Online Learning of Patch Perspective Rectification for Efficient Object Detection

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

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

Abstract

For a large class of applications, there is time to train the system. In this paper, we propose a learning-based approach to patch perspective rectification, and show that it is both faster and more reliable than state-of-the-art ad hoc *affine region detection methods.*

Our method performs in three steps. First, a classifier provides for every keypoint not only its identity, but also a first estimate of its transformation. This estimate allows carrying out, in the second step, an accurate perspective rectification using linear predictors. We show that both the classifier and the linear predictors can be trained online, which makes the approach convenient. The last step is a fast verification –made possible by the accurate perspective rectification– of the patch identity and its sub-pixel precision position estimation. We test our approach on real-time 3D object detection and tracking applications. We show that we can use the estimated perspective rectifications to determine the object pose and as a result, we need much fewer correspondences to obtain a precise pose estimation.

1. Introduction

Recently, it has been shown that taking advantage of a training phase, when possible, greatly improves the speed and the rate of keypoint recognition tasks [11]. However, the approach proposed in [11] does not provide any local image transformation. This is a clear disadvantage compared to affine region detectors [8] since these transformations are very useful in many applications such as robot localization [5], object recognition [14] or image retrieval [13] to constrain the problem at hand.

In this paper, we propose a fast learning-based matching method that yields accurate estimates of the image transformations around the keypoints. As shown in Fig. 1, training avoids *ad hoc* estimation of the transformation and gives us much better results than standard affine region detectors. In addition, our method is much faster and can be trained incrementally, which is a highly desirable feature: This allows for an arbitrary large number of training samples, as well as new incoming images to be exploited in order to continuously improve the method's accuracy and robustness.

Our method is not restricted to specific keypoint or affine region detectors. Given a set of patches around the keypoints or affine region centroids, it proceeds in three steps:

- We first apply an extended version of the Ferns classifier [11]: for each patch, our classifier provides not only the patch identity, but also an estimation of its orientation. Every keypoint in our database has several classes and each class covers its possible appearances for a restricted range of poses.

- The second step applies to the patch a linear predictor, or linear regressor, similar to the one described in [6]. The linear predictor is initialized with the orientation provided by our classifier, allows us to obtain the accurate full perspective transformation of the considered patches. Each class has a linear predictor, and we show that these linear predictors can be trained using an efficient and incremental framework.

- Finally, the accurate full perspective transformations recovered allow checking for the patch identities and serve as a constraint for further pose estimation.

For each patch, the whole process requires few hundreds of image accesses, additions and multiplications. It is therefore very fast, while being more robust and accurate than state-of-the-art approaches.

In the remainder of the paper, we first discuss related work on affine region detectors and linear predictors. Then,

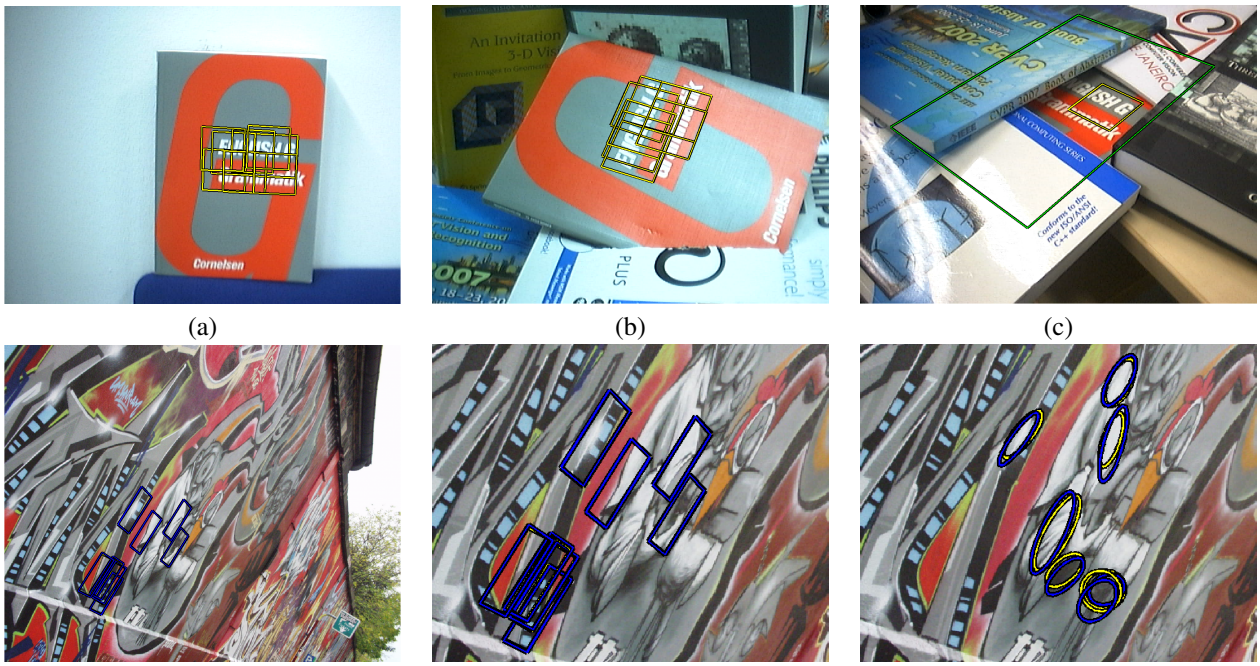


Figure 1. The advantages of learning for patch recognition and pose estimation. **(a)** Given a training images or a video sequence, our method learns to recognize patches and in the same time to estimate their transformation. **(b)** The results are very accurate and mostly exempt of outliers. Note we get the full perspective pose, and not only an affine transformation. **(c)** Hence a single patch is often sufficient to detect objects and estimate their pose very accurately. **(d)** To illustrate the accuracy, we use the 'Graffiti 1' image to train our method and detect patches in the 'Graffiti 6' image. We then superimpose the retrieved transformations with the original patches warped by the ground truth homography. **(e)** Even after zooming, the errors are still barely visible. **(f)** By contrast, the standard methods retrieve comparatively inaccurate transformations, which are limited to the affine transformation group.

we describe our method, and compare it against state-of-the-art ones on standard benchmark images. 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 [8] 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 [7]. 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 [10], 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.

Our method performs the other way around: We first get the identity of the patch and its orientation using an extension of the fast classifier [11]. Then, we apply a dedicated linear predictor to the patch in order to retrieve its perspective transformation. This avoids warping the patches, which in practice, often produces artifacts and as our comparisons show, this approach performs better than previously proposed methods.

There is also another fundamental difference between our approach and affine region detectors. Affine region detectors only provide a canonical transformation, while our method is able to provide the homographic transformation that actually corresponds to the patch perspective orientation if a 3D model is available. When no 3D model is available, we can still provide the transformation with respect to a reference image.

This is done assuming a local linear relation between image differences and the geometric transformation. As in some template matching approaches, this relation can be estimated from the analytical derivation of the Jacobian matrix of some correlation function [2] or using a second-order approximation [3]. We use another approach in which we learn this relation as a linear predictor from a set of couples made of image differences and the corresponding corrective motion [4, 6]. This learning-based method is faster and has a larger convergence region. Its drawback is that it is very sensitive to occlusions. This is not a problem in our case since we apply it only on local regions. While in the original formulation of [4] and [6], the linear transformation is computed once and for all, we show that it can be incrementally estimated without the need of any additional approximation. Since the region classifier can already be estimated incrementally, the full method can be trained online with an arbitrary large number of images.

Another work related to this paper is [9], which exploits the perspective transformation of patches centered on landmarks in a SLAM application. However, it is still very depending on the tracking prediction to match the landmarks and to retrieve their transformations, while we do not need any prior on the pose. Moreover, in [9], these transformations are recovered using a Jacobian-based method while, in our case, a linear predictor can be trained very efficiently for faster convergence.

3. Proposed Approach

Given an image patch, we want to match it against a database of possible patches defined around keypoints and to estimate its perspective orientation. Our approach performs in three steps. First, we use an extended version of the Ferns classifier [11]. The extended version we propose allows retrieving not only the corresponding patch in the database but also to provide an estimate of its orientation. This estimate serves in the second step as an initialization for a linear predictor that provides an accurate full perspective transformation. Once this rectification is found, the third step consists in checking the patch identity by simple correlation.

3.1. Matching and Initializing

First, similarly to [11], we retrieve the keypoint to which the patch corresponds using a classifier trained with patches centered on the keypoints of the database and seen under different viewing conditions as in Fig. 3(a). Formally, for a given patch $\mathbf{p}$, this gives us:

$$\widehat{id} = \underset{id}{\operatorname{argmax}}\, P(Id = id \mid \mathbf{p})\,, \qquad (1)$$

where Id is a random variable representing the identity of the patch $\mathbf{p}$. The identity is simply the index of the corresponding keypoint in the database. The classifier represents the patch $\mathbf{p}$ as a set of simple image binary features that are grouped into subsets, and Id is estimated following a semi-Naive Bayesian scheme that assumes the feature subsets independent. For more details, refer to [11]. This classifier is usually able to retrieve the patch identity Id under scale, perspective and lighting variations.

Once Id is estimated, our objective is then to get an estimate of the transformation of the patch around the keypoint. We tried several approaches to discretize the transformation, and the best results were obtained with the parametrization described in Fig. 2. The transformation has 4 degrees of freedom that corresponds to the trigonometric

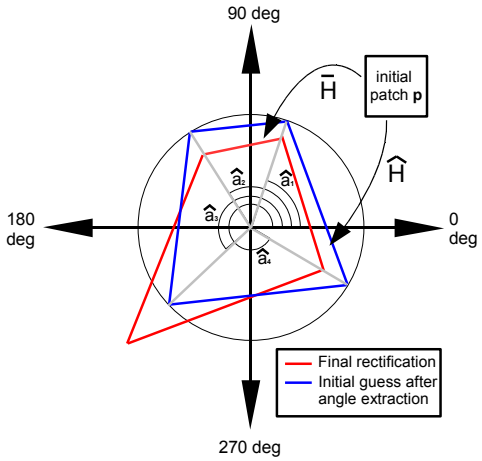
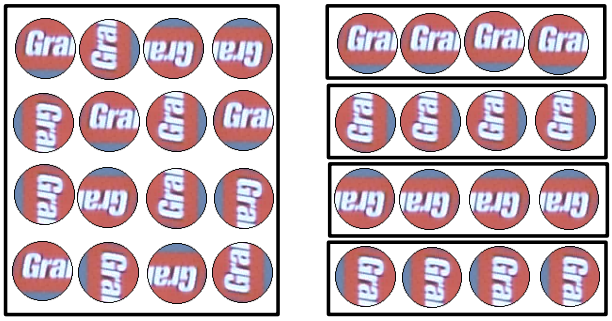


Figure 2. A first estimate of the patch transformation is obtained using a classifier that provides the values of the angles $\mathbf{a}_i$ defined as the angles between the lines that go through the patch center and each of the four corners.



(a) (b)

Figure 3. Examples of patches used for classification. (a) To estimate the keypoint identity, patches from the same keypoint are grouped in a single class. (b) To estimate the patch transformation, several classes for different transformations are created for each keypoint in the database.

angles $\mathbf{a}_i$ defined between the horizontal axis and the semi-lines going from the patch center and passing through the patch corners. The angles are discretized into 36 values and are estimated using a classification:

$$\widehat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmax}}\, P(A = \mathbf{a} \mid Id = id, \mathbf{p}) \,, \tag{2}$$

where $\widehat{\mathbf{a}} = (\widehat{\mathbf{a}}_1,\ \widehat{\mathbf{a}}_2,\ \widehat{\mathbf{a}}_3,\ \widehat{\mathbf{a}}_4)$ are the angles to be estimated. Eq. (2) is solved using a classifier specific to the patch identity Id. There is one classifier for each keypoint in the database, each of them trained with registered patches of the corresponding keypoint as shown in Fig. 3(b). The four angles $\widehat{\mathbf{a}}$ allow a first estimate of the patch transformation. This transformation is refined with the next step described below.

3.2. Refining

In this section, we explain how we compute the full perspective transformation of the patches. It is based on linear regression, and we show how the linear predictors can be computed incrementally.

3.2.1 Linear Prediction

As shown in Fig. 4, we model the full perspective transformation of the patches $\mathbf{p}$ by a homography defined with respect to a reference frame, but the derivations below are independent of this choice. The angles $\widehat{\mathbf{a}}$ computed in the first stage give an initial homography estimate $\widehat{\mathbf{H}}$ of the true homography $\overline{\mathbf{H}}$. We use the hyperplane approximation of [6] and obtain an estimate of a corrective homography parameters $\widetilde{\mathbf{x}}$ using the following equation:

$$\widetilde{\mathbf{x}} = \mathbf{A}\left(\mathbf{p}(\widehat{\mathbf{H}}) - \mathbf{p}^*\right) \,, \tag{3}$$

where

- $\mathbf{A}$ is the matrix of our linear predictor, and depends on the retrieved patch identity $\widehat{id}$;
- $\mathbf{p}(\widehat{\mathbf{H}})$ is a vector that contains the intensities of the original patch $\mathbf{p}$ warped by the current estimate $\widehat{\mathbf{H}}$ of the transformation. Note that we do not actually warp the patch, we simply warp back a set of locations in the patch;
- $\mathbf{p}^*$ is a vector that contains the intensity values of the reference patch, which is the image patch centered on the keypoint $\widehat{id}$ in a reference image.

This equation gives us the parameters $\widetilde{\mathbf{x}}$ of the incremental homography that updates $\widehat{\mathbf{H}}$ to produce a better estimate of true homography $\overline{\mathbf{H}}$:

$$\widehat{\mathbf{H}} \longleftarrow \widehat{\mathbf{H}} \circ \mathbf{H}(\widetilde{\mathbf{x}}) \,. \tag{4}$$

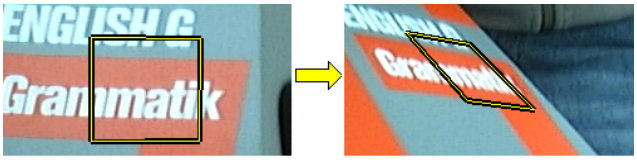


Figure 4. Original patch with its perspective warped counterpart extracted by our method.

For better accuracy, we iterate Eqs. (3) and (4) using a series of linear predictors $\mathbf{A}$, each matrix being dedicated to smaller errors than its predecessor: Applying successively these matrices remains fast and gives a more accurate estimate than with a single level. In practice, our vectors $\mathbf{p}(\widehat{\mathbf{H}})$ and $\mathbf{p}^*$ contain the intensities at locations sampled on a regular grid of 13×13 over image patches of size 75×75 pixels, and we normalize them to be robust to light changes. We parametrize the homographies by the 2D locations of the patch four corners. This parametrization is proved to be more stable than others in [1].

3.2.2 Incrementally Learning the Linear Predictor

The linear predictor $\mathbf{A}$ in Eq. (3) can be computed as the pseudo-inverse of the analytically derived Jacobian matrix of a correlation measure [2, 3]. However, the hyperplane approximation [6] computed from several examples yields a much larger region of convergence. The matrix $\mathbf{A}$ is then computed as:

$$\mathbf{A} = \mathbf{X}\mathbf{D}^\top \left(\mathbf{D}\mathbf{D}^\top\right)^{-1} \,, \tag{5}$$

where $\mathbf{X}$ is a matrix made of $\mathbf{x}_i$ column vectors, and $\mathbf{D}$ a matrix made of column vectors $\mathbf{d}_i$. Each vector $\mathbf{d}_i$ is the difference between the reference patch $\mathbf{p}^*$ and the same patch after warping by the homography parametrized by $\mathbf{x}_i$: $\mathbf{d}_i = \mathbf{p}(\mathbf{H}(\mathbf{x}_i)) - \mathbf{p}^*$.

Eq. (5) requires all the couples $(\mathbf{x}_i, \mathbf{d}_i)$ to be simultaneously available. If it is applied directly, this prevents incremental learning but this can be fixed. Suppose that the matrix $\mathbf{A} = \mathbf{A}_n$ is already computed for n examples, and then a new example $(\mathbf{x}_{n+1}, \mathbf{d}_{n+1})$ becomes available. We want to update the matrix $\mathbf{A}$ into the matrix $\mathbf{A}_{n+1}$ that takes into account all the $n+1$ examples. Let us introduce the matrices $\mathbf{Y}_n = \mathbf{X}_n\mathbf{D}_n^\top$ and $\mathbf{Z}_n = \mathbf{D}_n\mathbf{D}_n^\top$. We then have:

$$\begin{aligned}
\mathbf{A}_{n+1} &= \mathbf{Y}_{n+1}\mathbf{Z}_{n+1}^{-1} \\
&= \mathbf{X}_{n+1}\mathbf{D}_{n+1}^\top \left(\mathbf{D}_{n+1}\mathbf{D}_{n+1}^\top\right)^{-1} \\
&= [\mathbf{X}_n|\mathbf{x}_{n+1}][\mathbf{D}_n|\mathbf{d}_{n+1}]^\top \left([\mathbf{D}_n|\mathbf{d}_{n+1}][\mathbf{D}_n|\mathbf{d}_{n+1}]^\top\right)^{-1} \\
&= \left(\mathbf{X}_n\mathbf{D}_n^\top + \mathbf{x}_{n+1}\mathbf{d}_{n+1}^\top\right)\left(\mathbf{D}_n\mathbf{D}_n^\top + \mathbf{d}_{n+1}\mathbf{d}_{n+1}^\top\right)^{-1} \\
&= \left(\mathbf{Y}_n + \mathbf{x}_{n+1}\mathbf{d}_{n+1}^\top\right)\left(\mathbf{Z}_n + \mathbf{d}_{n+1}\mathbf{d}_{n+1}^\top\right)^{-1}
\end{aligned} \tag{6}$$

where $\mathbf{x}_{n+1}$ and $\mathbf{d}_{n+1}$ are concatenated to $\mathbf{X}_n$ and $\mathbf{D}_n$ respectively to form $\mathbf{X}_{n+1}$ and $\mathbf{D}_{n+1}$. Thus, by only storing the *constant size* matrices $\mathbf{Y}_n$ and $\mathbf{Z}_n$ and updating them as:

$$\mathbf{Y}_{n+1} \longleftarrow \mathbf{Y}_n + \mathbf{x}_{n+1}\mathbf{d}_{n+1}^\top \quad (7)$$

$$\mathbf{Z}_{n+1} \longleftarrow \mathbf{Z}_n + \mathbf{d}_{n+1}\mathbf{d}_{n+1}^\top , \quad (8)$$

it becomes possible to incrementally learn the linear predictor without storing the previous examples, and allows for an arbitrary large number of examples.

Since the computation of $\mathbf{A}$ has to be done for many locations in each incoming image and $\mathbf{Z}_n$ is a large matrix in practice, we need to go one step further in order to avoid the computation of $\mathbf{Z}_n^{-1}$ at every iteration. We apply the Sherman-Morrison formula to $\mathbf{Z}_{n+1}^{-1}$ and we get:

$$\begin{aligned} \mathbf{Z}_{n+1}^{-1} &= \left(\mathbf{Z}_n + \mathbf{d}_{n+1}\mathbf{d}_{n+1}^\top\right)^{-1} \\ &= \mathbf{Z}_n^{-1} - \frac{\mathbf{Z}_n^{-1}\mathbf{d}_{n+1}\mathbf{d}_{n+1}^\top\mathbf{Z}_n^{-1}}{1 + \mathbf{d}_{n+1}^\top\mathbf{Z}_n^{-1}\mathbf{d}_{n+1}} . \end{aligned} \quad (9)$$

Therefore, if we store $\mathbf{Z}_n^{-1}$ instead of $\mathbf{Z}_n$ itself, and update it using Eq. (9), no matrix inversion is required anymore, and the computation of matrix $\mathbf{A}_{n+1}$ becomes very fast. In our implementation, this drops the time to update the linear predictors by a factor 5, from 300ms to 60ms.

3.3. Correlation-based Verification

The final step of our algorithm checks the identity of patches by correlation. Thanks to the high accuracy of the retrieved transformation, we are able to reject matches based on the Normalized Cross-Correlation between the reference patch $\mathbf{p}^*$ and the warped patch. Since the patch intensities are normalized, our test can be written as

$$\mathbf{p}(\widehat{\mathbf{H}}_{\text{final}})^\top \cdot \mathbf{p}^* > \tau_{\text{NCC}} , \quad (10)$$

where $\widehat{\mathbf{H}}_{\text{final}}$ is the final transformation obtained with the linear predictor. In practice, we use a threshold $\tau_{\text{NCC}} = 0.9$. Each patch that passes this test yields an accepted match.

3.4. Training Framework

Our algorithm can be trained using either a small set of training images or a video sequence. In the first case, we synthesize images by warping the original patches with random homographies and adding noise to train the classifiers and the linear predictors. If a video sequence and a 3D model are available, we proceed as proposed in [12]: The first image is registered manually and approximately. It is used to partially train the classifiers and linear predictors. Assuming a small interframe displacement in the training sequence, this is enough to recognize feature points in the next image, and register it. The process is iterated to process the whole sequence as shown in Fig. 5.

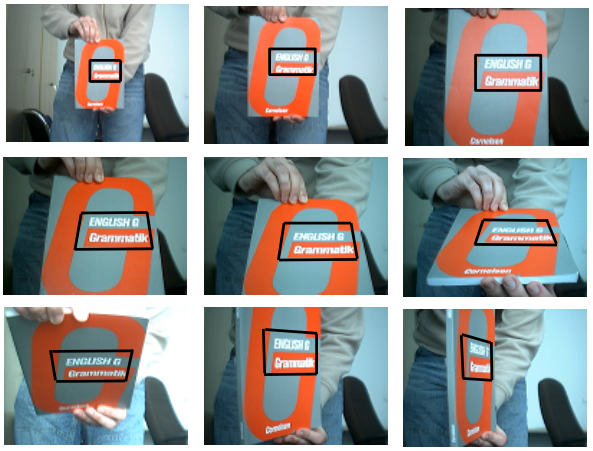


Figure 5. Training framework. We incrementally train the classifiers and the linear predictors over the frames of a training sequence. To this end, the object is automatically registered in each incoming frame using the current state of these classifiers and linear predictors.

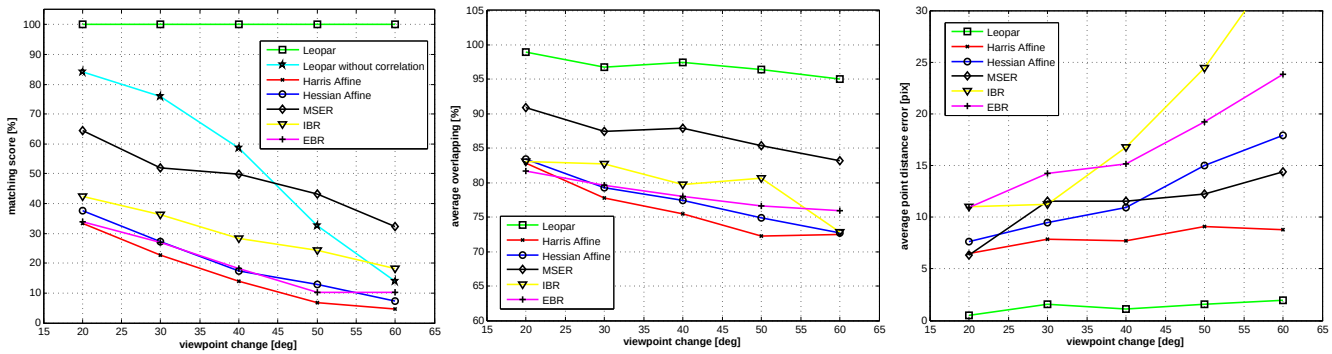
4. Experimental Validation

We compare our approach against *ad hoc* affine-invariant region detectors on the standard Graffiti image set [8]. The results of the comparisons are shown in Fig. 7 In these graphs, our method is denoted by 'Leopar' for LEarning Of PAtch Rectification. We then present results obtained with other video sequences and supply the computation times.

4.1. Robustness

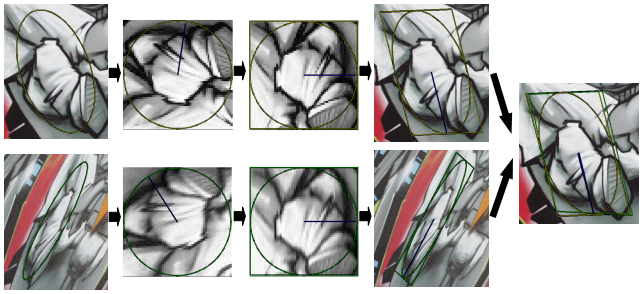
We extracted 150 interest points from the first image of the sequence using a simple Harris corner detector. Since no video sequence is available, we synthesized a training set by scaling and rotating the first image for changes in viewpoint angle up to 65 degrees and adding noise.

In Fig. 7(a), we plot the matching score defined as the ratio between the number of correct matches and the smaller number of regions detected in one of the two images, as defined in [8]. We extract affine regions using different region detectors and match them using SIFT. Two of them are said to be correctly matched if the overlap error is smaller than 40%. In our case, the regions are defined as the patch surfaces warped by the retrieved transformation. For a fair comparison, we first turned off our final check on the correlation since there is no equivalent for the affine regions in [8]. This yields the 'Leopar without Correlation' curve. Even then, our method performs much better at least up to an angle of $50°$. When we turn the final check on, the performances of our method become so good that not a single outlier is kept.



(a) (b) (c)

Figure 7. Comparing our method against affine region detectors on the Graffiti image set. **(a)** Matching score for viewpoint change. The curve 'Leopar without correlation' plots our results with the correlation test disabled. Even then, our method compares very favorably with the affine regions detectors. The curve 'Leopar' corresponds to our method with the correlation test turned on. No outlier is produced. **(b)** Average overlapping area of all correctly matched regions. Our method is very close to 100% and always more accurate than the other methods. **(c)** Average sum of the distances from the ground truth for the corner points. Once again, our method is much more accurate.



(a) (b) (c) (d) (e)

Figure 6. Measuring the overlapping errors and the corners distances. (a) Two matched affine regions. (b) The same regions, after normalization by their affine transformations and canonical orientations. (c) Squares are fitted to the normalized regions. (d) The squares are warped back into quadrangles in the original images. (e) The quadrangle of the second region is warped back with the ground truth homography and compare with the quadrangle of the first image. Ideally the two quadrangles should overlap.

4.2. Accuracy

In Fig. 7(b) and Fig. 7(c), we compare the accuracy obtained with the affine regions against our method. To create these graphs, we proceed as shown in Fig. 6. We first fit a square tangent to the normalized region, take into account the canonical orientation retrieved by SIFT and warp these squares back with the inverse transformation to get a quadrangle. Two corresponding regions should overlap if one of them is warped using the ground truth homography.

A perfect overlap cannot be expected since the affine region detectors are unable to retrieve the full perspective. As in SIFT, several orientations were considered when ambiguity arise and we kept the one that yields the most accurate correspondence. In the case of our method, the quadrangles are simply taken to be the patch borders after warping by the retrieved transformations.

Harris Point Extraction	0.028sec
Matching and Initialization	0.016sec
Linear Predictors	0.060sec

Table 1. Average run-time for detecting and matching 200 candidate keypoints against 50 learned patches.

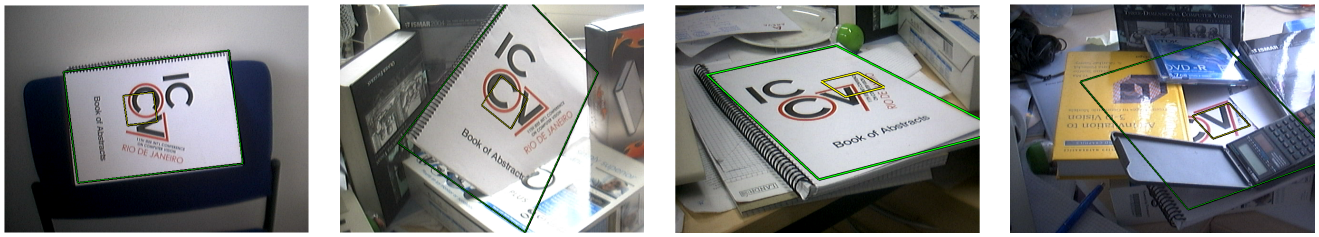
In Fig. 7(b), we compare the average overlap between the quadrangles and their corresponding warped versions obtained with our method and with the affine regions detectors. This overlap is very close to 100% for our method, about 10% better than MSER and about 20% better for the other methods. In Fig. 7(c), we compare the average error between the quadrangle corners. Once again, our method performs much better than the other methods. The error of the patch corner is less than two pixels in average.

4.3. Applications

In Figs. 8, 9, 10, and 11, we apply our method to object detection and pose estimation application using a low-quality camera. The method is robust and accurate even in presence of drastic perspective changes, light changes, blur, occlusion, and deformations. In Fig. 10 and Fig. 11, we used the template matching-based ESM algorithm [3] to refine the pose obtained from a single patch.

(a) (b) (c) (d)

Figure 8. Robustness to deformation and occlusion. **(a)** Patches detected on the book in a frontal view. **(b)** Most of these patches are detected even under a strong deformation. **(c)** The book is half occluded but some patches can still be extracted. **(d)** The book is almost completely hidden but one patch is still correctly extracted. No outliers were produced.



(a) (b) (c) (d)

Figure 9. Accuracy of the retrieved transformation. For each of these images, we draw the borders of the book estimated from a single patch. This is made possible by the fact we estimate a full perspective transform instead of only an affine one.

4.4. Computation Times

Our current implementation runs at 10 frames per second on a standard notebook (Intel M processor with 1.8GHz and 1GB RAM), without any special optimization, using 50 keypoints in the database and extracting 200 candidate keypoints in the input images. The average times for the most expensive steps are shown in Tab. 1. This compares very favorably with affine regions detectors which publicly available implementations typically take more than a second per image. A fast implementation of MSER exists but it uses intensive graphics card-based optimization.

5. Conclusion

We showed in this paper that a training phase not only for patch recognition but also for the transformation estimation considerably improves the robustness and the accuracy of the results of object detection, and this makes our approach highly desirable whenever the application permits it. Thanks to a three-step algorithm, it is possible to get match sets that do not usually contain any outliers. Even low-textured objects can therefore be well detected and their pose estimated.

We demonstrated our approach on a simple 3D tracking-by-detection application but many other applications could benefit from the proposed method, such as robot localization, object recognition, or image retrieval.

6. Acknowledgments

The authors are grateful for a quick and non-bureaucratic support through the Bayerische Forschungsstiftung represented by Prof. Dr. Friedrich R. Kreissl.

References

[1] S. Baker, A. Datta, and T. Kanade. Parameterizing homographies. Technical Report CMU-RI-TR-06-11, CMU, 2006.

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56:221–255, March 2004.

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

[4] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active Appearance Models. *PAMI*, 23(6):681–685, 2001.

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

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

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

[8] 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.

[9] N. Molton, A. Davison, and I. Reid. Locally Planar Patch Features for Real-Time Structure from Motion. In *BMVC*, 2004.

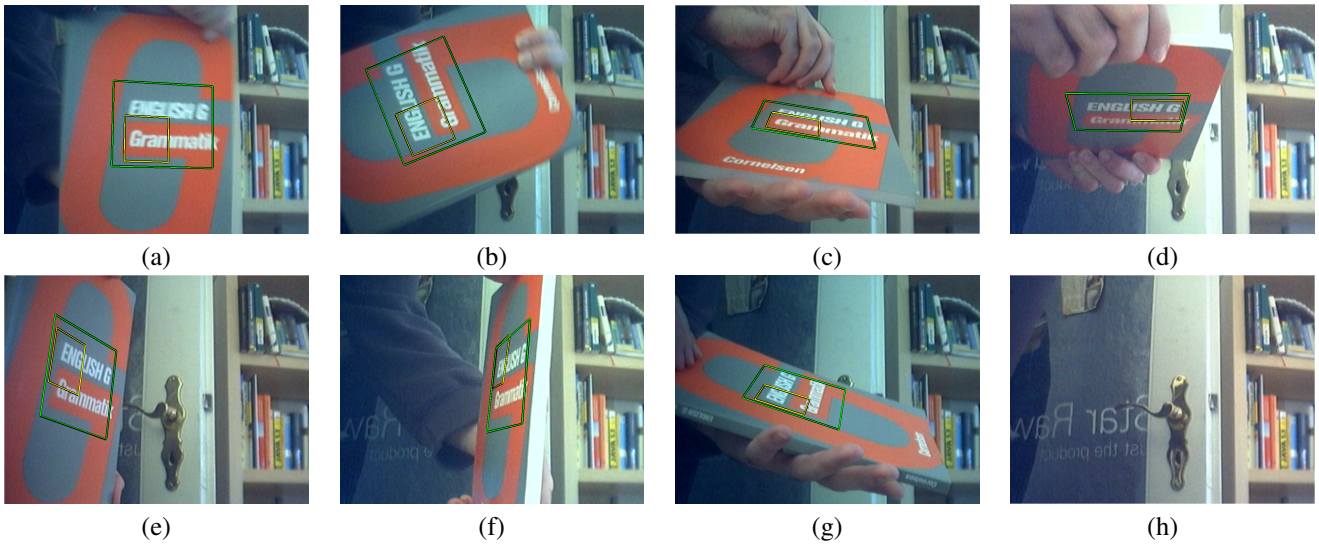


Figure 10. Some frames of a Tracking-by-Detection sequence shot with a low-quality camera. **(a)-(g)** The book pose is retrieved in each frame independently at 10fps. The yellow quadrangle is the best patch obtained by Leopar. The green quadrangle is the result of the ESM algorithm [3] initialized with the pose obtained from this patch. The retrieved pose is very accurate despite drastic perspective and intensities changes and blur. **(h)** When the book is absent, our method does not produce a false positive. *The corresponding video is available on http://campar.in.tum.de/Main/StefanHinterstoisser.*

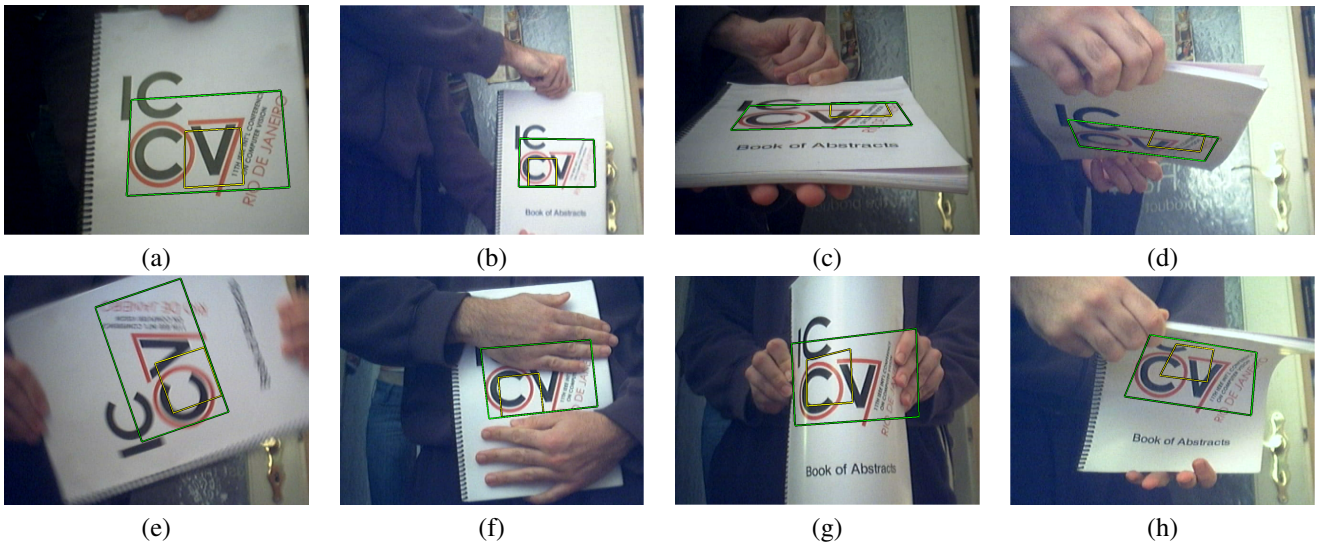


Figure 11. Another example of a Tracking-by-Detection sequence. The book pose is retrieved under **(b)** scale changes, **(c-d)** drastic perspective changes, **(e)** blur, **(f)** occlusion, and **(g-h)** deformations. *The corresponding video is available on http://campar.in.tum.de/Main/StefanHinterstoisser.*

[10] 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.

[11] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *CVPR*, June 2007.

[12] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature Harvesting for Tracking-by-Detection. In *ECCV*, 2006.

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

[14] 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.