Distance Transform Templates for Object Detection and Pose Estimation

Stefan Holzer[1], Stefan Hinterstoisser[1], Slobodan Ilic[2], Nassir Navab[1]

[1]Department of Computer Science, Technical University of Munich (TUM)
Boltzmannstrasse 3, 85748 Garching, Germany
[2]Deutsche Telekom Laboratories, TU Berlin
Ernst-Reuter-Platz 7, 10587 Germany
{holzers,hinterst,navab}@in.tum.de, slobodan.ilic@tu-berlin.de

Abstract

We propose a new approach for detecting low textured planar objects and estimating their 3D pose. Standard matching and pose estimation techniques often depend on texture and feature points. They fail when there is no or only little texture available. Edge-based approaches mostly can deal with these limitations but are slow in practice when they have to search for six degrees of freedom. We overcome these problems by introducing the Distance Transform Templates, generated by applying the distance transform to standard edge based templates.
We obtain robustness against perspective transformations by training a classifier for various template poses. In addition, spatial relations between multiple contours on the template are learnt and later used for outlier removal. At runtime, the classifier provides the identity and a rough 3D pose of the Distance Transform Template, which is further refined by a modified template matching algorithm that is also based on the distance transform. We qualitatively and quantitatively evaluate our approach on synthetic and real-life examples and demonstrate robust real-time performance.

1. Introduction

In recent years, many efficient methods for 2D and 3D object detection and pose estimation from monocular images were developed. Most of them assume textured objects and are based on template matching [20, 32, 2, 14] or on feature point recognition [21, 30, 25, 27, 34]. These approaches are efficient for matching, but in general do not provide the 3D pose. To recover the object pose 2D–3D correspondences are usually assumed [26, 22, 16, 31, 1, 7]. However, in many applications only little or no texture is present while only closed contours are available. Therefore, texture-based approaches for detection would fail, whereas on the other hand, classical template-based matching algorithms [19, 8, 28], which use a priori edge information, can deal with these limitations. Unfortunately these classical template matching methods are based on exhaustive search and are therefore known to be slow when exploring six degrees of freedom. In addition, depending on the scene clutter they can still have false positives. In this paper we address the problem of real-time planar object detection and 3D pose estimation. We concentrate on objects which have little or no texture but where closed contours are available as it is the case for many man-made objects, and overcome the limitations of the above mentioned approaches. Our method is based on the distance transform of edge-based templates, used for matching and pose refinement. The templates can be complex and composed of one or multiple closed contours. We developed a robust contour extraction algorithm based on the distance transform which is able to extract multiple closed contours robust to eventual gaps in the contour boundaries. Normalized template representations of the extracted contours are produced by

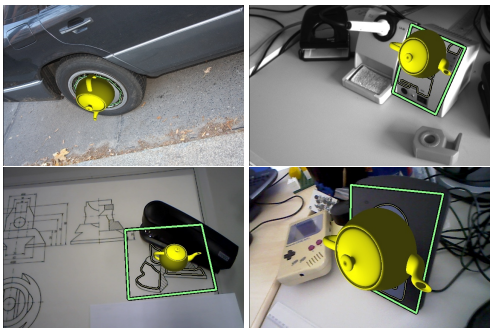
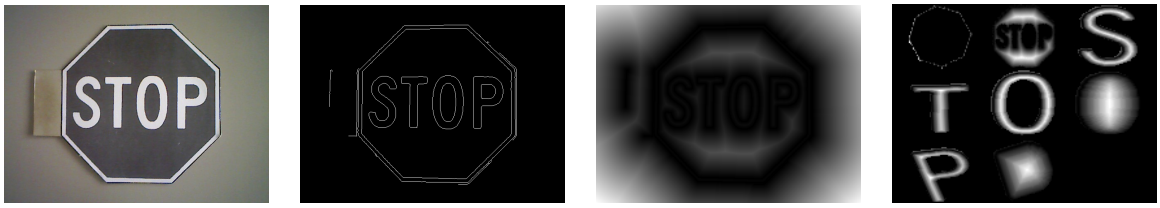


Figure 1. The results of detection and pose estimation obtained using the proposed approach. Different objects are detected and their pose estimate is visualized using the OpenGL teapot. Note that we can handle clutter, difficult view angles and partial occlusion.



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

Figure 2. An example input image **(a)**, the corresponding Canny edge image **(b)**, the distance transform computed from the edge image **(c)** and the eight extracted templates used for object detection **(d)**.

fitting an ellipse to them, which is done in a least squares sense [12]. The affine elliptical regions are normalized to the circular one similar to [25]. Perspective robustness is achieved by training the Ferns classifier [29] with various differently warped instances of the templates. Additionally, spatial relations between multiple contours are learnt and later used for outlier removal. Given a cluttered input image at runtime, we robustly extract independent object contours using the distance transform as discussed above. Such normalized Distance Transform Contour Templates are classified using the trained classifier to retrieve their identities and rough poses. The rough poses of the templates are further refined by a modified template matching algorithm. It is similar to the algorithm of Lucas-Kanade [24, 2], but instead of using the pixel intensities we use the image distance transform values. This simplifies the Jacobian calculation and speeds up the refinement step. Possible outliers are removed in two complementary steps. In one step, we evaluate the mean square error between the template warped with respect to the refined pose and the input image, and in the other we verify it by using offline learnt spatial relations between itself and its neighboring contours. Consensus must be obtained among the detected contour templates by verifying their spatial relations.

We perform qualitative and quantitative evaluation of our approach on real-life examples and demonstrate robust real-time performance. In the remainder of the paper we review related work, explain all the steps of the proposed approach and finally provide experimental evaluation.

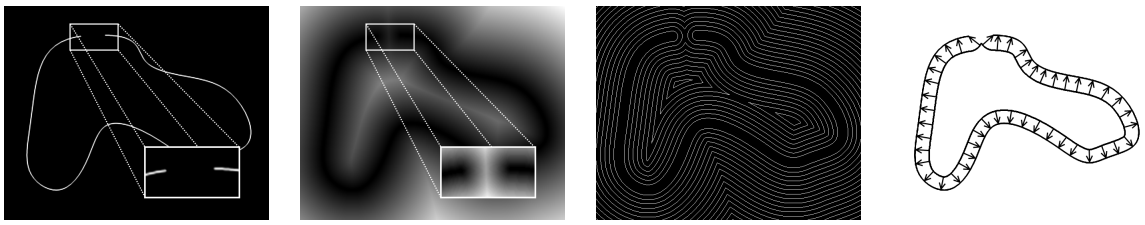
2. Related Work

The general problem of real-time object detection has been and still is one of the major computer vision problems. The challenge becomes bigger if the requirement is also to compute the 3D pose of the object. There are several approaches to this problem, which have been proven to work in practice. However, they solve either the detection or the pose estimation problem independently and rarely address both of them at the same time. Object detectors, which are efficient and work in real-time are specialized and designed for specific objects such as faces [33, 23, 5], pedestrians [8, 14, 6] etc. They are all based on learning where a large number of training images is used. The training is usually done only for the objects in one position and additional training effort has to be made to extend it to various poses. In this case, the obtained poses are just qualitative, like profile, frontal, etc., while the recovery of the precise 3D pose is not possible.

While the above techniques tend to be generic, [21] is designed to be object specific. Given one textured model image the appearance of the feature points invariant to the viewpoint is learnt by a classifier. These feature points are recognized in the incoming images, but no object pose information can be extracted directly. In order to estimate the pose other typical algorithms such as Posit [7], PnP [22, 26, 31, 1], etc. have to be applied. All these approaches can be efficiently used for tracking, but tend to fail if only few features can be extracted, which could be the case due to the lack of texture. Recently, the local planar patch pose has been learnt together with the feature point identities [18]. One retrieved local pose allows directly for rough pose estimation.

Contrary to feature-based approaches, template matching approaches can directly be used for the estimation of planar homographies between the planar model patch and the input image. They are either based on textured patches [24, 2] or on edges [28, 19]. The first approach minimizes an objective function which correlates the warped version of the patch with the image where the later one uses extensive search until the best matching score is obtained.

In this paper we employ a similar learning principle as [18], but instead of using feature points we use the distance transform as input information. The distance transform is computed on our model template we intend to detect. It is robust to illumination changes and does not depend on the texture. Although the distance transform has already been used for template matching techniques [13, 15], we use it in a completely new way for fast and robust matching and for initial rough pose estimation. In order to refine this initial pose, we also introduce a novel technique inspired by the Lucas-Kanade algorithm, but instead of using the pixel intensities we again use the distance transform of the model template.



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

Figure 3. The first three images from the left show an edge image, its corresponding distance transform and different level curves of the distance transform. Dark values of the distance transform correspond to small distances. Small gaps in the contour vanish if the level curve is extracted at a high enough value. The most right image shows the propagation of the extracted level curve to real edges.

3. Distance Transform Templates

The model image of Fig. 2(a) is composed of multiple closed contours and each contour is extracted and associated to a single Distance Transform Template as shown in Fig. 2(d). The approach we propose relies on a training phase in which the poses, identities and the spatial relations of the Distance Transform Templates are learnt. At runtime extracted closed contour templates are classified in order to simultaneously retrieve their identities and the corresponding pose approximations represented by homographies. The retrieved poses are refined using distance transform based template matching and outliers are removed by using learnt spatial relationships among contours. In the remainder of this section we discuss these steps in details.

3.1. Creating templates

In this section we show how to extract, normalize and mask contours in order to get template patches that can be used for matching. This procedure is applied to the model image before training and to all input images at runtime.

The preliminary step of our approach is the extraction of closed contours from cluttered input images. This is challenging because of large perspective distortions and possible edge discontinuities occurring sometimes due to bad lighting conditions or due to reflections. We start by extracting seed points that will be used for contour extraction. To do this, we compute the distance transform of the edge map, shown in Fig. 2(b), corresponding to the input image. In practice, we use the Canny edge detector [4] for creating the edge map. The distance transform [3, 10] of an edge map is a map, in which each entry, or pixel, has the value of the distance to the nearest edge pixel in the edge map as shown in Fig. 2(c) and Fig. 3(b). The seed points are defined as local maxima of the distance transform and are processed in descending order of their distance transform values. We define a closed contour of a shape as connected and closed sequence of zero valued pixels in the edge distance map around an initial seed point. However, due to small gaps, a closed and connected sequence of zero valued pixels is very often impossible to find. Therefore, we compute the level curve of the distance transform at a specific value β, where the contour appears to be closed as shown in Fig. 3(c). In practice, β is computed by multiplying the seed point value with a predefined value α, and it is reached by going downhill starting at the seed point. Once a pixel on the distance map with value β is arrived the corresponding level curve of the same value is followed to extract the contour. Multiplying the seed point value with α makes the selected value β invariant to scale. In order to ensure the robustness to perspective transformations and to close eventual gaps in the initial contour, the extracted contour points are propagated to the zero values of the distance transform as shown in Fig. 3 (d).

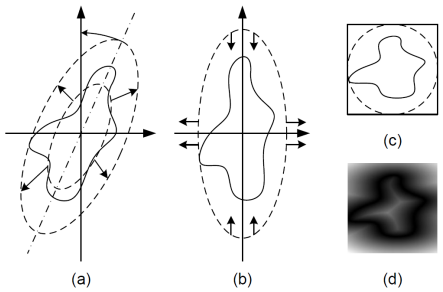


Figure 4. The normalization of the contour patch to a canonical scale and orientation. (a) Fixing the orientation to the y-axis. (b) Fixing the scale to the size of the predefined patch (c) used for classification. (d) The final normalized Distance Transform Template obtained by warping the distance transform of the input image.

Seed points within the same shape will obviously produce the same contour many times. In order to prevent this an indexing map is used that marks a contour as soon as it has been extracted.

The normalization of the template patch consists of transforming the contour to a canonical scale and orientation. The first step is to fit an ellipse into the contour points in a least square sense [12]. Then the orientation of the contour is fixed by rotating the fitted ellipse such that its dominant direction is aligned to the y-axis. The canonical scale is obtained by scaling all contour points to be inside the ellipse area. Finally, the rotated contour is normalized to a circle as depicted in Fig. 4.

The so created normalized Distance Transform Templates should be discriminative enough to be used to train

(a) (b) (c)

Figure 5. Different maskings of the second template shown in Fig. 2(d). **(a)** The unmasked Distance Transform Template. **(b)** The outer parts of the contour, shown in black, are masked. **(c)** The complete masked template with inner and outer contours masked.

and test a classifier. However, due to background clutter and eventual contour discontinuities it is not desirable to take the entire distance transform map into account. For this reason we consider only the area enclosed by the contour for the classification, which leads, in practice, to better classification results. To achieve this, we introduce masking. We mask the neighborhood of the outer contour and the inner contours. The masking process is shown in Fig. 5.

3.2. Training

We train the Ferns classifier of [29] using the Distance Transform Templates extracted form the model image. We do not only learn the template identity T_{id}, but also the template pose T_{pos}. This is done by creating view-dependent classes for each closed contour template. In practice, we randomly warp the model image, extract the contours and normalize them. We then check for each extracted contour if it fits to one of the already created classes using the pose refinement described in the next Sec. 3.3. If the mean square error of the refined pose is below a specific threshold we update the posterior probability $P_{pos,id} = P(T_{id} = id, T_{pos} = pos|\mathcal{P})$ of the template identity at the specified pose. Otherwise, the extracted contour is considered as a new class. The total number of classes is equal to the total number of non-equal poses obtained for all template classes. Once trained, the classifier retrieves the template identity $\hat{T}_{id}$ and its pose $\hat{T}_{pos}$ simultaneously, as:

$$(\hat{T}_{id}, \hat{T}_{pos}) = \underset{(Id,Pos)}{\operatorname{argmax}} \; P(T_{id} = Id, T_{pos} = Pos|\mathcal{P}), \tag{1}$$

where Id is a random variable representing the identity of the patch $\mathcal{P}$ and Pos is a random variable representing the pose of the patch. The classifier is usually working on the intensity values of an image and is able to retrieve the patch identity under scale, perspective and lighting variations. Since the gray value image of the contour generally gives low information due to its low texturing, we apply the classifier directly on the Distance Transform Template map that represents the given information in a more dense way.

In addition to learning the pose and identity of the template we also learn spatial relations between multiple closed contours of the same model template. By this, we are able to remove outliers that are still present after the self-verification procedure of Section 3.3. For the spatial relationship verification, we use the pose estimated from the matched and self-verified template to project the relative positions of its neighboring contours into the current image. The contours found at these projected positions will be checked regarding to their compatibility to the expected contours. Since the model template can consist of many ambiguous contour templates - usually those having similar shapes - we first have to check whether each closed contour, chosen to verify another contour template, is located and shaped distinctively enough. To do this, we cluster similar contours and identify critical spatial relationships offline. The clustering is done by randomly warping the model image, extracting the contours and matching them against all collected model contours using the already trained Ferns classifier. If the warped instance of the model template is matched to the wrong identity, we add this identity to a list of wrongly matched template classes. By doing so, we get a list of ambiguous template classes for each closed contour. Due to their ambiguity, there is a high probability that a contour within a cluster is matched to another model contour of the same cluster. For example, looking at Fig. 6 the contours 2, 4 and 6 and the contours 1, 5 and 7 will be clustered together. Therefore, we check for each class if one of its relations is also valid for another class of the same cluster. If this is not the case, we mark the relation as non-critical. Fig. 6 shows an exemplary configuration of contours, where the arrows indicate the relations used for the spatial relationship verification. For example, neither the shapes of the templates nor the relations between the contours 4 and 5 and between the contours 6 and 7 are distinctive enough to be used for verification without any further information. The final training step consists of the creation of a distance and a gradient map for each template as presented in Sec. 3.3 - similar to the ones used for self-verification, but this time from the frontal and unnormalized view of the model image. Using these special maps for the final pose refinement allows to take all templates into account that were involved in the spatial relation verification.

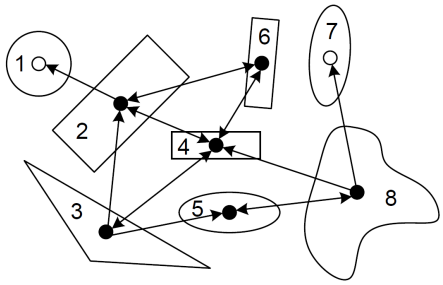


Figure 6. Verification of contours using their neighbors. Contours with filled black points are validated using the spatial relations visualized by the arrows.

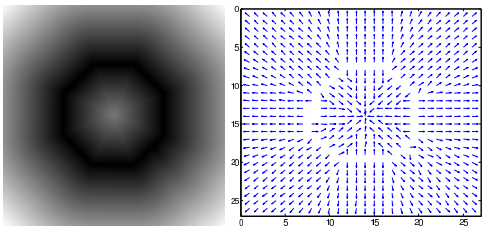


Figure 7. The patch distance (left) and gradient (right) maps as used for pose refinement. Shown maps correspond to the mask shown in Fig. 5. Dark values of the distance map correspond to small values.

3.3. Pose refinement

At runtime closed contours are extracted from the input image and Distance Transform Templates are created. Querying the classifier of Sec. 3.2, the identity and the approximate pose for each template is retrieved. The obtained pose is refined by an approach similar to the Lucas-Kanade template tracking algorithm [24]. In contrast to the Lucas-Kanade algorithm, which is based on the image intensities and gradients, we use distance transform values and simulated gradients. More precisely, for each template, we precompute the distance map $\mathcal{D}_\mathcal{T}$ and the gradient map $\mathcal{G}_\mathcal{T}$. The distance map $\mathcal{D}_\mathcal{T}$ is similar to the distance transform of the normalized model contour without considering the neighborhood and the inner contours as shown in Fig. 7. The gradient map $\mathcal{G}_\mathcal{T}$ is a map of two-dimensional values where each entry is a vector indicating the direction from the considered pixel to the nearest point on the normalized model contour. In practice, the direction of each vector is reverted such that it points from the normalized model contour away and matches the natural gradient direction as shown in Fig. 7. The gradient directions are normalized such that their length is equal to 1. Note that these maps are precomputed for all template patches.

The goal of the Lucas-Kanade algorithm is to align a template image $\mathcal{I}_T$ to an input image $\mathcal{I}$ by minimizing the vector of image value differences $\mathbf{y}\left(\Delta\mathbf{x}\right)$ with respect to the increment of the parameters $\Delta\mathbf{x}$ of the warp function $\mathbf{w}\left(\hat{\mathbf{x}}+\Delta\mathbf{x};\mathbf{p}_i\right)$, which is in our case a homography:

$$\mathbf{y}\left(\Delta\mathbf{x}\right)=\Big[\; y_1(\Delta\mathbf{x})\; y_2(\Delta\mathbf{x})\; \ldots\; y_q(\Delta\mathbf{x})\;\Big]^T, \qquad (2)$$

where

$$y_i(\Delta\mathbf{x})=\mathcal{I}\left(\mathbf{w}\left(\hat{\mathbf{x}}+\Delta\mathbf{x};\mathbf{p}_i\right)\right)-\mathcal{I}_T\left(\mathbf{p}_i\right), \qquad (3)$$

and q is the number of pixel locations of the template image, $\hat{\mathbf{x}}$ and $\Delta\mathbf{x}$ are k-dimensional vectors containing the parameters of the warp and its increments respectively, and $\mathbf{p}_i=[u_i,v_i,1]^T$ is an image point.

In order to use the offline computed distance and gradient maps in the Lucas-Kanade like optimization, we replace the template image $\mathcal{I}_T$ and the input image $\mathcal{I}$ with the corresponding template distance map $\mathcal{D}_\mathcal{T}$ and image distance map $\mathcal{D}$ respectively. Furthermore, we only consider parts of the template covered by the projected contour points and switch the roles of the template and the image of the classical Lucas-Kanade objective function of Eq. 3 and redefine $y_i(\Delta\mathbf{x})$:

$$y_i(\Delta\mathbf{x}):=\mathcal{D}_T\left(\mathbf{w}^{-1}\left(\hat{\mathbf{x}}+\Delta\mathbf{x};\mathbf{p}_i\right)\right)-\mathcal{D}\left(\mathbf{p}_i\right). \qquad (4)$$

This is solved by first using a linear approximation of $\mathbf{y}\left(\Delta\mathbf{x}\right)$ and then minimizing this approximation iteratively using Gauss-Newton gradient descent.

Using the chain rule, the ith line of the Jacobian matrix $\mathbf{J}_\mathbf{y}(\mathbf{0})$ can be written as product of two Jacobian matrices:

$$\begin{aligned}\mathbf{J}_{y_i}(\mathbf{0}) &= \left.\frac{\partial y_i(\Delta\mathbf{x})}{\partial\Delta\mathbf{x}}\right|_{\Delta\mathbf{x}=\mathbf{0}}\\ &= \left.\frac{\partial\mathcal{D}_T\left(\mathbf{p}\right)}{\partial\mathbf{p}}\right|_{\mathbf{p}=\mathbf{w}^{-1}(\hat{\mathbf{x}};\mathbf{p}_i)}\cdot\left.\frac{\partial\mathbf{w}^{-1}\left(\mathbf{x};\mathbf{p}_i\right)}{\partial\mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}\\ &= \mathbf{J}_{\mathcal{D}_\mathcal{T}}(\mathbf{w}^{-1}\left(\hat{\mathbf{x}};\mathbf{p}_i\right))\cdot\mathbf{J}_{\mathbf{w}^{-1}(\cdot;\mathbf{p}_i)}(\hat{\mathbf{x}}) \qquad (5)\end{aligned}$$

where $\mathbf{J}_{\mathcal{D}_\mathcal{T}}(\mathbf{w}^{-1}\left(\hat{\mathbf{x}};\mathbf{p}_i\right))$ is a (1×3) matrix and $\mathbf{J}_{\mathbf{w}^{-1}(\cdot;\mathbf{p}_i)}(\hat{\mathbf{x}})$ is a $(3\times k)$ matrix. Therefore, $\mathbf{J}_{y_i}(\mathbf{0})$ is a $(1\times k)$ matrix and $\mathbf{J}_\mathbf{y}(\mathbf{0})$ is a $(n\times k)$ matrix, where n is the number of contour points.

The first and the second row of the Jacobian matrix $\mathbf{J}_{\mathcal{D}_\mathcal{T}}(\mathbf{w}^{-1}\left(\hat{\mathbf{x}};\mathbf{p}_i\right))$ are the gradients of the distance transform map $\mathcal{D}_T$ in x- and y-direction evaluated at $\mathbf{w}^{-1}\left(\hat{\mathbf{x}};\mathbf{p}_i\right)$ and the third row is 0, since the third component of a pixel location is always 1. However, instead of using the real gradients in x- and y-direction, we use the x- and y-values of the corresponding precomputed gradient map $\mathcal{G}_T$. The entries of the Jacobian matrix $\mathbf{J}_{\mathbf{w}^{-1}(\cdot;\mathbf{p}_i)}(\hat{\mathbf{x}})$ depend on the specific parameterization of the warp. For details about possible warps refer to [2]. Since only the points that are covered by a contour pixel are considered for optimization, $\mathcal{D}\left(\mathbf{p}_i\right)$ is zero for all evaluated points and, therefore, only the precomputed distance map $\mathcal{D}_T$ is needed to compute $\mathbf{y}\left(\Delta\mathbf{x}\right)$. Once the pose is estimated using the proposed procedure, we perform a self-verification step to identify obvious outliers. This is done by evaluating the mean square error of the recovered pose.The corresponding 3D pose can be computed using homography decomposition [9].

4. Runtime and Results

In this section, we describe how to get during run time from an input image to a final pose estimate. We also present results of qualitative and quantitative evaluation on different real examples. Note that all our experiments run in real-time on a laptop with a 2 GHz Intel(R) Core(TM)2 Duo CPU and 2 GB RAM.

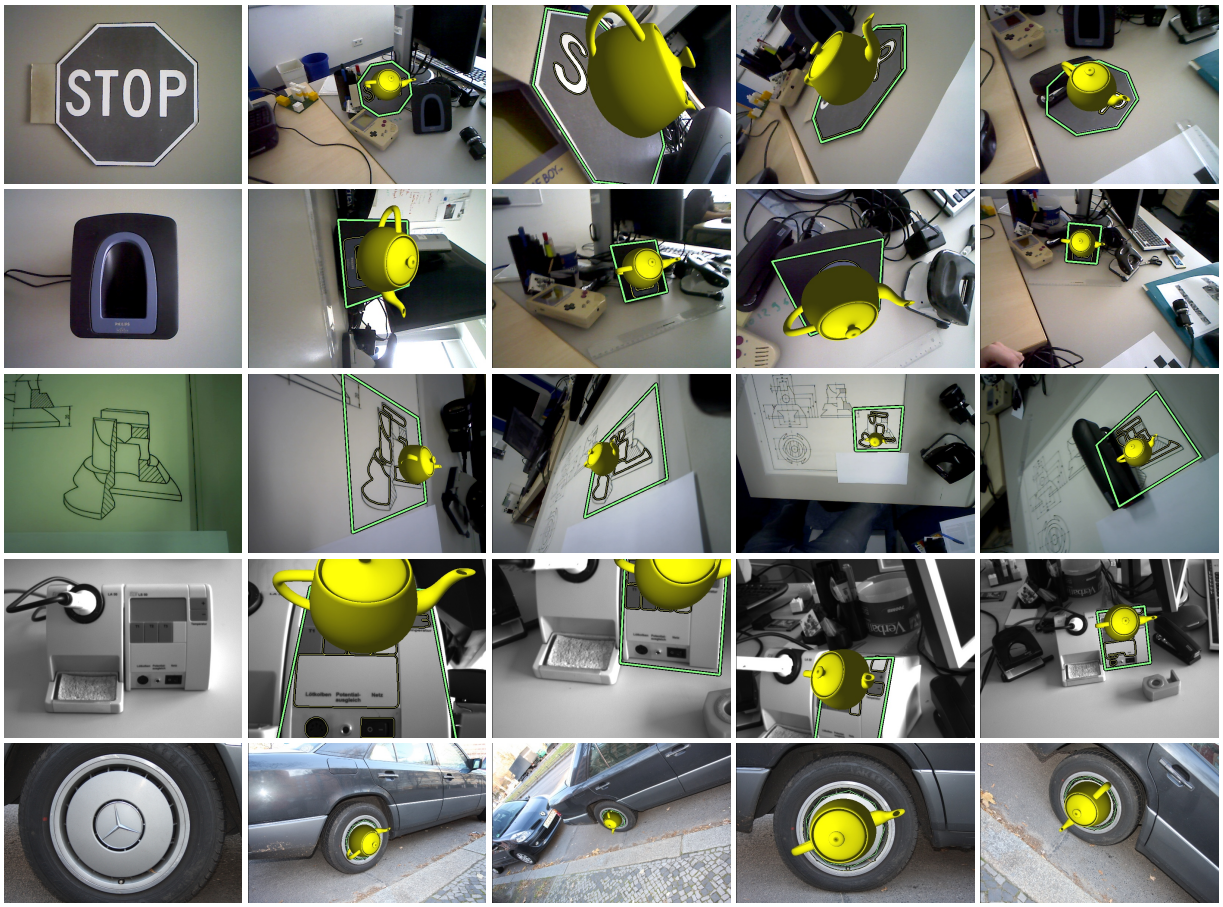


Figure 8. The results of the proposed approach applied to the real world examples. The first image of each row shows the object to be detected and the other images show the detected objects in cluttered images under different perspectives. The estimated pose is visualized using the OpenGL teapot.

4.1. Runtime and Qualitative Evaluation

The first column of Fig. 8 depicts several model template images used for training. The other images of the same Figure represent input camera images processed in real-time at runtime. The model templates are detected and their poses are estimated, what is visualized by drawing an OpenGL teapot on them. The overall procedure goes as follows. First, all closed contours are extracted in the input image using the distance transform. Then, they are normalized to the canonical scale and orientation and masked as described in Section 3.1 to create Distance Transform Templates. The extracted templates are classified using the previously trained Ferns classifier and their identities and approximate poses are retrieved as discussed in Section 3.2. The retrieved pose estimates are refined using the optimization method described in Section 3.3 and the resulting mean square errors are used for self-verification in order to discard obvious outliers. Finally, the spatial relations among the contours, learnt offline, are verified in order to remove remaining outliers. This is done by initially considering each contour independently and trying to verify its detected identity using its neighbors as demonstrated in Fig. 6. In addition, the identities of its neighboring contours are verified. If a template contour is verified by a given number of neighboring contours, it is considered as inlier. In practice it has been shown that two neighbors are enough to robustly remove outliers.

4.2. Quantitative Evaluation

To evaluate the quality of our approach, we performed several experiments on synthetic images. The synthetic images are created by rendering the template image at randomly chosen poses on a highly cluttered background and by adding white noise and affine illumination. In Fig. 9 and in Fig. 10, we show the percentage of successful pose estimations at particular viewing angles for an engineering drawing and for the front of a soldering iron station. Our approach is compared to Ferns [29] followed by RANSAC [11] (‘Ferns+RANSAC’) and N3Ms [17] (‘Ferns+N3Ms’) respectively. N3Ms are investigated since they use a similar local verification strategy as we do. Ad-

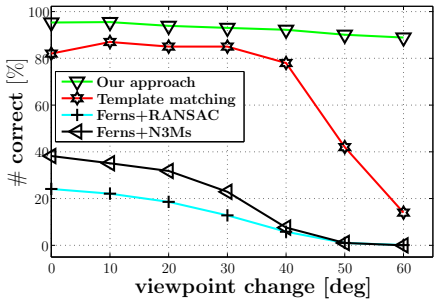


Figure 9. Comparison of different approaches applied on an engineering drawing.

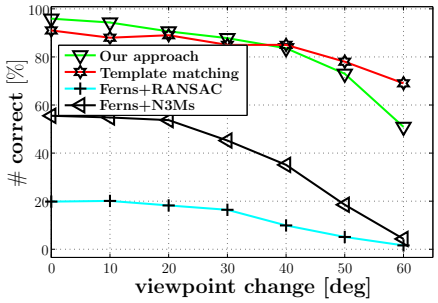


Figure 10. Comparison of different approaches applied on the front of a soldering iron station.

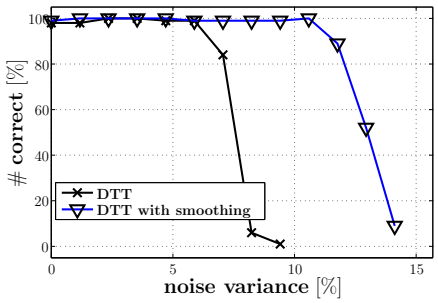


Figure 11. Evaluation of the proposed approach regarding to noise.

ditionally, we compared to an edge-based template matching method [19] ('Template matching'). Note, that our approach outperforms both approaches based on Ferns in terms of detection and pose estimation. The decrease of the performance of our algorithm for the soldering iron station is mainly due to the similarity of the available template contours. They are often mutually confused and classified as the same contour. Comparing to edge-based template matching [19] that uses exhaustive search, we achieve better results for the engineering drawing, especially for large viewing angles, and get slightly better results for the soldering iron station for small viewing angles. One reason for the superiority in the case of the engineering drawing is that thin lines can vanish in the higher pyramid levels as they are used for template matching to increase speed. This is not the case for the soldering iron station since clear edges are available. Since our approach is independent of scale, it clearly outperforms edge-based template matching in terms of speed in applications, where high scale ranges are necessary. For the accomplished tests the average runtime of our approach was approximately $0.15s$, while the edge-based template matching method of [19] needed approximately $1.0s$. The tests were done within a scale range of $[0.9...1.9]$ on images of size 1280×960. Note that our method is an unoptimized version whereas for the edge-based template matching a highly optimized implementation was used. Our approach also works on larger scale ranges without significantly decreasing the runtime, which is not the case for [19]. Fig. 11 shows the results of the evaluation regarding to synthetically added noise. The black line hereby shows the results for the proposed approach whileas the blue line shows the results for a modified version, where the image is smoothed before processing.

5. Conclusion

We introduced a new way of using the distance transform for detecting textureless planar objects and estimating their 3D pose. The model image of the object of interest is used to extract and create Distance Transform Templates corresponding to the closed contours on the object. The identities, poses and spatial relations of the Distance Transform Templates are learnt by training the Ferns classifier. From the cluttered input image, containing the object of interest at some position, the closed contours are extracted and template patches are created using the distance transform. Querying the classifier the identities and the approximate poses are obtained and further refined by using a modified template matching algorithm based on the distance transform. Finally, we showed that distinctive spatial relationships coupled with a distance transform based self-verification are useful to robustly remove wrongly matched templates.

We evaluated the performance of our approach on synthetic and real-life examples and compared it to other detection methods. We demonstrated that we perform better, in terms of pose estimation and computational efficiency, than existing methods, such as Ferns [29] followed by RANSAC [11], N3Ms [17] and edge-based template matching [19]. In particular the performance of our method is better in the case of large scale changes. We applied our approach to 3D tracking-by-detection, however, in practice, it can be used in many other applications such as object recognition, image retrieval or robot localization.

Acknowledgments

We want to thank Andreas Hofhauser for providing the results for his template matching approach. This project was funded by the BMBF project AVILUSplus (01IM08002).

References

[1] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

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

[3] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371, June 1986.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[5] Y. L. Chang Huang, Haizhou Ai and S. Lao. Vector boosting for rotation invariant multi-view face detection. In *International Conference on Computer Vision*, 2005.

[6] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, 2005.

[7] D. DeMenthon and L. S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141, 1995.

[8] M. Dimitrijevic, V. Lepetit, and P. Fua. Human Body Pose Detection Using Bayesian Spatio-Temporal Templates. *Computer Vision and Image Understanding*, 104(2-3):127–139, 2006.

[9] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 2:485–508, 1988.

[10] P. F. Felzenszwalb and D. P. Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, September 2004.

[11] M. Fischler and R. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications ACM*, 24(6):381–395, 1981.

[12] A. W. Fitzgibbon and R. B. Fisher. A buyer's guide to conic fitting. In *BMVC '95: Proceedings of the 6th British conference on Machine vision (Vol. 2)*, pages 513–522, Surrey, UK, 1995. BMVA Press.

[13] D. M. Gavrila. Multi-feature hierarchical template matching using distance transforms. *Pattern Recognition, International Conference on*, 1:439, 1998.

[14] D. M. Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1408–1421, 2007.

[15] A. Ghafoor, R. Naveed Iqbal, and S. Shoad Khan. Image matching using distance transform. In *Scandinavian Conference on Image Analysis*, pages 654–660, Göteborg, Sweden, June 2003.

[16] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[17] S. Hinterstoisser, S. Benhimane, and N. Navab. N3m: Natural 3d markers for real-time object detection and pose estimation. In *IEEE International Conference on Computer Vision*, pages 1–7, Rio de Janeiro, Brazil, October 2007.

[18] S. Hinterstoisser, S. Benhimane, N. Navab, P. Fua, and V. Lepetit. Online learning of patch perspective rectification for efficient object detection. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, Alaska, 2008.

[19] A. Hofhauser, C. Steger, and N. Navab. Harmonic deformation model for edge based template matching. In *Third International Conference on Computer Vision Theory and Applications*, volume 2, pages 75–82, Funchal, Portugal, January 2008.

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

[21] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, Sept. 2006.

[22] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 2008.

[23] S. Z. Li and Z. Zhang. FloatBoost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9), 2004.

[24] B. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[25] 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.

[26] F. Moreno-Noguer, V.Lepetit, and P.Fua. Pose priors for simultaneously solving alignment and correspondence. *European Conference on Computer Vision*, October 2008.

[27] 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.

[28] C. F. Olson and D. P. Huttenlocher. Automatic target recognition by matching oriented edge pixels. *IEEE Transactions on Image Processing*, 6:103–113, Jan. 1997.

[29] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *Conference on Computer Vision and Pattern Recognition*, Minneapolis, MI, USA, June 2007.

[30] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature Harvesting for Tracking-by-Detection. In *European Conference on Computer Vision*, 2006.

[31] L. Quan and Z. Lan. Linear N-Point Camera Pose Determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(7):774–780, jul 1999.

[32] A. Thayananthan, R. Navaratnam, P. Torr, and R. Cipolla. Likelihood models for template matching using the pdf projection theorem. In *British Machine Vision Conference*, pages 949–958, 2004.

[33] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[34] K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1), 2008.