Chapter 2
Three-Dimensional Pose Estimation and Segmentation Methods

In the previous chapter it has been described how a three-dimensional reconstruction of a scene can be obtained from point correspondences between images, and how this problem is related with the problem of determining the intrinsic and extrinsic camera parameters. In this context, the reconstruction result is always obtained as a cloud of three-dimensional points. This point cloud contains information about the presence of matter at a certain position in space. However, no information is available about the presence or even the position and orientation of objects in the scene.

Hence, this chapter provides an analysis of traditional and newly introduced methods for the segmentation of point clouds and for the three-dimensional detection and pose estimation of rigid, articulated, and flexible objects in a scene based on single or multiple calibrated images, where it is assumed that a (more or less detailed) model of the object is available.

For rigid objects, pose estimation refers to the estimation of the six degrees of freedom of the object, i.e. three rotation angles and three translation components (cf. Sect. 2.1). For objects with a limited number of internal degrees of freedom ('articulated objects') and objects with an infinite number of internal degrees of freedom ('non-rigid objects') pose estimation may correspond to a much higher dimensional optimisation problem (cf. Sect. 2.2.1.2). Here, the three-dimensional object detection is based directly on the acquired images of the scene or suitable features derived from them. A class of complementary approaches deals with the analysis of point clouds obtained e.g. by stereo image analysis with respect to their three-dimensional structure and motion behaviour, aiming for a segmentation into physically meaningful domains representing individual objects or object parts. Such methods are regarded in Sect. 2.3.

2.1 Pose Estimation of Rigid Objects

To estimate the pose of a rigid object, a set of three-dimensional scene points with coordinates given by a geometry model of the object is brought into correspondence with points in the image such that the rotation angles and the translation of the object with respect to the camera can be determined by minimising a suitable error measure, e.g. the reprojection error of the model points. Accordingly, pose estimation of rigid objects is equivalent to the problem of determining the extrinsic camera parameters (cf. Sect. 1.4). This is an important insight, since all approaches described in Sect. 1.4 in the context of camera calibration can in principle be applied to pose estimation as well. Hence, an important issue in the context of pose estimation is to establish reliable correspondences between model points and points in the image. This section provides an overview of pose estimation techniques for rigid objects based on point features but also on edges or lines, and then regards in more detail the edge-based hierarchical template matching approach to pose estimation introduced by von Bank et al. (2003).

2.1.1 General Overview

2.1.1.1 Pose Estimation Methods Based on Explicit Feature Matching

A seminal work in the domain of three-dimensional pose estimation is provided by Haralick et al. (1989). They introduce several classes of pose estimation problems: 2D–2D pose estimation determines the pose of a two-dimensional object from an image, while 3D–3D pose estimation deals with three-dimensional data points from which the pose of a three-dimensional object is inferred. The estimation of the six pose parameters from a single two-dimensional image of the object, given a geometry model, or an estimation of the relative camera orientation based on two or more images of the object without making use of a geometry model of the object, corresponds to 2D–3D pose estimation. The solutions proposed by Haralick et al. (1989) are based on point correspondences between the images and the object model. They rely on a minimisation of the distances in three-dimensional space between the model points and the observed rays to the corresponding scene points. A linear method based on singular value decomposition is proposed to determine the rotation angles, and it is demonstrated that using a robust M-estimator technique instead of least-mean-squares minimisation may increase the performance of the optimisation procedure, as a reasonable pose estimation result is still obtained when the rate of outliers is as high as 50 %.

Image features beyond points are examined in the classical work by Lowe (1987). Linear structures, whose appearance does not change significantly even with strongly varying camera viewpoints, are grouped according to geometric criteria such as their similarity in orientation or the distances between their end points, a process termed ‘perceptual organisation’ by Lowe (1987). The three-dimensional model is projected into the image plane. The complexity of the subsequent step, during which correspondences between the model and the extracted image features are established, is reduced by prioritising those correspondences which involve features that are known to have a high probability of occurrence.

A further important classical approach to 2D–3D pose estimation, introduced by Lamdan and Wolfson (1988), is geometric hashing. An object is represented in terms of its geometric features, where coordinate systems are constructed based on specific features, especially points. For example, for a set of points lying in a plane, an orthogonal coordinate system is given by any pair of points on the plane, and the other points can be expressed in that coordinate system. All point pair-specific sets of coordinates (‘models’) along with the points that define them (‘basis pair’) are stored in what is called the ‘hash table’. Non-planar objects are assumed to be composed of planar parts by Lamdan and Wolfson (1988). The pose estimation process is then performed based on a random selection of pairs of observed scene points, where all other scene points are again expressed in the correspondingly defined coordinate system. Each selected pair of scene points increments a counter for one such model and basis pair stored in the hash table. All entries of the hash table for which the counter is larger than a previously defined minimum value are regarded as occurrences of the modelled object in the observed scene, where the parameters of the corresponding coordinate system denote the pose parameters, respectively.

An important edge-based approach to 2D–3D pose estimation is proposed by Lowe (1991). The object model is assumed to be composed of plane parts of polygonal shape as an approximation to the true surface shape. The contours of the object in the image are obtained by projecting the silhouette of the object into the image plane. Edge segments are extracted from the utilised greyscale image with the Canny edge detector (Canny, 1986). The error function is given by the sum of the perpendicular distances in the image plane between the extracted edge segments and the nearest projected object contour line. The pose parameters, including internal degrees of freedom of the object, are obtained by least-mean-squares minimisation of the error function with the Gauß–Newton method or alternatively with the Levenberg-Marquardt algorithm (Press et al., 2007), where the latter generally displays a more robust convergence behaviour.

A similar 2D–3D pose estimation approach based on point and line correspondences is described by Phong et al. (1996). They propose a quadratic error function by expressing the pose parameters in terms of a quaternion. Minimisation of the error function to determine the pose parameters is performed using a trust-region method, which yields a superior performance when compared with the Gauß–Newton method.

In the framework developed by Grebner (1994), 2D–3D pose estimation of industrial parts is performed based on edges and corner points, where the parameter search, corresponding to the minimisation of an appropriately chosen cost function, is performed using the A* algorithm (cf. e.g. Sagerer, 1985 for an overview).

2.1.1.2 Appearance-Based Pose Estimation Methods

A different class of methods consists of the appearance-based approaches, which directly compare the observed image with the appearance of the object at different poses without explicitly establishing correspondences between model features and parts of the image.

Methods Based on Monocular Image Data A probabilistic approach to simultaneous pose estimation and object recognition is proposed by Niemann and Hornegger (2001). However, they only regard the problem of 2D–2D pose estimation. They follow a statistical approach to object recognition, while localisation is performed based on estimating the corresponding parameters. An object model is represented by the position-dependent and typically multimodal probability densities of the pixel grey values in an image that displays an object of a certain class. Relying on empirical data, the number of the components of these multimodal distributions is determined by vector quantisation, while the parameters of the distribution are obtained with the expectation–maximisation algorithm. Hence, pose estimation is performed based on a maximum likelihood estimate.

In the monocular system of Kölzow and Ellenrieder (2003), a unified statistical approach for the integration of several local features, such as edges or textures, is employed. The utilised object model consists of CAD data complemented by the local features, resulting in an 'operator model'. The six pose parameters of a rigid object are then obtained by adaptation of the visible model features to the image. The accuracy of the obtained pose estimation results is discussed in Sect. 6.1.

Other pose estimation algorithms rely on geometric (edges) and on intensity (surface radiance) information. In this context, Nayar and Bolle (1996) introduce an object representation based on reflectance ratios, which is used for object recognition using monocular greyscale images. Pose estimation is performed relying on the reflectance ratio representation and a three-dimensional object model, thus taking into account physical properties of the object surface in addition to purely geometric information.

Another technique which relies on the simultaneous extraction of edge and shading information for 2D–3D pose estimation is the appearance-based approach proposed by Nomura et al. (1996), who utilise synthetic edge and intensity images generated based on an object model. A nonlinear optimisation procedure based on a comparison between the observed and the synthetic images yields the pose parameters. The accuracy of the obtained pose estimation results is discussed in Sect. 6.1.

The approach by Ando et al. (2005) is based on a set of grey value images of an object with known associated three-dimensional poses. After reducing the dimension of the images by principal component analysis, the rotational pose parameters of the object are learned directly from the image data using support vector regression.

The integrated 2D–3D pose estimation approach by Barrois and Wöhler (2007), which in addition to edges and surface brightness also takes into account polarisation and defocus features, is discussed in Sects. 5.6 and 6.1.

The three-dimensional pose estimation method of Lagger et al. (2008) is based on an 'environment map', i.e. a direction-dependent map of the light intensity and colour incident on the object surface. An image sequence showing the object is acquired, where a three-dimensional model of the object is available. For each image, the diffuse reflection component is separated from the specular component, where the latter is then used to infer the corresponding environment map. The pose parameters are refined by minimising an error measure which consists of a weighted sum of the negative normalised cross-correlation coefficients between the pixel grey values of the first and the current image and the squared differences between the corresponding environment maps.

Chang et al. (2009) propose two appearance-based methods for estimating the three-dimensional pose of objects with specularly reflecting surfaces based on CAD data. A mirror-like reflectance behaviour of the object surface is assumed. The first approach is based on the determination of the position and appearance of specular reflections in the image by rendering and a subsequent comparison with the observed image. The second technique relies on the 'specular flow', i.e. the optical flow associated with specularly reflecting surface parts. The accuracy of the obtained three-dimensional pose estimation results is discussed in Sect. 6.1.

Methods Based on Multiocular Image Data Classical monocular pose estimation approaches have in common that they are not able to estimate the distance to an object at high accuracy, since the only available depth information is the scale of a known object in the resulting image, and the appearance of the object in the image is not very sensitive to small depth variations. In comparison, for a convergent stereo setup with a baseline similar to the object distance, for geometrical reasons a depth accuracy of the same order as the lateral translational accuracy is obtainable. Accordingly, a variety of three-dimensional pose estimation methods relying on multiple images of the scene have been proposed.

The system of Bachler et al. (1999) performs a pose estimation of industrial parts based on a pair of stereo images in the context of taking the parts out of a bin with an industrial robot. In a first step, planes are detected in the scene relying on the projection of structured light. After isolating an object based on the plane detection result, the rotation angle around the optical axis is determined. Furthermore, the translational pose parameters parallel to the image plane are estimated using a CAD model, while the distance to the object is assumed to be known. The accuracy of the approach is discussed in Sect. 6.1.

A fast tracking algorithm for estimating the pose of an automotive part using a pair of stereo images is presented by Yoon et al. (2003). This method is regarded further in the application scenario of industrial quality inspection in Sect. 6.1.

Rosenhahn et al. (2003) introduce a method for appearance-based three-dimensional pose estimation from several images, regarding objects characterised by free-form surfaces modelled in terms of Fourier descriptors, thus providing a computationally favourable approximate representation on large spatial scales. The pose parameters are determined in the framework of conformal geometric algebra. The optimisation of the pose parameters is performed based on the appearance of the object contour projected into the image, applying the iterative closest point (ICP) algorithm by Zhang (1999b), which is especially designed for analysing free-form surfaces (cf. also Sect. 2.3).

Rosenhahn et al. (2006) compare the ICP algorithm for three-dimensional pose estimation in stereo image pairs with a level set approach formulated in terms of a computational technique from the field of optical flow analysis. The pose estimation is based on silhouettes. A quantitative evaluation of the two methods and their combination is performed. It demonstrates that the highest performance is achieved by a combination of both approaches, especially when regarding the convergence radius, i.e. the ability to converge towards the true pose from a considerably different initial pose.

The method of von Bank et al. (2003), which is described in detail in Sect. 2.1.2, is extended by Krüger (2007) to a multiocular setting characterised by three calibrated cameras. The accuracy of the three-dimensional pose estimation results obtained by Krüger (2007) is discussed in Sect. 6.1.

The object recognition and pose estimation system proposed by Collet et al. (2011) for the manipulation of objects by a robot relies on one or several calibrated images of the scene. Three-dimensional models of the objects are constructed using a structure from motion approach, where a model is associated with a set of features. The three-dimensional scene reconstruction and the estimation of the pose parameters are performed simultaneously based on the 'iterative clustering estimation' algorithm, where features detected in the images are clustered and associated with objects and their corresponding pose parameters in an iterative manner by employing robust optimisation methods. Collet et al. (2011) use the mean-shift algorithm as proposed by Cheng (1995) for clustering. The resulting object hypotheses are again clustered, using the 'projection clustering' approach, and a pose refinement is applied, which yields the objects in the scene with their associated pose parameters. An optimised hardware architecture using graphical processing units for the computationally complex parts of the algorithm, such as the feature detection step, results in cycle times of about two seconds for a real-world image sequence with 60 objects per image. The pose estimation accuracy of the system is discussed in Sect. 6.1.

2.1.2 Template-Based Pose Estimation

Many industrial applications of pose estimation methods for quality inspection purposes impose severe constraints on the hardware to be used with respect to robustness and easy maintenance. Hence, it is often not possible to utilise multiocular camera systems since they have to be recalibrated regularly, especially when the sensor unit is mounted on an industrial robot. As a consequence, employing a monocular camera system may be favourable from a practical point of view, while nevertheless a high pose estimation accuracy is required to detect subtle deviations between the true and the desired object pose.

The presentation in this section is adopted from von Bank et al. (2003). The appearance-based 2D–3D pose estimation method described in this section involves a viewer-centred representation of the image data. The views are generated automatically from a three-dimensional object model by rendering, and the pose parameters of each view are stored in a table. Edge templates are computed for each view. For the input image, the best-fitting template and thus the corresponding pose parameters are determined by a template matching procedure. The difficult trade-off between the tessellation constant, i.e. the difference between the pose parameters of neighbouring views, and the accuracy of pose estimation is alleviated by a technique for hierarchical template matching (Gavrila and Philomin, 1999).

The input image first undergoes an edge detection procedure. A distance transform (DT) then converts the segmented binary edge image into what is called a distance image. The distance image encodes the distance in the image plane of each image point to its nearest edge point. If we denote the set of all points in the image as $A = \{{}^S\mathbf{a}_1, \ldots, {}^S\mathbf{a}_N\}$ and the set of all edge points as $B = \{{}^S\mathbf{b}_1, \ldots, {}^S\mathbf{b}_M\}$ with $B \subseteq A$, then the distance $d({}^S\mathbf{a}_n, B)$ for point ${}^S\mathbf{a}_n$ is given by

$$d\left({}^S\mathbf{a}_n, B\right) = \min_m\left(\left\|{}^S\mathbf{a}_n - {}^S\mathbf{b}_m\right\|\right), \tag{2.1}$$

where $\|\ldots\|$ is a norm on the points of A and B (e.g. the Euclidean norm). For numerical simplicity we use the chamfer-2–3 metric (Barrow, 1977) to approximate the Euclidean metric.

The chamfer distance $D_C(T, B)$ between an edge template consisting of a set of edge points $T = \{{}^S\mathbf{t}_1, \ldots, {}^S\mathbf{t}_Q\}$ with $T \subseteq A$ and the input edge image is given by

$$D_C(T, B) = \frac{1}{Q}\sum_{n=1}^{Q} d\left({}^S\mathbf{t}_n, B\right). \tag{2.2}$$

A correspondence between a template and an image region is assumed to be present once the distance measure (‘dissimilarity’) $D(T, B)$ becomes smaller than a given threshold value θ. To reduce false detections, the distance measure was extended to include oriented edges (Gavrila and Philomin, 1999).

In order to recognise an object with unknown rotation and translation, a set of transformed templates must be correlated with the distance image. Each template is derived from a certain rotation of the three-dimensional object. In previous work, a uniform tessellation often involved a difficult choice for the value of the tessellation constant. If one chooses a relatively large value, the views that lie ‘in between’ grid points on the viewing sphere are not properly represented in the regions where the aspect graph is undergoing rapid changes. This decreases the accuracy of the measured pose angles. On the other hand, if one chooses a relatively small value for the tessellation constant, this results in a large number of templates to be matched online; matching all these templates sequentially is computationally intensive and prohibitive to any real-time performance. The difficult trade-off regarding tessellation constant is alleviated by a technique for hierarchical template matching, introduced by Gavrila and Philomin (1999). That technique, designed for distance transform-based matching, in an offline stage derives representation which takes into account the structure of the given distribution of templates, i.e. their mutual degrees of similarity. In the online stage, this approach allows an optimisation of the matching

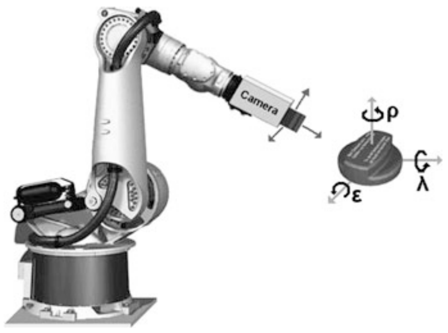


Fig. 2.1 Sketch of the robot-based inspection system with a definition of the pose angles ε (roll), λ (pitch), and ρ (yaw) in the camera coordinate system

procedure by representing a group of similar templates by a single prototype template and a distance parameter. A recursive application of this grouping scheme eventually results in a hierarchy of templates, which is generated in a bottom-up manner using simulated annealing-based partitional clustering. Each node of the hierarchical tree of templates requires matching of a prototype template **p** with a part of the image. To avoid an exhaustive search, locations of interest for the nodes in the hierarchy one level below a certain prototype template ('children nodes') are determined by selecting the positions for which the distance measure between the respective prototype and the image falls below a predefined threshold θ_p. This procedure is repeated for the children nodes until the tree has been traversed. If, on the other hand, the distance measure exceeds θ_p, the search is not continued in the next level of the tree. The high computational efficiency of this method results from the fact that, according to the hierarchical approach, only a small fraction of the templates actually need to be matched with parts of the image.

In our system, we do not need to estimate scale—the distance to the object is assumed to be known at an accuracy of better than 3 % due to the fact that the system is designed for an industrial quality inspection scenario in which the approximate position of the parts is provided by CAD data. Template matching does not have to search all scales explicitly. Hence, the original pose estimation problem of determining six degrees of freedom can be reduced to a five-degree-of-freedom (three pose angles and two image position coordinates) problem.

For pose fine-tuning, the pose angles are interpolated between the n_b 'best' template matching solutions, with $n_b = 30$ in our system. This is justified, because in our pose estimation scenario the dissimilarity values of the 30 best solutions usually do not differ by more than about 20 %, and thus all these solutions contain a significant amount of information about the pose.

In many applications, templates are generated from real-world image data (Demant, 1999). For inspection tasks, however, one can assume that a CAD model of the object to be inspected is available. We therefore generate realistic two-dimensional templates from CAD data using the public domain software POVRAY, simulating the properties of the surface material and the illumination conditions by employing ray tracing techniques. The pose of the object is defined by the three angles ε (roll), λ (pitch), and ρ (yaw), as shown in Fig. 2.1. Typical matching results are shown

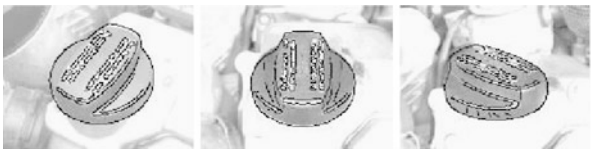


Fig. 2.2 Matching results (best solution) for several example poses

for an automotive part in Fig. 2.2. A quantitative evaluation of the described edge-based pose estimation technique is performed in the scenario of industrial quality inspection in Sect. 6.1.1.

To improve the robustness of monocular pose estimation in the presence of a cluttered background, the edge-based method is extended by Barrois and Wöhler (2007) to appearance-based monocular pose estimation based on geometric, photopolarimetric, and defocus cues (cf. Sect. 5.6).

2.2 Pose Estimation of Non-rigid and Articulated Objects

In contrast to rigid objects, articulated objects consist of several rigid subparts which are able to move with respect to each other. Methods that aim for a pose estimation of such objects need to determine these internal degrees of freedom in addition to the six rotational and translational degrees of freedom encountered for rigid objects. Non-rigid objects have no rigid subparts at all and therefore have an infinite number of internal degrees of freedom. In this section we first give an overview of pose estimation methods for articulated and non-rigid objects. Then we regard the contour-based algorithm of d'Angelo et al. (2004) for three-dimensional reconstruction of non-rigid objects such as tubes and cables, which may be regarded as a multiocular extension of the concept of active contours (Blake and Isard, 1998). Subsequently, the multiocular contracting curve density algorithm (Hahn et al., 2007, 2010a) is described, which allows a three-dimensional reconstruction of non-rigid objects and a pose estimation of articulated objects in the presence of a cluttered background.

2.2.1 General Overview

2.2.1.1 Non-rigid Objects

The extraction of two-dimensional object contours from images is an essential part of image segmentation. A classical approach to this problem is that of active contours or snakes. The original snake algorithm by Kass et al. (1988) determines a curve for which an optimisation is performed simultaneously in terms of its length or curvature and its correspondence with edges in the image. Many variations and improvements of the original snake algorithm have been proposed, such as 'balloon snakes' (Cohen, 1991), 'ziplock snakes' (Neuenschwander et al., 1997), 'gradient vector field snakes' (Xu and Prince, 1998), and implicit active contour models (Caselles et al., 1995, 1997; Sethian, 1999).

In the balloon snake approach of Cohen (1991), the adaptation of the curve to intensity gradients in the image is based on an 'inflation force', which increases the robustness of the contour adaptation in the presence of small edge segments which do not correspond to the true object border. The ziplock snake algorithm of Neuenschwander et al. (1997) is described in Sect. 2.2.2. Xu and Prince (1998) adapt a contour to the intensity gradients in the image based on a 'generalised gradient vector field'. In their approach, the process of the adaptation of the curve to the image is described by a set of partial differential equations modelling a force exerted on the contour, which decreases smoothly with increasing distance from an intensity gradient. In the approach by Caselles et al. (1997), the contour adaptation process is performed in a Riemannian space with a metric defined according to the image intensities. The contour adaptation thus corresponds to the minimisation of the length of a curve in that space, leading to a set of partial differential equations describing the adaptation of the curve to the image.

In many medical imaging applications, volumetric data need to be analysed, leading to the three-dimensional extension of the snake approach by Cohen and Cohen (1993). For pose estimation of non-rigid objects from multiple images, it is assumed by most approaches that the non-rigid object is adequately described by a one-dimensional curve in three-dimensional space. Such techniques are primarily useful for applications in medical imaging, e.g. for the extraction of blood vessels from a set of angiographies (Cañero et al., 2000) or for the inspection of bonding wires on microchips (Ye et al., 2001). A related method for extracting the three-dimensional pose of non-rigid objects such as tubes and cables from stereo image pairs of the scene based on three-dimensional ribbon snakes is described in detail later in this section.

For the segmentation of synthetic aperture radar images, which typically display strong noise, Gambini et al. (2004) adapt B-spline curves to the fractal dimension map extracted from the image, where gradients of the inferred fractal dimension are assumed to correspond to the borders of contiguous image regions.

Mongkolnam et al. (2006) perform a colour segmentation of the image in a first step and then adapt B-spline curves to the resulting image regions. The border points extracted by the colour segmentation step do not have to lie on the adapted curve, such that the approach yields smooth borders of the extracted regions.

Another approach to two-dimensional curve fitting is the contracting curve density (CCD) algorithm introduced by Hanek and Beetz (2004). The CCD algorithm employs a likelihood function as a quantitative measure of how well a curve is able to describe the boundary between different image regions, where the degree of similarity is described by the local probability distributions of the pixel grey values determined based on the local vicinity of the expected curve. The posterior probability density is iteratively maximised with respect to the parameters of the curve model, which are defined by a Gaussian distribution rather than a set of sharply defined quantities. Adapting this 'blurred curve model' (Hanek and Beetz, 2004) to the local statistical properties of the pixel grey values yields a large convergence radius and at the same time a high accuracy of the determined boundary. As a result, the CCD algorithm is capable of separating objects with ill-defined outlines from a cluttered background. Section 2.2.3 describes in detail a multiocular variant of the CCD algorithm developed by Hahn et al. (2007, 2010a).

Ellenrieder (2004) proposes a three-dimensional active contour method which incorporates a technique similar to the shape from texture approach (Jiang and Bunke, 1997) to estimate the normal of a surface displaying a pronounced texture (e.g. the surface of a tube laminated by textile material) based on spatial variations of the amplitude spectrum of the surface texture. In the context of industrial quality inspection of tubes and cables, Ellenrieder (2005) introduces a method for three-dimensional pose estimation of non-rigid objects which is based on the analysis of the contour of the shadow of the non-rigid object cast on a surface of known shape under known illumination conditions.

An approach to the computation of the derivatives of the bundle adjustment error function (1.25) for non-rigid objects is introduced by Krüger (2007), who adapts the model to the image based on a gradient descent scheme. The method relies on the sign of the gradient of the error function, and its determination reduces to one look-up per feature for which a correspondence with the image is established in a table that is computed offline. Once the space of pose parameters is divided into appropriate sections, it is sufficient to memorise one bit, denoting the sign of the gradient of the error function, for each pixel, each pose parameter of the regarded pose estimation problem, and each defined section in the pose parameter space. These bit matrices, which are termed 'gradient sign tables' by Krüger (2007), have the same size as the image. The computational complexity of this optimisation approach is quite low, while its memory demand may become fairly high.

2.2.1.2 Articulated Objects

Most pose estimation approaches regarding articulated objects address the scenario of human body pose estimation (cf. e.g. Moeslund et al. (2006) for an introduction to and overview of the large field of pose estimation and tracking of the human body).

Many approaches, especially those aiming for gesture recognition in the context of human–robot interaction, rely on monocular image sequences. A more detailed overview of such techniques is thus given in Sect. 7.1.2. As an example, Schmidt et al. (2006) adapt a three-dimensional articulated body model consisting of chains of cylinders to monocular colour images of the person, where the optimisation of the pose parameters basically relies on skin colour detection as well as on intensity, edges, and the spatially varying statistical distribution of colour cues. Sminchisescu (2008) provides a broad discussion of the advantages and limitations resulting from monocular body pose estimation. Specifically, the problem of ambiguities of the appearance of the object in the image as a result of the projection from the three-dimensional scene into the two-dimensional image plane is addressed, which may lead to reconstruction errors, especially when partial self-occlusions of the body occur. Body pose estimation methods are divided by Sminchisescu (2008) into generative algorithms, relying on a model of the observation likelihood which is supposed to obtain its maximum value once the pose parameters have been estimated correctly, and discriminative algorithms, which learn the probability distribution of the pose parameters from examples and predict them using Bayesian inference.

An early approach by Gavrila and Davis (1996) to full body pose estimation involves template matching in several distance-transformed images acquired from different viewpoints distributed around the person. Plänkers and Fua (2003) and Rosenhahn et al. (2005) apply multiple-view three-dimensional pose estimation algorithms which are based on silhouette information.

Plänkers and Fua (2003) make use of three-dimensional data generated by a stereo camera to obtain a pose estimation and tracking of the human upper body. The upper body is modelled with implicit surfaces, and silhouettes are used in addition to the depth data to fit the surfaces. Lange et al. (2004) propose a method for tracking the movements of a human body in a sequence of images acquired by at least two cameras based on the adaptation of a three-dimensional stick model with a stochastic optimisation algorithm. A comparison between the appearance of the stick model after projection into the image plane with the acquired images yields an appropriate error function. A refinement of the correspondingly estimated joint angles of the stick model is obtained based on several pairs of stereo images.

Rosenhahn et al. (2005) track the upper body of a person, which is represented by a three-dimensional model with 21 body pose parameters consisting of connected free-form surfaces. The pose estimation is based on silhouettes which are extracted using level set functions. Tracking is performed by using the pose in the last frame as the initial pose in the current frame. Using images acquired synchronously by four cameras distributed around the person, they achieve a high reconstruction accuracy of about $2°$ for the joint angles under laboratory conditions without a cluttered background. As a ground truth, the joint angles determined by a commercial marker-based tracking system with eight cameras are used. In an extension of this method by Brox et al. (2008), the silhouette of the person is inferred from a single image or several images acquired by cameras distributed around the person, and a three-dimensional model representing the body surface is adapted to the silhouettes. The procedures of pose estimation and silhouette extraction based on level sets are alternated in order to allow tracking in scenes with a non-uniform and non-static background. In this context, Bayesian inference involving the local probability density models of image regions with respect to different features such as grey value, RGB colour values, or texture is used for simultaneously extracting a contour and a set of pose parameters. For large pose differences between successive images, prediction of the pose is achieved based on the optical flow. Since the pose estimation yields correspondences between the two-dimensional silhouettes in the images and the three-dimensional body model, while the optical flow yields correspondences between two-dimensional image positions in the current and the subsequent time step, the three-dimensional pose of the body model which is consistent with the optical flow can be predicted for the subsequent time step. A priori knowledge about likely and unlikely configurations of joint angles is used to impose constraints on the pose by learning the probability distribution of the joint angles from examples and incorporating it as an a priori probability into the Bayesian inference scheme.

Rosenhahn et al. (2008a) propose a method for tracking the motion of dressed persons by integrating a kinematic simulation of clothes covering parts of the body into a silhouette-based system for body pose estimation. The body pose is determined by minimising an appropriately defined error function, where the correspondences between the observed and modelled silhouettes, the parameters of the kinematic chain defining the body model, the appearance of the clothes on the body, and the forces exerted on the clothes are taken into account. A fairly detailed modelling is performed, since parameters of the clothes of the person such as the length of a skirt are extracted during the optimisation process, and physical kinematic modelling of the motion of the clothes resulting e.g. from wind is performed. A quantitative evaluation demonstrates that despite the fact that parts of the tracked body are occluded by clothes, the error of the proposed method is less than one degree higher than typical inaccuracies of tracking systems relying on markers attached to the person.

Grest and Koch (2008) adapt a three-dimensional body model consisting of rigid fixed body parts to a three-dimensional point cloud extracted from a pair of stereo images with a dynamic programming-based dense stereo technique. A maximum number of 28 pose parameters is estimated for the human body model using a 3D–3D pose estimation technique based on the ICP algorithm. The Gauß–Newton, gradient descent, and stochastic meta-descent optimisation methods are compared with respect to their convergence behaviour, where the Gauß–Newton method is found to be the superior approach.

A markerless system for three-dimensional body pose estimation specifically designed for the distinction between normal and pathological motion behaviour of a person is described by Mündermann et al. (2008). It is based on an ICP technique involving an articulated surface model designed such that the exact positions at which the articulated motion is performed within the joints can be refined (within certain limits) during the model adaptation procedure, where multiple (between 4 and 64) images of the scene acquired from viewpoints distributed around the person are used. Segmentation of the person from the background is performed by applying an intensity and colour threshold to the background-subtracted images, which yields a three-dimensional visual hull of the person to which the articulated model is adapted. A direct comparison to a marker-based body pose estimation system yields accuracies of 10.6 ± 7.8 mm, 11.3 ± 6.3 mm, and 35.6 ± 67.0 mm for the full body and 8.7 ± 2.2 mm, 10.8 ± 3.4 mm, and 14.3 ± 7.6 mm for the lower limbs of the human body for 64, 8, and 4 cameras, respectively.

Gall et al. (2009) rely on silhouette and colour features and assume the existence of an accurate three-dimensional model of the analysed human body. They utilise a local optimisation technique for three-dimensional pose estimation which is similar to the approach introduced by Rosenhahn et al. (2005), combined with a global stochastic optimisation and filtering stage based on a technique named ‘interacting simulated annealing’.

Hofmann and Gavrila (2009) suggest an extraction of three-dimensional human body pose parameters from single images based on a hierarchical matching scheme similar to the one described by Gavrila and Philomin (1999) and a simultaneous adaptation of the parameters of the human body model based on multiple image sequences acquired from different viewpoints distributed around the observed persons. The results are integrated over time relying on representative sequences of three-dimensional pose parameters extracted from the acquired image sequences, which are used to generate a model of the apparent surface texture of the person. Texture information is used in combination with contour information to arrive at a final estimate of the three-dimensional pose parameters.

Salzmann and Urtasun (2010) introduce a method which combines discriminative approaches (such as regression or classification techniques) estimating the three-dimensional pose of an articulated object and the three-dimensional shape of an arbitrary surface with the minimisation of a likelihood function depending on the image information (such as the reprojection error or the distances between edges generated by the three-dimensional model pose and those observed in the image). ‘Distance preservation constraints’ are introduced by Salzmann and Urtasun (2010) into the estimation of the three-dimensional pose parameters. These constraints impose constant distances between reference points, i.e. the joints in the case of an articulated human body model and the mesh points in the case of a surface model.

Approaches like those introduced by Plänkers and Fua (2003), Rosenhahn et al. (2005), and Brox et al. (2008) determine a single pose which is updated at every time step. A more refined tracking scheme is proposed by Ziegler et al. (2006), who employ an unscented Kalman filter for tracking the pose parameters of the body. An ICP-based approach to estimate the pose of the upper human body is used, relying on a comparison of a three-dimensional point cloud generated by the analysis of several pairs of stereo images with a synthetically rendered depth map, obtained with a polygonal model of the upper body using the z-buffer technique. The system of Ziegler et al. (2006) determines the position of the torso along with the joint angles of the upper arms at the shoulders and the joint angles characterising the postures of the forearms relative to the upper arms.

To increase the robustness of tracking, other approaches such as those proposed by Deutscher et al. (2001) and Schmidt et al. (2006) rely on the particle filter approach introduced by Isard and Blake (1998) in order to take into account multiple pose hypotheses simultaneously. In this probabilistic framework, the probability distribution of the parameters to be estimated is modelled by a (typically large) number of random samples (‘particles’). For tracking the human body based on a small number of particles, Deutscher et al. (2001) introduce an approach inspired by the optimisation technique of simulated annealing, termed ‘annealed particle filtering’, which allows one to determine the absolute maximum of the (generally multimodal) probability distribution of the pose parameters by introducing several subsequent annealing stages. For weighting the particles, edge detection and background subtraction are used. A relatively small number of at least 100 and typically a few hundreds of particles are sufficient for tracking an articulated model of the human body with 29 pose parameters. For three-dimensional body tracking, Schmidt et al. (2006) employ the 'kernel particle filter', which approximates the probability density in state space by a superposition of Gaussian kernels. They use 150 particles to track a three-dimensional model of the upper human body defined by 14 pose parameters, relying on monocular colour images. The particles are weighted by the use of colour cues which are combined with ridge and edge cues. Due to the monocular approach, pose ambiguities may be encountered in this framework.

To alleviate the ambiguity of the three-dimensional pose estimation result of the human body, Pons-Moll et al. (2011) propose a method which combines image information with data acquired by inertial sensors attached to the body. The inertial sensors provide information about the orientation of the body parts to which they are attached, which are converted into three-dimensional poses based on an inverse kinematics approach. This technique allows one to reduce the number of pose parameters of the utilised human body model, corresponding to 31, to an effective number of 16 parameters when employing five inertial sensors. Three-dimensional pose hypotheses consistent with the inertial sensor data are compared by Pons-Moll et al. (2011) with the observed image features, especially silhouette information, using a particle filter framework, where the sensor noise is modelled by the von Mises–Fisher distribution. The experimental evaluation shows a high accuracy and robustness of the correspondingly obtained three-dimensional pose estimation results.

At this point it is illustrative to mention methods for three-dimensional pose estimation and tracking of the human hand, which also represents a complex articulated object with a large number of degrees of freedom. From the methodical point of view, methods for hand pose estimation tend to be fairly similar to many of the previously described full body pose estimation approaches. The extensive survey by Erol et al. (2007) provides an overview of hand pose estimation techniques. They divide methods for hand modelling into geometric techniques, usually involving a considerable number of pose parameters, and kinematic approaches that learn typical dynamical patterns of the hand motion, where they point out that in most systems a manual user-specific calibration of the kinematic hand model is required. Furthermore, they distinguish between two-dimensional and three-dimensional methods for hand pose estimation, where the three-dimensional techniques may rely on colour, edges, point correspondences, disparity information, or actively scanned range data. A further distinction is made by Erol et al. (2007) between tracking of single hypotheses, typically using a Kalman filter, and tracking of multiple hypotheses, e.g. involving an extended or unscented Kalman filter or a particle filter. For details refer to Erol et al. (2007) and references therein.

The work by Stößel (2007) is one of the few studies that examine the problem of three-dimensional pose estimation of articulated objects in the context of industrial quality inspection. The described system performs a three-dimensional pose estimation of objects which consist of several connected rigid parts, termed 'multi-part assemblies' by Stößel (2007), relying on a monocular image, where the corresponding articulated object models are characterised by up to 29 pose parameters. The method is based on the minimisation of the Hausdorff distance between edges observed in the image and those inferred from the model projected into the image, taking into account mutual occlusions of different parts of the articulated object. For minimisation of the error function, the 'extended kernel particle filter' approach is introduced by Stößel (2007) and employed as a stochastic optimisation technique.

2.2.2 Three-Dimensional Active Contours

This section is adopted from d'Angelo et al. (2004), who describe a parametric active contour framework for recovering the three-dimensional contours of rotationally symmetric objects such as tubes and cables. The proposed algorithm is a three-dimensional ziplock ribbon active contour algorithm based on multiple views.

2.2.2.1 Active Contours

In the snake approach by Kass et al. (1988), the basic snake is a parametric function **p** representing a contour curve or model:

$$\mathbf{p} = \mathbf{v}(s) \quad \text{for } s \in [0, l], \tag{2.3}$$

where **p** is a contour point for a certain value of the length parameter s. An energy function E_C is minimised over the contour $\mathbf{v}(s)$ according to

$$E_C = \int_0^l E_{\text{snake}}\big(\mathbf{v}(s)\big)\, ds. \tag{2.4}$$

The snake energy E_{snake} is separated into four terms:

$$E_{\text{snake}}\big(\mathbf{v}(s)\big) = \alpha E_{\text{cont}}\big(\mathbf{v}(s)\big) + \beta E_{\text{curv}}\big(\mathbf{v}(s)\big) + \gamma E_{\text{ext}}\big(\mathbf{v}(s)\big) + \delta E_{\text{con}}\big(\mathbf{v}(s)\big). \tag{2.5}$$

The 'internal energy' $E_{\text{int}} = \alpha E_{\text{cont}}(\mathbf{v}(s)) + \beta E_{\text{curv}}(\mathbf{v}(s))$ regularises the problem by favouring a continuous and smooth contour. The 'external energy' E_{ext} depends on the image at the curve point $\mathbf{v}(s)$ and thus links the contour with the image. Here we use the negative gradient magnitude of the image as the external energy, which then becomes $E_{\text{ext}} = -\|\nabla I(\mathbf{v}(s))\|$. E_{con} is used in the original snake approach by Kass et al. (1988) to introduce constraints, e.g. linking of points of the active contour to other contours or springs. Balloon snake techniques (Cohen, 1991) use E_{con} to 'inflate' the active contour in order to compensate for the shrinking induced by the internal energy E_{int}. The weight factors α, β, γ, and δ can be chosen according to the application.

The dependence of the snake model on its parameterisation may lead to numerical instabilities and self-intersection problems when it is applied to complex segmentation tasks. Such problems are avoided by implicit active contour models (Caselles et al., 1995, 1997). Furthermore, a contour is not necessarily a single curve, and modifications to extract ribbon structures consisting of parallel lines, like roads in aerial images (Fua and Leclerc, 1990) or blood vessels in angiographic images (Hinz et al., 2001), have been proposed in the literature. Snakes can also be used as a segmentation tool in an interactive manner, where a human operator provides a rough estimate of the initial contour and can move the snake such that local minima of the energy function are avoided (Kass et al., 1988).

For the method described in this section, the greedy active contours approach introduced by Williams and Shah (1992) is used as the basis for the three-dimensional snake framework. The contour is modelled by a polyline consisting of n points, and finite differences are used to approximate the energy terms E_{cont} and E_{curv} at each point $\mathbf{p}_s$, $s = 1, \ldots, n$ according to

$$\begin{aligned} E_{\mathrm{cont}}\big(\mathbf{v}(s)\big) &\approx \big| \|\mathbf{p}_s - \mathbf{p}_{s-1}\| - h \big| \\ E_{\mathrm{curc}}\big(\mathbf{v}(s)\big) &\approx \|\mathbf{p}_{s-1} - 2\mathbf{p}_s + \mathbf{p}_{s+1}\|, \end{aligned} \tag{2.6}$$

where h is the mean distance between the polyline points. The greedy minimisation algorithm is an iterative algorithm which selects the point of minimal energy inside a local neighbourhood. The greedy optimisation is applied separately to each point, from the first point at $s = 0$ to the last point at $s = l$. The energy $E_C(\mathbf{v}(s))$ is calculated for each candidate point $\mathbf{p}$ in a neighbourhood grid $H \in \mathbb{R}^d$, where d is the dimensionality of the curve. The point $\mathbf{p}_{\mathrm{min}}$ of minimum energy inside H is selected as the new curve point at $\mathbf{v}(s)$. This procedure is repeated until all points have reached a stable position or a maximum number of iterations has been reached.

Since the greedy optimisation algorithm does not necessarily find a global minimum, it needs a good initialisation to segment the correct contours. Especially in segmenting non-rigid objects, however, providing a suitable initialisation along the whole contour might not always be feasible. In such cases, the ziplock snake algorithm introduced by Neuenschwander et al. (1997) is used. Ziplock snakes are initialised by the end points of the contour segment to be extracted and the tangents of the curve at these points. The contour consists of active parts, which are subject to the full energy term E_{snake}, and inactive parts, which are influenced only by the internal energy E_{int}. The 'force boundaries' between the active and inactive parts of the snake start close to the end points of the contour segments and move towards each other in the course of the optimisation.

2.2.2.2 Three-Dimensional Multiple-View Active Contours

If volumetric images are available, the image energy E_{ext} of a three-dimensional contour can be calculated directly from the volumetric data. In industrial quality inspection applications, volumetric data are not available, but it is usually possible to obtain images acquired from multiple viewpoints. In that case E_{ext} can be calculated by projecting the contour into the image planes of the cameras (Fig. 2.3a). The camera system is calibrated with the method of Krüger et al. (2004) (cf. Sect. 1.4). An arbitrary number N of images ($i = 1, \ldots, N$) can be used for this projection. The intrinsic and extrinsic parameters of each camera are assumed to be known.

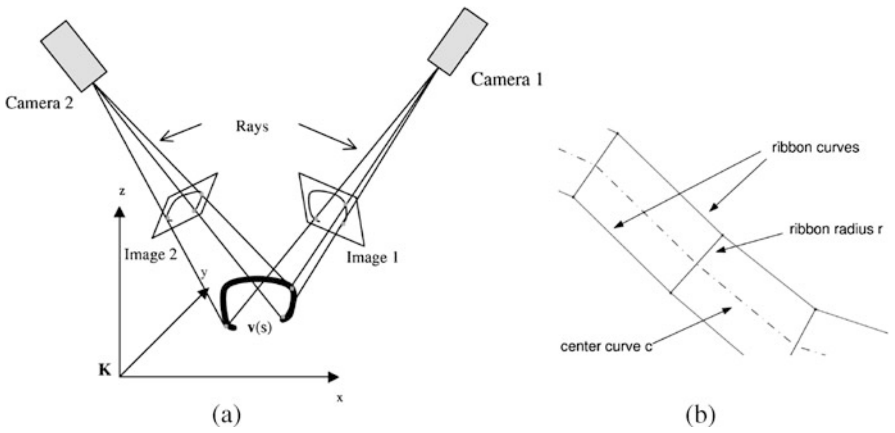


Fig. 2.3 (**a**) Projection of the three-dimensional curve model $\mathbf{v}(s)$ into multiple images. Two cameras are observing the scene. Calculation of the external energy requires the contour curves in the image planes of all cameras. (**b**) Sketch of a ribbon snake

Each scene point ${}^{W}\mathbf{p}$ is projected to the corresponding image point ${}^{S_i}\mathbf{p}$ defined in the sensor coordinate system. The energy term E_{ext} that connects the contour with the images is calculated based on the projection of the contour into each image plane according to $E^*_{\text{ext}} = \sum_{i=1}^{N} E_{\text{ext}}({}^{S_i}\mathbf{p})$. Hence, generating different active contours separately for each image or a depth map determined e.g. by stereo image analysis is not necessary. All image and constraint information is used by the energy minimisation step in the model parameter space. In our examples, we use the image gradient as the external energy term, i.e. $E_{\text{ext}}({}^{S_i}\mathbf{p}) = -\|\nabla I_i({}^{S_i}\mathbf{p})\|$. Occlusions of the object are not considered during modelling, but the algorithm copes with partial occlusions and holes in the contour if they do not appear excessively. The energy terms E_{int} and E_{con} are independent of occlusions that may occur in some views.

The described approach requires that either the points of the extracted three-dimensional snake correspond to the same scene points, respectively, or that the object displays certain symmetries. In our scenario of inspection of tubes and cables, the objects to be segmented are rotationally symmetric with respect to a centreline, such that their silhouette is similar from multiple viewpoints and can be described by a centre curve $(x(s), y(s), z(s))^T$ and a radius $r(s)$, leading to $\mathbf{v}(s) = (x, y, z, r)^T$ (Fig. 2.3b). In this case, the contour model is not a simple curve and cannot be projected into the images as described above. Instead, the centreline $(x, y, z)^T$ is projected into the images while the corresponding ribbon curves are calculated by projecting the radius r into the image planes.

If prior knowledge about the contour is available, it can be integrated into the snake optimisation algorithm to improve its convergence. In an industrial production environment, CAD models of the parts that are subject to quality inspection are available and can be used to provide prior knowledge to the active contour segmentation. If the shape of objects should be recovered, only constraints that are invariant to the possible shapes of the object should be used. For quality inspection of tubes, this includes elasticity, length, radius, and sometimes the mounting position, depending on the application. The model information is introduced into the optimisation process in two ways. The first is by adding additional energy terms, the second by using a constrained optimisation algorithm. Additional model-based energy terms that favour a certain shape of the contour can be added to E_{con}. For example, the approximate radius of a cable is known, but it may vary at some places due to labels or constructive changes.

As a first approach, we utilise a three-dimensional ribbon snake to detect the cable shape and position. Hence, $E_{\text{con}} = E_{\text{rib}} = [r(s) - r_{\text{model}}(s)]^2$ can be used as a 'spring energy' to favour contours with a radius r close to a model radius given by the function $r_{\text{model}}(s)$. However, adding constraint terms to the objective function may result in an ill-posed problem with poor convergence properties and adds more weight factors that need to be tuned for a given scenario (Fua and Brechbühler, 1996). The second approach is to enforce model constraints through optimiser constraints. In the greedy algorithm this is achieved by intersecting the parameter search region H and the region C permitted by the constraints to obtain the allowed search region $H_c = H \cup C$. This ensures that these constraints cannot be violated (they are also called hard constraints).

For some applications like glue line detection, the surface in which the contour is located is known, for example from CAD data. In other cases, the bounding box of the object can be given. This knowledge can be exploited by a constraint that restricts the optimisation to the corresponding surface or volume. Model information can also be used to create suitable initial contours—for example, tubes are often fixed with brackets to other parts. The pose of these brackets is usually given a priori when repeated quality inspection tasks are performed or can be determined by using pose estimation algorithms for rigid objects (cf. Sect. 2.1). These points can be used as starting and end points, i.e. boundary conditions, for three-dimensional ziplock ribbon snakes.

2.2.2.3 Experimental Results on Synthetic Image Data

As a first test, the described algorithm has been applied to synthetically generated image data, for which the ground truth is inherently available. For all examples, a three-dimensional ribbon snake was used. The weight factors of (2.5) were set to $\alpha = 1$, $\beta = 1$, $\gamma = 3$, and $\delta = 0$. Additionally, a hard constraint has been placed on the minimum and maximum ribbon width, which avoids solutions with negative or unrealistically large width. To estimate the reconstruction quality, a synthetically rendered scene was used as a test case, as this allows a direct comparison to the known ground truth. Figures 2.4a and b show the example scene and its three-dimensional reconstruction result. The start and end points of the object and their tangents were specified. In a real-world application these could either be taken from a CAD model, or estimated by pose estimation of the anchoring brackets. The ground truth used to produce the image is known and is compared to the segmented contour. The utilised error measure is the root-mean-square error (RMSE) of the

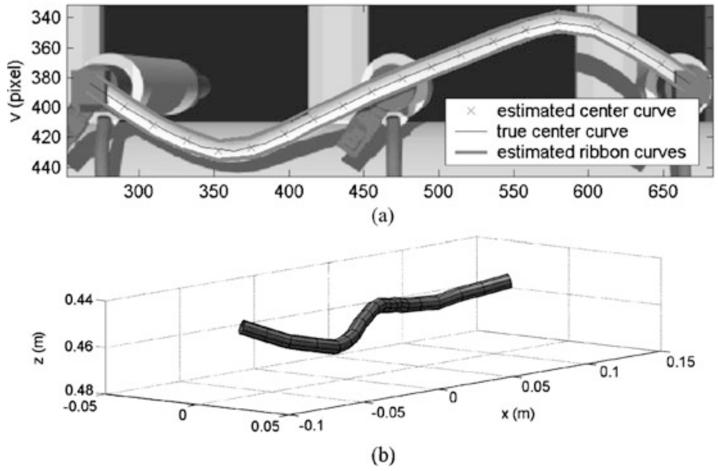


Fig. 2.4 (**a**) One of the three artificially rendered input images with the overlaid reprojections of the three-dimensional ribbon snake. A virtual trinocular camera with a resolution of 1024 × 768 pixels and a base distance of 100 mm has been used to generate the images. (**b**) Reconstructed tube, shown from a different viewpoint. The RMSE between ground truth and reconstruction is 1.5 mm

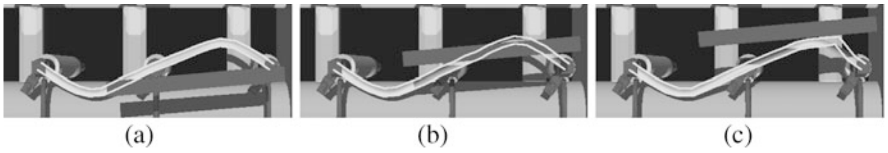


Fig. 2.5 Behaviour of the three-dimensional ziplock ribbon snake algorithm with respect to partial occlusion of the object. A virtual rod is moved across the rendered scene of Fig. 2.4

centre curve of the estimated ribbon snake with respect to the model centre curve. In the example shown in Fig. 2.4, the RMSE amounts to 1.5 mm, which roughly corresponds to 1 pixel disparity error. Subpixel greedy stepwidths and interpolated image energy calculation were used to obtain this result.

The behaviour of the algorithm in the presence of partial occlusions has been tested by moving a virtual rod over the scene, as shown in Fig. 2.5. In Fig. 2.5a, where only a small part of the object is occluded, the RMSE with respect to the ground truth amounts to 1.1 mm, while for stronger occlusions as in Figs. 2.5b and c the RMSE corresponds to 3.1 mm and 3.8 mm, respectively. All described test cases were run on a 1.7 GHz Pentium Mobile Processor. The computation time of the optimisation procedure amounts to between 1 and 23 seconds, depending on the complexity of the scene and the parameters chosen for the optimisation procedure.

The experimental results show that the proposed three-dimensional ziplock ribbon snake algorithm is able to perform a fairly accurate three-dimensional contour segmentation. The stability of the algorithm is mainly due to the usage of model-based constraints and initial contour curves based on model information. However, the proposed method is sensitive to occlusions and self-intersections if they appear excessively. It is limited to lines and tube-shaped objects, which is useful for a variety of applications, e.g. in the field of industrial quality inspection. As real-world applications of the three-dimensional ziplock ribbon snake method, the three-dimensional reconstruction of a cable and of a glue line on a car body part are addressed in Sect. 6.2.

2.2.3 Three-Dimensional Spatio-Temporal Curve Fitting

As an example of three-dimensional pose estimation of articulated objects, this section addresses the problem of markerless pose estimation and tracking of the motion of human body parts in front of a cluttered background. The multiocular contracting curve density (MOCCD) algorithm inspired by Hanek and Beetz (2004) and its spatio-temporal extension, the shape flow algorithm, are introduced by Hahn et al. (2007, 2008b, 2010a) to determine the three-dimensional pose of the hand–forearm limb.

Due to the limited resolution of the trinocular greyscale camera setup it is unfeasible in the system to model each finger of the hand, as is possible e.g. in the work by Stenger et al. (2001). On the other hand, a cylindrical model of the forearm as proposed by Schmidt et al. (2006) is too coarse due to the variability of human appearance, e.g. clothes. Hence, the methods described in this section are based on a three-dimensional hand–forearm model which represents the three-dimensional contour by an Akima spline (Akima, 1970) using control points defined by a parameter vector. The MOCCD algorithm is computationally too expensive to be used in a particle filter framework. Hence, it is integrated into a Kalman filter-based tracking framework which estimates more than one pose hypothesis at a single time step. The presentation in this section is adopted from Hahn et al. (2010a). Further details are provided by Hahn (2011).

2.2.3.1 Modelling the Hand–Forearm Limb

In the application scenario of safe human–robot interaction described in Sects. 7.3 and 7.4, a three-dimensional model of the human hand–forearm limb will be used which consists of a kinematic chain connecting the two rigid elements forearm and hand. The model consists of five truncated cones and one complete cone, as shown in Fig. 2.6. The cones are defined by nine parameters according to

$$\mathbf{T} = [p_{1x}, p_{1y}, p_{1z}, \alpha_1, \beta_1, \alpha_2, \beta_2, r_1, r_4]^T. \tag{2.7}$$

The three-dimensional point ${}^W\mathbf{p}_1 = [p_{1x}, p_{1y}, p_{1z}]$ defines the beginning of the forearm and is part of the pose parameter vector $\mathbf{T}$. The wrist (${}^W\mathbf{p}_2$) and fingertip (${}^W\mathbf{p}_3$) positions are computed according to

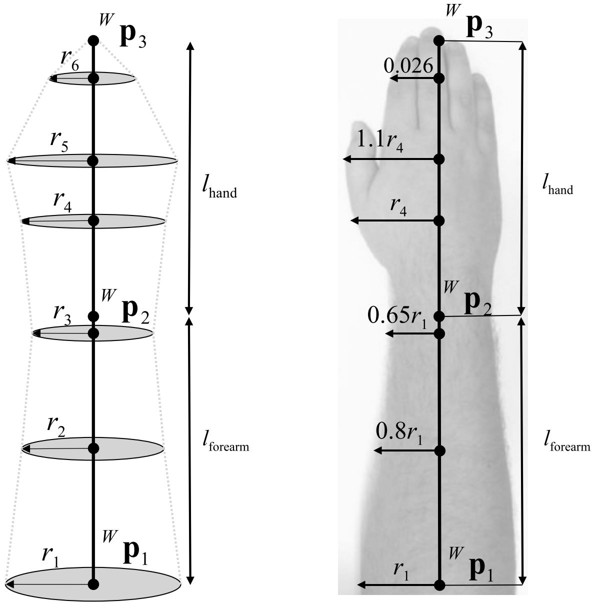


Fig. 2.6 Hand–forearm model. *Left*: Definition of the cones. *Right*: Dependencies of the radii derived from human anatomy

$$ {}^W\mathbf{p}_2 = {}^W\mathbf{p}_1 + R_Z(\beta_1) \cdot R_Y(\alpha_1) \cdot l_{\text{forearm}} \cdot [1,0,0]^T \tag{2.8}$$

$$ {}^W\mathbf{p}_3 = {}^W\mathbf{p}_2 + R_Z(\beta_2) \cdot R_Y(\alpha_2) \cdot l_{\text{hand}} \cdot [1,0,0]^T, \tag{2.9}$$

where l_{forearm} and l_{hand} are the predefined lengths of the human hand–forearm limb. The x and y axes are running in the horizontal and vertical directions parallel to the image plane, respectively, while the z axis denotes the depth. The matrix $R_Y(\alpha)$ represents a rotation around the y axis by the angle α, and $R_Z(\beta)$ a corresponding rotation around the z axis.

The lengths l_{hand} and l_{forearm} of the hand and forearm are set to uniform fixed values for all image sequences regarded in the experimental evaluation described in Sect. 7.4. Although human hands and forearms may actually have fairly different lengths, it is shown in Sect. 7.4 that the hand–forearm limb of all test persons is tracked successfully, and we found that differences between the modelled and the actual lengths of 100–200 mm are easily tolerated by the system. Such differences may even occur as short-term variations within a sequence, e.g. when the hand is grabbing, holding, and depositing a tool.

The shapes of the hand and the forearm relative to the maximal radii r_1 and r_4 were derived from human anatomy, as shown in Fig. 2.6, and are defined according to

$$\begin{aligned} r_2 &= 0.8 \cdot r_1 \\ r_3 &= 0.65 \cdot r_1 \\ r_5 &= 1.1 \cdot r_4 \\ r_6 &= 0.026\ \text{m} = \text{const.} \end{aligned} \tag{2.10}$$

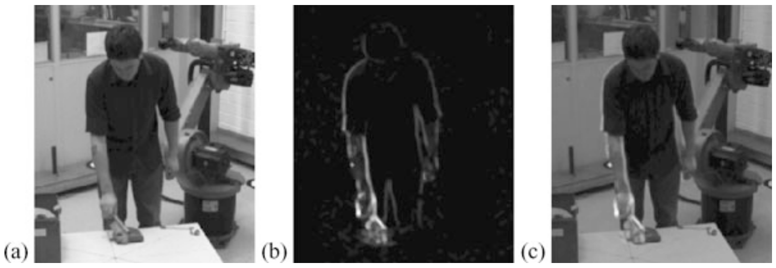


Fig. 2.7 (**a**) Original 8-bit image I_t. (**b**) Absolute difference image. (**c**) Input image I_t^* (scaled to 8 bit) for $\lambda = 1$ according to (2.11)

Hence, in contrast to their lengths, the maximal radii r_1 and r_4 of the hand and the forearm are part of the parameter vector $\mathbf{T}$ and can thus be adapted to the images during the optimisation process.

2.2.3.2 Principles and Extensions of the CCD Algorithm

The CCD algorithm introduced by Hanek (2001) and refined by Hanek and Beetz (2004) adapts a curve model to an image based on the probability distributions of the pixel grey values on the inner and the outer side of the curve. A computationally efficient real-time variant of the CCD algorithm is described by Panin et al. (2006).

A difference to the work of Hanek (2001) is that Hahn et al. (2010a) rely on an extended input image, since in the regarded application scenario the model-based image segmentation is challenging due to noise, a cluttered background, and the coarse object description. In order to obtain an accurate and robust model-based image segmentation, we take advantage of the constant camera position in our application. Our input image $I^*(t)$ is computed by

$$I^*(t) = I(t) + \lambda\Big|I(t) - I(t-1)\Big|, \qquad (2.11)$$

where $I(t)$ is the image at time step t and $|I(t) - I(t-1)|$ is the absolute difference image of the current and the previous image. The factor λ defines the influence of the absolute difference image. The influence of the absolute difference image increases the pixel values in areas where motion occurs, which allows a more robust segmentation, since the CCD algorithm adapts the curve model by separating the probability distributions of the pixel grey values on the inner and the outer side of the object curve. Another advantage is that if there is no motion, the input image $I^*(t)$ corresponds to the original image $I(t)$ and it is still possible to fit the model. We experimented with different values of λ in the range $[0.5, \ldots, 3]$ and found empirically that the dependence of the segmentation result does not critically depend on the value of λ. For a moving camera, the original image $I(t)$ would have to be used as the input image $I^*(t)$, corresponding to $\lambda = 0$. Figure 2.7 shows the original camera image, the absolute difference image, and the input image obtained for $\lambda = 1$.

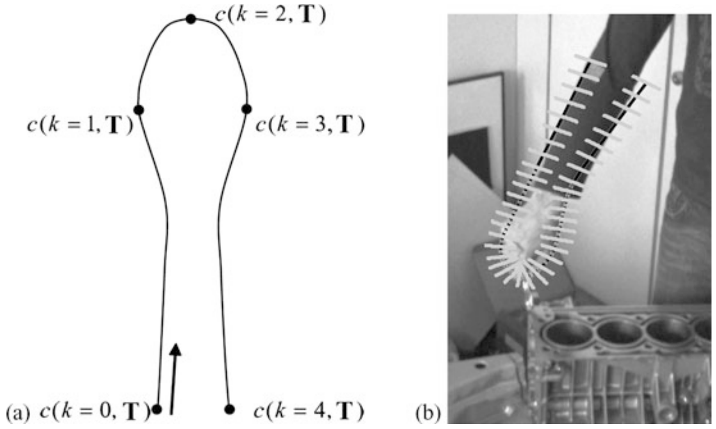


Fig. 2.8 (**a**) The two-dimensional model curve described by the curve function $c(k, \mathbf{T})$. The parameter k defines a point on the curve, and the vector $\mathbf{T}$ contains the curve parameters. (**b**) Curve function $c(k, \mathbf{T})$ with the k perpendiculars centred at the curve points

The CCD algorithm fits a parametric curve $c(k, \mathbf{T})$ to the image I^*. The parameter $k \in [0, K]$ of the parametric model curve $c(k, \mathbf{T})$ shown in Fig. 2.8a is incremented with the points defining the curve, and the vector $\mathbf{T}$ denotes the set of estimated curve parameters. The CCD algorithm relies on the image I^* and the initial probability distribution $p(\mathbf{T}) \approx p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ of the parameters $\mathbf{T}$ of the curve model, which is assumed to be Gaussian and is thus given by the mean parameter vector $\widehat{\mathbf{m}}_\mathbf{T}$ and the corresponding covariance matrix $\widehat{\Sigma}_\mathbf{T}$. As initial values, the a priori density parameters $(\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ are used. The CCD algorithm adapts the model to the image by repeatedly performing steps 1 and 2 described below until the reduction of $\mathbf{m}_\mathbf{T}$ and $\Sigma_\mathbf{T}$ becomes smaller than a predefined value or after a given maximum number of iteration cycles has been performed. This procedure can be summarised by the two steps.

Step 1: Learning Local Probability Distributions The local probability distributions $S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$, which are given by the pixel grey value means and standard deviations, are computed on both sides of the curve.

In this step pixels close to the curve (along the perpendiculars as shown in Fig. 2.8b) are probabilistically assigned to the inner and the outer side of the expected curve. Then the probability distributions $S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ of the pixel grey values on both sides of the curve are computed. For this porpose only pixels on $(K + 1)$ different lines segments are used. Each line segment k is a perpendicular on the curve point $\mathbf{C}_k = c(k, \mathbf{m}_\mathbf{T})$ of the expected curve. In contrast to the original real-time CCD algorithm (Hanek, 2001), the perpendiculars are not chosen to be equally spaced along the curve, because our applied curve model is not a closed curve. Thus there are more perpendiculars on the closed side (hand part) of our curve, to obtain a stronger dependence of the objective function on the shift along the forearm axis (cf. Fig. 2.8b). The length of a perpendicular k depends on the local uncertainty σ_k of the curve given by

$$\sigma_k^2 = \mathbf{n}_k^T \cdot \mathbf{J}_{\mathbf{C}_k} \cdot \Sigma_{\mathbf{T}} \cdot \mathbf{J}_{\mathbf{C}_k}^T \cdot \mathbf{n}_k \tag{2.12}$$

and changes in each iteration step. The variable $\mathbf{n}_k$ defines the curve normal and $\mathbf{J}_{\mathbf{C}_k}$ the Jacobian, i.e. the partial derivative of the curve with respect to the model parameters $\mathbf{T}$. The original real-time CCD algorithm according to Hanek (2001) computes for every perpendicular k a local uncertainty σ_k of the curve. In contrast to Hanek (2001), we apply a fixed local uncertainty for every perpendicular of the curve, thus avoiding a large number of nonlinear function evaluations. The local uncertainties σ_0 and $\sigma_{K/2}$ at the perpendiculars $k = 0$ and $k = K/2$ are computed using (2.12), and the fixed local uncertainty σ results from $\sigma = (\sigma_0 + \sigma_{K/2})/2$. The fixed local uncertainty σ is now used to compute the probabilistic assignment for every point $\mathbf{v}_{k,l}$ on the perpendiculars; l denotes the pixel index on perpendicular k. For every point $\mathbf{v}_{k,l}$ on perpendicular k the probability $a_{l,1}(d_l)$ that the point lies on side 1 (inner side) of the curve (probabilistic assignment) is computed by

$$a_{l,1}(d_l) = \frac{1}{2}\left[\operatorname{erf}\left(\frac{d_l}{\sqrt{2}\sigma}\right) + 1\right]. \tag{2.13}$$

In (2.13), $\operatorname{erf}(x)$ is the Gaussian error function and $d_l(\mathbf{v}_{k,l}) = \mathbf{n}_k^T(\mathbf{v}_{k,l} - \mathbf{C}_k)$ the signed distance of the pixel coordinate $\mathbf{v}_{k,l}$ from the curve point $\mathbf{C}_k$. The probabilistic assignment $a_{l,2}(d_l)$ of side 2 (outer side) is defined by $a_{l,2}(d_l) = 1 - a_{l,1}(d_l)$.

To compute the two-sided probability distributions of the pixel grey values we follow the suggestions of Hanek (2001) and apply as a weighting function the expression

$$w_{l,s} = w_A(a_{l,s}) \cdot w_B(d_l, \sigma) \cdot w_C(\sigma) \quad \text{with } s \in \{1, 2\}, \tag{2.14}$$

where

$$w_A(a_{l,s}) = \max\left(0, \left[\frac{a_{l,s} - \gamma_1}{1 - \gamma_1}\right]^{(2 \cdot E_A)}\right) \tag{2.15}$$

assesses the probabilistic assignment. The parameter $\gamma_1 \in [0, 1[$ describes the minimum probability $a_{l,s}$ that the pixel is used to compute the probability distributions. We use $\gamma_1 = 0.5$ and $E_A = 3$. The weight $w_B(d_l, \sigma)$ considers the signed distance of the pixel $\mathbf{v}_{k,l}$ to the curve and is given by

$$w_B(d_l, \sigma) = C \cdot \max\left(0, e^{(-d_l^2/(2 \cdot \widehat{\sigma}))} - e^{(-\gamma_4)}\right) \quad \text{with} \tag{2.16}$$

$$\widehat{\sigma} = \gamma_3 \cdot \sigma + \gamma_4. \tag{2.17}$$

Here C is a normalisation constant, where we set $C = 5$. For the other parameters we use $\gamma_2 = 4$, $\gamma_3 = 6$, and $\gamma_4 = 3$. The weighting function $w_C(\sigma)$ evaluates the local uncertainty σ according to

$$w_C(\sigma) = (\sigma + 1)^{-E_C}. \tag{2.18}$$

We recommend the choice $E_C = 4$. Since we consider a fixed local uncertainty σ, the probabilistic assignment and the weights depend only on the signed distance

d_l to the curve, where the sign indicates the side of the curve on which a pixel is located. Hence, the weighting function and the probabilistic assignment need to be computed only once and can be obtained by a distance-dependent look-up operation, thus avoiding a large number of nonlinear function evaluations without loss of accuracy. Based on the weights $w_{l,s}$ we obtain the statistical moments $m_{k,s,o}$ of order $o \in \{0, 1, 2\}$ for every perpendicular k and side $s \in \{1, 2\}$ of the curve, using

$$m_{k,s,0} = \sum_{l=1}^{L_k} w_{l,s} \tag{2.19}$$

$$m_{k,s,1} = \sum_{l=1}^{L_k} w_{l,s} \cdot I^*_{k,l} \tag{2.20}$$

$$m_{k,s,2} = \sum_{l=1}^{L_k} w_{l,s} \cdot I^*_{k,l} \cdot {I^*_{k,l}}^T \tag{2.21}$$

with $I^*_{k,l}$ as the pixel value of perpendicular k and pixel index l. The original CCD tracker according to Hanek (2004) uses a spatial and spatio-temporal smoothing of the statistical moments to exploit the spatial and spatio-temporal coherence of pixel values. This improves the robustness of the tracker, since the influence of strong edges and outliers is reduced. In contrast to the original work of Hanek (2004), we apply only a spatial smoothing of the statistical moments, since our input images I^* according to (2.11) are not temporally coherent due to the addition of the absolute difference image. We rely on a blurring with fixed filter coefficients by applying to the statistical moments a Gaussian convolution operator of size 1×5 with a standard deviation of 2 contour points. This leads to the spatially smoothed moments $\widetilde{m}_{k,s,o}$ of order $o \in \{0, 1, 2\}$ for every perpendicular k and side $s \in \{1, 2\}$. The local mean vectors $m_{k,s}(t)$ and covariance matrices $\Sigma_{k,s}(t)$ for every perpendicular k at time step t on both sides $s \in \{1, 2\}$ of the curve are computed by

$$m_{k,s}(t) = \frac{\widetilde{m}_{k,s,1}(t)}{\widetilde{m}_{k,s,0}(t)} \tag{2.22}$$

$$\Sigma_{k,s}(t) = \frac{\widetilde{m}_{k,s,2}(t)}{\widetilde{m}_{k,s,0}(t)} - m_{k,s}(t) \cdot m^T_{k,s}(t) + \kappa. \tag{2.23}$$

The parameter κ avoids singularities. We set it to $\kappa = 2.5$. The local probability distributions $S(\mathbf{m_T}, \Sigma_\mathbf{T})$ of the pixel grey values which are used in the second step to refine the estimate consist of $(K + 1)$ different local mean vectors $m_{k,s}(t)$ and covariance matrices $\Sigma_{k,s}(t)$. In the following we assume to be at time step t and denote the mean values by $m_{k,s}$ and the covariance matrices by $\Sigma_{k,s}$.

Step 2: Refinement of the Estimate (MAP Estimation) An update of the parameters $(\mathbf{m_T}, \Sigma_\mathbf{T})$ obtained in step 1, describing the probability distribution of the curve parameters, is performed based on a single Newton-Raphson step which increases the a-posteriori probability according to Eq. (2.24).

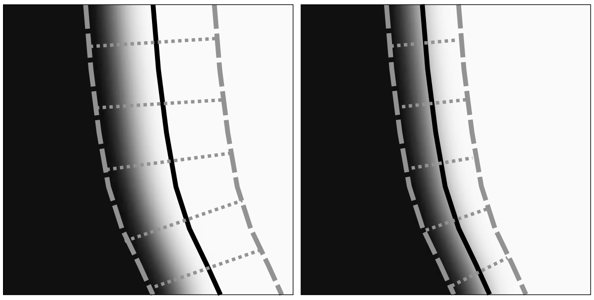


Fig. 2.9 The CCD algorithm determines the parameters of the curve (*black line*) by computing the probability distributions of the pixel grey values in a local neighbourhood of the curve (indicated by the *grey lines*) and by maximising the a posteriori probability of the curve model. Non-optimal (*left*) and optimal (*right*) configuration

The principle of the CCD algorithm is depicted in Fig. 2.9. The CCD algorithm estimates a model pose by computing the maximum of the a posteriori probability according to

$$p\big(\mathbf{T}|I^*\big) = p\big(I^*|\mathbf{T}\big) \cdot p(\mathbf{T}). \tag{2.24}$$

Since the probability distributions of $p(\mathbf{T})$ and $p(I^*|\mathbf{T})$ are unknown, they are approximated. The Gaussian a priori density $p(\mathbf{T}) \approx p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ of the model parameters $\mathbf{T}$ is defined by the mean $\widehat{\mathbf{m}}_\mathbf{T}$ and the covariance matrix $\widehat{\Sigma}_\mathbf{T}$. The likelihood function $p(I^*|\mathbf{T})$ is approximated by the Gaussian density function $p(I^*|S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T}))$ and describes how well the pixel values along the perpendicular fit estimated probability distributions $S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ close to the curve. The observation model, i.e. the likelihood function, is computed by

$$p\big(I^*|S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})\big) = \prod_{k,l} \frac{1}{\sqrt{2\pi} \cdot \Sigma_{k,l}} \cdot e^{-\frac{h_k^2}{2\Sigma_{k,l}^2}} \quad \text{with} \tag{2.25}$$

$$h_k = p\big(I^*_{k,l}|m_{k,l}, \Sigma_{k,l}\big) - m_{k,l} \tag{2.26}$$

where l defines the pixel index on the perpendicular k and $p(I^*_{k,l}|m_{k,l}, \Sigma_{k,l})$ a Gaussian probability density with mean $m_{k,l}$ and covariance $\Sigma_{k,l}$ according to

$$m_{k,l} = a_{l,1} \cdot m_{k,1} + a_{l,2} \cdot m_{k,2} \tag{2.27}$$

$$\Sigma_{k,l} = a_{l,1} \cdot \Sigma_{k,1} + a_{l,2} \cdot \Sigma_{k,2}. \tag{2.28}$$

These values depend on the probabilistic assignment $a_{l,s}$ of the pixel with index l and the two-sided probability distributions $S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ of the pixel grey values with the mean vector $m_{k,s}$ and covariance matrix $\Sigma_{k,s}$ for perpendicular k.

To increase the numerical stability of the optimisation procedure, the log-likelihood

$$X = -2\ln\big[p\big(I^*|S(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})\big) \cdot p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})\big], \tag{2.29}$$

rather than the a posteriori probability according to Eq. (2.24) is computed, and the curve density parameters $(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ are refined by a single step of a Newton–Raphson optimisation.

2.2.3.3 The Multiocular Extension of the CCD Algorithm

The multiocular extension of the CCD algorithm relies on the projection of the boundary of a three-dimensional contour model into each image. The intrinsic and extrinsic parameters of the camera model (Bouguet, 2007) are obtained by multiocular camera calibration (Krüger et al., 2004). An arbitrary number of images N_c can be used for this projection. The input values of the MOCCD algorithm are N_c images and the Gaussian a priori distribution $p(\mathbf{T}) \approx p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ of the model parameters $\mathbf{T}$, which define the three-dimensional object model. To achieve a more robust segmentation, the input image $I^*_{c,t}$ of the MOCCD algorithm is computed using (2.11) and the original camera images $I_{c,t}$ with $c \in \{1, \ldots, N_\mathrm{c}\}$ at the time steps t and $(t-1)$. Before the first iteration, the MOCCD algorithm is initialised by setting the mean vector and covariance matrix $(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ to be optimised to the given a priori density parameters $(\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$. The MOCCD algorithm then consists of three steps.

Step 1: Extraction and Projection of the Three-Dimensional Model The intrinsic and extrinsic camera parameters are used to project the extracted outline of the three-dimensional model to each camera image I^*_c. The MOCCD algorithm extends the CCD algorithm to multiple calibrated cameras by projecting the boundary of a three-dimensional model into each camera image I^*_c. Therefore, the MOCCD algorithm requires the extraction and projection of the outline of the used three-dimensional model.

A three-dimensional hand–forearm model (cf. Sect. 2.2.3.1) is fitted to the images of a trinocular camera. The outline of our three-dimensional model in each camera coordinate system is extracted by computing a vector from the origin of each camera coordinate system to the point in the wrist, e.g. ${}^{C_1}\mathbf{p}_2$ for camera 1. This vector and the direction vector $\overline{\mathbf{p}_1\mathbf{p}_2}$ of the forearm span a plane. The normal vector of this plane is intersected with the three-dimensional model to yield the three-dimensional outline observed from the camera viewpoint. The extracted three-dimensional contour model for the given camera, which consists of 13 points, is projected into the pixel coordinate system of the camera. The corresponding two-dimensional contour model is computed by an Akima interpolation (Akima, 1970) along the curve with the 13 projected points as control points. Figure 2.10 depicts the extraction and projection of the three-dimensional contour model for camera 1.

Step 2: Learning Local Probability Distributions from all N_c Images For all N_c camera images I^*_c compute the local probability distributions $S_c(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ on both sides of the curve. This step is similar to step 1 of the CCD algorithm; the only difference is that the probability distributions $S_c(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ on both sides of the curve are learned for all N_c camera images I^*_c.

Step 3: Refinement of the Estimate (MAP Estimation) The curve density parameters $(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ are refined towards the maximum of (2.30) by performing a

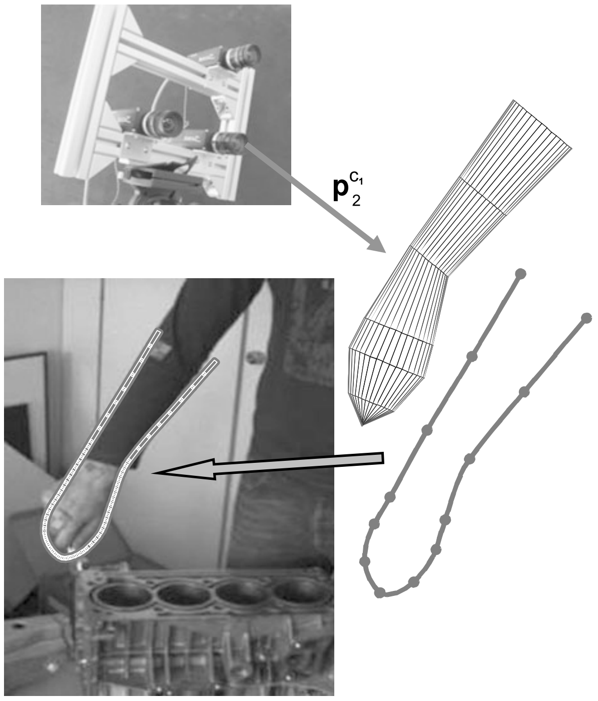


Fig. 2.10 Extraction and projection of the three-dimensional contour model for camera 1

single step of a Newton–Raphson optimisation procedure. The learned probability distributions $S_c(\mathbf{m_T}, \Sigma_\mathbf{T})$ are used to maximise the joint probability

$$p\big(\mathbf{T}|i_1^*, \ldots, I_{N_c}^*\big) = \left[\prod_{c=1}^{N_c} p\big(I_c^*|S_c(\mathbf{m_T}, \Sigma_\mathbf{T})\big)\right] \cdot p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T}) \tag{2.30}$$

with $S_c(\mathbf{m_T}, \Sigma_\mathbf{T})$ representing the probability distributions close to the projected curve in image I_c^*. As in the original CCD framework, we optimise the log-likelihood of (2.30) with a Newton–Raphson optimisation step. The MOCCD algorithm can be illustrated as follows: The Gaussian probability densities $p(I^*|S_c(\mathbf{m_T}, \Sigma_\mathbf{T}))$ are sensitive with respect to step-like structures along the curve perpendiculars in each image I_c^*, and the three-dimensional model is adapted to the N_c camera images by an implicit triangulation.

2.2.3.4 The Shape Flow Algorithm

The shape flow (SF) algorithm is introduced by Hahn et al. (2008b, 2010a) as a top-down approach for spatio-temporal pose estimation. In contrast to bottom-up motion estimation approaches, like the motion analysis module described in Sect. 2.3.3, the SF algorithm is generally able to estimate the three-dimensional pose parameters $\mathbf{T}$ and the temporal pose derivative $\dot{\mathbf{T}}$ with a spatio-temporal model and the images of

N_t time steps only. With the SF algorithm it is possible to estimate the velocity along the depth axis, which is not possible with the motion analysis module described in Sect. 2.3.3, as the regarded scene flow vectors provide no information about the velocity of the three-dimensional points along the depth axis.

The SF algorithm is a temporal extension of the MOCCD algorithm and fits a three-dimensional spatio-temporal contour model to multiocular images (N_c cameras) at N_t time steps. The input values of the SF algorithm are N_c images at N_t time steps and the Gaussian a priori distribution $p(\mathbf{T}) \approx p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ of the model parameters $\mathbf{T}$, which define the spatio-temporal three-dimensional object model. To achieve a more robust segmentation, the input image $I^*_{c,t}$ of the SF algorithm is computed using (2.11). Here, the SF algorithm is used to estimate the temporal pose derivative $\dot{\mathbf{T}}$ only. Before the first iteration, the SF algorithm is initialised by setting the mean vector and covariance matrix $(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ to be optimised to a priori density parameters $(\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T})$ which may e.g. be provided by the motion analysis module described in Sect. 2.3.3. The SF algorithm consists of three steps.

Step 1: Projection of the Spatio-Temporal Three-Dimensional Contour Model Based on the motion model, e.g. with constant velocity, the three-dimensional contour model for all time steps N_t is computed and camera parameters are used to project the extracted outline of the three-dimensional model to each camera image $I^*_{c,t}$. The SF algorithm extends the MOCCD algorithm to the temporal dimension by using a spatio-temporal three-dimensional model, defined by the parameter vector and an underlying motion model, e.g. constant velocity. In this step the observed boundary of the spatio-temporal three-dimensional model is computed for all time steps N_t and projected into each camera image $I^*_{c,t}$. The outline of our three-dimensional model is extracted and projected as described in step 1 of the MOCCD algorithm.

Now we describe the applied motion model and the computation of the spatio-temporal three-dimensional model. Note that the optimised parameter vector in the SF algorithm consists only of the temporal derivative $\dot{\mathbf{T}}(t)$. We assume that the forearm radius r_1 and the hand radius r_4 (cf. Sect. 2.2.3.1) are constant over short periods of time corresponding to a few frames; therefore these radii are not part of the temporal pose derivative $\dot{\mathbf{T}}$. However, the radii r_1 and r_4 are only excluded from the estimation of the temporal pose derivative $\dot{\mathbf{T}}$, not from the estimation of the pose $\mathbf{T}$ itself. Hence, the algorithm is able to adapt itself to short-term changes of the radii within a sequence; only their temporal derivatives are not estimated directly.

The three-dimensional pose $\mathbf{T}(t)$ at time step t is computed at time step t with the MOCCD algorithm or with the fusion module described in Sect. 7.4.3. The temporal pose derivative $\dot{\mathbf{T}}(t)$ at time step t is determined with the SF algorithm and the image triples at the time steps $(t + \Delta t)$ and $(t - \Delta t)$. The spatio-temporal three-dimensional curve model at the time steps $(t \pm \Delta t)$ is computed according to $\mathbf{T}(t) \pm \dot{\mathbf{T}}(t) \cdot \Delta t$, thus assuming constant motion. Figure 2.11 depicts the projected contour model for camera 1 (the one which defines the coordinate system) and the spatio-temporal three-dimensional pose estimation result (a model-based dense

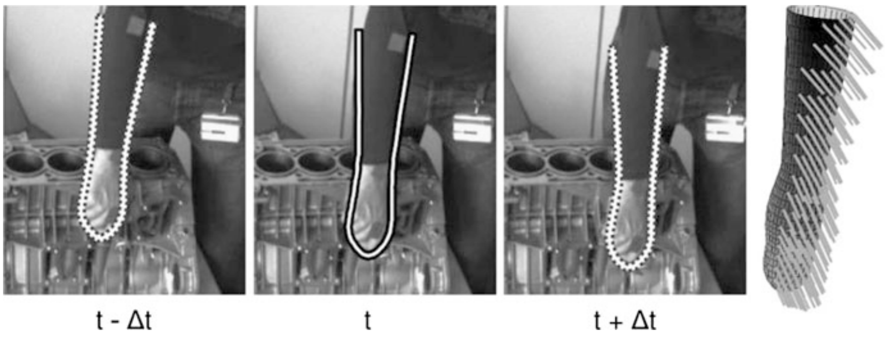


Fig. 2.11 Projected three-dimensional contour model for camera 1 at time steps $(t - \Delta t)$, t, and $(t + \Delta t)$. The *lines* indicate the amount and direction of motion for each point on the model surface

scene flow). At time step t the projected contour of the estimated three-dimensional pose is shown as a solid curve, while the images at the time steps $(t \pm \Delta t)$ depict the projected contour of the pose derivative.

Step 2: Learn Local Probability Distributions from all N_c Images For all N_c camera images I_c^* and for all N_t time steps, compute the local probability distributions $S_{c,t}(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ on both sides of the projected three-dimensional contour model, defined by camera c and time step t. This step is similar to step 1 of the CCD algorithm, the only difference is that the probability distributions $S_{c,t}(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ on both sides of the curve are learned for all N_c cameras images $I_{c,t}^*$ at all N_t time steps.

Step 3: Refine the Estimate (MAP Estimation) The curve density parameters $(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ are refined towards the maximum of (2.31) by performing a single step of a Newton–Raphson optimisation procedure. The learned spatio-temporal probability distributions $S_{c,t}(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ are used to maximise the joint probability

$$p\big(\mathbf{T}|\{I_{c,t}\}\big) = \left[\prod_c \prod_t p\big(I_{c,t}|S_{c,t}(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})\big)\right] \cdot p(\mathbf{T}|\widehat{\mathbf{m}}_\mathbf{T}, \widehat{\Sigma}_\mathbf{T}) \qquad (2.31)$$

with $S_{c,t}(\mathbf{m}_\mathbf{T}, \Sigma_\mathbf{T})$ representing the probability distributions of the pixel grey values close to the projected curve in image $I_{c,t}$ (camera c, time step t). The underlying assumption is that the images are independent random variables. As in the original CCD framework, a numerically favourable form of (2.31) is obtained by computing the log-likelihood. The SF algorithm can be interpreted as a top-down, model-based spatio-temporal three-dimensional pose estimation approach which determines a three-dimensional pose and its temporal derivative. The advantage is that temporal and spatial constraints are handled directly in the optimisation procedure.

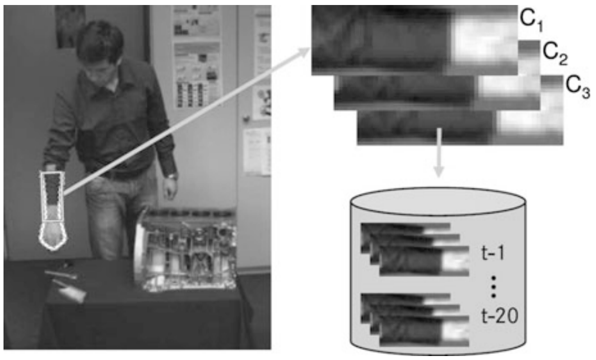


Fig. 2.12 Construction of the reference templates. The forearm rectangle is cropped from all camera images and transformed and interpolated to a predefined size and orientation

2.2.3.5 Verification and Recovery of the Pose Estimation Results

Pose Verification To verify and rate a set of calculated pose parameters, three quality measures are used. The first quality criterion is the point distance of the segmented point cloud to the model. Accordingly, a thin hull around the object model is used to determine the points inside the object hull. The weighting value σ_p is the quotient of object points inside the hull and all segmented points.

The second quality criterion is the orientation similarity σ_o, which is computed for a three-dimensional pose by extracting the contour of the three-dimensional model according to Sect. 2.2.3.1 and projecting it into the images acquired by the camera system. For calculating the quality of a contour, the algorithm 'walks along' small perpendiculars in each projected curve point of the contour. For each pixel on the perpendicular the image gradient is calculated. The image gradient orientation, which is between 0° and 180°, is then compared with the orientation of the model contour in the current curve point itself. While the contour gradient describes a reference orientation, the grey image gradient is the compared actual orientation. By simply counting the curve points that do not differ too much and normalising by all projected curve points in all camera images, one obtains an intuitive quality measure in the range $[0 \ldots 1]$.

The third quality criterion is the appearance similarity σ_c for a three-dimensional pose at time step t, which is based on a comparison of the current object appearance with the previous object appearances. With the three-dimensional pose at time step t and the known camera parameters we crop the image of the forearm in all camera images and transform the cropped images to a predefined size and orientation. The final pose estimation result is used to add the current reference template at time step t to a database (cf. Fig. 2.12). This database describes the appearance of the tracked object in the last 20 time steps. The appearance similarity σ_c of a new pose estimation result is computed based on a normalised cross-correlation of the current object appearance at time step t with the previous object appearance at time step $(t-1)$. Since we have three cameras, there are three correlation results for a pose comparison, where we choose the worst of the three as the final result. The assumptions made for this similarity measure are that

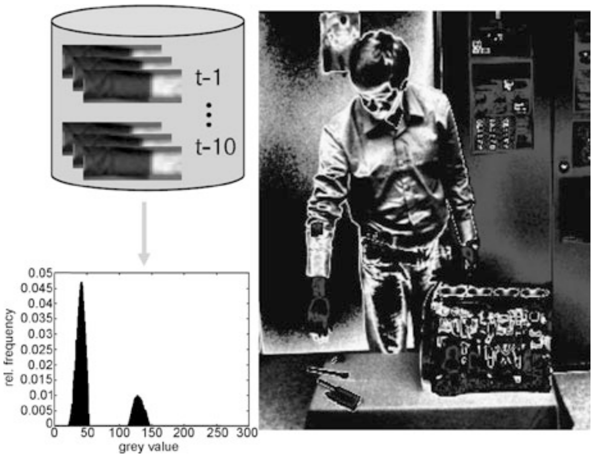


Fig. 2.13 Probability distribution image for camera 1, obtained based on the histogram of the relative frequency of greyscale pixel values

(i) a good three-dimensional pose estimation is available at the first time step and (ii) that the object appearance does not change considerably over a small period of time.

To verify an estimated three-dimensional pose, five criteria are used: (i) the temporal variation of the orientation similarity, (ii) the current orientation similarity, (iii) the temporal variation of the appearance similarity for the three-dimensional pose $\mathbf{T}(t)$ compared with the pose $\mathbf{T}(t - \Delta t)$, (iv) the appearance similarity for $\mathbf{T}(t)$ compared with $\mathbf{T}(t - \Delta t)$, and (v) the appearance similarity for pose $\mathbf{T}(t)$ compared with pose $\mathbf{T}(t - 2\Delta t)$.

If the five criteria of a three-dimensional pose pass a set of thresholds, the pose is defined as valid and the reference template is added to the database. Otherwise, the tracking can either be stopped with a warning that the object is lost, or a reference template-based pose recovery stage can be applied.

Pose Recovery on Loss of Object If the object is deemed to be lost by the verification stage, the pose recovery stage starts at the last valid three-dimensional pose vector $\mathbf{T}$ and computes two possible pose hypotheses: (i) based on a probability distribution image and (ii) by applying a correlation with the reference templates.

The first pose hypothesis is obtained using the last 10 reference templates of the forearm to construct a histogram which describes the forearm appearance. The histogram describes the relative frequency of greyscale pixel values (Bradski, 1998) and is computed only for camera 1. Similar to Bradski (1998), the histogram is used to compute a probability distribution image for camera 1, as shown in Fig. 2.13. The last valid three-dimensional pose vector $\mathbf{T}$ is used to project a rectangle circumscribing the forearm into the probability distribution image. A two-dimensional greedy optimisation in the image plane is applied to find the centre and rotation of the rectangle with the maximum sum of pixel probabilities. Using the best matching rectangle, we construct a three-dimensional pose under the assumption that the depth does not change. The resulting three-dimensional pose is computed with the MOCCD algorithm, where the pose of the best matching rectangle is used for initialisation.

The second hypothesis is computed using a two-dimensional correlation-based pose refinement algorithm. The last valid three-dimensional pose vector **T** is used to project a rectangle circumscribing the forearm into the image of camera 1. This rectangle is the starting position of a two-dimensional greedy optimisation which searches the centre and rotation of the rectangle with the highest normalised cross-correlation compared to the reference template of the last valid time step. A three-dimensional pose is inferred from the best matching rectangle, which is used as an initialisation for the MOCCD algorithm.

Relying on the criteria of the verification module, the better of the two hypotheses is determined. If the better hypothesis passes the verification module, the tracking is continued using the corresponding three-dimensional pose.

2.3 Point Cloud Segmentation Approaches

For the point-based three-dimensional pose estimation methods outlined in Sect. 2.1, explicit knowledge about correspondences between three-dimensional model points and two-dimensional image points is required. The problem of estimating the pose of an object is then equivalent to that of determining exterior camera orientation (cf. Sect. 1.4). In contrast, appearance-based pose estimation approaches like those described in Sects. 2.1 and 2.2.1.2 do not rely on explicit correspondences but, instead, minimise the difference between the expected appearance of the object according to the estimated pose and the true object appearance.

In many scenarios, a three-dimensional description of the scene is given as a point cloud obtained e.g. by stereo image analysis (cf. Sect. 1.2) or with active sensors such as laser scanning devices. Initially, this point cloud contains no information about objects in the scene. In such cases, an important task is the segmentation of the point cloud into objects, either without using a priori information or based on (weak or strong) model assumptions about the objects found in the scene. A scene segmentation without a priori knowledge can be achieved by clustering methods (Press et al., 2007; Marsland, 2009), while an important approach to model-based segmentation of point clouds is the iterative closest point (ICP) algorithm (Besl and McKay, 1992; Zhang, 1992). Similar methods have been developed in the domain of photogrammetry, e.g. to extract human-made objects such as buildings from topographic maps or terrestrial laser scanner data (Rottensteiner et al., 2005, 2006). We regard in detail a method introduced by Schmidt et al. (2007) for the detection and tracking of objects in a three-dimensional point cloud with motion attributes generated with the spacetime stereo approach described in Sect. 1.5.2.5. This method involves a clustering step relying on the spatial distribution and motion behaviour of the scene points, a subsequent model-fitting stage, and a kernel particle filter for tracking the detected objects.

2.3.1 General Overview

Segmentation of a point cloud corresponds to the subdivision of the cloud into subparts that likely represent different objects. Without a priori knowledge about the sizes and shapes of the objects encountered in the scene, i.e. in the absence of model information, an appropriate approach to scene segmentation is clustering, which corresponds to the subdivision of a set of points into parts consisting of mutually similar points (Press et al., 2007; Marsland, 2009). These points may e.g. be three-dimensional points extracted by a scene reconstruction method or three-dimensional points enriched by velocity information such as optical flow or scene flow (cf. Sect. 1.5.2.5), but may also be data points defined in an abstract feature space, where the position in the feature space is associated with certain properties. According to Marsland (2009), the similarity between the points is measured by an appropriately defined metric, where the most common choice is the Euclidean distance between the points. It is important to note that the subdivision of the data set into clusters is performed without information about the class to which each of the individual data points belongs, such that clustering is also termed 'unsupervised learning' (Press et al., 2007).

2.3.1.1 The k-Means Clustering Algorithm

An important hierarchical clustering algorithm is the k-means algorithm introduced by MacQueen (1967). According to Marsland (2009), the number k of clusters has to be given, and k random points are selected as cluster centres. In the first step, each data point is assigned to the cluster centre with the smallest distance. In the second step, the new cluster centres are computed as the averages of the data points according to their assignments in the first step. These two steps are repeated until the assignment of the data points to the established cluster centres (and thus also the cluster centres themselves) remains constant. A new, unknown data point is assigned to the cluster with the smallest distance. Notably, as a consequence of the random initialisation, the k-means algorithm does not necessarily yield the same result for different initialisations.

2.3.1.2 Agglomerative Clustering

According to Press et al. (2007), agglomerative clustering starts with each data point representing a cluster and combines small clusters into larger ones until a dissimilarity criterion between the clusters is met. Press et al. (2007) describe several approaches to determine the distances d_{pk} between a newly formed cluster k and a reference cluster p in comparison to the distances d_{pi} and d_{pj} of the previous cluster pair (i, j) to the same reference cluster as a criterion to combine the cluster pair into a new single cluster. Here, the 'weighted pair group method using arithmetic averages' (WPGMA) implies $d_{pk} = d_{kp} = (d_{pi} + d_{pj})/2$, the ‘unweighted pair group method using arithmetic averages’ (UPGMA) yields $d_{pk} = d_{kp} = (n_i d_{pi} + n_j d_{pj})/(n_i + n_j)$ with n_i and n_j as the number of data points in the clusters i and j, respectively, the ‘single linkage clustering method’ relies on the minimum distance $d_{pk} = d_{kp} = \min(d_{pi}, d_{pj})$, and the ‘complete linkage clustering method’ on the maximum distance $d_{pk} = d_{kp} = \max(d_{pi}, d_{pj})$. For segmentation of a three-dimensional point cloud, the object detection and tracking method by Schmidt et al. (2007) described in Sect. 2.3.4 relies on the complete linkage clustering method.

2.3.1.3 Mean-Shift Clustering

The mean-shift clustering approach is a non-parametric clustering technique which estimates local density maxima of the data points. It is described in detail by Comaniciu and Meer (2002), who mention as precursors to their approach the works by Fukunaga and Hostetler (1975) and Cheng (1995). According to Comaniciu and Meer (2002), the mean-shift algorithm relies on a kernel-based estimation of the density gradient of the distribution of the N data points $\{\mathbf{x}_i\}_{i=1,\ldots,N}$. The function $k(x)$ is termed the ‘profile’ of the kernel function. Several kernel functions are discussed by Cheng (1995). Comaniciu and Meer (2002) state that two important kernel profiles are the Epanechnikov kernel with $k_E(x) = 1 - x$ for $0 \leq x \leq 1$ and $k_E(x) = 0$ otherwise, and the normal kernel $k_N = \exp(-\frac{1}{2}x)$ with $x \geq 0$. The kernel function is assumed to be radially symmetric with $K(\mathbf{x}) = c_{k,d}\; k(\|\mathbf{x}\|^2)$, where d is the feature space dimension and $c_{k,d} > 0$ normalises the kernel function such that its integral over the complete feature space amounts to unity. To estimate the density gradient, Comaniciu and Meer (2002) define $g(x) = -dk(x)/dx$ as the profile of the kernel $G(\mathbf{x}) = c_{g,d}\; g(\|\mathbf{x}\|^2)$. The mean-shift algorithm is then initialised with the position $\mathbf{y}_0$ in the feature space, and the kernel width h needs to be given. The position in the feature space is then updated iteratively for $j > 0$ according to

$$\mathbf{y}_{j+1} = \frac{\sum_{i=1}^{N} \mathbf{x}_i\, g(\|\frac{\mathbf{y}_j - \mathbf{x}_i}{h}\|^2)}{\sum_{i=1}^{N} g(\|\frac{\mathbf{y}_j - \mathbf{x}_i}{h}\|^2)}. \tag{2.32}$$

Comaniciu and Meer (2002) show that $\mathbf{y}_j$ converges towards a local maximum of the point density as long as the profile $k(x)$ of the kernel $K(\mathbf{x})$ is convex and decreases monotonically with increasing values of x.

2.3.1.4 Graph Cut and Spectral Clustering

A more recent approach to the clustering problem is the normalised graph cuts method introduced by Shi and Malik (1997, 1998, 2000), which is closely related to the method of spectral clustering (Fowlkes et al., 2004). In the context of image segmentation, Shi and Malik (2000) model the image in the form of a weighted undirected graph, where each node is assigned to a pixel while an edge of the graph is defined by a pair of pixels. The similarity between the pixel pair determines the weight of an edge. Removing the edges connecting the segments out of which the image consists leads to a segmentation of the image into disjoint sets. The optimal subdivision is the configuration with the minimal removed edge weights, which is termed the 'cut'. Hence, such techniques are also known as 'graph cut' methods. They have become fairly popular in the domain of image segmentation.

Fowlkes et al. (2004) assume that a number of N image pixels is given by their previously determined properties such as position, grey value, texture, or depth. The basis of spectral clustering is the $N \times N$ symmetric similarity matrix W of the graph. The set of nodes is denoted by V. When a bipartition of V is given by A and B with $A \cup B = V$ and $A \cap B = 0$, the 'cut' is defined by

$$\mathrm{cut}(A, B) = \sum_{i \in A} \sum_{j \in B} W_{ij}. \tag{2.33}$$

The degree d_i of node i corresponds to $d_i = \sum_j W_{ij}$, and the volumes of A and B are given by $\mathrm{vol}(A) = \sum_{i \in A} d_i$ and $\mathrm{vol}(A) = \sum_{i \in B} d_i$. The 'normalised cut' is then defined according to

$$\mathrm{ncut}(A, B) = \frac{2\,\mathrm{cut}(A, B)}{\mathrm{vol}(A) \| \mathrm{vol}(B)} \tag{2.34}$$

with the harmonic mean $a \| b = 2ab/(a + b)$. At this point it is desired to determine the graph bipartition which minimises the value of $\mathrm{ncut}(A, B)$. A solution of this complex optimisation problem based on an eigenvalue decomposition of the Laplacian matrix $\mathcal{L} = D^{-1/2}(D - W)D^{-1/2}$ with the diagonal matrix D containing the values d_i as diagonal elements is proposed by Shi and Malik (2000). Fowlkes et al. (2004) extend the clustering framework towards more than two groups. The computational effort of the graph cut approach may become considerable, as it increases quadratically with the number of pixels. Hence, Fowlkes et al. (2004) suggest that one utilise the Nyström method to obtain an approximate solution of the eigensystem.

2.3.1.5 The ICP Algorithm

A method to register a point cloud to a geometric object model is the iterative closest point (ICP) algorithm introduced by Besl and McKay (1992). Given an initial estimate of the object pose, the pose parameters are updated by minimising the mean squared distance between the measured data and the model, and a new assignment of measured points to model points is performed. This procedure is applied in an iterative manner. As a result, the algorithm yields the three-dimensional object pose. Besl and McKay (1992) apply this method to point clouds, parametric and non-parametric curves, and to parametric, non-parametric, and triangulated (tessellated) surfaces. In the ICP algorithm proposed by Zhang (1992), the scene points as well as the object model are represented as connected ('chained') points. During each iteration step the pose parameters are updated while at the same time some of the scene points are assigned to the model, based on the distance to the model and the similarity between the directions between neighbouring points for the data and the model curve, such that outliers in the data are not taken into account for matching. The ICP algorithm yields those measured points that can be assigned to the model, i.e. a scene segmentation, along with the pose parameters.

2.3.1.6 Photogrammetric Approaches

In the domain of photogrammetry, an important application which requires the segmentation of three-dimensional point clouds is the three-dimensional reconstruction of buildings, mainly from airborne or terrestrial laser scanner data. Rottensteiner (2006) states that polyhedral models are appropriate for modelling buildings on spatial scales at which topographic mapping is performed, where symmetry constraints or a priori knowledge, e.g. about the orthogonality of walls or the orientation of parts of the building, can be exploited. Rottensteiner (2006) defines one group of common approaches to building extraction as bottom-up methods, where the geometric primitives of the building are constructed from the measured three-dimensional point cloud and are then used to build up a model of the building. This approach is used by Rottensteiner et al. (2005) to determine the roof planes of buildings in three-dimensional point clouds acquired by a laser scanner based on statistical considerations. The second basic class of methods discussed by Rottensteiner (2006) is top-down reconstruction, where given geometric primitives are adapted to parts of the three-dimensional point cloud. A priori knowledge can either be incorporated by 'hard constraints', which are always exactly satisfied by the inferred model, or by 'soft constraints', which allow a residual deviation of the adapted model from the three-dimensional point cloud data.

2.3.2 Mean-Shift Tracking of Human Body Parts

This section describes the mean-shift-based approach of Hahn et al. (2010b) to tracking human body parts, which extracts all moving objects or object parts from the scene based on a simple ellipsoid model. The presentation in this section is adopted from that work. Further details are provided by Hahn (2011).

2.3.2.1 Clustering and Object Detection

Object detection and three-dimensional tracking are based on a scene flow field. We use a combination of dense optical flow and sparse correlation-based stereo. The dense optical flow algorithm described by Wedel et al. (2008a, 2008b) is used to

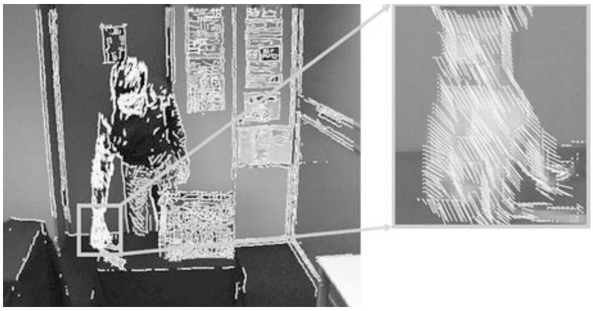


Fig. 2.14 Reprojected sparse scene flow field

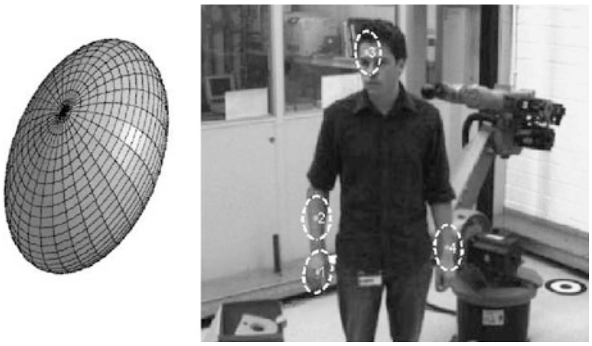


Fig. 2.15 *Left*: Ellipsoid model used for tracking arbitrary objects or object parts in the three-dimensional point cloud. *Right*: Reprojection of four tracked objects

determine the object motion in the image sequence. The optical flow field is combined with the three-dimensional points from the blockmatching stereo algorithm by Franke and Joos (2000) to obtain the scene flow field (cf. Fig. 2.14). The velocity component parallel to the depth axis is not computed.

At each time step, a graph-based clustering stage extracts all moving objects from the scene flow field, essentially separating moving objects from the (stationary or differently moving) background. The computed clusters are approximated as ellipsoids, and the three-dimensional ellipsoid pose $\mathbf{T} = (c_x, c_y, c_z, \beta)^T$ is determined (cf. Fig. 2.15). Only the centre $\mathbf{c} = (c_x, c_y, c_z)^T$ of the ellipsoid and the rotation angle β around the depth axis are part of the pose vector $\mathbf{T}$, since we assume that the approximated objects are parallel to the image plane. The three-dimensional pose update of all tracked objects is based on a two-stage three-dimensional extension of the mean-shift algorithm according to Cheng (1995), Comaniciu et al. (2000). If a tracked object is not moving for more than five time steps it is deleted.

2.3.2.2 Target Model

The target model $\widehat{\mathbf{q}}_{(id)}$ with the object index id is computed based on the first three-dimensional ellipsoid pose $\mathbf{T}^{(id)}$ and is updated at every time step. It consists of a one-dimensional histogram of greyscale values. To compute the histogram $\widehat{\mathbf{q}}_{(id)}$ we place a grid on the surface of the ellipsoid, the resolution of which is equal

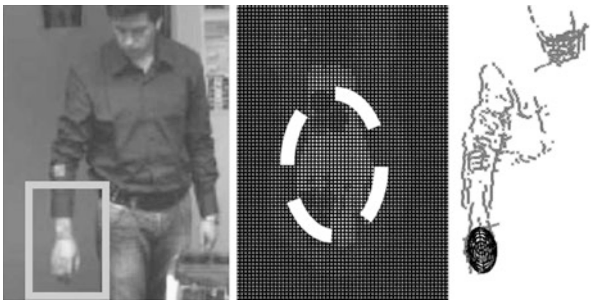


Fig. 2.16 Two-stage mean-shift procedure. Projected search region (*left*), three-dimensional probability grid (*middle*), final result (*right*)

to the pixel resolution at the current depth. Every three-dimensional point on the surface grid of the ellipsoid is projected into the images of all three cameras. The histogram bin of a three-dimensional point $\mathbf{p}$ is obtained with the look-up function $iBin(\mathbf{p}) = \frac{1}{3}(I_{uv}^{C_1} + I_{uv}^{C_2} + I_{uv}^{C_3})$, where $I_{uv}^{C_c}$ is the greyscale value in the image from camera c at the projected position (u, v) of the three-dimensional point $\mathbf{p}$. Using all three-dimensional points on the surface grid of the ellipsoid, a one-dimensional histogram of the object appearance is computed. A normalisation of the histogram yields the relative frequency of each greyscale value on the ellipsoid surface, which is interpreted as a probability.

2.3.2.3 Image-Based Mean-Shift

In the first stage, the mean-shift procedure is applied to a search region, a three-dimensional plane parallel to the image plane centred at the last object position. Similar to the method of Bradski (1998) in two dimensions, the target model $\widehat{\mathbf{q}}_{(id)}$ is used as a look-up table to compute the probability value for all three-dimensional points in the search region. The look-up function *iBin* is used to obtain a probability value for each three-dimensional point on the grid. Figure 2.16 (left) depicts the projected search region in the image of one camera and Fig. 2.16 (middle) shows the inferred three-dimensional probability grid. The three-dimensional centre point $\widetilde{\mathbf{c}}$ is estimated with the mean-shift procedure using a geometric ellipse model. The ellipse orientation β is computed similarly as in Bradski (1998). This mean-shift stage allows only for an update of the lateral pose of the tracked ellipsoid, since the probability grid is parallel to the image plane. No information from the scene flow field is used; thus a pose update of the ellipsoid is computed even if there is no new three-dimensional information available.

2.3.2.4 Point Cloud-Based Mean-Shift

In this stage all moving three-dimensional points of the scene flow field are used to update the three-dimensional pose of the tracked ellipsoid. At the first iteration $j = 1$ of the mean-shift procedure, the ellipsoid centre $\mathbf{c}_{j=1}$ is initialised with the estimated three-dimensional centre point $\widetilde{\mathbf{c}}$ of the image-based mean-shift stage. For all subsequent iterations, the ellipsoid model is moved to the new position

$$\mathbf{c}_{j+1} = \frac{\sum_{n=1}^{N} \mathbf{s}_n \cdot g(\mathbf{s}_n, \mathbf{c}_j) \cdot \widehat{\mathbf{q}}_{(id)}(iBin(\mathbf{s}_n))}{\sum_{n=1}^{N} g(\mathbf{s}_n, \mathbf{c}_j) \cdot \widehat{\mathbf{q}}_{(id)}(iBin(\mathbf{s}_n))}, \tag{2.35}$$

where $\mathbf{c}_j$ is the previous centre position. In the mean-shift procedure a truncated Gaussian kernel $g(\mathbf{s}_n, \mathbf{c}_j)$ is used according to Cheng (1995), for which the weight decreases with increasing distance from the ellipsoid centre $\mathbf{c}_j$. The mean-shift-based three-dimensional tracking approach incorporates an appearance weighting $\widehat{\mathbf{q}}_{(id)}(iBin(\mathbf{s}_n))$ obtained by looking up the appearance probability of the moving three-dimensional point $\mathbf{s}_n$ in the target model $\widehat{\mathbf{q}}_{(id)}$, such that a three-dimensional point with an appearance similar to the target appearance is assigned a higher weight. Figure 2.16 (right) depicts all moving three-dimensional points and the final result of the two-stage mean-shift procedure for three-dimensional tracking.

2.3.3 Segmentation and Spatio-Temporal Pose Estimation

The problem of three-dimensional scene segmentation along with the detection and pose estimation of articulated objects has been addressed primarily in the context of human motion capture (cf. Sect. 2.2.1.2 for an overview). A technique for model-based three-dimensional human body tracking based on the ICP algorithm is presented by Knoop et al. (2005). Normal optical flow is used by Duric et al. (2002) for positional short-term prediction in an image-based system for the detection and tracking of human body parts. The motion of a camera through a scene is estimated by Gonçalves and Araújo (2002) based on a combined analysis of stereo correspondences and optical flow.

This section describes a vision system for model-based three-dimensional detection and spatio-temporal pose estimation of objects in cluttered scenes proposed by Barrois and Wöhler (2008). The presentation is adopted from that work. Further details are provided by Barrois (2010). As low-level features, this approach requires a cloud of three-dimensional points attributed with information about their motion and the direction of the local intensity gradient. These features are extracted by spacetime stereo based on local image intensity modelling according to Sect. 1.5.2.5. After applying a graph-based clustering approach to obtain an initial separation between the background and the object, a three-dimensional model is adapted to the point cloud based on an ICP-like optimisation technique, yielding the translational, rotational, and internal degrees of freedom of the object. An extended constraint line approach is introduced which allows one to estimate the temporal derivatives of the translational and rotational pose parameters directly from the low-level motion information provided by the spacetime stereo data.

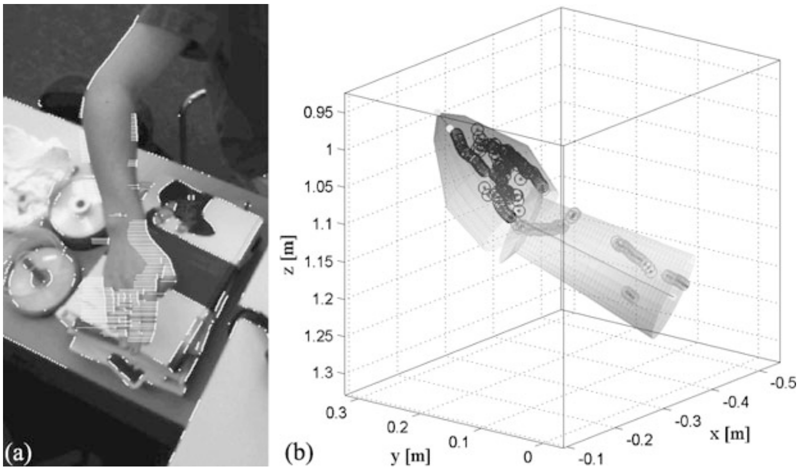


Fig. 2.17 (**a**) Image from a test sequence. Interest pixels for which stereo correspondences are established are shown as *bright dots*. Epipolar velocities are indicated as *white lines*. (**b**) Three-dimensional model adapted to the point cloud

2.3.3.1 Scene Clustering and Model-Based Pose Estimation

An initial segmentation of the attributed three-dimensional point cloud extracted with the spacetime stereo technique is obtained by means of a graph-based unsupervised clustering technique (Bock, 1974) in a four-dimensional space spanned by the spatial coordinates and the epipolar velocity of the three-dimensional points. This clustering stage generates a scene-dependent number of clusters, essentially separating the moving object from the (stationary or differently moving) background. For the first image of a sequence, the approximate position and orientation of the object are estimated based on a principal component analysis of the corresponding cluster points and used as initial values for the model adaptation procedure. For the subsequent images, the initial pose parameters are inferred for the current time step from the previous spatio-temporal pose estimation result, as described later.

We follow the ICP approach according to Zhang (1992) in order to fit a three-dimensional model of the hand–forearm limb (which does not necessarily represent the object at high accuracy) to the three-dimensional points determined to belong to the moving foreground object by the preceding clustering stage. We utilise the hand–forearm model introduced in Sect. 2.2.1.2 (Hahn et al., 2007, 2010a), consisting of a kinematic chain connecting the two rigid elements forearm and hand. The model consists of five truncated cones and one complete cone (cf. Fig. 2.17b). The cone radii corresponding to the hand and the upper end of the forearm are both set to 60 mm, and the lengths of the forearm and the hand are fixed to 220 mm and 180 mm, respectively. The other radii are inferred from human anatomy, as described in Sect. 2.2.1.2. For each of the two rotationally symmetric model parts, the five-dimensional vector **T** of translational and rotational pose parameters is determined. The relative orientation between forearm and hand is described by two an-

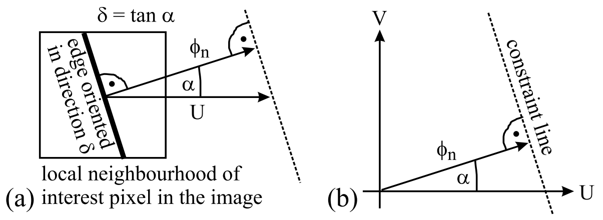

δ = tan α
edge oriented in direction δ
ϕn
α
U
local neighbourhood of interest pixel in the image
(a)
V
constraint line
ϕn
α
U
(b)

Fig. 2.18 (**a**) Relation between edge direction δ and normal velocity ϕ_n. (**b**) Definition of the constraint line in UV space according to Schunck (1989), representing the configurations (U, V) which are consistent with the observed normal velocity ϕ_n

gles, which are included in the model as internal degrees of freedom. In the course of the adaptation process, three-dimensional points not previously determined to belong to the object may be added to it while others may be rejected, resulting in a robust behaviour with respect to errors of the preceding clustering stage. The optimisation procedure is implemented as an M-estimator (Rey, 1983). It is straightforward to utilise the result of this three-dimensional pose estimation procedure as an initialisation for the appearance-based MOCCD technique described in Sect. 2.2.1.2.

2.3.3.2 Estimation of the Temporal Pose Derivatives

Both motion components of a scene point parallel to the image plane can only be recovered from the corresponding local pixel neighbourhood if the intensity pattern around the pixel is corner-like. Edge-like intensity patterns only allow the determination of one velocity component, such as the component parallel to the epipolar lines computed by the spacetime stereo algorithm (cf. Sect. 1.5.2.5). This ambiguity is a consequence of the well-known aperture problem (Horn, 1986). Restricting the stereo and motion analysis to corner-like image features (Franke et al., 2005) may result in fairly sparse depth maps. If edge-like image features are evaluated, as is the case in all image sequences regarded in this study, projecting the determined velocity component onto a line orthogonal to the local edge direction yields the normal velocity ϕ_n, as depicted in Fig. 2.18a. The angle α between the direction of the horizontal epipolar lines and the direction of the normal velocity is given by $\delta = \tan\alpha$ with δ as defined by (1.121) in Sect. 1.5.2.5.

In the following, the translational velocity components of the object parallel to the x, y, and z axes are denoted by U_{obj}, V_{obj}, and W_{obj}, respectively, and expressed in metres per second. A two-dimensional space is spanned by the horizontal and vertical velocity components U and V measured in the scene and expressed in metres per second. This space is termed UV space. Given the observed normal velocity ϕ_n, all consistent configurations (U, V) are represented by the corresponding constraint line in UV space as defined by Schunck (1989) (cf. Fig. 2.18b). Fermüller and Aloimonos (1994) extend the concept of constraint lines towards the analysis of 'image displacement fields', i.e. optical flow and disparity, that arise from the motion of a camera through a static scene or the motion of a rigid object relative to the camera ('rigid motion'). Of special importance is the normal flow, i.e. the component of the optical flow perpendicular to the edge in the image at which it is measured. It is shown by Fermüller and Aloimonos (1994) that an image displacement field displays a structure which does not depend on the scene itself, where vectors with specific values of their norm and direction are located on curves of certain shapes in the image plane. Specifically, sets of identical optical flow (also normal flow) or disparity vectors correspond to conic curves, termed 'iso-motion contours' by Fermüller and Aloimonos (1994). The properties of these curves only depend on the parameters of the three-dimensional rigid motion.

In the spatio-temporal pose estimation scenario regarded in this section, for an object performing a purely translational motion parallel to the image plane all constraint lines belonging to pixels on the object intersect in a single point in UV space. Both components of the translational motion are thus uniquely recovered. For objects with a rotational motion component in the image plane, a case which is not addressed by Schunck (1989), the intersection points between constraint lines are distributed across an extended region in UV space. This situation is illustrated in Fig. 2.19a for an ellipse rotating counterclockwise. The constraint lines belonging to the indicated contour points are shown in Fig. 2.19b. In this example, the U coordinates of the constraint line intersection points are a measure for the mean horizontal velocities of the corresponding pairs of image points. The V coordinates have no such physical meaning. The distribution of intersection points is elongated in vertical direction due to the fact that a vertical edge detector is used for interest pixel extraction and because only image points with associated values of $|\delta| < \delta_{\mathrm{max}}$ with δ_{max} typically chosen between 1 and 2 (cf. Sect. 7.3) are selected by the space-time stereo approach. Hence, constraint lines running at small angles to the U axis do not exist.

Figure 2.19c shows a distribution of constraint line intersection points obtained from the scene shown in Fig. 2.17a, typical of a rotationally symmetric and elongated object like the forearm partial model used in this study. The points in UV space are weighted according to the spatial density of the corresponding three-dimensional points along the longitudinal object axis. The mean $(U_{\mathrm{obj}}, V_{\mathrm{obj}})$ of the intersection point distribution, corresponding to the translational motion component of the object, has already been subtracted from the intersection points in Fig. 2.19c. The translational motion component W_{obj} parallel to the z axis is given by the median of the (fairly noisy) values of $\partial z/\partial t$ for all three-dimensional points assigned to the object or object part.

In the example regarded in Fig. 2.19c, scene points near the wrist are moving faster in the image plane than scene points near the elbow. The resulting intersection points are strongly concentrated near the points (U_1, V_1) and (U_2, V_2) depicted in Fig. 2.19c, which represent the motion of the scene points near the elbow and near the wrist. In this scenario, two circular markers attached to the upper and the lower end of the forearm, respectively, would yield two narrow clusters of intersection points in UV space at (U_1, V_1) and (U_2, V_2). Regarding scene points at arbitrary positions on the forearm instead of well-localised markers yields a distribution which is largely symmetric with respect to the line connecting the points

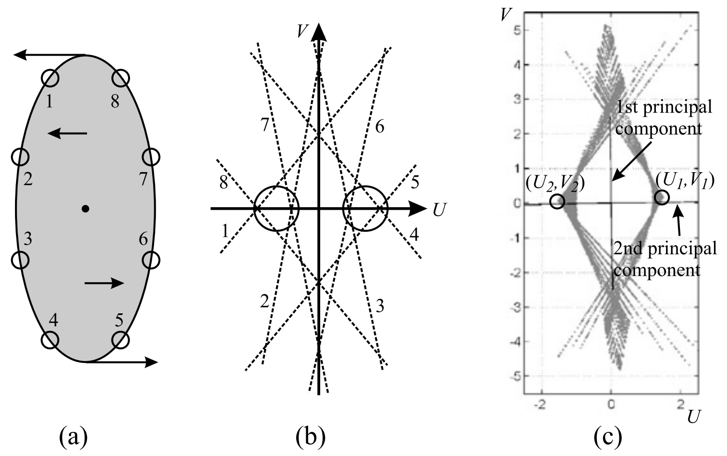


Fig. 2.19 (**a**) Rotating ellipse with *reference points* marked on its boundary. (**b**) Constraint lines resulting from the rotation of the ellipse. (**c**) Typical distribution of constraint *line intersections* in UV space for a real-world image from the first test sequence (cf. Fig. 2.17). The mean of the distribution has been subtracted from all points; the principal components are drawn as *solid black lines*

(U_1, V_1) and (U_2, V_2). The information about the rotational motion of the object is thus contained in the range Δv covered by the projections of the intersection points on the principal component of the distribution which is oriented perpendicular to the longitudinal axis of the object (the second principal component in Fig. 2.19c). The value of Δv then corresponds to the velocity dispersion across the object caused by rotational motion in the image plane. In our system, we robustly estimate Δv based on the 10 % and 90 % quantiles of the distribution of the projection values. The angular velocity ω_p of the object rotation parallel to the image plane is then obtained by $\omega_p = \Delta v/\Delta l$ with Δl as the length interval parallel to the longitudinal object axis covered by the assigned three-dimensional points.

The rotation orthogonal to the image plane is determined based on the values of $\partial z/\partial t$ determined in Sect. 1.5.2.5 for the extracted three-dimensional points. For each model part, the projections $p^{(i)}$ of the assigned three-dimensional points on the longitudinal object axis are computed, and a regression line is fitted to the $(p^{(i)}, \partial z/\partial t^{(i)})$ data points. The slope of the regression line directly yields the velocity dispersion Δw in the z direction and thus the angular velocity ω_o of the object rotation orthogonal to the image plane. Due to the rotational symmetry of the object models regarded in this study, the rotational motion of the object is already fully determined by the two components ω_p and ω_o of the angular velocity.

The technique described in this section allows us to extend the determination of the vector $\mathbf{T}$ of pose parameters by a direct estimation of the temporal pose derivative $\dot{\mathbf{T}}$ without the need for an object tracking stage.

2.3.4 Object Detection and Tracking in Point Clouds

This section describes the approach to the detection and tracking of objects in three-dimensional point clouds suggested by Schmidt et al. (2007). The presentation is adopted from that work. The method relies on a motion-attributed point cloud obtained with the spacetime stereo approach described in Sect. 1.5.2.5. In a subsequent step, motion-attributed clusters are formed which are then used for generating and tracking object hypotheses.

2.3.4.1 Motion-Attributed Point Cloud

A three-dimensional representation of the scene is generated with the correlation-based stereo vision algorithm by Franke and Joos (2000) and with the spacetime stereo algorithm described by Schmidt et al. (2007) (cf. Sect. 1.5.2.5). Both stereo techniques generate three-dimensional points based on edges in the image, especially object boundaries. Due to the local approach they are independent of the object appearance. While correlation-based stereo has the advantage of higher spatial accuracy and is capable of generating more point correspondences, spacetime stereo provides a velocity value for each stereo point. However, it generates a smaller number of points and is spatially less accurate, since not all edges are necessarily well described by the model defined in (1.118). Taking into account these properties of the algorithms, the results are merged into a single motion-attributed three-dimensional point cloud. For each extracted three-dimensional point c_k an average velocity $\bar{v}(c_k)$ is calculated, using all spacetime points s_j, $j \in (1, \ldots, J)$ in an ellipsoid neighbourhood defined by $\delta_S(s_j, c_k) < 1$ around c_k. To take into account the spatial uncertainty in depth direction of the spacetime data, $\delta_S(s_j, c_k)$ defines a Mahalanobis distance whose correlation matrix Σ contains an entry $\Sigma_z \neq 1$ for the depth coordinate which can be derived from the recorded data, leading to

$$\bar{v}(c_k) = \frac{\rho}{J} \sum_{j=1}^{J} v(s_j) \quad \forall s_j : \delta_S(s_j, c_k) < 1. \tag{2.36}$$

The factor ρ denotes the relative scaling of the velocities with respect to the spatial coordinates. It is adapted empirically depending on the speed of the observed objects. This results in a four-dimensional point cloud, where each three-dimensional point is attributed with an additional one-dimensional velocity component parallel to the epipolar lines; see Fig. 2.20d.

A reference image of the observed scene is used to reduce the amount of data to be processed by masking out three-dimensional points that emerge from static parts of the scene, as shown in Figs. 2.20a and b. Furthermore, only points within a given interval above the ground plane are used, as we intend to localise objects and humans and thus always assume a maximum height for objects above the ground.

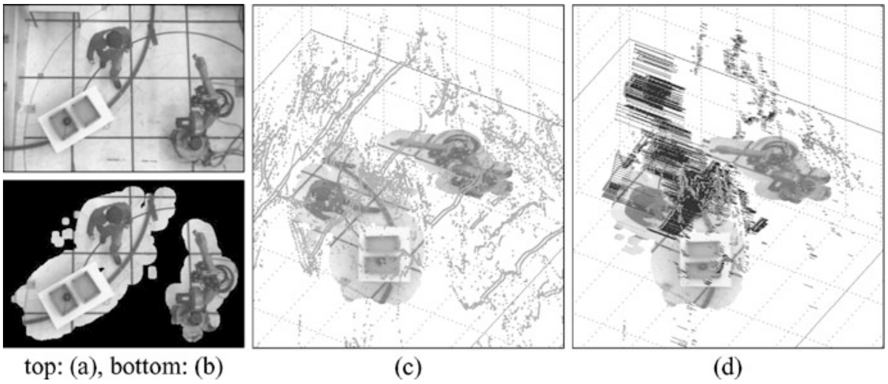


Fig. 2.20 (**a**) Original image (*left camera*). (**b**) Background subtracted image. (**c**) Full point cloud obtained with the correlation-based stereo vision technique. (**d**) Reduced motion-attributed point cloud

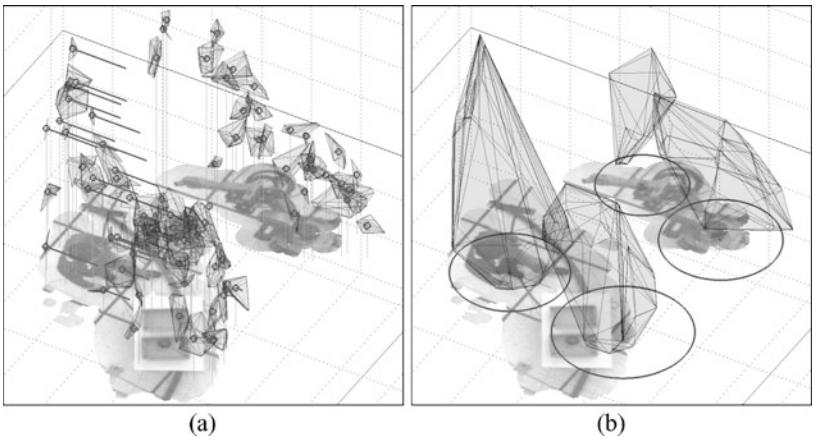


Fig. 2.21 (**a**) Over-segmentation and cluster velocities. (**b**) Objects with convex hull

2.3.4.2 Over-Segmentation for Motion-Attributed Clusters

To simplify the scene representation, we apply a hierarchical clustering algorithm, recognising small contiguous regions in the cloud, based on features like spatial proximity or homogeneity of the velocities. This procedure deliberately over-segments the scene, generating motion-attributed clusters. By incorporating velocity information for clustering, we expect an improvement in segmentation at these early stages of the algorithm, without the need for strong models to ensure separation of neighbouring objects. For clustering, we apply the complete linkage algorithm to describe the distance between two clusters. The resulting hierarchical tree is partitioned by selecting a clustering threshold and addressing each subtree as an individual cluster (cf. Fig. 2.21a). The criterion for selecting the threshold is the increase in distance between two adjacent nodes in the tree, for which a maximum allowed value is determined empirically. For each resulting cluster l, the weight $w(l)$ is set according to the number of points P belonging to l as $w(l) = \sqrt{P}$. The square root is used to constrain the weight for clusters consisting of many points. For each cluster the mean velocity of all points belonging to it is determined.

2.3.4.3 Generation and Tracking of Object Hypotheses

From here on, persons and objects can be represented as a collection of clusters of similar velocity within an upright cylinder of variable radius. An object hypothesis $R(a)$ is represented by a four-dimensional parameter vector $\mathbf{a} = (x, y, v, r)^T$, with x and y being the centre position of the cylinder on the ground plane, v denoting the velocity of the object, and r the radius. This weak model is suitable for persons and most encountered objects.

To extract the correct object positions, we utilise a combination of parameter optimisation and tracking. We first generate a number of initial hypotheses, optimise the location in parameter space, and then utilise the tracking algorithm to select hypotheses which form consistent trajectories. Initial object hypotheses are created at each time step by partitioning the observed scene with cylinders and by including the tracking results from the previous frame, respectively. Multidimensional unconstrained nonlinear minimisation (Nelder and Mead, 1965), also known as the simplex algorithm, is applied to refine the position and size of the cylinders in the scene, so that as many neighbouring clusters with similar velocity values as possible can be grouped together to form compact objects, as shown in Fig. 2.21b. An error function $f(\mathbf{a})$ used for optimisation denotes the quality of the grouping process for a given hypothesis. Each hypothesis is weighted based on the relative position, relative velocity, and weight of all clusters l within the cylinder $R(\mathbf{a})$ using Gaussian kernels according to

$$f(\mathbf{a}) = f_r(\mathbf{a}) \sum_{l \in R(\mathbf{a})} w(l) f_d(l, \mathbf{a}) f_v(l, \mathbf{a}) \tag{2.37}$$

with $f_r(\mathbf{a}) = \exp(-\frac{r(\mathbf{a})^2}{2H_{r,\min}^2}) - \exp(-\frac{r(\mathbf{a})^2}{2H_{r,\max}^2})$ keeping the radius in a realistic range, $f_d(l) = \exp(-\frac{[s(l)-s(\mathbf{a})]^2}{2H_d^2})$ reducing the importance of clusters far away from the cylinder centre, and $f_v(l, \mathbf{a}) = \exp(-\frac{[v(l)-v(\mathbf{a})]^2}{2H_v^2})$ masking out clusters having differing velocities. The functions $r(\mathbf{a})$, $s(\mathbf{a})$, and $v(\mathbf{a})$ extract the radius, the two-dimensional position on the ground plane, and the velocity of the hypothesis $\mathbf{a}$, respectively. The kernel widths H are determined empirically. Figure 2.22 shows the error function from (2.37), parameterised for opposing velocities. Local minima are centred on top of the objects of interest.

After optimisation, hypotheses with identical parameterisation are merged and those without any clusters within $R(\mathbf{a})$ are removed. The remaining hypotheses are tracked over time using a particle filter, keeping only object hypotheses forming a

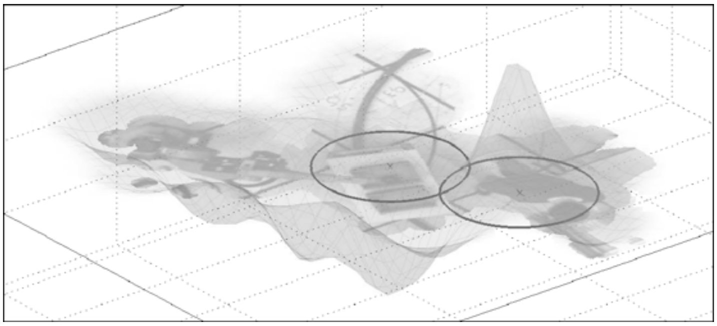
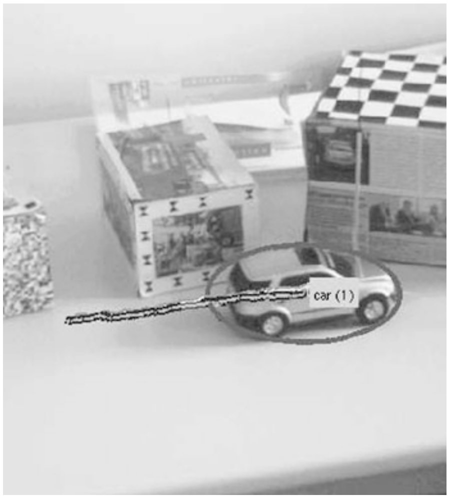


Fig. 2.22 Error function plot for minimisation, showing the error surface for $v = 0.26\ \mathrm{m\,s^{-1}}$, $r = 0.53$ m (*lower surface*), and $v = -0.79\ \mathrm{m\,s^{-1}}$, $r = 0.53$ m (*upper surface*, values mirrored for clearer display)

Fig. 2.23 Trajectory of the tracked object (*dark grey*) with annotated ground truth (*light grey*) for a tabletop sequence



consistent trajectory. The trajectory of a moving object in a fairly simple tabletop sequence as determined with the proposed algorithm is shown in Fig. 2.23 in comparison with the manually determined ground truth. This example demonstrates that a moving object can be separated from stationary background objects. In Sect. 7.2 we provide a detailed quantitative evaluation of our method in a complex industrial production scenario.