Remove isolated points.
4. Iterate through steps 1,2,3 until no more points are removed.

Algorithm 3 Multiview stereo

Input: N images $I_1 \dots I_N$
Output: photoconsistency map $\varphi(\mathbf{x})$

1. Initialize $\varphi(\mathbf{x}) = 0$
2. For each image I_i, build the depth map δ_i as follows:
   a) for each point $\mathbf{m} \in I_i$,
      i. for each I_j with $j \in \mathcal{N}(i)$ (neighborhood of I_i)
      ii. compute $C_j(\zeta)$, the NCC of $\mathbf{m}$ along its epipolar line in I_j,
      iii. compute depths candidates for $\mathbf{m}$: $\delta_i(\mathbf{m}) = \arg\operatorname{localmax}_\zeta C_j(\zeta)$ with $j \in \mathcal{N}(i)$,
      iv. record the correlation score of the candidates in: $\gamma_i(\mathbf{m}) = C_j(\delta_i(\mathbf{m}))$ for some j.
   b) assign a unique depth to every point of I_i, by MRF relaxation of the depth map δ_i, and update γ_i accordingly.
3. For each depth map δ_i, lift it in 3D space as follows
   a) for each point $\mathbf{m} \in \delta_i$,
      i. $\varphi(\mathbf{x}) = \varphi(\mathbf{x}) + \gamma_i(\mathbf{m})$ where $\mathbf{x}$ is the point at depth $\delta_i(\mathbf{m})$ along the optical ray of $\mathbf{m}$ in I_i.
4. Remove spurious points using visibility (Algorithm 2),
5. Compute approximate normal at each point,
6. Run Poisson surface reconstruction.

At the end of the process a surface is generated using the Poisson algorithm [93]. A normal for each 3D point is computed by fitting a plane using the closest neighbors. Normal direction is disambiguated with visibility.

Finally a texture map is built by wisely making use of both the visibility and the view angle information.

The overall procedure is summarized in Algorithm 3. As an example, Figure 3.7 shows the reconstructed surface of a marble artifact from 7 pictures of size ≈ 15 Mpixels.

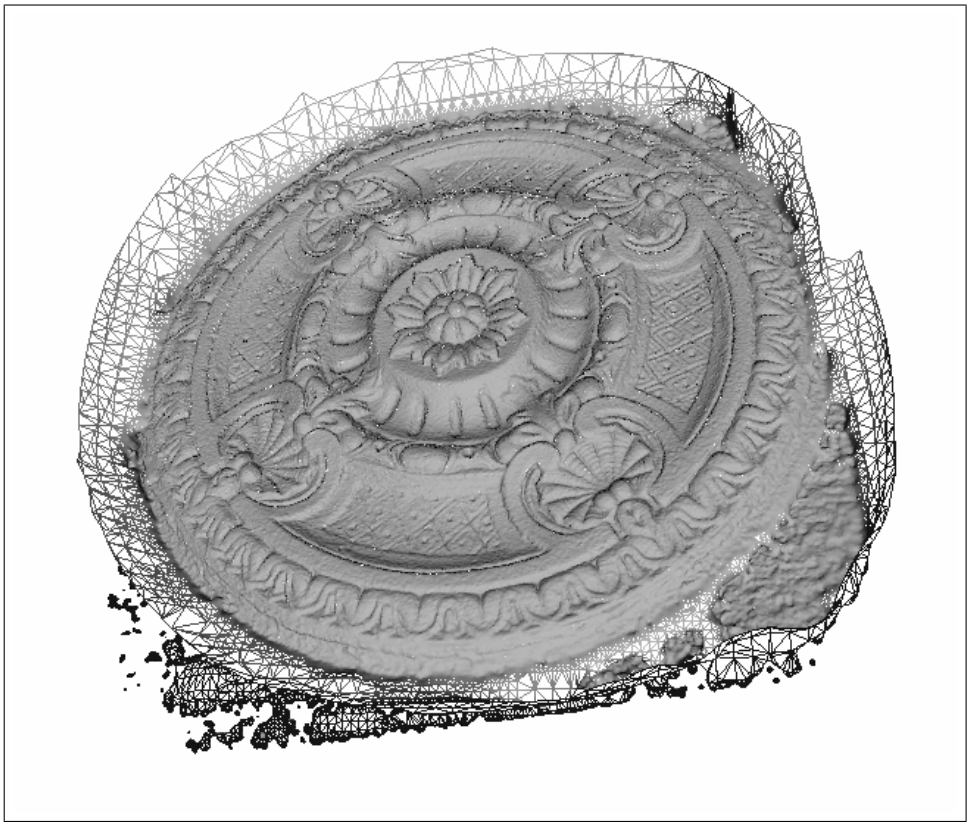


Figure 3.7. Reconstructed surface of a marble artifact. Please observe how the fine details are reproduced.

3.3 Results

In order to test the proposed algorithm, it has been run on several real-cases datasets from the MVS benchmark presented in [159]. The datasets are public and can be downloaded from the author website[1]. They are composed of challenging

[1] http://cvlab.epfl.ch/~strecha/multiview/denseMVS.html

urban scenes. In particular, a fountain, a church gate and a yard have been captured. The number of images ranges from a minimum of 8 to a maximum of 30, the size being ≈ 6 Mpixels.

The results are reported in Figure 3.8. Although camera matrices were already available, we run Samantha with known internal parameter to produce structure and camera matrices.

From a qualitative point of view, the method matches the best results of the benchmark. Unfortunately, at the time of writing, the benchmarking service was not available, so we cannot show a quantitative comparison.

The running time of the method is linear with respect to the number of views. On the average, it took about 10 minutes to generate candidate depth hypothesis for each depth map and 30 seconds to compute the MRF optimization. The rogue points removal based on visibility took from 5 to 10 minutes to compute, while the mesh generation with Poisson took from 10 to 20 minutes. All the experiments were carried out on a entry level machine equipped with a Quad-Core 3.2Ghz CPU and a GeForce GTX 460 GPU.

3.4 Final Remarks

In this chapter a novel multiview stereo algorithm that fully takes advantage of the output of a structure and motion pipeline has been presented. The experiments carried out showed the effectiveness of the method with real cases. Future developments will aim to knock down the computing times by moving more computation on the GPU (specifically, the stereo correlation) and to develop of specific a detail-preserving surface generation algorithm.

In the next Chapter a registration algorithm that can be applied to any set of dense point clouds will be presented, while in Chapter 5 it will be shown how descriptors can be attached to meshes for identification and retrieval tasks.

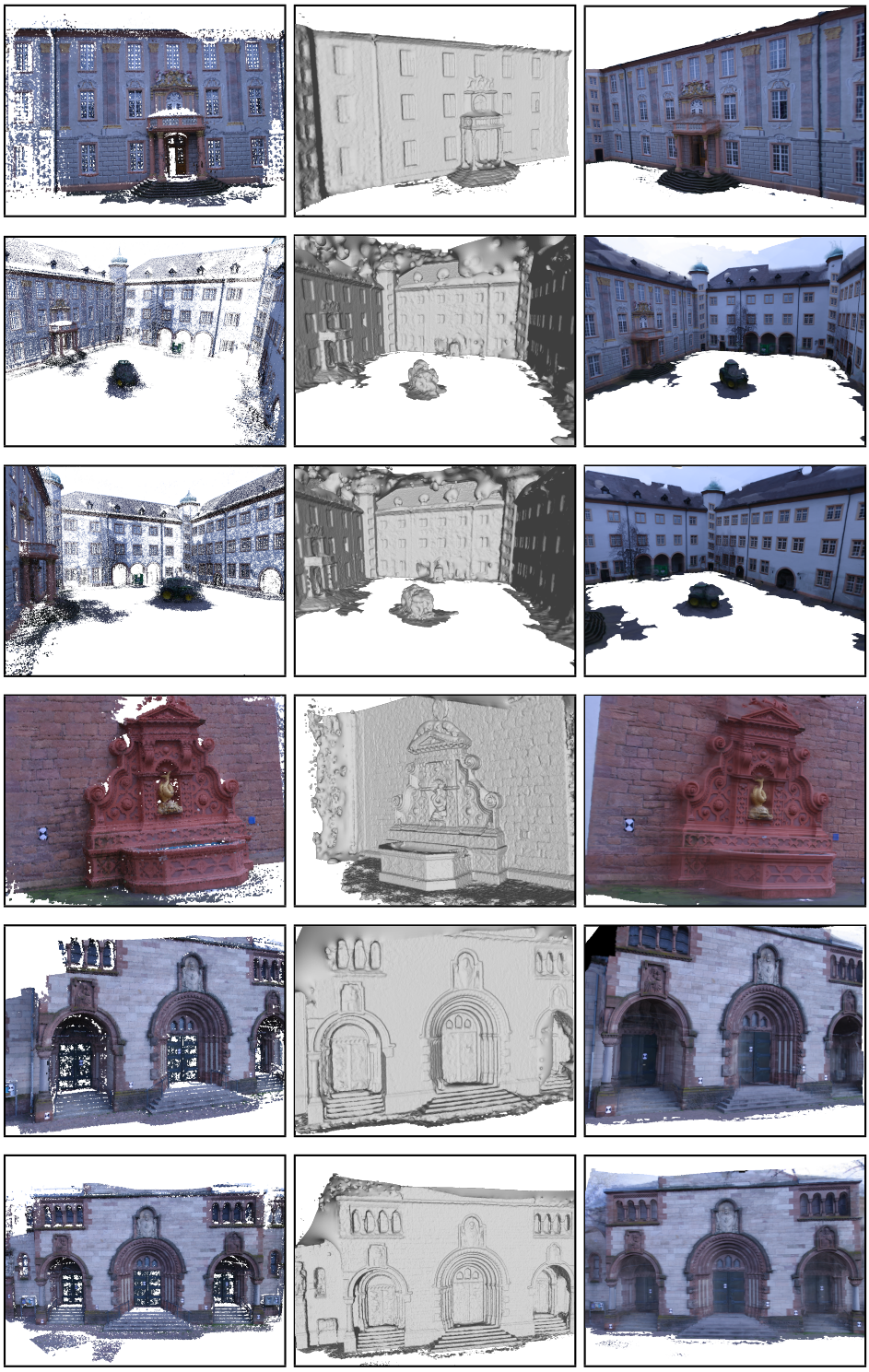


Figure 3.8. Multiview Stereo Results. From left to right, Stereo Points, Shaded Surfaces and Textured Surfaces are reported. The datasets are, from top to bottom, Entry-P10, Castle-P19, and Castle-P30, Fountain-P11, Herz-Jesu-P8, and Herz-Jesu-P25.

4

Global Multiple View Registration

Extracting a model of a real object from tree-dimensional surface measurements is a common task with several applications in various contexts of computer vision, computer graphics, reverse engineering and digital photogrammetry. In most cases, a single view is not sufficient to describe the entire object and multiple acquisitions of the surface are necessary. Typically the views are obtained from multiple scanners or from a single scanner stationed at different locations and orientation or from different runs of a multiview stereo approach like the one presented in Chapter 3. Each view is usually represented as a dense point cloud and the views are partially overlapping each other. When multiple views are present, a problem of surface registration arises, i.e. the different views have to be located in a common reference system. In other words, the objective is to determine a rigid transformation for each partial surface view, in order to find the optimal alignment among all of them. In real cases, the point-to-point correspondences for the overlapping regions are unknown and the view sequence, i.e. the neighborhood of each view, is not always available.

If the correspondences among the views are known, Generalized Procrustes Analysis (GPA) could be employed in order to estimate each transformation in one step [34]. In this chapter a novel multiple view registration technique, that makes use of the GPA in Iterative Closest Point (ICP) framework, is presented. The concept of *mutual correspondence* is introduced to automatically define the matches at each step. The proposed solution has several advantage over existing sequential approaches: it is completely automatic and neither requires the prior knowledge of the view sequence nor of the correspondences, it is simple, theoretically sound and computationally efficient.

This Chapter is structured as follows: in Section 4.1 the current state of the art is reported, in Section 4.2 the GPA is overviewed, in Section 4.3 the method is presented, in Section 4.4 a weighted variant is proposed while in Section 4.5 several experiments are shown, comparing our approach with a classical sequential ICP.

4.1 Related Works

When only two-views are present, we are required to find the transformation that minimizes the distance between the two point clouds. Two-views registration is a well studied problem in the literature. The Iterative Closest Point (ICP) algorithm is the most common solution [6]. It iteratively revises the transformation - usually composed of scale, rotation and translation (SRT) - needed to minimize the distance between the points of the two clouds. The point correspondances are extracted considering the closest point in the other view. Several variants have been proposed to improve either the speed or the robustness of the method, e.g. employing a kd-tree, thresholding the maximum point to point correspondence distance, or weighting every correspondence with a similarity measure. An excellent review about the existing variants can be found in [135].

Multiple view registration is a more complex problem. There are two strategies towards solving the problem, local (sequential) registration and global (simultaneous) registration. The sequential registration approach involves the sequential alignment of two overlapping views at a time. This commonly used approach is not optimal because errors can accumulate and propagate. Moreover the view sequence has to be known or manually specified. On the other hand, global registration attempts to align all scans at the same time by distributing the registration errors evenly over all the overlapping views.

Global multiple view registration schemes were considered by several researchers. One of the first method was proposed in [124] where the "mean rigid shape" of a set of possibly incomplete tuples in any dimension is computed and revised at each step. The method is simple and iterative, but does not include a matching algorithm and the correspondences were manually specified. A comparative study of similar multiple view registration schemes was performed by Cunnington and Stoddart [36]. In [130] a method that first aligns the scans pairwise with each other and then uses the pairwise alignments as constraints in a multiview step has been presented. The aim is to evenly diffuse the pairwise registration errors, but the method itself is still based on pairwise alignments. In [198] an optimization in the transformations matrix space is derived. Point matches are found using a closest point relation for each subsequent view.

More recently, in [18] a novel global registration method, that distributes registration errors evenly across all views, has been proposed. It works in the transformation space, however the sequence of the views is required. In [100] a technique that involves a manifold optimization approach has been presented. It performs an explicit optimization on the manifold of rotations. The algorithm shows a fast convergence, but it does not include a matching strategy and point to point correspondences are assumed to be known. A technique that derives a bayesian formulation of the registration have been introduced in [84]. It aims to register the partial views to a final reconstructed watertight surface.

4.2 Overview of the Generalized Procrustes Analysis

The Orthogonal Procrustes (OP) problem [141] is an optimization problem which solution gives the orthogonal transformation matrix R between two $p \times k$ dimen-

sional matrices A and B that minimizes the least squares of the residuals:

$$E = AR - B \tag{4.1}$$

The problem has been extended in [142] where an unknown rotation R, an unknown translation t and an unknown scale factor c were introduced. This method is often referred as an Extended Orthogonal Procrustes (EOP). The residuals to minimize are the following:

$$E = cAR + jt^T - B \tag{4.2}$$

where j is a $(p \times 1)$ unit vector, t is a $(k \times 1)$ dimensional translation vector and c is a scale factor. Weight matrices, both for components and points, can be added to the EOP model, giving rise to the Weighted Extended Orthogonal Procrustes Analysis (WEOP). The residuals to minimize become:

$$tr((cAR + jt^T - B)^T W_P (cAR + jt^T - B) W_K) = e \tag{4.3}$$

where W_P and W_K are respectively $(p \times p)$ and $(k \times k)$ dimensional weight matrices. A direct solution can be inferred if $W_K = I$, otherwise an iterative solution must be applied. Generalized Procrustes Analysis (GPA) is a well-known technique that provides a least-squares solution when more than two model points matrices are present [9, 40, 70, 72, 169]. It minimizes the following least squares objective function:

$$\begin{aligned} tr(\sum_{i=1}^{m} \sum_{j=i+1}^{m} ((c_i X_i R_i + jt_i^T) - (c_j X_j R_j + jt_j^T))^T \\ ((c_i X_i R_i + jt_i^T) - (c_j X_j R_j + jt_j^T))) = e \end{aligned} \tag{4.4}$$

where $X_1, X_2, \ldots, X_m$ are m model points matrices, which contain the same set of p points in k dimensional m different coordinate systems. The GPA problem has an alternative formulation. Said $X_i^p = c_i X_i R_i + jt_i^T$, the following measures:

$$\sum_{i<j}^{m} ||X_i^p - X_j^p||^2 = \sum_{i<j}^{m} tr((X_i^p - X_j^p)^T (X_i^p - X_j^p)) \tag{4.5}$$

$$m \sum_{i<j}^{m} ||X_i^p - K||^2 = m \sum_{i<j}^{m} tr((X_i^p - K)^T (X_i^p - K)) \tag{4.6}$$

are perfectly equivalent [9], where K is the unknown geometrical centroid. Therefore Equation 4.6, instead of Equation 4.5, can be minimized in order to determine the unknowns $c, R, t_i (i = 1...m)$. The Matrix

$$K = \frac{1}{m} \sum_{i=1}^{m} X_i^p \tag{4.7}$$

corresponds to the least squares estimation of the centroid. As suggested in [34] the solution of the system can be found iteratively. First the centroid K is initialized. At each step a direct solution of the transformation parameters of each

model points matrix A_i with respect to the centroid K is found by means of a WEOP solution. After the update, a new centroid can be estimated. The procedure continues until global convergence, i.e. the stabilization of the centroid K. In real applications all of the p points may not be visible in all of the model points matrices $X1, X2, ..., X_m$. In order to cope with the missing point case, Commandeur [29] proposed a method based on the association of a diagonal binary $(p \times p)$ matrix M_i, in which the diagonal elements are 1 or 0, according to the existence or absence of the point in the i-th model. This solution can be considered as zero weights for the missing points. Least squares objective function is defined as follows in the missing point case:

$$m \sum_{i<j}^{m} ||X_i^p - K||^2 = m \sum_{i<j}^{m} tr((X_i^p - K)^T M_i (X_i^p - K)) \tag{4.8}$$

with:

$$K = (\sum_{i=1}^{m} M_i)^{-1} (\sum_{i=1}^{m} M_i (c_i X_i T_i + j t_i^T)) \tag{4.9}$$

In order to obtain a more general scheme, one should consider the combined weighted/missing point solution. The weight matrix P_i and the binary matrix M_i can be combined in a product matrix $D_i = M_i P_i = P_i M_i$. The corresponding least squares objective function becomes:

$$m \sum_{i<j}^{m} ||X_i^p - K||^2 = m \sum_{i<j}^{m} tr((X_i^p - K)^T P_i M_i (X_i^p - K)) \tag{4.10}$$

with:

$$K = (\sum_{i=1}^{m} P_i M_i)^{-1} (\sum_{i=1}^{m} P_i M_i (c_i X_i T_i + j t_i^T)) \tag{4.11}$$

4.2.1 Overview of the Iterative closest point algorithm

iterative Closest Point (ICP) is an algorithm employed to minimize the difference between two point clouds [203]. It iteratively revises the SRT transformation needed to minimize the distance between the points of the two clouds.

Algorithm 4 Iterative Closest Point

Input: two point clouds X_1 and X_2.
Output: refined transformation from X_1 to X_2.

1. Associate points using the nearest neighbor criteria.
2. Estimate transformation parameters using a mean square cost function.
3. Transform the points using the estimated parameters.
4. Iterate until global convergence or for a fixed number of steps.

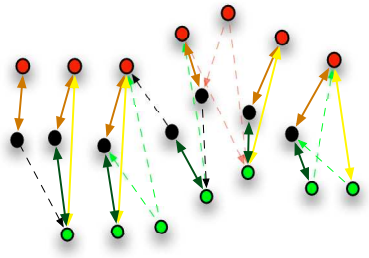
The classical Iterative Closest Point paradigm, makes use of the the EOP as mean square cost function to estimate the transformation parameters at each step. Some variants make use of the WEOP to integrate additional weights for each correspondence [135], based on the compatibility of each point with its closest neighbor. Various compatibility criteria exists, such as the difference of normal, curvature or color.

4.3 Generalized Iterative Closest Point algorithm

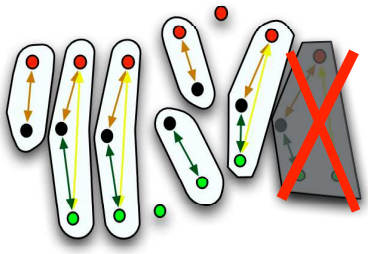
In this section it will be explained how to take advantage of the previously described GPA to define an iterative closest point algorithm that can register multiple views in a simultaneous way. The theoretical result presented in [9], where Equation 4.5 and 4.6 were demonstrated to be perfectly equivalent, gives an important hint on the problem resolution. We are not required to use all the pairwise combinations, thus avoiding ordering or computational issues. The only ambiguity to solve is the definition of the centroid K at every step of the algorithm, so that the transformation of each view can be updated according to the new centroid position. Unlike the algorithm introduced in [34], where the correspondences where manually specified, in an ICP framework matches are not fixed and are assigned at every step by finding the closest neighbor of each point. In general, not every point will be visible by every view. In the following Equation 4.10 and Equation 4.11 will be considered with uniform weights, i.e. $P_i = I$.

Our proposed solution stems from the observation of considering only at a subset of compatible matches, as also suggested in [135]. Instead of fixing a threshold on the distance for the compatible matches, we take into account only matches of points, belonging to different views, that are *mutually nearest neighbor*. We say that a point $x \in A$ is mutually nearest neighbor to another point $y \in B$ if the closest neighbor $\in B$ of x is y and the closest neighbor $\in A$ of y is x (Figure 4.1(a)). This is a simple yet powerful rule to discard matches that are very far without introducing any threshold. It is in fact more probable that the *mutual nearest neighbor relation* is verified for closer matches.

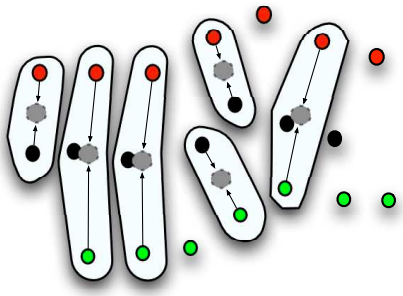
One could think to build an imaginary graph, considering every pair of points that are in a mutual nearest neighbor relation. The edge of the graph will be represented by the mutual nearest neighbor relation itself. In this way, the graph will be composed by completely independent sets as shown in Figure 4.1(b). Each independent set will be represented by an unique point in K, computed as the centroid of the points of the independent set. Every point of an independent set is matched with the newly computed centroid in K (Figure 4.1(c)). Using these correspondences, we can thus update every transformation from the views to K in a simultaneous way. Points of different views belonging to the same independent set are matched with the same element in K and are thus driven into the same position. It may occurs that more than one point of the same view are present in the same independent set; this is a rare degenerate case and the set is simply discarded.



(a) Nearest Neighbor and Mutual Nearest Neighbor relation.



(b) Independent sets.



(c) Centroids correspondences.

Figure 4.1. Example with three different clouds of bi-dimensional points (red, green and black points). In the Figure (a) an arrow depicts a closest neighbor relation and a double pointed arrow denotes a mutual closest neighbor relation. In the Figure (b) the resulting independent sets are shown. The rightmost independent set contains two points from the green set and is thus discarded. In the bottom Figure (c) the arisen correspondences with the centroid (gray points - elements of K) of each independent set are shown.

Algorithm 5 GPA-ICP

Input: m clouds of points $[X_i...X_m]$.
Output: m RST transformation $[T_i...T_m]$.

1. For every view, find the pairs of points that are *mutually nearest neighbor* of points in other views.
2. Define a new point in K as the centroid of each independent set of mutually nearest neighbors.
3. Estimate the RTS transformation parameters $[T_i...T_m]$ using 4.10.
4. Transform each view using the estimated parameters.
5. Iterate until global convergence or for a fixed number of steps.

4.4 Weighted GPA-ICP

In the previous Section we considered only uniformly weighted correspondences, i.e. $P_i = I$. In the general case, however, it is possible to assign different weights to matches. In fact, at each step a subset of points in a view is put in correspondence with a subset of elements of K, which are computed as the centroids of the independent sets. The natural solution would be to define a different weight for every element of K, that can be derived from a compatibility criteria among the points of each independent set.

Matches are weighted by employing the *Shape Index* (SI), since it represents a concise way to express the type of shape of a region. *Shape Index* [125] is defined as:

$$s = -\frac{2}{\pi} \arctan\left(\frac{k_1 + k_2}{k_1 - k_2}\right) \quad k_1 > k_2 \tag{4.12}$$

where k_1, k_2 are the principal curvatures of a generic vertex. The curvature for a vertex is computed by fitting a quadric surface on the neighboring vertices. The SI is bounded and varies in the interval $[-1, 1]$: a negative value corresponds to concavities, whereas a positive value represents a convex surface. A weight is associated to each element of K and consequently to each correspondence. The weight is then computed by using the Median of Absolute Deviations (MAD) among the points of each Independent Set (IndS):

$$MAD_{IndS} = median_i(|SI_i - median_j(SI_j)|) \qquad i, j \in IndS \tag{4.13}$$

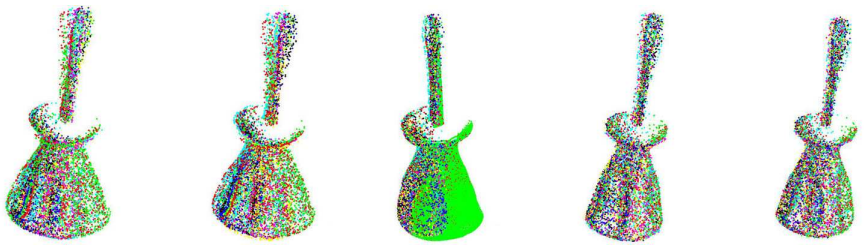
Also the MAD varies in the interval $[0, 1]$, the weight associated to each match is the inverse of the MAD, i.e. $w = 1 - MAD_{IndS}$.

4.5 Results

The proposed approach (ICP-GPA) and its weighted variant (ICP-WGPA) has been tested on 8 different datasets: 4 composed of 10 views and 10000 points per view and 4 obtained by sub-sampling the views to 1000 points. The initial configurations, showed in Figures, were misaligned to some extent. The views come from laser scanning and the subjects are a screw, a dog, a bunny and a dinosaur. The proposed approach has been compared with the classical ICP algorithm applied sequentially for each pair of subsequent views, and a variant where distant matches were rejected according to the X84 rule (ICP-x84) [74]. In this case the order of the views is know, but, unlike for the sequential ICP, is not needed for our approach. The final alignments for the different methods, along with their initial configurations, are shown in Figure 4.2, Figure 4.3, Figure 4.4, Figure 4.5.

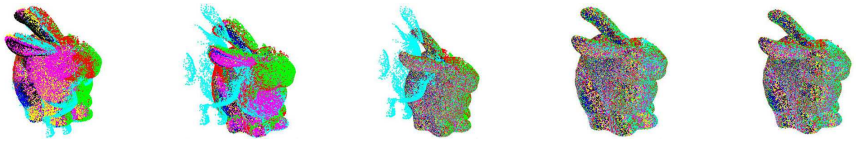
It can be noticed how both the plain and weighted variant of our method converge to a global minima in every example, whereas ICP and its variant ICP-x84 fail in some of them. In particular, ICP without threshold roughly fails with the simplified "screw" and with both "bunny" and "pitbull" experiments, whereas the x84 variant fails with both the "bunny" experiments and with the simplified "pitbull" experiment.

(a) Initial Configuration. (b) Final Configuration ICP. (c) Final Configuration ICP-x84. (d) Final Configuration ICP-GPA. (e) Final Configuration ICP-WGPA.



(f) Simpified mesh initial Configuration. (g) Final Configuration ICP. (h) Final Configuration ICP-x84. (i) Final Configuration ICP-GPA. (j) Final Configuration ICP-WGPA.

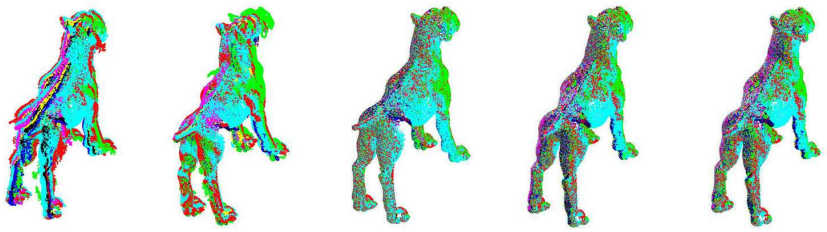
Figure 4.2. "Screw" example.



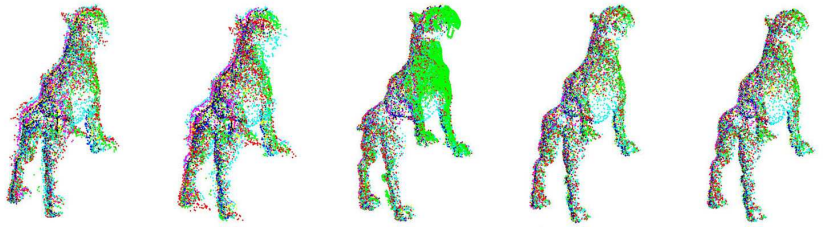
(a) Initial Configuration. (b) Final Configuration ICP. (c) Final Configuration ICP-x84. (d) Final Configuration ICP-GPA. (e) Final Configuration ICP-WGPA.

(f) Simpified mesh initial Configuration. (g) Final Configuration ICP. (h) Final Configuration ICP-x84. (i) Final Configuration ICP-GPA. (j) Final Configuration ICP-WGPA.

Figure 4.3. "Bunny" example.

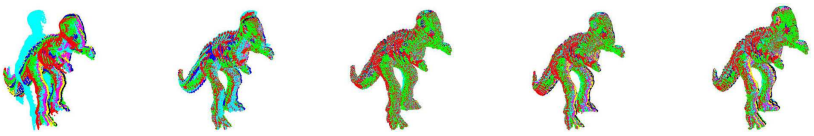


(a) Initial Configuration. (b) Final Configuration ICP. (c) Final Configuration ICP-x84. (d) Final Configuration ICP-GPA. (e) Final Configuration ICP-WGPA.

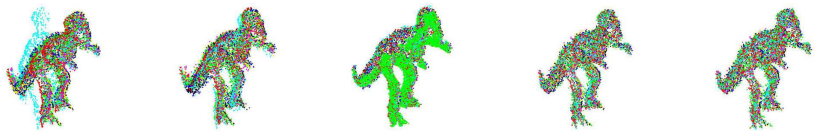


(f) Simpified mesh initial Configuration. (g) Final Configuration ICP. (h) Final Configuration ICP-x84. (i) Final Configuration ICP-GPA. (j) Final Configuration ICP-WGPA.

Figure 4.4. "Pitbull" example.

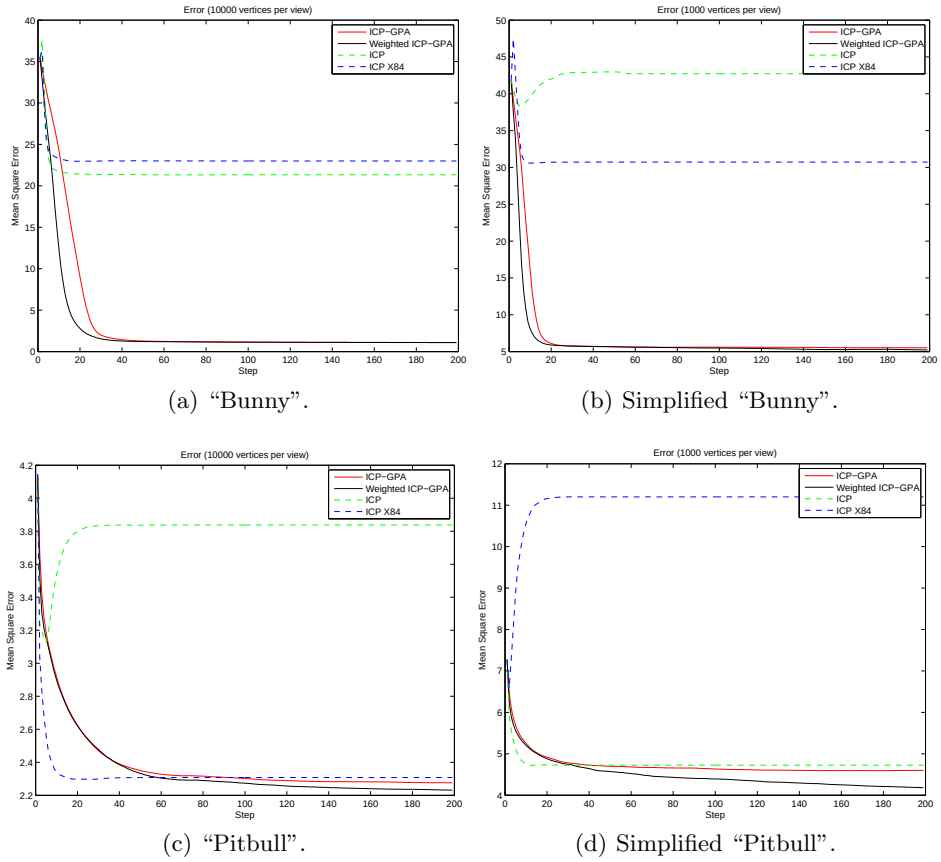


(a) Initial Configuration. (b) Final Configuration ICP. (c) Final Configuration ICP-x84. (d) Final Configuration ICP-GPA. (e) Final Configuration ICP-WGPA.



(f) Simpified mesh initial Configuration. (g) Final Configuration ICP. (h) Final Configuration ICP-x84. (i) Final Configuration ICP-GPA. (j) Final Configuration ICP-WGPA.

Figure 4.5. "Dino" example.



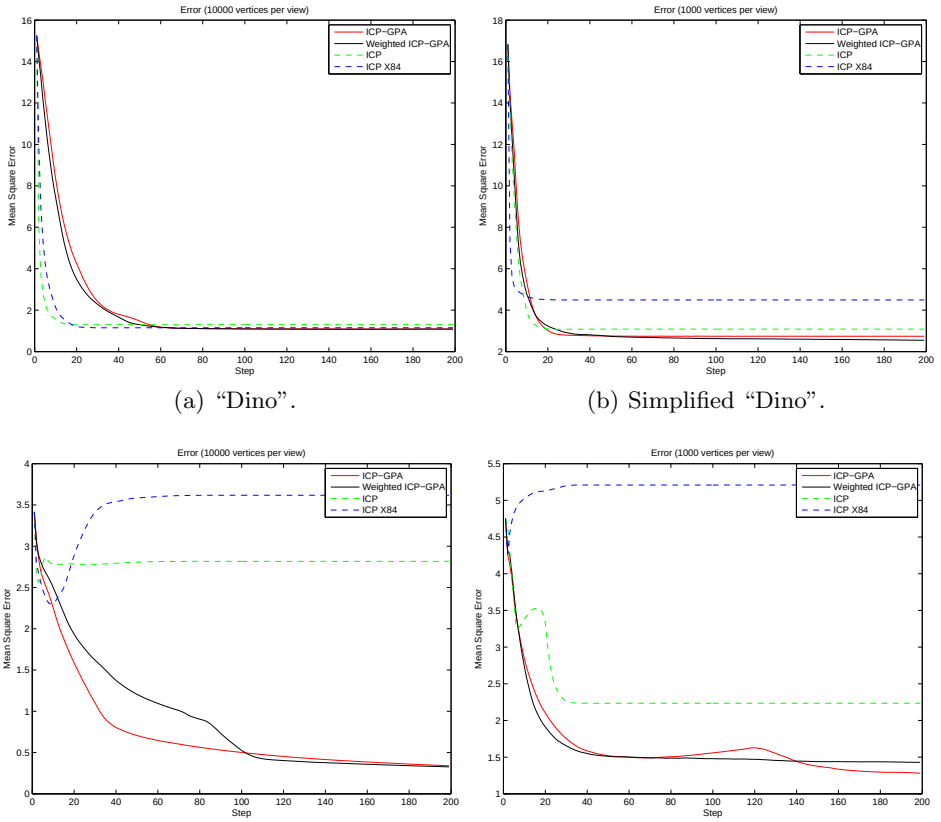
(a) "Bunny". (b) Simplified "Bunny".

(c) "Pitbull". (d) Simplified "Pitbull".

Figure 4.6. Residuals at each step of the algorithms. The residual error is computed as the mean square distance between every point of every view and its closest neighbor considering every other view.

We further analyzed the results by computing the residual at each step of the algorithms (Figure 4.6 and 4.7). The residuals are computed as the mean square distance between the points of every view and their closest neighbor considering every other view. The proposed approach and its weighted variant, perform better in every experiments, with the weighted variant showing, with the exception of the "Screw" experiments, a slightly faster convergence rate. In some experiments sequential ICP and its thresholded variant, converge to a solution that degeneratively increases the global residual. Only in a few experiments the global residual is comparable to out approach, but never lower. It should also be noticed that the number of points seems not to affect the effectiveness of the method, since it works well even with the subsampled cloud of points.

The algorithm has been written entirely in matlab language. The running times are comparable with the sequential ICP: it takes about 12 minutes to process



(a) "Dino". (b) Simplified "Dino".

(c) "Screw". (d) Simplified "Screw".

Figure 4.7. Residuals at each step of the algorithms. The residual error is computed as the mean square distance between every point of every view and its closest neighbor considering every other view.

200 steps with 10 views and 10000 points per view, whereas the sequential ICP approach takes about 10 minutes.

4.6 Final Remarks

In this chapter, a novel algorithm for registering multiple views has been presented. The algorithm has a wide range of application and showed superior performances in terms of accuracy and convergence ratio with respect to sequential approaches.

5

Mesh Description and Retrieval

In the last years, there has been a proliferation of large databases of 3D models caused by the increasing improvements of automatic acquisition techniques from images, like the ones presented in the previous chapters, and the growing accessibility of laser scanners. In this chapter a novel retrieval technique for 3D meshes will be introduced. A consistent segmentation algorithm will be defined first; the resulting sub-parts of the mesh will be then used in a *Bag-of-Words* (BoW) of Words framework.

Recently, there has been a surge of interest in methods for content-based object retrieval [14,54,86,165]. One of the major challenges in the context of data retrieval is to elaborate a suitable canonical characterization of the entities to be indexed. In the literature, this characterization is referred to as *descriptor* or *signature*. Since the descriptor serves as a key for the search process, it decisively influences the performance of the search engine in terms of computational efficiency and relevance of the results. Descriptors are *global* or *local*. The former consists in a set of features that effectively and concisely describe the entire 3D model [55]. The latter are instead collections of local features of relevant object sub-parts [149].

In the work presented in this chapter, the BoW approach will be exploited in order to combine and merge local information into a global object signature. The BoW framework has been proposed for textual document classification and retrieval. A text is represented as an unordered collection of words, disregarding grammar and even word order. The extension of such approach to visual data requires the building of a *visual vocabulary*, i.e., the set of the visual analogue of words. For example, in [35] 2D images are encoded by collecting interest points which represent local salient regions. This approach has been extended in [73] by introducing the concept of *pyramid* kernel matching. Instead of building a fixed vocabulary, the visual words are organized in a hierarchical fashion in order to reduce the influence of the free parameters (e.g., the number of bins of the histogram). Finally, in [103] the BoW paradigm has been introduced for human actions categorization from real movies. In this case, the visual words are the quantized vectors of spatio-temporal local features. The extension of the BoW paradigm to 3D objects is non-trivial and has been proposed only in few recent works [104, 105, 122]. In [122] range images are synthetically generated from the full 3D model and subsequently treated as 2D (intensity) images. In [104,105] Spin Images are chosen as

local shape descriptors after sampling the mesh vertices. Usually local techniques are defined by point-based features rather than by segmentation. More recently a part-based retrieval method [146] has been proposed. The method partitions the object into meaningful segments and finds analogous parts in other objects. Very recently [49] proposed an approach that uses a collection-aware shape decomposition combined with a shape thesaurus and inverted indexes to cope with the partial matching problem.

In the proposed approach a 3D visual vocabulary is defined by extracting and grouping the geometric features of the object sub-parts. Thanks to this *part-based* representation of the object we achieve pose invariance, i.e., insensitivity to transformations that change the articulations of the 3D object [61]. In particular, the proposed method is able to discriminate objects with similar skeletons, a feature that is shared by very few other works like [7, 119, 163, 188].

Beside being very effective in object retrieval, the BoW representation proved valuable also in the task of 3D object categorization. In particular we devised a *learning-by-example* approach [42]: Geometric features representing the query-model are fed into a Support Vector Machine (SVM) which, after a learning stage, is able to assign a *category* (or a *class*) to the query-model without an explicit comparison with all the models of the dataset.

In summary, the proposed approach is composed by the following main steps:

Object sub-parts extraction (Section 5.1). Spectral clustering is used for the selection of seed-regions. Being inspired by the *minima-rule* [83], the adjacency matrix is tailored in order to allow convex regions to belong to the same segment. Furthermore, a multiple-region growing approach is introduced to expand the selected seed-regions, based on a weighted fast marching. The main idea consists on reducing the speed of the front for concave areas which are more likely to belong to the region boundaries. Then, the segmentation is recovered by combining the seeds selection and the region-growing steps.

Object sub-parts description (Section 5.2). Local region descriptors are introduced to define a compact representation of each sub-part. Working at the part level, as opposed to the whole object, enables a more flexible class representation and allows scenarios in which the query model is significantly deformed. We focus on region descriptors easy to compute and partially available from the previous step (see [149] for an exhaustive overview of shape descriptors).

3D visual vocabularies construction (Section 5.3). The set of region descriptors are properly clustered in order to obtain a fixed number of 3D visual *words* (i.e., the set of clusters centroids). In fact, the clustering defines a vector quantization of the whole region descriptor space. Note that the vocabulary should be large enough to distinguish relevant changes in object parts, but not so large as to discriminate irrelevant variations such as noise.

Object representation and matching (Section 5.4). Each 3D object is encoded by assigning to each object sub-part the corresponding visual word. The BoW representation is defined by counting the number of object sub-parts assigned to each word. In practice, a histogram of visual words occurrences is built for each 3D object which represent its *global* signature [35]. Matching is accomplished by comparing the signatures.

Object categorization by SVM (Section 5.5). A SVM is trained by adopting a learning by example approach. In particular, a suitable kernel function is defined in order to implicitly implement the sub-part matching.

Finally, the proposed approach has been successfully applied on different applicative scenarios, namely i) 3D object retrieval, ii) partial shape matching, and iii) 3D object categorization.

The presented retrieval and categorization method can also be applied on meshes coming from a typical SfM + MVS pipeline, like the ones presented in Chapter 2 and 3.

5.1 Objects segmentation

The recent survey by [147] and the comparative study by [2] have thoroughly covered the several different approaches developed in literature.

In the following a novel mesh segmentation technique is presented. The method provides a consistent segmentation of similar meshes, depends on very few parameters and is very fast. It is inspired by the *minima rule* [83]: "for the purposes of visual recognition, the human visual system divides 3D shapes into parts at negative minima of principal curvature". The rule suggests to cluster in the same set convex regions and to detect boundary parts as concave ones. A concise way to characterize the shape in terms of principal curvatures is given by the *Shape Index* (SI) [125], introduced in the previous Chapter (Equation 4.12). The SI varies in $[-1, 1]$: a negative value corresponds to concavities, whereas a positive value represents a convex surface.

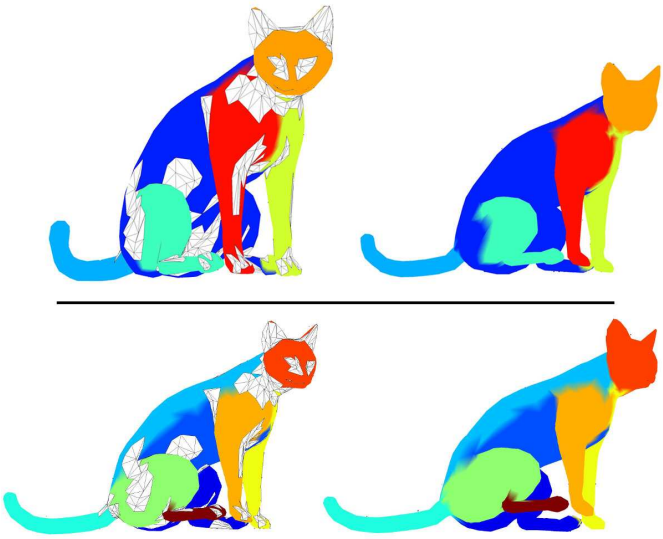


Figure 5.1. An example of segmentation for different number of seeds regions. The starting seeds regions are shown in left, while the final segmentations are shown in right.

The key idea behind the proposed algorithm is the synergy between two main phases: (i) the detection of similar connected convex regions, and (ii) the expansion of these seed-regions using a multiple region growing approach. According to the minima-rule the SI is employed in both phases. An example of extracted seed regions and corresponding segmentation is shown in Figure 5.1.

5.1.1 Seed-regions detection by Spectral Clustering

The extraction of the seed-regions is accomplished with Normalized Graph Cuts [148]. This approach has been firstly applied to image segmentation although it is stated as a general clustering method on weighted graphs. In our case, the weight matrix $w(x_i, x_j)$ is built using the SI at each vertex:

$$w(x_i, x_j) = e^{-|SI(x_i)-SI(x_j)|} \tag{5.1}$$

where the vertices with negative SI – i.e., those corresponding to concave regions – have been previously discarded. In this way we cluster together vertices representing the same convex shape.

The number of clusters, needed by the Spectral clustering approach, is linked, but not equal, to the number of final segments. Indeed, clusters are not guaranteed to be connected in the mesh. This happens because we do not take into account geodesic distance information at this stage: we cluster only according to the curvature value at each vertex. Hence, we impose connection as a post-processing step: the final seed regions are found as connected components in the mesh graph, with vertices belonging to the same cluster.

5.1.2 Multiple region growing by weighted fast marching

Once the seed regions are found, we must establish a criteria to assign the vertices that don't belong to any initial seed region. The key idea is to expand the initial seeds region using a *weighted* geodesic distance. Again, the weight at each vertex is chosen according to the minima-rule. In formulae, given two vertices $x_0, x_1 \in V$, we define the *weighted geodesic distance* $d(x_0, x_1)$ as

$$d(x_0, x_1) = min_\gamma \left\{ \int_0^1 \|\gamma'\| w(\gamma(t)) dt \right\} \tag{5.2}$$

where $w(\cdot)$ is a weight function (if $w(\cdot) = 1$ this is the classic geodesic distance) and γ is a piecewise regular curve with $\gamma(0) = x_0$ and $\gamma(1) = x_1$. Our weight function is again based on the Shape Index SI:

$$w(x) = e^{\alpha SI(x)} \tag{5.3}$$

where α is an arbitrary constant. An high α value heavily slow down the front propagation where the concavity are more prominent. In our experiments we used a fixed $\alpha = 5$.

An example segmentation along with the starting seed regions is shown in Figure 5.1. Several other examples of segmentation on different objects are shown in Figure 5.2. The reader might notice how similar parts are segmented in a similar manner (provided that the parameters of the segmentations are equal).

Figure 5.2. Examples of segmentation of some objects from the Tosca Dataset.

5.2 Segment descriptors

We chose four type of descriptors to represent each extracted region: the *Shape Index Histogram* (SIH), the *Radial Geodesic Distance Histogram* (RGDH), the *Normal Histogram* (NH) and the *Geodesic Context* (GC). The first three are defined as the normalized histograms of local measures computed for each point of the region, namely shape index, radial geodesic distance and normal. The fourth descriptor depends on the relative positions of the regions and thus it is a context descriptor.

The *radial geodesic distance* measures the geodesic distance of a surface point to the geodesic centroid of the region. In our case, for computation efficiency, we approximate the geodesic centroid as the closest point on the mesh to the Euclidean centroid.

The *Geodesic Context* descriptor for a region is built computing the histogram of the geodesic distance between its centroid and the centroids of the other regions. The *GC* descriptor, defined for regions, resembles the shape context descriptor [5], defined for points.

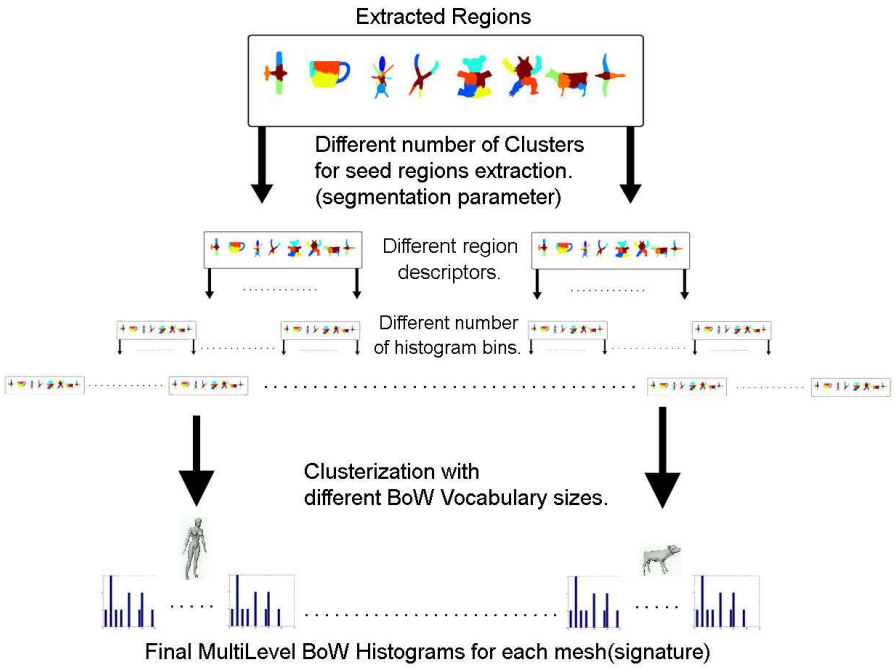


Figure 5.3. The construction of the vocabularies is performed in a multilevel way. At the beginning all regions are extracted for different numbers of seed regions (variable segmentation parameter). For every region, different descriptors are attached. The different region descriptors are divided by the type of descriptor and its number of bins. The final clusterizations are obtained with varying number of clusters. At the end of the process we obtain different Bag-of-Words histograms for each mesh.

Please note that the number of bins chosen for each histogram of the four descriptors is a critical choice. A small number reduce the capability of the region descriptor in discriminating among different segments. On the other hand, a high number increases the noise conditioning. Hence we introduce, for each descriptor, histograms with different number of bins in order to obtain a *coarse-to-fine* regions representation.

5.3 3D visual vocabularies construction

The different sets of region descriptors must be clustered in order to obtain several visual words. Since we start with different segmentations and different types of descriptors, we adopted a multi-clustering approach rather than merging descriptors in a bigger set. Before the clusterization, the sets of descriptors are split in different subsets as illustrated in Figure 5.3. The final clusters are obtained with a k-means algorithm. Instead of setting a fixed free parameter k, namely the number of cluster, we carry out different clusterizations while varying this value.

Once the different clusters are found we retain only their centroids, which are our *visual words*. In Figure 5.4 an example of descriptors subset clusterization with relative distance from centroid is shown. Note that object sub-parts from different categories may fall in the same cluster since they share similar shape.

More in details, at the end of this phase we obtain the set of visual vocabularies $V_s^{d,b,c}$ (i.e., *multiple*-vocabulary), where:

1. s identifies the index of the multi-level 3D segmentation (variable segmentation parameter) :
   $s \in \{6, 10, 14\}$
2. d identifies the region descriptor types :
   $d \in \{SIH, RGH, NH, GC\}$
3. b identifies the refined level of the region descriptor (number of histogram bins):
   $b \in \{15, 30, 50\}$
4. c identifies the refined level of the vocabulary construction (number of clusters):
   $c \in \{50, 100, 150\}$

In this fashion, a total of $3 \times 4 \times 3 \times 3 = 108$ different, non intersecting visual vocabularies are built. It is worth noting that in general, the choice of free parameters (i.e., number o of bins of a histogram, the number of visual words and so on), are a critical issue aiming at finding the best trade-off between generalization and overfitting [42]. Here, the values assigned to s, d, b, c are heuristically chosen in order to obtain a course-to-fine representation which allow each descriptor to be adaptively represented at its best refined level.

5.4 3D representation and matching

In order to define a global signature of a new 3D object, we compute the multiple-vocabulary Bag-of-Words representation. After object segmentation, we compare each region descriptors with the visual words of the corresponding visual vocabulary. More in details, an object is segmented according to the multi-level segmentation procedure, then each region descriptors is assigned to the most similar words for each i) level of segmentation, ii) type of descriptor, iii) refined level of region descriptor, and iv) refined level of vocabulary construction. In practice, for an object a Bag-of-Words representation is obtained by counting the number of segments assigned to each word. This procedure is carried out for each of the 108 visual vocabularies, leading to a very sparse signature. Finally, the objects matching is obtained by comparing their respective signature by using standard metric for histograms.

5.5 Object categorization by SVM

One of the most powerful classifier for object categorization is the Support Vector Machine (SVM) (see [13] for a tutorial). The SVM works in a vector space, hence the Bag-of-Words approach fits very well, since it provides a vector representation

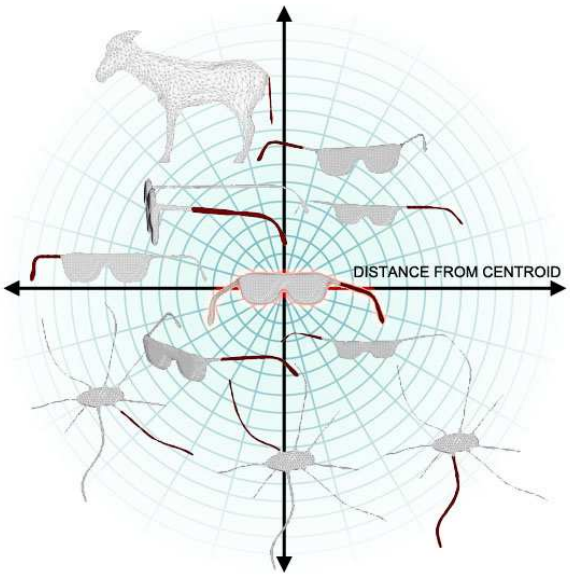


Figure 5.4. Example of a Bag-of-Words cluster for SI descriptors. The centroid is highlighted with red and others region in the same cluster are sorted by distance from centroid. Note that sub-parts of meshes from different categories may fall in the same cluster since they share similar shape.

for objects. In our case, since we work with multiple vocabularies, we define the following positive-semi-definite kernel function:

$$K(A,B) = \sum_{s,d,b,c} k(\phi_s^{d,b,c}(A), \phi_s^{d,b,c}(B)), \tag{5.4}$$

where (A,B) is a pair of 3D models, and $\phi_s^{d,b,c}(\cdot)$ is a function which returns the Bag-of-Words histogram with respect to the visual vocabulary $V_s^{d,b,c}$. The function $k(\cdot,\cdot)$ is in turn a kernel which measures the similarity between histograms h^A, h^B:

$$k(h^A, h^B) = \sum_{i=1}^{c} min(h_i^A, h_i^B), \tag{5.5}$$

where h_i^A denotes the count of the i^{th} bin of the histogram h^A with c bins. Such kernel is called *histogram intersection* and it is shown to be a valid kernel [73]. Histograms are assumed to be normalized such that $\sum_{i=1}^{n} h_i = 1$. Note that, as observed in [73], the proposed kernel implicitly encodes the sub-parts matching since corresponding segments are likely to belong to the same histogram bin. In fact, the histogram intersection function counts the number of sub-parts matching being intermediated by the visual vocabulary.

Finally, since the SVM is a binary classifier, in order to obtain an extension to a multi-class framework, a one-against-all approach [42] is followed.

5.6 Results

In order to prove the effectiveness and the generalization capability of the proposed paradigm we tested it with several different retrieval and categorization tasks, also working with compound or partial mashes. The two datasets employed are the Aim@Shape watertight dataset and the Tosca dataset. The first is composed of 400 meshes of 20 different classes (see Figure 5.5), with a remarkable inter-class variability. The second is composed of 13 shape classes. In each class, the shape underwent different types of transformations, namely: null (no transformation), isometry, topology (connectivity change obtained by welding some of the shape vertices), isometry+topology, triangulation (different meshing of the same shape) and partiality (missing information, obtained by making holes and cutting parts of the shape). In this case difficulties arise because the categories are very similar each other (see Figure 5.6 and Figure 5.7 for more details).

5.6.1 Aim@Shape Watertight

The Aim@Shape Watertight dataset has been used for various retrieval contests [191]. Firstly, we compared our method with the participant of the Aim@Shape Watertight 2007 contest [191] for object retrieval. We used precision and recall to evaluate our results, that are two fundamental measures often used in evaluating search strategies. Recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database, while precision is the ratio of the number of relevant records retrieved to the size of the return vector [137]. In Table 5.2 the precision and recall of our approach along with the results of the other methods are reported, while in Figure 5.8 the precision versus recall plot of our method is shown. The results, divided by category, are shown in Table 5.1. The algorithm fails with some meshes, but the overall rate of success is still fairly good.

Category	**Precision after 20**	**Category**	**Precision after 20**
Human	0.53	Cup	0.46
Glasses	0.90	Airplane	0.73
Ant	0.92	Chair	0.57
Octopus	0.61	Table	0.52
Teddy	0.94	Hand	0.32
Plier	0.99	Fish	0.8
Bird	0.4	Spring	0.96
Armadillo	0.94	Buste	0.57
Mechanic	0.80	Bearing	0.44
Vase	0.8	Four Legs	0.32

Table 5.1. Precision for each category of the Aim@Shape dataset after 20 retrieved items.

In the second task we tested our method with some query test models that are composed of parts of the original dataset. The query test models are 30 and

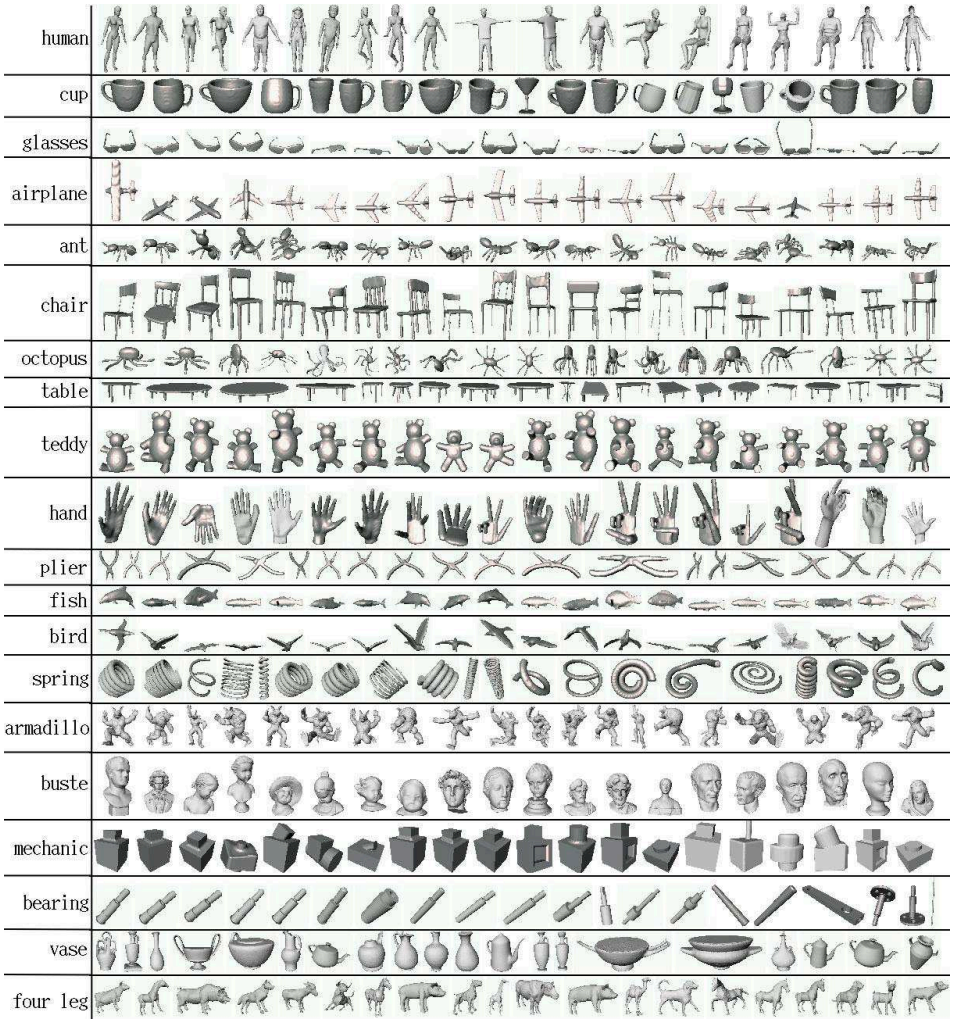


Figure 5.5. Aim@Shape Watertight Dataset

each query model shares common sub-parts with (possibly) more than one model belonging to the ground-truth dataset. The query set is shown in Figure 5.9. Again, we compared our method with the participant of the Aim@Shape Partial Matching 2007 contest [191]. In order to evaluate the performance, a set of highly relevant, marginally relevant and non-relevant models belonging to the dataset has been associated to each query model. The performance indicator used is the Discounted Cumulated Gain vector (DCG) [87], which is recursively defined as

$$DCG[i] = \begin{cases} G[i] & \text{if i} = 1 \\ DCG[i-1] + G[i]\log_2(i)) & \text{otherwise} \end{cases} \tag{5.6}$$

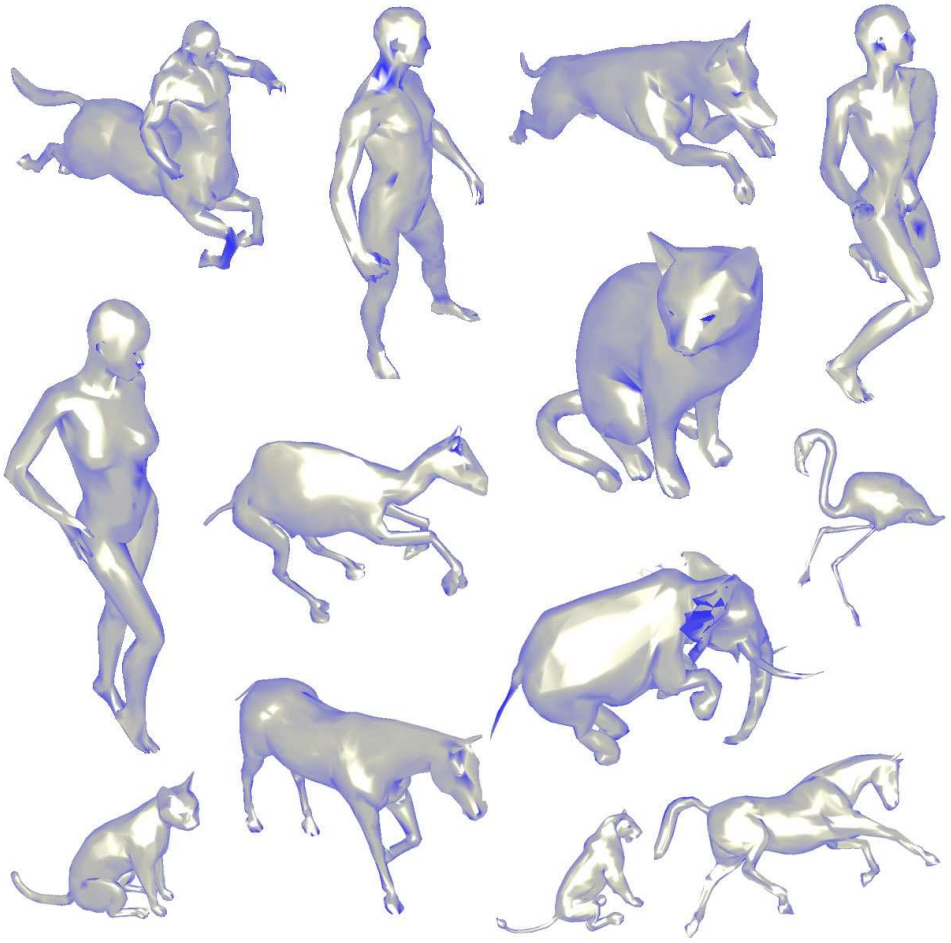


Figure 5.6. Example of different kind of objects in the Tosca dataset. The category are 13, namely: centaur,horse,two males,female,two cats,dog, horse, tiger, elephant, dromedary and flamingo

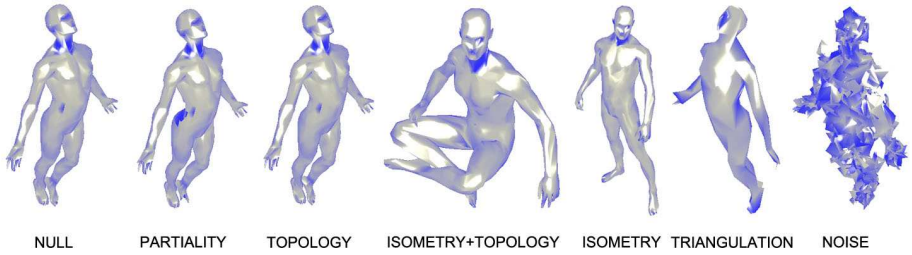


Figure 5.7. Example of different type of transformation in the Tosca dataset.

Precision after	20	40	60	80
Ideal	1	0.5	0.333	0.25
Tung et al.	0.714	0.414	0.290	0.225
Our Approach	**0.648**	**0.379**	**0.270**	**0.210**
Akgul et al.	0.626	0.366	0.262	0.205
Napoleon et al.	0.604	0.366	0.262	0.205
Daras et al.	0.564	0.346	0.252	0.199
Chaouch et al.	0.546	0.329	0.241	0.190
Recall after	20	40	60	80
Ideal	1	1	1	1
Tung et al.	0.714	0.828	0.872	0.902
Our Approach	**0.648**	**0.758**	**0.808**	**0.841**
Akgul et al.	0.626	0.732	0.786	0.821
Napoleon et al.	0.604	0.732	0.788	0.822
Daras et al.	0.564	0.692	0.756	0.798
Chaouch et al.	0.546	0.658	0.724	0.763

Table 5.2. Precision and Recall after 20, 40, 60 and 80 retrieved items for the Aim@Shape dataset.

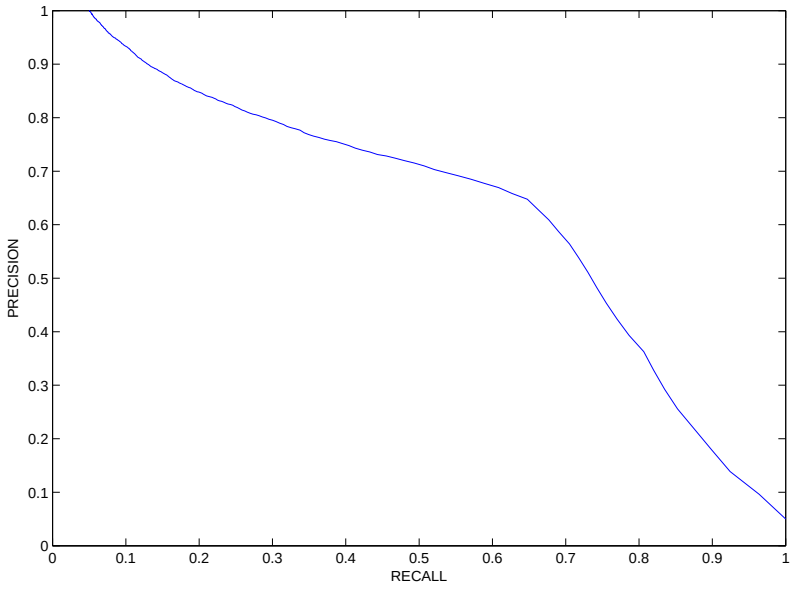


Figure 5.8. Precision vs Recall for the Aim@Shape dataset.

.

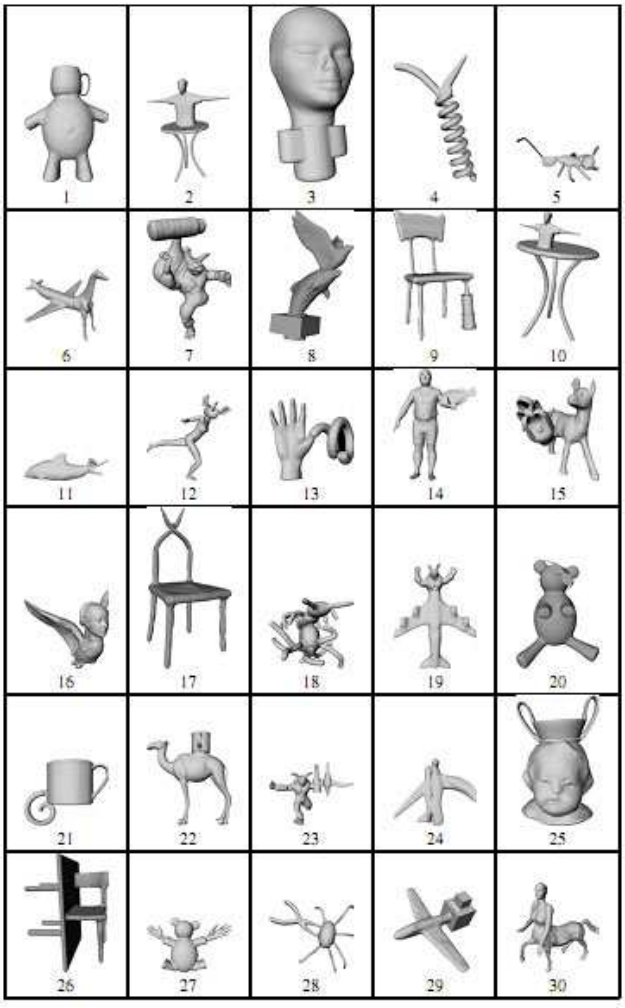
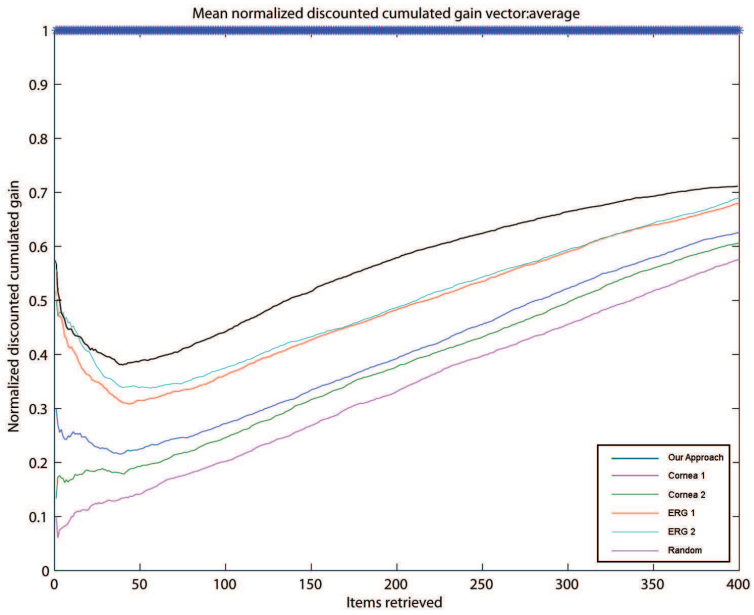


Figure 5.9. Aim@Shape Partial Matching query objects.

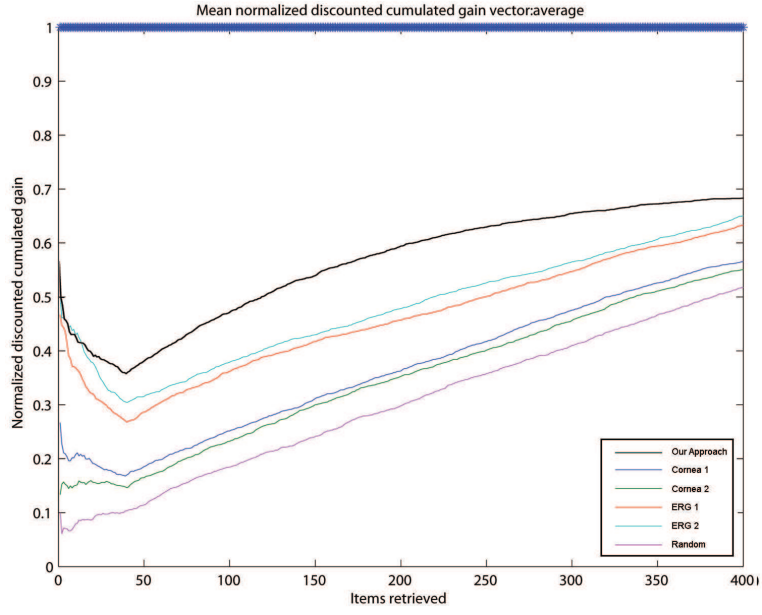
.

where $G[i]$ represents the value of the gain vector at the position i. In our case, for a specific query, $G[i]$ equals 2 for highly relevant models, 1 for marginally relevant models and 0 for non-relevant models. The normalized discounted cumulated gain vector NDCG is obtained by dividing DCG by the ideal cumulated gain vector. In Figure 5.10 the NDCG of our approach along with the results of the other methods are reported. We can notice how our method performs better than the other methods considered.

Finally, we tested the dataset in a categorization problem. We performed the test using a Leave-One-Out approach. The Overall success rate is high: **87.25%**. In Table 5.3 the different results for each category are reported.



(a)



(b)

Figure 5.10. Overall Normalized Discount Cumulated Gain considering only highly relevant models 5.10(a) and both highly relevant and marginally relevant models 5.10(b) for the Aim@Shape partial matching contest.

Category	S.R.	Category	S.R.
Human	0.8	Cup	0.85
Glasses	0.95	Airplane	0.9
Ant	1.0	Chair	0.95
Octopus	0.95	Table	0.8
Teddy	1.0	Hand	0.8
Plier	1.0	Fish	0.85
Bird	0.8	Spring	0.95
Armadillo	1.0	Buste	0.95
Mechanic	0.75	Bearing	0.6
Vase	0.75	Four Legs	0.8

Table 5.3. Success Rate (S.R.) of categorization of the Aim@Shape dataset. The overall rate is **87.25%**.

5.6.2 Tosca

We tested also the Tosca [123] dataset with a retrieval and a categorization task. In this case we divided the results for the different type of transformation.

Again, for the retrieval task, we measured the performance using the precision and the recall. In this case the number of object per category is variable. The query length have been made variable according to size of the specific category, so that 1 is the maximum value of precision obtainable. The overall precision is 0.74%.

For the categorization task, the Leave-One-Out validation have been used. In this case, the overall success rate is very high: 0.98%.

The precision and the success rate for the categorization task, divided for the different transformation are shown in the Table 5.4. In Figure 5.11 the plot of the precision versus recall for the retrieval task is shown.

Transformation	S.R. Precision	S.R. Categorization
Isometry	0.77	1.0
Topology	0.44	0.99
Isometry + Topology	0.71	1.0
Noise	0.6	0.8
Null	0.86	1.0
Triangulation	0.58	1.0
Partially	0.68	1.0

Table 5.4. Precision and Success Rate of the categorization task for the Tosca dataset. The results are divided for type of transformation.

.

5.6.3 Timing and complexity

The entire pipeline is computationally efficient in each stage. We used an entry level laptop at $1.66Ghz$ to perform tests. The code is written in Matlab with some

parts in C. The complexity of segmentation depends by the normalized graph cuts algorithm (i.e., n^2) and by fast marching (i.e., $nlogn$), while the complexity of visual vocabulary construction depends by the k-means algorithm (i.e., n over the number of descriptors). An entire mesh segmentation of 3500 vertices is computed in less than 5 seconds, of which $\sim 2.8s$ are necessary to extract all the seed regions, and $\sim 2.1s$ are needed to compute the entire hierarchical segmentation. Region descriptors are computed efficiently: on the average it takes $\sim 0.5s$ to extract all the four descriptors of a single region. As for the k-means clusterization, 10 clusters for 300 points each composed of 200 feature are extracted in less than one second. Finally, the time needed to train a SVM with 400 elements is $\sim 80s$, while the time needed to validate a single element is about $\sim 2s$. The overall process, for a dataset composed of 400 elements, takes about 2 hours. The method is super-linear both in the number of vertices of a mesh and in the number of total objects. It is worth noting that, although the time is expected to increase with the number of objects, the proposed methods is very local and therefore easily parallelizable.

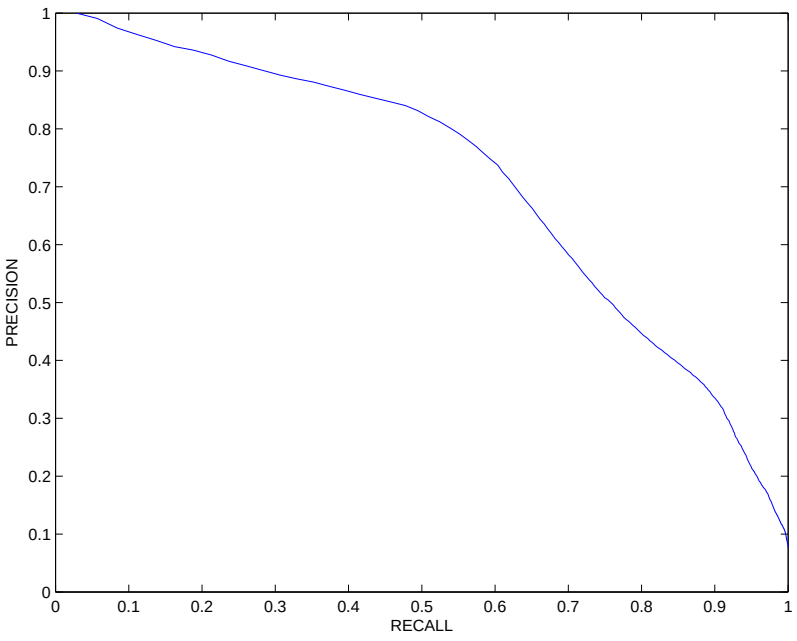


Figure 5.11. Precision versus Recall for the Tosca dataset.

.

5.7 Final Remarks

In this chapter the Bag-of-Words paradigm has been proposed for the 3D domain. The main steps of the involved processing pipeline have been carefully designed by focusing on both the effectiveness and efficiency.

The Bag-of-Words approach fits naturally with sub-parts encoding by combining segment descriptors into several visual vocabularies. In this fashion, our method is able to satisfy query models of composed objects. Moreover, we have proposed a Learning-by-Example approach by introducing a local kernel which implicitly performs the object sub-parts matching. In particular, the object categories are inferred without an exhaustive pairwise comparison between all the models.

The experimental results are compelling. The framework is versatile in reporting satisfying performances for different applicative scenarios such as object retrieval, partial matching and shape categorization as shown in the comparison with other methods.

6

Robust Multiple Model Fitting

In this chapter a novel robust method for fitting multiple models to noisy data will be introduced. In the first part, the method, named J-Linkage, will be presented and compared with other state of the art algorithms for multiple model fitting. In the second part of the Chapter, an automatic heuristic for estimating the inlier threshold and the number of models, will be introduced. Finally, it will be shown how to knock down the quadratic complexity of J-Linkage by employing efficient data structures and by making some non influential approximations. The techniques presented next will be then used in Chapter 7 in order to extract a compact, high level representation of the scene by means of planar patches.

6.1 Related Works

A widespread problem in Computer Vision is fitting a model to noisy data: The RANSAC algorithm [50] is the common practice for that task. It works reliably when data contains measurements from a single structure corrupted by gross outliers. When multiple instances of the same structure are present in the data, the problem becomes tough, as the robust estimator must tolerate both gross outliers and *pseudo-outliers*. The latter are defined in [158] as "outliers to the structure of interest but inliers to a different structure". The difficulty arises because robust estimators, including RANSAC, are designed to extract a single model. Sequential RANSAC – sequentially apply RANSAC and remove the inliers from the data set as each model instance is detected – has been proposed as a solution, but [205] argued that the approach is non optimal, and introduced the multiRANSAC algorithm. The method is effective (provided that the models do not intersect each other), but the number of models is user specified, and this is not acceptable in some applications.

Another popular method based on random sampling and voting is the Randomized Hough Transform (RHT) [201]. It builds an histogram over the parameter space. A minimal sample set is randomly selected and the parameters of the unique model that it defines are recorded in the histogram. Peaks in the histogram correspond to the sought model. Its extension to multiple models is straightforward: they are revealed as multiple peaks in the parameter space. Hence, RHT does not

need to know the number of models beforehand. However, RHT suffers from the typical shortcomings of Hough Transform methods, such as limited accuracy and low computational efficiency. Ultimately, the choice and the discretization of the parameter space turn out to be crucial.

RHT can be seen as an instance of a more general approach consisting of finding modes in parameter space (see e.g. [160]). Instead of quantize the parameter space and accumulate votes, one can map data into the parameter space through random sampling and then seek the modes of the distribution with mean-shift [28]. This, however, is not an intrinsically robust technique, even if it can be robustified with outliers rejection heuristics. Moreover, the choice of the parametrization is critical, as in RHT.

In summary, RANSAC is very robust, but it is not suited to deal with multiple structures. Mode finding in parameter space (and RHT), on the contrary, copes naturally with multiple structures, but cannot deal with high percentage of gross outliers, especially as the number of models grows and the distribution of inliers per model is uneven. Also the algebraic technique presented in [193] is effective in estimating multiple models, but it is not robust to gross outliers.

Recently [202] proposed a novel method for estimating multiple structures based on the analysis of the distribution of residuals of individual data points with respect to the hypotheses, generated by a RANSAC-like sampling process. It turns out that the modes of the residuals distribution reflects the model instances. With respect to RANSAC and other robust methods (such as LMeds, for example) this entails a change of perspective: "studying the distribution of the residuals for each data point instead of studying the distribution of residuals per each hypothesis" [202].

Residuals for each data point have peaks corresponding to the true models because hypotheses generated with random sampling tend to cluster around the true model, a fact that is also at the basis of RHT. The method, in principle, can discover the number of models automatically as in RHT and is effective as RANSAC. However, finding modes ends up to be cumbersome, as proved in our experiments. One reason is that the peak corresponding to a given model becomes less localized as the point-model distance increases. As a result, the rightmost modes in the histogram are usually drowned in the noise.

In the same spirit of RHT and [202] we exploit clustering of the hypotheses. However J-Linkage does not work in the parameter space, which is at the root of the shortcoming of Hough Transform, nor in the residual space, which leads to the difficulties of modes estimation [202]. We adopt instead a *conceptual representation*: each data point is represented with the characteristic function of the set of models preferred by that point[1]. Multiple models are revealed as clusters in the conceptual space. As in RANSAC, the inlier threshold ϵ needs to be manually input.

Some works [22, 44, 183, 196] deal with the automatic computation of ϵ in the case of *one* model – i.e., in the case of RANSAC – but that are not extendible to the case of multiple models. In Section 6.4, we will define a method to estimate ϵ when using a random sampling when *multiple instances* of a model are presents.

[1] According to [43] the posterior probabilities of an object x given C classes form a *similarity conceptual representation*: $[P(x|\text{ class } 1) \cdots P(x|\text{ class } C)]$.

The recent development of computer hardware have made it possible to perform an increasing number of tasks in real-time. Among the latter, robust model fitting is often a crucial operation. Unfortunately J-Linkage has a quadratic complexity, making it difficult to use in real-time application. A number of efforts have been made in order to increase the efficiency of the standard RANSAC. Some improvements exploit the idea that bad hypotheses can be discarded at an early stage, without the need of verifying them against all data points. In [23] a model is preliminarily tested using a subset of d randomly selected points. The remaining set of points is evaluated only if the first d points are inliers to the generated hypothesis. This pre-verification test allows a boost in the efficiency even if more hypotheses need to be generated.

Further exploiting this idea, Capel [17] proposed a formula to approximate the probability that the current generated hypothesis, evaluated only on a subset of points, is better than the best generated model in the previous steps. Most recently, [25, 109] described an optimal randomized verification strategy based on the theory of sequential decision making.

Sometimes a similarity function between points is available, such as in the case of correspondences between two or more images, or can be estimated. The Progressive Sample Consensus (PROSAC) algorithm [24] is designed to use this similarity score and generate the hypothesis with high similarity first. Assuming that points with high similarity are more likely to be inliers than points with low similarity, the good hypotheses should be generated first.

The problem of obtaining the best guess in a fixed amount of time has been faced for the first time by the preemptive RANSAC procedure [121]. A fixed number of hypotheses are generated and compared against each other in parallel. Using a *breadth-first* approach the hypothesis are tested on a subset and only a fraction of them are evaluated on the next subset.

A combination of techniques based on pre-verification are used to evaluate a small subset of points giving an estimation of the inlier fraction. This preemption scheme is the basis of the recent Adaptive Real-Time Random Sample Consensus (ARRSAC) algorithm [132].

Real-time applications usually requires on-line processing, meaning that data is not available as a batch but it arrives progressively. Thus, a good algorithm not only needs to be fast, but also *incremental*, i.e., it must process the data as they become available, exploiting the results of the previous steps. In [164] an incremental scheme is applied to the Preemptive RANSAC in a vehicle relocation problem. The new features are used to generate new hypotheses and updated accordingly with the preemption scheme.

In Section 6.9, we will see how to knock down the complexity of J-Linkage using an incremental framework.

6.2 J-Linkage

The method starts with random sampling: M model hypothesis are generated by drawing M minimal sets of data points necessary to estimate the model, called minimal sample sets (MSS). Then the consensus set (CS) of each model is com-

puted, as in RANSAC. The CS of a model is the set of points such that their distance from the model is less than a threshold ε.

Imagine to build a $N \times M$ matrix where entry (i, j) is 1 if point i belongs to the CS of model j, 0 otherwise. Each column of that matrix is the characteristic function of the CS of a model hypothesis. Each row indicates which models a points has given consensus to, i.e., which models it prefers. We call this the *preference set* (PS) of a point. Figure 6.1 shows an example of such a matrix in a concrete case.

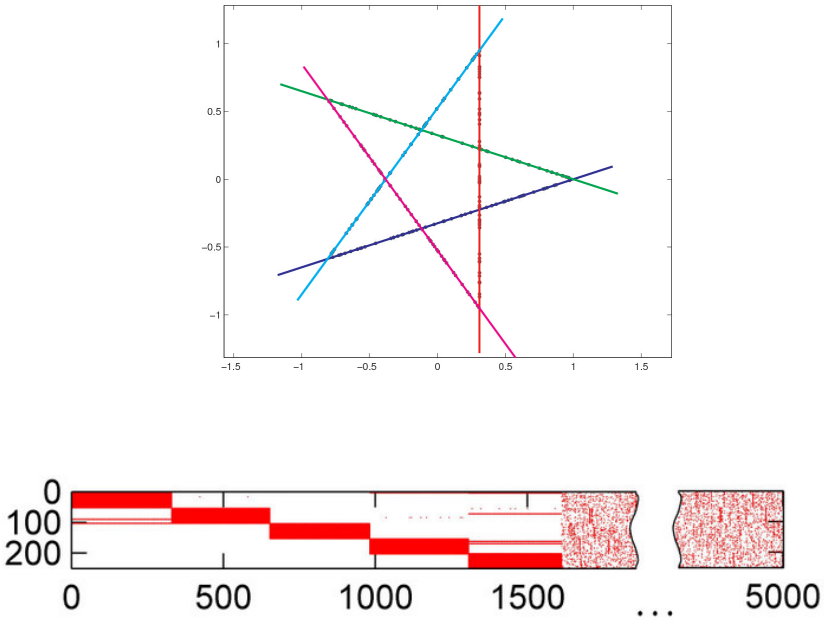


Figure 6.1. Top: the data consist of 250 points on five segments forming a star. Bottom: Preference matrix. The rows are points (ordered by cluster), the columns are models (ordered by cluster size)

The characteristic function of the preference set of a point can be regarded as a conceptual representation of that point. Points belonging to the same structure will have similar conceptual representations, in other words, they will cluster in the *conceptual space* $\{0, 1\}^M$. This is, again, a consequence of the fact that models generated with random sampling cluster in the hypothesis space around the true models.

6.2.1 Random sampling

Minimal sample sets are constructed in a way that neighboring points are selected with higher probability, as suggested in [91, 205]. Namely, if a point $\mathbf{x}_i$ has already

been selected, then $\mathbf{x}_j$ has the following probability of being drawn:

$$P(\mathbf{x}_j|\mathbf{x}_i) = \begin{cases} \frac{1}{Z}\exp -\frac{||\mathbf{x}_j-\mathbf{x}_i||^2}{\sigma^2} & \text{if } \mathbf{x}_j \neq \mathbf{x}_i \\ 0 & \text{if } \mathbf{x}_j = \mathbf{x}_i \end{cases} \tag{6.1}$$

where Z is a normalization constant and σ is chosen heuristically.

Then for each point its preference set is computed as the set of models such that the distance from the point is less than the inlier threshold ϵ (same as RANSAC).

The number M of MSS to be drawn is related to the percentage of outlier and must be large enough so that a certain number (at least) of outlier-free MSS are obtained with a given probability for all the models. Please note that if this condition is verified for the model with less inliers, it is automatically verified for all the other models.

Let S be the number of inliers for a given model and N be the total number of points. The probability of drawing a MSS of cardinality d composed only of inliers is given by:

$$p = P(E_1)P(E_2|E_1)\dots P(E_d|E_1, E_2\dots E_{d-i}) \tag{6.2}$$

where E_i is the event "extract an inlier at the i-th drawing". In the case of uniform sampling $P(E_i|E_1, E_2\dots E_{i-1}) = \frac{S-i+1}{N-i+1}$. In our case, the first point is sampled with uniform probability, hence $P(E_1) = S/N$, while the others are sampled with the probability function (6.1), therefore, after expanding the normalization constant Z, the conditional probability can be approximated as

$$P(E_i|E_1, E_2\dots E_{i-1}) = \frac{(S-i+1)\exp -\frac{\alpha^2}{\sigma^2}}{(N-S-i+1)\exp -\frac{\omega^2}{\sigma^2} + (S-i+1)\exp -\frac{\alpha^2}{\sigma^2}} \quad i = 2\dots d \tag{6.3}$$

where α is the average inlier-inlier distance, and ω is the average inlier-outlier distance. If $S \gg d$ then

$$p \simeq \delta\left(\frac{\delta\exp -\frac{\alpha^2}{\sigma^2}}{(1-\delta)\exp -\frac{\omega^2}{\sigma^2} + \delta\exp -\frac{\alpha^2}{\sigma^2}}\right)^{d-1}. \tag{6.4}$$

where $\delta = S/N$ is the inlier fraction for a given model. Therefore, assuming that ω is larger than α, the sampling strategy increases the probability of extracting an outlier-free MSS, as the intuition would also suggests.

Finally, the probability of drawing at least K outlier-free MSS out of M, for a given model, is given by [202]:

$$\rho = 1 - \sum_{k=0}^{K-1}\binom{M}{k}p^k(1-p)^{M-k}. \tag{6.5}$$

This Equation is used to compute the required number of samples M for a given confidence ρ and a given K. The value of δ in (6.4) must be set to the smallest inliers fraction among all the models. Values of ρ vs M are shown in Figure 6.2.

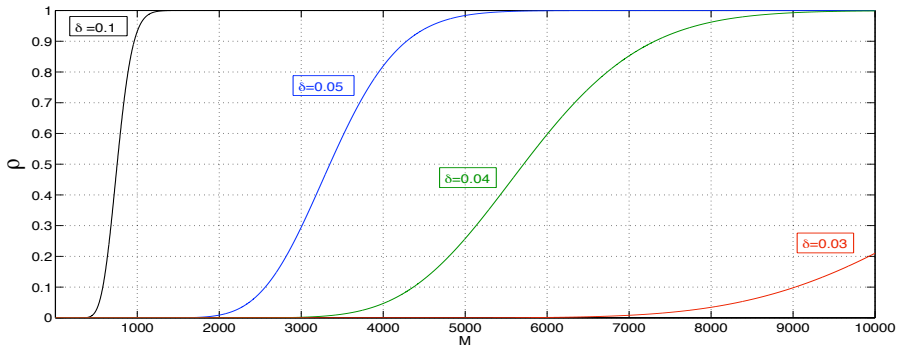


Figure 6.2. Plot of ρ vs M for different values of δ with $K = 25, \alpha^2 = 0.5\sigma^2 \beta^2 = 3.0\sigma^2$.

6.2.2 J-linkage clustering

Models are extracted by agglomerative clustering of data points in the conceptual space, where each point is represented by (the characteristic function of) its preference set.

The general agglomerative clustering algorithm proceeds in a bottom-up manner: Starting from all singletons, each sweep of the algorithm merges the two clusters with the smallest distance. The way the distance between clusters is computed produces different flavors of the algorithm, namely the simple linkage, complete linkage and average linkage [41].

We propose a variation that fits very well to our problem, called *J-linkage* (see algorithm 1). First the preference set of a cluster is computed as the *intersection* of the preference sets of its points. Then the distance between two elements (point or cluster) is computed as the *Jaccard distance* between the respective preference sets.

Definition 6.1 (Jaccard distance). *Given two sets A and B, the Jaccard distance is*

$$d_{\mathrm{J}}(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

The Jaccard distance measures the degree of overlap of the two sets and ranges from 0 (identical sets) to 1 (disjoint sets).

The cut-off value is set to 1, which means that the algorithm will only link together elements whose preference sets overlap. Please note that the cut-off distance is not data dependent, but defines a qualitative behavior of the J-linkage algorithm. Indeed, as a result, clusters of points have the following properties:

- for each cluster there exist at least one models that is in the PS of all the points (i.e., a model that fits all the points of the cluster)
- one model cannot be in the PS of *all* the points of two distinct clusters (otherwise they would have been linked).

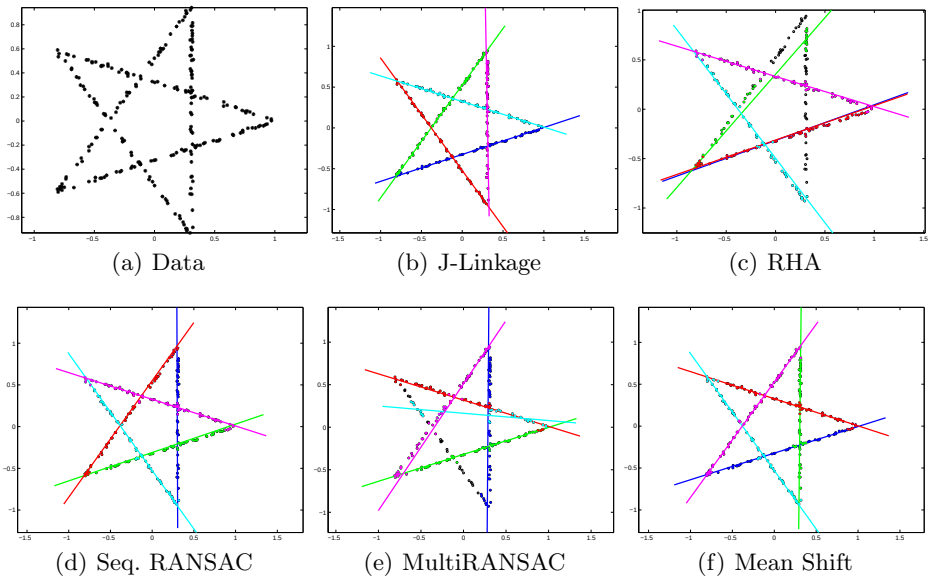
Each cluster of points defines (at least) one model. If more models fit all the points of a cluster they must be very similar. The final model for each cluster of points is estimated by least squares fitting.

Outliers emerge as small clusters. Depending on the application, one may set different rejection thresholds. If the percentage of outliers is known or can be estimated (as it is assumed in RANSAC), one may reject all the smallest clusters up to the number of outliers.

Algorithm 6 J-linkage

Input: the set of data points, each point represented by its preference set (PS).
Output: clusters of points belonging to the same model.

1. Put each point in its own cluster.
2. Define the PS of a cluster as the *intersection* of the PSs of its points.
3. Among all current clusters, pick the two clusters with the smallest Jaccard distance between the respective PSs.
4. Replace these two clusters with the union of the two original ones.
5. Repeat from step 3 while the smallest Jaccard distance is lower than 1.



(a) Data (b) J-Linkage (c) RHA

(d) Seq. RANSAC (e) MultiRANSAC (f) Mean Shift

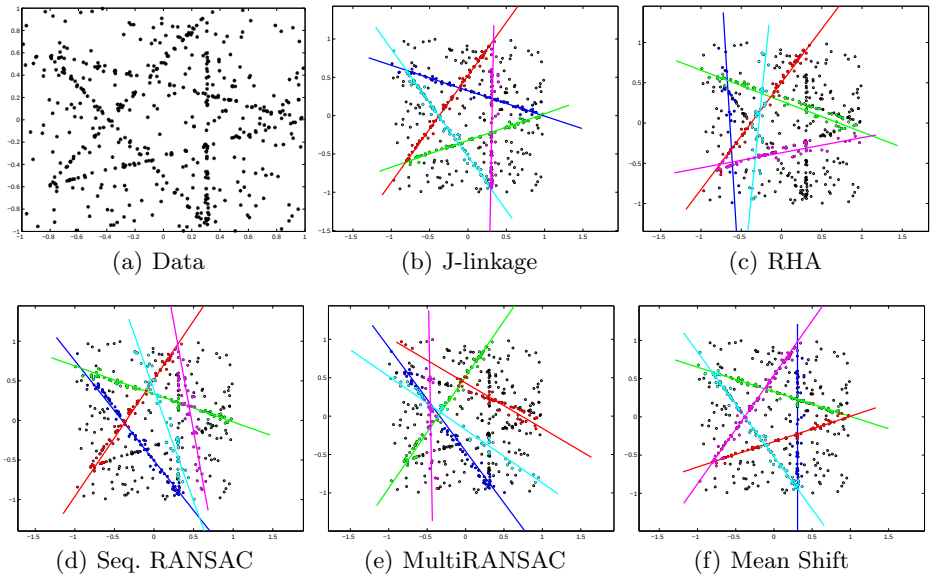
Figure 6.3. *star5* set perturbed with Gaussian noise ($\sigma_n = 0.0075$) and no outliers.

6.3 J-Linkage Results

We performed comparative experiments with sequential RANSAC, multiRANSAC, residual histogram analysis [202] (henceforth RHA) and mean-shift (MS). In all the experiments each model consists of 50 inliers, corrupted by variable Gaussian noise and variable outliers percentage. The data sets consist of segments in several configuration: star (*star5* and *star11*), circles (*circle5*), and horizontal (*stair4*). The latter was used also in [205].

All the methods being compared are based on random sampling, hence we used the same sampling strategy (Equation 6.1) and number of samples (5000) in all the experiments. The scale parameter σ in the sampling strategy is 0.2 in all the experiments but *stair4*, where it has been set to 0.05. The inlier threshold ϵ – variable with the noise level – was the same for sequential RANSAC, multiRANSAC and our method. The parameters needed by MS (bandwidth) and by RHA have been tuned manually to optimize performances. The best outcome out of several trials have been recorded. As multiRANSAC requires prior specification of the number n of models, for the sake of fairness we used the same information also with the other algorithms: only the best or strongest n models among the ones produced by the algorithm were considered. For example, with J-linkage we retained the n largest clusters, and the same for MS. In RHA we sought the n strongest modes.

The results on synthetic data are reported in Figures 6.3, 6.4, 6.5, 6.6, 6.7. In summary, we see that MS on the *star5* data set always produces the correct result, whereas all the other methods break when the outlier percentage grows to 50%. If the number of models increases (*star11* data set), however, only J-linkage produces the correct result, at the same level of noise and outliers. Also on the



(a) Data (b) J-linkage (c) RHA

(d) Seq. RANSAC (e) MultiRANSAC (f) Mean Shift

Figure 6.4. *star5* set perturbed with Gaussian noise ($\sigma_n = 0.0075$) and 50% outliers.

circle5 data set, with random noise and 50% outliers, J-linkage is the only one that works correctly. On the *stair4* data set with and 60% outliers both multiRANSAC and J-linkage yield the correct result.

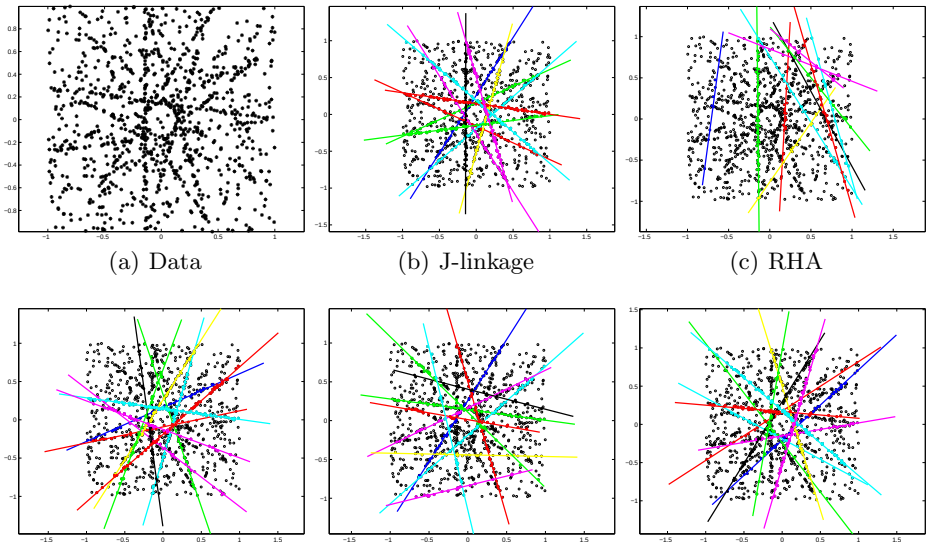
We noted that multiRANSAC systematically fails when the models intersect each other (the problem is more evident when the models are cyclically intersecting). The reason is that the greedy approach adopted by multiRANSAC tries to maximize the number of total inliers, implicitly assuming non intersecting models.

As an example of a real case, Figure 6.8 shows the results of fitting planes to a cloud of 3D points. They were produced by the Structure and Motion pipeline presented in Chapter 2 fed with a set of images of the church of Pozzoveggiani (Italy). Despite the fact that gross outliers are absent, pseudo-outliers and the uneven distribution of points among the models (ranging from 9 to 1692) challenges any model fitting algorithm. Indeed, our method is the only one that produces the correct result. To appraise further the result of J-linkage, Figure 6.9 show two images of Pozzoveggiani church with points marked according to the plane they belong to.

The C++ code of JLinkage, developed by the Author, is available for download from the web[2].

6.4 Fitting validation

The inlier threshold (or scale) ϵ is used to define which points belong to which model. We can observe that if ϵ is too small, noisy data points are explained by

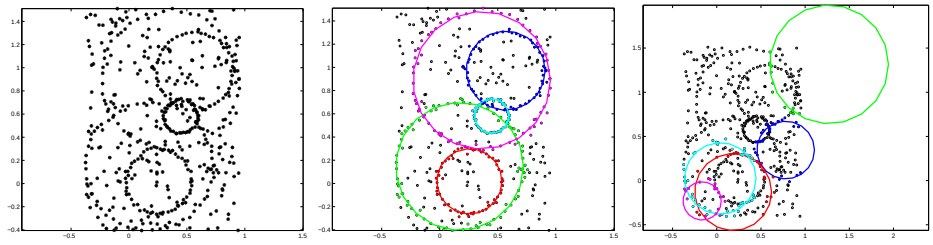


(a) Data (b) J-linkage (c) RHA

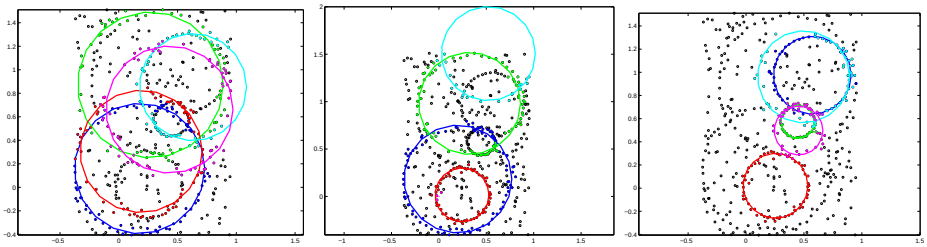
(d) Seq. RANSAC (e) MultiRANSAC (f) Mean Shift

Figure 6.5. *star11* set perturbed with Gaussian noise ($\sigma_n = 0.0075$) and 50% outliers.

[2] http://jlinkage.3dflow.net/

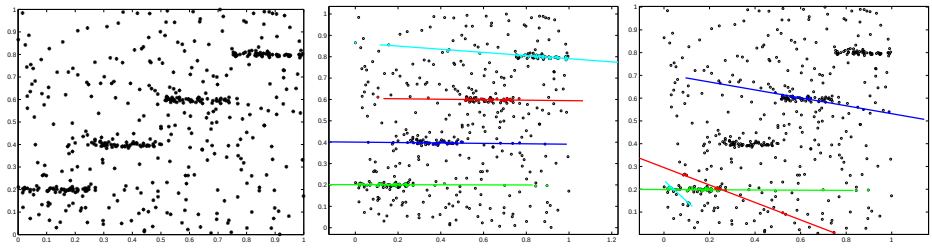


(a) Data (b) J-Linkage (c) RHA

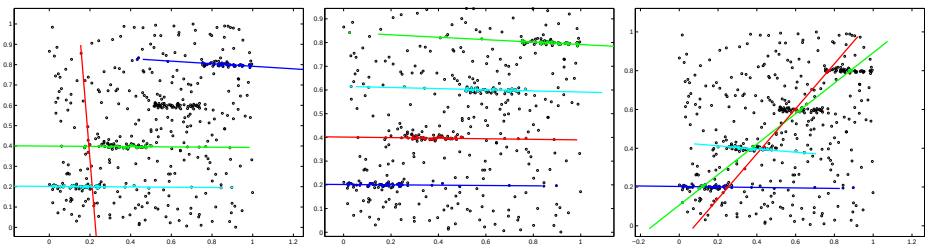


(d) Seq. RANSAC (e) MultiRANSAC (f) Mean Shift

Figure 6.6. *circle5* set perturbed with Gaussian noise ($\sigma_n = 0.0075$) and 50% outliers.

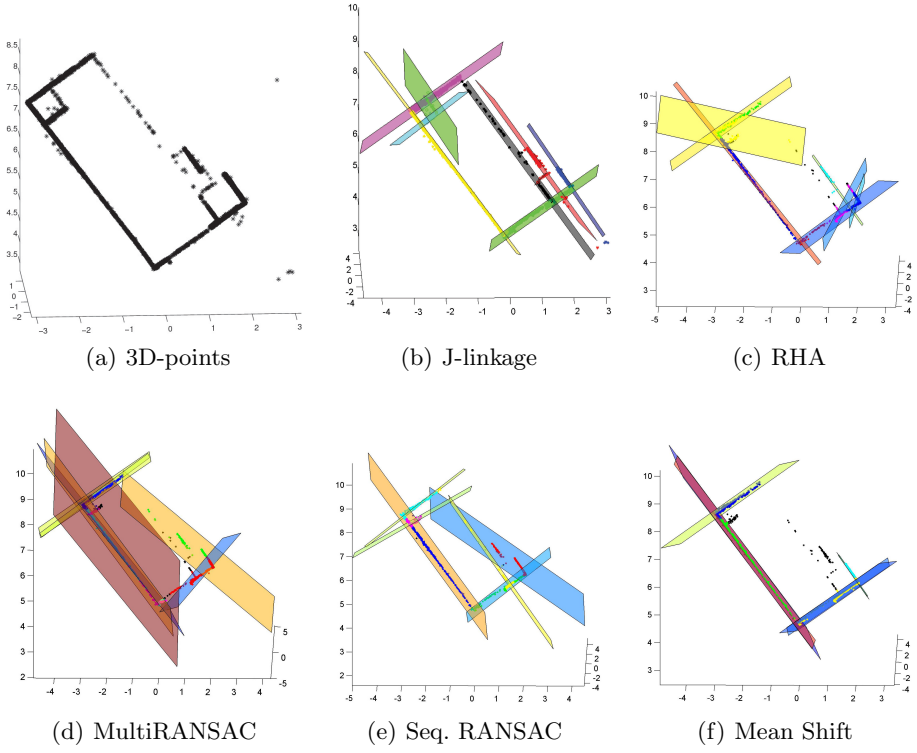


(a) Data (b) J-Linkage (c) RHA



(d) Seq. RANSAC (e) MultiRANSAC (f) Mean Shift

Figure 6.7. *stair4* set perturbed with Gaussian noise ($\sigma_n = 0.0075$) and 60% outliers.



(a) 3D-points (b) J-linkage (c) RHA

(d) MultiRANSAC (e) Seq. RANSAC (f) Mean Shift

Figure 6.8. "Pozzoveggiani" dataset. 3D planes extracted by J-linkage and other algorithms viewed from the zenith.

multiple similar well-fitted models, that is, the separation of the models is poor; instead, if ϵ is too large, they are explained by a single poorly-fitted model, that is, the compactness of the model is poor. The "just right" ϵ should strike the balance between model separation and model compactness.

The degree of separation with respect to compactness is measured in the following proposed method by a score very similar to the Silhouette index [133] for clustering validation. We compute the difference between the second and the first least model distance for each data point. The scale ϵ provides a normalizing denominator for the error difference. Consider some perturbation to the "just right" scale. As ϵ increases, a model recruits new data points, increasing the first error while decreasing the second error, causing the index to drop. As ϵ decreases, new models are created, decreasing the second error, causing the index to drop as well. Therefore, the "just right" scale is the one that yields the largest overall score from all the points.

The proposed technique is inspired to the literature on clustering validation, and in particular we evaluate the fitting results produced by sequential RANSAC or J-linkage with different values of ϵ in terms of compactness and separation. The value that yields the best score is the optimal ϵ.

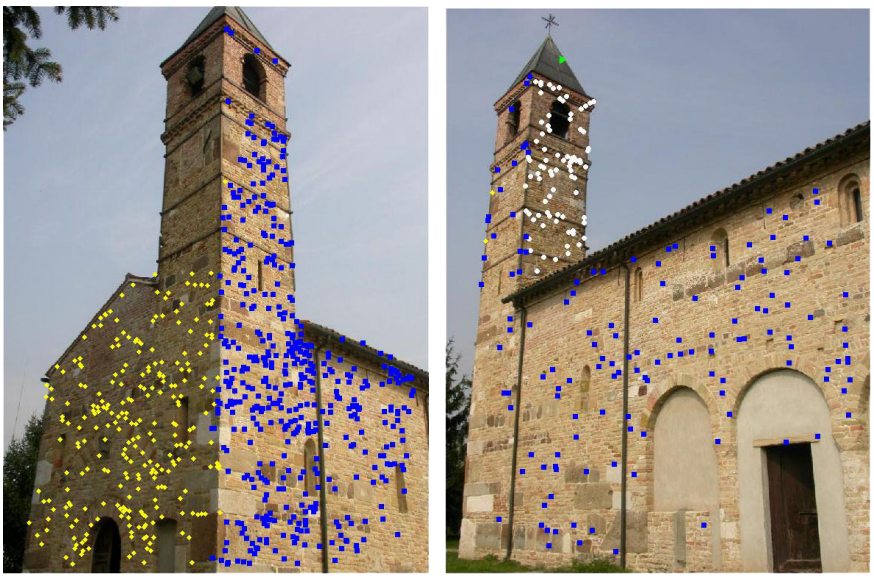


Figure 6.9. Two images of the "Pozzoveggiani" dataset. Points belonging to different planes according to J-linkage are plotted with different markers.

6.4.1 Review of the Silhouette Index

Cluster validation is commonly used for evaluating the quality of partition produced by any clustering algorithm. Cluster validity measures the goodness of a clustering relative to others created by other clustering algorithms, or by the same algorithms using different parameter values. Cluster validation is a very important issue in clustering analysis because the result of clustering needs to be validated in most applications. In most clustering algorithms, the number of clusters is set as user parameter. There are a lot of approaches to find the best number of clusters.

The Silhouette validation technique [133] calculates the silhouette width for each sample, average silhouette width for each cluster and overall average silhouette width for a total data set. Using this approach each cluster could be represented by so-called silhouette, which is based on the comparison of its tightness and separation. The average silhouette width could be applied for evaluation of clustering validity and also could be used to decide how good is the number of selected clusters.

To construct the silhouettes S(i) the following formula is used

$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{6.6}$$

where a(i) average dissimilarity of i-object to all other objects in the same cluster; b(i) minimum of average dissimilarity of i-object to all objects in other cluster (in the closest cluster).

It is followed from the formula that if silhouette value is close to 1, it means that sample is well-clustered and it was assigned to a very appropriate cluster. If

silhouette value is about zero, it means that that sample could be assign to another closest cluster as well, and the sample lies equally far away from both clusters. If silhouette value is close to 1, it means that sample is misclassified and is merely somewhere in between the clusters. The overall average silhouette width for the entire plot is simply the average of the S(i) for all objects in the whole dataset.

The largest overall average silhouette indicates the best clustering (number of clusters). Therefore, the number of clusters with the maximum overall average silhouette width is taken as the optimal one.

6.4.2 Method overview

In this section we shall concentrate on a method for estimating the inlier threshold ϵ based on *validating* the output of the robust fitting, which consists in a grouping of data points according to the model they belong to, plus some points (outliers) that have not been fitted. A criterion for discarding outliers is instead described in Section 6.5.

As said previously, the validation of the fitting is based on the concepts of *compactness* and *separation*, drawn from the clustering validation literature. In particular, our method stems from the following observation (see Figure 6.10):

- if ϵ is too small a single structure will be fitted by more than one model, resulting in a low separation between points belonging to different models;
- if ϵ is too large the points fitted by one model will have a low compactness (or, equivalently, a high looseness), which produces a sub-optimal estimate. As an extreme case, different structures might be fitted by a single model.

This idea is captured by the following index for a given point i:

$$r_i = \frac{b_i - a_i}{\epsilon} \tag{6.7}$$

where a_i is the distance of i to the model it belongs to (looseness) and b_i is the distance of i to the second closest model (separation). The global score, function of ϵ, is the average of r_i over all the points. We claim that the "just right" ϵ is the one that yields the maximum global score. Imagine to start with the "just right" ϵ. If we decrease it, a new model is created which causes the average separation b_i to drop. If ϵ is increased such that at least one new point is added to the model, this point will increase the average looseness a_i and decrease the average b_i, resulting in a decrease of the score (see Figure 6.11). A score with a similar behavior (but slightly worse performances with very small ϵ) is the ratio b_i/a_i which resembles the ratio matching used in [106].

The algorithm for finding the optimal ϵ is iterative: the fitting algorithm is run several times with different ϵ and the value that obtained the higher score is retained.

6.5 Outliers rejection

Sequential RANSAC and J-linkage – like clustering algorithms – in principle fit all the data. Bad models must be filtered out a-posteriori. Usually it is assumed that

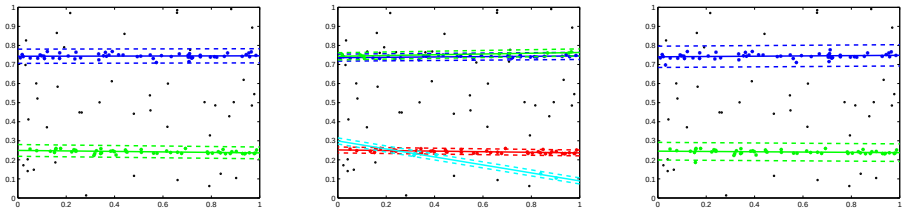


Figure 6.10. From right to left: an optimal fit (according to our score), a fit with a smaller ϵ and a fit with a bigger ϵ. The dotted lines depicts the inlier band of width 2ϵ.

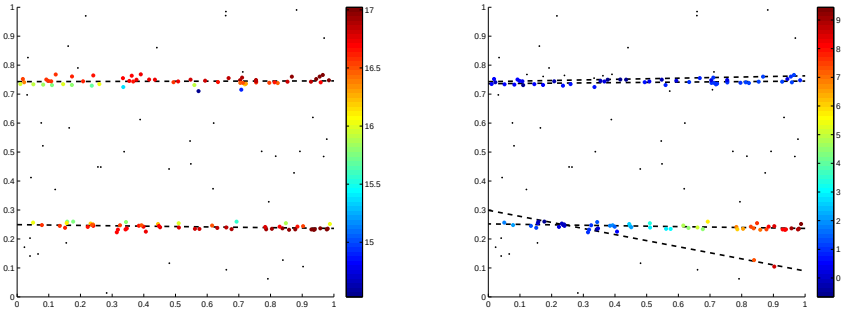


Figure 6.11. This Figure is best viewed in color. The left plot depicts a "good fit", whereas the right plot depicts a "bad fit" (the upper structure is fitted by two lines very close to each other). The color of the points encodes the score values, which are consistently lower in the "bad fit".

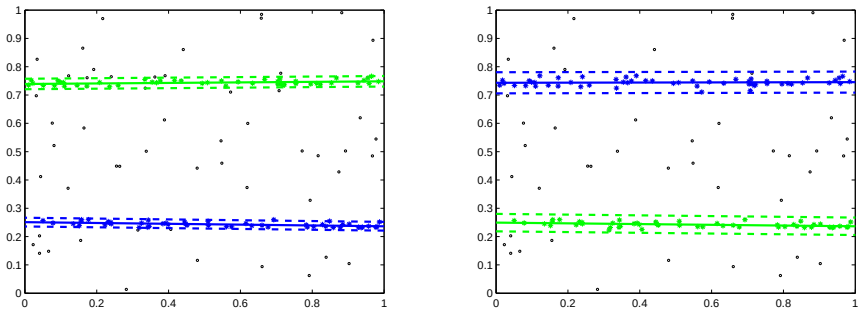
the number of models is known beforehand, but this assumption is too strong and we prefer not to rely on it. Thresholding on the cardinality of the consensus set may be an option, but this does not work if the number of data points supporting the different model instances are very different. A better approach would exploit the observation that outliers possess a diffused distribution whereas structures are concentrated. In particular, we used here a robust statistics based on points density.

Let e_i be distance of point i to its closest neighbor belonging to the same model. According to the X84 rejection rule [74], the inliers are those points such that

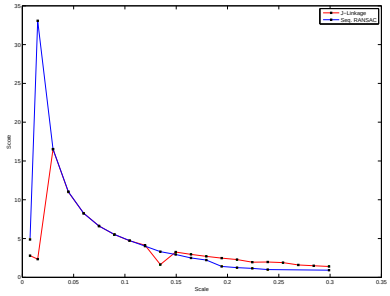
$$e_i < 5.2 \operatorname{med}_i |e_i - \operatorname{med}_j e_j|. \tag{6.8}$$

where med is the median. The models that are supported by the majority of outliers are discarded. Points that are classified as outliers but belongs to a "good" model are retained.

As the robust fit algorithm is executed repeatedly in order to estimate the best ϵ, the models that are fitted change, and hence changes the inlier/outlier classification.



(a) Seq. RANSAC optimal configuration. (b) J-Linkage optimal configuration.



(c) Score for different ϵ values.

Figure 6.12. Two lines example.

6.6 Localized sampling

The assumption that outliers posses a diffused distribution implies that the a-priori probability that two points belong to the same structure is higher the smaller the distance between the points [115], as previously explained in Section 6.2. Hence minimal sample sets are constructed in a way that neighboring points are selected with higher probability. This increases the probability of selecting a set of inliers as described by Equation 6.1. Given an estimate α of the average inlier-inlier distance, the value of σ is selected such that a point at a distance α has a given probability P to be selected (we set $P = 0.5$).

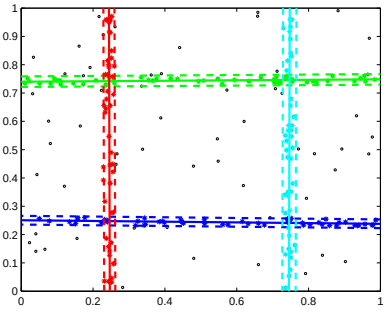
$$\sigma = \frac{\alpha}{\sqrt{-log(P) - log(Z)}}. \tag{6.9}$$

The value of α is iteratively estimated as the outlier/inlier classification changes.

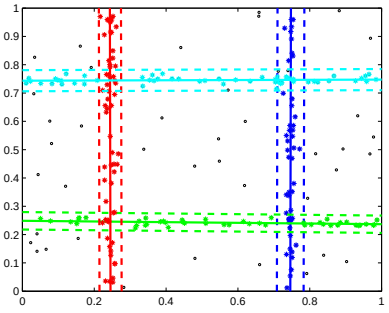
The normalization constant can be approximated as:

$$Z \simeq (1-\delta)e^{-\frac{\omega^2}{\sigma^2}} + \delta e^{-\frac{\alpha^2}{\sigma^2}} \tag{6.10}$$

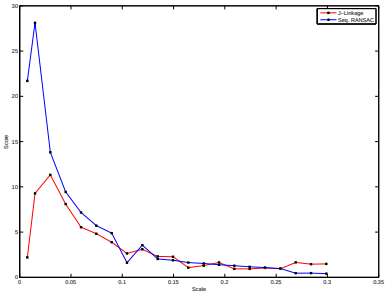
where ω is the average inlier-outlier distance and δ is the lowest inlier fraction among the models.



(a) Seq. RANSAC optimal configuration.



(b) J-Linkage optimal configuration.



(c) Score for different ϵ values.

Figure 6.13. Four lines example.

6.7 Summary of the fitting validation method

The proposed method can be seen as a meta-heuristic that is able to set automatically all the thresholds, perform outlier rejection and validate the final fitting. Any algorithm based on random sampling, from RANSAC to J-linkage, could fit.

The Algorithm 7 presents the method in pseudo-code. The interval search for ϵ, $[\epsilon_{\mathrm{L}}, \epsilon_{\mathrm{H}}]$ must be specified.

6.8 Experiments on Fitting Validation

We tested our scale estimation method with two different multiple models fitting algorithm based on random sampling: Sequential RANSAC and J-Linkage. The scale is initialized to an huge value such that in the first step only one model arises. Subsequently the scale is decreased by a constant value. We used the same number of samples generated (10000) for all the experiments.

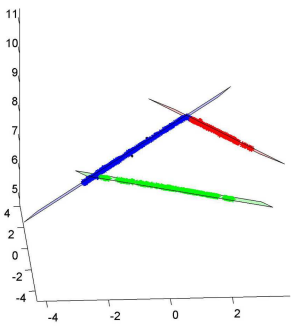
The examples reported consist of both synthetic and real data. For the synthetic ones (Figure 6.12 and Figure 6.13) we used two and four lines in the plane, respectively The inlier points of the lines are surrounded by 50 pure outliers.

Algorithm 7 Model Fitting Validation

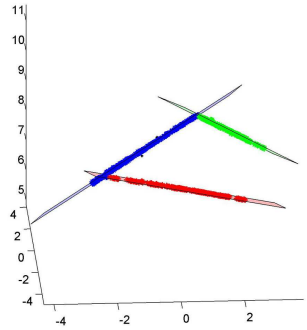
Input: the set of data points.
Output: clusters of points belonging to the same model.

1. Set $\alpha = \epsilon_{\mathrm{H}}$, $\omega = 2\alpha$.
2. for $\epsilon = \epsilon_{\mathrm{H}}$ to ϵ_{L}
   a) Compute σ, using α and ω (Equation 6.9);
   b) Run multiple-models fitting (e.g. sequential RANSAC, J-linkage) using σ for sampling and ϵ for consensus;
   c) Identify outliers (Section 6.5);
   d) Compute the average score (Equation 6.7);
   e) Compute α, ω;
3. end
4. Retain the result with the highest score.

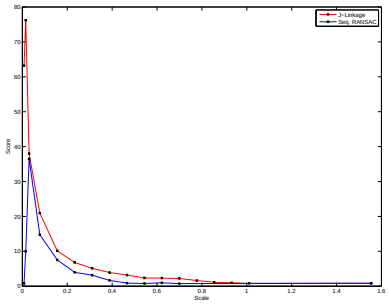
Gaussian noise with standard deviation of 0.1 is added to coordinate of each inlier point.



(a) Seq. RANSAC optimal configuration.

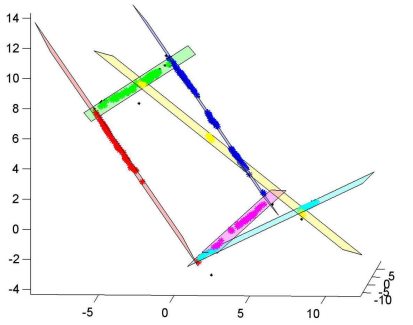


(b) J-Linkage optimal configuration.

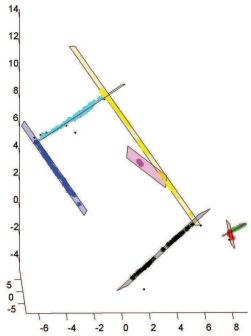


(c) Score for different ϵ values.

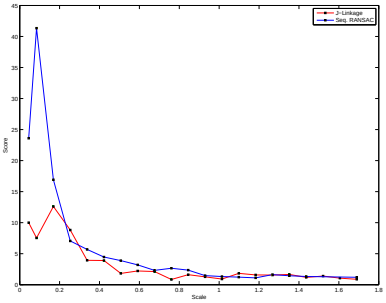
Figure 6.14. Example of planes fitting in 3D from real data (castle).



(a) Seq. RANSAC optimal configuration.



(b) J-Linkage optimal configuration.



(c) Score for different ϵ values.

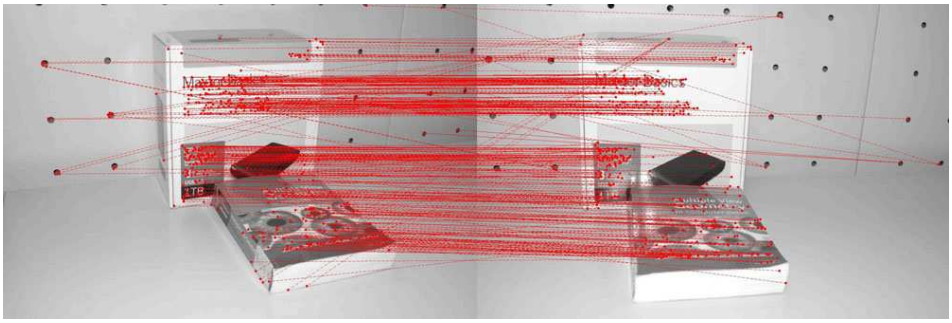
Figure 6.15. Example of planes fitting in 3D from real data (square).

In these experiments our meta-heuristic always proved able to converge toward a correct solution, provided that the underlying fitting algorithm (namely RANSAC or J-linkage) produced at least one correct fit.

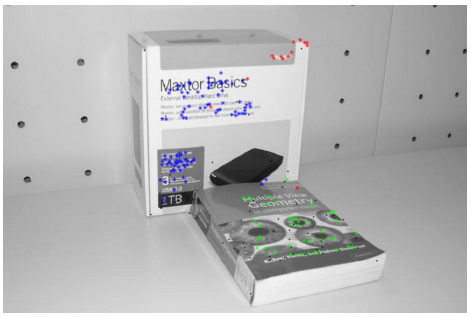
In the real 3D data example (Figure 6.14 and Figure 6.15) planes are fitted to a cloud of 3D points produced by the Structure from Motion pipeline introduced in Chapter 2 fed with images of a castle and a square, respectively.

Finally, in the last two datasets (Figure 6.16 and Figure 6.17), SIFT features [106] are detected and matched in real images and homographies are fitted to the set of matches.

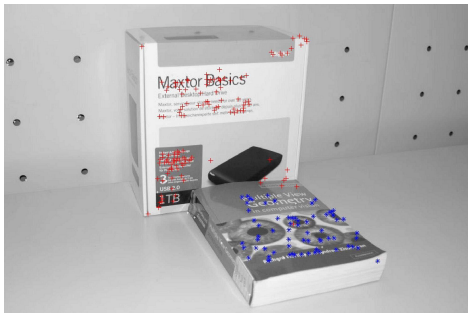
In the last three cases our scale estimation produced qualitatively different results when applied to sequential RANSAC or J-linkage, as not only the value of the optimal ϵ are different, but also estimated models are different. The J-linkage results are generally more correct, but the output of sequential RANSAC is not completely wrong, suggesting that our meta-heuristic is able to produce a sensible result even when applied to model fitting algorithms that produces sub-optimal solutions.



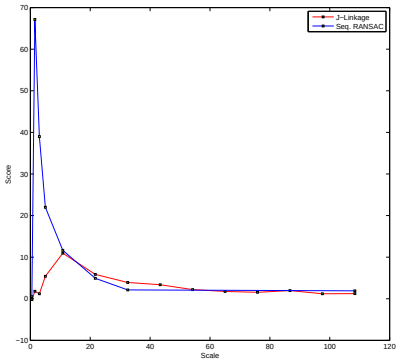
(a) SIFT Matches



(b) Seq. RANSAC optimal configuration (points membership is color coded).



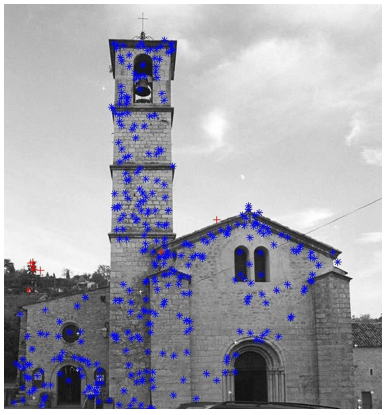
(c) J-Linkage optimal configuration (points membership is color coded).



(d) Score for different ϵ values.

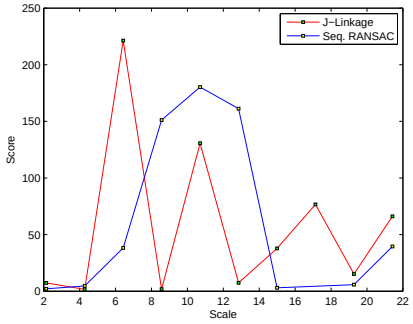
Figure 6.16. Example of homography fitting (books).

(a) SIFT Matches



(b) Seq. RANSAC optimal configuration (points membership is color coded).

(c) J-Linkage optimal configuration (points membership is color coded).



(d) Score for different ϵ values.

Figure 6.17. Example of homography fitting (Valbonne).

6.9 Real-time J-linkage

The original J-linkage algorithm works in batch, i.e., it processes all the data at once, and its complexity does not suite real-time constraints. This Section will describe how we modified J-Linkage in order to make it work *on-line* in *real-time*. This means that the data becomes available in pieces at each time step, and that it must be processed to yield partial results before the next time step. This new *Real-time J-linkage* is *incremental* (it processes the data as it becomes available and reuses the information extracted in the previous time step) and improves the *time efficiency* of J-linkage.

6.9.1 Random sampling

A non-uniform sampling strategy for the points of the sample other than the first, as explained in Section 6.2, is important in most of the problems. However, a continuous probability function defined over the distances between all the points is too expensive to compute for a real-time application. We propose to simplify Equation 6.1 so that the probability of choosing a point $\mathbf{x}_j$ depends on whether or not it is in the k-neighborhood of the already selected point $\mathbf{x}_i$, namely:

$$P(\mathbf{x}_j|\mathbf{x}_i) = \begin{cases} P_h & \text{if } \mathbf{x}_j \in k\text{−Neighborhood}(\mathbf{x}_i) \\ P_l & \text{if } \mathbf{x}_j \notin k\text{−Neighborhood}(\mathbf{x}_i) \end{cases} \tag{6.11}$$

with $P_h >> P_l$. The k-neighborhood can be computed efficiently by storing the data points in a KD-tree data structure. Inserting or removing a node in the tree requires $O(\log(n))$ time, while querying requires $O(\log(n^{\frac{2}{3}} + k))$ time in the case of three-dimensional data.

6.9.2 On-line processing

The Real-time J-linkage performs an agglomerative clustering every time period. The time between two consecutive time steps is spent in three activities, as illustrated in Figure 6.18: the agglomerative clustering, the insertion/removal of new/old data points and the update of the hypotheses pool. In the following of this Section we will describe these three stages.

Clustering

The agglomerative clustering consists in iteratively choosing the closest two elements to be merged until the smallest Jaccard distance is 1, as explained in Section 6.2. The bottleneck is determining – at each iteration – the two closest elements. At a first glance, storing the pairwise distances in a heap seems the best choice, since the structure can be built in $O(n)$ time and the minimum key can be extracted in constant time. In our application, however, the keys are updated at every step, or eliminated (when the Jaccard distance is equal to 1) and each of these operations would cost $O(\log n)$ time. In contrast, a double linked ordered list requires $O(n \log(n))$ time to be constructed, but removing an element from the list can be

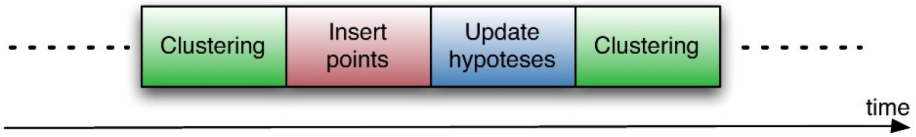


Figure 6.18. The on-line processing time is spent in three stages: clustering step, point insertion and update of hypotheses.

performed in constant time. Every time a distance update occurs the list should be sorted again, but a lazy update approach can be implemented. The minimum updated distance is kept and compared against the first element of the list: only when the latter is greater the sorting takes place, and this occurs a very few times during the clusterization, because distances are usually incremented during the updates. For these reasons, we employed a double linked list.

An ordered list is a better choice for our application, but it's still not enough and some simplification must be performed.

A further approximation came from the same observation underlying the local random sampling: inliers tend to be closer to one another than outliers. Exploiting this, one can conjecture that the distance of two points in the conceptual space tends to be lower if the points are closer in the Euclidean space. Thus, the algorithm does not calculate the pairwise distance between all the points in conceptual space, but only for points in the Euclidean k-neighborhood. The latter is readily available from the KD-tree that was built previously. When two clusters are merged together, the neighborhoods of the points are merged accordingly: The k-neighborhood of a cluster is defined as the union of the k-neighborhoods of its members. (Figure 6.19).

Thus, the initial distances can be computed in $O(n * k)$ time.

Point insertion

In an incremental framework not only points are processed (i.e., clustered) as soon as they become available, but they are also discarded when a certain event occurs (e.g. a buffer size is reached, a fixed time is elapsed, or the point is not visible any more).

In principle, when a point is inserted the Real-time J-linkage needs to compute its preference set and add it to the agglomerative clustering as a singleton. In order to satisfy the real-time constraint, incoming points are stored in a FIFO buffer and fetched for insertion in batches of constant size.

When a point is removed, instead, the algorithm updates the preference set of the cluster it belonged to (which is the intersection of the preference sets of the points that belong to the cluster).

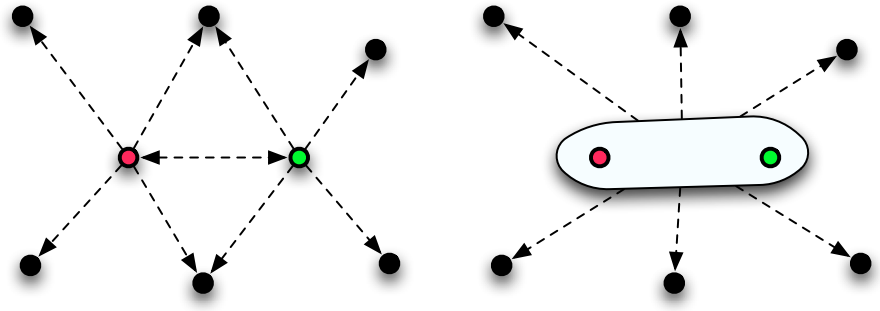


Figure 6.19. When the red and green points (left) are merged in a cluster, the k-neighborhood of the resulting cluster (right) is the union of the k-neighborhoods of the two points.

Update of the hypotheses

Whereas in the batch version of the J-linkage all the hypotheses are generated at once, in this on-line version a fixed number of hypotheses is generated at each time step and added to the pool, replacing the oldest ones. When a new hypothesis is added, the preference set of all the points must be updated accordingly, and the preference sets of the clusters must be updated as well.

As a consequence, a cluster may broken apart during this operation, due to the fact that the intersection of the preference sets may be empty. In this case the points that belonged to the clusters are re-inserted as singletons.

Then the agglomerative clustering step is run again on the updated set of clusters.

The computational time of the agglomerative clustering depends on the number of initial clusters c, that in our case is equal to n. However, since we have a limited amount of time, we can compute the clusterization only on a subset of points. At each step, we can add a fixed maximum amount of points; in this way the complexity of the clusterization will be bound to the points added plus the clusters computed at the previous step.

6.10 Experiments for Real Time J-Linkage

We tested both the time efficiency and the accuracy of the proposed Real-time J-linkage framework on the task of 3D planes extraction (data are 3D points and models are planes).

We run three categories of experiments: simulation with synthetic data, simulation with real data coming from a batch structure-and-motion pipeline and experiments with PTAM[3], a real-time software for tracking and mapping in small environments [94, 95].

[3] Freely available at http://www.robots.ox.ac.uk/~gk/PTAM/

Algorithm 8 Real Time J-Linkage

Input: points stored in a queue Q.
Output: detected geometric primitives R.

1. Fetch n points from Q and add them to the active points set X and to the clustering C as singletons.
2. Generate k hypotesis by randomly sampling the points in X and add them to the hypotesis pool H, thereby substituting the k oldest hypotesis.
3. Update the preference set of each cluster in C.
4. Perform agglomerative clustering on C (Section 6.9.2).
5. Fit a geometric primitive to each cluster of C and output them.
6. Remove from X and C points that are no longer visible.
7. Goto step 1.

The algorithm have been coded C++ code and carefully optimized. The bitwise operations are carried out using the BitMagic library[4], that implements several performance optimization for Intel platforms.

6.10.1 Synthetic data

Two examples have been synthetically generated. In the first one, two planes are present, each one composed of 100 points corrupted by a small amount of Gaussian noise. Additionally, 50 pure outliers were introduced. Real-time J-linkage was fed with 10 points and generated 1000 hypothesis per time step. The original J-linkage was repeatedly run on the partial data.

Two sample results are shown in Figure 6.20, while a comparison with the original J-linkage algorithm in terms of accuracy and time is reported in Table 6.1. The *accuracy* is defined by considering this as the problem of classifying inliers vs outliers, so it is (# true positives + # true negatives) / # of points.

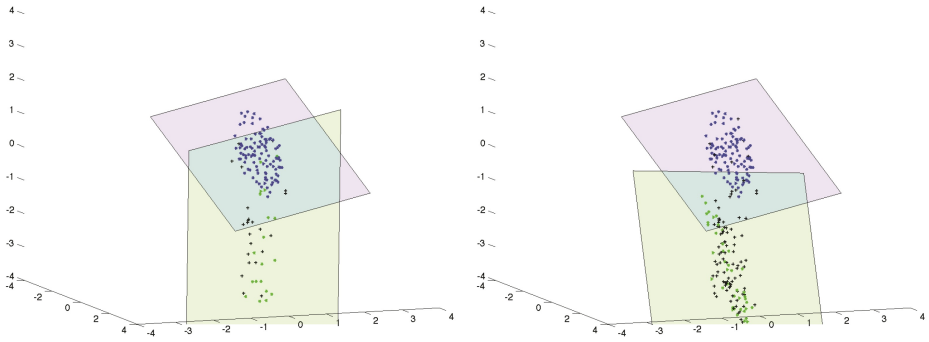
In the second example, four planes have been generated, each one composed of 100 points corrupted by a small amount of Gaussian noise and 50 pure outliers withal. Real-time J-linkage was fed with 10 points and 2000 hypothesis per time step has been generated. The size of the hypothesis pool was 10000 for both examples.

Two sample results are shown in Figure 6.22, while a comparison with the original J-linkage algorithm in terms of accuracy and time is reported in Table 6.2. The reader might notice how the loss in accuracy with respect to J-linkage is balanced out by a pronounced speed-up in execution time.

6.10.2 Real data from SaM

We tested Real-time J-linkage with data coming from Samantha, the SfM pipeline introduced in Chapter 2. In order to simulate a real-time setting, points were stored in a queue from which Real-time J-linkage fetched subsets of ten at each time step.

[4] Freely downloadable from http://bmagic.sourceforge.net



(a) Frame 15. (b) Frame 25.

Figure 6.20. Planes extracted at different times in the 2-planes example.

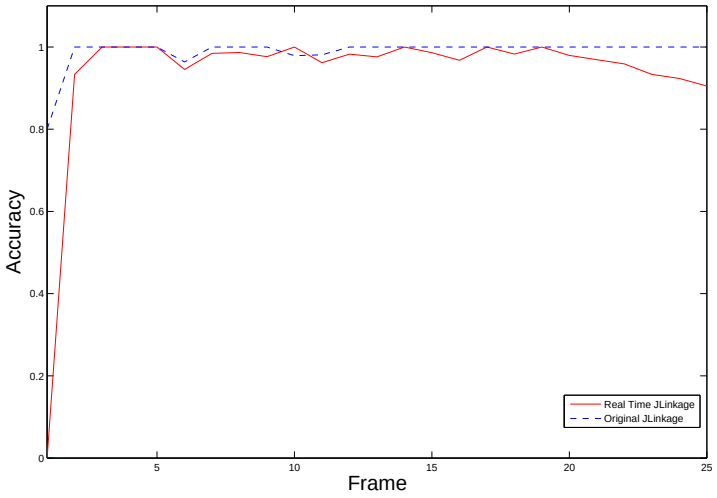
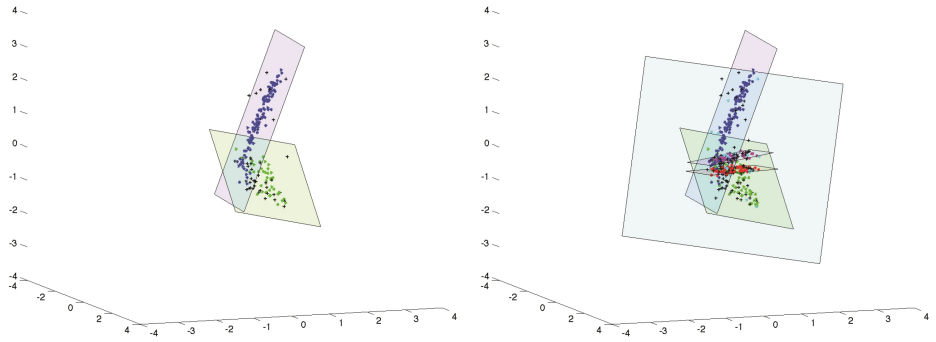


Figure 6.21. Accuracy comparison between J-l and RTJ-l for the 2-planes example.

Table 6.1. Comparison of J-linkage (JL) vs Real-time J-linkage (RTJL) in the 4-planes example.

	Time [s]	Average fps	Speed up	Accuracy
JL	70.1	0.35	1	1.0
RTJL	3.1	8	22.61	0.72



(a) Frame 20. (b) Frame 45.

Figure 6.22. Planes extracted at different times in the 4-planes example.

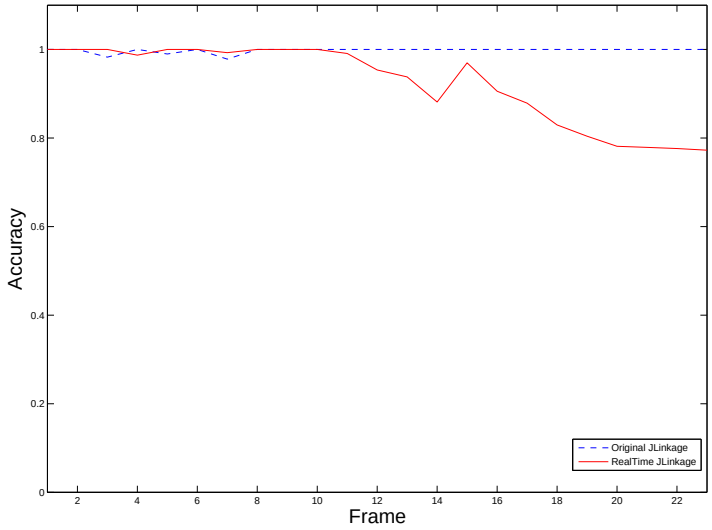


Figure 6.23. Accuracy comparison between J-l and RTJ-l for the 4-planes example.

Table 6.2. Comparison of J-linkage (JL) vs Real-time J-linkage (RTJL) in the 4-planes example.

	Time [s]	Average fps	Speed up	Accuracy
JL	459.85	0.097	1	1.0
RTJL	8.82	5.1	52.13	0.7125

Two data-set have been tested: *Castelvecchio* and *Valbonne*. The first one is composed of 871 points, and it took 23.36 seconds to run all the 88 steps. Two results are shown in Figure 6.24. The second example is composed of 673 points and it took 14.69 seconds to run all the 63 steps. Two sample results are shown in Figure 6.25.

(a) Frame 15. (b) Frame 88.

Figure 6.24. Extracted planes at different times from *Castelvecchio* data-set. Different colors encode different planes.

6.10.3 Real data from PTAM

In order to test the proposed algorithm in a real environment, we plugged it to the output of the PTAM (Parallel Tracking and Mapping for Small Augmented Reality Workspaces) software. Although PTAM is mainly used for augmented reality tasks, it can provide a rough three-dimensional reconstruction of points in real-time.

Some output examples are shown in Figure 6.26. Points belonging to the same plane share the same color. Provided that the data is reliable enough, our method is capable of detecting planes in real-time with good accuracy. Points were inserted as soon as they were detected. Real-time J-linkage generated 1000 hypothesis per time step (in this case the clock is given by the frame rate of the input video), with an hypothesis pool of size 10000. Planes with less than 20 points were discarded.

A video of the system in action is available for download [5].

6.11 Final Remarks

In this chapter we described a novel method for fitting multiple instances of a model to data corrupted by noise and outliers. The experiments showed that J-linkage compares favorably with other state-of-the-art methods. Our insight about the reason for this is that it derives the best features from the competing algorithms.

[5] `http://www.toldo.info`

(a) Frame 40. (b) Frame 63.

Figure 6.25. Extracted planes at different times from *Valbonne* data-set. Different colors encode different planes.

Like RANSAC it is robust because only the points below the outlier threshold matters. Compared to RHA, it consider only the the first bin of the residuals histogram, while RHA takes the whole the histogram into account, which is corrupted by outliers. Like RHA, however, our conceptual representation casts the problem from the perspective of the data point, which is a strength of RHA. The discovery of multiple models is devolved to clustering, thereby gaining a global view of the problem, whereas sequential RANSAC and multiRANSAC are forced to make local, greedy choices. Clustering, however, is not an intrinsically robust technique, in general. J-linkage, instead, is inherently robust, because it is based on the intersection of preference sets, hence it favors the growing of clusters made up of inliers only. On the contrary, modes of the residual histogram or of the parameters distribution are difficult to locate, especially when the number of gross outliers and pseudo-outliers grows, and the fraction of outlier-free sets generated by the random sampling decreases accordingly.

In addition, a meta-heuristic for scale estimation in RANSAC-like approaches to multiple models fitting has been introduced. The technique is inspired by clustering validation, and in particular it is based on a score function that resembles the Silhouette index. The fitting results produced by sequential RANSAC or J-

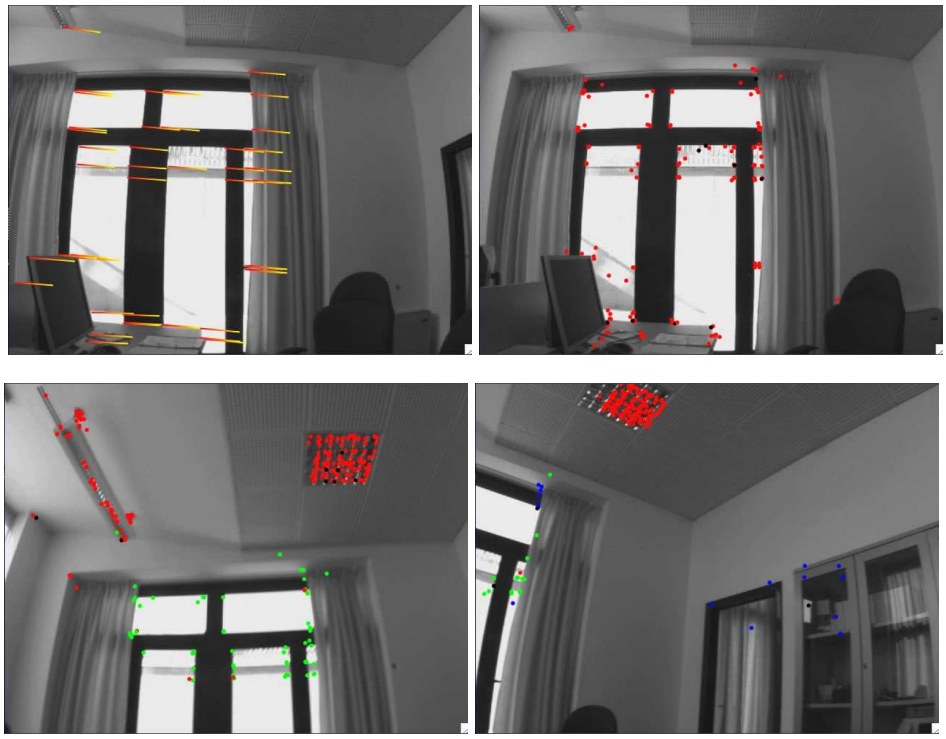


Figure 6.26. Different frames of the real-time simulation with PTAM software. Different colors encode different planes.

linkage with different values of scale are evaluated and the best result – according to the score – is retained.

Finally we have knocked down the complexity of the original J-Linkage, making it capable of detecting multiple instances of a model from data corrupted by noise and outliers in real-time. Starting from the original J-linkage algorithm, several adaptations have been made to make it incremental and faster. The trade-off between time efficiency and accuracy can be controlled parametrically by changing the number of points or hypothesis processed at each step, the total length of the hypothesis pool or the number k of neighbors to take into account.

In the next Chapter we will see how we can use the just introduced J-Linkage framework to produce high-level representations of the scene, starting directly from the SfM output.

7

Image-Consistent Patches

In Chapter 3 we have seen how high-detailed models can be automatically extracted, starting from the output of a Structure from Motion pipeline. Multiview stereo methods have a wide range of application and the problem is well-studied in Computer Vision. However, fewer researchers have addressed the problem of extracting more abstract representations of the scene. A novel algorithm to extract planar patches will be introduced next. This represents a step forward towards the definition of higher level renditions of the scene. The method uses J-Linkage, the multiple models fitting method presented in Chapter 6.

The rest of the Chapter is organized as follows. Section 7.2 describes how geometrical and image constraints have been added to J-Linkage, in order to address the patch fitting problem. Additional post-processing steps are illustrated in Section 7.3. Section 7.4 presents the hierarchical processing. Experiments are reported in Section 7.5 and, finally, conclusions are drawn in Section 7.6.

7.1 Related Works

Although the current state of the art in three-dimensional (3D) reconstruction from images addresses the recovery of dense and accurate models [15,58,69], there is still an unfulfilled need for compact, abstract representations of objects.

What separates unstructured point clouds from higher-level renditions of a model is a *semantic gap*, which could be bridged by leveraging additional information, such as meshing, surfaces, ordering, occlusion, parallelism and orthogonality of structures. Increasing levels of abstraction can then be obtained progressing to recognition of scene elements or entire architectural scenes.

As we saw in Chapter 3, very recent works based on multiview stereo coupled with a structure-and-motion pipeline [15, 58, 69] produce visually compelling results, but they do not address the semantic gap issue at all, since the output is a dense, non-compact representation of the scene.

This is one of the most challenging research area in Computer Vision. Three main approaches can be recognized that aims at bridging the semantic gap: interactive, top-down and bottom-up.

Interactive approaches require user intervention to recognize higher level structures, usually basing on the 3D information previously extracted [27,168,189,197].

Top-down or model-based approaches start from the prior knowledge of the set of potential parametric models and try to infer the best fitting one along with its parameters [10,37,140,171]. Potentially, only one image could be employed if the prior knowledge is enough to derive the 3D model [75,114].

Bottom-up methods start directly from 3D data points trying to aggregate them in progressively higher-level structures, possibly using the reflectance information coming from the images, namely *image-consistency*. The method proposed next falls in this category: The aim is to leverage models from unorganized point clouds to an intermediate representation made of planar patches as they are a good starting point for a complete automatic reconstruction of surfaces.

Some methods try to optimize an initial triangulation using visibility [82] or image-consistency [112,116] only. They work only with very simple convex polyhedral objects, and they assume all points visible in at least one view. Others [4,30] extract the planes underlying the scene using RANSAC (or MSAC) with spatial or image-consistency information. However, the sequential application of an algorithm designed for *single* model extraction is not suitable for large, noisy datasets. In fact, the experiments reported in those papers involve extremely simple objects.

In this chapter the J-Linkage framework presented in Chapter 6 will be specialized to the task of fitting planar patches to large unorganized point clouds, by integrating in a seamless way image-consistency and visibility constraints.

The closely related work is the one described in [21], which starts from a *dense* reconstruction (from multiview stereo) and obtains a piecewise planar representation. In this work, instead, we face a more difficult problem, for we start with the *sparse* output of the SfM pipeline introduced in Chapter 2, thereby avoiding the multiview stereo stage.

7.2 Constraints integration

In the previous Chapter we have saw how planes can be fitted to a sparse point cloud in a robust way using J-Linkage. However, fitting planes does not solve the problem of extracting planar patches, for a patch is a *region* of the plane, and the same plane may contain more patches (see Figure 7.1).

This Section describes how to leverage the J-linkage algorithm to fit planar patches to a cloud of 3D points that are considered as samples of actual surfaces in the observed scene.

The planar patch associated to a set of coplanar points is the *convex hull* of the projection of the points onto the supporting plane[1]. In order for a patch to represent an actual surface, it must satisfy a number of constraints, beside coplanarity, that will be described later. This Section will concentrate on how these constraints can be seamlessly integrated inside J-linkage.

We define the planar patch associated to a cluster of points as the surface delimited by the convex hull given by the projection of the points onto the fitting

[1] According to this definition patches are convex. This requirement will be relaxed in the post-processing step (see Section 7.3)

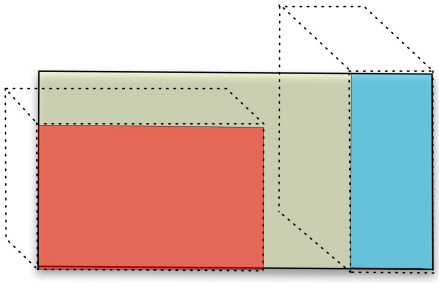


Figure 7.1. A single plane may contain several patches (blue and red).

plane. Making use of the solely spatial information is not enough for our goal. Furthermore we can exploit additional information coming from the images and the Structure from Motion pipeline itself. Unlike other algorithms, J-linkage extracts models in an incremental way, by merging smaller structures at each step. In the case of planes, two clusters can merge only if the result is a set of coplanar points (within the inlier threshold ε): Coplanarity is the invariant property for plane fitting.

In the case of planar patches, other constraints can be enforced as invariant properties, so that two patches can be merged if and only if the resulting patch does not violate these constraints. The base step of the algorithm is the merge step in the agglomerative clustering process. Inlier points are thus assigned to each structure by merging two existing structures. The merge is already vetoed if the Jaccard distance between the two preference of the clusters is equal to 1, i.e. if they are not coplanar to some extent. The validation of additional constraints can be integrated at this level so that two clusters can be merged if and only if the new cluster does not violate additional rules. The additional constraints must hold for the planar patch associated to the new cluster. The problem can be formulated by means of triangles by simply triangulating the planar patch. When a point is added to a cluster, a new triangle is created that must be tested against the constraints. By taking advantage of the incremental nature of J-linkage, we can test the constraints in an inductive way. At the base step, a single point give rise to a zero-dimensional patch and thus the constraints are always verified. When two patches are being considered for possible merging, a new tentative patch is computed as the convex hull of the union of the points. Consider the triangulation of this convex hull: by the inductive hypothesis the triangles belonging to the two original patches satisfy the constraints, whereas the other triangles S (those belonging to the "seam" between the two patches) must be tested against the constraints. If a single triangle fails the merging is rejected. A graphical explanation of this incremental step is shown in the Figure 7.2.

More in detail, three constraints are considered:

- **Visibility**: a triangle must not occlude any visible point (See Section 7.2.1).
- **Image-Consistency**: the projections of a triangle onto the images where it is visible must consist of conjugate points (See Section 7.2.2).

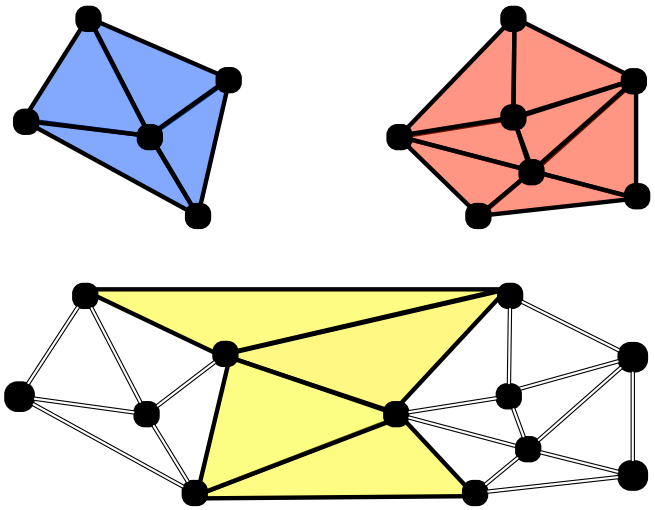


Figure 7.2. Incremental step. Top: the two patches that are to be merged (blue and red). Bottom: the "seam" triangles S (in yellow) that need to be checked against constraints violation.

- **Non Intersection**: a triangle must not intersect any previously defined surface (See Section 7.2.3).

Sometimes the image-consistency test fails because of small imprecision in the localization of 3D points. In this case, adjusting the 3D points so that to optimize photo-consistency (See Section 7.2.4) of the region around them could cure the problem, beside improving the overall quality of the 3D reconstruction.

7.2.1 Visibility Constraint

A structure-and-motion pipeline typically produces the *visibility* of each point $V(P)$, i.e. the views from which point P is visible. This information derives from the initial stage of the pipeline where keypoints are extracted and matched with other keypoints from different images. This information can be exploited to formulate a simple yet powerful constraint: a triangle must not occlude a 3D point from the view where it is visible.

Mathematically, this translates into a segment-triangle intersection test between the triangle and the line segments joining P and every camera in $V(P)$. The interSection test can be performed efficiently at constant time.

This test must be performed for each view and for each visible point from that view. In order to speed up the process, we precompute the axis aligned bounding box (AABB) for each view that contains every visible points and the optical center. We also compute and update an AABB that contains every point of a patch. A prior intersection test is made between the AABB of the patch and the AABB of a view: if no intersection occurs we are assured that no triangle of the patch will intersect a segment in that view. The intersection test between two AABB also takes constant time.

7.2.2 Photo-Consistency Constraint

A patch in space is *image-consistent* if all its projections onto the images where it is visible consist of conjugate points. Image consistent patches are attached to actual object surfaces in the scene (see Figure 7.3). Image-consistency can be checked through *photo-consistency*, the property that the projections of a patch are equal up to a projective transformation and photometric nuances.

We can check the planar consistency of a triangle with the underlying real structure by checking its photo-consistency or image-consistency. If the underlying structure is planar, the projection of a triangle on the images where it is visible should be identical in every view. This is a consequence of the fact that no parallax error would occur. However, some photometric differences may be present and should be taken into account. Also occlusion may be present, i.e. in the image an object may occlude the observed region.

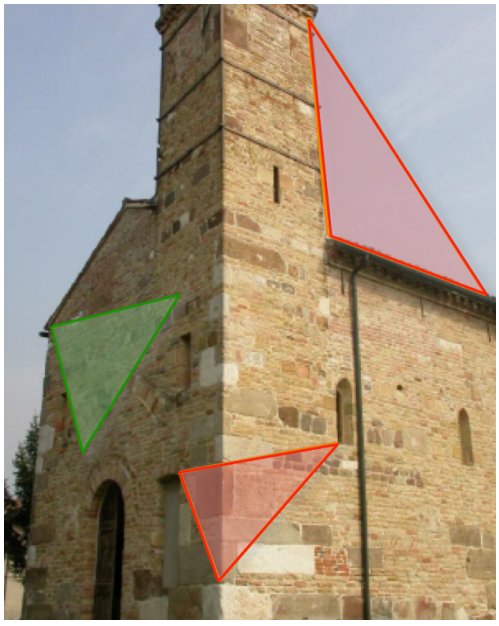


Figure 7.3. The green triangle is image-consistent, the red ones are not, because they are not part of an actual surface.

Let us consider $V(\tau)$, the set of images where the vertices of a given triangle τ are visible. Among them, the one where the projected triangle exhibits the maximum area is chosen as the reference. Using the solely images where all the three points of the triangle are visible is too restrictive. We instead chose as compatible the ones who have at least one point of the triangle visible and where the projections of the points of the triangle fall inside the image space. A reference image is then established, choosing the one in which more points of the triangle are visible. This is done to choose an image with a low probability of occlusion. If two or more

images have the same maximum number of visible points, the one where the projected triangle exhibits the maximum area is preferred. All the triangles in $V(\tau)$ are projectively warped onto the triangle in the reference image and compared to it through normalized cross-correlation (NCC). A weight is then associated at each NCC coefficient computed for every compatible view. The weight is set equal to the number of points of the triangle that were visible from the view. This rewards views less prone to occlusions.

The final photo-consistency of the 3D triangle is obtained as the average of the NCC scores of its projections (the value ranges from -1 to 1), and its is considered photo-consistent if this value is below a fixed threshold.

7.2.3 Non Intersection Constraint

It is often desirable to deal with tridimensional meshes that are not self-intersecting. This derives from the fact that surfaces are assumed to be manifolds.

During the patch growing, it may happen that patches end up intersecting each other in their interior. This is clearly an unwanted situation, as the customary assumption holds that surfaces are manifolds. To avoid this, we embed the non intersection constraint directly inside J-linkage.

When creating a new patch we check that it is not intersecting any previously defined patch. This translates into a triangle-triangle intersection test among all the triangles of two patches. The triangle-triangle intersection test can be computed in constant time. However, when dealing with surfaces composed by many triangles, it may require many checks. We speed up the process taking advantage of the AABB computed for every patch.

We perform a prior AABB-AABB intersection test: if the bounding boxes do not overlap, we are assured that the surfaces are not intersecting each other and we do not need any further testing.

7.2.4 Photo-consistent adjustment

Points coming from a structure-and-motion pipeline derive their position from triangulation and bundle adjustment based on correspondences among image keypoints. After these keypoints have been extracted in the early stage of the pipeline, the photometric information is not taken into account any more, as all the processing is purely geometric. Therefore, the photo-consistency of those points might be less than optimal. This is the rationale for the photo-consistent adjustment (or *photo-adjustment* step that will be described.

The photo-consistency of a point P with fan[2] $\mathcal{F}$ is defined as:

$$\phi(P,\mathcal{F}) = \frac{1}{|V(\mathcal{F})|-1} \sum_{i \in V(\mathcal{F}) \setminus r} \mathrm{ncc}(\Pi_r(\mathcal{F}), T_{i\to r}(\Pi_i(\mathcal{F}))) \tag{7.1}$$

where $V(\mathcal{F})$ is the set of images where $\mathcal{F}$ is visible, r is a reference image (we choose the one where the area of the projection of $\mathcal{F}$ is larger), $T_{i\to r}$ is the projective

[2] A *fan* is a set of connected triangles that share one vertex

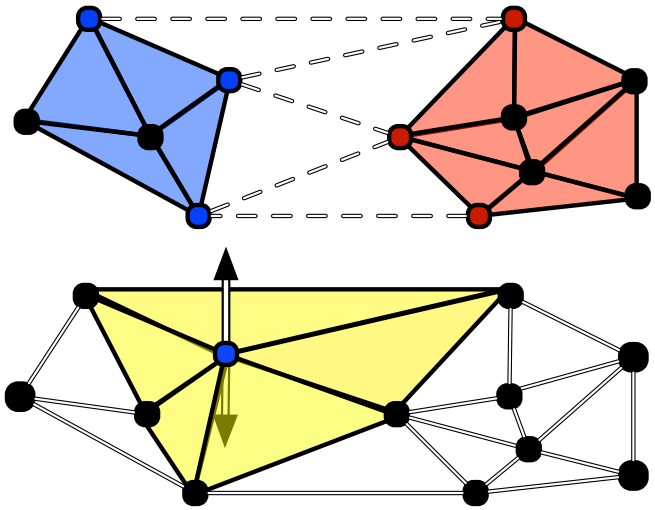


Figure 7.4. Photo-adjustment. Top: The points on the boundary of two patches that are to be merged (red and blue respectively) are those that needs to be photo-adjusted before the constraints check. Bottom: A point gets displaced along its normal using the fan of triangles (yellow) around it.

mapping of the plane passing through $\mathcal{F}$ from image i to image r, Π_i is the operator that projects onto view i, and ncc is the normalized cross correlation.

The point P is adjusted by moving along the normal $\mathbf{n}$ to the plane it belongs to, so as to maximize photo-consistency:

$$\max_{d \in [d-\theta,\, d+\theta]} \phi(P + \mathbf{n}d, \mathcal{F}) \tag{7.2}$$

It is customary in surface multiresolution decomposition (see e.g. [97]) to consider a base surface and a normal displacement vector. The tangential component can be taken into account by changing the point on the base surface.

Photo-adjustment takes place *during* the J-linkage clustering: when two patches are going to be merged, for every new triangle that is instantiated its vertices are photo-adjusted (see Figure 7.4). This guarantees that a point is photo-adjusted before the photo-consistency of the triangle it belongs to is checked.

In our tests we observed that the ratio of triangles that would have failed photo-consistency without the adjustment to the total number of triangles, ranges from 1.8% to 3.5%.

7.3 Post-processing

During the agglomerative clustering of J-linkage, it is sufficient that a single triangle does not satisfy a constraint to discard the entire merge, because it is inductively assumed that patches are *convex*. As a result, triangles that fulfill the constraints are discarded, thereby leaving gaps in the surfaces between neighboring patches. Gaps arises also between non-coplanar patches because in J-linkage a

Algorithm 9 Convex Planar Patches Extraction

Input: cloud of unstructured 3D points.
Output: clusters of points belonging to the same convex planar patch.

1. Compute PS of points with plane models;
2. Put each point in its own cluster;
3. Let C_1 and C_2 the two clusters with the smallest Jaccard distance between the respective PSs;
4. Compute the convex hull of $C_1 \cup C_2$, and identify the set of "seam" triangles S;
5. Do photo-adjustment on vertices of S;
6. If visibility and image-consistency constraints are satisfied by every triangle in S, replace C_1 and C_2 with $C_1 \cup C_2$.
7. Repeat from step 3 while the smallest Jaccard distance is lower than 1.

point can belong to only one patch (or plane): as a result non-coplanar triangles cannot share a common edge (Figure 7.5). This issues are solved *a-posteriori*, by a gap-filling heuristics that relaxes the convexity assumption and the uniqueness of point assignment.

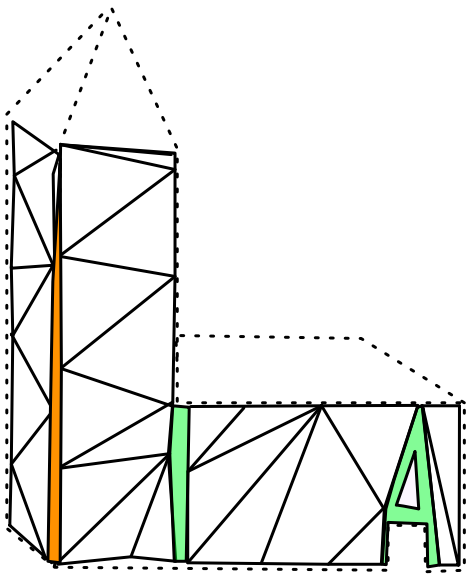


Figure 7.5. Green regions are gaps between adjacent patches that are to be filled. Blue regions are gaps between orthogonal patches.

Two patches are said to be *adjacent* if at least one of the points of one patch contains a point of the other patch in its k-neighborhood (we used $k = 10$ in our experiments). Adjacent patches can be *quasi-coplanar*, if the angle between

the respective support planes is less than 30 degrees, and *quasi-orthogonal*, if the angle lies between 60 and 120 degrees.

First, the algorithm extracts clusters of quasi-coplanar and mutually adjacent patches, with agglomerative clustering using the angle as distance. A patch can be added to a cluster if it is quasi-coplanar to all the patches of the cluster and adjacent to at least one. Eventually, a 2D Delaunay triangulation of the support plane of the whole cluster is run and the new triangles that are thereby created – which cover the regions that connects different patches – are tested against photo-consistency and visibility, as inside the J-linkage (see Figure 7.6). The resulting clusters are the new planar patches, which are possibly larger than the original and non-convex.

Second, the gaps between quasi-orthogonal patches are filled. The algorithm first identifies points compatible with two quasi-orthogonal patches, using the inlier threshold ε, then it tries to add these points to the two patches, checking all the appropriate constraints.

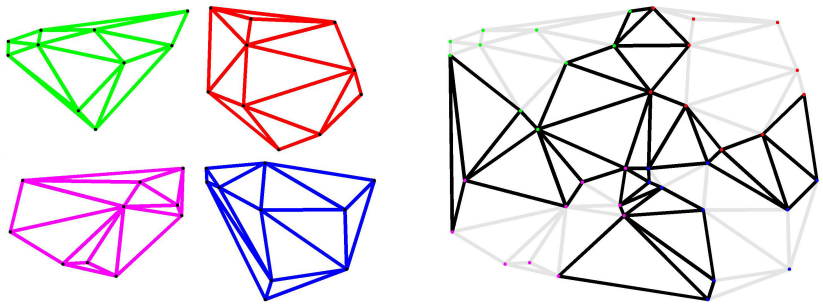


Figure 7.6. Merge of several quasi-coplanar patches into one new patch. The black triangles on the left are those tested against photo-consistency, visibility and non intersection. If any of them fails the patch becomes non convex.

Finally, the resulting patches are cleaned using a procedure that resembles the morphological opening (erosion followed by dilation). First, in the "erosion" step, triangles that have one or zero neighbor are deleted, together with those having an angle smaller than 10 degrees ("skinny" triangles). Then, in the "dilation step", holes, are filled by triangulation.

Figure 7.7 shows the application of the opening procedure on a mesh portion: isolated and skinny triangles are removed and small holes are filled.

Figure 7.8 shows the overall effect of the post processing step. The gaps between coplanar and orthogonal patches are filled with photo-consistent triangles.

The effect of the post-processing is to fill gaps, simplify the results and lighten over-segmentation in a general and problem-independent way. Other obvious problem-driven heuristics could have been implemented in this stage.

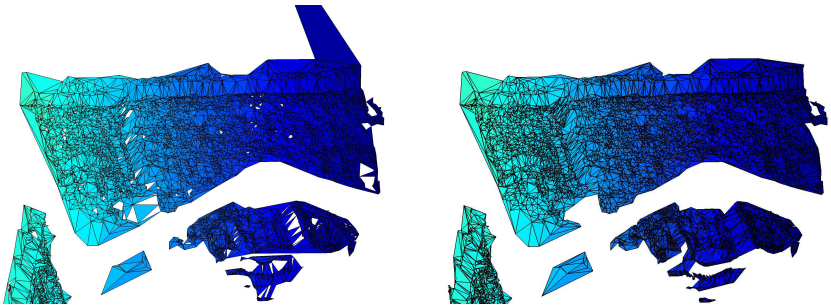


Figure 7.7. Example of the effect of the "opening" procedure. Input mesh on the left, output mesh on the right.

Figure 7.8. Some examples of problems cured by post processing (left-before, right-after). The triangles are colored (in transparency) according to the patch they belong to.

7.4 Hierarchical processing

In this section we leverage the hierarchical partitioning of data (camera and points) provided by Samantha (presented in Chapter 2) to obtain a hierarchical patch fitting procedure which is more computationally efficient, inherently parallel and suitable for out-of-core processing. As we saw, Samantha is a structure-and-motion pipeline that first runs an agglomerative clustering on the set of images and then processes them following the dendrogram. As a result, a hierarchy of contained cluster of points is created, where the final reconstruction rests at the root.

Therefore, instead of processing all the points at the same time, the algorithm presented in the previous Section is run on partial reconstructions and then the results are merged at the father node. In particular, in every node Samantha can perform two operations: (a) add new points to an existing reconstruction (possibly empty) and/or (b) merge two reconstructions.

In case (a) the planar patch fitting algorithm is run on new points and the resulting patches are appended; in case (b) the two set of patches are joined and common points are assigned to the biggest planar patch. As for step a), in order to link the patches the fit the partial reconstruction with the new points, the support planes of the existing patches are forced into the hypothesis pool of J-linkage. Moreover, we implemented a lazy update strategy, i.e., the processing is triggered only when a certain number of points k are waiting to be added.

Figure 7.9 compares the running time of this method (without photo-adjustment) with the sequential approach: It clearly appears that the speed up gained by this approach is remarkable.

7.4.1 Computational complexity

This processing scheme follows a tree, so the total complexity is reduced, with respect to the straightforward sequential approach, which takes $O(n^2 \log n)$ time. In this case, we solve n/k instances of the original problem at the leaves of the tree, with a cost of $n/k\ O(k^2 \log k)$, plus the cost of the internal nodes (merge) which is linear in the number of points to be merged: $n + 2n/2 + 4n/4 + ...hn/h = nh$, where $h = \log(n/k)$ is the height of the tree. Summing up, the time complexity is $n/k\ O(k^2 \log k) + n \log(n/k)$, which is equal to $n \log(n)$ asymptotically, being k constant. Please note, however, that there is a concrete advantage as soon as $k \ll n$.

7.5 Results

Experiments were performed on publicly available data and other datasets that will be released to the public domain. The 3D points were produced by the Structuref from Motion pipeline presented in Chapter 2, together with the hierarchical partitioning of the data.

Four datasets, namely "Piazza Bra[3]", "Campidoglio", "Duomo[3]" and "Castle-P30[4]" have an architectural scale, whereas "Bas-relief" and "Small sculpture" rep-

[3] http://profs.sci.univr.it/~fusiello/demo/samantha/
[4] http://cvlab.epfl.ch/~strecha/multiview/denseMVS.html

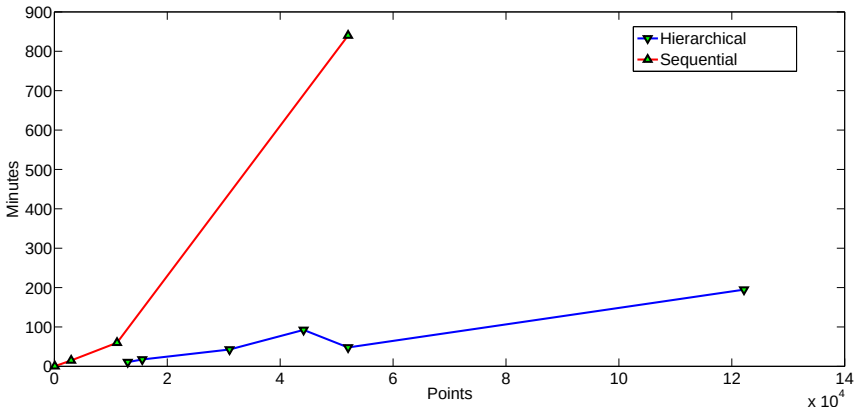


Figure 7.9. Running times of the hierarchical approach versus a batch one (not on the same datasets)

resent smaller objects captured closer by. The latter, in particular, is not piecewise planar and has a complex topology, compared to the others.

The output produced by our method is shown in Figure 7.10, and Figure 7.11. It can be noticed that, besides some minor imprecisions, a meaningful triangulation is extracted from every dataset. On "CastleP-30" and "Campidoglio", the scene is mostly composed by planar regions, and our approach is able to detect them correctly, despite the sparsity of the points in some spots. The "Duomo" dataset is the most challenging among architectural ones. Some minor imprecisions are noticeable in the semicircular apse, but overall the method is able to decompose it in piecewise planar patches, that could be aggregated in higher level primitives by further processing. "Bas-relief" was a fairly easy job, considering the fact that the object is planar and the points cloud is very dense, due to the high resolution (2808x1972) of the images. Finally, in "Small sculpture", although the object is non-planar, our method was able to decompose the surface into piecewise planar regions. Some minor imprecisions are present between these regions, since in most cases they are neither orthogonal nor coplanar.

Running times an entry level PC with a single core 2.4Ghz are reported in Table 7.1.

Finally, thanks to the availability of ground truth for "Duomo" and "Piazza Bra" datasets, we have been able to assess the improvement in accuracy brought in by the photo-adjustment. The laser data had been subsampled in such a way that they have roughly double the number of points of our reconstruction, then we run Iterative Closet Point (ICP) in order to find the best similarity that brings our data onto the model. The average residual distances between closest pairs was taken as reconstruction accuracy. After photo-adjustment the Figure improves by about 4%.

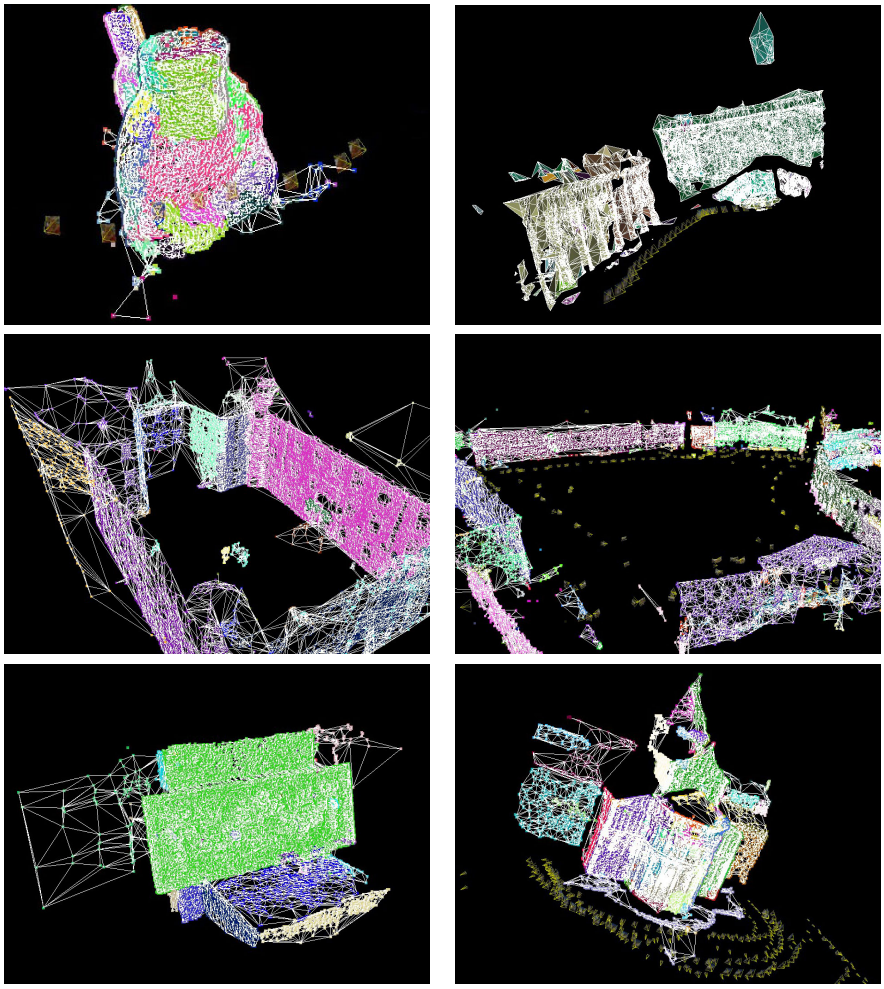


Figure 7.10. Screenshots of the results shown as triangle meshes. Triangles are colored according to the patch they belong to. From left to right: "Small sculpture","Campidoglio", "CastleP-30", "Piazza Bra", "Bas-relief", and "Duomo".

Table 7.1. Running times.

Dataset	# points	# images	Time [min]
Small sculpture	12994	58	10.8
Campidoglio	15571	51	17.5
CastleP-30	31030	30	42.8
Piazza Bra	52024	380	48.0
Bas-relief	44147	45	92.5
Duomo	122159	309	194.8

Figure 7.11. Screenshots of the results. The colored triangle mesh (as in Figure 7.10) is shown projected onto one of the images.

7.6 Final Remarks

In this chapter, we presented a method for extracting a triangle mesh from an unstructured cloud of points, based on the detection of image-consistent planar patches with J-linkage. The proposed approach copes with *sparse* and *real* data, optimizes the position of the points with regard to image-consistency (photo-adjustment) and follows a hierarchical processing scheme that cuts the computing time from $O(n^2 \log n)$ to $O(n \log n)$. These results, obtained with no manual intervention, are a good starting point for further processing, that could include the user in the loop and represents an intermediate, compact and abstract rendition of the scene.

8

Conclusions

In this thesis we have described several improvements to the current state of the art in the context of automatic extraction of high-level models from images. The covered topics were many and several different aspects and problems were treated, both by introducing completely novel solutions and by proposing improvements to existing methods.

In first place, a complete Structure from Motion pipeline has been presented in Chapter 2. Thanks to its hierarchical framework, the complexity has been knocked down and the overall robustness has been increased. Moreover, the pipeline is completely auto-calibrated and does not requiring any external information. Many sub-parts are completely novel, e.g. the original matching phase - which includes symmetric matches, densification and bucketing - and the initial guess of the neighbor views using a novel procedure based on the minimum spanning tree. In the aggregate, this represents a complete, engineered system, that has been later used as base for the other methods presented in this dissertation.

In the subsequent branch of the thesis we have seen how an accurate mesh can be extracted from the output of a SfM pipeline and properly handled by extracting high-level descriptors.

In Chapter 3 a novel multiview stereo algorithm has been introduced. While multiple view stereo is a well studied problem in Computer Vision, the proposed framework has showed to be very effective in real cases. The algorithm is based on [16], but has been improved by including a novel visibility based outlier rejection procedure, rectification in the matching phase, an improved MRF-optimization with the computation done in GPU, a novel surface and texture generation algorithms and many minor accuracy-driven improvements.

A novel global registration algorithm has been later proposed in Chapter 4. An elegant yet simple solution to the registration problem has been derived using the Generalized Procrustes Analysis framework. The algorithm has been demonstrated to best classical approaches.

In Chapter 5 the Bag-of-Words paradigm has been proposed for the 3D domain. This gave rise to an effective retrieval method for 3D meshes that is able to fit naturally with sub-parts encoding. The novelty of the method is two-fold. Firstly, a consistent segmentation algorithm for meshes, based on the so called Minima Rule, has been proposed. Secondly, the Bag-of-Words paradigm has been applied

successfully for describing 3D meshes for the first time. The proposed pipeline has been demonstrated to best many of the state of the art retrieval methods on publicly available datasets.

In the final part of the thesis we explored a completely different way of extracting high-level information directly from unstructured point clouds, trough the robust fitting of high level geometric primitives.

In Chapter 6 we described a novel method for fitting multiple instances of a geometric model to data corrupted by noise and outliers. The method, called J-Linkage, is inherently robust robust as it extract models in a parallel way by clustering points using a robust signature based on random sampling. In the same Chapter we have also introduced a meta-heuristic for scale estimation, which has been inspired by cluster validation techniques. Finally we have seen how the original J-Linkage quadratic complexity can be knocked down thanks to non influential approximation and the usage of appropriate data structures. The algorithm has also been run successfully in real-time using an incremental framework.

The general J-Linkage framework introduced in Chapter 6 has been employed in Chapter 7 in order to extract image-consistent planar patches from the unstructured point cloud coming from the output of a Structure from Motion pipeline. We have seen how planar patches represent a novel, abstract, intermediate and high level representation of the scene. The proposed method copes well with sparse and real data, typically coming from a Structure from Motion pipeline. Several heuristics has been introduced to improve the accuracy and the scalability of the method.

In this dissertation the problem of automatically extracting high level models from images has been discussed in details. The thesis treated both well-studied problems and less explored topics. Even if in the very recent past, the advancements on certain areas of Computer Vision research (e.g. Structure from Motion and Multiple view Stereo) has been outstanding, they cannot be considered a solved problem and much more can be done to improve further the results. On the other hand, the automatic extraction of high level representations and the semantic gap filling is still an unexplored topic. In this dissertation, completely novel solutions has been explored and results showed the effectiveness of the proposed approaches. However, the algorithms and methods presented here represent just the peak of the iceberg of entire problem. In future, we should expect a growing interest on this unexplored and challenging research topic.

References

1. Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *IEEE International Conference on Computer Vision*, 2009.
2. M. Attene, S. Katz, M. Mortara, G. Patane, M. Spagnuolo, and A. Tal. Mesh segmentation - a comparative study. In *IEEE International Conference on Shape Modeling and Applications*, page 7, 2006.
3. Autodesk. 123d catch. http://www.123dapp.com/catch.
4. A. Bartoli. A random sampling strategy for piecewise planar scene segmentation. *Computer Vision and Image Understanding*, 105:42–59, 2007.
5. S. Belongie and J. Malik. Matching with shape contexts. *Content-based Access of Image and Video Libraries, 2000. Proceedings. IEEE Workshop on*, 1:20–26, 2000.
6. P.J. Besl and H.D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on pattern analysis and machine intelligence*, 14(2):239–256, 1992.
7. S. Biasotti, S. Marini, M. Spagnuolo, and B. Falcidieno. Sub-part correspondence by structural descriptors of 3d shapes. *Computer-Aided Design*, 38(9):1002–1019, 2006.
8. Benoît Bocquillon, Adrien Bartoli, Pierre Gurdjos, and Alain Crouzil. On constant focal length self-calibration from multiple views. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
9. I. Borg and P.J.F. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Verlag, 2005.
10. C. Brenner and N. Ripperda. Extraction of facades using rjM-CMC and constraint equations. *Photogrammetric Computer Vision*, 26:155–160, 2006.
11. M. Brown and D. Lowe. Recognising panoramas. In *IEEE International Conference on Shape Modeling and Applications*, volume 2, pages 1218–1225, 2003.
12. Matthew Brown and David G. Lowe. Unsupervised 3d object recognition and reconstruction in unordered datasets. In *International Conference on 3D Digital Imaging and Modeling*, 2005.
13. C. Burges. A tutorial on support vector machine for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
14. B. Bustos, D.A. Keim, D. Saupe, T. Schreck, and D.V. Vranić. Feature-based similarity search in 3d object databases. *ACM Computing Surveys (CSUR)*, 37(4):387, 2005.
15. P. Fua C. Strecha, T. Pylvanainen. Dynamic and scalable large scale image reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

16. N. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *European Conference on Computer Vision*, 2008.
17. D. Capel. An effective bail-out test for ransac consensus scoring. In *British Machine Vision Conference*, volume 10, 2005.
18. U. Castellani, A. Fusiello, and V. Murino. Registration of multiple acoustic range views for underwater scene reconstruction. *Computer Vision and Image Understanding*, 87(1):78–89, 2002.
19. Manmohan Chandraker, Sameer Agarwal, Fredrik Kahl, David Nister, and David Kriegman. Autocalibration via rank-constrained estimation of the absolute quadric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
20. Manmohan Chandraker, Sameer Agarwal, David Kriegman, and Serge Belongie. Globally optimal affine and metric upgrades in stratified autocalibration. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.
21. A.-L. Chauve, P. Labatut, and J.-P. Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1261–1268, 2010.
22. Haifeng Chen and Peter Meer. Robust regression with projection based m-estimators. In *IEEE International Conference on Computer Vision*, pages 878–885, 2003.
23. O. Chum and J. Matas. Randomized ransac with $t_{d,d}$ test. In *British Machine Vision Conference*, pages 448–457, 2002.
24. O. Chum and J. Matas. Matching with prosac-progressive sample consensus. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2005.
25. O. Chum and J. Matas. Optimal randomized ransac. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1472–1482, 2008.
26. Ondřej Chum, Tomáš Pajdla, and Peter Sturm. The geometric error for homographies. *Computer Vision and Image Understanding*, 97(1):86–102, 2005.
27. R. Cipolla, D. Robertson, and E. Boyer. Photobuilder - 3d models of architectural scenes from uncalibrated images. In *IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 25–31, 1999.
28. D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.
29. J.J.F. Commandeur. *Matching configurations*. DSWO Press, Leiden University, Leiden, Netherlands, 1991.
30. O. Cooper, N. Campbell, and D. Gibson. Automatic augmentation and meshing of sparse 3d scene structure. In *IEEE Workshops on Application of Computer Vision*, volume 1, 2005.
31. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2001.
32. N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. 3d urban scene modeling integrating recognition and reconstruction. *International Journal of Computer Vision*, 78(2-3):121–141, 2008.
33. David Crandall, Andrew Owens, Noah Snavely, and Daniel P. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
34. F. Crosilla and A. Beinat. Use of generalised procrustes analysis for the photogrammetric block adjustment by independent models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 56(3):195–209, 2002.
35. G. Cruska, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *European Conference on Computer Vision Workshop on Statistical Learning in Computer Vision*, pages 1–22, 2004.

36. SJ Cunnington and AJ Stoddart. N-view point set registration: A comparison. In *British Machine Vision Conference*, volume 2, 1999.
37. AR Dick, PHS Torr, and R. Cipolla. Modelling and interpretation of architecture from several images. *International Journal of Computer Vision*, 60(2):111–134, 2004.
38. J.R. Diebel, S. Thrun, and M. Brünig. A bayesian method for probable surface reconstruction and decimation. *ACM Transactions on Graphics*, 25(1):39–59, 2006.
39. M.A. Drouin, M. Trudeau, and S. Roy. Geo-consistency for wide multi-camera stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 351–358, 2005.
40. I.L. Dryden and K.V. Mardia. *Statistical shape analysis*. Wiley New York, 1998.
41. Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*, pages 95–101. John Wiley and Sons, 1973.
42. R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley and Sons, 2001.
43. R.P.W. Duin, E. Pekalska, P. Paclik, and D.M.J. Tax. The dissimilarity representation, a basis for domain based pattern recognition? In *Pattern representation and the future of pattern recognition*, pages 43–56, 2004.
44. Lixin Fan and Timo Pylvänäinen. Robust scale estimation from ensemble inlier sets for random sample consensus methods. In *European Conference on Computer Vision*, pages 182–195, 2008.
45. M. Farenzena, A. Fusiello, and R. Gherardi. Structure-and-motion pipeline on a hierarchical cluster tree. In *IEEE International Workshop on 3-D Digital Imaging and Modeling*, 2009.
46. M. Farenzena, A. Fusiello, R. Gherardi, and R. Toldo. Towards unsupervised reconstruction of architectural models. In *Vision, Modeling, and Visualization*, pages 41–50, 2008.
47. M. Farenzena, A. Fusiello, R. Gherardi, and R. Toldo. Automatic structure recovery and visualization. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, volume 18, 2009.
48. O. Faugeras and R. Keriven. Variational principles, surface evolution, pde's, level set methods and the stereo problem. In *IEEE International Summer School on Biomedical Imaging*, 2002.
49. A. Ferreira, S. Marini, M. Attene, M.J. Fonseca, M. Spagnuolo, J.A. Jorge, and B. Falcidieno. Thesaurus-based 3d object retrieval with part-in-whole matching. *International Journal of Computer Vision*, 89(2):327–347, 2010.
50. M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
51. A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed and open image sequencese. In *European Conference on Computer Vision*, pages 311–326, 1998.
52. T. Fromherz and M. Bichsel. Shape from multiple cues: Integrating local brightness information. In *International Conference for Young Computer Scientists*, volume 95, pages 855–862, 1995.
53. P. Fua and Y.G. Leclerc. Object-centered surface reconstruction: Combining multi-image stereo and shading. *International Journal of Computer Vision*, 16(1):35–56, 1995.
54. T. Funkhouser and M. Kazhdan. Shape-based retrieval and analysis of 3d models. In *ACM SIGGRAPH: International Conference on Computer Graphics and Interactive Techniques*, page 16, 2004.

55. T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, and D. Dobkin. A search engine for 3D models. *ACM Transactions on Graphics*, 22:83–105, 2003.
56. Y. Furukawa. *High-fidelity image-based modeling.* PhD thesis, University of Illinois at Urbana-Champaign, 2008.
57. Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *IEEE International Conference on Computer Vision*, pages 80–87. IEEE, 2009.
58. Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1434–1441, 2010.
59. A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti. Globally convergent autocalibration using interval analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(12):1633–1638, 2004.
60. A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.
61. R. Gal, A. Shamir, and D. Cohen-Or. Pose-oblivious shape signature. *IEEE Transaction on Visualization and Computer Graphics*, 13(2):261–271, 2007.
62. S. Ganan and D. McClure. Bayesian image analysis: an application to single photon emission tomography. In *American Statistical Association*, pages 12–18, 1985.
63. P. Gargallo and P. Sturm. Bayesian 3d modeling from images using multiple depth maps. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 885–891, 2005.
64. R. Gherardi, R. Toldo, M. Farenzena, and A. Fusiello. Samantha: towards automatic image-based model acquisition. In *Visual Media Production (CVMP), 2010 Conference on*, pages 161–170, 2010.
65. R. Gherardi, R. Toldo, V. Garro, and A. Fusiello. Automatic camera orientation and structure recovery with samantha. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, pages 38–5, 2011.
66. Riccardo Gherardi, Michela Farenzena, and Andrea Fusiello. Improving the efficiency of hierarchical structure-and-motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1594–1600, 2010.
67. Riccardo Gherardi and Andrea Fusiello. Practical autocalibration. In *European Conference on Computer Vision*, pages 790–801, 2010.
68. Simon Gibson, Jon Cook, Toby Howard, Roger Hubbold, and Dan Oram. Accurate camera calibration for off-line, video-based augmented reality. In *International Symposium on Mixed and Augmented Reality*, pages 37–46, 2002.
69. Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *IEEE International Conference on Computer Vision*, 2007.
70. C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1:285–339, 1991.
71. Google. Earth project. http://www.google.com/earth/index.html.
72. J.C. Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.
73. K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8(2):725–760, 2007.
74. F.R. Hampel, P.J. Rousseeuw, E.M. Ronchetti, and W.A. Stahel. *Robust Statistics: the Approach Based on Influence Functions.* John Wiley & Sons, 1986.
75. F. Han and S.C. Zhu. Bayesian reconstruction of 3d shapes and scenes from a single image. In *Workshop on High Level Knowledge in 3D Modeling and Motion*, volume 2, 2003.
76. R. Hartley, E. Hayman, L. de Agapito, and I. Reid. Camera calibration and the search for infinity. In *IEEE International Conference on Computer Vision*, 1999.

77. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.
78. R. I. Hartley. Estimation of relative camera position for uncalibrated cameras. In *European Conference on Computer Vision*, pages 579–587, 1992.
79. R. I. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.
80. C. Hernández Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004.
81. V.H. Hiep, R. Keriven, P. Labatut, and J.P. Pons. Towards high-resolution large-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1430–1437, 2009.
82. A. Hilton. Scene modelling from sparse 3d data. *Image and Vision Computing*, 23(10):900–920, 2005.
83. D. D. Hoffman and W. A. Richards. Parts of recognition. *Cognition*, 18:65–96, 1987.
84. Q.X. Huang, B. Adams, and M. Wand. Bayesian surface reconstruction via iterative scan alignment to an optimized prototype. In *Eurographics symposium on Geometry processing*, page 223, 2007.
85. Arnold Irschara, Christopher Zach, and Horst Bischof. Towards wiki-based dense city modeling. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.
86. N. Iyer, S. Jayanti, K. Lou, Y. Kalynaraman, and K. Ramani. Three dimensional shape searching: State-of-the-art review and future trend. *Computer Aided Design*, 5(37):509–530, 2005.
87. Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.
88. H. Jin, S. Soatto, and A.J. Yezzi. Multi-view stereo beyond lambert. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–171, 2003.
89. G. Kamberov, G. Kamberova, O. Chum, S. Obdrzalek, D. Martinec, J. Kostkova, T. Pajdla, J. Matas, and R. Sara. 3d geometry from uncalibrated images. In *International Symposium on Visual Computing*, 2006.
90. K. Kanatani. *Geometric Computation for Machine Vision*. Oxford University Press, 1993.
91. Y. Kanazawa and H. Kawakami. Detection of planar regions with uncalibrated stereo using distributions of feature points. In *British Machine Vision Conference*, pages 247–256, 2004.
92. S.B. Kang, R. Szeliski, and J. Chai. Handling occlusions in dense multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–103, 2001.
93. M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics symposium on Geometry processing*, pages 61–70, 2006.
94. Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.
95. Georg Klein and David Murray. Improving the agility of keyframe-based SLAM. In *European Conference on Computer Vision*, pages 802–815, 2008.
96. L Kneip, D Scaramuzza, and R Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

97. L. Kobbelt, J. Vorsatz, and H.P. Seidel. Multiresolution hierarchies on unstructured triangle meshes. *Computational Geometry*, 14(1-3):5–24, 1999.
98. V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, 2006.
99. V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conference on Computer Vision*, pages 8–40, 2002.
100. S. Krishnan, P.Y. Lee, J.B. Moore, and S. Venkatasubramanian. Global registration of multiple 3d point sets via optimization-on-a-manifold. In *Eurographics symposium on Geometry processing*, page 187, 2005.
101. K. Kutulakos. Approximate n-view stereo. In *European Conference on Computer Vision*, pages 67–83, 2000.
102. K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000.
103. I. Laptev, M. Marsza, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
104. Y. Li, H. Zha, and H. Qin. Sapetopics: A compact representation and new algorithm for 3d partial shape retrieval. In *International Conference on Computer Vision and Pattern Recognition*, 2006.
105. Xiaolan Lin, Afzal Godil, and Asim Wagan. Spatially enhanced bags of words for 3d shape retrieval. In *International Symposium on Advances in Visual Computing*, volume 5358, pages 349–358, 2008.
106. David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
107. Q.-T. Luong and O. D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17:43–75, 1996.
108. A. Manessis, A. Hilton, P. Palmer, P. McLauchlan, and X. Shen. Reconstruction of scene models from sparse 3d structure. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 666–671, 2000.
109. J. Matas and O. Chum. Randomized ransac with sequential probability ratio test. In *IEEE International Conference on Computer Vision*, volume 2, 2005.
110. S. J. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.
111. Microsoft. Photosynth. http://www.photosynth.net/.
112. D.D. Morris and T. Kanade. Image-consistent surface triangulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2000.
113. David M. Mount and Sunil Arya. Ann: A library for approximate nearest neighbor searching. http://www.cs.umd.edu/ mount/ANN/, 1996.
114. P. Muller, G. Zeng, P. Wonka, and L. Van Gool. Image-based procedural modeling of facades. *ACM Transactions on Graphics*, 26(3):85, 2007.
115. D. R. Myatt, Philip H. S. Torr, Slawomir J. Nasuto, J. Mark Bishop, and R. Craddock. Napsac: High noise, high dimensional robust estimation - it's in the bag. In *British Machine Vision Conference*, 2002.
116. A. Nakatsuji, Y. Sugaya, and K. Kanatani. Optimizing a triangular mesh for shape reconstruction from images. *IEICE Transactions on Information and Systems*, 88:2269–2276, 2005.
117. PJ Narayanan, P.W. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *IEEE International Conference on Computer Vision*, pages 3–10, 1998.
118. Kai Ni, Drew Steedly, and Frank Dellaert. Out-of-core bundle adjustment for large-scale 3D reconstruction. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

119. Cornea Nicu D., Demirci M. Fatih, Silver Deborah, Shokoufandeh Ali, Dickinson Sven J., and Kantor Paul B. 3d object retrieval using many-to-many matching of curve skeletons. In *IEEE International Conference on Shape Modeling and Applications*, 2005.
120. D. Nistér. Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In *European Conference on Computer Vision*, pages 649–663, 2000.
121. D. Nister. Preemptive ransac for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329, 2005.
122. R. Ohbuchi, k. Osada, T. Furuya, and T. Banno. Salient local visual features for shape-based 3d model retrieval. In *International Conference on Shape Modelling and Applications*, 2008.
123. M. Ovsjanikov, A.M. Bronstein, M.M. Bronstein, and L.J. Guibas. Shape google: a computer vision approach to isometry invariant shape retrieval. In *IEEE International Conference on Computer Vision Workshops*, pages 320–327, 2009.
124. X. Pennec. Multiple registration and mean rigid shapes: Application to the 3d case. In *Image Fusion and Shape Variability Techniques*, pages 178–185, 1996.
125. S. Petitjean. A survey of methods for recovering quadrics in triangle meshes. *ACM Computing Surveys*, 34(2):211–262, 2002.
126. M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *IEEE International Conference on Computer Vision*, pages 90–95, 1998.
127. M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3D reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, 2008.
128. M. Pollefeys, F. Verbiest, and L. Van Gool. Surviving dominant planes in uncalibrated structure and motion recovery. In *European Conference on Computer Vision*, pages 837–851, 2002.
129. J.P. Pons, R. Keriven, and O. Faugeras. Modelling dynamic scenes by registering multi-view image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 822–827, 2005.
130. K Pulli. Multiview registration for large data sets. In *International Conference on 3-D Digital Imaging and Modeling*, pages 160–168, 1999.
131. Till Quack, Bastian Leibe, and Luc Van Gool. World-scale mining of objects and events from community photo collections. In *International Conference on Content-based Image and Video Retrieval*, pages 47–56, 2008.
132. R. Raguram, J.M. Frahm, and M. Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *European Conference on Computer Vision*, pages 500–513, 2008.
133. Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, 1987.
134. S. Roy and I.J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *IEEE International Conference on Computer Vision*, pages 492–499, 1998.
135. S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *International Conference on 3D Digital Imaging and Modeling*, pages 145–152, 2001.
136. H. Saito and T. Kanade. Shape reconstruction in projective grid space from large number of images. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, 1999.
137. G. Salton and M.McGill. *Introduction to modern information retrieval.* McGraw Hill, 1983.

138. S. Savarese, H. Rushmeier, F. Bernardini, and P. Perona. Shadow carving. In *IEEE International Conference on Computer Vision*, volume 1, pages 190–197, 2001.
139. Frederik Schaffalitzky and Andrew Zisserman. Multi-view matching for unordered image sets, or "how do i organize my holiday snaps?". In *European Conference on Computer Vision*, pages 414–431, 2002.
140. K. Schindler and J. Bauer. A model-based method for building reconstruction. In *IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis*, pages 74–82, 2003.
141. P.H. Schonemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.
142. P.H. Schonemann and R.M. Carroll. Fitting one matrix to another under choice of a central dilation and a rigid motion. *Psychometrika*, 35(2):245–255, 1970.
143. S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528, 2006.
144. S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173, 1999.
145. Yongduek Seo, Anders Heyden, and Roberto Cipolla. A linear iterative method for auto-calibration using the dac equation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 880, 2001.
146. S. Shalom, L. Shapira, A. Shamir, and D. Cohen-Or. Part analogies in sets of objects. In *Eurographics Workshop on 3D Object Retrieval*, 2008.
147. Ariel Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27:1539–1556, 2008.
148. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
149. Philip Shilane and Thomas Funkhouser. Selecting distinctive 3d shape descriptors for similarity retrieval. In *International Conference on Shape Modelling and Applications*, 2006.
150. Heung-Yeung Shum, Qifa Ke, and Z. Zhang. Efficient bundle adjustment with virtual key frames: A hierarchical approach to multi-frame structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1999.
151. Ian Simon, Noah Snavely, , and Steven M. Seitz. Scene summarization for online image collections. In *IEEE Internation Conference on Computer Vision*, 2007.
152. S.N. Sinha and M. Pollefeys. Multi-view reconstruction using photo-consistency and exact silhouette constraints: A maximum-flow formulation. In *IEEE Internation Conference on Computer Vision*, volume 1, pages 349–356, 2005.
153. G.G. Slabaugh, W.B. Culbertson, T. Malzbender, M.R. Stevens, and R.W. Schafer. Methods for volumetric reconstruction of visual scenes. *International Journal of Computer Vision*, 57(3):179–199, 2004.
154. N. Snavely, S. M. Seitz, and R. Szeliski. Skeletal graphs for efficient structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.
155. Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM SIGGRAPH: International Conference on Computer Graphics and Interactive Techniques*, pages 835–846, 2006.
156. Drew Steedly, Irfan Essa, and Frank Dellaert. Spectral partitioning for structure from motion. In *IEEE International Conference on Computer Vision*, pages 649–663, 2003.
157. C. V. Stewart. Robust parameter estimation in computer vision. *SIAM Review*, 41(3):513–537, 1999.

158. Charles V. Stewart. Bias in robust estimation caused by discontinuities and multiple structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8):818–833, 1997.
159. C. Strecha, W. Von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.
160. Raghav Subbarao and Peter Meer. Nonlinear mean shift for clustering over analytic manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1168–1175, 2006.
161. R. Szeliski. A multi-view approach to motion and stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 1999.
162. R. Szeliski. Prediction error as a quality metric for motion and stereo. In *IEEE International Conference on Computer Vision*, volume 2, pages 781–788, 1999.
163. G. K. L. Tam and W. H. R. Lau. Deformable model retrieval based on topological and geometric signatures. *IEEE Transaction on Visualization and Computer Graphics*, 13(3):470–482, 2007.
164. K. Tanaka and E. Kondo. Incremental ransac for online relocation in large dynamic environments. In *IEEE International Conference on Robotics and Automation*, pages 68–75, 2006.
165. J. W. Tangelder and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. In *International Conference on Shape Modelling and Applications*, pages 145–156, 2004.
166. T. Tasdizen and R. Whitaker. Higher-order nonlinear priors for surface reconstruction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(7):878–891, 2004.
167. C.J. Taylor. Surface reconstruction from feature based stereo. In *IEEE International Conference on Computer Vision*, pages 184–190, 2003.
168. C.J. Taylor, P.E. Debevec, and J. Malik. Reconstructing polyhedral models of architectural scenes from photographs. *Lecture Notes in Computer Science*, 1065:659–670, 1996.
169. J.M.F. Ten Berge. Orthogonal procrustes rotation for two or more matrices. *Psychometrika*, 42(2):267–276, 1977.
170. Thorsten Thormählen, Hellward Broszio, and Axel Weissenfeld. Keyframe selection for camera motion and structure estimation from multiple views. In *European Conference on Computer Vision*, pages 523–535, 2004.
171. Y. Tiana, Q. Zhub, M. Gerkea, and G. Vosselmana. Knowledge-based topological reconstruction for building façade surface patches. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, volume 18, 2009.
172. R. Toldo, A. Beinat, and F. Crosilla. Global registration of multiple point clouds embedding the generalized procrustes analysis into an icp framework. In *3D Data Processing Visualization and Transmission Conference*, 2010.
173. R. Toldo, U. Castellani, and A. Fusiello. A bag of words approach for 3d object categorization. In *Computer Vision/Computer Graphics Collaboration Techniques*, pages 116–127, 2009.
174. R. Toldo, U. Castellani, and A. Fusiello. Visual vocabulary signature for 3d object retrieval and partial matching. In *Eurographics Workshop on 3D Object Retrieval*, pages 21–28, 2009.
175. R. Toldo, U. Castellani, and A. Fusiello. The bag of words approach for retrieval and categorization of 3d objects. *The Visual Computer*, 26(10):1257–1268, 2010.
176. R. Toldo, F. Fantini, L. Giona, S. Fantoni, and A Fusiello. Accurate multiview stereo reconstruction with fast visibility integration and tight disparity bounding. In *3D Virtual Reconstruction and Visualization of Complex Architectures*, In press.

177. R. Toldo and A. Fusiello. Image-consistent patches from unstructured points with j-linkage. Submitted to IMAVIS journal.
178. R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *European Conference on Computer Vision*, pages 537–547, 2008.
179. R. Toldo and A. Fusiello. Automatic estimation of the inlier threshold in robust multiple structures fitting. In *International Conference on Image Analysis and Processing*, pages 123–131, 2009.
180. R. Toldo and A. Fusiello. Photo-consistent planar patches from unstructured cloud of points. In *European Conference on Computer Vision*, pages 589–602, 2010.
181. R. Toldo and A. Fusiello. Real-time incremental j-linkage for robust multiple structures estimation. In *International Symposium on 3D Data Processing, Visualization and Transmission*, page 6, 2010.
182. P. H. S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:2000, 2000.
183. Philip H. S. Torr and David W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24(3):271–300, 1997.
184. PHS Torr. An assessment of information criteria for motion model selection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 47–52, 1997.
185. A. Treuille, A. Hertzmann, and S. Seitz. Example-based stereo with general brdfs. In *European Conference on Computer Vision*, pages 457–469, 2004.
186. B. Triggs. Autocalibration and the absolute quadric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 609–614, 1997.
187. Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *International Workshop on Vision Algorithms*, pages 298–372, 2000.
188. T. Tung and F. Schmitt. Augmented reeb graphs for content-based retrieval of 3d mesh models. In *IEEE Conference on Shape Modeling and Applications*, pages 157–166, 2004.
189. A. van den Hengel, A. Dick, T. Thormählen, B. Ward, and P.H.S. Torr. Videotrace: rapid interactive scene modelling from video. In *ACM SIGGRAPH: International Conference on Computer Graphics and Interactive Techniques*, volume 26, 2007.
190. A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.
191. Remco C. Veltkamp and Frank B. ter Haar. Shrec 2007 3d retrieval contest. Technical Report UU-CS-2007-015, Utrecht University, Technical Report UU-CS-2007-015, 2007.
192. Maarten Vergauwen and Luc Van Gool. Web-based 3D reconstruction service. *Machine Vision and Applications*, 17(6):411–426, 2006.
193. R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, 2005.
194. G. Vogiatzis, P.H.S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 391–398, 2005.
195. H. Vu, P. Labatut, J. Pons, and R. Keriven. High accuracy and visibility-consistent dense multi-view stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(99):889–901, 2012.
196. Hanzi Wang and David Suter. Robust adaptive-scale parametric model estimation for computer vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(11):1459–1474, 2004.

197. M. Wilczkowiak, P. Sturm, and E. Boyer. Using geometric constraints through parallelepipeds for calibration and 3d modeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(2):194–207, 2005.
198. J Williams and M Bennamoun. Multiple view 3d registration using statistical error models. *Vision Modelling and Visualization*, 99:83–90, 1999.
199. Changchang Wu. Visualsfm: A visual structure from motion system. http://homes.cs.washington.edu/ ccwu/vsfm/.
200. Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M Seitz. Multicore bundle adjustment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
201. L. Xu, E. Oja, and P. Kultanen. A new curve detection method: randomized Hough transform (RHT). *Pattern Recognition Letters*, 11(5):331–338, 1990.
202. Wei Zhang and Jana Kosecká. Nonparametric estimation of multiple structures with outliers. In *Workshop on Dynamic Vision, European Conference on Computer Vision*, volume 4358, pages 60–74, 2006.
203. Z. Zhang. Iterative point matching for registration of free-form curves. *Int. J. Comp. Vis*, 7(3):119–152, 1994.
204. Zhengyou Zhang, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78:87 – 119, 1995.
205. M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiRANSAC algorithm and its application to detect planar homographies. In *IEEE International Conference on Image Processing*, 2005.

Credits

Parts of this thesis are adapted from papers written in conjunction with other authors. I wish to thank them for their contribution and support: Andrea Fusiello, Michela Farenzena, Riccardo Gherardi, Umberto Castellani, Alberto Beinat, Fabio Crosilla, Valeria Garro, Filippo Fantini, Luca Giona and Simone Fantoni. In particular, Chapter 2 comes from [46,47,64,65] and a journal paper which is in preparation, Chapter 3 is based on [176], Chapter 4 derives from [172], Chapter 5 comes from the merging of [173–175], Chapter 6 has been adapted from [178, 179, 181] and Chapter 7 is based on [180] and a journal paper recently submitted [177].

I wish also to thank the examiners and reviewers of my Thesis, Carlo Colombo, George Vogiatzis and Francesca Odone, who provided very constructive feedback.

Acknowledgments

I wish to sincerely thank my advisor Andrea for convincing me to apply for the PhD. I admit I was skeptical when I started. Thanks to him I have learned many things; he guided me during this years integrating my practical inclination with theoretical principles. More than this, we became good friends.

During my PhD I've also had the opportunity to collaborate and discuss with many brilliant people who influenced my research. To mention a few (sorry if I forgot anybody): George, Filippo, Fabrizio, Fernando, Dave and Javier.

A special thanks goes to Simone, Filippo, Luca, Vito and Andrea of the 3Dflow team. I play, eat, discuss and live every day with them and we became really a group of good friends.

Thanks to my family Lauretta, Alfredo, Giantonio and Rino for always supporting me - especially Rino for the generous tips. Sorry mum if I didn't come home or call very often.

I'm grateful to Lilly for accompanying me on my travels and to my *Gatto* for making my life so full of hairs. Examples are shown in Figure 8.1.

Figure 8.1. From left to right, top to bottom: Lilly in Amsterdam, Bratislava, Luxemburg, Paris, Pisa, Strasbourg, Vienna and *Gatto* at home.

Last but not least, I wish to thank Lucia for always standing next to me, especially when I was trapped by my fears. She's my angel and my sunshine [1].

[1] and also my washing machine