5

3D Shape Reconstruction from 2D Images

5.1 Introduction

The human visual system exhibits a tremendous ability in subconsciously perceiving 3D shape information from the environment. Our natural stereo vision system can effortlessly feed the brain with biological signals describing the 3D world around us, consisting of complex 3D shapes and surfaces like faces, buildings, mountains, etc. A full understanding of the 3D vision complexity is still far away.

One of the main goals of computer vision is the reconstruction of the 3D world from a set of 2D images. There are numerous applications in which such a capability is required or is highly desirable, such as 3D cinema, video games, digital 3D asset creation, virtual and augmented reality and robot navigation [SKS05]. Automatic 3D object reconstruction can significantly lower the development cost of such applications.

There are different approaches to 3D shape reconstruction, depending on the number of scene views and the employed 3D reconstruction technique. This Chapter will focus on two main approaches: reconstruction from two (binocular) or three (trinocular) views, based on feature correspondences, and volumetric reconstruction from many views, using shape-from-silhouette, shape-from-photo-consistency and shape-from-surface integral minimization techniques.

In the case of reconstruction from two or three views, the matching of either all (*dense correspondence*) or some (*sparse correspondence*) features from one

view to corresponding features in the other ones is a prerequisite. In stereo reconstruction, for each such image feature pair, the difference between the matched features horizontal coordinates is called their *disparity*, similarly to the stereo disparity of the human visual system that was introduced in Chapter 3. The result of this process, based on whether dense or sparse correspondences were estimated, is a dense or sparse *disparity map*, respectively.

5.2 Feature Correspondence

The basic principle of a feature correspondence method is that the position of a point $\mathbf{P} = [X, Y, Z]^T$ in 3D world can be computed from at least two projections $\mathbf{p}_i = [x_i, y_i]^T,\ i = 1, 2$ of this point on two known image planes, corresponding to the two cameras taking scene snapshots, as shown in Figure 5.2.1. Given that the internal and external camera parameters are known, the 3D location of the point $\mathbf{P}$ can be calculated by simple *triangulation*. Even when the camera parameters are not known (uncalibrated cameras) the location of $\mathbf{P}$ may still be recovered, as multiple point correspondences can be used to calibrate the cameras [HZ00].

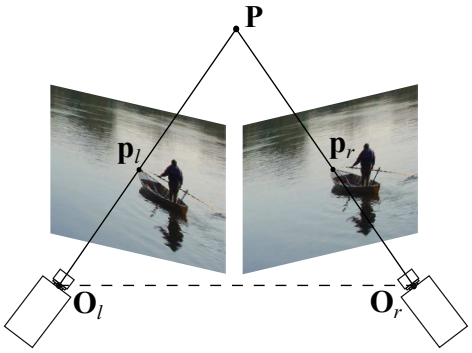
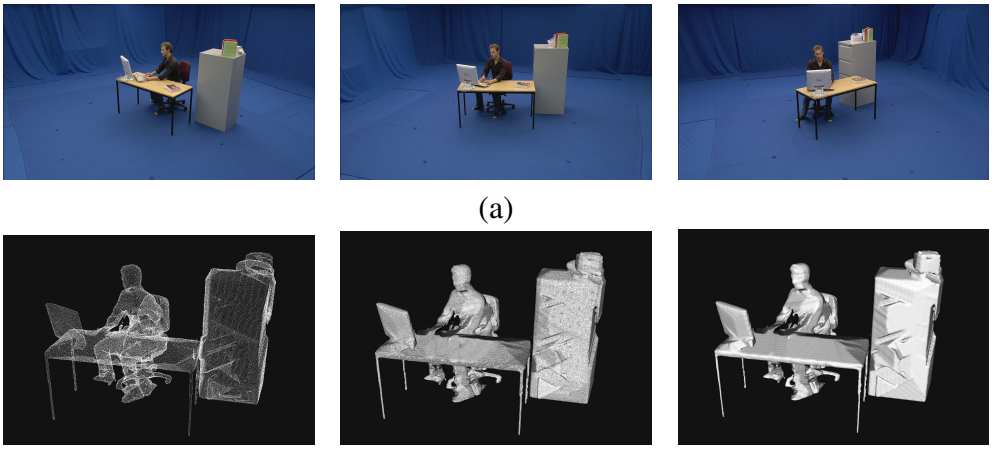


Figure 5.2.1: 3D point positioning by triangulation.

The triangulation of multiple point correspondences produces a 3D point cloud of the scene, as can be seen in Figure 5.2.2b, in the case of multiview images (Figure 5.2.2a). A polygonal mesh is subsequently created using the Delaunay triangulation of the point cloud, as described in Chapter 1 [NP01] and shown in Figures 5.2.2c and 5.2.2d. The final result of the reconstruction can be a textured mesh. The texture data are acquired by reprojecting the 2D images of the scene onto the 3D mesh.



(a)

(b) (c) (d)

Figure 5.2.2: 3D object reconstruction: a) three views of a 3D scene, b) reconstructed 3D point cloud, c) wireframe surface mesh, d) solid surface mesh *(Courtesy of the i3DPost FP7 project and the University of Surrey)*.

5.2.1 Feature extraction

The first step of the 3D shape reconstruction is the extraction of image features, as shown in Figure 5.2.3. They are subsequently used to find feature correspondences across the stereo image pair. A local feature is a part of an image having some interesting spatial characteristics. Since human vision is sensitive to abrupt luminance changes, edge or corner points [Pit00] or edge segments are often used as local image features. Other common image features are lines and line segments. Feature point selection is very important, because it forms the basis for subsequent 3D reconstruction steps. *Feature detectors* detect and output a list of image features, along with the associated *feature descriptors* [MS05].

Features can be roughly categorized into low-level and high-level ones. Low-level features, such as *edges*, *corners* and *line segments* have simple or no feature descriptors, besides the feature location coordinates. Low-level feature detectors are typically simple image analysis operators. Edges can be detected using the well known Roberts, Sobel, Prewitt or Canny edge detectors [Pit00]. Line and circle detection is accomplished via the Hough transform [Pit00]. Harris corner detector [HS88] is another widely used feature extraction method. It utilizes the spatial gradients of image intensity $f(x, y)$, namely $f_x = \frac{\partial f}{\partial x}$ and $f_y = \frac{\partial f}{\partial y}$. A

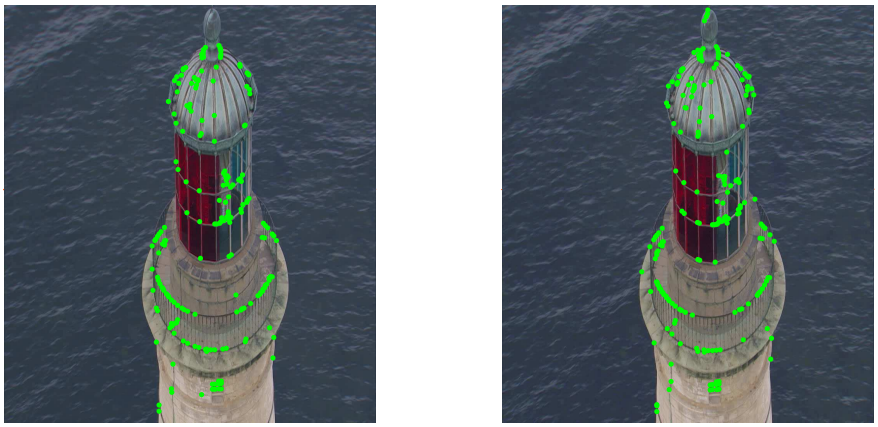


Figure 5.2.3: Corner features extracted from two views of a scene.

Harris matrix for an image neighborhood is constructed in the following way:

$$\mathbf{H} = \begin{bmatrix} \Sigma f_x^2 & \Sigma f_x f_y \\ \Sigma f_x f_y & \Sigma f_y^2 \end{bmatrix}, \tag{5.2.1}$$

where the sums are computed over a local image neighborhood. The eigenvectors and eigenvalues of this matrix encode the edge directions and strength, respectively, that are present in the local image neighborhood. If one of the eigenvalues has large value, then an edge is detected. If both eigenvalues have large values, then a corner is detected. In 3D scene reconstruction, corners are preferred to edges, since they are more localized, sparse and tend to be more stable to changes in ambient scene illumination, leading to better 3D reconstruction results. A variety of other corner detectors have also been suggested, such as [Bea78], [KR82], and [ZH83], with the former incorporating a rotationally invariant operator. In order to find the line segments, edge detection algorithms must be applied first. Several line detectors have been developed so far [NB80], [FB83], [WB86].

In the last decade, more complex feature detectors have been proposed for the detection of high-level features. The feature descriptors they produce, are typically rich (described by several bits) and invariant to some scene changes (e.g., illumination conditions) or camera properties (position, orientation, focus). These properties make them appropriate to be used in 3D reconstruction, since they provide enhanced feature matching reliability. The best known and most widely used detector in this category is probably SIFT (*Scale Invariant Feature Transform*)

[Low99]. Other similar detectors are [MS05]: RIFT (a rotation-invariant generalization of SIFT) [LSP04], SURF (*Speeded-Up Robust Features*), a high performance SIFT-like detector [BTVG06], PCA-SIFT, a method applying PCA to the SIFT descriptors [KS04], GLOH (*Gradient Location and Orientation Histogram*) and LESH (*Local Energy based Shape Histogram*) [SSH10].

5.2.2 Feature matching algorithms

Once the feature points are extracted, feature correspondences must be established across stereo or multiview image pairs. *Feature matching* results in *homologous* image features, which are i.e., projections of the same natural 3D point on each camera view. There are several approaches to feature correspondence search, depending on the feature type to be matched. Independently of the matching algorithm, the search space can be significantly reduced by enforcing several constraints, that not only speed-up the search process, but, more importantly, lead to better feature matches, thus enhancing the quality of the final 3D reconstruction. Commonly used constraints are the following ones [TV98], [SKS05]:

1. *Epipolar constraint.* When the projection geometry is known, the epipolar constraint restricts the search for a corresponding feature point to the epipolar line on the other image of the stereo image pair.

2. *Uniqueness constraint* states that a point in one image can have at most one correspondence in the other image, thus, requiring one-to-one matches.

3. *Continuity constraint* states that adjacent feature points in one image should correspond to adjacent features in the other image.

4. *Topological constraint* assumes that the relative position of 3D points remains unaltered in their projections.

Area-based matching algorithms are the oldest matching methods, used mainly for low-level feature matching. Since low-level feature tokens typically contain only their 2D geometrical coordinates, image windows around feature points are used to compare and match them. This approach is very similar to block matching used for motion estimation [WOZ02], as detailed in Chapter 9. Thus, a rich repository of relevant matching algorithms and extensive research is available. Such a matching of two feature points, is based on the minimization of some distance measure of the respective local image windows. Let us assume that two candidate

feature points $\mathbf{p}_l = [x_l, y_l]^T$ and $\mathbf{p}_r = [x_r, y_r]^T$ are to be matched. The grayscale intensity images $f_l(x, y)$ and $f_r(x, y)$ are going to be used to compare the pixels in a $L = (2N+1) \times (2M+1)$ local neighborhood window centered around these points. The simplest distance-based matching measure is the *sum of absolute differences* (SAD) [SKS05]:

$$SAD(\mathbf{p}_l, \mathbf{p}_r) = \sum_{i=-N}^{N} \sum_{j=-M}^{M} |f_l(x_l + i, y_l + j) - f_r(x_r + i, y_r + j)|, \quad (5.2.2)$$

which is essentially the L_1 norm. A slightly more complex distance measure is the *sum of squared differences* (SSD), which weights big differences more than small ones [SKS05]:

$$SSD(\mathbf{p}_l, \mathbf{p}_r) = \sum_{i=-N}^{N} \sum_{j=-M}^{M} \big(f_l(x_l + i, y_l + j) - f_r(x_r + i, y_r + j)\big)^2, \quad (5.2.3)$$

and is essentially the L_2 norm. Alternatively, correlation-based similarity measures can be used in feature matching. Since standard cross-correlation is sensitive to noise, the *normalized cross-correlation* (NCC) is preferred [SKS05]:

$$NCC(\mathbf{p}_l, \mathbf{p}_r) = \frac{\sigma_{lr}^2(p_l, p_r)}{\sqrt{\sigma_l^2(p_l)\sigma_r^2(p_r)}}, \quad (5.2.4)$$

where:

$$\sigma_{lr}^2(\mathbf{p}_l, \mathbf{p}_r) = \frac{1}{(2N+1)(2M+1)} \sum_{i=-N}^{N} \sum_{j=-M}^{M} \big(f_l(x_l + i, y_l + j) - \bar{f}_l\big) \cdot \big(f_r(x_r + i, y_r + j) - \bar{f}_r\big) \quad (5.2.5)$$

$$\sigma_l^2(\mathbf{p}_l) = \frac{1}{(2N+1)(2M+1)} \sum_{i=-N}^{N} \sum_{j=-M}^{M} \big(f_l(x_l + i, y_l + j) - \bar{f}_l\big)^2 \quad (5.2.6)$$

$$\sigma_r^2(\mathbf{p}_r) = \frac{1}{(2N+1)(2M+1)} \sum_{i=-N}^{N} \sum_{j=-M}^{M} \big(f_r(x_r + i, y_r + j) - \bar{f}_r\big)^2. \quad (5.2.7)$$

The *modified normalized cross-correlation* (MNCC) is given by:

$$MNCC(\mathbf{p}_l, \mathbf{p}_r) = \frac{2\sigma_{lr}^2(p_l, p_r)}{\sigma_l^2(p_l) + \sigma_r^2(p_r)} \quad (5.2.8)$$

and is somewhat more stable than NCC.

Alternatively, other image feature characteristics can be used for feature matching, including edge attributes (e.g., edge orientation, location, or even the intensity difference between the two sides of the edges). Unfortunately, edges may suffer from occlusion problems. Corner attributes, like coordinates and/or angle constitute an alternative for matching. The Harris corner detector is one of the most popular corner detectors, as already described in the previous section.

Furthermore, the orientation of line segments and the coordinates of the end or/and the mid points can also be used as attributes. Unfortunately, line segment detection methods are not robust against noise. Matching based on curve segments is not frequently used, not only because of its high computational complexity, but also due to matching ambiguities. Curve attributes, like their turning points, can be exploited [DF90], as well as circles, ellipses and polygonal regions. These higher-level image features are mostly applicable to indoor environments and in the detection of industrial product defects.

A combination of image features, for example, edges, curves, surface and region patches, is used in most of the feature-based stereo or multiview matching systems. Feature-based matching techniques compare the descriptor token vectors containing the attributes of each feature point. Since these tokens vary among different feature extractors, each feature descriptor can be matched differently among views. A general matching approach is based on a similarity metric between a token vector pair. If $\mathbf{x}$ and $\mathbf{y}$ are two feature descriptor vectors and $\mathbf{w}$ is the weight vector of the feature token type, then a similarity measure can be their weighted distance:

$$S = \frac{1}{\|\mathbf{w}^T(\mathbf{x} - \mathbf{y})\|}. \tag{5.2.9}$$

The distance metric $\|.\|$ can be the Euclidean one. If every feature characteristic is equally important, $\mathbf{w}$ can be omitted. Feature matching is performed by searching for a pair of feature descriptors maximizing the similarity metric. This is, in essence, a nearest neighbor search in a n-dimensional space, where n is the dimension of $\mathbf{x}, \mathbf{y}$. A brute force approach is to calculate the similarity of each feature point on one image with every other feature point on the other image and match the pair with maximum similarity. This method is called *naïve nearest neighbor search* and has $O(M^2)$ computational complexity, where M is the number of features in an image. A faster but approximate method is the *best-bin-first search* algorithm, which is a modification of the kd-tree search algorithm [BL97]. There is an abundance of literature on high-dimensional similarity search algorithms, primarily coming from database and pattern recognition fields.

In multiview feature matching, the same set of similarity metrics used in the case of stereo can also be applied in N-views. In this case, for each feature point, $N-1$ correspondences must be found in each of the remaining views, which is not always possible, for example, due to occlusions. A useful technique involves incremental feature matching. At first, matches are detected only in image pairs and the matches computed for different pairs are afterwards compared, in order to form triplets of corresponding image points.

5.3 3D Reconstruction Techniques in Stereo Vision

Once the feature correspondence pairs have been established, the process of 3D point cloud reconstruction begins, in order for a cloud of points in 3D space to be produced. Three general cases must be considered here. In the first case, in which the cameras are *calibrated* (their intrinsic and extrinsic parameters are known), the reconstruction is a simple matter of triangulation. On the other hand, in case the cameras are not calibrated (some or all of their parameters are unknown), some kind of calibration (parameter recovery) needs to take place first.

Alternatively, if only the intrinsic parameters are known, the problem can be solved by estimating the extrinsic parameters and the 3D geometry up to an unknown scaling factor. Finally, in case no parameters are known, 3D scene reconstruction is only possible up to an unknown projective transformation.

5.3.1 Parallel and Converging Camera Setups

The parallel and the converging stereo camera setups are very common and lead to very simple triangulation methods for 3D shape reconstruction, if the stereo camera parameters are known. Their geometries are presented in the following sections.

The geometry of a stereo camera with parallel optical axes is shown in Figure 5.3.1. The points $\mathbf{O}_l$ and $\mathbf{O}_r$ are the centers of projection of the left and right camera, respectively. The projection planes of the left and right cameras are denoted by $\mathcal{I}_l$ and $\mathcal{I}_r$, respectively, the distance between the two centers of projection is the camera baseline T_c and the distance between the center of projection of a camera and its projection plane is the camera focal length f. We assume that both cameras have equal focal lengths. The midpoint $\mathbf{O}_c$ of the baseline is the center of the world coordinate system (X_w, Y_w, Z_w). The projection centers $\mathbf{O}_l$ and $\mathbf{O}_r$ are the origins of the left and right camera coordinate systems, (X_l, Y_l, Z_l) and

(X_r, Y_r, Z_r), respectively. The respective optical axes coincide with the Z_l, Z_r axes, which are parallel to the world axis Z_w. The X_l, X_r axes are parallel to the baseline and the Y_l, Y_r axes are perpendicular to the $X_w Z_w$ plane. The world coordinate system can be transformed into the left/right camera system by a translation by $T_c/2$:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} X_l - \frac{T_c}{2} \\ Y_l \\ Z_l \end{bmatrix} = \begin{bmatrix} X_r + \frac{T_c}{2} \\ Y_r \\ Z_r \end{bmatrix}. \tag{5.3.1}$$

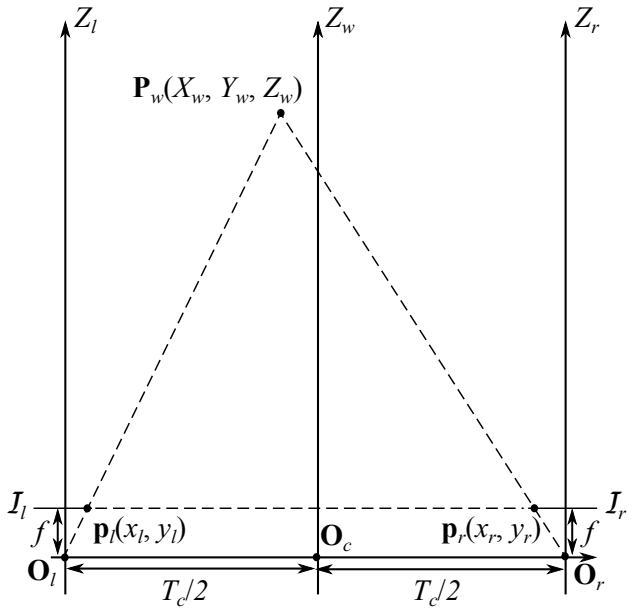


Figure 5.3.1: Parallel camera geometry.

A point of interest $\mathbf{P}_w = [X_w, Y_w, Z_w]^T$ in the world space is projected on the left and right image planes at the points $\mathbf{p}_l = [x_l, y_l]^T$ and $\mathbf{p}_r = [x_r, y_r]^T$, respectively. The notations $\mathbf{P}_l = [X_l, Y_l, Z_l]^T$ and $\mathbf{P}_r = [X_r, Y_r, Z_r]^T$ refer to the same point $\mathbf{P}_w$ in the left and right camera coordinate systems, respectively. The projections $\mathbf{p}_l$ and $\mathbf{p}_r$ are related to the 3D points $\mathbf{P}_l$ and $\mathbf{P}_r$ using perspective projection [TV98]:

$$x_l = f\frac{X_l}{Z_l}, \quad y_l = f\frac{Y_l}{Z_l}, \quad x_r = f\frac{X_r}{Z_r}, \quad y_r = f\frac{Y_r}{Z_r}. \tag{5.3.2}$$

Thus, the following equations give the transformation from the world space coordinates to the left/right camera plane coordinates:

$$x_l = f\frac{X_w + \frac{T_c}{2}}{Z_w}, \qquad y_l = f\frac{Y_w}{Z_w} \tag{5.3.3}$$

$$x_r = f\frac{X_w - \frac{T_c}{2}}{Z_w}, \qquad y_r = f\frac{Y_w}{Z_w}. \tag{5.3.4}$$

In order to recover the $\mathbf{P}_w$ world space coordinates from the $\mathbf{p}_l, \mathbf{p}_r$ projection, the triangle $(\mathbf{p}_l\mathbf{P}_w\mathbf{p}_r)$, $(\mathbf{O}_l\mathbf{P}_w\mathbf{O}_r)$ similarities can be used:

$$Z_w = -\frac{fT_c}{d_c}, \tag{5.3.5}$$

$$X_w = -\frac{T_c(x_l + x_r)}{2d_c}, \qquad Y_w = -\frac{T_c y_l}{d_c} = -\frac{T_c y_r}{d_c}, \tag{5.3.6}$$

where $d_c = x_r - x_l$ is the disparity of point $\mathbf{P}$. These are the triangulation formulas for the specific case of parallel stereo camera geometry. It must be obvious that such a camera setup produces non-positive disparity values. Therefore, during stereo projection, all points appear in front of the screen in the theater space, as detailed in Chapter 12. The points at infinity ($Z_l = Z_r = Z_w = \infty$) producing zero camera disparity, are displayed on the screen. The closer the 3D point is to the camera during filming, the larger its camera disparity is (in absolute value).

In the converging camera setup, the left and right camera optical axes form an angle θ with the coordinate axis Z_w, as shown in Figure 5.3.2. The left/right camera axes converge on the axis Z_w at distance Z_c from the camera centers. Z_c can be easily found to be $Z_c = \frac{T_c}{2}\frac{1}{\tan(\theta)} = \frac{T_c}{2}\tan\left(\frac{\pi}{2} - \theta\right)$. Typically, the angle θ is small, so that $\sin\theta \approx 0$, $\tan\theta \approx 0$ and $\cos\theta \approx 1$. Again, $\mathbf{O}_l$, $\mathbf{O}_r$, $\mathcal{I}_l$ and $\mathcal{I}_r$ are the projection centers and image planes of the left and right cameras, while f and T_c are the focal length and the distance between $\mathbf{O}_l$ and $\mathbf{O}_r$, respectively. The origin of the world space coordinate system $\mathbf{O}_c$ is placed at the midpoint between the left and right camera centers. The X_w axis of the world coordinate system is parallel to the baseline and the Y_w, Y_l, Y_r axes are perpendicular to the plane X_wZ_w. A point of interest $\mathbf{P}_w = [X_w, Y_w, Z_w]^T$ in the world space, which is projected on the left and right image planes at the points $\mathbf{p}_l = [x_l, y_l]^T$ and $\mathbf{p}_r = [x_r, y_r]^T$, respectively, can be transformed into the left camera system by translating it first

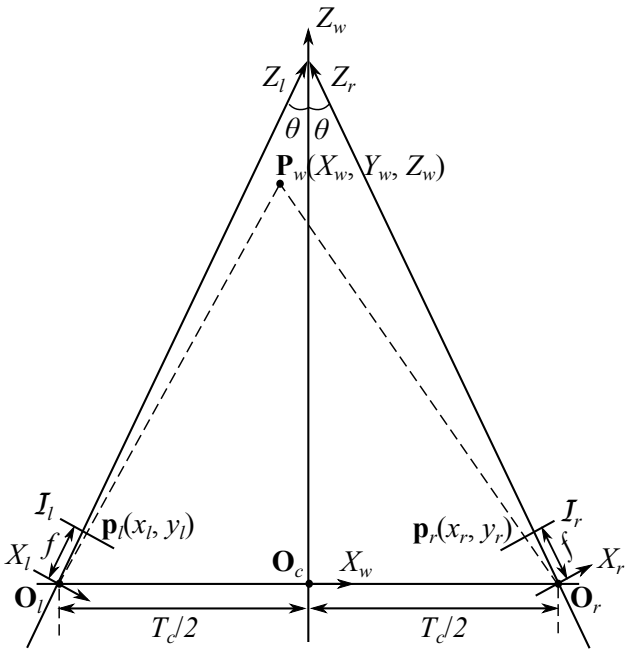


Figure 5.3.2: Converging camera geometry.

by $T_c/2$ and then rotating it by angle $-\theta$ about the Y_w axis:

$$\begin{bmatrix} X_l \\ Y_l \\ Z_l \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} X_w + \frac{T_c}{2} \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} \left(X_w + \frac{T_c}{2}\right)\cos\theta - Z_w \sin\theta \\ Y_w \\ \left(X_w + \frac{T_c}{2}\right)\sin\theta + Z_w \cos\theta \end{bmatrix}. \tag{5.3.7}$$

Similarly, for the right camera coordinates we have:

$$\begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} X_w - \frac{T_c}{2} \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} \left(X_w - \frac{T_c}{2}\right)\cos\theta + Z_w \sin\theta \\ Y_w \\ -\left(X_w - \frac{T_c}{2}\right)\sin\theta + Z_w \cos\theta \end{bmatrix}. \tag{5.3.8}$$

Using (5.3.2), the following equations map the world space coordinates to the left/right image plane coordinates:

$$x_l = f\frac{\left(X_w + \frac{T_c}{2}\right)\cos\theta - Z_w \sin\theta}{\left(X_w + \frac{T_c}{2}\right)\sin\theta + Z_w \cos\theta} = f\tan\left(\arctan\left(\frac{X_w + \frac{T_c}{2}}{Z_w}\right) - \theta\right), \tag{5.3.9}$$

$$y_l = f\frac{Y_w}{\left(X_w + \frac{T_c}{2}\right)\sin\theta + Z_w\cos\theta}, \tag{5.3.10}$$

$$x_r = f\frac{\left(X_w - \frac{T_c}{2}\right)\cos\theta + Z_w\sin\theta}{-\left(X_w - \frac{T_c}{2}\right)\sin\theta + Z_w\cos\theta} = -f\tan\left(\arctan\left(\frac{-X_w + \frac{T_c}{2}}{Z_w}\right) - \theta\right), \tag{5.3.11}$$

$$y_r = f\frac{Y_w}{-\left(X_w - \frac{T_c}{2}\right)\sin\theta + Z_w\cos\theta}. \tag{5.3.12}$$

In order to revert from the left/right image plane coordinates to the world space coordinates, the following equations can be used:

$$X_w = T_c\frac{x^l + \tan\theta\left(f + \frac{x^l x^r}{f} + x^r\tan\theta\right)}{x^l - x^r + \tan\theta\left(2f + 2\frac{x^l x^r}{f} - x^l\tan\theta + x^r\tan\theta\right)} - \frac{T_c}{2}, \tag{5.3.13}$$

$$Y_w = T_c\frac{y^l}{f}\frac{\cos\left(\arctan\left(\frac{x^l}{f}\right) + \theta\right)\cos\left(\arctan\left(\frac{x^l}{f}\right)\right)}{\sin\left(\arctan\left(\frac{x^l}{f}\right) + \arctan\left(\frac{x^r}{f}\right) + 2\theta\right)}, \tag{5.3.14}$$

$$Z_w = T_c\frac{f - \left(d + \frac{x^l x^r}{f}\tan\theta\right)\tan\theta}{x^l - x^r + \tan\theta\left(2f + 2\frac{x^l x^r}{f} - x^l\tan\theta + x^r\tan\theta\right)}. \tag{5.3.15}$$

The converging camera setup produces negative, zero, or positive camera disparities for objects having $Z_w < Z_c$, $Z_w = Z_c$ and $Z_w > Z_c$, respectively. Therefore, in the theater space, they appear to be in front, on or behind the screen plane, respectively. This is the main reason for choosing the converging camera setup, while shooting. Small convergence angle θ is chosen, since, otherwise, the distorting keystone effect [LNJ12] is significant. Objects at infinity ($Z_l = Z_r = Z_w = \infty$) have large positive disparity.

5.3.2 General 3D reconstruction in a calibrated stereo camera system

Triangulation for generic 3D reconstruction, applicable in all camera geometry models, is simple, if calibrated cameras are used. $\mathbf{P}$ must be located at the

intersection of two rays emanating from the optical centers of the cameras and passing through its left and right projections $\mathbf{p}_l$ and $\mathbf{p}_r$, respectively. Since camera calibration is frequently only approximate, the rays will usually not intersect, but pass near each other. Thus, the reconstructed point $\mathbf{P}'$ is estimated to lie at the midpoint of the shortest line segment connecting those two rays, as shown in Figure 5.3.3.

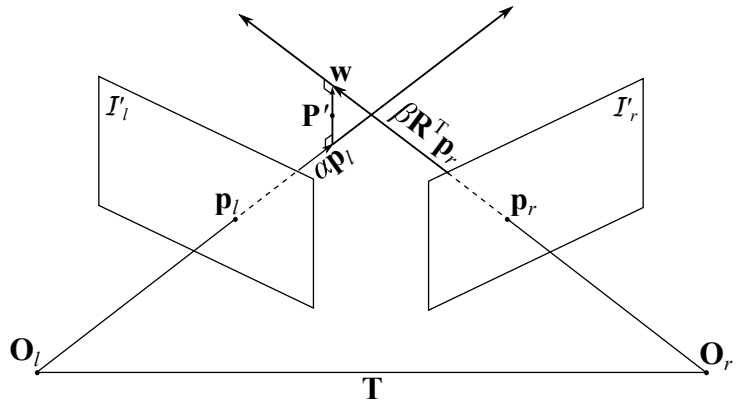


Figure 5.3.3: Triangulation with non-intersecting rays.

The location of $\mathbf{P}'$ can be calculated using the following procedure [TV98]. Let $\alpha\mathbf{p}_l$ be the left ray and $\mathbf{T} + \beta\mathbf{R}^T\mathbf{p}_r$ be the right ray, expressed in the left camera coordinate system. Let $\mathbf{w} = \mathbf{p}_l \times \mathbf{R}^T\mathbf{p}_r$ be a vector orthogonal to the left and right ray and $\alpha\mathbf{p}_l + \gamma\mathbf{w}$ be a line passing through the left ray point $\alpha\mathbf{p}_l$ and parallel to $\mathbf{w}$. Since the vectors $\alpha\mathbf{p}_l$, $\gamma\mathbf{w}$ and $-\beta\mathbf{R}^T\mathbf{p}_r$ sum to vector $\mathbf{T}$, the parameters α, β and γ can be determined by solving the equation:

$$\alpha\mathbf{p}_l + \gamma(\mathbf{p}_l \times \mathbf{R}^T\mathbf{p}_r) - \beta\mathbf{R}^T\mathbf{p}_r = \mathbf{T}. \tag{5.3.16}$$

This is a linear system of three equations and three unknowns α, β and γ. Once the parameters α, β and γ are calculated, the endpoints of $\mathbf{w}$ are given by $\alpha\mathbf{p}_l$ and $\beta\mathbf{R}^T\mathbf{p}_r$, respectively. The midpoint of vector $\mathbf{w}$ is the location of $\mathbf{P}'$.

Assuming that the two lines $\alpha\mathbf{p}_l$ and $\beta\mathbf{R}^T\mathbf{p}_r$ intersect, 3D reconstruction in a parallel and a converging camera setup are special cases of the general calibrated 3D reconstruction. It must be obvious, that dense disparity maps are required for a complete scene reconstruction, since pixel correspondences between the two views need to be estimated in order to determine the left and right projections $\mathbf{p}_l$ and $\mathbf{p}_r$ of each scene point $\mathbf{P}$. In the case of sparse disparity maps, only points projected onto pixels of known disparity can be reconstructed.

5.3.3 3D reconstruction from known intrinsic camera parameters only

In case only the intrinsic parameters of the cameras are known and at least eight point correspondences are established, 3D reconstruction is possible up to an unknown scaling factor. This factor can be determined if the distance between two points in the 3D world is provided.

The first step is the estimation of the essential matrix $\mathbf{E}$, as described in Section 4.2.4. Subsequently, the translation vector $\mathbf{T} = [T_x, T_y, T_z]^T$ can be calculated. $\mathbf{E}$ and $\mathbf{T}$ are related as follows [TV98]:

$$\mathbf{E}^T\mathbf{E} = \mathbf{S}^T\mathbf{R}^T\mathbf{R}\mathbf{S} = \mathbf{S}^T\mathbf{S} = \begin{bmatrix} T_y^2 + T_z^2 & -T_xT_y & -T_xT_z \\ -T_yT_x & T_z^2 + T_x^2 & -T_yT_z \\ -T_zT_x & -T_zT_y & T_x^2 + T_y^2 \end{bmatrix}, \quad (5.3.17)$$

where:

$$\mathbf{T}_\times = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}, \quad (5.3.18)$$

and

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} \quad (5.3.19)$$

Since the essential matrix $\mathbf{E}$ can be recovered up to an arbitrary scaling factor, it has to be normalized. The trace of the $\mathbf{E}^T\mathbf{E}$ is given by:

$$Tr(\mathbf{E}^T\mathbf{E}) = 2||\mathbf{T}||^2. \quad (5.3.20)$$

Thus, in order to normalize the length of the translation vector, the entries of the essential matrix must be divided by $\sqrt{Tr(\mathbf{E}^T\mathbf{E})/2}$, resulting in:

$$\tilde{\mathbf{E}}^T\tilde{\mathbf{E}} = \begin{bmatrix} 1 - \tilde{T}_x^2 & -\tilde{T}_x\tilde{T}_y & -\tilde{T}_x\tilde{T}_z \\ -\tilde{T}_y\tilde{T}_x & 1 - \tilde{T}_y^2 & -\tilde{T}_y\tilde{T}_z \\ -\tilde{T}_z\tilde{T}_x & -\tilde{T}_z\tilde{T}_y & 1 - \tilde{T}_z^2 \end{bmatrix}. \quad (5.3.21)$$

The normalized translation vector $\tilde{\mathbf{T}} = \mathbf{T}/||\mathbf{T}||$ can be estimated from any row or from the diagonal of the normalized matrix $\tilde{\mathbf{E}}^T\tilde{\mathbf{E}}$. Since this matrix contains quadratic components, the estimated $\tilde{\mathbf{T}}$ may have incorrect sign. Subsequently the rotation matrix $\mathbf{R}$ can be obtained using the following procedure. First, we define:

$$\mathbf{w}_i = \tilde{\mathbf{E}}_i \times \tilde{\mathbf{T}}, \quad (5.3.22)$$

$i = 1, 2, 3$, for each row of the normalized essential matrix. Each row of the rotation matrix can be computed as follows [TV98]:

$$\hat{\mathbf{R}}_i = \mathbf{w}_i + \mathbf{w}_j \times \mathbf{w}_k, \tag{5.3.23}$$

by setting (i, j, k) to be the cyclic permutations of numbers $(1, 2, 3)$. Now, the rotation matrix and translation vector estimates are found up to a scale and sign factor. There are four possible rotation matrices, because of the two-fold sign ambiguity of $\hat{\mathbf{T}}$ and $\hat{\mathbf{E}}$. To determine the correct $(\hat{\mathbf{T}}, \hat{\mathbf{R}})$ pair, the reconstructed depth components Z_l and Z_r of the feature points must be calculated [TV98]:

$$Z_l = f_l \frac{(f_r \mathbf{R}_1 - x_r \mathbf{R}_3)^T \hat{\mathbf{T}}}{(f_r \mathbf{R}_1 - x_r \mathbf{R}_3)^T \mathbf{p}_l} \tag{5.3.24}$$

$$Z_r = \mathbf{R}_3^T (\mathbf{P}_l - \hat{\mathbf{T}}). \tag{5.3.25}$$

As an object point resides in front of the left/right cameras (since both Z_l and Z_r should be positive numbers), the pair $(\hat{\mathbf{T}}, \hat{\mathbf{R}})$ which reconstructs each point $\mathbf{P}$ with positive Z_l and Z_r, is the correct one.

The coordinates of $\mathbf{P}_l$, $\mathbf{P}_r$ can be found from the relations

$$\mathbf{P}_l = \frac{Z_l}{f_l} \mathbf{p}_l, \qquad \mathbf{P}_r = \mathbf{R}(\mathbf{P}_l - \hat{\mathbf{T}}). \tag{5.3.26}$$

Since the entire computation is based on $\hat{\mathbf{E}}$, which is known up to a scale factor, the same holds for the 3D object points $\mathbf{P}_l$, $\mathbf{P}_r$.

5.3.4 Uncalibrated 3D reconstruction

Let us consider the case of uncalibrated 3D reconstruction, where no camera parameters are known and the only information available is the point correspondences. In this case, the 3D reconstruction is only possible up to an unknown projective transformation. Therefore, the actual absolute position, orientation and scale of the 3D scene remain undetermined, along with the true line parallelism or perpendicularity in the 3D scene. Formally, this is expressed through the following equations [HZ00]:

$$\mathbf{p}_l = \mathcal{P}_l \mathbf{P} = (\mathcal{P}_l \mathbf{H}^{-1})(\mathbf{H}\mathbf{P}) \tag{5.3.27}$$

$$\mathbf{p}_r = \mathcal{P}_r \mathbf{P} = (\mathcal{P}_r \mathbf{H}^{-1})(\mathbf{H}\mathbf{P}), \tag{5.3.28}$$

where $\mathbf{P}$ are the homogeneous coordinates of a 3D point, $\mathbf{H}$ is an unknown projective transformation matrix and $\mathcal{P}_l$/$\mathcal{P}_r$ are the left/right camera projection matrices, respectively. Intuitively, it is impossible to determine the matrix $\mathbf{H}$, without calibration information. Consequently, it is impossible to uniquely estimate the point $\mathbf{P}$ from its left and right projections $\mathcal{P}_l\mathbf{P}$ and $\mathcal{P}_r\mathbf{P}$, by applying the triangulation approach of Section 5.3.2. Essentially, we are forced to settle with an arbitrary selection for $\mathbf{H}$. Therefore, an inevitable projective ambiguity is inherent in the 3D reconstruction.

The first step for uncalibrated reconstruction is the estimation of the projection matrices of the cameras from a set of feature point correspondences. To this end, the eight-point algorithm described in Chapter 4 [Har97] can be applied on the correspondence pairs to recover the fundamental matrix $\mathbf{F}$, which relates the two views. The algorithm needs at least 8 feature correspondences in order to be applied, but can handle more points. However, the solution $\hat{\mathbf{F}}$ is sensitive to outliers. Therefore, the eight-point algorithm is typically coupled with RANSAC to increase robustness and significantly improve the results. Next, the two camera matrices $\mathcal{P}_l$ and $\mathcal{P}_r$ can be determined from the fundamental matrix, up to a projective ambiguity as previously discussed [HZ00]:

$$\hat{\mathcal{P}}_l = [\mathbf{I}|\mathbf{0}] \tag{5.3.29}$$

$$\hat{\mathcal{P}}_r = [\mathbf{E}'\hat{\mathbf{F}}|\mathbf{e}_r], \tag{5.3.30}$$

where $\mathbf{I}$ is the 3×3 identity matrix, $\mathbf{e}_r$ is the right epipole, $\hat{\mathbf{F}}$ is the fundamental matrix and $\mathbf{E}'$ is the 3×3 skew-symmetric matrix derived from $\mathbf{e}_r = [e_1, e_2, e_3]^T$, as follows:

$$\mathbf{E}' \triangleq \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix}. \tag{5.3.31}$$

The above equations represent the so-called *canonical form* of the projection matrices. Equation (5.3.29) is derived from (2.5.20) for $\mathbf{R} = \mathbf{I}$, $\mathbf{T} = 0$ and $f = 1$. It implicitly identifies the left camera reference frame with the world reference frame, which greatly simplifies the calculations. Note that, in principle, any skew-symmetric matrix can be used instead of the specific product $\mathbf{E}'\mathbf{F}$. A proof for the derivation of the canonical form can be found in [HZ00]. Finally, given the two computed camera matrices $\mathcal{P}_l$ and $\mathcal{P}_r$, the 3D point $\mathbf{P}$ can be estimated from its two projections in the usual manner, through triangulation.

With a small amount of extra information and minimal effort, the limitation of projective ambiguity can be bypassed and a *metric* 3D reconstruction of the scene,

i.e., a similarity transformation of the unknown real 3D scene, can be obtained from an uncalibrated stereo rig. A self-calibration method, such as the ones discussed in Chapter 4, based on Kruppa equations and the fundamental matrix, can be employed so that estimates of the intrinsic camera parameters can be acquired. Subsequently, the corresponding matrices $\mathbf{P}_{Il}$ and $\mathbf{P}_{Ir}$ have to be multiplied with the computed projection matrices of the projective 3D reconstruction, to arrive at more accurate projection matrices that can provide us with a metric (similarity) 3D scene reconstruction, using triangulation.

A different self-calibration approach, specifically oriented towards metric reconstruction, exploits the computed projective 3D reconstruction of the scene and incrementally refines it. The basic idea is the identification of the projection of the plane at infinity and the absolute conic lying in it on the two-dimensional images, described in Section 2.5.5 as the Image of the Absolute Conic (IAC). Once this is achieved, the metric geometry of the reconstructed scene can be extracted. This approach is called *stratified* [HZ00], because we proceed sequentially from a projective to an affine 3D reconstruction, where true line parallelism is recovered. Then we proceed from an affine to a metric 3D reconstruction and true line perpendicularity is recovered as well. However, in general, there are two solutions to the problem of estimating the absolute conic from two images. Thus, this metric 3D reconstruction is not unique in the case of stereo imaging.

One way to proceed, is by identifying the plane at infinity of the 3D scene, if possible. Thus, the affine geometry of the world is recovered, i.e., true line parallelism. Extraneous information must be exploited to identify the plane at infinity. One possible way is by detecting corresponding points with zero disparity in the two image views, which means that they are projections of 3D points lying at infinity. This approach obviously demands a reliable disparity estimation algorithm. The 3D back-projections of such corresponding pairs can be assumed to lie on the plane at infinity, at maximum distance from the cameras. This assumption is compatible with our everyday experience of faraway objects seeming to stay still as we perform a translational motion in real 3D space. The plane at infinity can be identified algebraically from three such points $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$, by using the cross-product of the two vectors formed by the points as the normal vector of the desired plane:

$$\mathbf{n} = (\mathbf{u}_2 - \mathbf{u}_1) \times (\mathbf{u}_3 - \mathbf{u}_2). \tag{5.3.32}$$

However, such an approach would require that the two cameras differ by a translation only, i.e., there is no rotation, as is the case of a parallel stereo camera setup. An alternative way to proceed, is to use knowledge concerning line parallelism:

if two lines are known to be truly parallel, their intersection in the 3D projective reconstruction is a point at infinity. With three such sets of known parallel lines, in most cases manually selected, as shown in Figure 5.3.4, the plane can be identified following the same procedure as above.

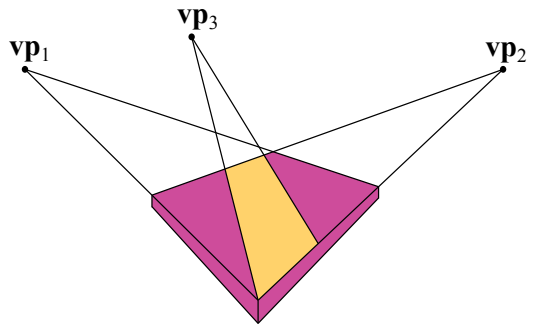


Figure 5.3.4: Recovery of the plane at infinity using three vanishing points $\mathbf{vp}_1$, $\mathbf{vp}_2$ and $\mathbf{vp}_3$.

Once the plane at infinity has been recovered, it can be represented by a 4-dimensional vector $\boldsymbol{\pi}$ in the coordinate system of the projective reconstruction. However, it is known from projective geometry that in the true 3D reconstruction the plane at infinity has coordinates $[0,0,0,1]^T$ [HZ00]. This allows finding a projective transformation that maps $\boldsymbol{\pi}$ to $[0,0,0,1]^T$. Such a transformation is the following one:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}|\mathbf{0} \\ \boldsymbol{\pi}^T \end{bmatrix}. \tag{5.3.33}$$

Subsequently, the transformation $\mathbf{H}$ can be applied to the two camera matrices (5.3.29)-(5.3.30) and to all the reconstructed points, resulting in the desired affine 3D reconstruction. Thus, the resulting affine camera matrices are:

$$\hat{\mathcal{P}_l^{af}} = [\mathbf{M}|\mathbf{m}] = [\mathbf{I}|\mathbf{0}]\mathbf{H} \tag{5.3.34}$$

$$\hat{\mathcal{P}_r^{af}} = [\mathbf{M}'|\mathbf{m}'] = [\mathbf{E}'\hat{\mathbf{F}}|\mathbf{e}_r]\mathbf{H}, \tag{5.3.35}$$

where $\mathbf{M}$ and $\mathbf{M}'$ are the first 3×3 sub-matrices of $\hat{\mathcal{P}_l^{af}}$ and $\hat{\mathcal{P}_r^{af}}$, respectively. A proof that a projective transformation fixing the plane at infinity is in fact an affine transformation is provided in [HZ00].

The next step towards metric reconstruction, involves estimating the IAC $\boldsymbol{\omega}$ in each image, mainly through extraneous information about perpendicular lines

in the 3D scene. If both cameras are known to share the same calibration matrix, it follows from (2.5.25) that $\boldsymbol{\omega} = \boldsymbol{\omega}_l = \boldsymbol{\omega}_r$. Since an affine reconstruction has already been computed, a transformation $\mathbf{H}_\infty$ can be derived that maps points from one image to the other via the known plane at infinity, by extending the ray corresponding to one image point until it intersects with the plane at infinity and then projecting this intersection point to the other image [HZ00]. The estimated $\mathbf{H}_\infty$ depends solely on the two projection matrices and the computed plane $\boldsymbol{\pi}$, and is given by [HZ00]:

$$\mathbf{H}_\infty = \mathbf{M}'\mathbf{M}^{-1}, \tag{5.3.36}$$

where $\mathbf{M}$ and $\mathbf{M}'$ are given by (5.3.34) and (5.3.35), respectively. Thus, a complete IAC can be mapped from one view to the other one via $\mathbf{H}_\infty$ as follows:

$$\boldsymbol{\omega}_r = {\mathbf{H}_\infty}^{-T} \boldsymbol{\omega}_l {\mathbf{H}_\infty}^{-1}. \tag{5.3.37}$$

Combining equations (2.5.23), (2.5.24) and (5.3.37) and exploiting the fact that $\boldsymbol{\omega} = \boldsymbol{\omega}_l = \boldsymbol{\omega}_r$, a system of linear equations can be formulated and solved to derive the entries of $\boldsymbol{\omega}$ [HZ00]. More than two views, assuming constant intrinsic camera parameters, would provide additional linear constraints and, therefore, lead to a more robust least-squares estimation of the IAC, by making the system heavily over-determined. Note, also, that a linear system can be formed by using only equation (5.3.37), thus relaxing the need for manual determination of vanishing points or lines. However, in this case, more than two views are necessary to get a unique solution [HZ00], since the system matrix (composed of the entries of $\mathbf{H}_\infty$) is rank-deficient. This limitation, however, can be bypassed by imposing hard constraints on the entries of $\boldsymbol{\omega}$, thus reducing the number of unknowns. Such constraints can be derived from knowledge of the camera geometry, which is captured in $\mathbf{P}_I$, e.g., assuming that the aspect ratio is equal to 1 for square pixels.

Whatever the case, one technique to exploit the derivation of the IAC is to directly compute $\mathbf{P}_I$ from $\boldsymbol{\omega}$, using equation (2.5.25) and carry on with calibrated 3D reconstruction. Alternatively, once the IAC has been recovered, the absolute conic can be identified by back-projecting $\boldsymbol{\omega}$ to a 3D cone that intersects the known plane at infinity in the desired conic. Subsequently, the latter can be mapped to the absolute conic in absolute Euclidean coordinates (known to satisfy the equation $X^2 + Y^2 + Z^2 = 0$ from projective geometry). If the transformation $\mathbf{H}_m$ that achieves this mapping, i.e., a projective transformation that fixes the absolute conic, is subsequently applied on the whole affine scene reconstruction, the result is a metric reconstruction.

Thus, the problem is reduced to calculating $\mathbf{H}_m$. Given the IAC $\boldsymbol{\omega}$ on any of the two images, as well as the related affine projection matrix $\mathcal{P}^{af}$, already known from the affine 3D reconstruction step, $\mathcal{P}^{af}$ may be decomposed into $[\mathbf{M}|\mathbf{m}]$, where $\mathbf{M}$ is the first 3×3 sub-matrix of $\mathcal{P}^{af}$. Subsequently, $\mathbf{H}_m$ is given by:

$$\mathbf{H}_m = \begin{bmatrix} \mathbf{A^{-1}} & \\ & 1 \end{bmatrix}, \tag{5.3.38}$$

where $\mathbf{A}$ is obtained by Cholesky factorization from the equation [HZ00]:

$$\mathbf{AA^T} = (\mathbf{M}^T \boldsymbol{\omega} \mathbf{M})^{-1}. \tag{5.3.39}$$

This technique indirectly exploits the relationship of the IAC with the intrinsic camera parameters, during the construction of $\mathbf{H}$, without explicitly computing $\mathbf{P}_I$.

A related, but not stratified, approach to uncalibrated metric reconstruction, involves direct recovery of the absolute conic from the two image views, without the intermediate affine reconstruction step, given the projective 3D reconstruction and the IAC estimation in each image. This straightforward process is based on back-projecting the two images of the absolute conic and taking their intersection as the absolute conic. However, in the case of stereo imaging, the two cones intersect in two different conics, located on different planes [HZ00]. The reason is that the intersection of two cones, in general, is a fourth-degree curve and, consequently, the computed curve has to be divided to two conics. Therefore, as previously discussed, the solution to the metric reconstruction problem from uncalibrated stereo is not unique.

5.4 Trinocular and n-view 3D reconstruction

Adding a third or more views of the 3D scene to the reconstruction process leads to an improvement of the final 3D reconstruction result, since more information about the 3D scene becomes available, at the expense of additional computational cost. The problem here is to incorporate this additional information into the procedures outlined for the stereo case.

The multiview image point correspondences obtained through feature matching, as already described in Section 5.2.2, are typically utilized to estimate the trifocal tensor relating 3-view images, as in the case of stereo. With more redundant information available, the effect of noise is reduced and, as already mentioned, increased robustness and superior results are achieved in comparison to the 2-view

case. The reason is that each 2D point correspondence now provides additional constraints (6 measurements instead of 4) for the same 3 degrees of freedom in 3D space [HZ00]. The reconstruction methodology outlined for stereo imaging is also applicable here, both in the calibrated and in the uncalibrated version, with the additional benefit of a unique solution to the metric reconstruction problem from uncalibrated cameras. Therefore, no reconstruction ambiguity exists up to a similarity transformation of the real 3D scene, even when no information about the camera parameters is provided.

The three camera matrices $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ can be extracted from the trifocal tensor, as described in Section 4.4.3. Given the camera projection matrices, the 3D world coordinates $[X_w, Y_w, Z_w]$ of each visible scene point $\mathbf{P}$ can be simply recovered through triangulation.

In the general N-view reconstruction, the tensor formulation of the problem is no longer applicable. Note also that, when all views share the same intrinsic camera parameters and the extrinsic parameters relating the pose of the corresponding cameras are initially unknown, e.g., in cases of multiple images taken with the same camera as it moves through space, the 3D reconstruction problem is traditionally called *Structure from Motion* [Pol00].

Assuming that the multiple image views $S_i,\ i = 1, \ldots, N$ are arranged in an ordered sequence $S = \{S_1, S_2, \ldots, S_N\}$, a common way to achieve N-view reconstruction [PCF05] is to sequentially retrieve the 2-view epipolar geometry among all pairs of consecutive views. During the first step of this process, a partial initial reconstruction is formed based on the first image pair, in the way outlined in previous sections for stereo imaging. The camera coordinate frame of the first view is set up as the world frame. In each subsequent step this partial reconstruction is refined and extended, incorporating information from the view $S_i,\ i = 2, \ldots, N$, currently under consideration. To this end, the pose of the camera corresponding to $S_i,\ i = 3, \ldots, N$ is recovered through feature matching with points already reconstructed in the projective reference frame defined by S_1. This is actually conventional calibration, since the projection matrix $\mathcal{P}_i$ of $S_i,\ i = 3, \ldots, N$ is estimated based on the correspondence between known, reconstructed 3D points and their image projection. The computed fundamental matrix relating S_i and S_{i-1} can be exploited to reliably detect feature matches with known points, through the epipolar constraint. Once the camera pose has been recovered, additional points available in S_i can be triangulated and merged with the partial reconstruction. Moreover, the 3D coordinates of any previously reconstructed points can be further refined, provided that they are present in S_i, by employing an Extended Kalman Filter as proposed in [Pol00]. Alternatively, a

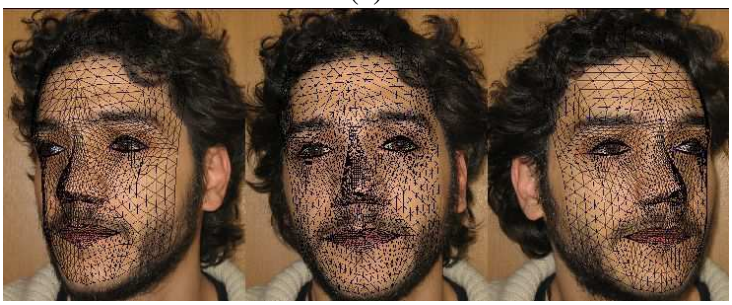
least-squares approach can be applied for each point, which simultaneously minimizes the distance between the predicted 2D projections and the observed image positions in all views, from S_1 up to S_i [PCF05]. The preceding procedure is repeated until the end of S is reached.

This sequential methodology has important limitations, since it may fail, e.g., in degenerate cases of camera motion. An alternative approach includes *factorization* methods, i.e., batch algorithms that compute camera pose and scene geometry by taking into account all views simultaneously. However, these methods also suffer in degenerate cases and demand all the 3D points to be present in every view. Therefore, it is much more common to couple standard sequential N-view reconstruction with a final iterative optimization step based on *bundle adjustment* [PCF05]. A maximum likelihood estimator is employed in order to find the N projection matrices $\hat{\mathcal{P}}_i,\ i = 1, 2, \ldots, N$, and the M 3D points $\hat{\mathbf{P}}_j,\ j = 1, 2, \ldots, M$, which minimize the mean squared distances between the image points $\mathbf{p}_{ij}$ and the corresponding reprojected image points $\hat{\mathbf{p}}_{ij}$ (projections of current $\hat{\mathbf{P}}_j$ to all image planes using the current matrices $\hat{\mathcal{P}}_i$). The Levenberg-Marquardt optimization method is typically used. However, any non-linear least squares algorithm can be utilized as well [LA05]. Bundle adjustment is a demanding minimization problem with $3M + 11N$ parameters, where M is the number of 3D points and N is the number of views. Moreover, the results heavily depend on the initialization of these parameters [HZ00]. However, the method can be efficiently implemented and is able to implicitly handle cases of corresponding points missing from consecutive views.

It must be noted that, without any prior calibration information, the result of the preceding methodology is a projective reconstruction. This can be subsequently upgraded to affine or metric reconstruction by autocalibration approaches, i.e., by determining the absolute conic. As in stereo imaging, autocalibration can be either stratified (from projective to affine and, afterwards, from affine to metric reconstruction) or direct (straight from projective to metric). In both cases, the image of the absolute conic is estimated over multiple views. For instance, the projection matrices of all views can be exploited to formulate an overdetermined linear system based on equation (5.3.37), allowing robust least-squares estimation of the IAC. Subsequently, the absolute conic is recovered and used to obtain a rectifying homography $\mathbf{H}$ that transforms the reconstructed points and projection matrices to a similarity transformation of the actual scene. Further details about such methods were covered in the previous section. Note, however, that many techniques for autocalibration have been proposed over the years, since in cases

of degenerate camera motions or temporally varying intrinsic camera parameters difficulties arise. Self-calibration for a N-view imaging setup or for the structure from motion problem, is still an active field of research.

(a)



(b)

Figure 5.4.1: a) 3 facial views, b) 3D reconstructed face.

An example of a 3D reconstruction from 3 consecutive views is shown in Figure 5.4.1. In case the 3D object geometry can be modeled, e.g., by the CANDIDE 3D human face model, 3D grid deformations can be combined with 3D reconstruction to attain better feature point matching, removal of outliers and improved 3D reconstruction accuracy. In the end, very small back-projection errors can be achieved.

5.5 3D surface triangulation

If correctly computed, 3D reconstruction using the previous methods, provides a cloud of isolated 3D points lying on the surface of the 3D object. Once the point cloud is produced, the 3D surface points must be triangulated to form

the triangles of the object surface mesh [NP01]. The method usually chosen for surface triangulation is *Delaunay triangulation*. Its basic property, is that any circle fitted to the vertices of a triangle does not contain other vertices. Such circles, shown in Figure 5.5.1, are called *circum-circles*. Delaunay triangulation maxi-

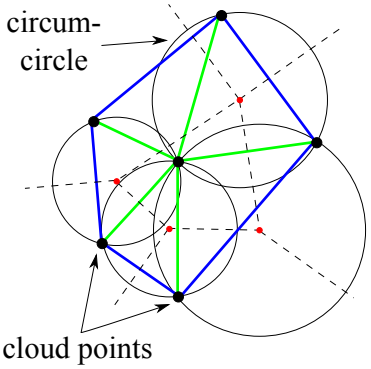


Figure 5.5.1: Delaunay triangulation of a point cloud.

mizes the minimum angles of all triangles, thus avoiding producing long thin triangles. Many algorithms have been developed to compute the Delaunay triangulation [NP01]. The basic operation of these algorithms, is testing if a given point is inside the circum-circle of a triangle, thus violating the circum-circle property. An alternative approach to triangulation is based on *Voronoi* tessellation, described in Chapter 1 [Vor08]. Delaunay triangle edges connect neighboring Voronoi region centers. The main concept of this approach is to connect the centers of neighboring Voronoi regions. It must be noted that, in the general 3D case, Delaunay triangulation produces 3D tetrahedra, rather than triangles and only their triangles that are on the 3D object surface must be retained during surface triangulation.

5.6 Volumetric 3D reconstruction

Volumetric 3D reconstruction produces volumetric 3D object models from multiple camera views. Volumetric methods usually require many more images than feature-based methods in order to achieve good 3D reconstruction results. Two main methodologies are presented here, namely the shape-from-silhouette and shape-from-photo-consistency including space carving. Additionally, a more recent approach based on the minimization of the surface integral of a cost func-

tion is introduced. Subsequently, the views are considered to be taken from calibrated cameras.

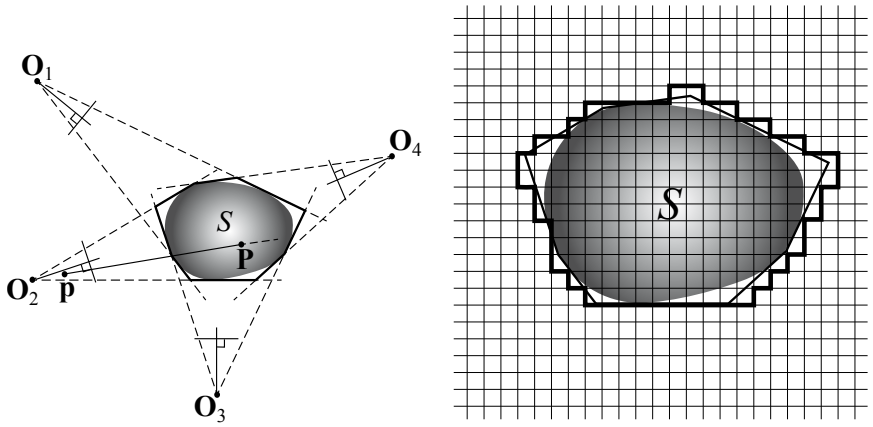
5.6.1 Shape-from-silhouette

Shape-from-silhouette, also known as *shape-from-contour*, is a 3D reconstruction methodology using the object outline for 3D shape recovery [SKS05]. A silhouette image is a binary image, where each pixel either belongs to the object (foreground) or to a background point. More specifically, each foreground pixel indicates if the ray emitted from the optical center of the camera intersects with the object surface in the 3D scene. A generalized *back-projected cone* is defined by each silhouette and the respective camera parameters. This cone, also known as *silhouette cone* or *visual cone*, contains the entire 2D object. The *visual hull* is the intersection of the silhouette cones taken from different views. The visual hull $VH(S,R)$ of a 3D object $S \in \mathbb{R}^3$ from a viewing region R, is a region in $\mathbb{R}^3$. For each point $\mathbf{P} \in VH(S,R)$ and each viewpoint $\mathbf{p} \in R$, the half-line beginning from $\mathbf{p}$ and passing through $\mathbf{P}$ contains one or more points of S [Lau94], as shown in Figure 5.6.1a. Essentially, the visual hull is the maximal object having the same silhouette as S from all viewpoints.

Unfortunately, the 3D object shape is only approximated by the visual hull [Lau94]. Although it can be accurately represented using polyhedra (leading to a polygonal surface representation), a volumetric discretization of the 3D space is usually used to produce a voxel volume, due to the low complexity for deriving this representation [NP01]. An example of the resulting discretized volume cross-section is shown in Figure 5.6.1b.

A better approximation of the visual hull can be derived by increasing the number of images obtained from different points of view [Lau97]. However, shape-from-silhouette is not robust when dealing with concave objects, even if an infinite number of images is available. In other words, such techniques are mostly effective when small and solid objects are captured. Visual hull techniques are quite robust, when applied to an indoor environment, with static illumination and camera. On the other hand, the construction of 3D object shapes in outdoor environments, where shadows and moving background are present, is more difficult, primarily due to poor object silhouette segmentation.

Since a visual hull is formed by the intersection of visual cones, its calculation is typically very simple. However, in case the 3D object has an irregular surface, visual hull estimation can be computationally expensive. In order to overcome such problems, volumetric approaches can be used for visual hull approximation



(a) (b)

Figure 5.6.1: a) Visual hull, b) discretized visual hull.

[Pot87], [NFA88], [AV89], [Sze93], leading to approximate volume representations, as shown in Figure 5.6.1b. An example of the application of the shape-from-silhouette algorithm is shown in Figure 5.6.2. Models for volumetric constructions, based on a set of silhouette images are presented in [Bau74], [MA87], [Pot87], [NFA88], [AV89], [SA90], [Sze93], [Lau94], [Lau95], [Lau97], [Nie97], [MBR$^+$00].

The shape-from-silhouette problem can be cast in a regularization framework, where the global energy minimization is required [SVZ00]. Two regularization terms are used. The first one is related to voxel transparency, while the second implements smoothness constraint around each voxel.

Shape-from-silhouette methods have been used in generating and replaying 3D digital video [MTG97], in commercial object modeling packages [NB94], in virtual reality [CZ00] and in building coarse scene models [CZ00].

5.6.2 Shape-from-Photo-consistency

Shape-from-silhouette takes advantage neither of the color nor of the grayscale texture of the object image. This photometric information can be used as a constraint, called *photo-consistency*, in order to improve 3D shape reconstruction. It refers to the fact that a valid 3D surface point must project to pixels of similar color across its N different image views. This constraint is exploited to determine proper voxel color and transparency properties in the discretized 3D space (voxel

volume), by testing the photo-consistency of each voxel as it is being projected to the N available image views of the scene. Such a test usually involves a thresholding operation, applied on the pixel color variance across the different views and leading to the removal of all voxels that exceed the threshold from the reconstruction. Shape-from-photo-consistency methods assume a Lambertian scene reflectance model and require an accurate camera projection matrix for each view. Shape-from-photo-consistency does not guarantee a unique 3D reconstruction of a scene, since it uses not only geometric information, but also scene illumination and surface reflectance models, resulting in artifacts and 3D reconstruction ambiguities [Cha97], [PD98], [CMS99], [KS00].

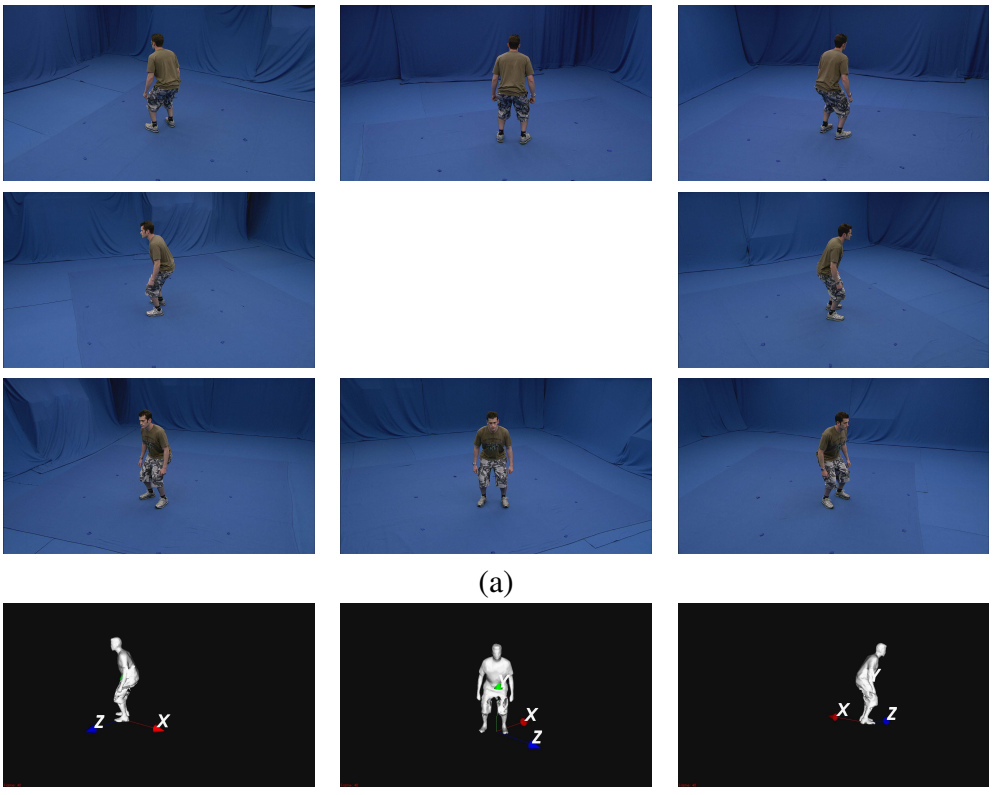


Figure 5.6.2: a) 8 views of a person, b) SFS reconstruction of the human body, as viewed from the left, center and right cameras *(Courtesy of the i3DPost FP7 project and the University of Surrey)*.

Space carving is an improvement over the shape-from-silhouette method, that

also considers the photo-consistency constraint. It overcomes the SFS inability to recover object surface concavities, by incorporating texture data in addition to silhouette contours [KS00]. The main idea behind space carving is to back-project the scene views into the voxel volume and then to “carve out" (make transparent) any voxel that is not photo-consistent with the available images, as shown in Figure 5.6.3. The voxel volume is said to be *photo-consistent*, if every reprojection of this volume onto the views results in the recovery of the original views. The result of space carving is a maximal but unique *photo hull* of the scene, which is the tightest possible volume containing all photo-consistent scene reconstructions. It also includes texture data and object concavities that are not present in a SFS-derived visual hull.

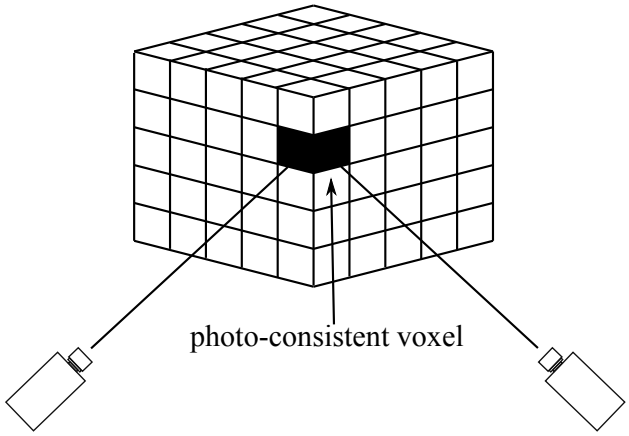


Figure 5.6.3: Space carving.

A space carving algorithm is introduced in [KS00]. Six planes, parallel to the cube sides that define the scene, sweep iteratively the voxels that are not photo-consistent. Without keeping a specific orientation of the sweep planes, the same technique can be applied to every border voxel, resulting in reduced efficiency. To overcome this problem, the voxel coloring algorithm [CMS99] is applied. Limitations present in traditional space carving are the unchecked propagation of reconstruction errors due to the erroneous carving out of a single voxel, resulting in 3D volume holes, as well as the restriction of the method applicability to highly textured Lambertian scene surfaces. Such drawbacks have been addressed in [BFK], where a probabilistic framework is employed to alleviate the former problem, and in [YPW03], where a novel photo-consistency metric is introduced to tackle the latter issue. Finally, a method based on geometric characteristics is presented in

[FK98], where surfaces defined by Euler-Lagrange equations approximate objects placed on the scene, resulting in a minimization problem.

Volumetric shape reconstruction becomes computationally expensive when large scale 3D scenes have to be reconstructed or when the reconstructed objects are far away from the camera. For this reason, coarse-to-fine strategies are adopted. For instance, 3D scene reconstruction with lower resolution voxels is proposed in [PD98]. Even though gaps from missing voxels may occur, simple methods like nearest neighbor search can be applied to fill these gaps. Another interesting fine-to-coarse approach is proposed in [KS00]. Photo-consistency is tested in a circle around the pixel in which the corresponding voxel is projected. This consistency is also known as r-consistency, where r is the circle radius.

5.6.3 Shape-from-surface integral minimization

More recently, a number of volumetric 3D reconstruction techniques based on the minimization of a cost function of the surface integral over the scene surface have emerged. Such a function may be derived, e.g., from the normalized cross-correlation of a small region in the tangent plane of the surface across the various image views. The optimization process may be initialized with a generic 3D volume, such as a sphere, that gradually evolves into the desired scene reconstruction. Several approaches of this nature have been proposed, including the use of the level set method to numerically solve systems of differential equations [JSY05], or the use of global optimization techniques such as graph-cuts [YAC06], [PSQ06], [LBN08]. Such algorithms may consider the photo-consistency constraint, but are typically designed to handle non-Lambertian surfaces, as well. They tend to produce successful reconstructions and do not suffer from an important drawback of the traditional space carving approach, the destructive accumulative effect of hard voxel removal decisions at each step over time. However, they exhibit structural bias and favour smaller shapes, thus limiting the recovery of details and surface extrusions.

5.7 Conclusion and further reading

Many techniques have been proposed for solving the problem of 3D shape reconstruction. This chapter presents two families of such techniques: the feature-based and the volumetric ones. Both consist of multitude of algorithms, have extensive literature coverage and remain active research fields.

The feature-based algorithms detect sparse point correspondences among multiple images of the 3D scene and exploit them to recover geometric relations among the different camera views. Subsequently, these relations are mathematically utilized to triangulate the 3D position of all scene points projected in more than one images, resulting in a 3D point cloud. Different results are achieved with calibrated and uncalibrated cameras, with the latter case being significantly more difficult to tackle. Additionally, several algorithmic details differ among the 2-view (stereo), the 3-view, the 4-view and the n-view case.

The volumetric methods exploit multiple images of the 3D scene to geometrically construct surface models that approximate the 3D shape of the imaged object and voxel arrays are commonly chosen for the representation of these surfaces. In general, volumetric algorithms require significant computational expense to produce results comparable to feature-based techniques. Other families of 3D reconstruction techniques are presented in the next Chapter.

Bibliography

[AV89] N. Ahuja and J. E. Veenstra. Generating octrees from object silhouettes in orthographic views. *IEEE Transanctions Pattern Analysis and Machine Intelligence*, 11(2):137–149, February 1989.

[Bau74] B. G. Baumgart. *Geometric modeling for computer vision*. PhD thesis, Stanford University, October 1974.

[Bea78] P. R. Beaudet. Rotationally invariant image operators. In *4th International Joint Conference on Pattern Recognition (ICPR '78)*, pages 579–583, November 1978.

[BFK] R. Bhotika, D. J. Fleet, and K. N. Kutulakos. A probabilistic theory of occupancy and emptiness. In *European Conference on Computer Vision (ECCV '02*.

[BL97] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pages 1000–1006. IEEE, June 1997.

[BTVG06] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision (ECCV '06)*, pages 404–417. Springer Berlin Heidelberg, September 2006.

[Cha97] S. S. Charles. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, pages 1067–1073, 1997.

[CMS99] W. B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. *International Workshop on Vision Algorithms*, pages 100–115, September 1999.

[CZ00] G. Cross and A. Zisserman. Surface reconstruction from multiple views using apparent contours and surface texture. *Confluence of Computer Vision and Computer Graphics*, pages 25–47, 2000.

[DF90] R. Deriche and O. D. Faugeras. Tracking line segments. *Image and Vision Computing*, 8(4):261–270, 1990.

[FB83] M. A. Fischler and R. C. Bolles. Perceptual organization and the curve partitioning problem. In *10th International Joint Conference on Artificial Intelligence (IJCAI '83)*, volume 2, pages 1014–1018. William Kaufmann, 1983.

[FK98] O. Faugeras and R. Keriven. Complete dense stereovision using level set methods. In *5th European Conference on Computer Vision (ECCV '98)*, pages 379–393. Springer-Verlag, 1998.

[Har97] R. I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, June 1997.

[HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference*, pages 147–151. The Plessey Company plc., September 1988.

[HZ00] R. I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 1st edition, 2000.

[JSY05] H. Jin, S. Soatto, and A. J. Yezzi. Multi-view stereo reconstruction of dense shape and complex appearance. *International Journal of Computer Vision*, 63(3):175–189, July 2005.

[KR82] L. Kitchen and A. Rosenfeld. Gray-level corner detection. *Pattern Recognition Letters*, 1(2):95–102, December 1982.

[KS00] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, July 2000.

[KS04] Y. Ke and R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, volume 2, pages 506–513. IEEE Computer Society, May 2004.

[LA05] M. I. A. Lourakis and A. A. Argyros. Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment? In *10th IEEE International Conference on Computer Vision (ICCV '05)*, volume 2, pages 1526–1531. IEEE Computer Society, October 2005.

[Lau94] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions Pattern Analysis Machine Intelligence*, 16(2):150–162, February 1994.

[Lau95] A. Laurentini. How far 3D shapes can be understood from 2D silhouettes. *IEEE Transanctions on Pattern Analysis and Machine Intelligence*, 17(2):188–195, February 1995.

[Lau97] A Laurentini. How many 2D silhouettes does it take to reconstruct a 3D object? *Computer Vision and Image Understanding*, 67(1):81–87, July 1997.

[LBN08] A. Ladikos, S. Benhimane, and N. Navab. Multi-view reconstruction using narrow-band graph-cuts and surface normal optimization. In *British Machine Vision Conference*, pages 15.1–15.10. BMVA Press, 2008.

[LNJ12] F. Liu, Y. Niu, and H. Jin. Keystone correction for stereoscopic cinematography. In *IEEE Computer Society Computer Vision and Pattern Recognition Workshops (CVPRW '86)*, pages 1–7. IEEE, June 2012.

[Low99] D. G. Lowe. Object recognition from local scale-invariant features. In *7th IEEE International Conference on Computer Vision (ICCV '99)*, volume 2, pages 1150–1157. IEEE, September 1999.

[LSP04] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *British Machine Vision Conference*, pages 779–788, September 2004.

[MA87] W. N. Martin and J. K. Aggarwal. Volumetric description of objects from multiple views. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, PAMI-5(2):150–158, March 1987.

[MBR+00] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. *27th annual Conference on Computer*

Graphics and Interactive Techniques (SIGGRAPH '00), pages 369–374, 2000.

[MS05] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005.

[MTG97] S. Moezzi, L.-C. Tai, and P. Gerard. Virtual view generation for 3D digital video. *IEEE Multimedia*, 4(1):18–26, January - March 1997.

[NB80] R. Nevatia and K. R. Babu. Linear feature extraction and description. *Computer Graphics and Image Processing*, 13(3):257–269, July 1980.

[NB94] W. Niem and R. Buschmann. Automatic modelling of 3D natural objects from multiple views. In *European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, pages 181–193. Springer-Verlag, 1994.

[NFA88] H. Noborio, S. Fukuda, and S. Arimoto. Construction of the octree approximating a three-dimensional object by using multiple views. *IEEE Transanctions on Pattern Analysis and Machine Intelligence*, 10(6):769–782, November 1988.

[Nie97] W. Niem. Error analysis for silhouette-based 3D shape estimation from multiple views. In *International Workshop on Synthetic-Natural Hybrid Coding and 3D Imaging*, pages 143–146, 1997.

[NP01] N. Nikolaidis and I. Pitas. *3D image processing algorithms*. Wiley-Interscience publication. Wiley & Sons, Inc., 1st edition, 2001.

[PCF05] N. Paragios, Y. Chen, and O. D. Faugeras. *Handbook of mathematical models in computer vision*. Springer, 2005.

[PD98] A. C. Prock and C. R. Dyer. Towards real-time voxel coloring. In *Proceedings of Image Understanding Workshop*, pages 315–321, 1998.

[Pit00] I. Pitas. *Digital image processing algorithms and applications*. AWiley-Interscience publication. Wiley & Sons, Inc., 1st edition, 2000.

[Pol00] M. Pollefeys. Tutorial on 3D modelling from images. *Lecture in conjunction with ECCV2000*, June 2000.

[Pot87] M. Potmesil. Generating octree models of 3D objects from their silhouettes in a sequence of images. *Computer Vision, Graphics and Image Processing*, 40(1):1–29, October 1987.

[PSQ06] S. Paris, F. X. Sillion, and L. Quan. A surface reconstruction method using global graph cut optimization. *International Journal of Computer Vision*, 66(2):141–161, February 2006.

[SA90] S. K. Srivastava and N. Ahuja. Octree generation from object silhouettes in perspective views. *Computer Vision, Graphics, and Image Processing*, 49(1):68–84, January 1990.

[SKS05] O. Schreer, P. Kauff, and T. Sikora. *3D Videocommunication: Algorithms, concepts and real-time systems in human centred communication*. John Wiley & Sons, July 2005.

[SSH10] M. Saquib Sarfraz and O. Hellwich. Probabilistic learning for fully automatic face recognition across pose. *Image and Vision Computing*, 28(5):744–753, May 2010.

[SVZ00] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 345–352, 2000.

[Sze93] R. Szeliski. Rapid octree construction from image sequences. *CVGIP: Image Understanding*, 58(1):23–32, July 1993.

[TV98] E. Trucco and A. Verri. *Introductory techniques for 3D computer vision*. Prentice Hall, 1998.

[Vor08] G. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les paralléllοèdres primitifs. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1908(134):198–287, January 1908.

[WB86] R. Weiss and M. Boldt. Geometric grouping applied to straight lines. In *IEEE Computer Society Computer Vision and Pattern Recognition Workshops (CVPRW '86)*, pages 489–495, 1986.

[WOZ02] Y. Wang, J. Ostermann, and Y.-Q. Zhang. *Video processing and communications*. Prentice Hall, 1st edition, 2002.

[YAC06] T. Yu, N. Ahuja, and W.-C. Chen. SDG cut: 3D reconstruction of non-lambertian objects using graph cuts on surface distance grid. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, volume 2, pages 2269–2276. IEEE, 2006.

[YPW03] R. Yang, M. Pollefeys, and G. Welch. Dealing with textureless regions and specular highlights - a progressive space carving scheme using a novel photo-consistency measure. In *9th International Conference on Computer Vision (ICCV '03)*, pages 576–584. IEEE, October 2003.

[ZH83] O. A. Zuniga and R. M. Haralick. Corner detection using the facet model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '83)*, pages 30–37. IEEE, 1983.