Linear and Quadratic Subsets for Template-Based Tracking

Selim Benhimane[1] Alexander Ladikos[1] Vincent Lepetit[2] Nassir Navab[1]

[1]Department of Computer Science, Technical University of Munich
Boltzmannstr. 3 , 85748 Garching, Germany
[2] École Polytechnique Fédérale de Lausanne (EPFL), Computer Vision Laboratory
CH-1015 Lausanne, Switzerland

selim.benhimane@in.tum.de, ladikos@in.tum.de, vincent.lepetit@epfl.ch, navab@cs.tum.edu

Abstract

We propose a method that dramatically improves the performance of template-based matching in terms of size of convergence region and computation time. This is done by selecting a subset of the template that verifies the assumption (made during optimization) of linearity or quadraticity with respect to the motion parameters. We call these subsets linear or quadratic subsets.

While subset selection approaches have already been proposed, they generally do not attempt to provide linear or quadratic subsets and rely on heuristics such as texturedness. Because a naive search for the optimal subset would result in a combinatorial explosion for large templates, we propose a simple algorithm that does not aim for the optimal subset but provides a very good linear or quadratic subset at low cost, even for large templates. Simulation results and experiments with real sequences show the superiority of the proposed method compared to existing subset selection approaches.

1. Introduction

Since the seminal work of Lucas and Kanade [8], many improvements to template-based tracking have been proposed, mostly focusing on improving the efficiency [6, 5, 11, 1, 7, 4, 2, 3, 9]. Template-based tracking is usually performed by minimizing the sum-of-squared intensity differences, and many new optimization schemes have been introduced to enable the precomputation of all of or parts of the matrices involved. Other approaches focus on the restriction to some pixels of the template to ameliorate the computational cost without decreasing the performance in terms of convergence too much.

To improve the performance of template-based tracking even further, we propose an approach that aims to improve the convergence behavior of the algorithm. Surprisingly, very few approaches considered that direction, while the convergence region has an obvious influence on the computation time. A notable exception is the recent work of Matas *et al.* [9] that proposes to select the pixels that most closely verify the approximation used by the optimization. This greatly improves the tracking result. However, the pixel subset is selected by a greedy search over randomly sampled subsets and each subset is tested against the linearity assumption on synthetically warped views of the template. The computation time quickly increases with the size of the template. Moreover, a series of linear predictors has to be learned in order to handle different motion ranges.

We therefore introduce a subset construction algorithm suitable for large templates, making the selection approach more useful in practice. We call these subsets *linear* or *quadratic subsets*, depending on the optimization method chosen for the template-based tracking. The proposed algorithm makes it possible to select subsets that enlarge the convergence region of the optimization method used. Our algorithm is based on a simple remark that allows to build large subsets very efficiently: If two subsets are linear (or quadratic) with respect to the motion parameters under any local motion, then their union is also a linear (or quadratic) subset. Since we only learn the subsets for a finite number of local motions our algorithm does not necessarily provide the largest linear or quadratic subset, but as shown by our experiments, it is sufficient to considerably improve the convergence properties of the tracking.

We tested our approach with two popular optimization methods for template-based tracking, namely the Inverse Compositional algorithm and the Efficient Second-order Minimization. In both cases, it always results in a significant reduction in terms of computation time and a significant gain in terms of the convergence region. In particular, it doubles the convergence frequencies of the Inverse Compositional algorithm for large motions.

The rest of this paper is structured as follows: Section 2 describes some work related to our approach. Section 3 quickly recalls the formulation of the tracking algorithms considered for subset selection. Our algorithm for selecting linear and quadratic subsets is presented in Section 4. Section 5 and Section 6 provide results obtained on simulations and real sequences. We conclude with Section 7.

2. Related Work

One of the first publications on template-based tracking was the Lucas-Kanade algorithm [8], which uses optical flow for recovering the translation of an object in the image plane. Since then, many other approaches have been proposed to improve the tracking efficiency and accommodate more complex movements of the tracked objects, either by modifying the cost function or by performing different approximations and linearizations [6, 11, 1]. Baker and Matthews [2] proved that all of these approaches are equivalent in terms of convergence. However, in terms of computation time, the Inverse Compositional (IC) algorithm [1] gives the best results by making it possible to precompute many terms. Another way to improve the efficiency of template-based tracking is to improve the convergence behavior of the algorithm. Benhimane and Malis [3] propose the Efficient Second-order Minimization (ESM), which exhibits the advantages of second-order optimization, i.e. it converges faster and it has a larger convergence region, without the need of the prohibitive Hessians computation.

In order to increase the speed of template-based tracking approaches it is also possible to consider only a subset of the template, and several methods have been proposed for selecting pixels without degrading the convergence behavior too much. The main difficulty is that the number of possible subsets increases exponentially with the size of the template —it is equal to 2^N where N is the number of pixels of the template— making an exhaustive search even for small templates intractable. Therefore, mainly heuristics have been proposed. In particular, Shi and Tomasi [10] consider the pixels that can be localized precisely under affine deformations and end up with a "texturedness measure" for efficient extraction. In [5], Dellaert and Collins perform the selection based on the reduction of the uncertainty of the estimated motion and of the redundancy of information provided by the pixels. They end up with a measure closely related to the one of Shi and Tomasi except that a prior on motion can also be taken into account when available. However, this is not sufficient alone, and they have to enforce a good distribution of the pixels over the template.

It is true that pixels with high "texturedness" contain rich information, and restricting the template to these pixels makes the algorithm faster without loosing too much robustness. However, this does not mean that it will lead to a better convergence. In fact, Zivkovic and van der Heijden [12] show in the context of interest point tracking that extracting pixels with large convergence regions gives much better results in practice.

More recently, a method based on the convergence properties has been proposed by Matas *et al.* [9]. They use the optimization method proposed by [7], where instead of relying on analytical Jacobians, a linear relation between the image differences and the motion is estimated from motions generated during an off-line phase. Since this relation is not necessarily exactly linear, they look for a set of pixels so that the linear relation is verified as closely as possible. However, the search for a good subset is performed by a greedy algorithm over randomly sampled subsets that is not suitable for large templates. Moreover, they loose the advantage of analytical expressions, and have to learn a series of linear predictors to ensure an acceptable precision.

We propose a method that avoids the combinatorial explosion and is therefore suitable for large templates, making the selection based on linearity useful in practice.

3. Template-Based Tracking

In order to correctly present our method, we quickly recall the foundations of template-based tracking, the IC and the ESM algorithms that will be used to test our approach. The interested reader can refer to [2] for more details.

Let $\mathcal{I}^*$ be an image containing the reference template of an object we aim to track, and let $\mathcal{I}$ be the current image of the observed scene. Let $\{\mathbf{p}_i^*\}$ be the set of coordinates of the projections in the reference image $\mathcal{I}^*$ of a set of 3D points lying on the object of interest. Tracking the reference template means finding the projective space automorphism $\mathbf{w}$ that minimizes:

$$\sum_i \left(\mathcal{I}\left(\mathbf{w}(\mathbf{p}_i^*)\right) - \mathcal{I}^*(\mathbf{p}_i^*)\right)^2 \tag{1}$$

The 3D motion of the object of interest can be extracted from $\mathbf{w}$. For simplicity reasons, we consider here only a planar object, and $\mathbf{w}$ will be based on a homography $\mathbf{G}$ parametrized over a vector $\mathbf{x}$. However, all the derivations in this paper are generic and our approach can be applied to more complex shapes. During tracking, an approximation $\widehat{\mathbf{G}}$ of the true automorphism $\overline{\mathbf{G}}$ is available, and the problem can be redefined as finding an incremental transformation $\mathbf{G}(\mathbf{x})$ such that the composition of $\widehat{\mathbf{G}}$ and $\mathbf{G}(\mathbf{x})$ gives the true automorphism $\overline{\mathbf{G}}$. Then, the problem consists in finding the optimal parameters $\widetilde{\mathbf{x}}$ that minimize:

$$\frac{1}{2}\|\mathbf{y}(\mathbf{x})\|^2 \tag{2}$$

where $\mathbf{y}(\mathbf{x})$ is the vector made of the image differences:

$$y_i(\mathbf{x}) = \mathcal{I}\left(\mathbf{w}(\widehat{\mathbf{G}}\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*)\right) - \mathcal{I}^*(\mathbf{p}_i^*)\,. \tag{3}$$

This problem is usually solved using an iterative minimization after a Taylor series approximation of the cost function (2). The IC algorithm of Baker and Matthews [2] considers a first-order approximation:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\ \mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^{\mathbf{2}}) \tag{4}$$

while the ESM algorithm of Benhimane and Malis [3] relies on a second-order approximation:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\ \mathbf{x} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{M}(\mathbf{0}, \mathbf{x})\ \mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^{\mathbf{3}}) \tag{5}$$

where $\mathbf{J}(\mathbf{x}) = \nabla_{\mathbf{x}}\mathbf{y}(\mathbf{x})$ is the Jacobian matrix of the vector $\mathbf{y}(\mathbf{x})$ with respect to the motion parameters $\mathbf{x}$, $\mathbf{M}(\mathbf{x}_1, \mathbf{x}_2) = \nabla_{\mathbf{x}_1}(\mathbf{J}(\mathbf{x}_1)\mathbf{x}_2)$ is based on the Hessian matrices, and $\mathbf{O}(\|\mathbf{x}\|^i)$ is a remainder of order i. However, both minimize the cost function iteratively by estimating $\widetilde{\mathbf{x}}$:

$$\widetilde{\mathbf{x}} = \left(\mathbf{J}_x^\top \mathbf{J}_x\right)^{-1} \mathbf{J}_x^\top \mathbf{y}(\mathbf{0}) \tag{6}$$

The two algorithms use different expressions for $\mathbf{J}_x$. In the IC algorithm, it is a constant matrix which we denote as $\mathbf{J}_{ic}$ and which is of the form: $\mathbf{J}_{ic} = \mathbf{J}_{\mathcal{I}^*}\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{G}}$ where $\mathbf{J}_{\mathcal{I}^*}$ depends on the gradient of the reference template and $\mathbf{J}_{\mathbf{w}}$ and $\mathbf{J}_{\mathbf{G}}$ depend on the warping function and the homography parameterization respectively. The ESM algorithm uses another expression which we denote as $\mathbf{J}_{esm} = \frac{1}{2}\left(\mathbf{J}_{\mathcal{I}} + \mathbf{J}_{\mathcal{I}^*}\right)\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{G}}$, where $\mathbf{J}_{\mathcal{I}}$ has to be computed at each minimization step. However, it is shown in [3] that, with a suitable parameterization, using this Jacobian gives the desirable properties of second-order methods with the same computation complexity as standard first-order approaches. What is important here for our approach is that the two methods rely on the iteration given by (6). Our selection algorithm will be the same for these two methods except for the expression of $\mathbf{J}_x$.

4. Determining Linear and Quadratic Subsets

4.1. Linear and Quadratic Subsets

A *linear subset* is a set of pixels on the template such that the approximation made by the IC algorithm becomes exact. Formally, a linear subset $\mathcal{E} = \{\mathbf{p}_i^*\}$ verifies:

$$\forall\mathbf{x}:\quad \mathbf{y}_{\mathcal{E}}(\mathbf{0}) = -\mathbf{J}_{ic,\mathcal{E}}\mathbf{x} \tag{7}$$

where $\mathbf{y}_{\mathcal{E}}$ is a vector of image differences and $\mathbf{J}_{ic,\mathcal{E}}$ is the IC Jacobian – both computed for the pixels of $\mathcal{E}$. As a result, the IC algorithm will converge toward the minimum ideally in one iteration when a linear subset is used. In practice, because of noise and because the relation between the image differences and the motion parameters is not exactly linear, we will see that it can sometimes take more than one iteration, but it still results in a decrease in the number of iterations and an increase in the convergence frequency.

Similarly, a *quadratic subset* is a subset of pixels such that the approximation made by the ESM algorithm becomes exact. Hence, a quadratic subset $\mathcal{E} = \{\mathbf{p}_i^*\}$ verifies $\forall\mathbf{x}:\quad \mathbf{y}_{\mathcal{E}}(\mathbf{0}) = -\mathbf{J}_{esm,\mathcal{E}}\mathbf{x}$. As will be shown, a similar gain in performance is then obtained with the ESM algorithm when using quadratic subsets.

4.2. Stability under Union Operation

It is easy to see that the set of linear subsets and the set of quadratic subsets are stable under the union operation. Let us consider two linear subsets $\mathcal{E}$ and $\mathcal{E}'$. Since we have:

$$\mathbf{y}_{\mathcal{E}\cup\mathcal{E}'}(\mathbf{0}) = \begin{pmatrix} \mathbf{y}_{\mathcal{E}}(\mathbf{0}) \\ \mathbf{y}_{\mathcal{E}'}(\mathbf{0}) \end{pmatrix} \text{ and} \tag{8}$$

$$\mathbf{J}_{\mathcal{E}\cup\mathcal{E}'}(\mathbf{0}) = \begin{pmatrix} \mathbf{J}_{ic,\mathcal{E}}(\mathbf{0}) \\ \mathbf{J}_{ic,\mathcal{E}'}(\mathbf{0}) \end{pmatrix}, \tag{9}$$

$\mathcal{E}\cup\mathcal{E}'$ verifies Equation (7), and is therefore a linear subset. Of course, a similar remark can be made for quadratic subsets. This property is extremely useful since it allows to group several linear or quadratic subsets to form a larger one, and our algorithm strongly relies on it.

4.3. Construction Algorithm

Here we present the algorithm we developed to create large linear or quadratic subsets.

In order to built the subsets, we consider all possible 3×3 pixel regions over the original template, and determine for each of them if they form a linear (or a quadratic) subset. Once this is done the linear subsets are merged to form the final subset. These 3×3 pixel regions have the advantage to be of (almost) the minimal size[1] that allows to determine a motion. Moreover, when such a compact region provides enough information to retrieve the object motion, it is very desirable to integrate it into the final subset. We can therefore build a very good final subset by testing Equation (7) on a relatively small number of small subsets.

Considering Equation (7), a naive criterion that could be used in practice would be to search for regions that minimize the differences between the computed image differences and the true ones. However, such a residual is not very meaningful for the problem at hand, and we prefer a criterion that favors the accuracy of the recovered motions. We first apply known random motions to the image. For each motion, and for each 3×3 region, we use Relation (6) to get an estimate of the motion from the pixels in the region. Each region has a counter initialized to 0 which is incremented each time the position of the region center is retrieved with an error lower than one pixel. At the end of this training phase, the regions with a counter value above

[1]The correct minimal size is 6 for a 3–D Euclidean motion of any 3D object (rotation and translation in the calibrated case) and 8 for a projective transformation of a planar object (homography in a non-calibrated case).

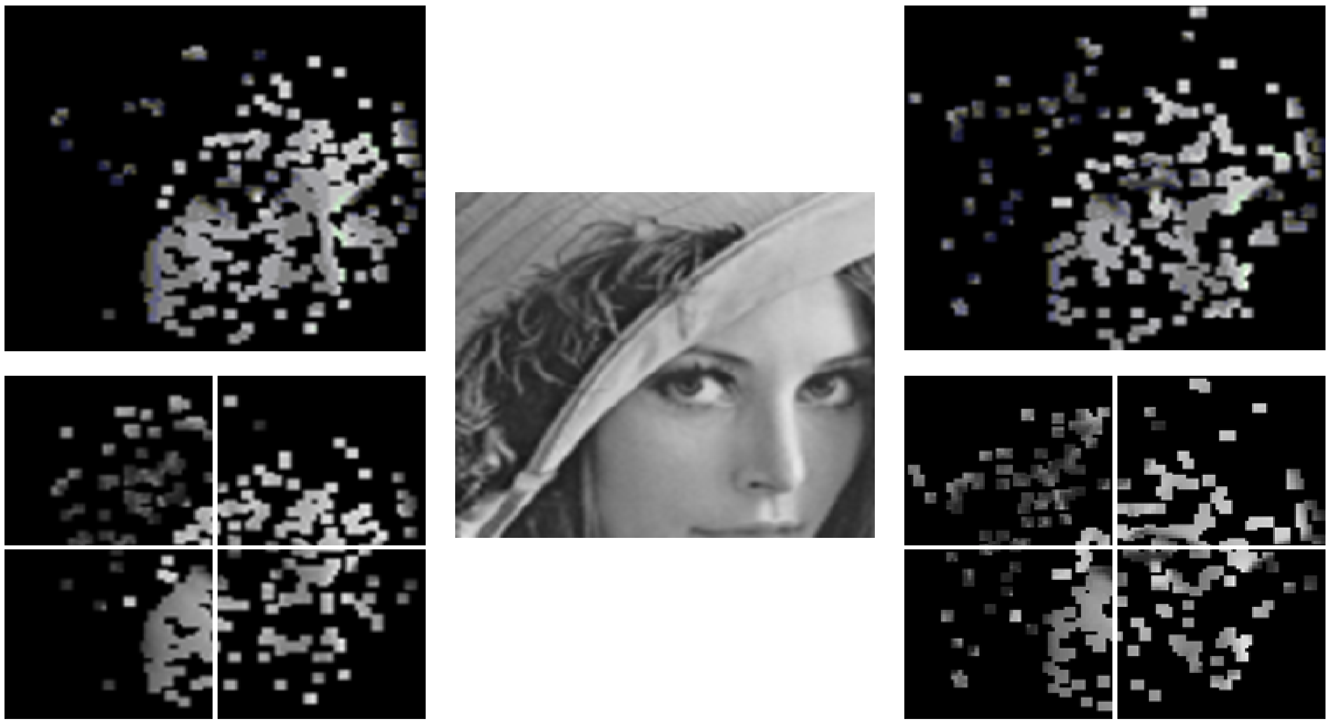

Figure 1. Linear and quadratic subsets learned for a sample template. The top left template shows the linear subsets used with the inverse compositional algorithm and the top right template shows the quadratic subsets used with the ESM algorithm. The bottom row shows the linear and quadratic subsets obtained when the subsets are constrained to be spread evenly over the template. Building such subsets takes less than 30 seconds on a standard computer.

a certain threshold τ are merged into a final subset $\mathcal{E}$ that will be used during tracking, while the other pixels are discarded. Figure 1 shows examples of linear and quadratic subsets. The threshold τ was set so that only 20% of the template pixels are contained in the final subset $\mathcal{E}$.

In order to be robust to partial occlusion that may occur during tracking, we also need to ensure a more or less uniform distribution of the subset over the template. Otherwise, it is possible that all the pixels concentrate in a small region, so that under partial occlusions the tracking will fail. We therefore subdivide the template with a virtual grid and require that there is a similar amount of pixels in each cell. The bottom row of Figure 1 shows the subsets built when using a 2×2 grid.

Our algorithm is linear w.r.t. the template size, and takes less than 30 seconds for 150×150 templates if we perform 100 motions on a 1.66 GHz Intel Core-Duo CPU with 1 GB Memory. We can see that the obtained subsets do not include corners and edges due to their high non-linearities and as expected uniform regions are not selected either since they do not add any information. When looking closely at the selected regions, one can actually realize that they often correspond to image parts where the intensities vary smoothly.

5. Simulation Results

We present simulations designed to validate our claims regarding the improvements in terms of accuracy and robustness. We applied random motions with increasing variances to a 100×100 template and then used different types of pixel subsets in both the IC and ESM algorithms to recover the motion parameters. We considered a test as converged if the template corners were retrieved with an RMS error lower than 1 pixel after 10 iterations, and plotted the convergence frequency against the motion variance measured on the template corners.

The results of the simulations are shown in Figure 2. Curves annotated with 'all pixels' were obtained when using the full template. The other curves were obtained using only 20% of the template pixels, selected with different methods: 'random subsets' refer to randomly selected pixels, 'regular subsets' to regularly sampled pixels, 'good-features-to-track' to pixels returned by the algorithm of Shi and Tomasi [10] and 'linear' and 'quadratic' to the subsets returned by the algorithm we propose. Note that all simulations have been conducted under the same conditions, and only the subsets were changed. In addition, neither preliminary image filtering nor multi-scale pyramid implementations have been used for this evaluation, to make the contribution of our approach clearer. For small motions all

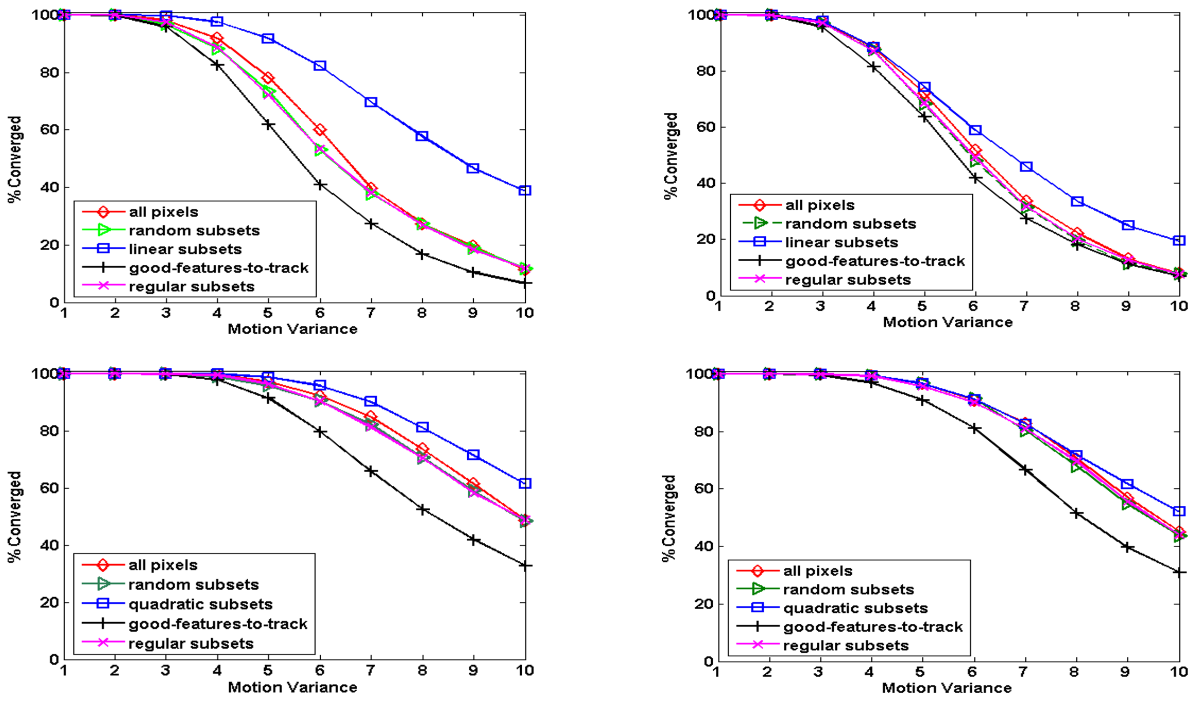

Figure 2. Convergence frequency vs. motion variances when using different types of subsets. First row: results obtained with the IC algorithm; Second row: results obtained with the ESM algorithm; Left column: without noise; Right column: in the presence of noise. Our method performs two times better with the IC algorithm than the other methods for large motions in noisy conditions. With the slower but more powerful ESM algorithm, the differences are smaller, but our approach still outperforms the other methods.

the approaches perform well and exhibit a high convergence frequency. If the motions are getting larger, we can see that the frequency of convergence decreases for all tested methods, however, our method always allows to achieve a high convergence frequency. In the case of the IC algorithm, the convergence frequency is four times the convergence frequency of the other methods under ideal noise-free conditions. When Gaussian noise with a standard deviation of 5 gray levels (over 255) is added, the convergence frequency obtained with our linear subset is still twice as big as the one of the other methods. Comparable results are obtained with the ESM algorithm. ESM is a little bit slower but more powerful than IC. Here, the differences are smaller, however, it is still clear that our approach outperforms the other methods. With the quadratic subsets, using the ESM algorithm, we achieve a convergence frequency of more than 60%. Apart from the accuracy and the robustness our approach also improves the speed of the tracking, since we only use a small number of pixels. If used for real-time tracking this directly results in a smaller interframe displacement, making the tracking easier.

6. Experimental Results

We performed numerous real-world experiments to test the performance of the proposed approach. Standard limitations to template-based tracking such as noise, partial occlusions, illumination changes, scale changes, oblique viewing angles, and fast motion were taken into consideration during this validation. These experiments confirm the fact that the subsets obtained with our algorithm can be used to robustly and accurately deal with real-world images.

In particular, Figure 3 shows some excerpts from one sample sequence[2] where the tracking output was used to perform an Augmented Reality task. Our algorithm was applied to a 155×168 pixels template of a book cover in order to extract a quadratic subset to be tracked by the ESM algorithm. The quadratic subset represents 32% of the size of the reference template. A maximum of 15 iterations were used during the optimization. The book was tracked correctly over the sequence, despite partial occlusions, changes in scale and oblique viewing angles. As can be assessed by watching the video, the augmentation is visually very stable, and only when significant parts of the template are covered there is a slight jittering, meaning that the pose was estimated very accurately.

The frame rate is between 30 fps and 60 fps on a 1.66 GHz Intel Core-Duo CPU with 1 GB Memory. The exact value of the frame rate depends on many factors including the size of the reference template, the number of scale levels

[2] See: http://campar.in.tum.de/files/publications/benhimane2007cvpr.video.avi

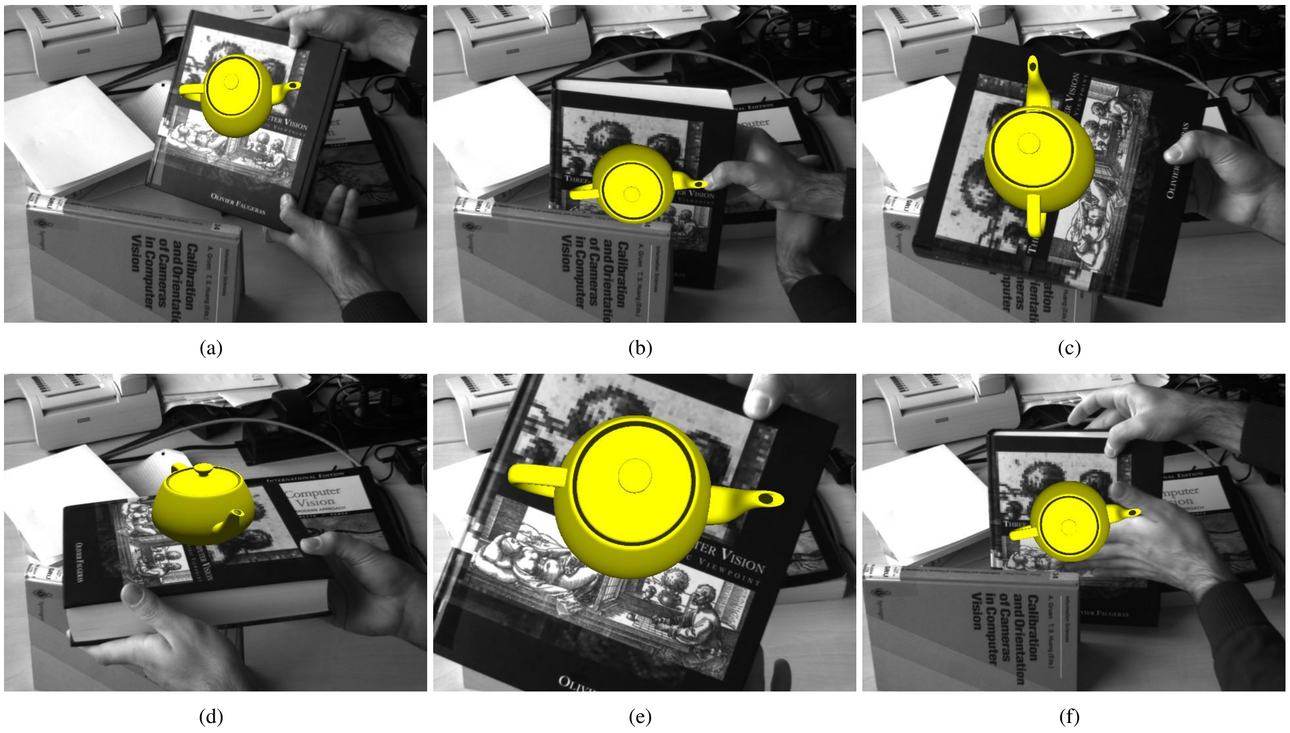

Figure 3. Tracking a quadratic subset for Augmented Reality. The book is tracked correctly despite partial occlusions, changes in scale and oblique viewing angles. The teapot is visually very stable, showing that the pose is estimated very accurately.

(if a multi-scale pyramid implementation is used) and the desired accuracy.

7. Conclusion

We proposed a simple algorithm based on a low-cost off-line learning step to determine which pixels can form a strong subset that verifies the linearization made during optimization. We validated our algorithm on synthetic and real data and showed that it outperforms other subset selection approaches for template-based tracking in terms of convergence frequency and in terms of efficiency. We also showed that it can be easily integrated into existing template-based tracking algorithms to improve their performance, making it very useful to people working with this approach.

References

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

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: a unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

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

[4] J. Buenaposada and L. Baumela. Real-time tracking and estimation of planar pose. In *Int. Conf. on Pattern Recognition*, pages 697–700, 2002.

[5] F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. In *Proc. of the Frame-Rate Vision Workshop (ICCV)*, 1999.

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

[7] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):996–1000, 2002.

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

[9] J. Matas, K. Zimmermann, T. Svoboda, and A. Hilton. Learning efficient linear predictors for motion estiamtion. Technical Report 5, Czech Technical University, 2006.

[10] J. Shi and C. Tomasi. Good features to track. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[11] H. Shum and R. Szeliski. Construction of panoramic image mosaics with global and local alignment. *IEEE Journal of Computer Vision*, 16(1):63–84, 2000.

[12] Z. Zivkovic and F. van der Heijden. Better features to track by estimating the tracking convergence region. In *Int. Conf. on Pattern Recognition*, pages 635–638, 2002.