A REAL-TIME TRACKING SYSTEM COMBINING TEMPLATE-BASED AND FEATURE-BASED APPROACHES

Alexander Ladikos, Selim Benhimane, Nassir Navab
Department of Computer Science, Technical University of Munich, Boltzmannstr. 3 , 85748 Garching, Germany
ladikos@in.tum.de, benhiman@in.tum.de, navab@in.tum.de

Keywords: Real-Time Vision, Model-Based Object Tracking, Feature-based Tracking, Template-based Tracking.

Abstract: In this paper we propose a complete real-time model-based tracking system for piecewise-planar objects which combines template-based and feature-based approaches. Our contributions are an extension to the ESM algorithm (Benhimane and Malis, 2004) used for template-based tracking and the formulation of a feature-based tracking approach, which is specifically tailored for use in a real-time setting. In order to cope with highly dynamic scenarios, such as illumination changes, partial occlusions and fast object movement, the system adaptively switches between the template-based tracking, the feature-based tracking and a global initialization phase. Our tracking system achieves real-time performance by applying a coarse-to-fine optimization approach and includes means to detect a loss of track.

1 INTRODUCTION

Tracking lays the foundation for many application areas, including Augmented Reality, visual servoing and vision-based industrial applications. Consequently, there is a huge amount of related publications. The methods used for real-time 3D-tracking can be roughly divided into four categories: Line-based tracking, template-based tracking, feature-based tracking and hybrid approaches.

Line-based tracking requires a line model of the tracked object. The pose is determined by matching a projection of the line model to the lines extracted in the image. One of the first publications in this field was (Bouthemy, 1989). Recently a real-time line tracking system which uses multiple-hypothesis line tracking was proposed in (Wuest et al., 2005). The main disadvantage of line tracking is that it has severe problems with background clutter and image blurring so that in practice it cannot be applied in the applications we are targeting.

Template-based tracking fits better into our scenarios. It uses a reference template of the object and tracks it using image differences. This works nicely for well-textured objects and small interframe displacements. One of the first publications on template-based tracking (Lucas and Kanade, 1981) was using the optical flow in order to recover the translations in the image plane of the tracked objects. In order to improve the efficiency of the tracking and to deal with more complex objects and/or camera motions, other approaches were proposed (Hager and Belhumeur, 1998; Baker et al., 2001). In (Baker and Matthews, 2001) the authors compare these approaches and show that they all have an equivalent convergence rate and frequency up to a first order approximation with some being more efficient than others. A more recently suggested approach is the Efficient Second Order Minimization (ESM) algorithm (Benhimane and Malis, 2004), whose main contribution consists in finding a parametrization and an algorithm, which allow to achieve second-order convergence at the computational cost and consequently the speed of first-order methods.

Similarly to template-based tracking feature-based approaches also require a well-textured object. They work by extracting salient image regions from a reference image and matching them to another image. Each single point in the reference image is compared with other points belonging in a search region in the other image. The one that gives the best similarity measure score is considered as the corresponding one.

A common choice for feature extraction is the Harris corner detector (Harris and Stephens, 1988). Features can then be matched using normalized cross correlation (NCC) or some other similarity measure (Zhang et al., 1994). Two recent feature-matching approaches are SIFT (Lowe, 2004) and Randomized Trees (Lepetit et al., 2005). Both perform equally well in terms of accuracy. However, despite a recently proposed optimization of SIFT called SURF (Bay et al., 2006), SIFT has a lower runtime performance than the Randomized Trees, which exhibit a fast feature matching thanks to an offline learning step. In comparison to template-based methods, feature-based approaches can deal with bigger interframe displacements and can even be used for wide-baseline matching if we consider the whole image as the search region. However, wide-baseline approaches are in general too slow for real-time applications. Therefore they are mostly used for initialization rather than tracking. A full tracking system using only features was proposed in (Vacchetti et al., 2004). They rely on registered reference images of the object and perform feature matching between reference image and current image as well as between previous image and current image to estimate the pose of the object. However, the frame rate is not very high because of their complex cost function. Moreover image blurring poses a problem for feature extraction.

Hybrid tracking approaches combine two or more of the aforementioned approaches. Some recent related publications include (Pressigout and Marchand, 2005), which combines template-based tracking and line-based tracking. In (Vacchetti et al., 2004) the authors combine line-based tracking and feature-based tracking. Even though these algorithms perform well, the line-based tracking only improves the results for a few cases and might corrupt the result in the case of background clutter. In (Masson et al., 2004) the authors use a template-based method for tracking small patches on the object, which are then used for a point-based pose estimation. Since this approach uses a template-based method for tracking it cannot deal with fast object motion.

Our proposed system combines template-based and feature-based tracking approaches. The template-based tracking is used as the default tracking since it handles small interframe displacements, image blur and linear illumination changes well. In our system we adopt an extended version of the ESM algorithm, due to its high convergence rate and accuracy. For larger interframe displacements, which cannot be handled by the template-based algorithm, we use a feature-based approach making use of Harris points and NCC. We decided against using both feature-based and template-based tracking at the same time in a combined cost function, since features do not add any precision for small displacements and for big displacements the gradient direction given by ESM is usually erroneous. A combined approach also increases the computational burden, which not only slows down the tracker but also increases the interframe displacement. For the (re-)initialization we use Randomized Trees, because of their good runtime performance.

The rest of the paper is structured as follows: Section 2 introduces the theoretical background used in our system and section 3 describes our system design. In section 4 we present some simulations with ground-truth and some real-world experimental results. We conclude with section 5.

2 THEORETICAL BACKGROUND

Every (4×4) matrix $\mathbf{T}$ defining a 3D rigid body transformation is an element of the special Euclidean group $\mathbb{SE}(3)$. Moreover the Lie-Algebra $\mathfrak{se}(3)$ is linked to $\mathbb{SE}(3)$ through the exponential map. The base elements of $\mathfrak{se}(3)$ can be chosen as follows:

$$\mathbf{A}_1 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_x \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_4 = \begin{bmatrix} [\mathbf{b}_x]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_y \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_5 = \begin{bmatrix} [\mathbf{b}_y]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

$$\mathbf{A}_3 = \begin{bmatrix} \mathbf{0} & \mathbf{b}_z \\ \mathbf{0} & 0 \end{bmatrix} \quad \mathbf{A}_6 = \begin{bmatrix} [\mathbf{b}_z]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

with $\mathbf{b}_x = [1\ 0\ 0]^\top$, $\mathbf{b}_y = [0\ 1\ 0]^\top$ and $\mathbf{b}_z = [0\ 0\ 1]^\top$. The matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$ generate the translations and $\mathbf{A}_4$, $\mathbf{A}_5$, $\mathbf{A}_6$ generate the rotations. Consequently, we can parameterize a transformation matrix:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{SE}(3)$$

where $\mathbf{R}$ is the rotation and $\mathbf{t}$ is the translation, using the parameter vector that consists of the coefficients for each base element. Hence given a coefficient vector $\mathbf{x} = [x_1, x_2, ..., x_6]^\top$ the corresponding transformation matrix $\mathbf{T}$ is obtained as:

$$\mathbf{T}(\mathbf{x}) = \exp(\sum_{i=1}^{6} x_i \mathbf{A}_i) \tag{1}$$

In our system we also make heavy use of the relation between the movement of a plane in 3D and its movement in the image, since we suppose that every object can be considered as piecewise planar. As shown in (Hartley and Zisserman, 2004) every plane movement

induces a homography. Let the plane be $\pi = [\mathbf{n}\ d]^\top$ with normal $\mathbf{n}$ and distance d from the camera. Then the homography describing the transformation of the imaged plane is given by:

$$\mathbf{H}(\mathbf{T}) = \mathbf{K}\left(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^\top}{d}\right)\mathbf{K}^{-1} \quad (2)$$

where $\mathbf{K}$ are the intrinsic parameters of the camera. The basic cost function used for template-based tracking is defined as follows: Let I^* be the reference image and I the current image. Further let $\mathbf{p}$ be the pixel coordinates of the pixels in the reference image and $\widehat{\mathbf{T}}$ an initial pose estimate for the current image. Our goal is to estimate an incremental pose update $\mathbf{T}(\mathbf{x})$ with $\mathbf{x}$ the parameter vector encoding rotation and translation. Let $\mathbf{w}$ be the warping function. The cost function is then given as:

$$f(\mathbf{x}) = \sum_{\mathbf{p}}\left[I\left(\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\right)\right)(\mathbf{p})\right) - I^*(\mathbf{p})\right]^2 \quad (3)$$

Due to the virtues of the parametrization it is possible to only evaluate a Jacobian, which depends on the reference image and the current image, and still achieve second order convergence (Benhimane and Malis, 2004).

3 PROPOSED SYSTEM

An overview of the proposed system as a finite state machine (FSM) is given in Figure 1.
The system starts with an initialization phase, which will be described in section 3.2. It then uses the template-based tracking algorithm to track the object as explained in section 3.3. In the event that template-based tracking fails the feature-based tracking, as described in section 3.4, is used. If the feature-based tracker is unable to recover the pose within a certain number of attempts the initialization is invoked again. Section 3.5 describes the transitions of the FSM and the reasoning behind them.

3.1 Required Information

In our system we use a textured 3D model of the object. This model can either be created manually or semi-automatically with commercially available products. One point to note is that it is advisable to use the same camera for texturing the model and for tracking, because this minimizes difficulties due to different image quality and image formation conditions.
For the initialization registered images of the object,

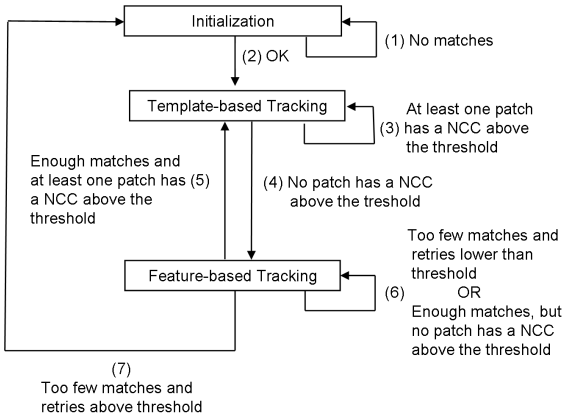


Figure 1: Overview of the proposed tracking system.

called keyframes, are required. They can be created directly from the textured model by rendering it from different views.
If the real-world metric pose is required, the correct intrinsic camera parameters have to be provided.

3.2 Initialization

Initialization is performed using Randomized Trees. The Randomized Trees algorithm requires a reference image of the object in order to learn the appearance of the feature points. When initializing, features are extracted from the current image and matched to the features extracted in the keyframe. The pose can then be estimated from the 3D object points and corresponding 2D feature points in the current image.
Since the tracker is using a textured model of the object the accuracy of the initial pose estimation is not very critical. If on the other hand the reference templates used for tracking were extracted from the current image, the precision of the initialization procedure would be a major issue, because the quality of the tracking result depends directly on the quality of the templates used for tracking. Hence we decided to directly use the templates taken from the textured model in our system.

3.3 Template-based Tracking

We use the ESM algorithm for template-based tracking. The object is tracked using this method until a loss of track is detected, in which case the feature-based tracker is invoked.

3.3.1 Reference Patch Extraction

The textures of the reference patches, which are required for tracking, are taken from the textured model.

For each patch the object is rendered so that the patch is oriented parallel to the image plane. It is also important to ensure that the relative sizes of the object patches are reflected in the size of the rendered patches, since the number of pixels in a patch is directly proportional to its importance during tracking. Since the pose parameters used to render the patches are known, the reference patches can be directly extracted from the rendered image. After this for every patch k the following information is available: The reference patch I_k^*, the pose $\widetilde{\mathbf{T}}_k$ under which it was extracted, the patch normal $\mathbf{n}_k$ and its distance to the camera d_k. These reference patches are then reduced a few times in size by a factor of two to create a stack of reference patches at different scales, which are used to speed up the tracking in a coarse-to-fine approach.

3.3.2 Visibility Test

Attempting to track patches which are not visible will lead to erroneous results. Hence it is necessary to ascertain the visibility of every patch. This test is performed by rendering the model with OpenGL and using the occlusion_query extension to test which patches are visible and which are occluded. The visibility test is performed for each frame using the pose estimated in the previous frame. Thanks to the occlusion_query extension the visibility test can be performed very fast, so that it does not interfere with the tracking performance.

3.3.3 The Extended ESM Algorithm

We extended the formulation of the ESM algorithm as given in section 2. This extension is required since in the original formulation it is implicitly assumed that all reference patches come from the same image, i.e. they were extracted in the same coordinate system. However, this is not possible when using the rendered patches, since each patch is seen under a different pose. For instance the front and back face of a cube can not be seen at the same time. Hence it would be impossible to track all the patches in the same coordinate system. This would mean that each patch had to be tracked independently without considering the constraints imposed by the object geometry. To overcome this problem the pose $\widetilde{\mathbf{T}}_k$ under which the reference patch was extracted has to be incorporated into the algorithm. This leads to the modified cost function:

$$f(\mathbf{x}) = \sum_k \sum_{\mathbf{p}_k} \left[I\left(\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\widetilde{\mathbf{T}}_k^{-1}\right)\right)(\mathbf{p}_k)\right) - I^*(\mathbf{p}_k)\right]^2 \tag{4}$$

In order to speed up the optimization, we start at the highest scale level (lowest resolution) and optimize the cost function on this level until convergence is achieved or until the maximum number of iterations has been exceeded. If the optimization converges before the maximum number of iterations has been reached it is restarted on the next scale level with the pose estimated on the previous level. This is continued until the lowest scale level (highest resolution) is reached or the maximum number of iterations is exceeded.

3.3.4 Loss Of Track

Determining when the tracker lost the object is important in order to switch to the feature-based tracking algorithm. In our system this is accomplished by computing the normalized cross correlation (NCC) between the reference patch I_k^* and the current patch I_k after the end of the optimization for all visible patches. The NCC between two patches is defined as:

$$\mathrm{NCC}(I_k^*, I_k) = \frac{\sum_{\mathbf{p}_k}(I_k^*(\mathbf{p}_k) - \mu_k^*)(I_k(\mathbf{p}_k) - \mu_k)}{N_k^2 \sigma_k^* \sigma_k} \tag{5}$$

where N_k is the number of pixels of each patch, μ_k^* and μ_k are the mean pixels intensities and σ_k^* and σ_k their standard deviations.
If the NCC of a patch falls below a certain threshold, it is excluded from the tracking. If all the patches fall below the threshold the feature-based tracker is invoked.

3.4 Feature-based Tracking

In the event that the template-based tracker fails, the feature-based tracker is invoked. For our feature-based tracking approach we extract Harris corner points on the same reference patches used for the template-based tracking and subsequently match them to the current patch (i.e. the patch as seen in the current image) using NCC. Because NCC is not scale and rotation invariant a method had to be devised to ensure that the two patches will be seen under almost identical poses.
This is achieved as follows: Since the pose $\widetilde{\mathbf{T}}_k$ under which the reference patch k and hence the feature points were extracted is known, it is possible to determine the homography by which the current image has to be warped to obtain the reference patch. However since the object pose in the current image is not known, the pose $\widehat{\mathbf{T}}$ recovered in the previous frame is used as an approximation. Hence the current image has to be warped with the homography:

$$\mathbf{H} = \left(\mathbf{H}(\widehat{\mathbf{T}}\widetilde{\mathbf{T}}_k^{-1})\right)^{-1} \tag{6}$$

Since the warping uses the pose from the previous frame the warped patch will not look exactly like the reference patch, but supposing reasonable constraints on the maximum speed of the object, it is safe to assume that the deformations will only be minor so that NCC can still be used as a similarity measure. The feature points are then extracted in the warped patch.
Let the matched points in the reference image and the current image be $\mathbf{p}_{k,i}$ and $\mathbf{p}'_{k,i}$ respectively. First outliers are removed using RANSAC (Fischler and Bolles, 1981). Then the pose is estimated by minimizing the cost function:

$$f(\mathbf{x}) = \sum_k \sum_i \|\mathbf{w}\left(\mathbf{H}\left(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})\widetilde{\mathbf{T}}_k^{-1}\right)\right)(\mathbf{p}_{k,i}) - \mathbf{p}'_{k,i}\|^2 \tag{7}$$

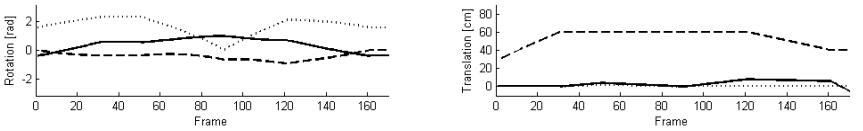
The parametrization is identical to that used in the template-based algorithm. Since RANSAC was already applied to remove the outliers there is no need to use a robust cost function, so a simple least-squares approach suffices.
Using the warped patches for the matching is advantageous for several reasons. First it allows the use of NCC for matching instead of a more expensive affine-invariant matching algorithm. Secondly it reduces the computational time for feature extraction, because it is only necessary to extract Harris points on the warped patch and not on the whole image. A further advantage is that this approach removes matching ambiguities in the case that multiple patches have the same texture, since by considering the previous pose only the correct patch will be used for the matching.

3.5 Finite State Machine

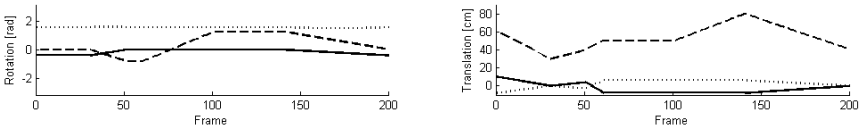
To decide which algorithm to use for a given frame we designed a finite state machine (cf. figure 1).
The system starts out in the initialization phase and stays in this phase until the the object is found in the image (transition (1)). Once the object has been found we switch to the template-based tracking phase (transition (2)). The reason for starting with template-based tracking rather than with feature-based tracking is the higher accuracy and the higher frame rate, since it is possible to use a coarse-to-fine optimization approach. As long as there is at least one patch left that has a NCC higher than the threshold the template-based tracker will be used (transition (3)).
When the NCC score of all patches falls below a certain threshold the system switches to the feature-based tracker (transition (4)), because otherwise the tracking would diverge. An important issue is choosing a good threshold for the NCC. We found that a value between 0.5 and 0.7 gives the best results. For lower values the system looses track, while for higher values the feature-based approach is used most of the time, even though the template-based tracker would be faster.
Even in the feature-based tracking phase the NCC between the reference patches and the current patches is computed. If there are enough feature matches to determine the pose, the system goes back to template-based tracking (transition (5)) unless there are no patches with a NCC above the threshold. In this case the system continues to use features (transition (6)) until at least one patch has a NCC above the threshold. If the pose cannot be recovered in the current frame the feature-based tracker is given another chance on the next few frames (transition (6)). The reason for this is that the object might just have been blurred in the current frame because of too fast motion, which makes both template-based tracking and feature extraction difficult. Often, however, the object slows down after a few frames, so that the feature-based tracker can find it again. If the object still cannot be found after a certain number of frames have been seen the initialization is invoked again (transition (7)).

4 EXPERIMENTS

To evaluate the validity of our approach we performed several experiments on synthetic data with ground-truth and real data.
The frame rate of our system is in the range between 25 fps and 40 fps on a 1.66 GHz Intel Core-Duo CPU with 1 GB of memory. The exact value depends on a multitude of factors including the size of the reference patches, the number of scale levels, the number of feature points and the desired accuracy.
The synthetic experiments consisted of creating an animation with a textured 3D model and comparing the recovered pose parameters to the actual ones. Figure 2 shows the ground-truth motion of one sequence with 170 and one sequence with 200 frames. There are big rotations, fast object movement and big scale changes present in both sequences. The range of the rotations is 120 degrees and the range of the translations is around 40 cm. Figure 3 and figure 4 show the absolute translation and rotation errors for the first sequence and second sequence respectively. All methods have a very small error of normally less than 3 degrees for the rotations and 4 mm for the translations. In the first sequence the extended ESM algorithm looses track at frame 162 (see figure 3(a)) due to fast object translation along the x-axis (see figure 2(a)). The feature-based algorithm already looses track much earlier at frame 31 (see figure 3(b)), because it cannot find any feature

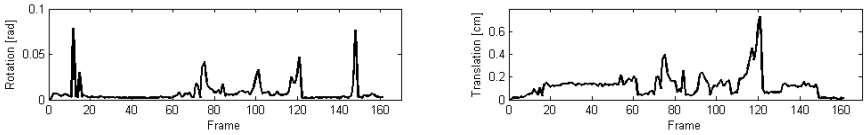


(a) Sequence 1

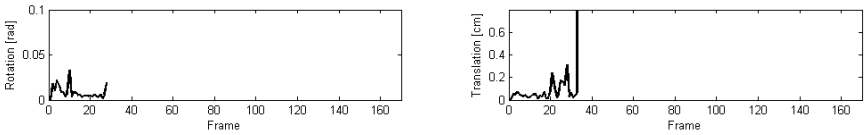


(b) Sequence 2

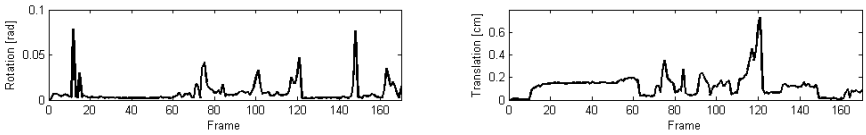
Figure 2: Ground-truth motion in synthetic sequences (solid = x-axis, dotted = y-axis, dashed = z-axis).



(a) Extended ESM

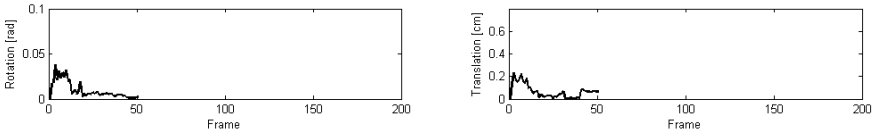


(b) Features

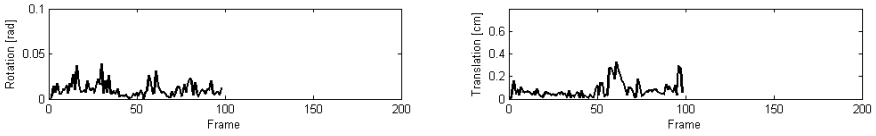


(c) Proposed Approach

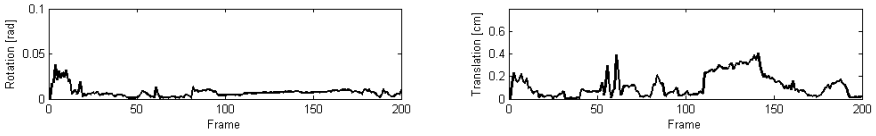
Figure 3: Absolute error on synthetic sequence 1.



(a) Extended ESM

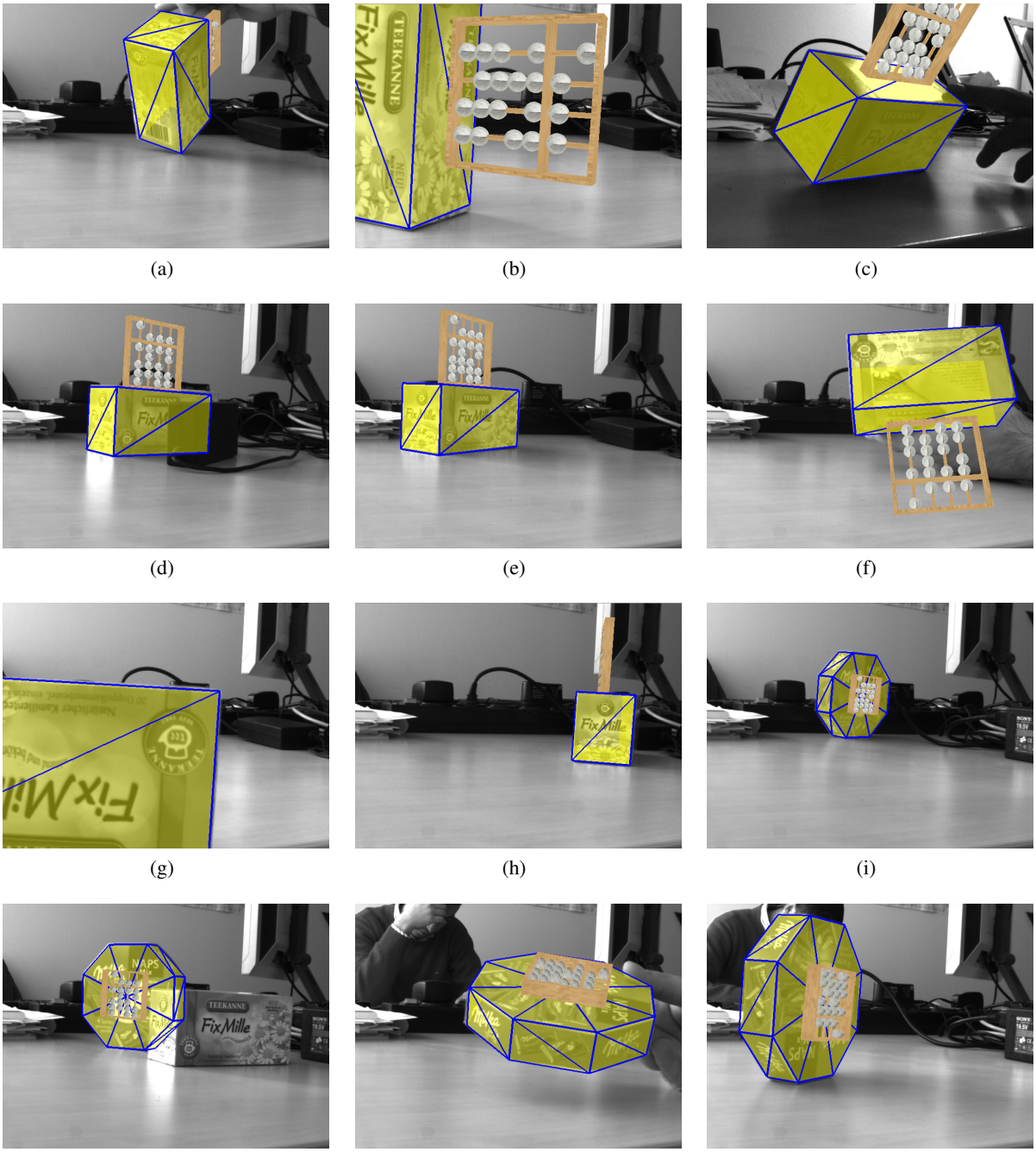


(b) Features



(c) Proposed Approach

Figure 4: Absolute error on synthetic sequence 2.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figure 5: Results on real data under different conditions such as illumination changes and occlusions.

matches when the object is seen at an oblique angle. In the second sequence the feature-based algorithm performs better than the extended ESM algorithm (see figure 4). However neither algorithm can track the whole sequence. Our tracking approach on the other hand successfully tracks both sequences entirely, because it changes the tracking algorithm used at the right moment. We obtained similar results on all synthetic sequences we simulated. Since there are no blurring, illumination changes or noise in the synthetic sequences it is not possible to show how our system deals with these conditions. Therefore we also performed many real-world experiments using different objects.

Figure 5 shows some experiments on real sequences made with a tea box and a candy box under varying tracking conditions. The images show how our system deals with partial occlusions (b,d,f,g,j), illumination changes (c), changes in scale (b,g,i,h) and severely oblique viewing angles (k,l). This shows that the proposed algorithm is able to deal with dynamic scenarios and solve the major limitations of classical tracking algorithms such as partial occlusions, illumination changes and fast object movement. We can also see that it is possible to robustly overlay virtual objects in order to perform Augmented Reality.

5 CONCLUSION

We presented a tracking system which intelligently combines template-based and feature-based tracking. The contributions are the extension of the ESM algorithm, the formulation of the feature-based tracking and the FSM for deciding which algorithm to use for the current frame. The system has been tested on real-world sequences as well as on simulations and performs at high frame rates on a standard PC.

Compared to other algorithms proposed in the literature we achieve a higher frame rate and more robustness to fast object motions. Our approach also gives good results in the face of partial occlusions and illumination changes.

REFERENCES

Baker, S., Dellaert, F., and Matthews, I. (2001). Aligning images incrementally backwards. Technical report, Robotics Institute, Carnegie Mellon University.

Baker, S. and Matthews, I. (2001). Equivalence and efficiency of image alignment algorithms. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1090–1097.

Bay, H., Tuytelaars, T., and van Gool, L. (2006). SURF: Speeded up robust features. *European Conference on Computer Vision*.

Benhimane, S. and Malis, E. (2004). Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*, pages 943–948.

Bouthemy, P. (1989). A maximum likelihood framework for determining moving edges. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(5):499–511.

Fischler, M. and Bolles, R. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Hager, G. and Belhumeur, P. (1998). Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conf.*, pages 147–151.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

Lepetit, V., Lagger, P., and Fua, P. (2005). Randomized trees for real-time keypoint recognition. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 775–781.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Lucas, B. and Kanade, T. (1981). An iterative image registration technique with application to stereo vision. In *Int. Joint Conf. on Artificial Intelligence*, pages 674–679.

Masson, L., Dhome, M., and Jurie, F. (2004). Robust real time tracking of 3d objects. In *Int. Conf. on Pattern Recognition*, pages 252–255.

Pressigout, M. and Marchand, E. (2005). Real-time planar structure tracking for visual servoing: a contour and texture approach. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*.

Vacchetti, L., Lepetit, V., and Fua, P. (2004). Combining edge and texture information for real-time accurate 3d camera tracking. In *Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 48–57.

Wuest, H., Vial, F., and Stricker, D. (2005). Adaptive line tracking with multiple hypotheses for augmented reality. In *Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 62–69.

Zhang, Z., Deriche, R., Faugeras, O., and Luong, Q.-T. (1994). A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Technical Report 2273, INRIA.