Simultaneous Recognition and Homography Extraction of Local Patches with a Simple Linear Classifier

Stefan Hinterstoisser[1], Selim Benhimane[1], Vincent Lepetit[2], Pascal Fua[2], Nassir Navab[1]

[1]Department of Computer Science,
Technical University of Munich (TUM),
Boltzmannstrasse 3, 85748 Garching, Germany
[2]École Polytechnique Fédérale de Lausanne (EPFL),
Computer Vision Laboratory,CH-1015 Lausanne, Switzerland
{hinterst,benhiman,navab}@in.tum.de, {pascal.fua,vincent.lepetit}@epfl.ch

Abstract

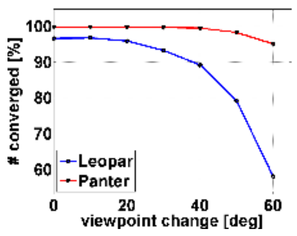
We show that the simultaneous estimation of keypoint identities and poses is more reliable than the two separate steps undertaken by previous approaches. A simple linear classifier coupled with linear predictors trained during a learning phase appears to be sufficient for this task. The retrieved poses are subpixel accurate due to the linear predictors. We demonstrate the advantages of our approach on real-time 3D object detection and tracking applications. Thanks to the high accuracy, one single keypoint is often enough to precisely estimate the object pose. As a result, we can deal in real-time with objects that are significantly less textured than the ones required by state-of-the-art methods.

1 Introduction

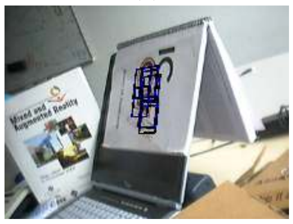
Retrieving the poses of keypoints in addition to matching them is an essential task in many applications such as vision-based robot localization [2], object recognition [10] or image retrieval [9] to constrain the problem at hand. It is usually done by decoupling the matching process from the keypoint pose estimation: The standard approach first uses some *ad hoc* affine region detectors and then relies on SIFT descriptors on the rectified regions to match the points. Since this process is conditioned by the detector, it is not possible to recover from incorrect detections. In our recent work [3], we proceeded the other way around: We first retrieved the keypoint identities using Ferns-based classifiers [8], and then we estimated their pose. While it improves the performances in terms of both speed and reliability when a training phase is possible, it is still vulnerable to errors made in one of the two steps. In this paper, we show that simultaneously matching the keypoints and estimating their poses makes the problem simpler, and is therefore a more reliable approach. As a result, a simple and fast linear classifier coupled with linear predictors is

(a)

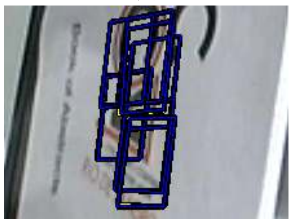
(b)



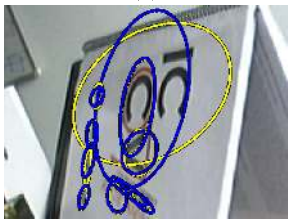
(c)



(d)



(e)



(f)

Figure 1: Overview of our approach. **(a)** Given a training image, we train a linear classifier to recognize patches and predict their transformation. **(b)** The results are very accurate and mostly exempt of outliers. Note we get the full perspective pose, and not only an affine transformation. **(c)** Comparison of the percentages of patches correctly retrieved by our previous approach and by the approach proposed in this paper. The new approach always performs better. **(d)** Compared to affine region detectors-based method, our approach is both more reliable and more accurate. On the images we superimpose the original patches warped by the ground truth homography (yellow) with the result of Panter (blue). **(e)** Even after zooming the errors of Panter are barely visible. **(f)** For comparison, we show the results obtained with the MSER detector.

sufficient to handle the problem. Since there is time to train the system for a large class of applications, we first build such a linear classifier and a set of linear predictors during a training phase. The classifier generates hypothesis of both the keypoint identities and the poses which the linear predictors refine. This enables us to retrieve the correct patch identity and pose with some simple correlation measurements. As Fig. 1 shows, it outperforms our previous approach, which has already been shown to be better than affine region detectors-based approaches. In our problem, the classes are made of a keypoint identity and a quantized pose. The linear classifier is able to provide a very small number of predictions for a given patch. We select the correct one with a few correlation tests taking into account the refined pose output of the linear predictors. The output consists of reliable matches and accurate pose estimates. The poses are represented by full perspective transformations that are very useful for object detection applications: Such a pose of a single keypoint is often enough to estimate the 3–D pose of the object the keypoint belongs to. As a result, we can handle very poorly textured objects. In the remainder of the paper, we first discuss related work on affine region detectors. Then, we describe our method, and compare it against state-of-the-art ones. Finally, we present an application of tracking-by-detection using our method.

2 Related Work

Affine region detectors are very attractive for many applications since they allow getting rid of most of the image warpings due to perspective transformations. Many different approaches have been proposed and [6] showed that the Hessian-Affine detector of Mikolajczyk and Schmid and the MSER detector of Matas et al. are the most reliable ones. In the case of the Hessian-Affine detector, the retrieved affine transformation is based on the image second moment matrix. It normalizes the region up to a rotation, which can then be estimated based on the dominant gradient orientation of the corrected patch. This implies using an *ad hoc* method, such as considering the peaks of the histogram of gradient orientations over the patch as in SIFT [5]. However, applying this heuristics on a warped patch tends to make it relatively unstable. In the case of the MSER detector, many different approaches exploiting the region shape are also possible [7], and a common approach is to compute the transformation from the region covariance matrix and solve for the remaining degree of freedom using local maximums of curvature and bitangents. After normalization, SIFT descriptors are computed in order to match the regions. The method we proposed in [3] performs the other way around: We first get the identity of the patch and then a coarse pose using a method close to [8]. Then, we apply a dedicated linear predictor to the patch in order to retrieve a fine perspective transformation. As [3] showed, this method performs better than previously proposed ones mainly because it can make use of a training phase. However, we give in this paper a new approach that outperforms this early method. We still use linear predictors to refine the poses because they perform well at this task. The main contribution is to show that a combination of a simple linear classifier coupled with linear predictors can simultaneously retrieve a keypoint identity and pose, and that it outperforms previous methods.

3 Proposed Approach

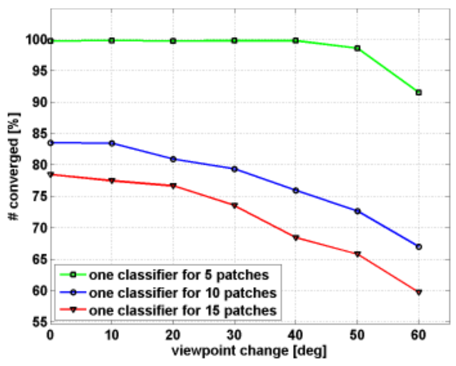
Given an image patch, we want to match it against a database of possible patches defined around keypoints in reference images, and accurately estimate its pose, represented by a homography. A linear classifier gives us a set of hypotheses on the identity of the corresponding patch and the pose. We then select the correct one by refining the pose using linear predictors and comparing the rectified patch and the predicted ones with normalized cross-correlation. The classification step requires the quantization of the pose space, and because a careful quantization method significantly improves the result, we also describe our quantization method here.

3.1 Linear Classification

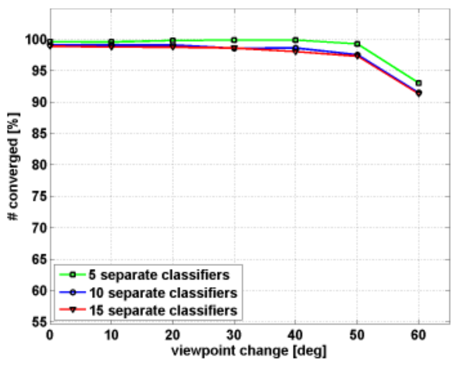
The classification step applies a matrix $\mathbf{A}$ to a vector $\mathbf{p}$ that contains the intensities of the patch we want to recognize:

$$\mathbf{A}^\top \mathbf{p} = \mathbf{y}\,. \tag{1}$$

The result $\mathbf{y}$ will give us a set of hypotheses. Each row $\mathbf{a}_{i,j}$ of $\mathbf{A}$ corresponds to a pair made of a patch of index i in the database, and a quantized homography of index j with $j = 1, ..., L$ where in practice $L \approx 1700$. Matrix $\mathbf{A}$ is built so that the corresponding values $\mathbf{y}_{i,j}$ are large when it is very likely that patch $\mathbf{p}$ corresponds to the i^{th} keypoint seen under



(a)



(b)

Figure 2: Comparison of the classification results obtained with the two strategies to estimate **A**, when the viewpoint angle increases. **(a)** shows the convergence rate for the first strategy with 5, 10 and 15 patches. **(b)** shows the convergence rate for the second strategy. The more patches one want to recover the more not considering the other keypoints makes the linear classifier work significantly better.

the j^{th} pose. When it is very unlikely, $\mathbf{y}_{i,j}$ has a large absolute value but is negative. To be robust to light changes, we normalize the patches by subtracting the mean and dividing by the standard deviation. For each possible corresponding keypoint of index i, we create a set of hypotheses Γ_i made of the pose indices j for which $\mathbf{y}_{i,j}$ are the largest. In practice, we retain 20 hypotheses for each possible patch. The next subsection explains how we select the correct hypotheses. We now detail how we compute **A**. **A** is computed during a learning phase using a training set generated from one or a few reference images of the keypoints to recognize. This training set is made of patches represented as $\mathbf{p}_{i,j,k}$ vectors, where i is the index of the corresponding keypoint, j the index of the pose, and k the index of the training patch. We tried two different strategies to compute **A**. As Fig. 2 shows, the second one works significantly better. In the first strategy, each row $\mathbf{a}_{i,j}^\top$ of **A** is estimated as the best solution in the least-squares sense of the optimization problem:

$$\forall i',\ j',\ k \quad \mathbf{a}_{i,j}^\top \cdot \mathbf{p}_{i',j',k} = \begin{cases} +1 & \text{if } i = i' \text{ and } j = j' \\ -1 & \text{otherwise} \end{cases} . \tag{2}$$

Here $\mathbf{a}_{i,j}$ contains the parameters of an hyperplane supposed to separate positive examples defined as the patches for a given keypoint i seen in a given pose j from negative examples defined as all the other examples for the same keypoint under other poses and also for the other keypoints. As Fig. 2(a) depicts, this problem seems too complex for a linear classifier, and we tried a simpler version in the second strategy. In the second strategy, the negative examples come only from the same keypoint, under different poses. $\mathbf{a}_{i,j}$ is now taken as the solution of the problem:

$$\forall j',\ k \quad \mathbf{a}_{i,j}^\top \cdot \mathbf{p}_{i,j',k} = \begin{cases} +1 & \text{if } j = j' \\ -1 & \text{otherwise} \end{cases} . \tag{3}$$

This approach works better because this problem is less constrained. The next step will select the correct hypothesis anyway. Because there are much less positive examples than negative ones, we give a weight $w = N - 1$ to the positive examples equations, where N is the number of possible poses, and a weight of 1 to the negative examples equations. The row vector $\mathbf{a}_{i,j}$ can therefore be computed as:

$$\mathbf{a}_{i,j} = \left(\mathbf{PWW}^\top\mathbf{P}^\top\right)^{-1}\mathbf{PWW}^\top\mathbf{y}\,, \tag{4}$$

where $\mathbf{P}$ is a matrix made of the patches column vectors $\mathbf{p}_{i',j',k}$, $\mathbf{y}$ is a row vector made of +1 and -1 values computed as in Eq. (3), and $\mathbf{W}$ is a diagonal matrix containing the equations weights. In practice, $\mathbf{y}$ and $\mathbf{P}$ are large and computing $\mathbf{a}_{i,j}$ directly using Eq. (4) can become heavy. We therefore decompose its computation into:

$$\begin{array}{rcl} \mathbf{a}_{i,j} & = & \left([\mathbf{P}_1 \ldots \mathbf{P}_L][w_1^2\mathbf{P}_1 \ldots w_L^2\mathbf{P}_L]^\top\right)^{-1}\left([w_1^2\mathbf{P}_1 \ldots w_L^2\mathbf{P}_L][y_1 \ldots y_L]^\top\right) \\ & = & \left(\sum_l w_l^2\mathbf{P}_l\mathbf{P}_l^\top\right)^{-1}\left(\sum_l y_l w_l^2\mathbf{P}_l\right)\,. \end{array} \tag{5}$$

Because the two terms of the products can be computed incrementally, this form can be computed without the full training set present in the computer memory.

3.2 Best Hypothesis Selection

For a given input patch $\mathbf{p}$, and for each keypoint in the database, the previous step gives us a list Γ of possible pose indices. We select for each keypoint i the best pose j that maximizes the normalized cross-correlation between $\mathbf{p}$ and the mean of the training examples of keypoint i under pose j. Since the patch intensities are already normalized, this can be written as looking for:

$$\forall i : \underset{j \in \Gamma_i}{\operatorname{argmax}}\ \mathbf{p}^\top.\overline{\mathbf{p}}_{i,j}\,, \tag{6}$$

where $\overline{\mathbf{p}}_{i,j}$ is computed as:

$$\overline{\mathbf{p}}_{i,j} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{p}_{i,j,k}\,. \tag{7}$$

Thus, for each keypoint i, we get the best quantized pose $\widehat{\mathbf{H}}_i$.

3.3 Final Keypoint Selection and Pose Extraction

For each best hypothesis consisting of keypoint i and the best quantized homography $\widehat{\mathbf{H}}_i$, we use the hyperplane approximation of [4] to obtain an estimate of the corrective homography parameters $\mathbf{x}_i$ using the following equation:

$$\mathbf{x}_i = \mathbf{B}_i\left(\mathbf{p}(\widehat{\mathbf{H}}_i) - \mathbf{p}_i^*\right)\,, \tag{8}$$

- where $\mathbf{B}_i$ is the matrix of our linear predictor that depends on the patch identity i;
- $\mathbf{p}(\widehat{\mathbf{H}}_i)$ is a vector that contains the intensities of the original patch $\mathbf{p}$ warped by the current estimate $\widehat{\mathbf{H}}_i$ of the transformation.
- $\mathbf{p}_i^*$ is a vector that contains the intensity values of the reference patch, which is the image patch centered on the keypoint i in a reference image.

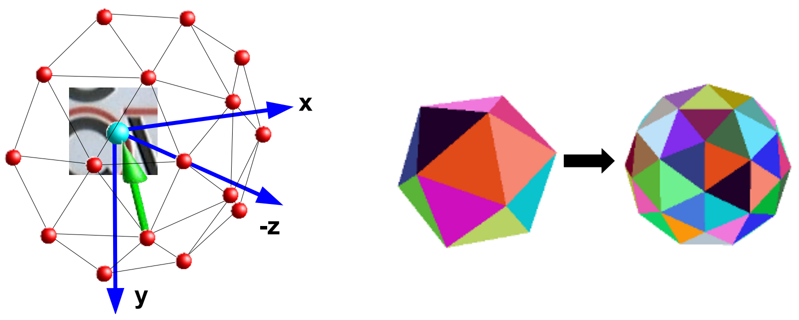


Figure 3: Homography space sampling using almost regular polyhedrons. Left: The red dots represent the vertices of an (almost) regular polyhedron generated by our recursive decomposition and centered on a planar patch. The sampled directions of views are given by vectors starting from one of the vertices and pointing toward the patch center. The green arrow is an example of such a vector. Right: The initial icosahedron and the result of the first triangle substitution.

Based on the algorithm proposed in [4], iteratively refining $\widehat{\mathbf{H}}_i$ allows us to obtain the refined homography $\widehat{\mathbf{H}}_{i,\text{final}}$. Finally, thanks to the high accuracy of the retrieved transformation, we can select the correct pair of keypoint i and pose $\widehat{\mathbf{H}}_{i,\text{final}}$ based on the normalized cross-correlation between the reference patch $\mathbf{p}_i^*$ and the warped patch. The selection is done by

$$\underset{i}{\operatorname{argmax}}\ \mathbf{p}_i^{*\top} \cdot \mathbf{p}(\widehat{\mathbf{H}}_{i,\text{final}}) \,, \tag{9}$$

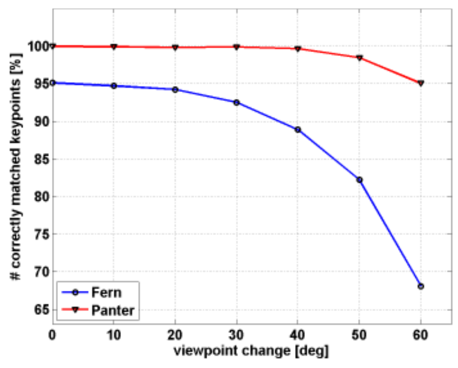
Furthermore, we also use a threshold $\tau_{\text{NCC}} = 0.9$ to remove wrong matches. Thus, each patch $\mathbf{p}(\widehat{\mathbf{H}}_{i,\text{final}})$ that gives the maximum similarity score, which exceeds τ_{NCC} at the same time, yields an accepted match.

3.4 Homography Space Quantization

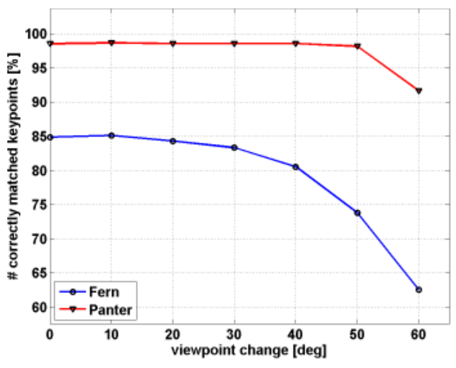
We still have to explain how we quantize the homography space. This is done based on the formula:

$$\mathbf{H} = \mathbf{K}\left(\Delta\mathbf{R} + \frac{\delta\mathbf{t}\cdot\mathbf{n}^\top}{d}\right)\mathbf{K}^{-1} \,, \tag{10}$$

which is the expression of the homography $\mathbf{H}$ relating two views of a 3–D plane, where $\mathbf{K}$ is the matrix of the camera internal parameters, $[n^\top, d]^\top$ the parameters of the plane in the first view, and $\Delta\mathbf{R}$ and $\delta\mathbf{t}$ the camera displacement between the two views. For simplification, we assume we have a frontal view of the reference patches. We first tried discretizing the motion between the views by simply discretizing the rotation angles around the three axes. However, for the linear predictors of Eq. (8) to work well, they must be initialized as close as possible to the correct solution, and we provide another solution that improves their convergence rates. As shown by the left image of Fig. 3, we found that the vertices of (almost) regular polyhedrons provide a more regular sampling that is useful to discretize the angle the second view in Eq. (10) makes with the patch plane. Unfortunately, there exists only a few convex regular polyhedrons - the Platonic solids - with the icosahedron the one with the largest number of vertices, 12. As the right image of Fig. 3 illustrates, we



(a)



(b)

Figure 4: Comparison with Ferns [8] on two different data sets **(a)**: the stop sign of Fig. 1 and **(b)**: the ICCV logo of Fig. 5. In each graph, we plot the percentage of correctly matched keypoints against the view angle. Panter is always performing better than Ferns.

obtain a finer sampling by recursively substituting each triangle into four almost equilateral triangles. The vertices of the created polyhedron give us the two out-of-plane rotation angles for the sampled pose, that is around the x- and y-axes of Fig. 3. We discretize the in-plane rotation angle to cover the $360°$ range with $10°$ steps. We still have to sample the scale changes. For that, we simply fix the translation and the plane equation but multiply the homography matrix obtained with Eq. (10) by a scaling matrix to cover three different scale levels $1/2$, 1, 2.

4 Experimental Validation

In this section, we compare our method that we call Panter against Ferns, and our previous method named Leopar. We don't compare here against affine region detectors but Leopar was shown to outperform them in [3] and our new method improves upon Leopar. We also present an application to tracking-by-detection and discuss computation times.

4.1 Comparison with Ferns

In Fig. 4, we compare our method that we call Panter against Ferns [8] since they are also based on a learning phase. The two graphs plot the percentage of correctly matched keypoints against the viewpoint angle with the object plane, for different databases. The two methods were trained using the same data set. For the experimental results we randomly generated 1000 test images for each viewpoint adding artificial noise and illumination changes. Panter always outperforms the Ferns, particularly for large viewpoint angles.

4.2 Comparison with Leopar

Fig. 5 shows the convergence results on three different data sets of the Leopar method [3] and Panter. Here again, Panter outperforms Leopar in particular for large viewpoint angles. We compared the percentage of keypoints for which the refinement given by Eq. (8)

correctly converged, when using the complete method, and when initializing the pose extraction and refinement with the correct keypoint position and identity. This is helpful to show the contributions of Panter with respect to different aspects. On the one hand side we show the improvements of Panter compared to Leopar considering the whole approach. On the other hand side we show only the improved ability of Panter to extract and to refine the right pose given the correct classification of the keypoint identity and its exact position. Improvements only due to the better pose extraction technique can be seen from the gap between the 'rectification-only' curves. Improvements due to the simultaneous classification and the selection among several hypotheses can be seen between the curves for the complete recovery of Panter and Leopar. The performances of Panter for the complete method are almost as good as when the correct keypoint identity and position are provided. Fig. 6 illustrates these comparison results. In practice, Panter always recovers more keypoints in particular under drastic pose changes.

4.3 Application and Computation Times

In Fig. 7, we apply our method to object detection and pose estimation using a low-quality camera. the method is robust and accurate even in presence of drastic perspective changes, light changes, blur, occlusion, and deformations. We initialized the template matching ESM algorithm [1] with the estimated pose for one of the keypoints to estimate the stop sign pose. Our current implementation runs at about 10 frames per second using 7 keypoints in the database and 50 candidate keypoints in the input image, on a standard notebook with an Intel Centrino Processor Core2Duo with 2.4GHz and 3GB RAM. Due to its robustness, it is enough to detect the target object and to estimate its pose reliably.

5 Conclusion

We showed in this paper that the simultaneous estimation of keypoint identities and poses is more reliable than the two separate steps undertaken consecutively and that simple linear classifiers and linear predictors are able to solve this task. This results in a highly robust method that is additionally fast, subpixel accurate due to its converging abilities. We demonstrated our approach on simple tracking-by-detection applications and showed in exhaustive experiments the improved performance compared to previous state-of-the-art methods. Thanks to the simplicity of our new approach, many other applications as robot localization, object recognition or image retrieval can benefit from its superior performance.

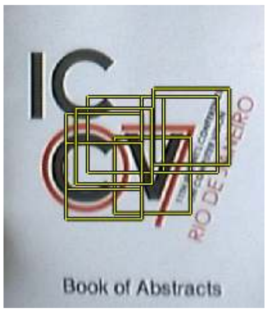
References

[1] S. Benhimane and E. Malis. Homography-based 2d visual tracking and servoing. *IJRR*, 26(7):661–676, July 2007.

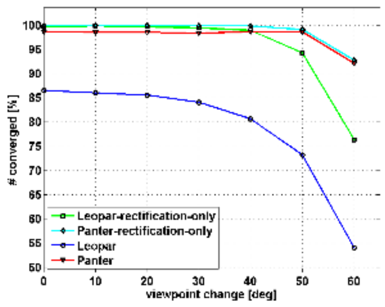
[2] T. Goedeme, T. Tuytelaars, and L. Van Gool. Fast wide baseline matching for visual navigation. In *CVPR*, 2004.

[3] S. Hinterstoisser, S. Benhimane, N. Navab, P. Fua, and V. Lepetit. Online learning of patch perspective rectification for efficient object detection. In *CVPR*, 2008.

[4] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *PAMI*, 24(7):996–100, 2002.

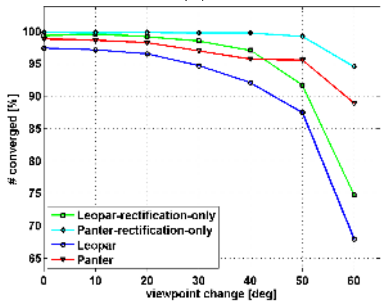


(a)

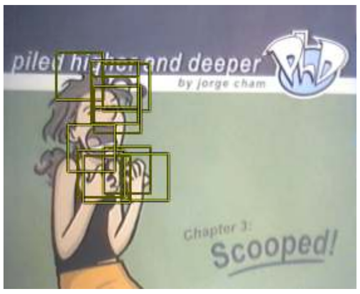


(b)

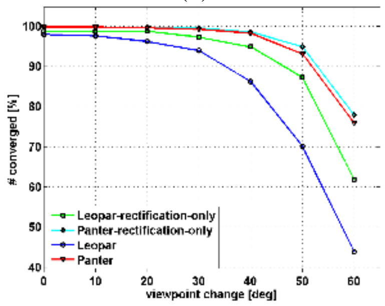
(c)



(d)



(e)



(f)

Figure 5: Comparison with Leopar [3]. Left: Test objects and the used patches. Right: Corresponding graphs displaying the percentage of patches for which the refinement step converged correctly. The '-rectification-only' curves correspond to the results obtained when using the correct keypoint position and identity to initialize the pose extraction and refinement. Panter always outperforms Leopar.

[5] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 20(2):91–110, 2004.

[6] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 65(1):43–72, 2005.

[7] Š. Obdržálek and J. Matas. *Toward Category-Level Object Recognition*, chapter 2, pages 85–108. J. Ponce, M. Herbert, C. Schmid, and A. Zisserman (Editors). Springer-Verlag, Berlin Heidelberg, Germany, 2006.

[8] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In

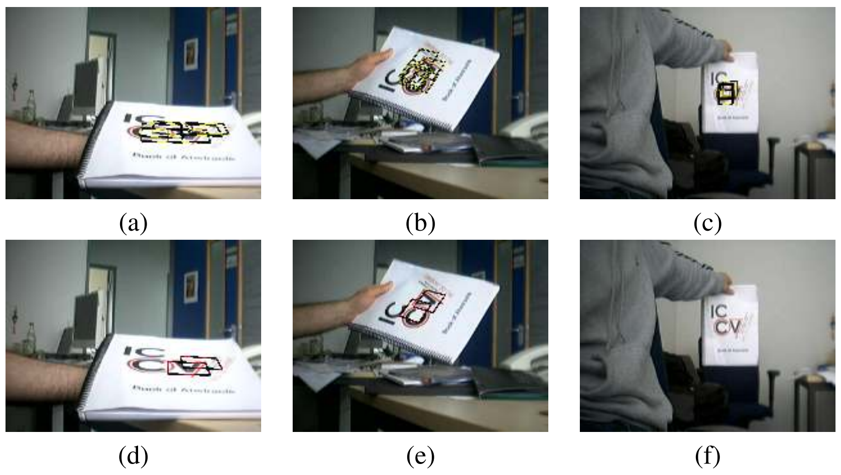


Figure 6: Comparison between Panter (upper row) and Leopar (lower row). Panter always correctly retrieves much more patches than Leopar. In particular, Leopar was not able to recognize any patch in the third image.

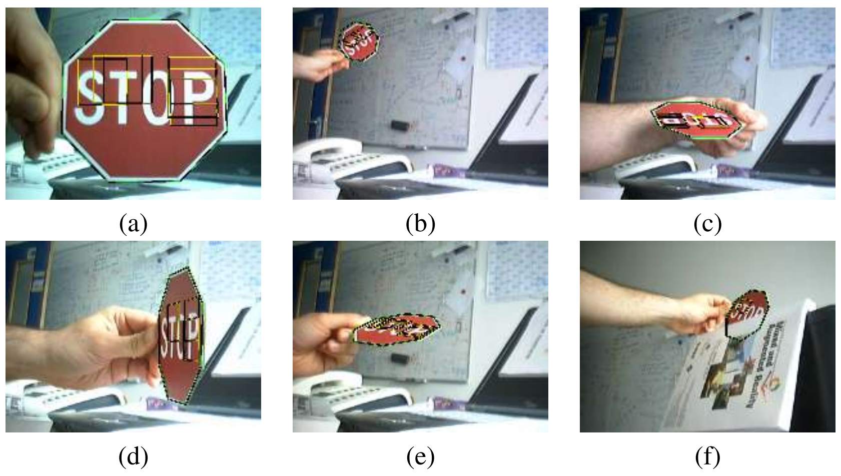


Figure 7: Some frames of a Tracking-by-Detection sequence shot with a low-quality camera. The stop sign pose is retrieved in each frame independently with 10-15 HZ at different scales **(a)**,**(b)**, under highly perspective transformations **(c)**-**(e)** and under occlusion **(f)**. The yellow rectangles display the recognized patches and the green octagon the refined result by initializing the ESM algorithm [1] with the pose estimate of one of the recognized patches.

CVPR, June 2007.

[9] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.

[10] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *IJCV*, 66(3):231–259, 2006.