in: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2012), Vilamoura, Portugal, pp. 2108-2115, IEEE 2012

# Vision-based Hyper-Real-Time Object Tracker for Robotic Applications

Alexander Kolarow[1], Michael Brauckmann[2], Markus Eisenbach[1], Konrad Schenk[1],
Erik Einhorn[1], Klaus Debes[1] and Horst-Michael Gross[1]

*Abstract*— **Fast vision-based object and person tracking is important for various applications in mobile robotics and Human-Robot Interaction. While current state-of-the-art methods use descriptive features for visual tracking, we propose a novel approach using a sparse template based feature set, which is drawn from homogeneous regions on the object to be tracked. Using only a small number of simple features, without complex descriptors in combination with logarithmic-search, the tracker performs at hyper-real-time on HD-images without the use of parallelized hardware. Detailed benchmark experiments show that it outperforms most other state-of-the-art approaches for real-time object and person tracking in quality and runtime. In the experiments we also show the robustness of the tracker and evaluate the effects of different initialization methods, feature sets, and parameters on the tracker. Although we focus on the scenario of person and object tracking in robot applications, the proposed tracker can be used for a variety of other tracking tasks.**

## I. Introduction

Tracking arbitrary objects or persons in video sequences in real-time, is a key condition for many applications in mobile service robotics and Human-Robot Interaction (HRI). Applications vary from tracking body parts (e.g. head or hand tracking for mimic and gesture classification), object tracking (e.g. manipulating objects in dynamic scenes), and person tracking (e.g. visual following of a person [1] or person re-identification [2]). Since typically only very limited hardware is available on an autonomous mobile robot platform, very hard computational restrictions are demanded for the used algorithms.

In recent years, many efficient tracking methods have been introduced for different tasks. Nevertheless, many of them have disadvantages regarding two mayor criteria:

- Often, underlying assumptions about the environment can not be met, including static background, no changes in lighting and inhomogeneous or invariant appearances. These idealized conditions are usually missing for object tracking in high dynamic environments, as they are common in challenging mobile robotics scenarios, as the one presented in [3]. Methods building complex object models in advance are often not applicable, due to the high variability in the appearances of the object or person to be tracked.
- Most methods are computationally very expensive. In the proposed applications, the objects often need to be detected and tracked in real-time, however, typically only 10% of the computational ressources of the robot are available for a tracking task, as the other main tasks (navigation, dialog, etc.) claim the remaing part. This may be achieved with increased processing power, e.g. by using GPU parallelization, but mobile robots often lack of the required hardware configurations of GPU extensions or the use is restricted, due to their high power consumption.

In this paper, we introduce a novel template based approach for hyper-real-time object tracking in dynamic environments. We define hyper-real-time as significantly faster than real-time, which is 25 fps. After initialization, the presented object tracker needs less than five milliseconds per frame on HD-images, using a single core of a Intel Core i7, while outperforming most other state-of-the-art approaches in tracking quality. Therefore, it can be used for a variety of applications in mobile autonomous service robotics and Human-Robot Interaction.

The remainder of the paper is organized as follows: In Sect. II, we summarize current state-of-the-art methods for object and person tracking. Then, we introduce the proposed tracking method and give a short overview of different configurations in Sect. III. In the experimental Sect. IV, we evaluate different initialization and parameter setups on public benchmark datasets and own human-robot interaction experiments. Finally, we summarize and give an outlook on upcoming improvements and applications.

## II. State-of-the-Art

For visual object tracking, many successful and accurate approaches have been proposed in recent years. Fig. 1 shows a categorization. Examples for each category can be found in [4].

Tab. I shows a comparison of the proposed method to common and related approaches: Some common methods (interest points) are only applicable to a certain degree, since they are designed for tracking structured regions which are not present in every scenario. Our method is able to track objects without structural information by using color features sampled from homogeneous regions or a collection of homogeneous subregions. Additionally, it can handle full occlusions and all affine transformations that typically occur when objects move in 3D. The method presented in [13]

*This work has received funding from the German Federal Ministry of Education and Research as part of the APFel project under grant agreement no. 13N10797.

[1] A. Kolarow, M. Eisenbach, K. Schenk, E. Einhorn and M. Gross are with the Neuroinformatics and Cognitive Robotics Lab at Ilmenau University of Technology, Germany `alexander.kolarow at tu-ilmenau.de`

[2] M. Brauckmann is with L-1 Identity Solutions AG, Bochum, Germany `michael.brauckmann at morpho.com`

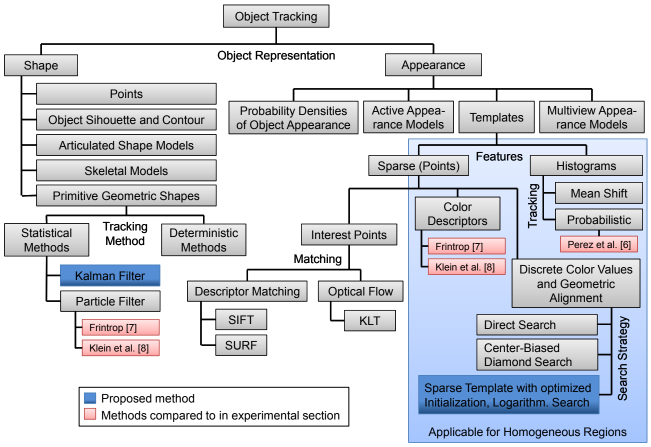


Fig. 1. Categorization of object tacking approaches and the proposed method following the scheme of [4]. Approaches applicable for tracking homogeneous regions are highlighted blue.

belongs to the same category as our approach. It is designed for tracking predefined homogeneous regions on a face with a fast search strategy (center-biased diamond search) using a set of randomly sampled points within the defined regions. However, this approach is not sufficiently versatile to be used in other scenarios. In comparison to that, our approach is able to detect regions suitable for sampling points on any object. It uses a logarithmic search for finding the best matching position of the template. The idea of using logarithmic search for tracking was introduced in [14]. Logarithmic search allows for speeding up the search enormously and increasing the accuracy. But the conditions for the applicability for tracking are hard to fulfill. As far as we know, our approach is the first one, that automatically initializes a sparse template that fulfills the requirements for logarithmic search, as we will show in Sect. III. The combination of *logarithmic search*, a *very small number of describing points* for the sparse template, and the *abstinence of complex descriptors* for these points allows to track an object with very few comparisons. Therefore, our approach is the only one as far as we know, being able to track an arbitrary object in hyper-real-time, even on HD-images, without using the effect of parallelization on special hardware.

| Method | Affine Trans. | Occl. Handl. | Init. Region | Appl. for Homog. Reg. | Speed SD | Speed HD |
|---|---|---|---|---|---|---|
| Mean-shift [5] | T+S | part. | opt. | yes | RT | S |
| Prob. Tr. of Hist. [6] | T+S | part. | opt. | yes | RT | S |
| Col. Desc. [7], [8] | T+S+R | full | opt. | yes | RT | S |
| SIFT [9], [10] | A | part. | opt. | no | RT | S |
| KLT [11], Opt. Flow [12] | A | part. | opt. | no | RT | S |
| Diam. Sear. [13] | T+S | none | predef. | yes | RT | RT |
| **Proposed Method** | A* | full | opt. | yes | HRT | HRT |

TABLE I

COMPARISON OF THE APPROACH PRESENTED IN THIS PAPER (LAST ROW) WITH RELATED APPROACHES. ABBREVIATIONS: AFFINE TRANSFORMATIONS (TRANSLATION, SCALE, ROTATION, AFFINE (*: WE USE T+S+R, BUT A CAN BE DONE IF NECESSARY); SPEED SD: COMPUTATIONAL SPEED ON SMALL IMAGES (S SLOW, RT REAL-TIME, HRT HYPER-REAL-TIME), SPEED HD: COMPUTATIONAL SPEED ON HD-IMAGES WITHOUT SPECIAL HARDWARE

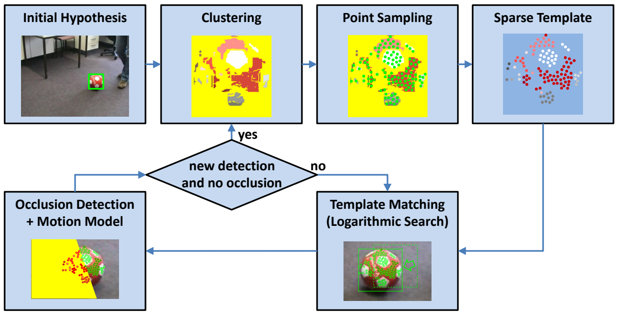

Initial Hypothesis
Clustering
Point Sampling
Sparse Template
yes
new detection and no occlusion
no
Occlusion Detection + Motion Model
Template Matching (Logarithmic Search)

Fig. 2. Proposed algorithm: Logarithmic search in combination with a sparse template. The sparse template is built from samples of homogeneous regions, extracted by clustering. The template matching is done using logarithmic search. To be able to react to perspective or lightning changes, the template is updated after a new detection if no occlusion is detected. If occlusions are detected, a motion model (Kalman filter) is used instead of logarithmic search, until the matching error decreases.

## III. SPARSE TEMPLATE TRACKING WITH FEATURES FROM HOMOGENEOUS REGIONS

In this section, we are introducing our proposed method for hyper-real-time template-based person tracking using logarithmic search. First, we will give a short overview, followed by a detailed description of the single steps in the enclosed subsections.

At first, an initial position of the tracked object must be supplied. This is usually done by an object or person detector. The position is given as an rectangle enclosing the object. Using the enclosed region, a template for representing the object is generated. Afterwards, the template has to be relocated in consecutive frames with a template matching procedure, described in Sect. III-A. For searching the optimal template position, we use a local logarithmic search strategy, presented in Sect. III. To fulfill the special conditions needed for logarithmic search, we use a sparse template, generated by sampling color features from homogeneous regions (Sect. III-C). Since adequate features are essential for matching the template in consecutive frames, we evaluate different color spaces and distance metrics in Sect. III-D. To continuously update the template during the tracking process, we reinitialize the template after each new detection. In order to avoid ID-switches, the template is only reinitialized when no occlusion is detected. Therefore, we include a motion model and occlusion detection to improve the precision of the tracker (Sect. III-E). The whole algorithm described here is illustrated in Fig. 2.

### A. Template Matching

In template matching, the presence of a known object is searched by comparing the object template $U(m, n)$ with the scene $V$, where $m$ and $n$ is the size of the template. For matching the template $U(m, n)$ with a position $(p, q)$ in the scene, the following error function (Eq. 1) is defined.

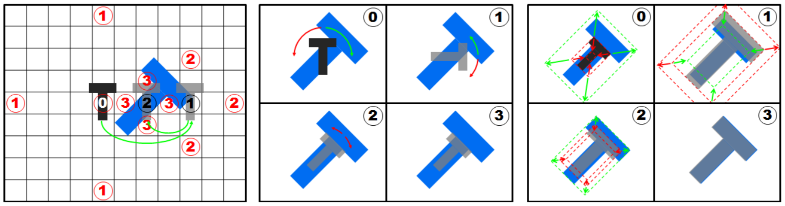


Fig. 3. Steps of the logarithmic search. The target is colored blue and the template is shown in grey. First, the translation is searchend (left) as described. Afterwards, the rotation (center) and the scale (right) is adapted.

$$err(p,q) = \sum_{i=1}^{m}\sum_{j=1}^{n} d(U(i,j), V(p+i, q+j)) \qquad (1)$$

where $d$ is a distance measure like the Euclidean or Manhattan distance. The position $(x,y)$ of $U(m,n)$ is calculated by searching the minimum error $err(p,q)$ (see Eq. 2).

$$(x,y) = \underset{p,q}{\operatorname{argmin}}(err(p,q)) \qquad (2)$$

Usually, not only the displacement, but also a new scale and rotation of the template is searched. Therefore, the search-space expands to $(x,y,\alpha,s)$, where $\alpha$ is the rotation and $s$ the scale of the template. Calculating the error function for every position and appearance of the template, using direct search, is very time consuming for HD-images.

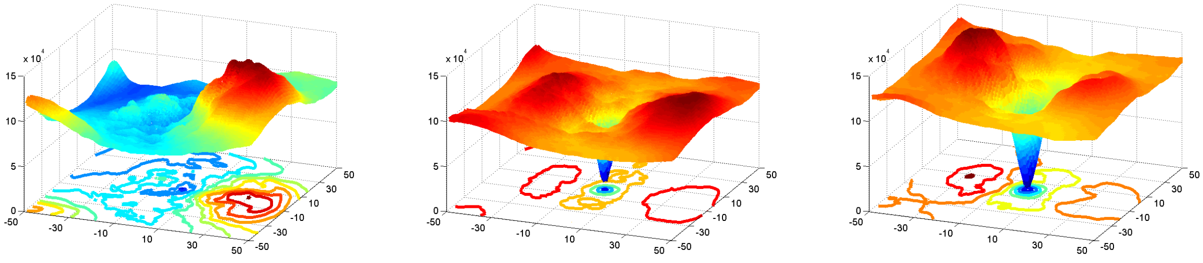
### B. Logarithmic Search

An efficient search strategy for direct search is sampling the search space logarithmically. This method reduces the search iterations dramatically. Using the last position $(x,y)$ of the object, the template is moved by the step width $p$ to the neighborhood positions. While in translation a $4N$ or $8N$ connected neighborhood $(N)$ is possible, the scale and rotation is limited to $2N$. For each neighbor position, the error is calculated using the defined error function (Eq. 1). The template is then moved to the position with the lowest error. This step is repeated until the position with the lowest error response is found, using the current step width. In the next iteration, the step-width $p$ is divided in half and the search is continued. The algorithm terminates after $p < w$, where $w$ is the threshold for the specific dimension.

Our proposed method uses the logarithmic search to find the translation (Fig. 3 left), rotation (Fig. 3 center) and scale (Fig. 3 right) parameters separately. This results in a runtime of $O(log(p_x \cdot p_y) + log(p_\alpha) + log(p_s))$, where $p$ is the maximum step-width for each dimension. It is also possible to perform a combined search of all dimensions. This is also evaluated in the experimental section.

### C. Generating the Sparse Template

The tracking quality of a template-based approach depends mainly on the choice of the template. A useful template must be a good representation of the object and needs to be robust during translation and perspective changes between two consecutive frames. Sparse templates can meet both of these criteria, if the right subset of features is chosen. Moreover, a sparse representation of the template has the advantage of matching only a subset of features of $U(m,n)$. Most state-of-the-art methods use descriptive features, as for example edges, textures, and interest points. While these features are robust during perspective changes, translation and scaling, the extraction is often time consuming. Furthermore, they often cannot be applied for tracking distant objects (no texture information) or fast objects (motion blur). An additional disadvantage of descriptive features is, that the template position cannot be searched with local search strategies, like logarithmic search. This is due to the cluttered search space of the error landscape in the local surroundings (see Figure 4(a)), which has the disadvantage that only



(a) edge features

(b) random features

(c) clustered features

Fig. 4. Error landscape for different feature selections on Seq. A of the "BoBoT" dataset [8] (see Sect. IV). Edge features or other interest points result in cluttered cluttered search space, as illustrated in (a). Random sampling (b) and features and clustered regions (c) have a smooth search space and do not violate the logarithmic search condition. Sampling from similar regions (c) results in a wider attractor basin and is therefore better suited for local search strategies

cost expensive search strategies can be applied for template matching.

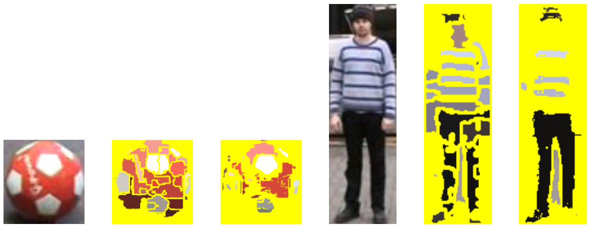
To achieve a smooth error function with a sparse representation of the template, we use a set $F$ of weak descriptive features $f_i$, sampled from homogeneous regions of the color space. Each feature $f_i$ contains the position $p_{x,y}$ relative to the template position and the color vector $\mathbf{c}_{\alpha,\beta,\gamma}$ of the used color space. We use the term ”weak descriptive” since one feature alone is not sufficient to describe the object. The appearance of the object is encoded by a set of these features and their spatial relations. Drawing the features from homogeneous regions enables us to use local search strategies, like logarithmic search, as the template error function response is continuous and smooth around the optimal position (Figure 4(b-c)). At the end of this subsection, we will discuss how descriptive such a sparse template is.

For finding homogeneous regions, we cluster the object to be tracked using well known methods from image processing. The features are sampled randomly within the found clusters, but not too close to the cluster borders. This approach is fast and ensures that only those features are selected which are homogeneous in the local surroundings. For clustering, we use a watershed approach and region growing (see Fig. 5 for the clustering results). These two methods only need a linear runtime of $O(M)$, where $M$ is the number of pixels to be clustered. For algorithmic details, it is referred to [15] and [16]. The cluster regions can also be used to estimate background regions, or regions very similar to the object. Assuming that an object is surrounded by background, a larger region is clustered around the template's initial position. All clusters which contain many pixels outside the initial (not enlarged) region are assumed as background and therefore excluded from sampling the features.

Fig. 4 illustrates the resulting error landscape when moving a template, built by the presented methods, in an area of fifty pixels around the initial position (translation on the $x$ and $y$ axis). The error landscape is a very good indicator in which range the logarithmic search is able to find a good matching position. Logarithmic search using descriptive features instead will fail due to the cluttered search space (Fig. 4(a)). Sampling the features randomly often gives a fair precision for logarithmic search (Fig. 4(b)), but placing the features into homogeneous regions using clustering (Fig. 4(c)) simplifies the search, because this allows for enlarging the step size for logarithmic search due to a wider attractor basin.

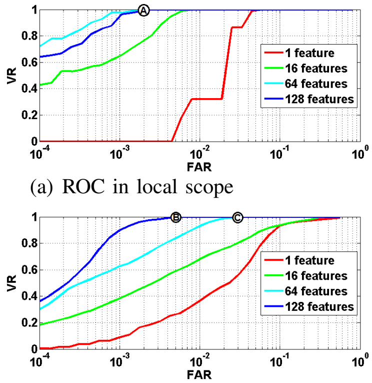
In the last part of this subsection, we are discussing how descriptive a sparse template is, which uses color features sampled from the clustered homogeneous regions. The template is evaluated using two experiments, on a challenging high definition sequence from person tracking with multiple persons. Experiment 1 evaluates how descriptive the template is on the initial frame in comparison to the local search area (area of 200 × 200 pixels around the center of the initial position. Experiment 2 focuses on how robust the template is during fifty consecutive frames for the full image. This results in 96,000,000 possible template positions. For evaluation we show the ROC Curve, which is commonly used for reidentification tasks. The experiments are repeated using a different number of sparse features (1, 16, 64, 128).

In experiment 1, the true positive positions (TP) are defined as a $10 \times 10$ area around the template's initial position (minimum area of the attractor basin). All other positions in the 200 × 200 search area are false positives (FP). Fig. 6(a) shows, that using more then 64 features results in a good template representation for the local search area (see marking (A)) since only 0.2% (FAR axis) of all FP positions (80 of 39,900) have a better or equal matching score than the worst matching position in the TP area. Experiment 2 evaluates the matching score of the template using 50 consecutive full frames. The TP positions are defined as a $10 \times 10$ area around the ground truth positions of the bounding box (for each of the 50 frames). All other possible template positions on each frame are FP. Fig. 6(b) illustrates (marking (B)) that even on this long sequence for 128 features only 0.5% (FAR axis) FP positions have a better or equal matching score than the worst



(a) Original (b) Watershed (c) Region Growing (d) Original (e) Watershed (f) Region Growing

Fig. 5. Homogeneous regions found by the proposed clustering algorithms, for initial hypothesis of Seq. A (a-c) and D (d-f) of the ”BoBoT” dataset [8]. The clusters are filled with the mean color of the region. Pixels that are excluded from feature sampling are displayed yellow. Watershed clustering results in more clusters but also permits sampling from non-homogeneous clusters, as for example the writing on the ball. Region growing clustering only permits sampling from homogeneous regions



(a) ROC in local scope

(b) ROC in global scope

Fig. 6. ROC curves (verification rate **VR**, false acceptance rate **FAR**) for local neighborhood (a) and full image on 50 consecutive frames (b); The FAR axis is logarithmically scaled.

matching position in the 50 attractor basins. With 64 features the number increases to 3% (marking (C)). The experiment also shows that using 16 or less weak descriptive features is not sufficent for a descriptive template.

### D. Color Spaces and Distance Metrics

Choosing an adequate color space and the belonging distance metric has a big influence on the precision and runtime of the clustering and matching procedures.

In our evaluation, we included four common color models: $RGB$, $HSV$, $CIELAB$, and the $Y$-channel of $YC_bC_r$. For a more detailed comparison and mathematical descriptions of the transformation from $RGB$ to the other color models, we refer to [17].

To compare colors, a distance metric is needed. The following distance measures are commonly used in real-time applications: The Manhattan distance ($d_1$), the Euclidean distance ($d_2$), and the maximum distance ($d_\infty$). Additionally, more complex distance measures are described in [18] and [19]. Since we need to calculate distances frequently, we only use the mentioned fast to compute metrics. To compare vectors of color pixels, we use the sum of absolute differences (SAD). Using the proposed template, this measure creates a continuous and smooth error landscape (Fig. 4), which is essential for efficient logarithmic search. We compare the different color spaces and distance metrics in the experimental section IV.

### E. Dealing with Occlusion

One difficulty in object tracking is dealing with occlusions. In template matching, the error of the best template match can be used as a good indicator for occlusion. A sudden increase of the error is often the result of a full or partial occlusion. Evaluating the region of the error increase in the template, as done in [20], gives a good estimation, which part of the template is covered. After masking the covered area, the template matching is continued with the remaining features. Although this simple strategy seems to be efficient, we use a different approach, since full occlusions can occur in our scenario and computational time has to be saved. Using a Kalman filter, we estimate the motion model of the tracked object. After detecting a sudden increase in the error of the template matching, the motion model is used to predict the next optimal position, instead of using the best template matching hypothesis (see Fig. 2, too). When the template matching error decreases again, the logarithmic search algorithm can be continued with the initial template. Occlusion detection and handling had influence on the following sequences in the experiments: Sequences F and I of the BoBoT-dataset [8] (Tab. III and Fig. 10), PETS 2009 dataset [21] (Fig. 9 A) and on our own dataset (Fig. 9 D). On the remaining sequences occlusion detection was performed, but since no significant occlusions were detected a correction was not necessary.

### F. Reducing the Search Space in Person Tracking

While tracking objects, the template position is usually adapted using translation $(x, y)$, scale $s$ and rotation $\alpha$. In the scenario of person tracking, the search dimensions can be reduced due to certain presumptions. In person tracking the rotation $\alpha$ does not need to be estimated, since people will only appear in an upright position. Additionally using calibrated cameras and the assumption that persons move on the ground and have a constant height, the search space can be reduced by omitting the scale $s$. The height of the template is estimated using the extrinsic parameters and the initial hypotheses, the scale is then adjusted depending on the translation. This is applicable on hard mounted cameras in mobile service robotics and surveillance scenarios with calibrated cameras.

| Seq. | Tracked Object | Challenges |
|---|---|---|
| A | ball | translation, rotation, and scale; fast speed changes |
| B | coffee mug | similar background and object color |
| C | juice box | fast speed changes, other objects close |
| D | person | perspective changes |
| E | person | partial occlusion |
| F | person | multiple full occlusions |
| G | rubiks cube | perspective changes and other objects close |
| H | toy | multiple lighting changes |
| I | person | very long track with multiple occlusions and perspective changes |

TABLE II

CHALLENGES OF TEST SEQUENCES

## IV. EXPERIMENTS

In this section, we present the quantitative and qualitative results of our tracking approach. To evaluate our approach, we used the public tracking data-set ”BoBoT” of the University of Bonn [8]. It contains nine sequences recorded at 25fps with a resolution of 320 × 240 pixels. The sequences cover different challenges of object tracking (Tab. II). Additionally, the ground truth is provided for each sequence. We compared our approach to five related state-of-the-art methods (see Fig. 1), evaluated in [8], using nine sequences of this dataset (Tab. III). The first tracking method is a simple histogram based method [6]. The second method is a component based tracker [7], which builds an object description of the surrounding feature maps from color and intensity. The remaining three methods are Haar-like center-surrounded feature based classifiers, introduced in [8].

We conducted four experiments to determine the influence of different parameters and configurations on the tracker. The first experiment analyses the usage of different distance metrics for each color space. Experiment 2 evaluates different ways for generating the sparse template. In experiment 3, we analyzed the usage of logarithmic search for translation in combination with scale, or the combination of scale and rotation. Experiment 4 evaluates how many sparse features are needed to robustly track the object and how the number of features influences the runtime. For a fair comparison to [8] and to show the robustness of the tracker, it is only initialized on the first frame using the ground truth. In real applications the tracker is initialized and updated by a detector. In order to evaluate the different setups, we run each experiment 100 times and calculated the average and variance of the overlap-join-ratio ($score = \frac{G \cap T}{G \cup T}$) between the bounding boxes of the ground truth ($G$) and the tracking hypothesis ($T$) [8]. In the

| Seq. | # Fr. | avarage score [%] | | | | | |
|---|---|---|---|---|---|---|---|
| | | [6] | [7] | [8](a) | [8](b) | [8](c) | our.<br>approach |
| A | 601 | **70.7** | 63.2 | 38.4 | 65.1 | 59.4 | 69.2 |
| B | 628 | 67.0 | 50.7 | 6.0 | 79.0 | 77.4 | **80.4** |
| C | 403 | 47.6 | 63.7 | 89.3 | 90.7 | **91.3** | 67.9 |
| D | 946 | 63.4 | 76.4 | 62.8 | 71.1 | 75.2 | **80.6** |
| E | 304 | 78.2 | 77.4 | 83.1 | 84.5 | **86.3** | 86.0 |
| F | 452 | 44.4 | 40.0 | 64.0 | 60.8 | **68.3** | 56.1 |
| G | 715 | 46.3 | 49.6 | 34.3 | 77.3 | 71.2 | **87.1** |
| H | 411 | 62.2 | 86.5 | 95.8 | 94.4 | 94.5 | **98.3** |
| I | 1016 | 68.9 | 47.6 | 49.0 | 75.0 | 56.3 | **77.7** |
| avg. | | 61.0 | 61.7 | 58.1 | 77.5 | 75.5 | **78.1** |

TABLE III

COMPARISON OF OUR APPROACH WITH CURRENT STATE-OF-THE-ART METHODS. IN SEQUENCES B, D, G, H, AND I, OUR APPROACH OUTPERFORMS CURRENT STATE-OF-THE-ART TRACKERS. IN SEQUENCES A, E AND F OUR APPROACH IS CLOSE TO THE BEST METHOD. ONLY IN THE SEQUENCES C AND F THE TRACKER PERFORMS AVERAGE. FIG. 10 SHOWS A VISUALIZATION OF THE TRACKING RESULTS.

last part of this chapter, we will also show examples of our tracking approach on other public and own data sets and on a mobile robot for real-time person tracking.

### A. Different Metrics and Color Spaces

In this experiment (Tab. IV), we evaluate the precision and runtime of different distance metrics and color spaces. While in the Y channel only the absolute distance is possible, we evaluate Manhattan, Euclidean and Maximum distance in the RGB and HSV color space for the calculation of the $SAD$. In the CIELAB color space the Euclidean distance is used, since it is optimized for it.

| Seq. | avarage score [%] | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Y | RGB | RGB | RGB | HSV | HSV | HSV | CIE<br>LAB |
| | | $d_1$ | $d_2$ | $d_\infty$ | $d_1$ | $d_2$ | $d_\infty$ | $d_2$ |
| | mean | mean | mean | mean | mean | mean | mean | mean |
| A | 45.3 | 55.9 | 54.0 | 53.1 | 63.3 | **69.2** | 66.6 | 39.4 |
| B | 75.1 | 78.1 | 78.1 | 79.0 | 78.6 | 79.8 | 79.2 | **80.4** |
| C | 55.2 | 62.9 | 61.8 | 61.8 | **67.9** | 66.9 | 66.6 | 66.1 |
| D | 74.7 | 78.0 | 80.1 | **80.6** | 66.2 | 78.4 | 79.2 | 70.0 |
| E | 85.9 | 81.3 | **86.0** | 85.9 | 77.3 | 83.7 | 82.1 | 85.9 |
| F | **56.1** | 48.8 | 54.8 | 39.7 | 8.8 | 49.6 | 32.1 | 26.2 |
| G | 76.5 | 80.4 | 80.7 | 78.2 | 81.9 | **82.1** | 80.9 | 80.4 |
| H | **96.7** | 96.3 | 96.3 | 96.4 | 96.0 | 95.7 | 95.9 | 96.0 |
| I | **77.7** | 77.2 | 77.5 | 74.8 | 59.5 | 66.3 | 66.0 | 57.0 |
| avg. | 71.5 | 73.2 | 74.4 | 72.2 | 66.6 | **74.6** | 72.1 | 66.8 |
| avg.<br>rtpf<br>in ms | 1.10 | 2.37 | 2.77 | 2.58 | 3.91 | 4.12 | 3.84 | 15.36 |

TABLE IV

TRACKING SCORE FOR DIFFERENT COLOR SPACES AND DISTANCE MEASURES AND AVERAGE RUNTIME IN MS FOR ITERATING ONE FRAME (RTPF)

The Y channel is the fastest of all setups, but only has a fair precision on most data sets. CIELAB is the computational most expensive color space and does not perform better on most datasets. Both RGB and HSV perform well with all three distance measures. For HSV the Euclidean distance seems to be the most reliable. In RGB, all three distance metrics perform well. Against this background, we recommend the RGB color space, since it is the most common one and does not need further transformation steps. For the RGB color space, we decide in favor of the Manhattan distance, since it is the fastest with equally good results.

### B. Initialization

We evaluated the different techniques for sparse template initialization in this experiment (Tab. V). Creating a template with random sampling seems to be sufficient for most sequences. Nevertheless, on objects with a lot of texture information, like sequence C and G, this kind of initialization fails. Clustering the object for finding homogeneous regions for sampling the features, with watershed or region growing, always results in a good sparse template representation. Since region growing is a little more stable, we prefer this method. The runtime of the different initializations is shown in the last row of Tab. V;

| Seq. | avarage score [%] | | | | | |
|---|---|---|---|---|---|---|
| | random | | whatershed | | region-<br>growing | |
| | mean | var. | mean | var. | mean | var. |
| A | 50.0 | 2.96 | 49.7 | 0.82 | **55.9** | 0.61 |
| B | 77.8 | 0.14 | **79.2** | 0.01 | 78.1 | 0.02 |
| C | 41.4 | 0.19 | 52.5 | 1.18 | **62.9** | 0.19 |
| D | **80.5** | 0.07 | 79.2 | 0.31 | 78.0 | 0.19 |
| E | 85.1 | <0.01 | **85.7** | <0.01 | 85.5 | <0.01 |
| F | 33.9 | 0.09 | 51.7 | 0.48 | **54.8** | 1.15 |
| G | 72.6 | 0.10 | 76.6 | 1.17 | **80.4** | 0.89 |
| H | **98.3** | <0.01 | 95.7 | <0.01 | 96.3 | <0.01 |
| I | 71.6 | 0.29 | 75.4 | 0.33 | **77.2** | 0.19 |
| avg. | 67.9 | | 71.8 | | **74.3** | |
| avg.<br>runtime | 15ms | | 46ms | | 46ms | |

TABLE V

TRACKING SCORE AFTER DIFFERENT TEMPLATE INITIALIZATIONS AND AVERAGE RUNTIME FOR INITIALIZATION

### C. Combined Dimension Search

The first part of this experiment investigates only searching the translation of the template. As expected, only searching the translation performs worse on sequences with dynamic cameras. In the second part of the experiment, we evaluated the usefulness of combining search dimensions with each other. Our usual setup is to search translation, rotation and scale separately. The combination of translation with scale

| Seq. | avarage score [%] | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | separate | | only<br>translation | | translation+<br>scale | | scale+<br>rotation | |
| | mean | var. | mean | var. | mean | var. | mean | var. |
| A | **59.9** | 0.61 | 58.4 | 0.36 | 40.9 | 2.26 | 52.0 | 1.08 |
| B | **78.1** | 0.02 | 72.96 | <0.01 | 68.7 | 0.16 | 55.1 | 2.24 |
| C | **62.9** | 0.19 | 43.10 | 0.12 | 34.3 | 0.37 | 37.7 | 0.10 |
| D | **78.0** | 0.19 | 61.70 | 0.95 | 69.9 | 0.75 | 55.3 | 0.02 |
| E | **86.1** | 0.08 | 84.57 | <0.01 | 78.1 | 1.75 | 65.1 | 2.29 |
| F | **54.8** | 1.15 | 42.37 | 1.32 | 23.0 | 0.88 | 0.4 | <0.01 |
| G | **80.4** | 0.89 | 70.96 | 0.23 | 59.5 | 7.82 | 79.3 | 0.33 |
| H | 96.3 | 0.01 | **98.59** | <0.01 | 95.8 | 0.02 | 60.0 | 4.25 |
| I | **77.2** | 0.19 | 51.29 | 0.38 | 64.4 | 1.22 | 46.5 | 2.30 |
| avg. | **73.88** | | 64.88 | | 59.40 | | 50.15 | |

TABLE VI

TRACKING SCORE FOR DIFFERENT COMBINATIONS OF SEARCH DIMENSIONS estimates the scale while searching for the new transition of the template. The rotation is adapted separately. Combining scale and rotation, first the translation is searched, then the rotation is adapted while scaling the template. Combining the search dimensions in this form increases the search space from $p_x \cdot p_y + p_\alpha + p_s$ (where $p$ is the maximum step width for the dimension) to $p_x \cdot p_y \cdot p_s + p_s$ for translation and scale and $p_x \cdot p_y + p_\alpha \cdot p_s$ for scale and rotation. Tab. VI shows, that a larger search space decreases the precision dramatically. Therefore, it is advisable to keep the search space as small as possible.

### D. Number of Features

In this experiment, we evaluated the influence of the number of features in the sparse template set (Tab. VII). In most scenes, a higher number of features increased the precision of the tracker. Particularly, on objects with small clusters, a higher number of features is helpful. Both setups with 400 and 600 features show a good performance. Since the difference of precision between 400 and 600 features is very small, we use a template of 400 features to save computational time.

| Seq. | avarage score [%] | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 100 | | 200 | | 400 | | 600 | |
| | mean | var. | mean | var. | mean | var. | mean | var. |
| A | 54.4 | 1.04 | 55.7 | 0.68 | **55.9** | 0.61 | 54.6 | 0.51 |
| B | 76.1 | 0.14 | 77.5 | 0.04 | **78.1** | 0.02 | 77.8 | 0.03 |
| C | 56.2 | 0.87 | 60.6 | 0.44 | 62.9 | 0.19 | **63.3** | 0.11 |
| D | 73.6 | 0.58 | 74.9 | 0.76 | 78.0 | 0.19 | **78.3** | 0.14 |
| E | 72.7 | 2.26 | 77.1 | 1.94 | 81.3 | 1.08 | **82.8** | 0.76 |
| F | 39.8 | 1.68 | 43.1 | 1.94 | **54.8** | 1.15 | 51.6 | 0.78 |
| G | 73.6 | 1,24 | 76.8 | 1,1 | 80.4 | 0.89 | **87.1** | 0.75 |
| H | 94.9 | 0.02 | 95.7 | 0.01 | 96.3 | <0.01 | **96.6** | <0.01 |
| I | 69.0 | 0.18 | 71.2 | 0.22 | **77.2** | 0.19 | 72.1 | 0.23 |
| avg. | 67.82 | | 70.27 | | **73.88** | | 73.78 | |
| avg. rtpf | 0.77ms | | 0.80ms | | 2.37ms | | 3.60ms | |

TABLE VII

TRACKING SCORE FOR DIFFERENT NUMBER OF USED FEATURES AND AVERAGE RUNTIME IN MS FOR ITERATING ONE FRAME (RTPF)

### E. Experiments on the Robot Platform CompanionAble

In this experiment, we address the problem of visual person following (visual servoing) in mobile robotics. For a first evaluation, we use following experimental setup. The robot (Fig. 7) is standing in our robotics lab and tracking a person simultaneously with a laser based leg detector and the proposed method for visual tracking. The visual tracker is initialized using a predefined region on the first frame. We evaluate three different sequences (person is standing still, person is moving with normal speed and person is moving with sudden direction and speed changes) with a length of 60 seconds each. The quality of the vision-based tracker is compared to the laser based leg detector, since the later provides very good ground truth. As comparative measure, the Euclidean distance and variance in world-coordinates is used. It is important to mention, that it is difficult to transform image coordinates into world-coordinates by just using a bounding box and the extrinsic parameters of a camera mounted on the robot at a low height (about 110cm - see Fig. 7), since only slight changes in position and scaling of the bounding box lead to a significant error in world coordinates. Despite that, the tracker performs well on all three sequences showing a maximum average error of 35cm (Tab. 8) on the third sequence. This is still sufficient for a robust vision-based person following. It is remarkable, that while tracking, our approach required less than one percent of the robots computational resources. This first experiment shows, that our approach is accurate enough to realize visual based person following for scenarios where no laser range data is present.

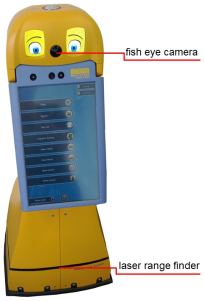


Fig. 7. CompanionAble robot (SCITOS G3 platform); For visual servoing we use the nose camera, which has a resolution of $1600px \times 1200px$ and a viewing angle of about 180°

| | Standing | Normal | Rapid |
|---|---|---|---|
| mean | 0.001m | 0.05m | 0.35m |
| var. | <0.001 | 0.004 | 0.23 |

Fig. 8. Mean distance and variance between our approach and laser based leg detection used as reference tracker while the test person stands still, during normal movement, and with rapid speed and direction changes. The mean Euclidean difference and the variance are determined over all frames in the sequence.

### F. Experiments on Other Datasets

As supplementary material, we attached videos of our experiments on the ”BoBoT” dataset [8] and additional datasets, which are often used in computer vision for benchmarking new algorithms. Since no person detection is used in these cases, the tracker was initialized per hand on the first frame of the sequence. Sequence A of Fig. 9 is included in the PETS2009 dataset [21]. Three persons are robustly tracked by our algorithm in hyper-real-time dealing with diverse occlusions and perspective changes. Sequence B is part of the Motinas Face Tracking dataset [22]. Sequence C is part of the PETS2006 dataset [23] for person tracking.

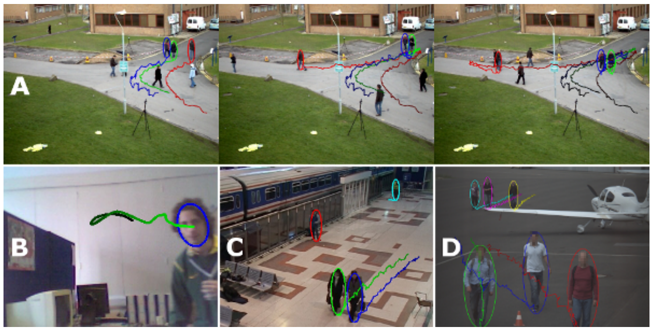
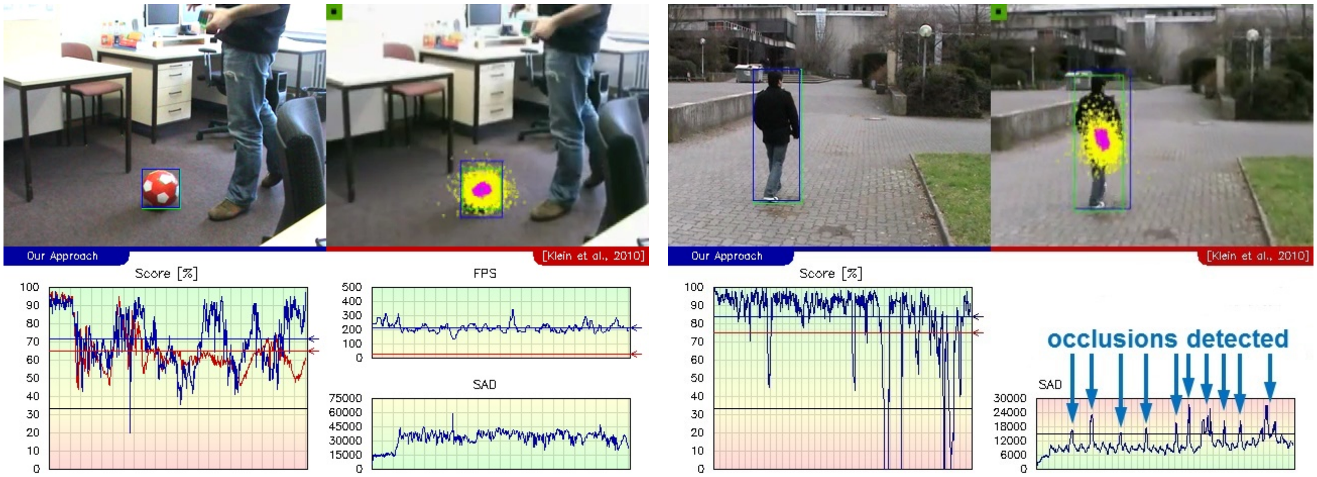


Fig. 9. Visualization of additional datasets. All sequences are attached as supplementary material.



(a) Ball

(b) Person in park

Fig. 10. Comparison to [8] on BoBoT-Dataset. Graphs show score, runtime and SAD of our approach (blue) vs. [8] (red). Additionally, in the right image the occlusion detection by sudden increases of the SAD values is highlighted. A video of all scenes is attached as supplementary material.

Here our algorithm simultaneously tracked four persons in hyper-real-time. The last experiment (D) shows a sequence of our own dataset where six persons are tracked in hyper-real-time even on HD-images (1600 × 1200), dealing with a broad spectrum of occlusions. For the last experiment and sequence, a calibrated camera was employed.

## V. Conclusion

We presented a new template based tracking approach for hyper-real-time object and person tracking. The innovation of our approach is the automatic initialization of a small set of features, sampled from suitable homogeneous regions on the object to be tracked. Not using descriptive feature points enables us to use logarithmic search as local search strategy. Disregarding the image size, our approach finds the optimal matching position with only few comparisons. The qualitative comparison shows that our method performs equal or better to current state-of-the-art methods in real-time object tracking, while being up to 40 times faster. Additionally, we showed that this approach is robust enough to realize visual person following (visual servoing) on a mobile robot. In our future work, we will implement the presented method in combination with laser range finders [24], [25] and sensor fusion algorithms into a general tracking and evaluation framework for robotic applications.

## References

[1] H. Gross, H. Boehme, C. Schroeter, S. Mueller, A. Koenig, E. Einhorn, C. Martin, M. Merten, and A. Bley, "Toomas: Interactive shopping guide robots in everyday use - final implementation and experiences from long-term field trials," in *IROS*, 2009, pp. 2005–2012.

[2] M. Eisenbach, A. Kolarow, K. Schenk, K. Debes, and H. Gross, "View invariant appearance-based person reidentification using fast online feature selection and score level fusion," in *9th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, 2012.

[3] H. Gross, C. Schroeter, S. Mueller, M. Volkhardt, E. Einhorn, A. Bley, C. Martin, T. Langner, and M. Merten, "Progress in developing a socially assistive mobile home robot companion for the elderly with mild cognitive impairment," in *IROS*, 2011, pp. 2430–2437.

[4] A. Yilmaz, O. Javed, *et al.*, "Object tracking: A survey," *CSUR*, vol. 38, no. 4, pp. 1–45, 2006.

[5] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *TPAMI*, vol. 24, no. 5, pp. 603–619, 2002.

[6] P. Perez, C. Hue, *et al.*, "Color-based probabilistic tracking," in *ECCV*, 2002, pp. 661–675.

[7] S. Frintrop, "General object tracking with a component-based target descriptor," in *ICRA*, 2010, pp. 4531–4536.

[8] D. Klein, D. Schulz, *et al.*, "Adaptive real-time video-tracking for arbitrary objects," in *IROS*, 2010, pp. 772–777.

[9] H. Zhou, Y. Yuan, *et al.*, "Object tracking using sift features and mean shift," *CVIU*, vol. 113, pp. 345–352, 2009.

[10] D. Wagner, G. Reitmayr, *et al.*, "Real-time detection and tracking for augmented reality on mobile phones," *TVCG*, vol. 16, pp. 355–368, 2010.

[11] J. Shi and C. Tomasi, "Good features to track," in *CVPR*, 1994, pp. 593–600.

[12] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI*, 1981, pp. 674–679.

[13] P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," in *CVPR*, 1997, pp. 21–27.

[14] A. Jain, *Fundamentals of Digital Image Processing*. Prentice-Hall, London, 1989, pp. 404–406.

[15] S. Beucher, "Watersheds of functions and picture segmentation," in *ICASSP*, 1982, pp. 1928–1931.

[16] A. Tremeau and N. Borel, "A region growing and merging algorithm to color segmentation," *PR*, vol. 30, pp. 1191–1203, 1997.

[17] H. Cheng, X. Jiang, *et al.*, "Color image segmentation: advances and prospects," *PR*, vol. 34, pp. 2259–2281, 2001.

[18] D. Androutsos, K. Plataniotiss, *et al.*, "Distance measures for color image retrieval," in *ICIP*, 1998, pp. 770–774.

[19] A. Ekin and A. Tekalp, "Robust dominant color region detection and color-based applications for sports video," in *ICIP*, 2003, pp. 21–24.

[20] Y. Pan and B. Hu, "Robust occlusion handling in object tracking," in *CVPR*, 2007, pp. 1–8.

[21] PETS2009, "Performance evaluation of tracking and surveillance," http://www.cvg.rdg.ac.uk/PETS2009/index.html, 2009.

[22] E. Maggio and A. Cavallaro, "Hybrid particle filter and mean shift tracker with adaptive transition model," in *ICASSP*, 2005, pp. 221–224.

[23] PETS2006, "Performance evaluation of tracking and surveillance," http://www.cvg.rdg.ac.uk/PETS2006/index.html, 2006.

[24] K. Schenk, M. Eisenbach, A. Kolarow, K. Debes, and H.-M. Gross, "Comparison of laser-based person tracking at feet and upper-body height," in *KI*, 2011, pp. 277–288.

[25] K. Schenk, A. Kolarow, M. Eisenbach, K. Debes, and H.-M. Gross, "Automatic calibration of multiple stationary laser range finders using trajectories," in *AVSS*, 2012.