Region-based Pose Tracking*

Christian Schmaltz[1], Bodo Rosenhahn[2], Thomas Brox[3], Daniel Cremers[3], Joachim Weickert[1], Lennart Wietzke[4], and Gerald Sommer[4]

[1] Mathematical Image Analysis Group, Faculty of Mathematics and Computer Science,
Building E1 1 Saarland University, 66041 Saarbrücken, Germany
{schmaltz,weickert}@mia.uni-saarland.de

[2] Max-Planck Institute for Informatics, 66123 Saarbrücken, Germany
rosenhahn@mpi-sb.mpg.de

[3] Department of Computer Science,University of Bonn, 53117 Bonn, Germany
{brox,dcremers}@cs.uni-bonn.de

[4] Institute of Computer Science, Christian-Albrecht-University, 24098 Kiel, Germany
{lw,gs}@ks.informatik.uni-kiel.de

Abstract. This paper introduces a technique for region-based pose tracking without the need to explicitly compute contours. We assume a surface model of a rigid object and at least one calibrated camera view. The goal is to find the pose parameters that optimally fit the model surface to the contour of the object seen in the image. In contrast to conventional contour-based techniques, which acquire the contour to be extracted explicitly from the image, our approach optimizes an energy directly defined on the pose parameters. We show experimental results for rather challenging scenes observed with a monocular and a stereo camera system.

1 Introduction

The task to pursuit the 3-D position and orientation of a known 3-D object model from a 2-D image data stream is called 2-D–3-D pose tracking [8]. The need for pose tracking occurs in several applications, e.g. self localization and object grasping in robotics, or camera calibration. Particularly in scenes with cluttered backgrounds, noise, partial occlusions, or changing illumination, pose tracking is still a challenging problem even after more than 25 years of research [10].

A lot of different approaches to pose tracking have been considered [7, 12]. In [6], an iterative algorithm for real-time pose tracking of articulated objects, which is based on edge detection, has been proposed. Often points [1] or lines [2] are used for feature matching, but other features such as vertices, t-junctions, cusps, three-tangent junctions, limb and edge injections, and curvature L-junctions have also been considered [9].

Another way to approach pose estimation is to match a surface model of the object to be tracked to the object region in the images. Thereby, the computation of this region yields a typical segmentation problem. It has been proposed to optimize a coupled formulation of both problems and to solve simultaneously for the contours and the pose parameters via graph cuts [3] or via iterative approaches [4]. Although the coupled estimation of contours and pose parameters is beneficial compared to the uncoupled case, segmentation results can be inaccurate, as seen in Figure 1.

* We acknowledge funding by the German Research Foundation under the projects We 2602/5-1 and SO 320/4-2, and the Max-Planck Center for Visual Computing and Communication.

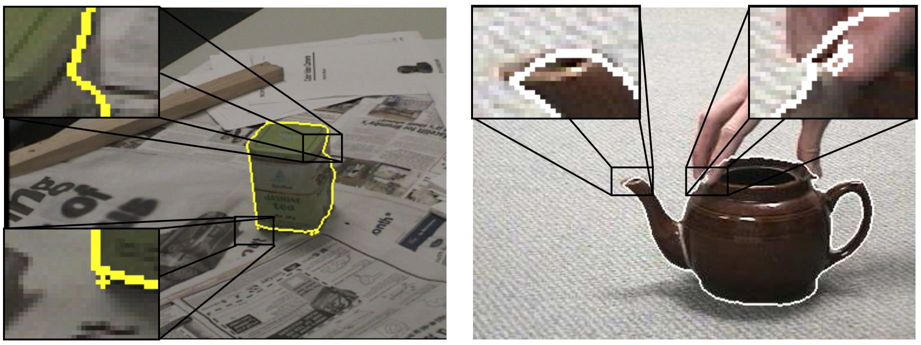


Fig. 1. Problems often occurring in variational segmentation algorithms: (a) An inaccurate segmentation. Note the split up into multiple connected components. (b) An error due to oversmoothing and another kind of undesired topological change.

In this paper, we build upon the method in [4] including its statistical representation of regions. However, instead of estimating 2-D segmentation and 3-D pose parameters separately we directly estimate 3-D pose parameters by minimizing the projection error in the respective 2-D images. Consequently, we can estimate segmentations which are by construction consistent with the 3-D pose. Moreover, the estimation of an infinite-dimensional level set function is replaced by the optimization of a small number of pose parameters. This results in a drastic speed-up and near real-time performance.
In the next section, we will briefly review pose estimation from 2-D–3-D point correspondences. We will then explain our approach in Section 3, followed by experimental results in Section 4. Section 5 concludes with a summary.

2 Pose Estimation from 2-D–3-D Point Correspondences

This section introduces basic concepts and notation and briefly describes the point-based pose estimation algorithm used in our approach [13]. Given some 3-D points x_i on the object, which are visible as 2-D points q_i in an image, the algorithm seeks a rigid body motion ξ such that each point x_i is on the line passing through q_i and the camera origin. Section 3 shows how such point correspondences are obtained with our method.

2.1 Rigid Motion and Twists

A rigid body motion in 3-D can be represented as $m(x) := Rx + t$, where $t \in \mathbb{R}^3$ is a translation vector and $R \in SO(3)$ is a rotation matrix with $SO(3) := \{R \in \mathbb{R}^{3\times 3} : \det(R) = 1\}$. By means of homogeneous coordinates, we can write m as a matrix M:

$$m((x_1,x_2,x_3)^T) = M(x_1,x_2,x_3,1)^T = \begin{pmatrix} R_{3\times 3} & t_{3\times 1} \\ 0_{1\times 3} & 1 \end{pmatrix} x \,. \tag{1}$$

The set of all matrices of this kind is called the *Lie group* $SE(3)$. To every Lie group there is an associated Lie algebra. Its underlying vector space is the tangent space of the

Lie group evaluated at the origin. The Lie algebra associated with the Lie group $SO(3)$ is $so(3) := \{A \in \mathbb{R}^{3\times 3} | A^T = -A\}$, whereas the Lie algebra corresponding to $SE(3)$ is $se(3) := \{(v,\omega) | v \in \mathbb{R}^3, \omega \in so(3)\}$. Since elements of $se(3)$ can be converted to $SE(3)$ and vice versa, we can represent a rigid motion as element of $se(3)$. Such an element is called *twist*. This is advantageous since a twist has only six parameters while an element of $SE(3)$ has twelve. Both have six degrees of freedom, though.

Elements of $so(3)$ and $se(3)$ can be written both as vectors $\omega = (\omega_1, \omega_2, \omega_3)$, $\xi = (\omega_1, \omega_2, \omega_3, v_1, v_2, v_3)$ and as matrices:

$$\hat{\omega} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \in so(3), \qquad \hat{\xi} = \begin{pmatrix} \hat{\omega} & v \\ 0_{3\times 1} & 0 \end{pmatrix} \in se(3) \ . \tag{2}$$

A twist $\xi \in se(3)$ can be converted to an element of the Lie group $M \in SE(3)$ by the exponential function $\exp(\hat{\xi}) = M$. This exponential can be computed efficiently with the Rodriguez formula. For further details we refer to [11].

2.2 Pose Estimation with 2-D–3-D Point Correspondences

Let (q,x) be a 2-D–3-D point correspondence, i.e. $x \in \mathbb{R}^4$ is a point in homogeneous coordinates on the 3-D silhouette of the object and $q \in \mathbb{R}^2$ is its position in the image. Furthermore, let $L = (n,m)$ be the Plücker line [14] through q and the respective camera origin. The distance of any point a to the line L given in Plücker form can be computed by using the cross product: $\|a \times n - m\|$, i.e., $a \in L$ if and only if $\|a \times n - m\| = 0$.

Our goal is to find a twist ξ such that the transformed points $\exp(\hat{\xi})x_i$ are close to the corresponding lines L_i. Linearizing the exponential function $\exp(\hat{\xi}) = \sum_{k=0}^{\infty} \frac{\hat{\xi}^k}{k!} \approx I + \hat{\xi}$ (where I is the identity matrix), we like to minimize with respect to ξ:

$$\sum_i \left\| \left(\exp\left(\hat{\xi}\right) x_i \right)_{3\times 1} \times n_i - m_i \right\|^2 \approx \sum_i \left\| \left(\left(I + \hat{\xi} \right) x_i \right)_{3\times 1} \times n_i - m_i \right\|^2 \to \min, \tag{3}$$

where the function $\cdot_{3\times 1} : \mathbb{R}^4 \mapsto \mathbb{R}^3$ removes the last entry, which is 1. Evaluation yields three linear equations of rank two for each correspondence (q_i, x_i). Thus, to solve for the 6 twist parameters, we need at least three correspondences for a unique solution. Usually, there are far more point correspondences and one obtains a least squares problem, which can be solved efficiently with the Householder algorithm. Since the twist ξ only corresponds to the pose change it is rather “small”. Thus, linearizing the exponential function does not create large errors. Moreover, we iterate this minimization process.

3 Region-based Model Fitting

Existing contour-based pose estimation algorithms expect an explicit contour to establish correspondences between contour points and points on the model surface. This involves a matching of the projected surface and the contour. Our idea is to avoid explicit computations of contours and the contour matching. Instead, we seek to adapt the pose parameters in such a way that the projections of the surface optimally split all images into the object and the background region. For simplicity, we will describe this setting for a single camera, but the concept is trivially extended to multiple views.

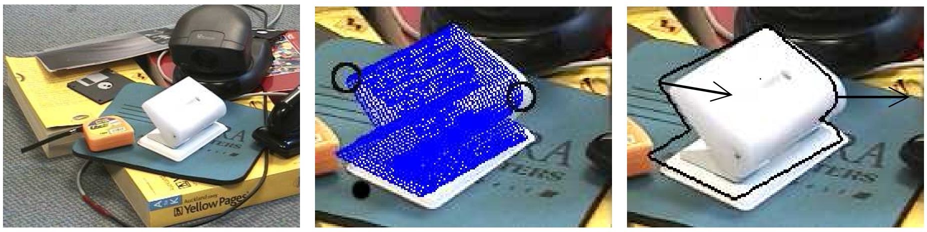


Fig. 2. From left to right: (a) Input image. The puncher is to be tracked. (b) Projection of the model in an inaccurate pose onto the image (magnified). The two marked points are the points referenced to in Section 3.2 (c) The 2-D contour of the projection (magnified). The arrows show into which directions these points should move in our algorithm.

3.1 Energy Model

Like in a segmentation task, we seek an optimal partitioning of the image domain Ω. This can be expressed as minimization of the energy function

$$E(\xi) = -\int_{\Omega} \left(P(\xi,q)\log p_1 + (1-P(\xi,q))\log p_2\right) dq \,, \tag{4}$$

where the function $P(\xi,q) \in (\mathbb{R}^6 \times \Omega \mapsto \{0,1\})$ is 1 if and only if the surface of the 3-D model with pose ξ projects to the point q in the image plane. P splits the image domain into two parts, in each of which different feature distributions are expected. These distributions are modeled by probability density functions p_1 and p_2.

Note the similarity of (4) to variational segmentation methods [4]. The important difference is that the partitioning is not represented by a contour, i.e. a function, but by only six parameters. Moreover, there is no constraint on the length of the boundary in (4).

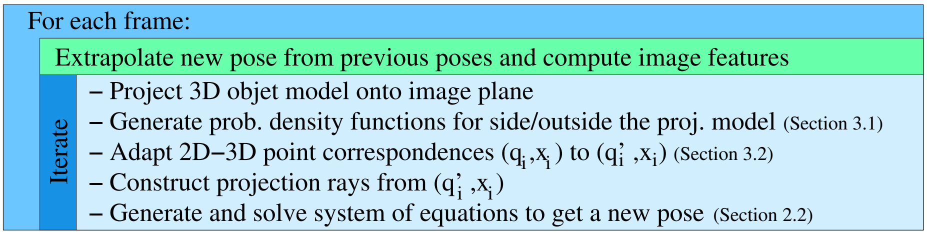
The probability densities are modeled by local Gaussian distributions [4] of the color in CIELAB color space, and texture in the texture feature space proposed in [5]. Since there is only a limited amount of data available to estimate the density functions, we consider the separate feature channels to be independent. Thus, the total probability density function is the product of the single channel densities.

The densities are adapted when the pose has changed. Given the projection of the model, and hence a partitioning of the image into object and background region, p_1 and p_2 can be computed from the local mean and variance in these regions.

3.2 Minimization

We minimize (4) by computing force vectors along the contour implicitly given by the projected surface. These force vectors indicate the direction to which the projection of the model should move to minimize $E(\xi)$. Using the framework from Section 2, we can transfer this force to the 3-D points and estimate the corresponding rigid body motion.

To this end, we create 2-D–3-D point correspondences (q_i, x_i) by projecting silhouette points x_i, given the current pose ξ, to the image plane where they yield q_i. If the function value $p_1(q_i)$ is greater than $p_2(q_i)$, it is likely that q_i belongs to the interior of the object



For each frame:

Extrapolate new pose from previous poses and compute image features

Iterate

- Project 3D objet model onto image plane
- Generate prob. density functions for side/outside the proj. model (Section 3.1)
- Adapt 2D–3D point correspondences (q_i, x_i) to (q'_i, x_i) (Section 3.2)
- Construct projection rays from (q'_i, x_i)
- Generate and solve system of equations to get a new pose (Section 2.2)

Fig. 3. Summary of the region-based pose tracking algorithm.

region. Thus, q_i will be shifted in inward normal direction to a new point q'_i. Vice versa, points q_i where the inequality $p_1(q_i) < p_2(q_i)$ holds will be shifted in outward normal direction. The normal direction is given by ∇P approximated with Sobel operators. The length $l := ||q' - q||$ of the shift vector is a parameter. More advanced methods how to choose l - including scaling l by $|\log p_1 - \log p_2|$, i.e. performing a gradient decent on E - have been tested but results were worse for our sequences.

This concept is illustrated in Figure 2. Figure 2b shows a white puncher, onto which the surface model has been projected. Figure 2c depicts the boundary between the interior and exterior of the projected model. Most of the points in the interior are white. So is the point marked by the right circle. Thus, it better fits to the statistical model of the object region than to the background and is moved away from the object. Vice-versa, the marked blue point on the left side is moved inwards as it better fits to the background.

We iterate this process. At some point, the pose changes induced by the force vectors mutually cancel out. We stop iterating when the average pose change after up to three iterations is smaller than a given threshold. Before changing frames in an image sequence, we predict the object's pose in the new frame by linearly extrapolating the results from the two previous frames. Figure 3 shows an overview of the algorithm.

4 Experiments

Figure 4 shows two frames of a monocular sequence, in which a wooden toy giraffe has been tracked. The projection of the estimated model fits the object in the image well.

Figure 5 depicts tracking results of a stereo sequence. First, a wooden beam moves between the cameras and the static object. Then the tea box is picked up and rotated several times. In the most challenging part of this sequence, the tea box is rotated around two different axis simultaneously while the bottom of the box reflects the background and moving specular highlights are visible. Nevertheless, our algorithm can track the tea box accurately over all 395 frames of this sequence.

For this sequence, an average of 12.03 iterations were necessary to reach the requested threshold (0.1mm for translation, 0.001 for rotations), with a maximum of 72 iterations. Approximately 28.75 minutes of processor time were needed on an Intel Pentium 4 with 3.2GHz ($\approx$ 12 frames per minute), about 86% of which was used for preprocessing (loading the images, computing texture features, etc.) while the rest was spent in the iteration steps. The parameters (i.e. the threshold and l) have been optimized to yield good poses. Faster but less accurate computations are possible, as explained below.

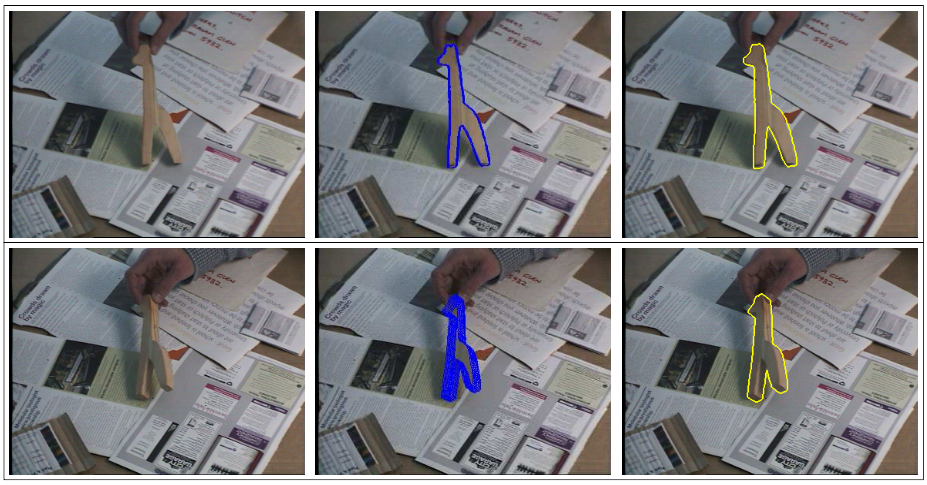


Fig. 4. From left to right: Input image, estimated pose and extracted contour for two frames of a color sequence with a wooden giraffe. **Top**: Frame 52. **Bottom**: Frame 68. The surface model consists of a single closed point grid. Thus, it is possible to look through the projected pose. Note that this is irrelevant for contour-based pose estimation, where only silhouette points are needed.

Figure 6 shows the time used by our program per frame. It can be seen that our algorithm is faster in "easy" situations, e.g. when nothing has moved. This figure also shows the changes in the translation and rotation parameters for the first 160 frames. Since tea box and camera are static in these frames no changes should occur. Our results have a standard deviation of about 1.79 degrees and 0.83mm.

When tracking objects that are clearly separated from the background (e.g. the puncher in Figure 2), features from the texture space can be neglected and the local Gaussian model can be replaced by a global model. These changes noticeably decrease the runtime of our algorithm. For example, the teapot shown in Figure 7 has been tracked in a stereo sequence with more than one frame per second. Ignoring texture information, the tea box sequence shown in Figure 5 can be tracked (with slightly less accurate results) in less than 4 minutes ($\approx$ 104 frames per minute). This indicates that real-time processing with a region-based approach is feasible.

5 Summary

We have presented an pose tracking algorithm from 2-D regional information which does not require a separate segmentation step. The implicit partitioning of the image by the projected object model is used for computing region statistics, which drive an evolution directly in the pose parameters. The algorithm can deal with illumination changes, cluttered background, partial occlusions, specular highlights and arbitrary rigid 3-D models. Experiments show that the results compare well to methods based on explicit contour representations. However, our approach is considerably faster and close to real-time performance.

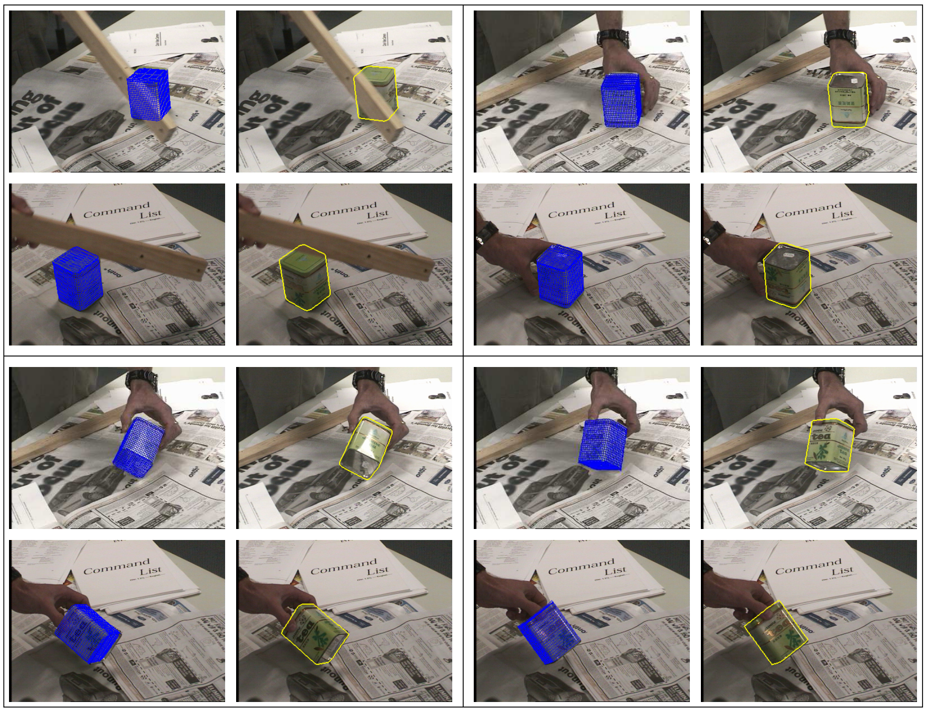


Fig. 5. Pose results for a tea box. Each block shows the computed pose (blue) and the contour (yellow) in the two views. The scene contains partial occlusions (frame 97, top left), the tea box is turned upside down (frame 230, top right), there are specular reflections (frame 269, bottom left) and the box is turned around different axes simultaneously (frame 277, bottom right).

References

1. M. A. Abidi and T. Chandra. Pose estimation for camera calibration and landmark tracking. In *Proc. International Conf. Robotics and Automation*, volume 1, pages 420–426, Cincinnati, May 1990.
2. J. Beveridge. *Local Search Algorithms for Geometric Object Recognition: Optimal Correspondence and Pose*. PhD thesis, Department of Computer Science, University of Massachusetts, Amherst, May 1993.
3. M. Bray, P. Kohli, and P. Torr. PoseCut: Simultaneous segmentation and 3D pose estimation of humans using dynamic graph-cuts. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision – ECCV 2006, Part II*, volume 3952 of *Lecture Notes in Computer Science*, pages 642–655, Berlin, 2006. Springer.
4. T. Brox, B. Rosenhahn, and J. Weickert. Three-dimensional shape knowledge for joint image segmentation and pose estimation. In W. Kropatsch, R. Sablatnig, and A. Hanbury, editors, *Pattern Recognition*, volume 3663 of *Lecture Notes in Computer Science*, pages 109–116, Berlin, August 2005. Springer.
5. T. Brox and J. Weickert. A TV flow based local scale estimate and its application to texture discrimination. *Journal of Visual Communication and Image Representation*, 17(5):1053–1073, October 2006.

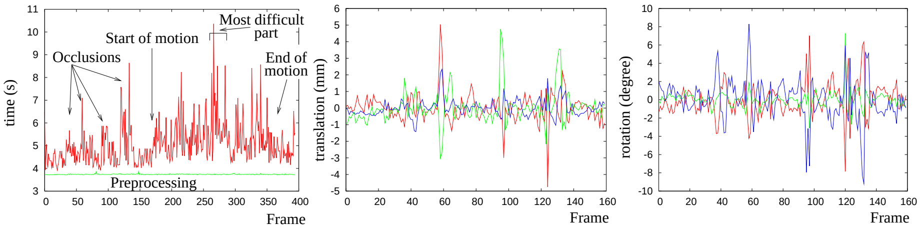


Fig. 6. Left: Time needed for preprocessing (straight green line) and the total time used per frame (red line) for the stereo image sequence shown in Figure 5. **Middle:** Changes in the three translation parameters in millimeters for the first 160 frames of this sequence. **Right:** Changes of the three Euler angles for the same frames, in degrees.

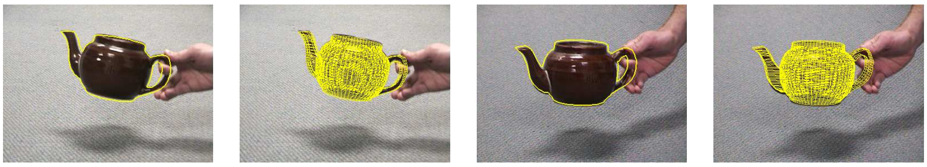


Fig. 7. Two views from a stereo image sequence in which a teapot has been tracked. Estimated contours and poses are shown in yellow.

6. T. Drummond and R. Cipolla. Real-time tracking of multiple articulated structures in multiple views. In D. Vernon, editor, *Computer Vision – ECCV 2000, Part II*, volume 1843 of *Lecture Notes in Computer Science*, pages 20–36, Berlin, 2000. Springer.
7. J. Goddard. *Pose And Motion Estimation From Vision Using Dual Quaternion-Based Extended Kalman Filtering*. PhD thesis, Imaging, Robotics, and Intelligent Systems Laboratory, The University of Tennessee, Knoxville-College of Engineering, Tennessee, 1997.
8. W. Grimson, T. Lozano Perez, and D. Huttenlocher. *Object Recognition by Computer: The Role of Geometric Constraints*. MIT Press, 1990.
9. D. Kriegman, B. Vijayakumar, and J. Ponce. Constraints for recognizing and locating curved 3D objects from monocular image features. In G. Sandini, editor, *Computer Vision – ECCV '92*, volume 588 of *Lecture Notes in Computer Science*, pages 829–833, Berlin, 2006. Springer.
10. D. Lowe. Solving for the parameters of object models from image descriptions. In *Proc. ARPA Image Understanding Workshop*, pages 121–127, College Park, April 1980.
11. R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton, 1994.
12. B. Rosenhahn. *Pose Estimation Revisited*. PhD thesis, Institute of Computer Science, Chair of Cognitive Systems, University of Kiel, Germany, 2003.
13. B. Rosenhahn and G. Sommer. Adaptive pose estimation for different corresponding entities. In L. Van Gool, editor, *Pattern Recognition*, volume 2449 of *Lecture Notes in Computer Science*, pages 265–273, Berlin, 2004. Springer.
14. F. Shevlin. Analysis of orientation problems using Plucker lines. In *Proc. 14th International Conference on Pattern Recognition*, volume 1, pages 685–689, Washington, DC, USA, 1998. IEEE Computer Society Press.