Chapter 3
Direct Structure from Motion

This chapter is concerned with robust 3D scene modeling using a novel Structure from Motion algorithm – Direct Pose Registration Structure from Motion (DPR-SfM). The aim is to obtain a high precision texture model of a generic scene acquired using any off the shelf camera undergoing an arbitrary trajectory. The reconstruction algorithm does not require any camera position / attitude information, endowing DPR-SfM with flexibility to be readily used for any type of 3D scene modeling application, both underwater and terrestrial.

3.1 Introduction

We have designed the DPR-SfM algorithm to cope with the most common challenges (see Section 1.3):

- Object occlusions and perspective distortions.
- Invalid image frames due to camera obstructions, motion blur, etc.
- Moving objects.
- Image noise, low contrast and illumination changes (especially in the underwater environment).

DPR-SfM computes directly the pose of the camera without the necessity to recover the inter-frame motion. The structure of the scene is formed by sets of 3D vertices characterized by affine invariant local image descriptors. In this way, by associating image patches extracted from camera views with the 3D vertices, we can recover the camera pose with respect to the scene model. In DPR-SfM, the camera pose is obtained using a novel dual approach, allowing accurate camera pose estimations even in the presence of planar scenes, where most 3D reconstruction algorithms would fail.

Subsequently, the obtained camera poses are used to update the scene model as new features are tracked. Both camera pose estimation and scene

T. Nicosevici & R. Garcia: *Efficient 3D Scene Modeling and Mosaicing*, STAR 87, pp. 39–86.
DOI: 10.1007/978-3-642-36418-1_3

model update steps use robust methods thus reducing the impact of poor camera pose/vertex estimations.

DPR-SfM algorithm works in two stages, as shown in Figure 3.1. First, it uses motion estimation techniques in order to obtain an initial model corresponding to a small subregion of the scene. In the second stage, using the initial model as a "seed", the subsequent camera poses are computed by registering 2D features with 3D vertices in the scene model. For each newly acquired image, once the camera pose is recovered, the scene model is updated by adding vertices corresponding to newly tracked features. In this way, as the camera moves, the model is extended to represent new regions of the scene.

As the data is being processed sequentially, camera pose and scene model estimations are constantly available, enabling the use of DPR-SfM for online applications such as robot navigation and mapping, *in situ* scientific studies, etc.

The remainder of this chapter details the flow of the DPR-SfM algorithm, followed by a discussion on various results that we have obtained by applying the proposed algorithm on outdoor and underwater image sequences. For the ease of the explanation, we illustrate the description of the DPR-SfM algorithm using a simple dataset[1] provided by the Visual Geometry Group of University of Oxford. Figure 3.2 depicts the input set of images of a house model.

3.2 Image Features

Feature tracking is the building block of any sparse 3D reconstruction algorithm. Tracking image features corresponding to a scene region (*i.e.* points, lines, patches, etc.), allows the 3D position of the scene features to be estimated.

Robust feature tracking is crucial to the accurate estimation of both the camera poses and the structure of the scene. Maximizing the number of frames where a given scene feature is tracked improves the precision of its 3D position estimation and increases the number of inter-frame constraints, allowing a higher precision in camera pose estimation.

In order to ensure robust feature tracking in presence of geometric distortions and illumination changes, we have tested various state of the art point and blob feature extractors (see Section 2.1.3): Harris Affine, Hessian Affine, SIFT, SURF and MSER. As expected, point feature extractors generate more dense sets of features than blob feature extractors, providing a better coverage of the scene but having less discriminative power, increasing the chances of mismatching. In contrast, blob extractors produce more sparse but more stable sets of features with higher discriminative power.

[1] `http://www.robots.ox.ac.uk/~vgg/data/dunster/images.tar.gz`

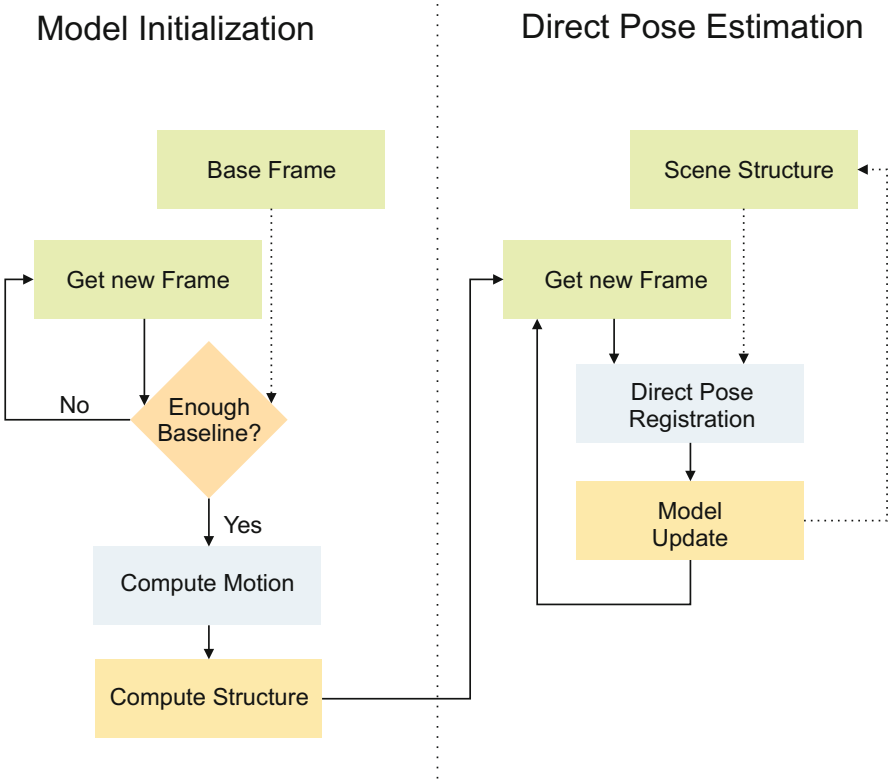


Fig. 3.1 Flowchart of the DPR-SfM algorithm. The *model initialization stage* estimates the baseline between the base frame and a newly acquired frame. If the baseline is wide enough, the motion between the base frame and the acquired frame is recovered. Using the motion, the scene structure is estimated and the algorithm passes to the direct pose registration stage, otherwise the process is restarted using the next acquired frame. In the *direct pose registration stage*, the camera poses are obtained by extracting correspondences between the acquired images and the model. After each new camera pose estimation, the algorithm updates the model with new vertices corresponding to features tracked in the current image. In this way, the scene model grows as the camera surveys new regions of the scene.

In terms of feature descriptors, Harris, Hessian and MSER can be described using both SIFT and SURF, while SIFT and SURF use their own descriptors only.

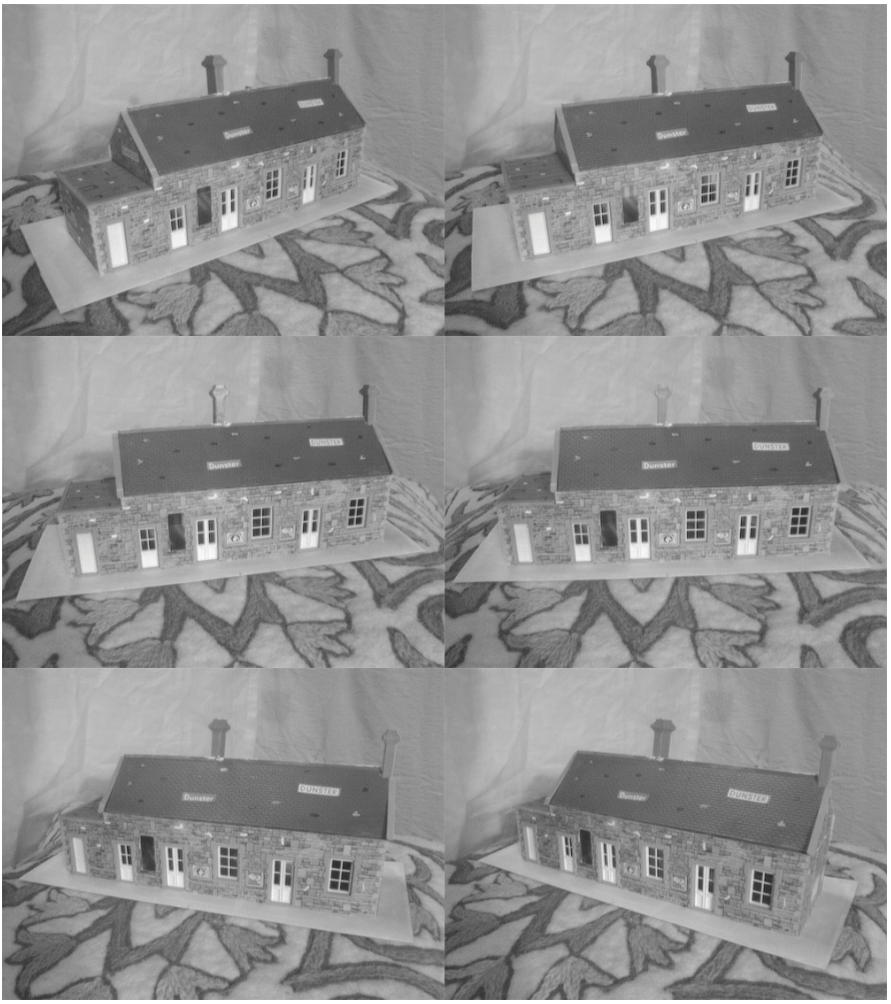


Fig. 3.2 DPR-SfM – House dataset. The input sequence of 6 images captured by a camera undergoing a rotation around a house model.

3.3 Model Initialization

This stage generates a subregion ("seed") of the 3D model corresponding to the first few frames of the image sequence. This initial subregion is required by the second stage that subsequently extends it to the full 3D scene model.

The model is initialized by first fixing the first frame of the sequence as the base frame I_b. The camera pose corresponding to I_b will serve as the global reference frame (world frame) for the entire model. During model initialization, the camera motion between the reference and some image I_i is

computed. I_i is chosen so that the baseline between I_b and I_i is sufficient to ensure a robust motion estimation. The baseline between images is approximated by translation induced by the homography ${}^{b}-0.8mm{}^{b}H_i$ on the image centers, where ${}^{b}H_i$ is a projective homography obtained from feature correspondences between images I_b and I_i (see Section 2.2).

Generally, SfM algorithms use fundamental matrix for camera motion estimation. However, when the scene is planar or the parallax effect is small (*i.e.* small scene depth variations with respect to scene-to-camera distance), the fundamental matrix can be ill-conditioned [67]. In this case, a more robust solution is to use homography-based motion computation. On the other hand, when scene geometry induces significant parallax, homographies cannot correctly model the camera motion. In order to cover both cases, we use a dual approach for motion computation:

Fundamental matrix motion computation. Using the feature correspondences between images I_b and I_i (see Figure 3.3), we estimate the fundamental matrix F_{bi} using RANSAC-based Least Squares (LS) methods[2] [3], with the cost function given by the Sampson distance [153] (see Figure 3.3c,d):

$$E^k_{sampson} = \frac{[(p^k_b)^T F_{bi} p^k_i]^2}{(F_{bi}p^k_i)^2_1 + (F_{bi}p^k_i)^2_2 + (F^T_{bi}x^k_l)^2_1 + (F^T_{bi}x^k_l)^2_2} \tag{3.1}$$

where $(Fp)^2_j$ represents the square of the j-th entry of vector Fp.
The camera rotation R^F_{bi} and translation t^F_{bi} are obtained by Singular Value Decomposition (SVD) of F_{bi} using [74, 94]:

$$F_{bi} = (A^{-1})^T \, \widehat{T}^F_{bi} \, R^F_{bi} \, A^{-1} \tag{3.2}$$

where A is the known camera intrinsic matrix, R is the rotation matrix of the camera and $\widehat{T}$ is the translation skew-symmetric matrix ($\widehat{T}_{[x]} = t \times x$ for any vector x with t representing the camera translation). The approach yields 4 possible solutions (2 translations and 2 rotations). The correct solution is obtained by applying cheirality constraints (*i.e.* reconstructed points must be in front of the camera) [146].

Homography motion computation. From the correspondences of I_b and I_i we compute the homography ${}^{b}H_i$ using RANSAC with the cost function given by:

$$E^H = p^k_b - {}^{b}H_i \, p^k_i$$

where p^k_b and p^k_i represent the k^{th} feature correspondence in images I_b and I_i respectively.

[2] After testing various fundamental matrix estimation methods, RANSAC-based LS method has been adopted as it proved to provide the most robust results in the case of small base lines.

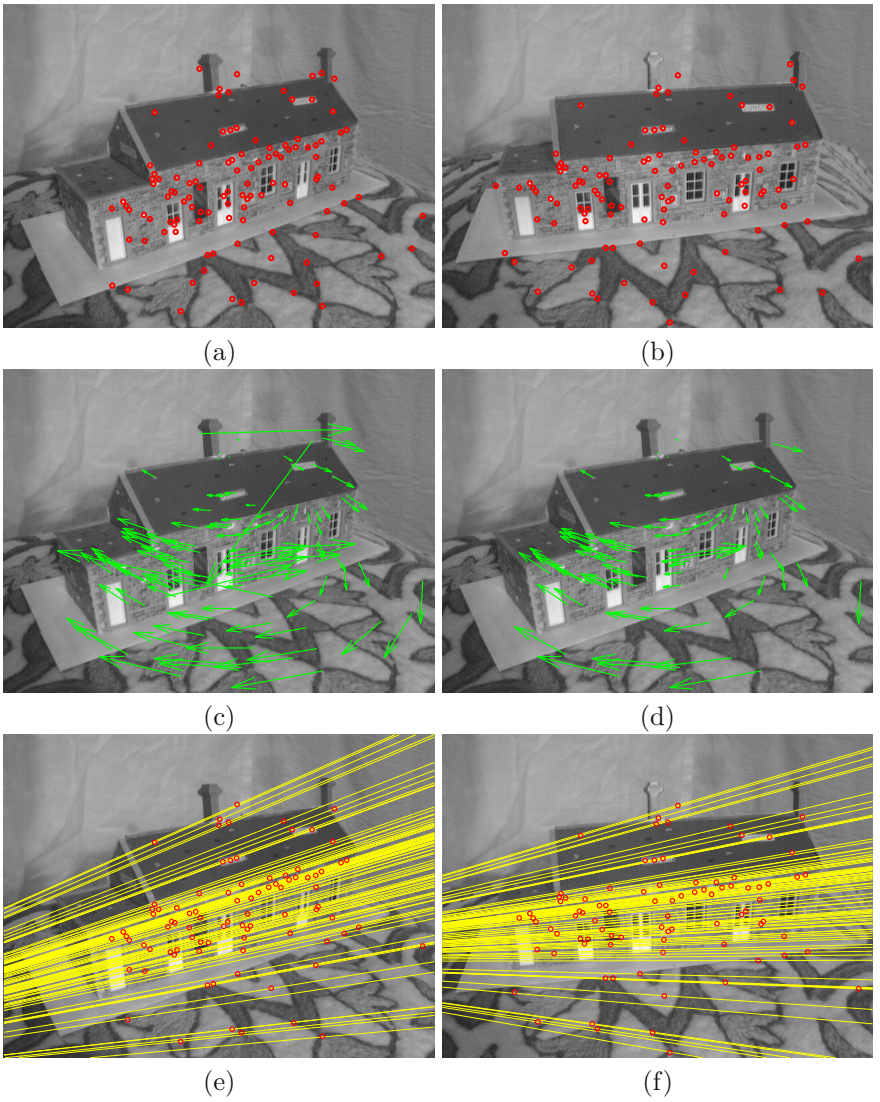


Fig. 3.3 DPR-SfM – Camera motion. When there is enough camera motion between the base frame (left column) and the current frame I_i (right column), the pose is computed. (a) and (b) show the extracted image features. (c) show the initial feature disparity after matching, (d) shows the feature disparity after outlier rejection, in this case using F. (e) and (f) illustrate the epipolar lines for I_b and I_i, respectively.

By normalizing the homography between I_b and I_i:

$$^b\widehat{H}_i = -A^{-1}\ {}^bH_i\ A$$

we obtain the camera camera rotation R^H_{bi} and translation t^H_{bi} using SVD [35]:

$$^b\widehat{H}_i = R^H_{bi} - t^H_{bi}\ \eta^T$$

where η is the normal of the scene plane. This type of decomposition raises two solutions. The correct one corresponds to the plane normal pointing towards the camera.

Between the two solutions $(R^F_{bi},\ t^F_{bi})$ and $(R^H_{bi},\ t^H_{bi})$, we choose the most accurate one. This is done by estimating the 3D position of the image features with respect to each solution using LS Intersection. Then, the accuracy of the camera motion is given by the back-projection error:

$$E_{bi} = \sum_{k=1}^{N}(\|p^k_b - \Pi_b P^k\| + \|p^k_i - \Pi_i P^k\|) \tag{3.3}$$

where, p^k_b and p^k_i are the corresponding image features in images I_b and I_i respectively; P^k is the estimated 3D position of kth feature.

The solution corresponding to the smallest retrojection error E_{bi} is chosen and the corresponding set of 3D points is used to initialize the scene model.

In order to complete the set of camera poses, we recover the pose of the cameras corresponding to the intermediate frames between I_b and I_i by directly registering the camera views with the 3D model (Section 3.5). Figure 3.4 illustrates the initial model for the House dataset, corresponding to the first three frames.

3.4 Scene Model

The scene model was designed to contain geometric along with photometric information. The geometry of the scene is described in terms of 3D vertices, defined by their position $[X\ Y\ Z]^T$ with respect to a common world frame. Photometrically, the vertices are characterized by descriptors obtained from their corresponding image feature descriptors.

The image descriptor vectors can be seen as noisy measurements of the image gradient within a feature patch. As the features are tracked, multiple measurements of the same patch are obtained. Hence, we improve feature tracking by modifying the similarity measurement in eq. (2.1) to include multiple observations:

$$s(\mathbf{f}^k, f^k_i) = \|\frac{\sum f^k}{n} - f^k_i\| \tag{3.4}$$

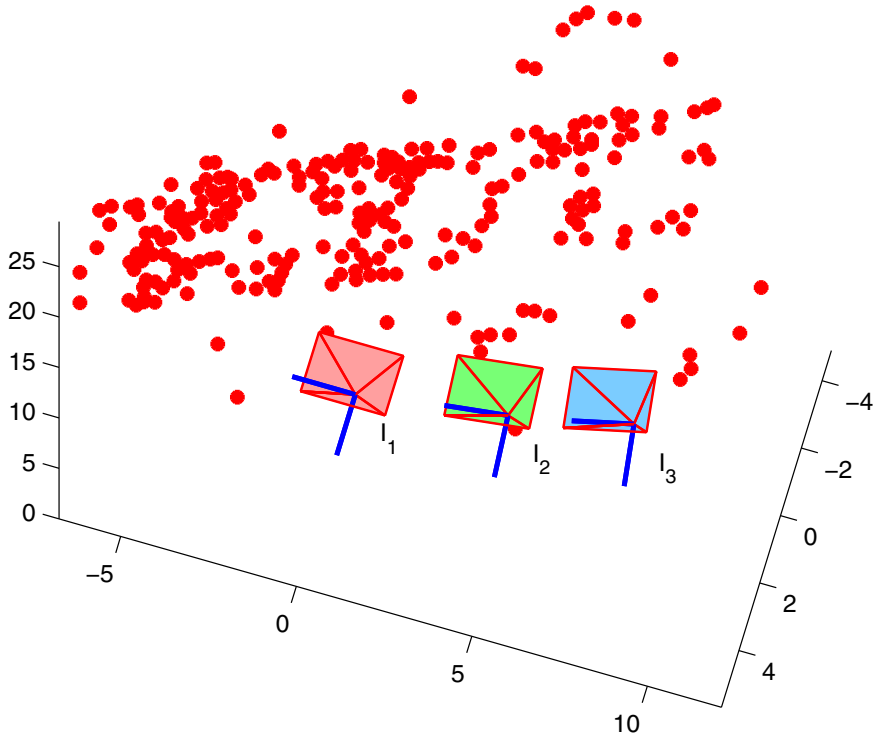


Fig. 3.4 DPR-SfM – Initial model. Initial 3D scene model (red dots) and camera poses. The model initialization was done using frame 1 (red) and 3 (blue). Camera pose for frame 2 (green) was obtained by direct registration.

where V^k represents the descriptor vector of vertex[3] k, and n represents the total number of images where the vertex was tracked. Using such a descriptor representation allows for more stable vertex tracking in presence of image noise, illumination changes and projective distortions.

When associating vertices with image features using eq. (3.4), we impose distance thresholds for $s(V^k, v_i^k)$ to reduce the number of outliers. The threshold values were established empirically. As all the feature descriptors are normalized, the established thresholds proved to provide optimum results (for both SIFT and SURF descriptors) in all the test sequences.

In practice, using a direct approach for feature association in eq. (3.4) involves a high computational load. Depending on the resolution and the feature extractor type, an image can yield thousands of features that have to

[3] Here, we use the term *vertex* to express a set of image features corresponding to the same scene point. The actual 3D position of the vertex does not need to be calculated at this point.

be associated with tens of thousands of features from each feature group[4] in the scene model. We highly reduce this computational load by using a k-dimensional tree (kd-tree) approach. Using kd-trees, we hierarchically decompose the scene model feature space into a relatively small number of subregions so that no region contains too many features [4] (see Figure 3.5). This provides a fast way to access any scene model feature. In order to associate an image feature, we traverse down the hierarchy until we find the subregion containing the match and then scan through the few features within the subregion to identify the correct match. In the implementation that we used [113], we obtained a decrease in the computational time with respect to classical NN of about 5 times.

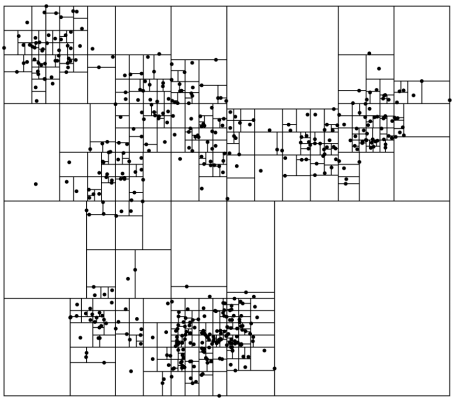


Fig. 3.5 Kd**-tree partitioning.** The k-dimensional feature space is hierarchically partitioned in subregions containing a small amount of features.

3.5 Direct Camera Registration

This section deals with the direct recovery of the camera pose with respect to the scene model, without the need of any *a priori* information on camera motion or pose. This way, the robustness of the DPR-SfM algorithm is increased, allowing it to naturally deal with camera occlusions, loop closures and position estimation errors.

In Section 3.4 we explain how to associate image and scene model features. From this, we obtain 3D-to-image correspondences with the aim of recovering camera pose (R_i, t_i) with respect to the world frame (see Figure 3.6). The camera pose is obtained using RANSAC with the cost function:

$$E_i = \sum_{k=1}^{N} \|p_i^k - \Pi_i P^k\| \tag{3.5}$$

[4] DPR-SfM supports simultaneous use of different feature types. In the scene model, the features are grouped by extractor/descriptor.

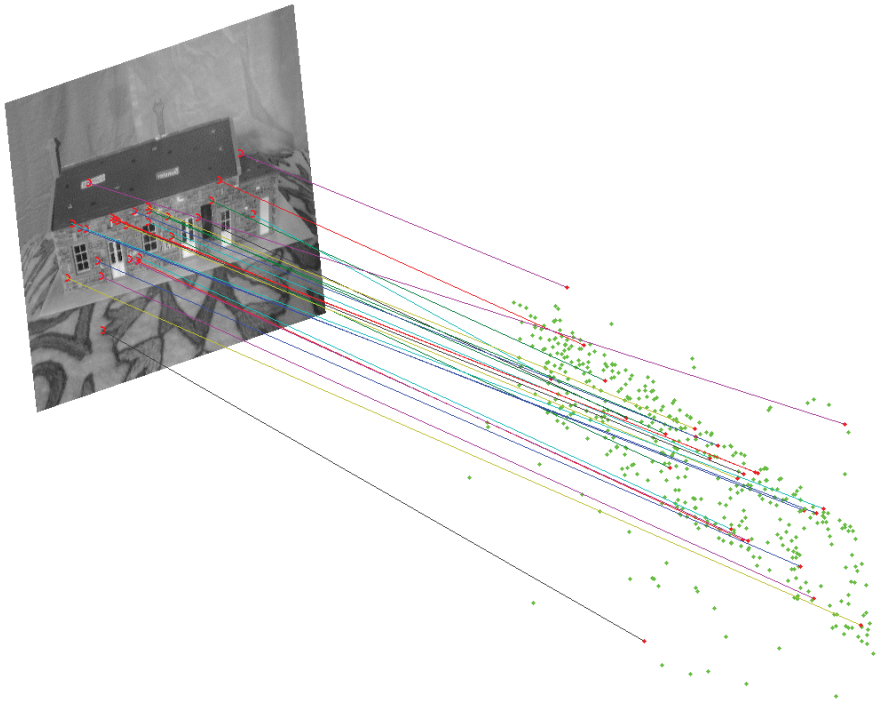


Fig. 3.6 DPR-SfM – Direct pose registration. Example of pose registration of frame 4: the image features are associated with the scene model. The camera pose is estimated using the projection matrix.

In order to robustly cope with different types of scenes, we propose a novel dual approach for camera pose recovery (similar to the one described in the Section 3.3): *(i)* if the scene region seen in the current image has enough parallax, we use projective matrix to recover the camera pose; *(ii)* if the scene region is planar or close to being planar, the projection matrix is ill-conditioned [67], in which case we use a homography approach. In order to determine the planarity of the scene, for each RANSAC sample, we fit a plane L to the 3D vertices using a LS method. If the distance between the plane L and all the other 3D vertices (from the 3D-to-image correspondences) is small enough, we consider the scene region as being planar. The method is summarized in Table 3.1. The camera pose estimation methods are detailed hereafter:

Projection matrix-based. Provided the set of 3D-to-image correspondences, we obtain the projection matrix Π_k using DLT. From equation (2.2), we obtain the camera pose (R_k, t_k).

Homography-based. We compute the planar transformation ${}^{i}H_{L}$, so that:

$$p_i^k = {}^{i}H_L \cdot p_L^k$$

where p_L^k is the projection of P_k onto plane L. Applying SVD on ${}^{i}H_{L}$, we obtain the relative transformation $({}^{i}R_L, {}^{k}t_L)$ between the plane L and the camera. Thus, the pose of the camera is obtained from:

$$t_i = t_L \cdot {}^{i}R_L + {}^{i}t_L$$

$$R_i = {}^{i}R_L \cdot R_L$$

with t_L and R_L representing the pose of plane L in the world coordinate system.

Once a (R_k, t_k) have been obtained using the RANSAC dual method, the camera pose is further adjusted using a LS method that minimizes the back-projection error shown in eq. (3.5).

Table 3.1 Camera pose recovery process

1. While not enough RANSAC samples.
2. Choose randomly a set of 3D-to-image correspondences.
3. Fit a plane L to the 3D vertices from the set.
4. Check if the other vertices (corresponding to I_k) lay close to plane L.
5. If yes, compute R and t based on the homography using the set of correspondences.
6. If no, compute R and t based on the projection matrix using the set of correspondences.
7. Go to 1.

3.6 Model Update

As the camera moves, the DPR-SfM algorithm updates the scene model as new features are extracted and tracked, generating new 3D vertices. This section discusses the model updating process along with the outlier management.

As new images are fed to the DPR-SfM algorithm and the image features are associated with scene model features (see Section 3.4), three scenarios arise:

Image features matched with model features with known 3D position. These feature associations are used to recover the camera pose, as explained in Subsection 3.5. The outliers are detected by reprojecting the 3D vertices into the image (eq. (3.5)). Vertices with a reprojection error higher than a pre-established threshold are eliminated. Inliers are added to the model to create new constraints. Every time an additional image feature is associated with a particular 3D vertex, the position of the vertex is refined, taking advantage of this new constraint. The refinement is done by minimizing the sum of the reprojection errors E_k in all the images where the vertex was tracked:

$$E_k = \sum_{i=1}^{M} \|p_i^k - \Pi_i P^k\| \tag{3.6}$$

Image features matched with model features with no 3D position. Adding new image features to already existing model features provides additional information that ultimately leads to the recovery of 3D vertex position. In this case, the back-projection approach cannot be used for outlier rejection as the 3D position of the vertex is unknown at the time. Alternatively, we use a fundamental matrix based approach. For each image feature p_i^k we choose a feature p_i^l from its associated feature track so that their corresponding camera poses (R_k, t_k) and (R_l, t_l) have the widest possible baseline (the wider the baseline the more discriminative the process). From the relative transformation between the two cameras (R_{kl}, t_{kl}) we compute the fundamental matrix F, as shown in equation (3.2). This allows us to use the Sampson distance shown in eq. (3.1).

If the image feature p_i^k yields a distance $E_{sampson}$ larger than a pre-established threshold, it is regarded as an outlier and the feature association is eliminated, otherwise it is added to the model. When enough views of a feature are available, the position of the corresponding vertex is calculated using a multi-view factorization approach [100]. The vertex position is then refined using a LS method (see eq. (3.6)).

Unmatched image features. If the image features could not be consistently associated to any model features, they are used to generate new feature entries in the model.

Since not all model features are tracked reliably enough to produce accurate 3D vertices, the model is constantly checked and features that do not provide a consistent tracking are eliminated in order to minimize the unnecessary clutter of the model.

Figure 3.7 illustrates the final 3D model of the House sequence along with the recovered camera poses.

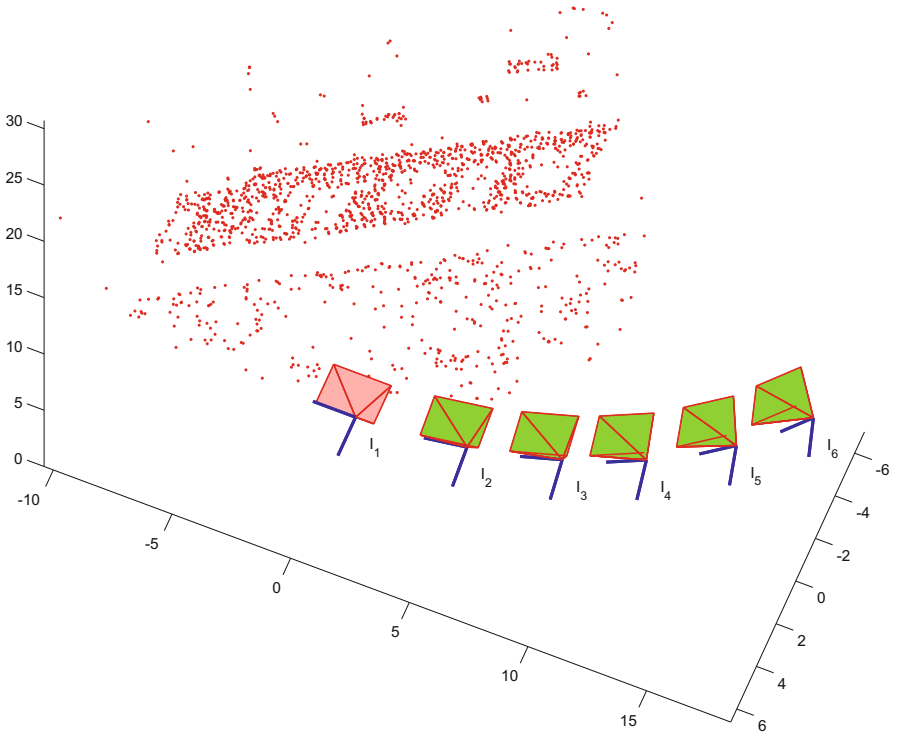


Fig. 3.7 DPR-SfM – Final model. 3D model of the House sequence containing $\simeq 2,000$ vertices (red dots) along with the camera poses. The first camera pose (shown in red) defines the global coordinate system of the model.

3.7 Ortho-mosaicing and 3D Representation

A great deal of underwater studies require the assessment of 2D visual maps (see Section 1.2). When the regions of interest contain significant 3D relief, classical mosaicing techniques prove inaccurate due to the parallax effect. We propose a solution to this shortcoming, where the 3D scene model is ortho-projected into a plane. The result is a virtual "high-altitude" view of the scene called *ortho-mosaic*. In other words, an ortho-mosaic is the equivalent to a 2D mosaic acquired from a camera located far from the scene.

The ortho-mosaic is obtained by first creating a continuous model of the scene. The continuous model is defined by triangular patches with the corners defined by the 3D vertices [6]. Within the patches, we can obtain the 3D position of any point using linear or cubic interpolation[5].

[5] For natural and unstructured scenes, where the shapes are usually smooth, cubic interpolation provides the best results.

An ortho-projection plane O is then chosen to have the same tilt as the average tilt of continuous model. This maximizes the projection area, providing the highest level of mosaic detail. Then, all the patches are mapped onto the destination plane along projection rays perpendicular to plane O (see Figure 3.8).

The plane O is digitized based on a predefined resolution; each point p_O^k on the grid corresponds to a pixel in the ortho-mosaic. In order to render the mosaic, we define the following transformation relating each point p_O^k to a corresponding point p_i^k from the original images:

$$p_i^k = \Pi_i T_n p_O^k \tag{3.7}$$

where T_n is the ortho-projection transformation of the patch $[P_1\ P_2\ P_3]$ and Π_i is the camera projection matrix corresponding to frame I_i, as shown in Figure 3.8a.

Figures 3.9a and 3.10 illustrate the results of the ortho-mosaicing process for the the House sequence and an underwater scene respectively.

For the cases where 3D information is required, the ortho-mosaic is used as texture for rendering the 3D surface. The result is a complete model that includes both geometrical and photometrical information of the scene. In Figure 3.11 we show two views of the 3D model of the underwater scene. Here, the surface was obtained by using cubic interpolation. In the case of the House scene, illustrated in Figure 3.9b, linear interpolation is more suitable.

3.8 Experimental Results

In this section, we discuss the performance of the DPR-SfM algorithm. The evaluation focused on two main aspects: *(i)* the accuracy of both scene model and camera pose estimations and *(ii)* the robustness of the algorithm when faced to common challenges such as: illumination changes, shadows, scattering, low contrast images, moving objects, specular surfaces, obstructions, objects with complex geometry, etc.

DPR-SfM has been successfully tested under various conditions, briefly discussed hereafter:

- We applied the algorithm on image sequences captured using both still and video cameras. The algorithm successfully coped with both high overlap images in video sequences and low overlap images in sequences acquired by still cameras. The DPR-SfM provides accurate estimations even in the case of temporarily static cameras, where most SfM algorithms would fail. The minimum overlap between images is given by the minimum number of views where a feature needs to be tracked before its 3D position is estimated, which can be set by the user. We generally use a minimum of 3 views per each tracked feature for redundancy.

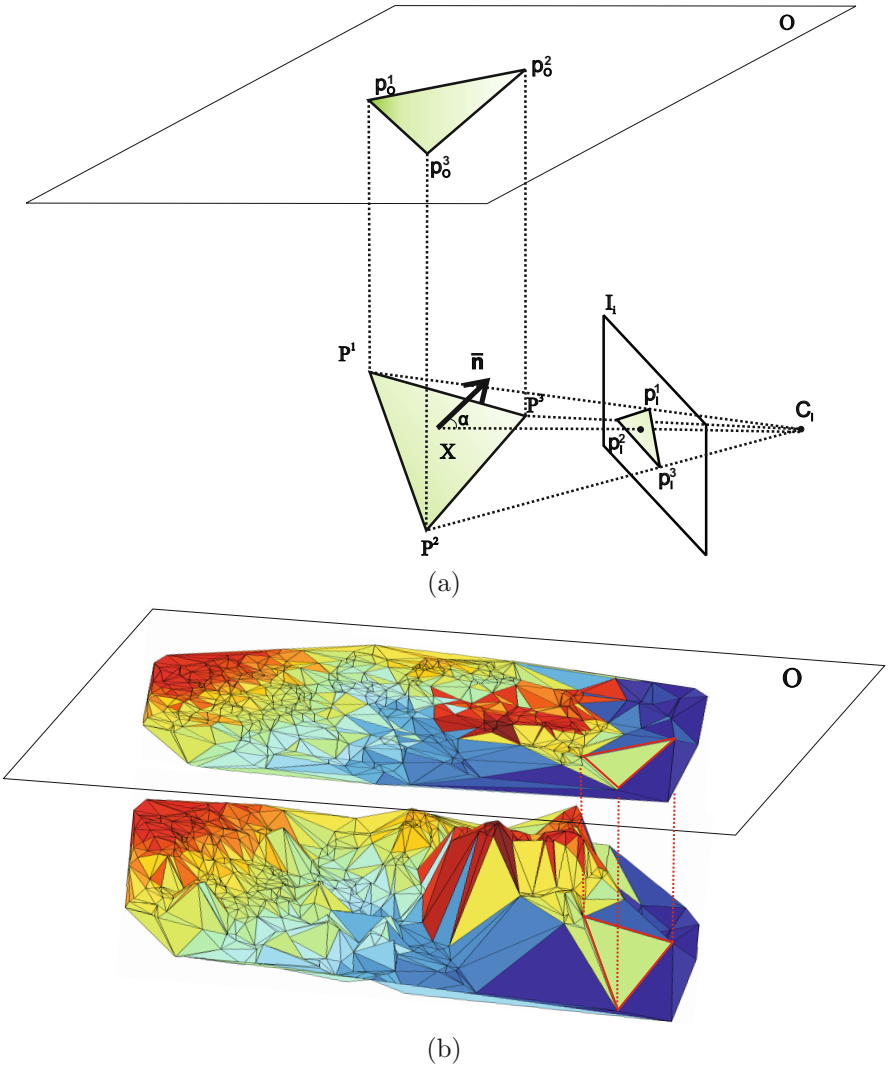
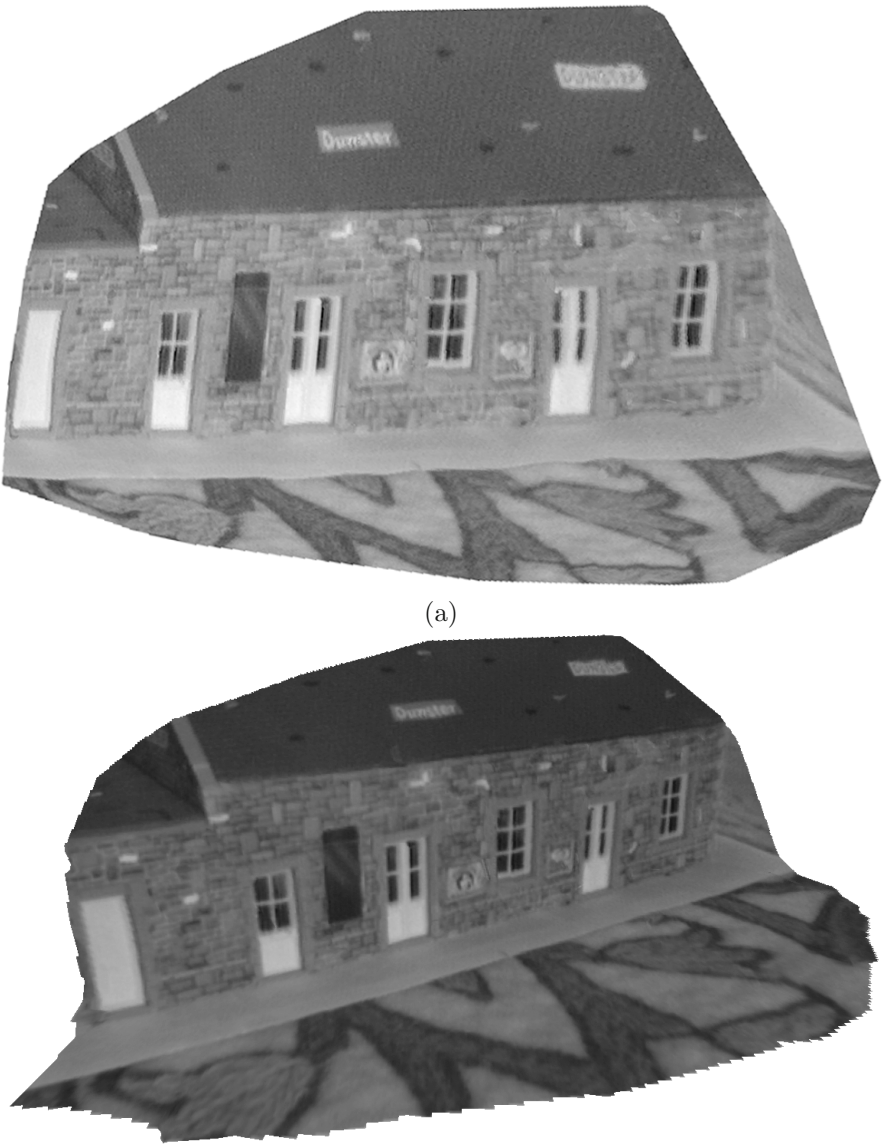


Fig. 3.8 Principles of ortho-mosaicing. In figure (a) The model patch $[P_1\ P_2\ P_3]$ is ortho-projected onto the plane O. The corresponding ortho-mosaic patch $[p_O^1\ p_O^2\ p_O^3]$ is rendered using eq. (3.7) from image I_i, chosen so that the angle α between the patch normal and the camera principal axis is minimum. In (b), for clarity purposes, we show the ortho-projection of a seafloor model containing a coral-reef formation (Bahamas dataset). This model will be discussed in detail in Section 3.8.



(a)

(b)

Fig. 3.9 Model of the house scene. (a) shows the ortho-mosaic of the house. In this case, there is no gain in using the ortho-mosaic since all the camera views cover the entire scene. (b) is a view of the textured model; the 3D surface was generated using linear interpolation, which is more suited for structured scenes, containing planes and straight edges.

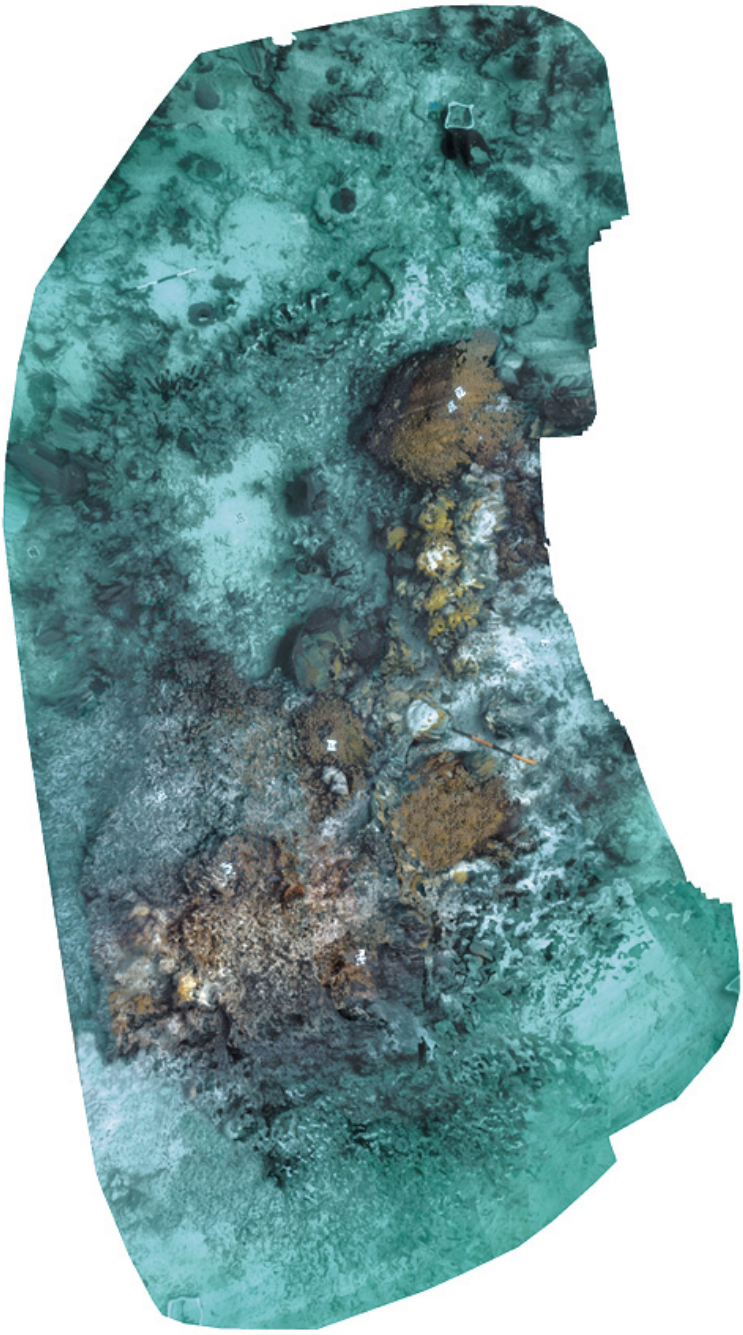
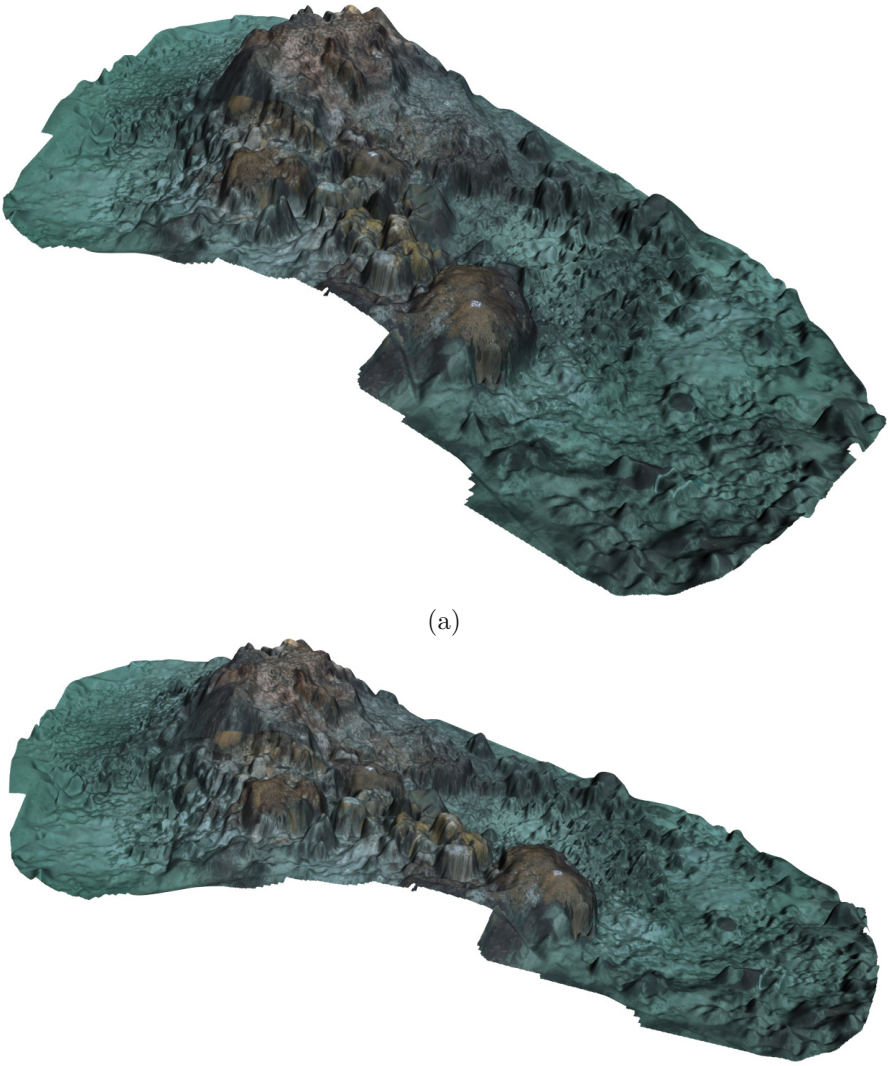


Fig. 3.10 Ortho-mosaic of an underwater scene. The rendered mosaic simulates a high-altitude view of the scene, depicting coral-reef formations.



(a)

(b)

Fig. 3.11 3D model of an underwater scene. Two views of the underwater scene model obtained by texture rendering the ortho-mosaic on the 3D surface. Here the surface was obtained using cubic interpolation.

- We tested the algorithm in the presence of occlusions and pose estimation failures (*e.g.* excessive motion blur). The pose of the camera was correctly estimated immediately after the situation disappeared. From our experiments, we have concluded that the camera pose can be correctly estimated, if there is at least $\sim 20\%$ overlap between the 3D model and the images.
- The conducted experiments included sequence acquisitions under extreme lighting conditions, obtaining accurate results: sun-flickering in shallow waters, low lighting and increased turbidity/scattering, strobe/focus lighting in deep waters.

In the discussion that follows, we generally assess the accuracy of the DPR-SfM algorithm on absolute basis as, to the best of our knowledge, there are no freely available SfM algorithms for comparison that can cope with such large scale reconstructions.

All the data-sets presented here were acquired using various off the shelf cameras, undergoing a random trajectory with no constraints. For all the sequences, we assume that the internal parameters of the cameras are known and do not change throughout the image acquisition (*i.e.* no zooming), and the radial distortion is corrected. The estimation of the camera internal parameters and radial distortion parameters were obtained using a checkerboard pattern and Bouguet's camera calibration toolbox [12].

3.8.1 Car Scene

In this sequence we used synthetically generated images, allowing the usage of ground truth in order to quantify the accuracy of the DPR-SfM on both camera pose and scene geometry estimations.

The scene, comprised by a parked car in front of a building, was chosen to incorporate common challenges in urban environments: occlusions, object transparency, light reflections, shadows, uniform textures, etc. The rendering of the scene was carried out using ray-tracing as it is capable of producing very high degree of photorealism [142]. Ray-tracing generates images by tracing the path of light through pixels in an image plane [159], accurately modeling light alterations (reflections, shadows, transparency).

The sequence consists of 20 frames with $1,024 \times 1,024$ pixels, captured from a camera undergoing a translation motion along the building facade with a slight panning (see Figure 3.12 for some examples). The length of the translation is 10m with a mean distance between the camera and scene (the facade of the building) of $\simeq$ 9m. In order to accurately compare the results with the ground truth, we fix the scale of the model by fixing the first two camera poses in the initialization step. The following camera poses are estimated by direct registration with the model (see Figure 3.13).

For comparison purposes, we used 4 types of feature extractors: Harris, Hessian, SIFT and SURF. The processing time for the sequence was

Fig. 3.12 Car Scene – Input images. Synthetic images generated using ray-tracing rendering. Here, we illustrate 4 of the 20 frames showing some of the challenges: specular objects (car body and building windows) induce inter-reflections, irregular illumination due to shadows (garage door, doors and pavement), transparency (car windows), etc.

$\simeq 14 mins$[6]. A detailed description of execution times is presented in Table 3.2. We processed this sequence using both NN and Approximated Nearest Neighbor (ANN). The use of ANN provides a significant gain in computation time (see Figure 3.14): NN times are quadratic in the number of features while ANN times are linear.

[6] The DPR-SfM algorithm was implemented in *Matlab*, partially using C++ routines. All the experiments presented in this work were executed on an Intel Core Duo 2.13 GHz 64-bit platform.

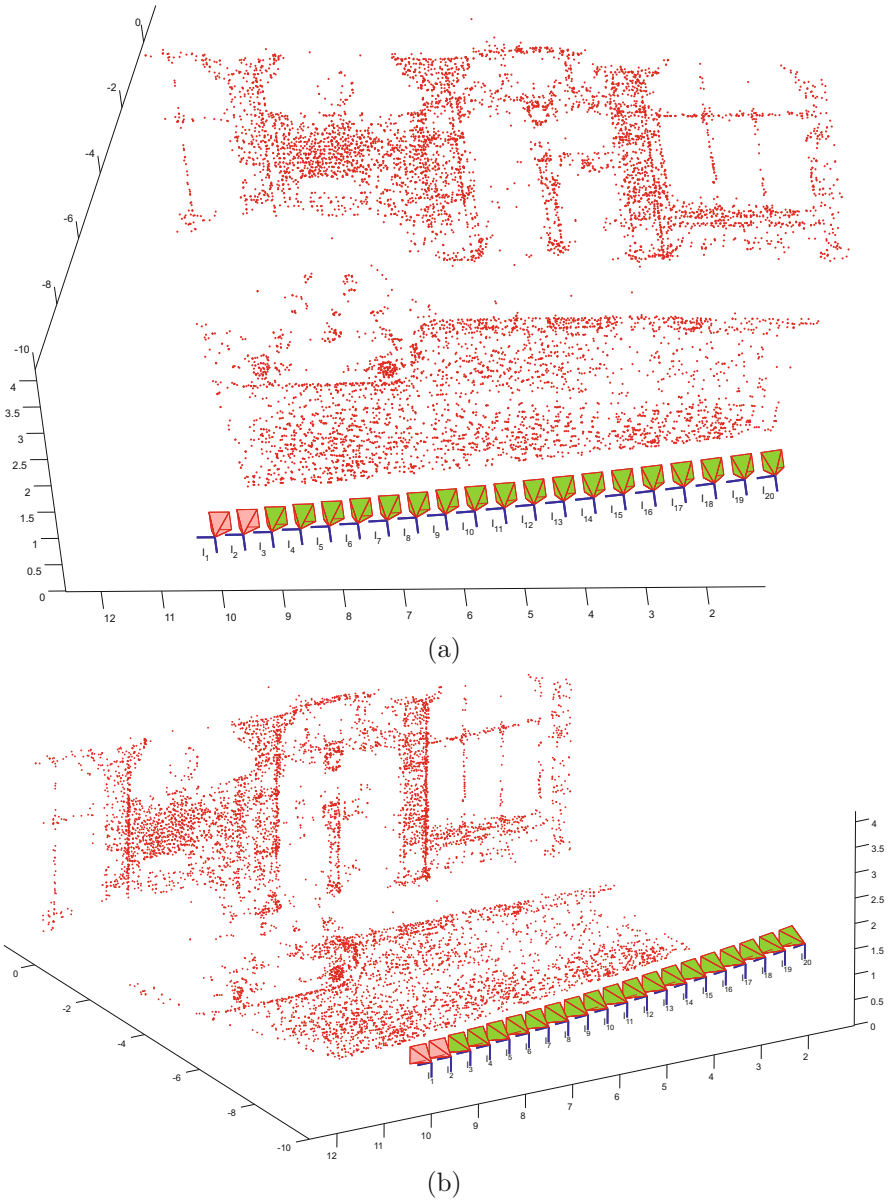


Fig. 3.13 Car Scene – 3D model. Two views of the 3D model containing 9,800 vertices – 2,900 Harris, 2,600 Hessian, 2,400 SURF and 1,800 SIFT. The first two camera poses (shown in red) were fixed in order to recover the scale. The remaining camera poses (green) were estimated by direct registration along with scene model (red dots).

Table 3.2 Car Scene – Processing time. Average processing time for each step ($seconds/frame$).

Feat. Extraction	Feat. Matching (ANN)	Camera Pose	Vertex Position
40.1	2.1	0.2	0.6

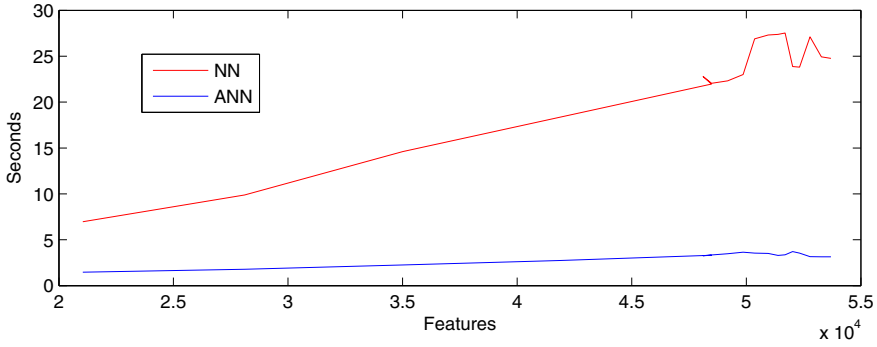


Fig. 3.14 Car Scene – Scene feature matching time. Comparison of average times for matching image and scene features vs. number of features in scene. The number of features in the image is constant (12,000). Using ANN decreases drastically the computation times.

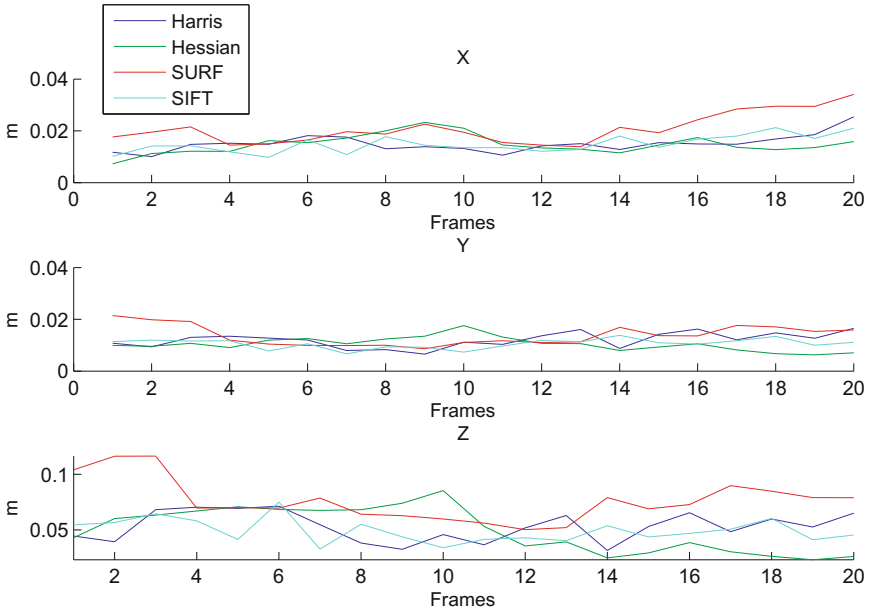
The resulting scene model is illustrated in Figure 3.13:

- Invalid vertices formed by reflective surfaces are removed.
- Facade regions partially occluded by the car are correctly modeled (*e.g.* left of the building entrance).

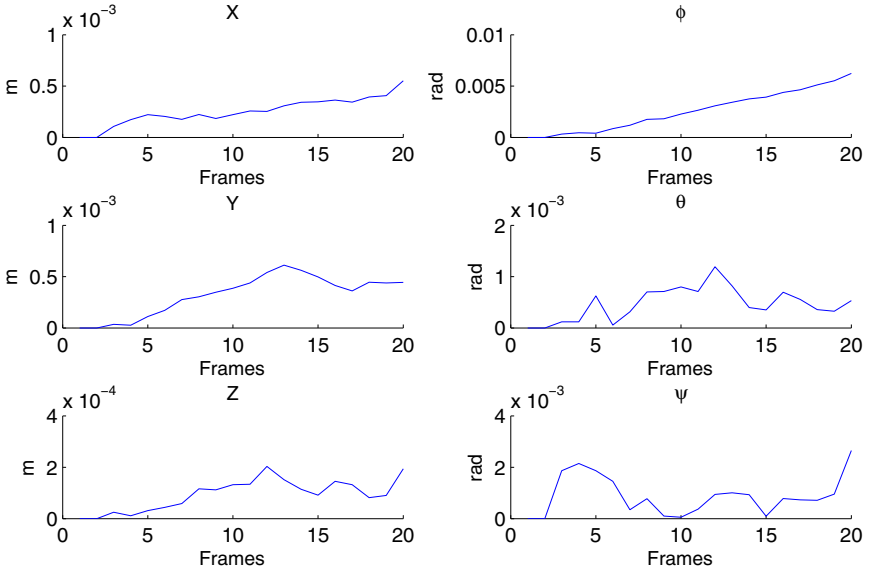
The ray-tracing software was modified to generate the ground truth 3D position of the points in the scene corresponding to each pixel in the rendered images. Knowing the position of the extracted visual features, the accuracy of the model is quantified by comparing the vertex position estimations with the ground truth.

Figure 3.15a illustrates the average residuals (XYZ) for the vertices generated by each feature extractor. While very similar, SIFT and SURF have slightly greater residuals than Harris and Hessian, due to the nature of the extractors (see Section 2.1.3). The evolution of error in camera pose estimation is shown in Figure 3.15b.

In ideal conditions (absence of noise, distortions, blurring, etc.), both scene geometry and camera pose estimations are accurate and the error accumulation (drifting) is very small. Additionally, we want to test the robustness and accuracy of DPR-SfM for realistic scenarios. For this, we use a Monte Carlo test by adding noise to image features, aiming to:



(a)



(b)

Fig. 3.15 Car Scene – Reconstruction errors. Figure (a) shows the vertex position residuals by frames, for each feature extractor. The extractors yield comparable results, with small error accumulation. In (b) we represent the error in camera pose. The residuals in both position and attitude are very small with a slow error accumulation.

- Assess the accuracy of the model and camera pose estimations in presence of noise.
- Robust camera pose estimation and vertex position estimation use a pre-established threshold ρ for outlier rejection (see Sections 3.5, 3.6). We test how this threshold affects the DPR-SfM accuracy.

As we consider feature localization errors to follow a normal distribution, we use a zero-mean gaussian noise with a known standard deviation σ. For each test, we fix the value of ρ and we generate the model with increasing values of σ until a valid model cannot be generated. We use two values for ρ: 1.5 and 2.5 (values typically used in DPR-SfM). The errors in scene model and camera pose are given by ε_v, ε_p and ε_a, where ε_v is the average error in vertex position estimation and ε_p is the average error in camera position estimation (both error measurements are given by the average Euclidian distance). The error in camera attitude estimations ε_a is given by the average of absolute differences over all the rotations:

$$\varepsilon_a = \frac{\sum_{i=1}^{N} |\phi_i - \overline{\phi}_i| + |\theta_i - \overline{\theta}_i| + |\psi_i - \overline{\psi}_i|}{3N}$$

where $(\overline{\phi}_i\ \overline{\theta}_i\ \overline{\psi}_i)$ is the estimated orientation and $(\phi_i\ \theta_i\ \psi_i)$ is the ground truth orientation for camera pose i; N is the total number of frames.

Table 3.3 details the results of the Monte Carlo tests. The noise in image features has little impact on both model and camera pose estimations,

Table 3.3 Car Scene – Monte Carlo test results. The results for two values of ρ. The values for ε_v and ε_p are expressed in $m\cdot 10{-}3$ and ε_a is expressed in $rad \cdot 10^{-3}$. *Vert./fr.* represents the average number of vertices registered in each frame.

	$\rho = 1.5$				$\rho = 2.5$			
σ	ε_v	ε_p	ε_a	vert./fr.	ε_v	ε_p	ε_a	vert./fr.
0	47.9	1.3	0.03	2226	61.2	3.0	0.05	2449
0.2	48.9	1.4	0.10	2211	61.9	3.2	0.14	2447
0.4	48.8	1.8	0.16	2148	63.6	3.3	0.18	2446
0.6	50.5	3.3	0.21	1939	65.5	4.0	0.23	2435
0.8	51.8	4.2	0.25	1598	66.0	4.4	0.27	2416
1.0	57.9	4.1	0.31	1285	66.4	3.7	0.29	2291
1.2	63.3	7.5	0.7	970	73.8	6.2	0.32	2309
1.4	69.1	14.5	0.81	848	74.0	5.1	0.34	2180
1.6	65.7	19.6	0.85	329	81.8	4.7	0.43	2035
1.8	–	–	–	–	85.6	6.0	0.45	1857
2.0	–	–	–	–	90.7	7.0	0.50	1671
2.2	–	–	–	–	106.6	7.8	0.61	1471
2.4	–	–	–	–	124.6	7.9	0.72	1237

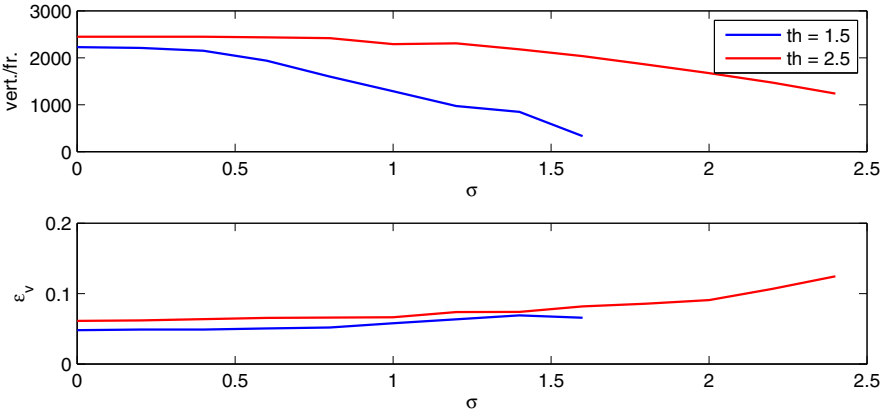


Fig. 3.16 Car Scene – Feature evolution in Monte Carlo test. The average number of vertices drops as the noise level increases (Top Figure). Using a more relaxed threshold keeps a larger number of vertices but slightly decreases the accuracy of the vertices (Bottom Figure).

especially when a low threshold is used. However, as the noise level increases, the use of a very restrictive threshold highly reduces the number of vertices (see Figure 3.16). This affects the camera registration precision, ultimately leading to the impossibility to generate a valid model.

Figure 3.17 illustrates the distribution of the noise in the image features for each threshold. The DPR-SfM can generate a valid scene model even in the presence of an overwhelming number of outliers (more than 60%).

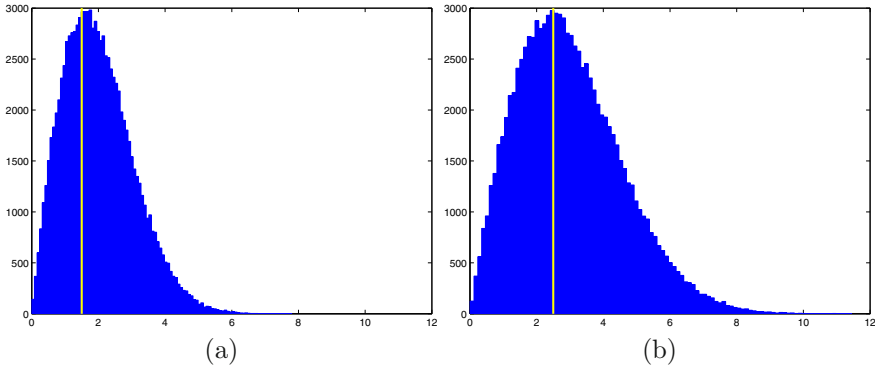


Fig. 3.17 Car Scene – Image feature noise distribution. The two histograms correspond to the maximum noise level where DPR-SfM could generate a valid model for $\rho = 1.5$ and $\rho = 2.5$ respectively. In (a) 35.4% of the features fall within the threshold (yellow line) while in (b) 39.2%.

3.8.2 Water-Tank Sequence

This sequence is part of a series of experiments, used for testing the performance of the DPR-SfM algorithm under realistic conditions. The dataset was acquired by a camera mounted on the Johns Hopkins University (JHU) ROV at the JHU test tank. The bottom of the tank was populated with rocks and shells, simulating the appearance and geometry of a typical seafloor scene. The size of the scene is $\simeq 5 \times 5$m. The sequence, comprised of 3,500 images (see Figure 3.18), was acquired at a constant distance of 1.2m above the bottom of the tank. After the visual survey, the tank was drained and scanned with a Leica Geosystems laser scanner, obtaining 3.8 millon points with an estimated accuracy of 1.2mm.

The objective of this experiment was to assess the 3D reconstruction accuracy of the DPR-SfM using the ground truth, under a realistic scenario, and compare it with state-of-the-art SfM algorithms.

For this purpose we have applied DPR-SfM on the dataset in conjunction with OVV (see Chapter 4) in order to efficiently detect loop closures, followed

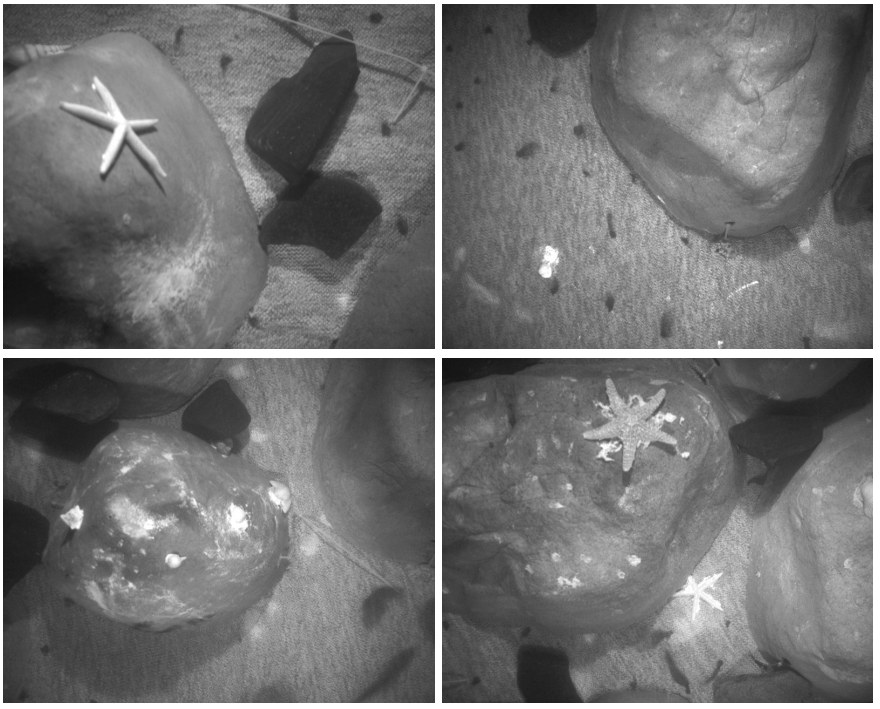


Fig. 3.18 Water-tank Sequence – Input images. Sample images from the dataset depicting some of the objects present in the scene.

by bundle adjustment. The resulting 3D model, consisting of 610,000 vertices, is illustrated in Fig. 3.19 along with the estimated camera trajectory.

The same dataset was processed using sparse reconstruction algorithm: VisualSFM [164, 175], and dense reconstruction algorithms: Patch-based Multi-view Stereo Software (PMVS) [44, 141] and Multi-View Reconstruction Software (CMPMVS) [22, 78]. It should be noted here that PMVS and CMPMVS are multi-view stereo approaches, meaning that these algorithms require camera poses to be computed prior to the reconstruction. In our experiment, we have used camera poses obtained using both DPR-SfM and VisualSFM as basis for the dense reconstructions.

The accuracy of the reconstructions was quantified by comparing the obtained models with the laser scan. For this, for each obtained 3D model, we first manually aligned it with the laser scan using 3D point correspondences. The alignment was further refined using Iteratively Closest Point (ICP) [186]. In order to assess the accuracy of the reconstructions, we quantify the reconstruction errors using the Hausdorff distance [115] between the models and the laser scan ground truth. Table 3.4 summarizes the reconstruction accuracy, 3D model complexity and computational times for each of the reconstruction techniques. DPR-SfM provides the most accurate 3D reconstruction, compared to either sparse and dense reconstruction techniques. Moreover, for both PMVS and CMPMVS, the models obtained using camera poses estimated using DPR-SfM yield a higher accuracy. The complexity of the model obtained using DPR-SfM is slightly higher than VisualSFM and $\simeq 60\%$ of the complexity of dense models in terms of number of vertices.

Regarding the computational costs, DPR-SfM had similar execution times with PMVS and CMPMVS, while VisualSFM has much higher execution times due to its brute-force approach for cross-over detection – tries to match any possible combination of two images in the sequence.

Table 3.4 Water-tank Experiment – Comparison between 3D reconstruction algorithms. The table summarizes the reconstruction errors for DPR-SfM, VisualSFM, PMVS and CMPMVS. Both the average error $\overline{E}$ and maximum error E_{max} shown here are provided in metric units and in percentages of scene depth. For PMVS and CMPMVS, we show the reconstruction accuracy when using camera poses recovered using both DPR-SfM and VisualSFM – the computational times shown here represent only the dense recontruction process and do not include the camera pose recovery process.

Algorithm	$\overline{E}$ [m]	$\overline{E}$ [%]	E_{max} [m]	E_{max} [%]	Vertices	Time [h]
DPR-SfM	0.011	0.91	0.092	7.60	610,000	4.1
VSFM	0.0125	1.03	0.116	9.59	560,000	97.5
DPR-SfM+PMVS	0.016	1.32	0.134	11.07	1,022,000	3.95
DPR-SfM+CMPMVS	0.015	1.24	0.129	10.66	1,343,000	4.8
VSFM+PMVS	0.0173	1.43	0.137	11.32	957,000	3.92
VSFM+CMPMVS	0.0165	1.36	0.133	10.99	1,256,000	4.82

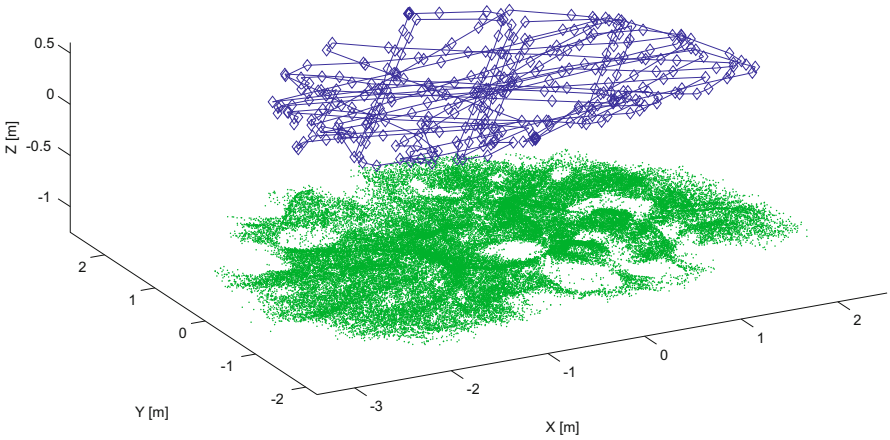


Fig. 3.19 Water-tank Sequence – Scene model and camera trajectory obtained using DPR-SfM. The model consists of 610,000 vertices, shown in green. The camera trajectory is marked in blue. Both the model and the camera trajectory were subsampled for illustrative purposes.

Figure 3.20 illustrates the error distribution within the reconstruction obtained using DPR-SfM. The wide regions of the tank bottom with higher error correspond to changes in the carpet shape as the tank was drained for the laser scanning. For details on the acquisition process refer to [138].

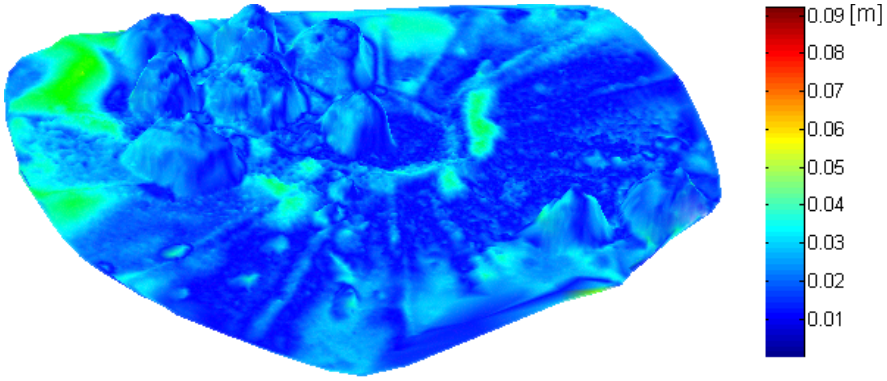


Fig. 3.20 Water-tank Sequence – Error distribution. The color encoded by error magnitude, lighter areas correspond to higher errors.

3.8.3 Rocks Loop

In this experiment, we discuss the capability of the DPR-SfM algorithm to model outdoor, unstructured scenes.

The scene, illustrated in Figure 3.21, is formed by a random arrangement of rocks. The image sequence was acquired using a monochrome camera with a resolution of 696×520 pixels. A sample of the images is shown in Figure 3.22. During the acquisition, the camera was looking downwards, towards the scene, and rotated so that its y axis was tangent to the direction of movement, simulating a down-looking camera mounted on an UUV.

Fig. 3.21 Rocks Loop – Overview. The scene is comprised by a round area with a diameter of $\simeq$ 8m. The area is covered by rocks with varying sizes and textures, ideal for simulating an underwater relief.

The sequence of 740 frames was processed using HarrisAffine-SURF and SURF-SURF, yielding 170,000 vertices – 86,000 Harris and 84,000 SURF (see Figure 3.23a). We obtain an average back-projection error of 1.72 pixels, with 1.67 pixels for Harris and 1.75 pixels for SURF. The average track length for Harris is 12.1 frames while for SURF is 14.3 frames. This shows that, in the case of unstructured environments, Harris provides better precision in feature localization, while SURF is more robust to image transformations.

The major drawback of these types of environments is the impossibility of an exact quantification of the reconstruction accuracy due to the lack of ground truth. We overcome this by designing the camera trajectory to have

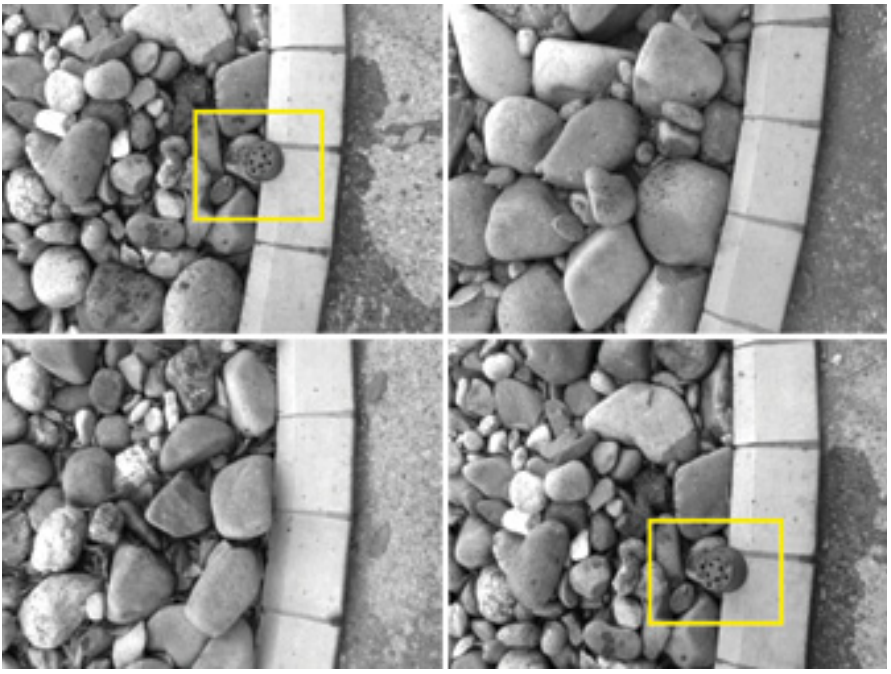


Fig. 3.22 Rocks Loop – Input images. A sample set of the input images. We used a plastic object (highlighted in yellow) to mark the beginning and ending of the loop.

a loop form, so that its beginning overlaps its ending (see Figure 3.22). In this way, we establish constraints between the two ends of the loops (see Appendix B). After detecting the loop closure and applying BA, we correct the estimation errors up to a high degree of precision (see Figure 3.23b). We use this corrected model as the ground truth and compare it with the original result, quantifying the accuracy of the DPR-SfM. Figures 3.24 and 3.25 illustrate the error evolution in vertex position and camera pose respectively.

3.8.4 Pool Trials

We present one of the experiments we have conducted in the Underwater Robotics Center of the University of Girona. Shown in Figure 3.26a, the center is endowed with a pool used for performing tests of small class underwater vehicles. The Underwater Vehicles (UVs) are controlled and monitored from a submerged control room, allowing the researchers to have live panoramic view of the experiments.

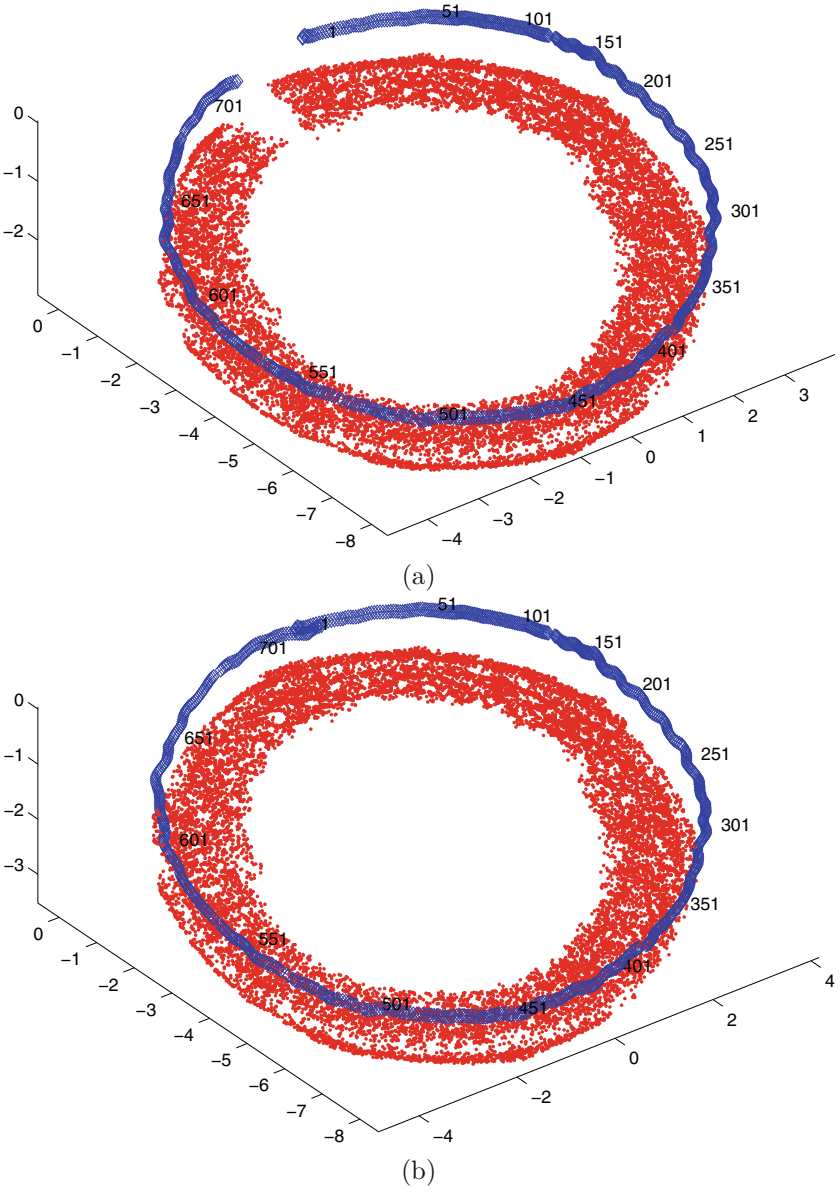


Fig. 3.23 Rocks Loop – 3D model and camera trajectory. Figure (a) illustrates the resulting model along with the estimated camera trajectory. The drifting generates a gap in the model where the loop should be completed. The model is corrected after loop closure detection and BA (b).

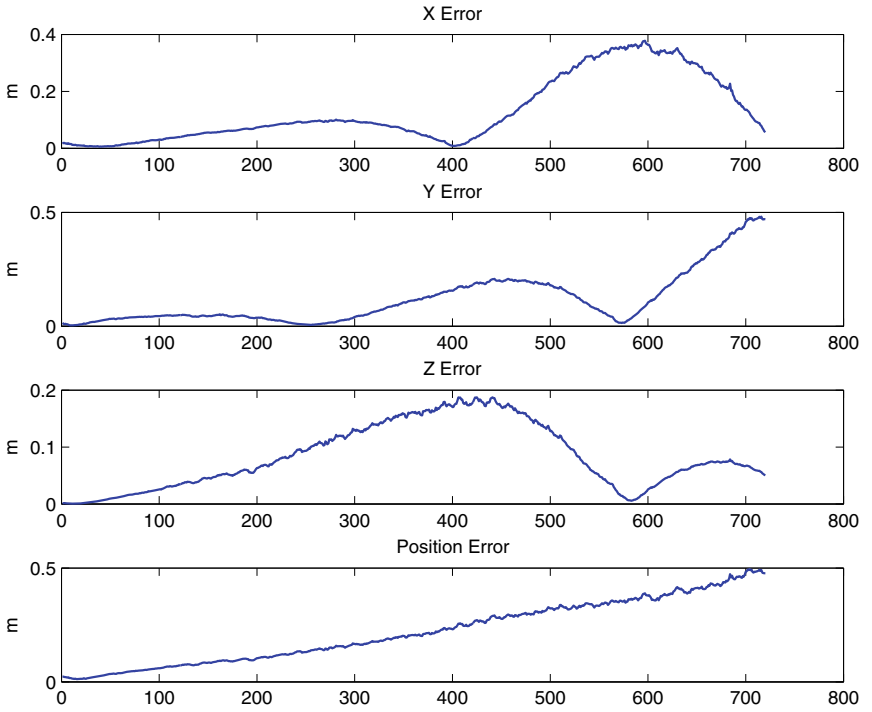


Fig. 3.24 Rocks Loop – model estimation errors. The evolution of the vertex position estimation errors by frame for each degree of freedom. Bottom plot illustrates the total vertex estimation error.

The tests were performed using *Ictineu*, an open frame, small class Autonomous Underwater Vehicle (AUV) (see Figure 3.26). The modular design of Ictineu allows us to set up different types of sensors, depending on the mission environment and purpose. For our experiments, we have used an off the shelf, low end, 384×288 pixels monochrome camera. The camera was mounted on Ictineu on a down-looking configuration.

The AUV was set to follow predetermined trajectories, while the camera was acquiring images of a poster mounted on the bottom of the pool, simulating a seafloor scene.

The aim of the experiments is to observe the behavior of the DPR-SfM algorithm in the presence of flat scenes. In these cases (*e.g.* sandy seafloor regions, building facades, etc.), SfM algorithms fail due to the lack of parallax. Our dual approach, on the other hand, allows us to handle these situations (see Sections 3.3 and 3.5).

In the presented experiment, we have acquired a sequence of 150 frames while Ictineu was following a straight trajectory, maintaining a constant

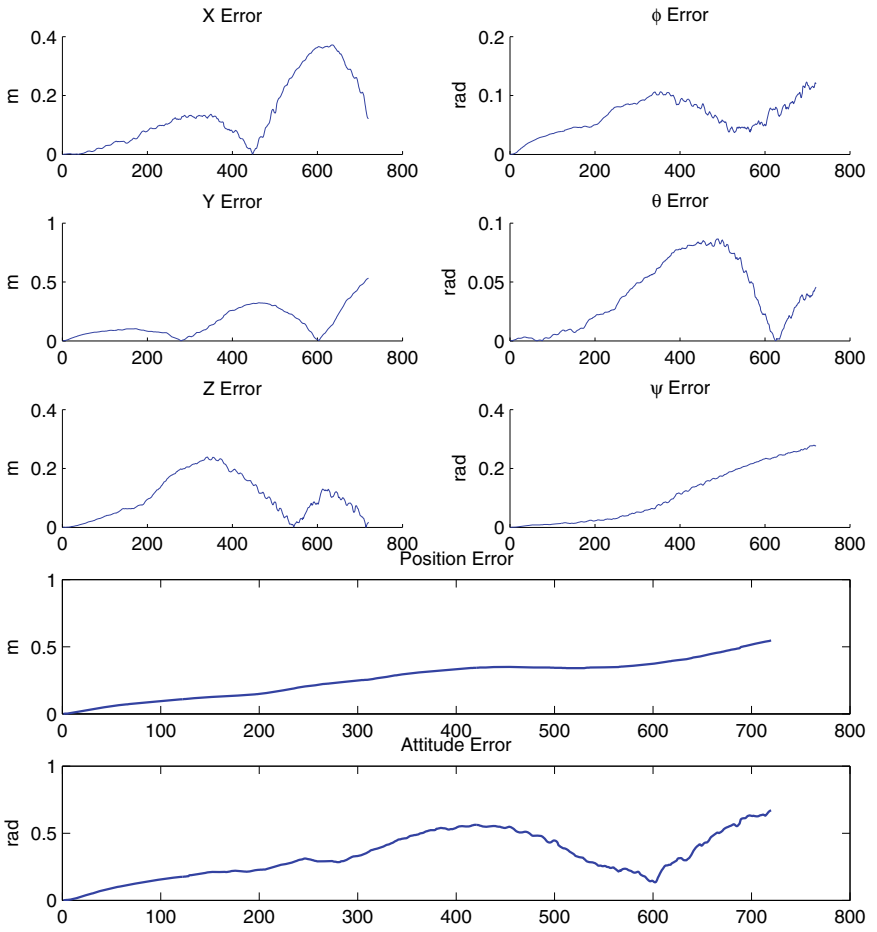
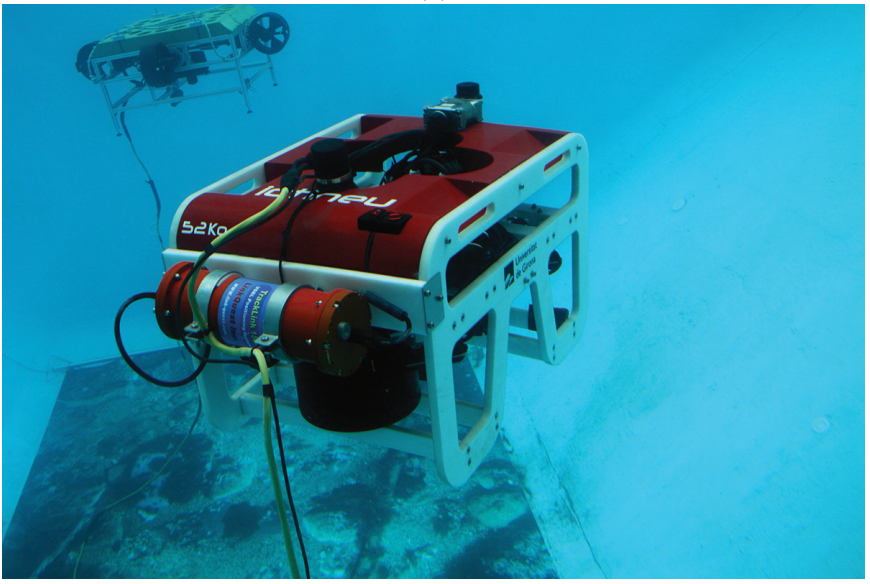


Fig. 3.25 Rocks Loop – Camera pose errors. Camera pose estimation error evolution by frame, for each degree of freedom. Bottom two plots illustrate total estimation errors for position and attitude respectively.

distance to the poster of $\simeq 1.5m$ (refer to Figure 3.27 for examples of images from the dataset). During the experiment, there was a brief communication error between Ictineu and the control room generating some invalid frames to be captured. This offered an ideal situation to test the robustness of the DPR-SfM algorithm when faced to camera obstructions / errors.

After processing the sequence, we obtained 10,000 HarrisAffine and 7,000 SURF vertices. In both cases, we used SURF for description. Figure 3.28 illustrates the result of the reconstruction. The gap in the camera trajectory

(a)



(b)

Fig. 3.26 Pool Trials – Experimental setup. (a) Underwater Robotics Laboratory of the University of Girona and (b) Ictineu AUV (foreground) with the seafloor poster during the experiments, photographed from the submerged control room.

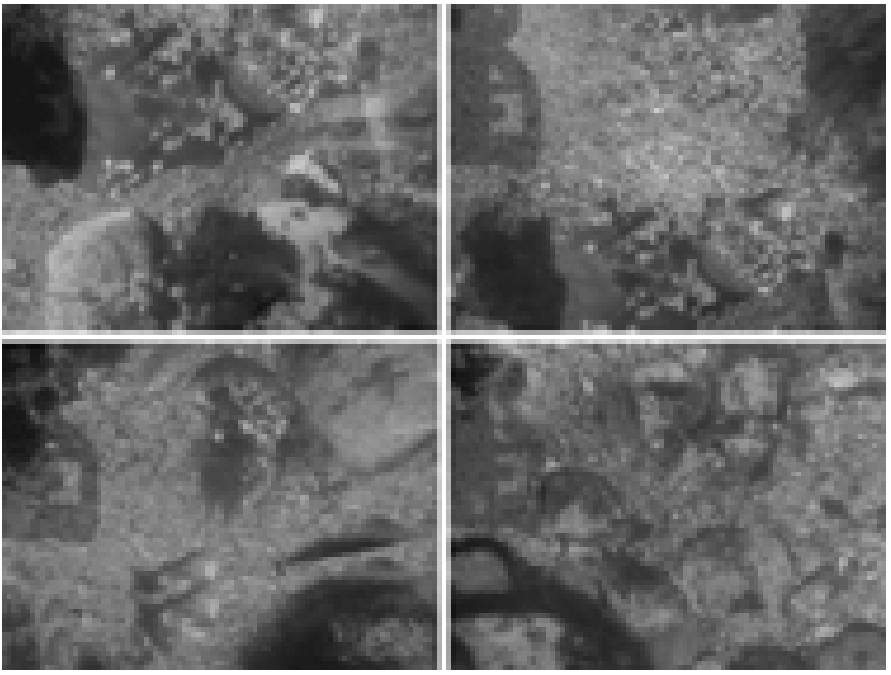


Fig. 3.27 Pool Trials – Input images. Images from the sequence of the poster simulating an underwater scene.

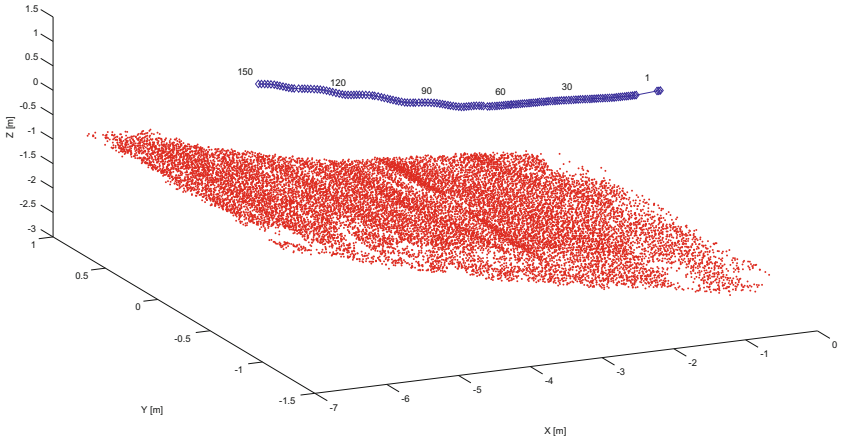


Fig. 3.28 Pool Trials – 3D model and camera trajectory. 3D model of the poster and camera trajectory. There is a gap in the camera trajectory due to a communication error between the UV and the control room.

corresponds to the communication error. DPR-SfM was able to recover from this situation, correctly registering the following frames.

In order to account for the precision of the reconstruction we first first determine the average scene plane using Least-Squares fitting to the 3D vertices. As the scene is planar, we define the reconstruction error as the Euclidean distance between the plane and the 3D vertices. The distribution of the reconstruction error is illustrated in Figure 3.29.

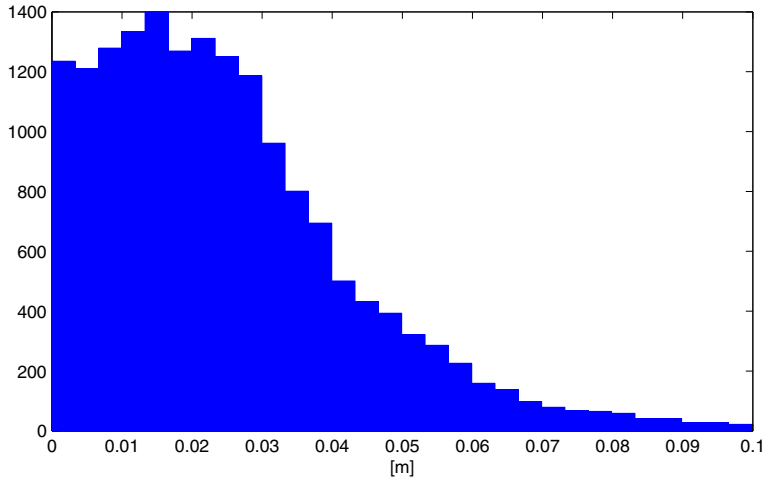


Fig. 3.29 Pool Trials – Reconstruction error histogram. We calculate the reconstruction error as the Euclidean distance between the scene plane and the vertices.

3.8.5 Coral Reef Sequence

Here we discuss the results obtained from sequence depicting a coral reef area. This dataset is part of a larger survey of a benthic habitat undertaken in shallow waters in The Bahamas. The images were acquired by the University of Miami (UoM) using a hand-held HD camera. The sequence consists of 1,100 images of 962×540 pixels (the resolution of the images was reduced from 1920×1080 due to interlacing). The area was surveyed with the camera following a "lawnmower" trajectory, with partial overlap between adjacent columns. This provides a complete coverage of the area while offering additional constraints in the model.

The sequence covers $\simeq 150m^2$ and was chosen to include different types of topologies and textures often found in underwater scenes. Figure 3.30 depicts typical entities found in the dataset. We recover the scene model using HessianAffine-SURF and SURF-SURF features with an outlier rejection threshold $\rho = 1.5$, obtaining 270,000 vertices (130,000 HessianAffine and

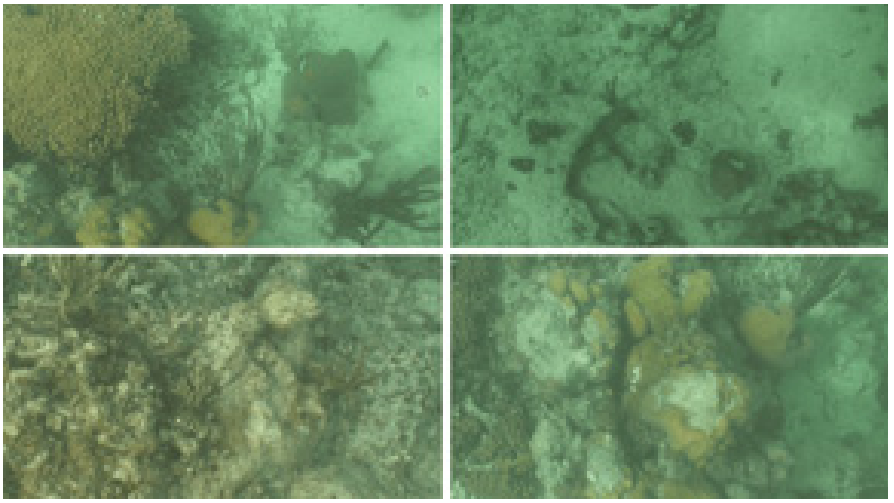


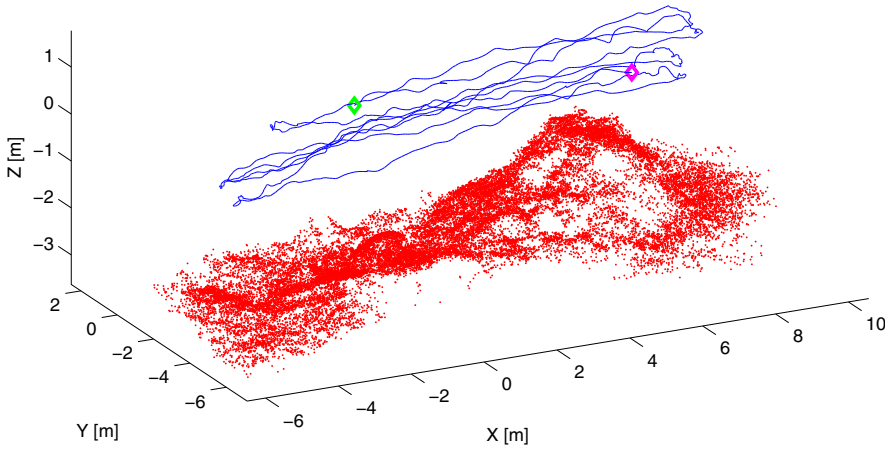
Fig. 3.30 Coral Reef Sequence – Input images. Sample images from the input sequence showing different types of regions: coral reef formations, rocks, algaes, sand, etc.

140,000 SURF). Figure 3.31 illustrates the scene model and camera trajectory – the number of vertices in the model has been reduced 10 times in order to avoid cluttering in the figure.

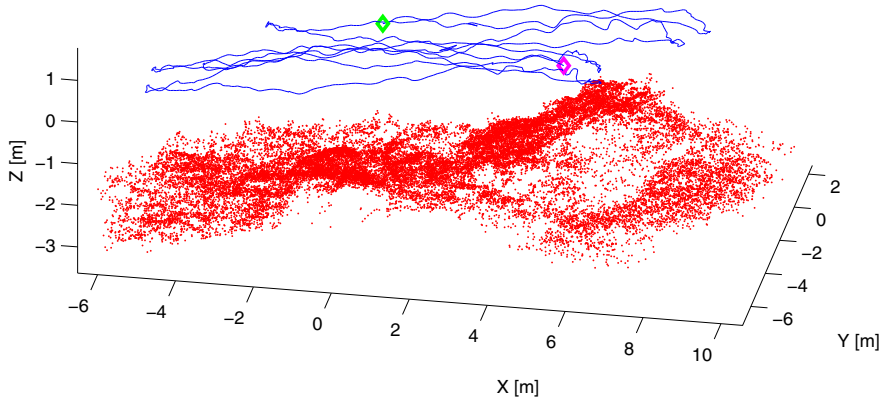
The aim of this experiment is to asses the accuracy of the model with respect to the texture types present in the scene. For this, we consider the average back-projection error for each reconstructed vertex. Figure 3.32 shows that the precision of the vertex reconstruction is highly related to the saliency of the corresponding image features[7]. Moreover, it can be observed that there is a strong correlation between the vertex precision and the type of its neighboring scene type (*e.g.* vertices in rocky and coral reef areas are more accurate than ones in sandy areas).

Using the constraints between adjacent columns in the camera trajectory (see Appendix B), we apply BA on the sequence. We use the result as reference to quantify the errors in the reconstruction. The error evolution in camera pose estimation is illustrated in Figure 3.33. As the camera is registered directly with the model, the errors do not increase significantly along the columns in the camera trajectory, reducing drastically the error accumulation.

[7] The saliency represents a quality measurement of the features. It is related to the image gradient in the neighborhood of the feature, so that higher saliency corresponds to more accurate and discriminant features.

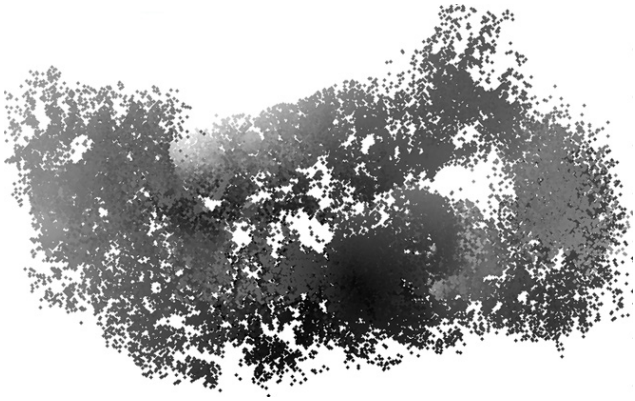


(a)

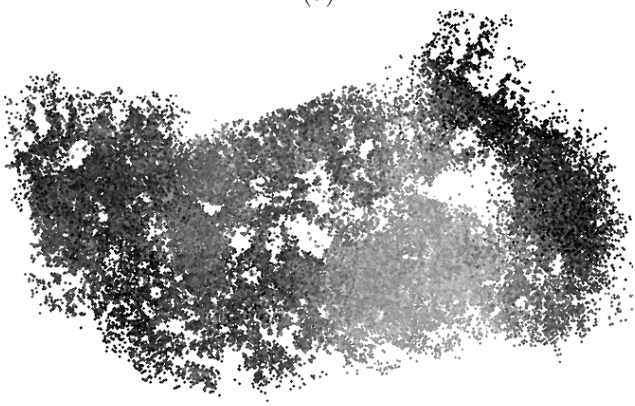


(b)

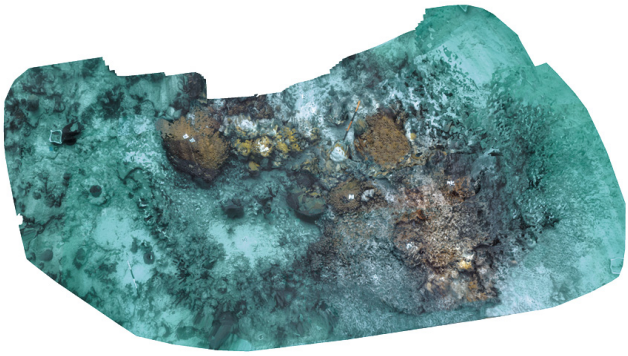
Fig. 3.31 Coral Reef Sequence – 3D model and camera trajectory. (a) simplified scene model and camera trajectory: green and magenta markers show the beginning and end of trajectory respectively; (b) another view of the scene model.



(a)

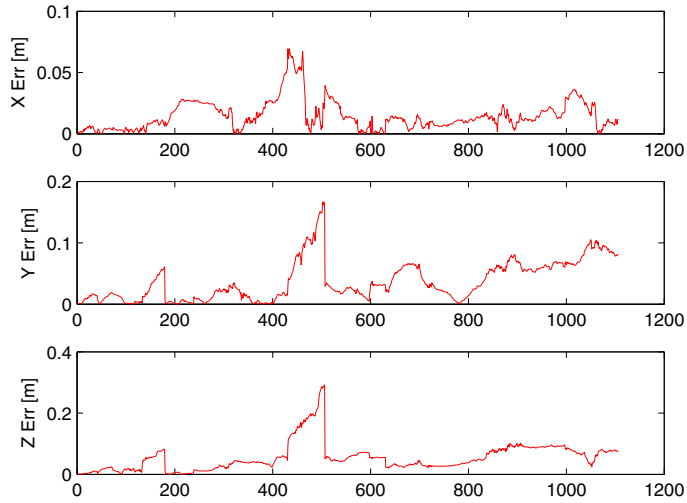


(b)

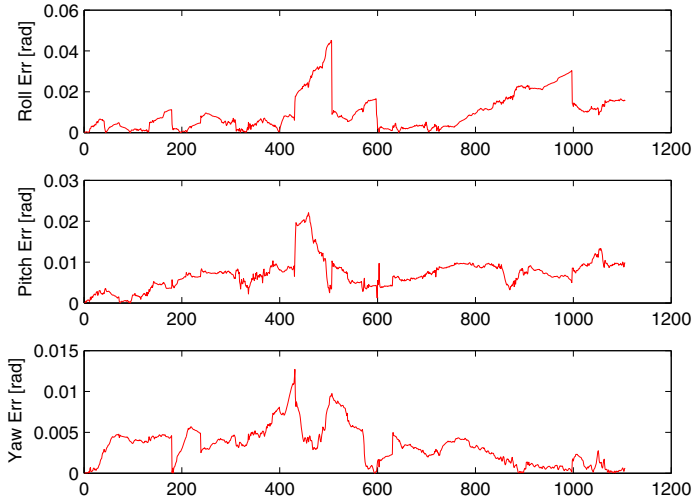


(c)

Fig. 3.32 Coral Reef Sequence – Vertex error. Figure (a) shows the back-projection error distribution. Darker values correspond to higher accuracy. The distribution of image feature saliency is shown in (b); lighter values correspond to higher saliency. The ortho-mosaic of the scene is provided for reference in (c), showing the relation between region types, feature saliency and vertex accuracy.



(a)



(b)

Fig. 3.33 Coral Reef Sequence – Camera pose errors by frames. (a) camera pose errors and (b) camera attitude errors.

3.8.6 Mequinenza Sequence

In this experiment, we aim to test the behavior of the DPR-SfM under difficult image conditions. The sequence was captured in the Ebro river, Mequinenza, Catalunya by the Ictineu AUV using a down-looking monochrome camera. Due to the high turbidity in the water, we used additional lighting, which increased the visibility but induced shadows and non-uniform illumination patterns. Moreover, due to back-scattering, the images have low contrast (see Figure 3.34).

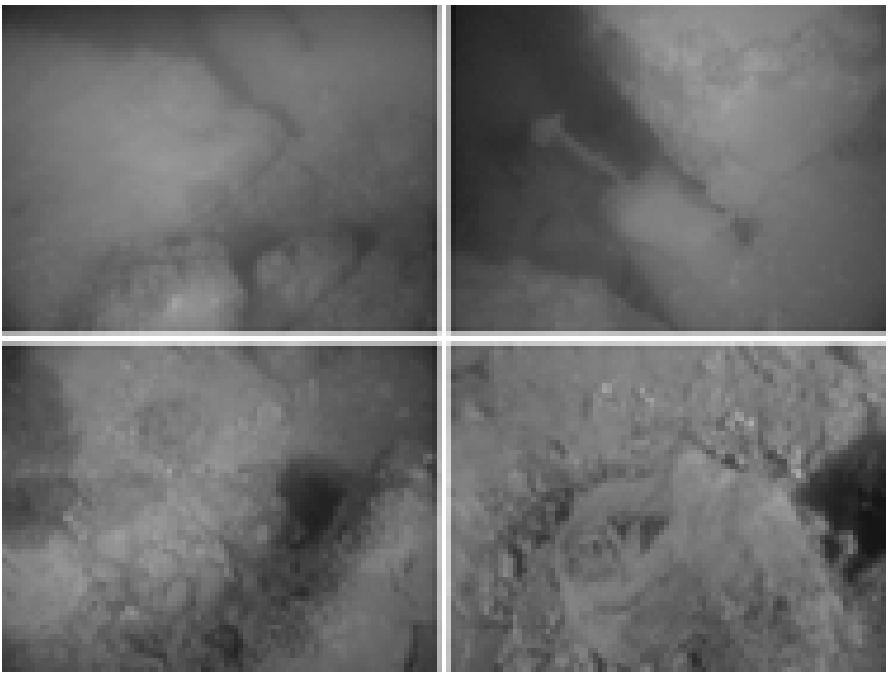
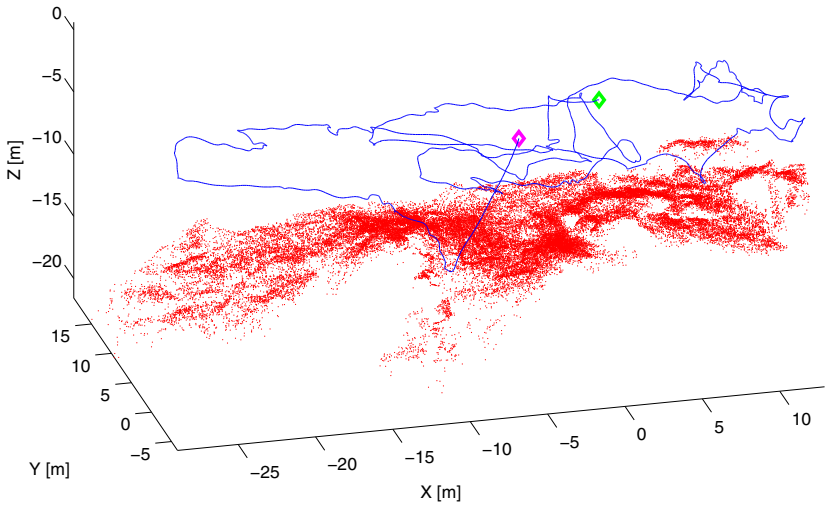
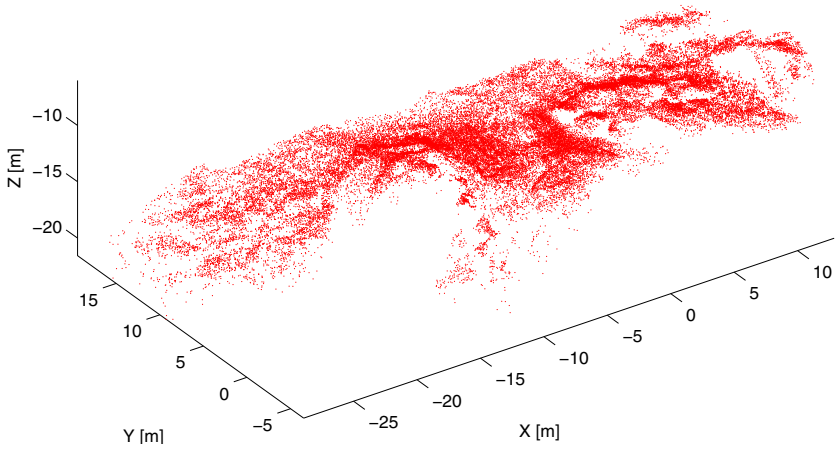


Fig. 3.34 Mequinenza Sequence – Input images. Image samples depicting some of the challenges of sequence: scattering, light absorbtion, shadows, complex scene geometry, etc.

The sequence, comprised by 2,900 frames of 384 × 288 pixels resolution, was first pre-processed using Contrast Limited Adaptive Histogram Equalization (CLAHE) [188] in order to enhance the quality of the images. Using SURF-SURF features, we obtained 220,000 vertices. Figure 3.35 illustrates the resulting camera trajectory and scene model (the number of features has been reduced for illustration clarity). The model shows an environment with complex geometry, also, the trajectory of the camera depicts a motion of Ictineu with sudden changes in heading and motion direction due to the water currents.



(a)



(b)

Fig. 3.35 Mequinenza Sequence – 3D model and camera trajectory. (a) scene model along with camera trajectory: green and magenta markers show the beginning and end of trajectory respectively; (b) another view of the scene model.

Using an outlier rejection threshold ρ of 1.5, we obtain an average back-projection error for the whole model of 0.9 pixels.

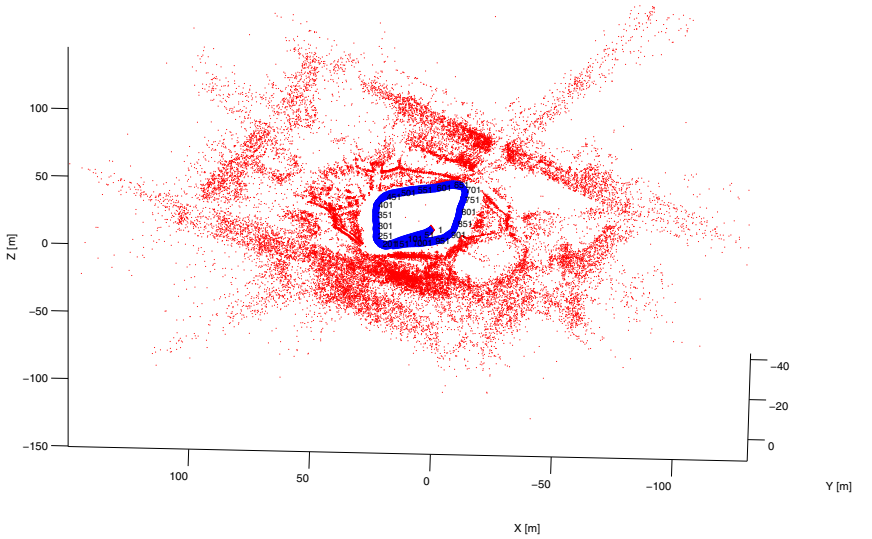
3.8.7 Urban Experiment

This experiment was aimed at testing the DPR-SfM algorithm for large-scale urban modeling applications. For this, we acquired a sequence of Unirii Square in Timisoara, Romania. The square, illustrated in Figure 3.36, has a rectangular shape, measuring $\simeq 155 \times 120$m and is surrounded by historical buildings of various shapes and textures. We used a low-end Pentax Optio A30 digital camera for video acquisition, while walking through the square following a loop trajectory. The resulting image sequence contains 961 frames of 640×460 pixels in resolution (see Figure 3.37).

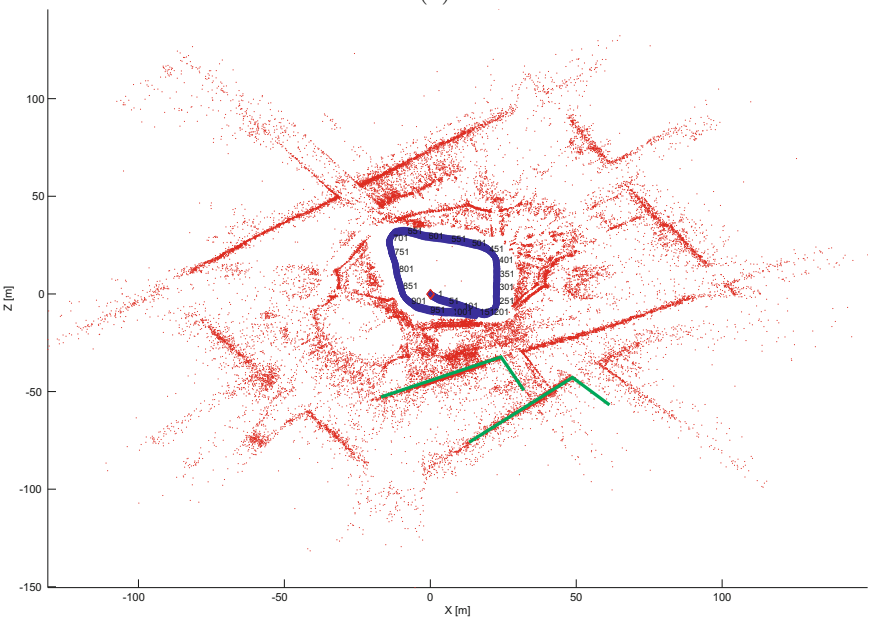
Fig. 3.36 Urban Experiment – Overview of the Unirii Square. Aerial view of the Unirii Square.

After applying DPR-SfM on the sequence using SURF-SURF, the resulting model, shown in Figure 3.38, contains 240,000 vertices. The drift due to error integration is obvious at the loop closure, where the facades of the buildings are repeated (see Figure 3.38b). The main reason behind the high drift in this dataset is the decreased precision in feature localization due to the low quality of the images: the camera uses a high compression ratio MPEG2 codec, which results in loss of details in images.

Fig. 3.37 Urban Experiment – Input images. Sample images from the dataset, showing some of the typical challenges such as moving objects, occlusions, sun flickering, lack of texture, etc. Also, the partial overlap between the first and last image can be clearly observed.



(a)



(b)

Fig. 3.38 Urban Experiment – 3D model and camera trajectory. (a) scene model (red) along with camera trajectory (blue) – the number of vertices in figure was reduced by 10 times to avoid cluttering; (b) top view of the scene model clearly depicting the drift at the loop closure (repeated edges at the bottom marked in green).

After the loop closure detection (see Section 4.4.5), we corrected the model, as shown in Appendix B. The result is shown in Figure 3.39.

Considering the model after BA as the ground truth, we calculate the camera pose and vertex position estimation errors by comparing the models before and after the BA (in a similar fashion to the experiment described in Section 3.8.3). Figure 3.40 illustrates the error for both camera and vertex estimations.

3.9 Discussion

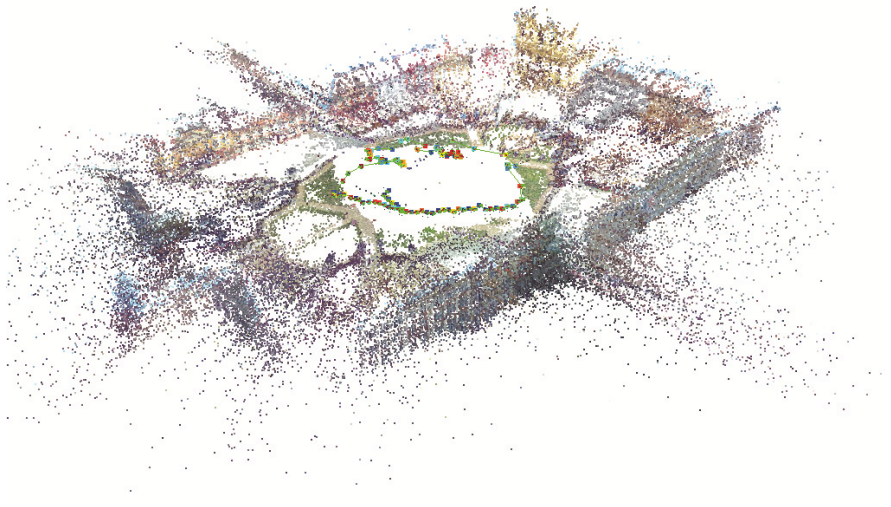
In this chapter we presented a novel SfM algorithm for large scale scene modeling. The algorithm generates the scene models sequentially, using a two stage approach. Initially, DPR-SfM creates a seed model corresponding to a small subregion of the scene, using camera motion estimation techniques. In the second stage, the scene model is extended to cover the entire surveyed area. During scene reconstruction, the camera pose is recovered by directly registering camera views with the scene model. This increases the accuracy and robustness of DPR-SfM, allowing it to successfully cope with situations often found in visual surveys such as occlusions, camera temporary failures, etc. Also, using direct camera pose registration highly increases the flexibility of the DPR-SfM.

Generally, state of the art SfM algorithms require additional sensor information or impose constraints on the image acquisition (*e.g.* minimum camera movement between frames for correct motion estimation). DPR-SfM can be readily applied on image sequences acquired with any type of camera, both still and video, with no constraints on the camera acquisition process. Also, the presented SfM algorithm does not require navigation priors. However, sensor information such as camera pose can be used to decrease the computational cost of the algorithm.

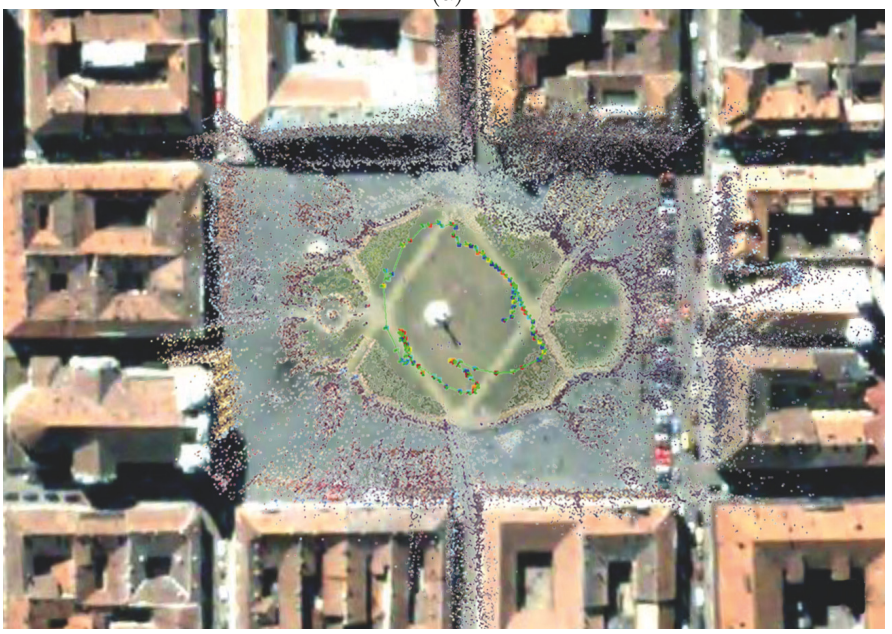
The direct camera pose registration uses a novel dual RANSAC projective/homography approach which allows the DPR-SfM algorithm to accurately model both planar and non-planar scenes. This is particularly important in underwater and urban scenes, where parts of the scene can have significant parallax while others can be perfectly planar.

Robust estimation methods are also used on vertex position recovery. Experiments show that using a dual layer (camera and model) RANSAC approach increases the stability and accuracy of the method, especially in challenging environments, such as underwater, where image blurring and low contrast decrease the efficiency of feature tracking.

We have also developed an efficient and flexible scene representation. It allows the 3D modeling of large and complex scenes while enabling the parallel use of multiple visual feature extractors/descriptors. In this context, we employed a *kd*-tree scheme for efficient feature matching and camera registration even for large scene models.



(a)



(b)

Fig. 3.39 Urban Experiment – 3D model and camera trajectory after BA. (a) view of the 3D model using colored vertices, and the camera trajectory; (b) top view of the 3D model aligned with an aerial view of Unirii Square from *Google Earth* – the reconstruction fits the photo accurately.

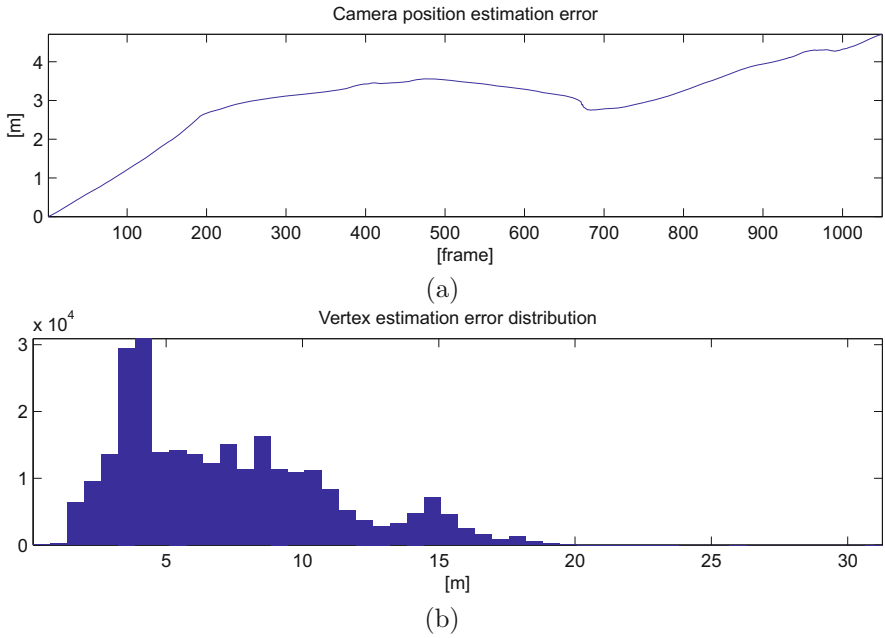


Fig. 3.40 Urban Experiment – Estimation errors. (a) total camera position drift: evolution by frames; (b) vertex estimation error distribution.

Experiments show the robustness of DPR-SfM in both land and underwater environments. In the Water-tank experiment we compare different state-of-the-art sparse and dense SfM techniques, showing that DPR-SfM has improved accuracy, both in terms of scene modeling and camera pose recovery.

Results demonstrate that DPR-SfM can efficiently cope with large and complex reconstructions[8] (*e.g.* Section 3.8.2).

There are several ongoing and future topics that may improve the work presented in this chapter. After camera pose registration, the image patches around features can be warped using camera-to-model transformations. This would reduce the limitations of feature extractor/descriptors of coping with extreme geometric distortions, increasing the efficiency of feature matching. Also, the accuracy of feature localization can be improved by using cross-correlation as a refinement step after feature tracking. Feature-to-model association computational costs can be highly decreased by using GPU-based parallel processing, *e.g.* using *NVIDIA CUDA*.

[8] We consider the complexity of the 3D modeling problem to be quantified by the amount of data involved in the reconstruction (*i.e.* number of camera poses and vertices), rather than the metric size of the scene, as the size of the reconstructed area depends only on the camera-to-scene distance and the properties of the camera lenses.