Robust contour-based object tracking integrating color and edge likelihoods

Giorgio Panin, Erwin Roth, Alois Knoll

Technische Universität München, Fakultät für Informatik
Boltzmannstrasse 3, 85748 Garching bei München, Germany
Email: {panin,knoll}@in.tum.de, eroth@mytum.de

Abstract

We present in this paper a novel object tracking system based on 3D contour models. For this purpose, we integrate two complimentary likelihoods, defined on local color statistics and intensity edges, into a common nonlinear estimation problem. The proposed method improves robustness and adaptivity with respect to challenging background and light conditions, and can be extended to multiple calibrated cameras. In order to achieve real-time capabilities for complex models, we also integrate in this framework a GPU-accelerated contour sampler, which quickly selects feature points and deals with generic shapes including polyhedral, non-convex as well as smooth surfaces, represented by polygonal meshes.

1 Introduction

Contour-based object tracking deals with the problem of sequentially estimating the 3D pose of an object in real-time, by making use of internal and external model edges. This information can be exploited by projecting, at a given pose hypothesis, the CAD model onto the current image, and identifying the visible *feature edges* [11]: for example, silhouette and flat surface boundaries, sharp internal edges of polyhedra, as well as texture edges, can be reliably identified in the image, since these are related to significant texture or shading discontinuities.

From the available feature edges, usually a set of points and screen normals is uniformly sampled and matched with the image data, by means of *likelihood* functions that can be defined and combined in several possible ways. In particular, we consider here intensity gradients and local color statistics, that provide two informative and complimentary visual modalities for an efficient data fusion (Fig. 1).

The idea of integrating intensity gradients with color likelihoods for improving robustness dates back to [2], where pose estimation was locally performed with a brute-force, discrete search for a 2D problem (elliptical head contour tracking).

When dealing with more general and complex 3D tasks, two efficient methods are provided by the well-known edge-based likelihood [8, 1, 4] and a color separation statistics known as the CCD algorithm [7, 13].

Both methods are based on Gaussian likelihoods, and correspond to a nonlinear least squares estimation (LSE) optimization starting from the predicted pose hypothesis of the previous frame. For the first frame (or after a track-loss condition), the system requires an initial pose information, which can be given manually or by an object detector algorithm. LSE problems are basically solved by means of Gauss-Newton optimization [6, Chap. 6] that can always be improved with more or less robust variants [10, 9, 5].

In this paper, we integrate the two methods into a common LSE framework, that can eventually be extended to multiple, calibrated cameras. Due to the improved robustness achieved by the multi-modal fusion, we are able to formulate the two cost functions in their basic version without any further improvement, thus keeping reasonable computational requirements.

Finally, since the contour sampling process itself may be a costly and object-specific procedure if performed with standard tools [14], we developed a GPU-accelerated visibility test and sampling technique derived from a related method for non-photorealistic rendering (NPR) [11], that makes use of generic polygonal meshes and deals with different shapes (polyhedral, smooth, planar); this proce-

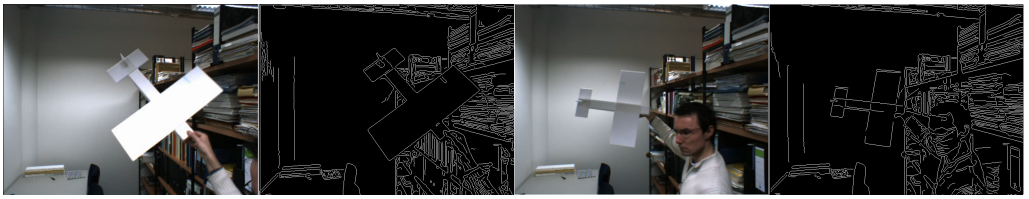

Figure 1: Local color statistics efficiently deal with situations showing a significant difference between foreground and background color regions (left frame), but may fail when a weak separation is present (right frame). On the contrary, an intensity-based tracker works well when luminance edges can be reliably detected (right edge map), but may be challenged by a locally cluttered background (left edge map).

dure enables to uniformly sample feature points and screen normals in real-time.

The paper is organized as follows: Section 2 formulates the multi-camera setup and pose parameters, while Section 3 shows our strategy for contour point sampling; the individual modalities, and our fusion methodology, are described in Section 4; experimental results of the proposed system are given in Section 5, and conclusions in Section 6.

2 Camera and pose parameters

Our basic scenario consists of a calibrated camera for tracking the 3D pose of a rigid object in Euclidean space, with respect to a given *world* reference frame.

We express and update in time the homogeneous (4×4) transformation matrix ${}^{W}_{O}T$ in terms of 6 incremental pose parameters p, through Lie algebras [3] and the exponential mapping

$$ {}^{W}_{O}T(p) = {}^{W}_{O}\bar{T} \exp(\sum_{i} p_i G_i) \qquad (1) $$

where G_i are the 6 generators for rigid motion, p is the corresponding *twist* vector [12], and ${}^{W}_{O}\bar{T}$ a reference transformation, estimated from the previous frame.

The following informations are supposed to be off-line available, via a standard camera calibration procedure:

- Intrinsic (3×4) projection matrix, K_C
- Extrinsic transformation matrix ${}^{W}_{C}T$ between world and camera frames

Therefore we define the *warp* function, mapping object points to camera pixels in homogeneous coordinates as

$$ \begin{aligned} \bar{y} &= K_C \left({}^{W}_{C}T \right)^{-1} {}^{W}_{O}T(p)\bar{x} \qquad (2) \\ y &= \begin{bmatrix} \frac{\bar{y}_1}{\bar{y}_3} & \frac{\bar{y}_2}{\bar{y}_3} \end{bmatrix}^T \end{aligned} $$

The LSE optimization requires also first derivatives of W in $p = 0$, that are straightforwardly computed from (2) and (1)

$$ \begin{aligned} \left. \frac{\partial y}{\partial p_i} \right|_{p=0} &= \frac{1}{\bar{y}_3^2} \begin{pmatrix} w_{i,1}\bar{y}_3 - w_{i,3}\bar{y}_1 \\ w_{i,2}\bar{y}_3 - w_{i,3}\bar{y}_2 \end{pmatrix} \qquad (3) \\ w_i &= K_C \left({}^{W}_{C}T \right)^{-1} {}^{W}_{O}\bar{T} G_i \bar{x} \end{aligned} $$

Due to the canonical methodology of using Lie algebras for linearization and the exponential representation (1), the estimation procedure can be extended to more complex transformations involving articulated or deformable models, by providing the related generators G_i. Uncalibrated problems can also be dealt with, by letting $K = [I\ 0]$ and defining the generators for a 2D transformation (similarity, affine, etc.).

3 Sampling contour points

The first step, common to both modalities, involves sampling good features for tracking from the object model, under a given pose and camera view.

Starting from a polygonal mesh model (Fig. 2), we first identify the visible feature edges at pose ${}^{W}_{O}T$:

- *silhouette*: boundary lines, located on the surface horizon
- *crease*: sharp edges between front-facing polygons of very different orientation

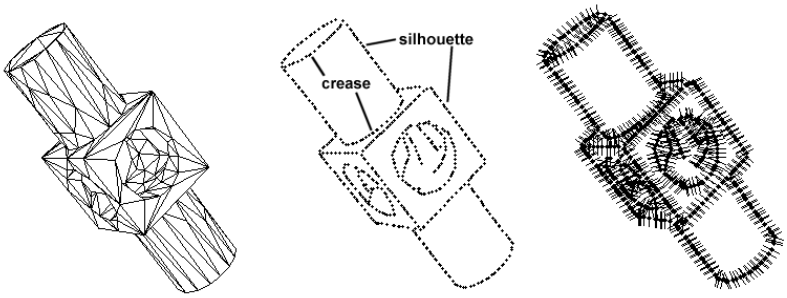

Figure 2: Sampling visible feature points and normals from a wireframe CAD model.

- *boundary*: boundary edges of flat surfaces

Afterwards, feature points h_i are uniformly sampled (right side of Fig. 2) in image space, also providing screen normals n_i, as well as Jacobians $\frac{\partial h_i}{\partial p}$ (3).

For this purpose, we developed a GPU-based procedure inspired by [11], that efficiently makes use of vertex and fragment shaders in order both to compute visible edges and to subsample them uniformly in screen space, also providing their location in 3D object space (which is required for computing Jacobians). Details about the procedure can be found in [15].

4 The integrated contour likelihood

4.1 Edge features matching

Edge measurements are performed by searching for image edges along normal segments to the contour

$$\mathcal{L}_i = \{h_i + dn_i;\ d = -D, ..., D\} \tag{4}$$

where D is a pre-defined search length.

This requires first to pre-process the image with a standard edge detector; we employ the Sobel filter, providing the 2D image gradient at point (x, y)

$$g(x, y) = [I_x(x, y)\ I_y(x, y)]^T \tag{5}$$

and match edge points according to the magnitude of their directional derivative:

$$z_{ij} = \left\{(x, y) \in \mathcal{L}_i : \left| n_i^T g(x, y) \right| > t \right\} \tag{6}$$

with t a detection threshold.

In order to obtain a uni-modal Gaussian likelihood, we keep the nearest neighbor

$$z_i = \arg\min_{z_{ij}}(\|z_{ij} - h_i\|) = \arg\min |d| \tag{7}$$

and the normal residual is given by

$$E_{i,edge} = n_i^T (h_i - z_i) \tag{8}$$

so that the edge-based Likelihood at pose T is

$$P_{edge}(z|T) \propto \prod_i \exp\left(-\frac{E_{i,edge}^2}{2R_{i,edge}^2}\right) \tag{9}$$

with measurement z, suitable variances $R_{i,edge}$, reflecting the average measurement uncertainty.

The overall residual vector $\mathbf{E}_{edge}$ and covariance matrix $\mathbf{R}_{edge} = blockdiag(R_{i,edge})$ are stored for the Gauss-Newton step, as well as the Jacobian matrix $\mathbf{J}_{edge}$, obtained by stacking together the normal-projected Jacobians J_i

$$J_i = n_i^T \frac{\partial h_i}{\partial p} \tag{10}$$

4.2 Color features: the CCD algorithm

A complimentary color-based Likelihood for single-contour models has been defined in [7] and formulated in the real-time version [13] as a non-linear LSE problem. This modality is based on the idea of maximizing separation of local color statistics between the two sides (foreground vs. background) of the object boundary.

This is achieved by iterating a two-step procedure akin to Expectation-Maximization:

1. *Collect local statistics*: Pixels are sampled in local areas on both sides around the contour, and color statistics are estimated
2. *Maximize separation likelihood*: Observed color pixels along the contour are classified according to the respective statistics (with a fuzzy assignment rule to each side), and the classification error is minimized with a Gauss-Newton step

The regions on which local statistics are collected, as well as the fuzzy membership function for pixel classification, are reduced at each iteration, providing a likelihood that contracts to a small, unimodal shape, which motivates the name CCD (*Contracting Curve Density*) for this algorithm.

In this paper, we apply the CCD algorithm to generic meshes like the example of Fig. 2, by limiting the contour sampling procedure to boundary and silhouette contours, while neglecting internal edges (crease, texture).

4.2.1 Step 1: compute local statistics

From each contour position h_i, foreground and background color pixels are collected along the normals n_i up to a distance L, and local statistics up to the 2^{nd} order are estimated

$$\begin{aligned} \nu_i^{0,B/F} &= \sum_{d=1}^{D} w_{id} \\ \nu_i^{1,B/F} &= \sum_{d=1}^{D} w_{id} I(h_i \pm L\bar{d}n_i) \\ \nu_i^{2,B/F} &= \sum_{d=1}^{D} w_{id} I(h_i \pm L\bar{d}n_i) I(h_i \pm L\bar{d}n_i)^T \end{aligned} \quad (11)$$

with the local weights w_{id} decaying exponentially with the normalized contour distances $\bar{d} \equiv d/D$, thus giving a higher confidence to observed colors near the contour. The $\pm$ sign is referred to the respective contour side ($-$ for the background region), and image values I are 3-channel RGB colors. Further details are given in [13].

Single-line statistics (11), are afterwards *blurred* along the contour, providing statistics distributed on local areas

$$\tilde{\nu}_i^{o,B/F} = \sum_j \exp(-\lambda\,|i-j|)\nu_j^{o,B/F};\ o = 0,1,2 \quad (12)$$

and finally normalized

$$\bar{I}_i^{B/F} = \frac{\tilde{\nu}_i^{1,B/F}}{\tilde{\nu}_i^{0,B/F}};\ \bar{R}_i^{B/F} = \frac{\tilde{\nu}_i^{2,B/F}}{\tilde{\nu}_i^{0,B/F}} \quad (13)$$

in order to provide the two-sided, local RGB means $\bar{I}$ and (3×3) covariance matrices $\bar{R}$.

4.2.2 Step 2: Compute color separation likelihood

The second step involves computing the residuals and Jacobian matrices for the Gauss-Newton pose update. For this purpose, observed pixel colors $I(h_i + L\bar{d}n_i)$ with $\bar{d} = -1, ..., 1$, are classified according to the collected statistics (13), under a fuzzy membership rule $a(x)$ to the foreground region

$$a(\bar{d}) = \frac{1}{2}\left[erf\left(\frac{\bar{d}}{\sqrt{2}\sigma}\right) + 1\right] \quad (14)$$

which becomes a sharp $\{0, 1\}$ assignment for $\sigma \to 0$; pixel classification is then accomplished by mixing the two statistics according to:

$$\begin{aligned} \hat{I}_{id} &= a(\bar{d})\bar{I}_i^F + (1 - a(\bar{d}))\bar{I}_i^B \\ \hat{R}_{id} &= a(\bar{d})\bar{R}_i^F + (1 - a(\bar{d}))\bar{R}_i^B \end{aligned} \quad (15)$$

and the color residuals are given by

$$E_{id} = I(h_i + L\bar{d}n_i) - \hat{I}_{id} \quad (16)$$

with measurement covariances $\hat{R}_{id}$.

By organizing in vector form the residual $\mathbf{E}_{ccd}$ and its covariance matrix $\mathbf{R}_{ccd} = blockdiag(\hat{R}_{id})$, we express the CCD likelihood as a Gaussian

$$P_{ccd}(z|T) \propto \exp\left(-\frac{1}{2}\mathbf{E}_{ccd}^T\mathbf{R}_{ccd}^{-1}\mathbf{E}_{ccd}\right) \quad (17)$$

which contracts after each update of T, by reducing the length parameter L via a simple exponential decay.

Finally, the $(3 \times n)$ derivatives of E_{id} can be computed by differentiating (15) and (14) with respect to the pose parameters p of the exponential mapping (1). As in [13], we neglect the dependence of R_{id} on p while computing the Jacobian matrices.

$$J_{id} = \frac{\partial \hat{I}_{id}}{\partial p} = \frac{1}{L}(\bar{I}_i^F - \bar{I}_i^B)\frac{\partial a}{\partial \bar{d}}\left(n_i^T\frac{\partial h_i}{\partial p}\right) \quad (18)$$

which are stacked together in a global Jacobian matrix $\mathbf{J}_{ccd}$.

4.3 Integrated Gauss-Newton update

Color and edge likelihoods (9), (17) can be jointly optimized in pose space, by assuming a basic independence condition for the two complimentary modalities

$$P\left(z|T\right) = P_{ccd}\left(z|T\right) P_{edge}\left(z|T\right) \tag{19}$$

which results in a joint LSE problem

$$\begin{aligned} T^{*} &= \arg\min_{T} \Big[w_{ccd} \left(\mathbf{E}^{T} \mathbf{R}^{-1} \mathbf{E} \right)_{ccd} \\ &\quad + w_{edge} \left(\mathbf{E}^{T} \mathbf{R}^{-1} \mathbf{E} \right)_{edge} \Big] \end{aligned} \tag{20}$$

with residuals and covariances given by (16),(15) and (8).

In order to optimize (20), we compute a Gauss-Newton update

$$\Delta p = H^{-1} \mathbf{g} \tag{21}$$

where

$$\begin{aligned} H &= \left(w \mathbf{J}^{T} \mathbf{R}^{-1} \mathbf{J} \right)_{ccd} + \left(w \mathbf{J}^{T} \mathbf{R}^{-1} \mathbf{J} \right)_{edge} \\ \mathbf{g} &= \left(w \mathbf{J}^{T} \mathbf{R}^{-1} \mathbf{E} \right)_{ccd} + \left(w \mathbf{J}^{T} \mathbf{R}^{-1} \mathbf{E} \right)_{edge} \end{aligned} \tag{22}$$

are the integrated Hessian matrix and gradient vector respectively, weighted by the inverse covariances $\mathbf{R}^{-1}$ and by the global weights w_{ccd}, w_{edges}.

Jacobian matrices $\mathbf{J}_{ccd}$ and $\mathbf{J}_{edge}$ are given by (18) and (10) respectively.

The resulting incremental pose Δp is used for updating the homogeneous transform T

$$T_{k+1} = T_{k} \exp(\Delta p_{i} G_{i}) \tag{23}$$

where the incremental matrix is computed with the exponential map (1).

This procedure is iterated until convergence, while contracting the CCD parameter L as described in 4.2.2. In order to save computational resources, the GPU visibility and sampling procedure is done only for the first Gauss-Newton iteration. Overall, we found a maximum of 10 iterations to be more than sufficient in most cases.

5 Experimental results

We tested the proposed system on simulated and real video sequences. The first experiment, see Fig. 3, shows tracking results based on a simulated video sequence using a toy airplane CAD model.

The toy airplane shows a nonconvex shape, with strong self-occlusions and a highly variable visible surface according to the attitude parameters. Despite this fact, the contour sampler gives satisfactory results at almost constant computation time, providing around 100 feature points for tracking at all poses.

The hardware for our experiments consists of a workstation with a 3.4GHz Intel Dual-Core processor, a programmable graphics card (NVIDIA 8600GT), and a standard FireWire camera for image acquisition, with a resolution of 640x480.

A simulated trajectory has been generated by specifying a set of key poses represented by twist vectors, interpolated in time with B-Splines. The object model has been rendered with artificial reflectance properties onto a constant background, as shown in Fig. 3. This background has, on the left side, a uniform color distribution very similar to the airplane surface, whereas on the right side provides a sharp color contrast, together with a highly cluttered edge map.

The three algorithms discussed so far have been run on this sequence and compared with the available ground truth; in particular, in order to evaluate the relative robustness, a track failure has been detected for estimated poses with orientation and/or position errors above a pre-defined threshold; in cases of failure, the system has been re-initialized by using the ground truth values. The ground truth pose data is also used for initializing the system.

The result of this comparison is shown in Fig. 4, where we can see how the fusion methodology achieves a lower failure rate (over 300 frames) compared to the CCD tracker, whereas the basic implementation of edge-based tracking has a higher failure rate under low contrast or cluttered background. The choice of fusion weights in (22) has been by a large extent arbitrary, and the default values $w_{ccd} = w_{edge} = 0.5$ showed to be satisfactory for all of the experiments (although a prior knowledge of the reliability of CCD could favor a higher weight for this modality). In the future it is planned to compute the weights dynamically from the re-

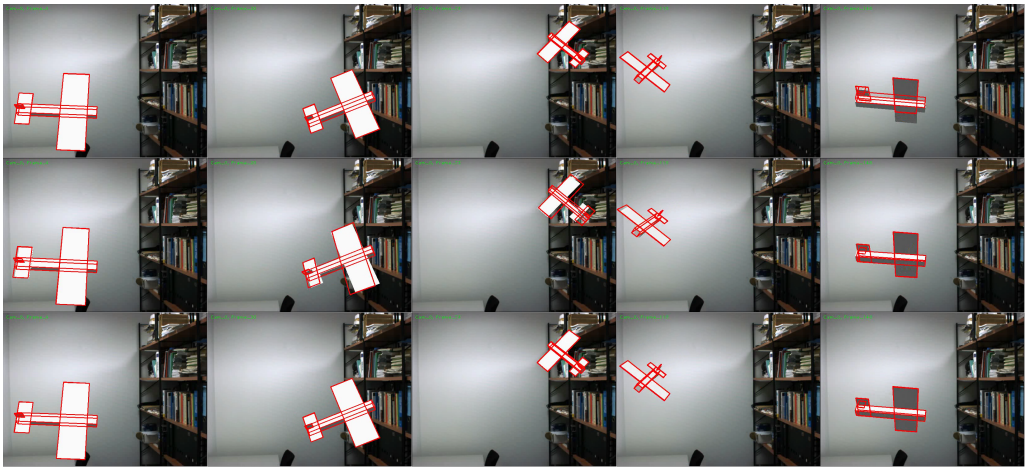

Figure 3: Tracking on a simulated sequence with the three methods: CCD (top row), Edges (middle row) and the proposed fusion technique (bottom row).

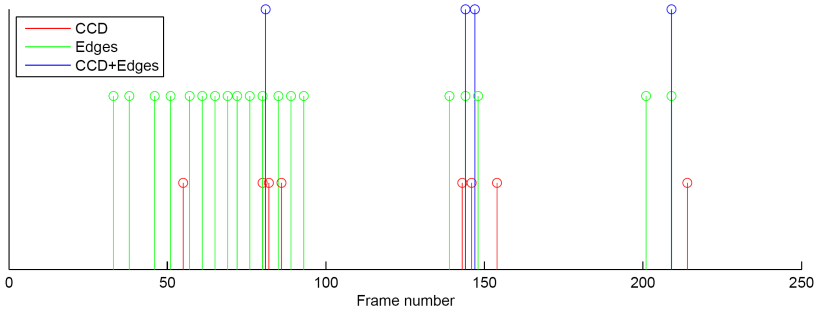

Figure 4: Failure cases on the simulated sequence.

spective covariance matrices.

Concerning the estimation precision for successful frames, orientation and position errors with respect to the ground truth have been computed, respectively, as the norm of the equivalent rotation vector (from the Rodrigues' formula) and the translation vector. Results are shown in Fig. 5, and average error values (rms) in Table 5. A slight improvement over CCD (that already provides a good precision) is observed, whereas the edge-based tracker also shows a less satisfactory precision.

Moreover, we observe that the average processing rate of the fusion method has been close to that of the CCD algorithm, which is more complex than the edge-based tracker; this is not surprising, since the most expensive processing steps (i.e. sampling contour points and computing screen Jacobians) are common to both algorithms.

Fig. 6 shows tracking results for a second experiment, where the proposed fusion method has been applied to a real video sequence with a more complex CAD airplane model. In this experiment, a Kalman filter is used additionally as state estimator, which receives the resulting incremental pose Δp from the Gauss-Newton update step (Section 4.3) as measurement input. The system is initialized by a manual pose estimation, and a white noise acceleration motion model is used to model the inertia of the airplane model.

Successful tracking of the model airplane has

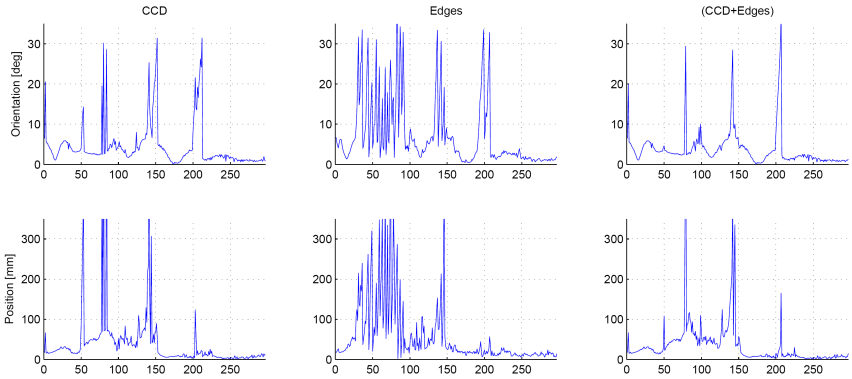

Figure 5: Position and orientation errors.

	Orient. errors [deg]	Pos. error [mm]	Failures	Frame rate [fps]
CCD	4.2274	31.2373	8	7.2
Edges	6.6262	50.5812	19	21.2
CCD+Edges	3.6608	31.3368	4	5.8

Table 1: Comparison of the three algorithms.

been achieved over more than 800 frames with an average frame rate of 14.4 fps for the integrated likelihood. The higher frame rate compared to the first experiment has been reached by limiting the maximum number of Gauss-Newton iterations to five and disabling visual debug output. The bottom row of Fig. 6 shows poses critical for tracking, where results could be significantly improved by applying a multi-camera setup.

6 Conclusions

We presented an integrated contour-based tracking methodology, combining two complimentary modalities within a common Gauss-Newton optimization scheme, that increases robustness against challenging background and foreground situations. The proposed system also includes an efficient GPU-accelerated procedure for feature edges selection and contour points sampling at each pose.

The integration achieves a better robustness with respect to each modality alone, while requiring a similar computational effort, due to the common operations involved. Future developments include a parallelization of the two likelihood and Jacobian computations, in order to provide a significant speed-up, as well as the extension of this methodology to multiple calibrated cameras, which can solve pose ambiguities and provide more robustness as well.

References

[1] M. Armstrong and A. Zisserman. Robust object tracking. In *Proceedings of the Asian Conference on Computer Vision*, volume I, pages 58–61, 1995.

[2] S.T. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *CVPR '98*, pages 232–237, 1998.

[3] T. Drummond and R. Cipolla. Visual tracking and control using lie algebras. In *CVPR '99*, page 2652, 1999.

[4] Tom Drummond and Roberto Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):932–946, 2002.

[5] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis

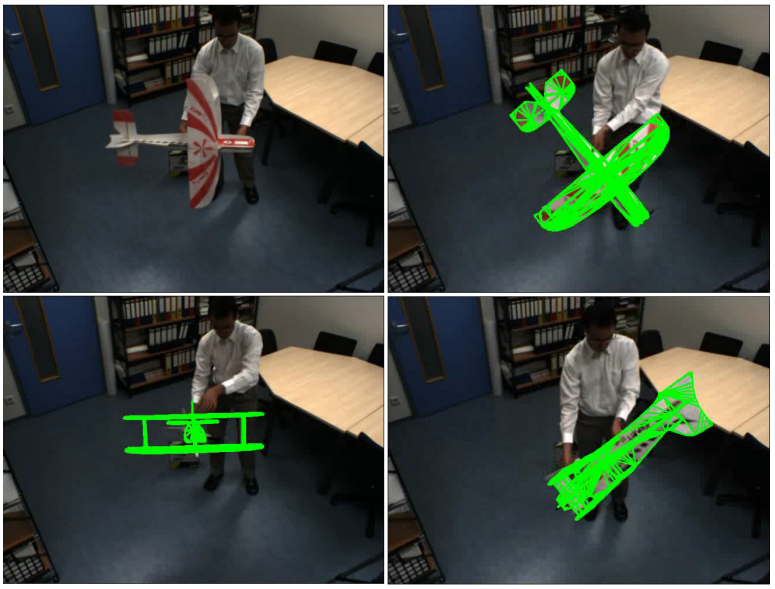

Figure 6: Tracking of a model aircraft with fused CCD and Edges likelihood based on a real video sequence. Critical poses for tracking (bottom row).

and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[6] R. Fletcher. *Practical methods of optimization; (2nd ed.)*. Wiley-Interscience, New York, NY, USA, 1987.

[7] Robert Hanek and Michael Beetz. The contracting curve density algorithm: Fitting parametric curve models to images using local self-adapting separation criteria. *Int. J. Comput. Vision*, 59(3):233–258, 2004.

[8] Chris Harris. Tracking with rigid models. In *Active Vision*, pages 59–73, Cambridge, MA, USA, 1993. MIT Press.

[9] P.J. Huber. *Robust Statistics*. Wiley, New York, 1981.

[10] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *j-J-SIAM*, 11(2):431–441, June 1963.

[11] Morgan McGuire and John F. Hughes. Hardware-determined feature edges. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pages 35–47, New York, NY, USA, 2004. ACM.

[12] Richard M. Murray, Zexiang Li, and Shankar S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC, March 1994.

[13] Giorgio Panin, Alexander Ladikos, and Alois Knoll. An efficient and robust real-time contour tracking system. In *ICVS '06: Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*, page 44, New York, USA, 2006.

[14] M. S. Paterson and F. F. Yao. Binary partitions with applications to hidden surface removal and solid modelling. In *SCG '89: Proc. of the fifth annual symposium on Computational geometry*, pages 23–32, New York, NY, USA, 1989. ACM Press.

[15] Erwin Roth, Giorgio Panin, and Alois Knoll. Sampling feature points for contour tracking with graphics hardware. In *Proceedings of Vision, Modeling, and Visualization*, 2008.