2D-Object Tracking Based on Projection-Histograms

S. Huwer and H. Niemann

FORWISS - Knowledge-Processing Research Group,
Bavarian Research Center for Knowledge-Based Systems,
Am Weichselgarten 7,
D-91058 Erlangen,
{huwer,niemann}@forwiss.de

Abstract. Image-sequence analysis for real-time applications requires high quality and highly efficient algorithms for tracking as there is no time to do the costly object recognition each time a new image is captured. Tracking with projection histograms revealed amazing results compared with standard correlation methods. Trackers based on projection histograms performed 31% up to 211% better than the reference methods on a common test set. The new template-based method relying on projection histograms (RPH) is described and compared with two commonly known template based methods namely the normalized cross-correlation (NCC) and displaced-frame-distance (DFD) methods. The input to the system consists of live or recorded video data where filterbased preprocessing can be applied before tracking in order to enhance features such as edges, textures etc. A region of interest (ROI) is taken as a template for tracking. In subsequent images tracking exploits a Kalman-filtered local search in order to renew correspondence between the object template and the new object location. Comparative tests were performed with real-live image-sequences taken in underground stations. Tracking with projection histograms outperformed tracking by NCC and DFD on grey-level image-sequences as well as on edge-enhanced image-sequences. Even the worst chosen parameter set for tracking by the new RPH method resulted in better tracking as with the best ones for both NCC and DFD.
keywords: **template matching, 2D-tracking, real-time tracking**

1 Tracking in Realtime

As computers get faster, more and more complex problems in image analysis become practically solvable. Yet image-sequence analysis, especially in real-time environments, often stride the frontieres of feasibility. Visual surveillance systems are but one example of an emerging branch where hard time-constraints coupled with the desire for low-cost components still induce the need for sophisticated image-processing methods. In the newly founded research project "Intelligent Platform Station"[1] static cameras placed at various locations on the underground platform produce video streams in which the detection and tracking of passengers represents an important task on the way to an automatic dispatch of the underground-trains. Information on the motion of passengers is to be gathered and analyzed for a detailed security check of the platform. Obviously **real-time** analysis of the image sequence is necessary for the passengers safety. The subtask of passenger safety monitoring can be divided into the object localization and object tracking phases. The object tracking is responsible for producing reliable information on an object's movements in an image-sequence.

Problems arise from the changes of the object's appearence during the monitoring process, e.g. due to varying lighting conditions or movements of the object or of the camera itself.

With the projection-histogram method as presented in this paper, tracking of ROIs in the 2D image plane can be performed even in the presence of highly textured and changing background.

In the following section first the related research will be shortly described. The basic principles of 2D-template tracking are explained and semantically divided in a template matching phase and a tracking phase. Both are explained subsequently. The projection histogram templates are then explained in detail in an chapter of their own. Then the principles of the tracking mechanisms used for the comparative tests are explained and two important quality measures are derived. After describing the test image sequences, the results are given. It must be noted that for all of the tests that were performed, the same tracking methods were used and only the template-representation technique was changed. Anticipating just one result: The projection histograms performed in all test cases better than the reference methods. The computation time for two of the three tested projection histogram methods (differing in the number of histogram bins) were even superior to the reference methods.

2 Related Research

Tracking of objects in the 2D image plane can be carried out in a variety of possible ways. Modelbased trackers exploit previously stored explicit knowledge about interesting objects. [1] demonstrates a simple head-tracker that relies on the elliptical shape of human heads. [7] uses a 3D-volumetric model of persons in

[1] funded by "Free State of Bavaria"

order to track people in an industrial environment. Finding appropriate model parameters and the right modelling description for the objects is a challenging task, which must be repeated for each new class of objects to track.

The active contour approach utilizes physics in order to give model-shapes intrinsic properties such as stiffness or curviness [6]. An Image is then interpreted as a greylevel-landscape onto which the model-shapes are adapted by deforming them according to their model properties. A fast extension to "active contours" gives [3] with "active rays", where the active contours are replaced by rays of variable length radiating from the center of gravity of the object. The endpoints of the rays describe the contour of the object. Common to these energy-based methods is their neglect of object features inside the object. So occlusions of similar objects usually end up in merging both objects.

The above mentioned methods more or less make use of prior object knowledge which might not always be available. Another category of tracking methods is solely based on 2D image-regions (templates) and can thus be used to track without any object specific knowledge. Among these, both the cross-correlation (e.g. *NCC*) and the frame-distance methods (e.g. *DFD*) are most commonly used. [5] gives a comprehensive overview of correlation and frame-distance methods. It is pointed out that 2D-window tracking with small inter-frame displacements can be accomplished. Still tests revealed that the new tracking method based on projection histograms, which is described in the next section, can track templates significantly longer and with a higher accuracy according to the given quality measures.

3 Problems and Principles of Template Tracking

Template-based methods can be described according to the following common features (see also fig. 1):

1. **Initialization:**
   The initial template region must be localized previously to tracking by an object-localization method and an internal template representation is computed depending on the chosen template-matching method.
2. **Tracking:**
   (a) The template tracker searches in the next image for the best matching image-region, which defines the new position of the object.
   (b) The new object postion may be denoised by applying a smoothing filter, that takes the last measurements into account.
   (c) The last step of tracking may be the invocation of a motion-prediction method in order to predict the position of the object in the next image.

Problems in 2D-tracking arise when the tracked object changes its appearance as a result of:

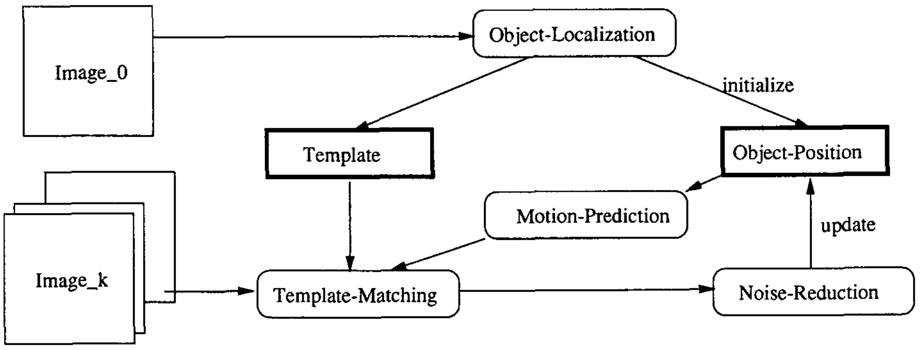

Fig. 1. Template-Tracking

- scaling or rotation
- a change of the object's 3D posture which results in a deformed object appearance in the 2D image-plane
- a large shift of the object in the image

Additional problems arise when lighting conditions or camera parameters change, such as lens-parameters or aperture.

Tracking with "Projection Histograms" (RPH) aims at solving the problem of changing object location with small changes in object appearance. So RPHs are compared with the two reference methods (NCC and DFD) , which aim at solving the same task.

3.1 Notation

For an easier understanding of the following formal description a few definitions are given first:

- **Bounded sets of natural numbers:**

$$\mathcal{N}_{j,k} := \{j, j+1, ..., j+k-1\} \tag{1}$$
$$\mathcal{N}_k := \mathcal{N}_{0,k} \tag{2}$$

- **Sets** :
  - The cardinality of a set S is given as $\mid S \mid$.
  - 2^S denotes the powerset of S
- **Cartesian set product:** $\mathcal{N}^j := \underbrace{\mathcal{N} \times ... \times \mathcal{N}}_{j \text{ times}}$
- An **Image** f or **Template t** $f : \mathcal{N}_m \times \mathcal{N}_n \longrightarrow \mathcal{N}_g$, with m rows, n columns and g image values (grey values).
- The **size of the template** is given by its height h and its width w.
- A **region of interest** $\mathcal{R}_{h,w}$ positioned at $(u, v) \in \mathcal{N}_m \times \mathcal{N}_n$ of width w and height h has the domain $\mathcal{R}_{h,w}(u, v) := \mathcal{N}_{u,h} \times \mathcal{N}_{v,w}$

3.2 Template Matching by *NCC* and *DFD*

While NCC represents a measure of similarity which measures correlation, the DFD is a measure of distance between an image f and a given template t of the size h and w. Both can be motivated by simple distance measures on image regions.

The cross-correlation term CC can be deduced from the squared euclidean distance measure $\delta^{(2)}$:

$$\delta^{(2)}_{f,t}(u,v) = \sqrt{\sum_{(x,y)\in\mathcal{R}_{h,w}(u,v)} [f(x,y) - t(x-u, y-v)]^2} \tag{3}$$

Where (u,v) positions the ROI $\mathcal{R}_{h,w}(u,v)$.

Expanding this term leaves two squared summands and the negated doubled cross-correlation term:

$$CC_{f,t}(u,v) = \sum_{(x,y)\in\mathcal{R}_{h,w}(u,v)} f(x,y)t(x-u, y-v) \tag{4}$$

Since CC varies strongly with the image energy and the size of the template, it is normalized to:

$$NCC^{\star}_{f,t}(u,v) = \frac{CC_{f,t}(u,v)}{\sqrt{\sum_{(x,y)\in\mathcal{R}_{h,w}(u,v)} f^2(x,y)}\sqrt{\sum_{(x,y)\in\mathcal{R}_{h,w}(0,0)} t^2(x,y)}} \tag{5}$$

Geometrically $NCC^{\star}$ can be interpreted as the cosine of the angle between the vectorized template and the current image region. Thus $NCC^{\star}$ will be near 1 for very similiar vectors and smaller otherwise.
As we consider the template-matching in this article as a minimization problem on distance measures, $NCC^{\star}$ is transformed into a distance measure NCC by $NCC_{f,t}(u,v) := 1 - NCC^{\star}_{f,t}(u,v)$.

The second chosen reference method for measuring the distance between t and a given image-region located at (u,v) utilizes the simpler distance measure $\delta^{(1)}$, which is also known as city-block metric :

$$\delta^{(1)}_{f,t}(u,v) = \sum_{(x,y)\in\mathcal{R}_{h,w}(u,v)} \mid f(x,y) - t(x-u, y-v) \mid \tag{6}$$

This can be computed very quickly on as one saves the computation time for the squares and squareroots. In order to provide some independence of the template-size, the $\delta^{(1)}$ measure is normalized by the area $h * w$ of the template. This gives DFD:

$$DFD_{f,t}(u,v) = \frac{\delta^{(1)}_{f,t}(u,v)}{h * w} \tag{7}$$

The next section explains the new template matching technique based on projection histograms.

4 Projection 2D-Histogram-Matching

Both, the *NCC* and *DFD* do not use any special coding or transformation for template representation. A given ROI of an image immediately represents its own template. The projective histograms (*PH*) are based on a special coding of the template before measuring the distance between a template and a given ROI.

Computing the projective histogram of a ROI results in a vector containing all histograms of the columns and all histograms of the rows of the ROI. This vector is then used for matching either two given ROIs or a template with a ROI.

In order to correctly specify the computation of the histogram a **quantization-function** ϑ is needed which transforms the g image-values into b bins of the histogram:

$$\vartheta : \mathcal{N}_g \longrightarrow \mathcal{N}_b \tag{8}$$

Three different quantization functions were used for the tests:

1. Number of bins is equal to the number of greylevels: $\vartheta_I(i) := i$
2. Number of bins is 8: $\vartheta_8(i) := i \quad \text{div} \quad 8$
3. Number of bins is 2: $\vartheta_2(i) := \begin{cases} 0 & ; \quad i < 32 \\ 1 & ; \quad i \geq 32 \end{cases}$

4.1 Absolute Projection-Histograms

In order to get the projection-histogram of a given image-region the related histograms of all region columns and all region rows are computed. The combination of those row- and column histograms gives the projection histogram for the given region. Thus the **absolute projection-histogram** transformation $\overline{\Phi}_f$ applied to a region $\mathcal{R} := \mathcal{R}_{h,w}(u,v)$ of the image f can be defined as:

$$\overline{\Phi}_{f,\mathcal{R}} : \mathcal{N}_{h+w} \longrightarrow \mathcal{N}^b \text{ with } \overline{\Phi}_{f,\mathcal{R}}(i) := (\overline{\varphi}_{f,\mathcal{R},0}(i), ..., \overline{\varphi}_{f,\mathcal{R},b-1}(i)) \tag{9}$$

and

$$\overline{\varphi}_{f,\mathcal{R},k}(i) := \begin{cases} \overline{\alpha}_k(i) & ; \quad i < h \quad \text{“row-histogram-entry”} \\ \overline{\beta}_k(i-h) & ; \quad i \geq h \quad \text{“column-histogram-entry”} \end{cases} \tag{10}$$

Each entry $\overline{\varphi}_{f,\mathcal{R},k}(i)$ gives the corresponding row- or column histogram entry which is defined by $\overline{\alpha}$ for the rows and $\overline{\beta}$ for the columns.

Finally the basic transformations to get row-histograms $\overline{\alpha} : \mathcal{N}_b \longrightarrow \mathbb{N}$ and column-histograms $\overline{\beta} : \mathcal{N}_b \longrightarrow \mathbb{N}$ are specified:

$$\overline{\alpha}_k(i) := |\ \{(x,y) \in \mathcal{R} \mid \vartheta(f(x,y)) = k \wedge x = i\}\ | \tag{11}$$

$$\overline{\beta}_k(i) := |\ \{(x,y) \in \mathcal{R} \mid \vartheta(f(x,y)) = k \wedge y = i\}\ | \tag{12}$$

Thus $\overline{\alpha}_k(i)$ gives the number of pixels in row i whose pixel-values are mapped onto k by the quantization function ϑ. $\overline{\beta}_k(i)$ analogously counts the number of entries in column i whose pixel-values are mapped onto k via the quantization function ϑ.

So $\overline{\Phi}_{f,\mathcal{R}}$ can also be written as :

$$\overline{\Phi}_{f,\mathcal{R}} = (\overline{\alpha}(0), ..., \overline{\alpha}(h-1), \overline{\beta}(0), ..., \overline{\beta}(w-1)) \tag{13}$$

4.2 Relative Projection-Histograms

For a more scale-invariant version of the projection histograms, the **relative projection-histogram** transformation Φ is given with $\overline{\Phi}$ when exchanging all absolute $\overline{\alpha}$ and $\overline{\beta}$ with their relative correspondents $\alpha : \mathcal{N}_b \longrightarrow [0,1]$ and $\beta : \mathcal{N}_b \longrightarrow [0,1]$ respectively. α can thus be defined by dividing it by the length of a row. β is divided by the length of a column:

$$\alpha := \frac{\overline{\alpha}_k(i)}{w} \text{ and } \beta := \frac{\overline{\beta}_k(i)}{h} \tag{14}$$

The rest of the paper considers exclusively relative projection-histograms (RPH) instead of the absolute projection-histograms (PH).

In order to specify matching with (RPH) a distance measure is needed for optimizing on. This is given as a squared Euclidian measure $\delta_{\Phi}^{(2)}$ (same as in the derivation for the NCC-Matching see formula 3).

$$RPH_{f,t}(u,v) := \frac{1}{w+h} \sum_{i \in \mathcal{N}_{w+h}} \delta^{(2)}(\Phi_{f,\mathcal{R}_f}(i), \Phi_{t,\mathcal{R}_t}(i)) \tag{15}$$

$$\text{with } \delta^{(2)}(c,d) := \sqrt{\sum_{j \in \mathcal{N}_b} (c_j - d_j)^2} \tag{16}$$

$$\text{and } \mathcal{R}_f := \mathcal{R}_{h,w}(u,v) \tag{17}$$

$$\text{and } \mathcal{R}_t := \mathcal{R}_{h,w}(0,0) \tag{18}$$

There are three important properties of RPHs worth noting:

- The chosen number of bins of the histogram is crucial for the computation times and with a low number of bins a high data reduction is achieved. Thus Matchung can be done more efficiently.
- The RPH-transformation is a oneway transformation only. There is no inverse transformation for general RPHs.
- The complexity of RPH-transformation is linear with the area of the ROI $\mathcal{R}$, since each pixel of $\mathcal{R}$ must be looked at only once to set the right values for α and β.

Matching templates is but a first step towards tracking a template over a sequence of input images. The second step is the tracking method which is described in the following section.

5 Tracking 2D-Templates

Given an Image f_i at the time i and the position (u_i, v_i) of the template t in f_i: Once the position (u_0, v_0) is known as the position of the template t in the image f_i, the problem of tracking can be described as finding the correct position (u_{i+1}, v_{i+1}) for this template in the image f_{i+1}. Provided that some knowledge on the maximal disparity between two successive template positions exists, it is reasonable to limit the template matching to a specified region of about this size.

5.1 Local Search for the Best Match

The local search for the best template match in the next image is limited to a search region $\mathcal{S} := \mathcal{S}_{w_s,h_s}(u'_{i+1}, v'_{i+1})$ of width w_s and height h_s which is centered on the expected position (u'_{i+1}, v'_{i+1}) of the template. Given a template matching strategy Ψ, tracking with respect to Ψ can be described as a minimization problem, where one minimizes the template distance over the search region S.

Tracking a template can now be defined by a search function Γ that gives the point $(u_{i+1}, v_{i+1}) \in \mathcal{S}$ with the smallest distance between image-region and the template:

$$\begin{aligned}
\Gamma_{f,t,\Psi} &: 2^{\mathcal{N}_m \times \mathcal{N}_n} \longrightarrow \mathcal{N}_m \times \mathcal{N}_n && (19)\\
\Gamma_{f,t,\Psi}(\mathcal{S}) &:= (x, y) && (20)\\
&\text{with } \Psi_{f,t}(x,y) = \min_{(x,y)\in\mathcal{S}} \Psi_{f,t}(x,y) && (21)\\
&\text{and } \Psi \in \{NCC, DFD, RPH\} && (22)
\end{aligned}$$

In cases where more than one (x, y) with the same minimal matching distance Ψ exist, such that a unique minimum cannot be identified, a **conflict resolution** strategy is needed. In this work the conflict resolution was chosen to depend on the distance between the new point (u_{i+1}, v_{i+1}) and the expected position (u'_{i+1}, v'_{i+1}). One expects this distance to be small, so in the case of a conflict we choose (u_{i+1}, v_{i+1}) not only to minimize Ψ but also to minimize the distance to (u'_{i+1}, v'_{i+1}). Other applications might need different heuristics to resolve this conflict.

5.2 Estimation of the Template-Position

There are various possibilities to estimate the expected position of a template:

- Use of the last position as the guess for the new position $(u'_{i+1}, v'_{i+1}) := (u_i, v_i)$. This approach can be used when no motion model of the object is available.
- Use of a modelling function π for the motion of a template $(u'_{i+1}, v'_{i+1}) := \pi((u_j, v_j) \mid j \leq i)$ (Note that the first case can be represented with a function π as well).

For the tracking tests performed in this paper the first method as well as the first-order Newtonian motion-model, which could also be interpreted as a linear extrapolation method for the positions, was applied.

But as the main focus of the research was at the difference between RPH, NCC and DFD a more detailed discussion of motion-estimation is not the scope of this article.

5.3 Noise-Filtering

In order to gain more independence of noise a Kalman-filtering system was applied. In this article the fundamental principles of linear Kalman-filtering for tracking are not explained, instead the reader is refered to study [2] or [4].

The first-order motion-model which was used for the filtering and motion-prediction tests is described in Kalman-filter terms:

$$s_{i+1} = A_i s_i + \xi_i \tag{23}$$

$$o_i = C_i s_i + \eta_i \tag{24}$$

with

- **Measurements** (Matched template positions) $o_i \in \mathcal{N}_m \times \mathcal{N}_n$ represents the sequence of observed template position.
- **Measurement-transformation** $C_i := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$
- **Measurement-noise** η_i and **system-noise** ξ_i are considered zero-mean Gaussian white noise processes.
- **System-states** $s_i \in (\mathbb{R} \times \mathbb{R})^2$
- **System-transformation** (motion-modell) $A_i := \begin{pmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ with a fixed Δ_t for equidistant o_i.

Usually one can use the Kalman-filter for both, noise-reduction and the motion-prediction.

The error covariance matrices were determined empirically with the following two matrices proven best:

- System error covariance matrix was set to:

$$\begin{pmatrix} 20 & 0 & 0 & 0 \\ 0 & 20 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Measurement error covariance matrix was set to:

$$\begin{pmatrix} 20 & 0 \\ 0 & 20 \end{pmatrix}$$

6 Quality Measures for Tracking

Good quality measures for tracking methods enable a quantitative comparison of approaches on a common test set. Thus the choice and definition of the quality measures is driven by two main requirements:

- The tracked templates should be as close as possible to the real object-region in the current image. This requirement defines a local property of the tracker which is represented by a local measure of quality. This was chosen as a measure of overlap between the correct template-region and the tracked template-region. Thus the measure is called the **local overlap** Λ_μ for a given reference track μ.
- A tracking method should also track the template as long as possible over the image-sequence. This defines a global property of the tracker as the fulfillment of this requirement can only be measured when considering the whole track of a template. A natural choice for a quality-measure that implements this requirement is given by the **tracking length** Θ_μ for a given reference track μ.

Both measures use template tracks, which are defined as mappings $\tau : \mathbb{N} \longrightarrow \mathcal{F} \times (\mathcal{N}_m \times \mathcal{N}_m)$, with $\mathcal{F}$ a set of images f and $\tau(i) := (f_i, u_i, v_i)$ producing the position of the template (u_i, v_i) in the image f_i. For clarity reasons the function $\tau_p : \mathbb{N} \longrightarrow \mathcal{N}_m \times \mathcal{N}_m)$ gives only the position of the tracked template: $\tau_p(i) := (u_i, v_i)$ ($\mathcal{T}$ denotes the set of all possible tracks τ and $\mathcal{T}_m$ denotes the set of all manually labeled tracks).

The quality-measures **local overlap** and **tracking length** can now be specified in terms of our tracking notation. For the following description $\mu \in \mathcal{T}$ denotes a manually labeled reference track.

6.1 Local Overlap

The **local overlap** Λ_μ measures the size of the region that belongs to both, the correct manually determined template-region positioned at $\mu_p(i)$ and the automatically tracked region which is positioned at $\tau_p(i)$. Thus $\Lambda_\mu : \mathcal{T} \times \mathbb{N} \longrightarrow [0..1]$ is defined by:

$$\Lambda_\mu(\tau_p, i) := \frac{\mid \mathcal{R}_{h,w}(\mu_p(i)) \cap \mathcal{R}_{h,w}(\tau_p(i)) \mid}{\mid \mathcal{R}_{h,w}(\tau_p(i)) \mid} \tag{25}$$

This measure gives a value near 1 when the two given regions show a high overlap. If the compared regions are disjoint regions the **local overlap** measure is 0.

6.2 Tracking Length

The **tracking length** measure is defined by the number of steps for which the tracking method supplies a sufficiently good guess of the template position. Measuring the end of a track becomes the key-point for determining the tracking length. In this paper the end of a track was defined as tracking point when the local overlap is zero for the first time. Thus for a given manually labeled reference track μ the tracking length $\Theta_\mu : \mathcal{T} \longrightarrow \mathbb{N}$ can be defined as:

$$\Theta_\mu(\tau) := \min\{l \in \mathbb{N} \mid \Lambda_\mu(\tau, l) = 0\} \tag{26}$$

Both, the local overlap and the tracking length combined provide tracking method independent techniques for comparing different template trackers on a common test base.

In the result section of this article each of the test runs is described with 2 quality values. Those values are combinations of the basic measures Λ_μ and Θ_μ:

1. Mean tracking lengths: $\overline{\Theta}$
2. Mean local overlaps: $\overline{\Lambda}$
   The means local overlaps averages over all local overlap values for all the tracking steps smaller than the corresponding tracking length:

$$\overline{\Lambda} := \frac{\sum_{\mu \in \mathcal{T}_m} \sum_{i < \Theta_\mu(\tau_\mu)} \Lambda_\mu(\tau_\mu, i)}{\sum_{\mu \in \mathcal{T}_m} \Theta_\mu(\tau_\mu)} \tag{27}$$

$$\text{with } \tau_\mu \quad \text{corresponding automatic track for test track} \quad \mu \tag{28}$$

6.3 Computation Times

Another measure of quality for a given tracker is represented by the time which is consumed for each call to the tracking function. During the tests the exection time of the tracking function was an average of all performed tracking calls. The tracking time is measured in milliseconds and denoted as t.

7 Tests on Real-Live Video-Sequences

The comparison of the new tracking method based on projection histograms with the described standard methods of template matching was conducted on a video-sequence which was taken in an underground platform environment. The sequence consists of **1479** images in total which were taken at a frame-rate of around 18 frames per second. For the test a number of 30 different template tracks were manually labeled, all of them showing the motion-tracks of different persons heads (see fig. 2 for two example images):

- Mean length: 178.5, median length: 179 , min length: 33, max length: 419
- Standard deviation of length: 102.64

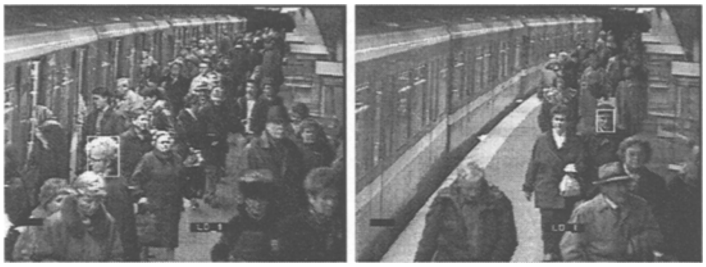

Fig. 2. 2 templates and related images

- Length of all manually segmented tracks: 5355

The tracking methods were tested by applying each method to all of the manually labeled template tracks in turn. Tracking the currently chosen track was stopped when the measured overlap quality became 0. The tracking test was then reinitialized with the template of the next track and tracking started over.

The following parameters were used for the tests on the real video-sequences:

- During the initialization of the tracker only the first manually labeled template region of each track is used to set up the template for tracking. Those initialization templates varied in size from around 15×20 to up to 40×50 pixel.
- During the tracking phase the manually labeled positions of the track were used only for calculating the tracking-quality measures.
- The search-region for the iteration of the template-matching was set to a rectangular region of size 20×20.
- The methods were tested on the grey-level sequence as well as on an edge-enhanced image-sequence which was used by applying a 3×3 Sobel-filter.
- The tests were carried out with both, the simple iteration of the template-matching approach on the one hand and the Kalman-filtering methods for noise-reduction and motion-prediction on the other hand.
- There were three different quantization functions ϑ used for the tests with projection-histograms RPHs.

The next section describes the results that were found for these tests.

8 Results

All the tests were performed on a SUN Ultra SPARC 2 station with two processors, each running at 200 MHz, still the function calls were not parallelized. The system had 256 MByte of RAM available and was driven by solaris2.5.

8.1 Results on Sobel-Filtered Images

The first test was run on the Sobel-filtered image sequence. All of the projection-based methods outperformed the standard template tracking methods in reference to the quality measures **overlaps** $\overline{\Lambda}$ and **tracking lengths** $\overline{\Theta}$ (see tab. 1). The following improvements of *RPH*s in contrast to the better reference method *NCC* can be observed:

- In case when simple position estimates were used *RPH*s performed by at least 31% to up to 89% better than *NCC*.
- When applying Kalman-filtering to the tracking methods, the *RPH*s even showed an improvement of 64 to 135% compared to *NCC*.

In all cases the *RPH* based methods performed better from a quality based point of view. Even when considering the execution times as the basis for comparing the different methods, the *RPH* method based on a threshold quantization of the template performs each call to the tracking function about 3[ms] faster than the reference methods.

Still for getting the best performance in tracking, about **20** more milliseconds must be spent for computing the *NCC* with ϑ_8 than for computing the *NCC* with ϑ_2. Similiar results were achieved for the tests on the grey-level image-sequences.

	Simple Search		Kalman NR		Kalman Motion		Execution Times
Methods	$\overline{\Theta}$	$\overline{\Lambda}$	$\overline{\Theta}$	$\overline{\Lambda}$	$\overline{\Theta}$	$\overline{\Lambda}$	t [ms]
RPH,ϑ_{256}	49.8	0.65	50.7	0.65	51.3	0.65	144 - 158
RPH,ϑ_8	**58.7**	**0.69**	**67.7**	**0.68**	**65.8**	**0.68**	36 - 37
RPH,ϑ_2	40.7	0.58	47.6	0.59	48.4	0.59	**15 - 16**
NCC	31.1	0.56	28.8	0.56	28.8	0.56	19 - 20
DFD	11.3	0.25	13.0	0.27	12.8	0.26	19 - 20

Table 1. Test results of tracking on Sobel filtered image sequences, with $\overline{\Theta}$ the mean tracking lengths, $\overline{\Lambda}$ the mean overlaps and the mean execution times t

8.2 Results on Grey-Level Images

The same tests were run on the grey-level images with even better results as with the Sobel-filtered image-sequence:

- *RPH*s again performed better than *NCC* and *DFD* by at least 69 to up to 211% better when considering the tracking-lengths.
- The execution times of the *RPH* based on a threshold quantization of the templates is faster than the *NCC* and *DFD* trackers by 1.5 to 3.4 [ms] for each call.

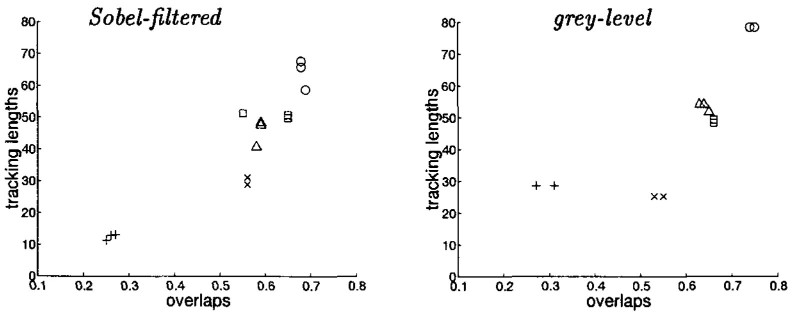

Fig. 3. Visualization of the Tracking Quality:The tracking lengths (Y-Axis) are plotted versus the overlaps (X-Axis). The left figure shows the results of the Sobel-filtered image-sequence. The results of the grey-level image-sequence are shown on the right. (□ = *RPH* with ϑ_{256}, ∘ = *RPH* with ϑ_8, Δ = *RPH* with ϑ_2, × = *NCC* and + = *DFD*)

	Simple Search		Kalman NR		Kalman Motion		Execution Times
Methods	$\overline{\Theta}$	$\overline{\Lambda}$	$\overline{\Theta}$	$\overline{\Lambda}$	$\overline{\Theta}$	$\overline{\Lambda}$	t [ms]
RPH,ϑ_{256}	49.3	0.66	49.3	0.66	48.3	0.66	144 - 151.4
RPH,ϑ_8	**78.4**	**0.74**	**78.4**	**0.75**	**78.4**	**0.75**	34 - 35
RPH,ϑ_2	54.0	0.63	54.0	0.64	51.6	0.65	**14 - 15**
NCC	25.2	0.53	25.2	0.55	25.2	0.55	18 - 19
DFD	28.6	0.27	28.6	0.31	28.6	0.31	17 - 18

Table 2. Test results of tracking on grey level image sequences, with $\overline{\Theta}$ the mean tracking lengths, $\overline{\Lambda}$ the mean overlaps and the mean execution times t

8.3 Different Thresholds for ϑ_2

Besides the quantization functions ϑ_{256} and ϑ_8, the quantization function ϑ_2 with a fixed threshold at 32 was chosen for the computation of the *RPH*s. This threshold value was not chosen with regard to the test-sequences, but instead can be chosen from a wide range of values as the figure 4 demonstrates. The figure shows the course of the achieved tracking-lengths on the test-sequences in relation to the chosen threshold value for ϑ_2. It can be observed that for a wide ranges of threshold values the tracking lengths are still better than the corresponding results for *NCC* and *DFD* Tracking. For the test-sequences of this paper the ranges were found as: thresholds with:

- $\{30, ..., 250\}$ for the Sobel-filtered image-sequences
- $\{30, ..., 170\}$ for the grey-level image-sequences

8.4 Different Quantizationsteps for ϑ_q

A similiar experiment was performed using different quantization steps $q \in \{2,4,8,16,32,64,128,256\}$ for the quantization function ϑ. It was observed that for all possible quantization values the *RPH* tracker performed better than both, *NCC* and *DFD* on either test-sequences (see 4). Thus the choice of a special quantization values is not critically for the good performance of the tracker on these test sets.

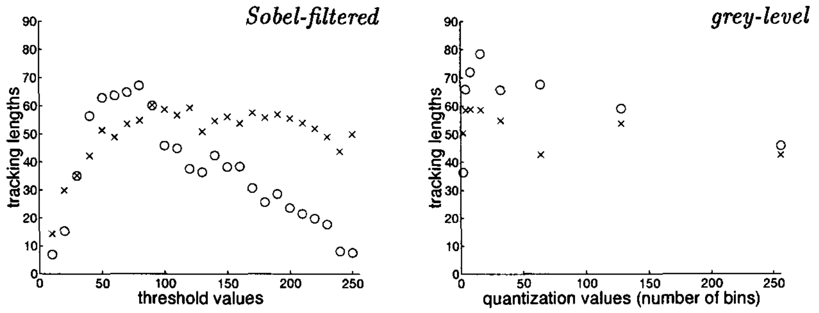

Fig. 4. Tracking-lengths in relation to different threshold values for ϑ_2 (left) or different quantization values for ϑ_q (right)
o results on grey-level sequence
× results on Sobel-filtered sequence

9 Conclusion

A new method for tracking two-dimensional image templates, based on projection histograms of the template-region, was presented. A comparison with the same tracking technique based on two standard template matching methods was performed and evaluated with previously defined quality measures. Both quality measures in combination give a means for rating different tracking techniques on the basis of a common test set.

Although the test-set included tracks of objects which changed their size over time, the method still does not incorporate a mechanism in order to account for significant changes in object size. Further work in this field of tracking objects with changing size has to be carried out in the future.

A significant superiority of all tested **projection-histogram** tracking methods was demonstrated, with improvements that ranged from 31% up to 211% in comparison with the reference methods .

The computation time depends on the chosen quantization function for the template coding with the projection histogram method. In the case of thresholded *RPH*s with a ϑ_2 the tracking methods also is the fastest methods of all tested techniques, with a computation time of 14.6 [ms] for one tracking step on a SPARC Ultra 2.

References

1. S. Birchfield. An elliptical head tracker. In *Asilomar Conference on Signals, Systems and Computers*, number 31. IEEE, November 1997.
2. C.K. Chui and G. Chen. *Kalman Filtering with Real-Time Applications.* Springer Series in Information Sciences. Springer, Berlin , Heidelberg, New York, 1987.
3. J. Denzler and H. Niemann. Active Rays: A New Approach to Contour Tracking. *International Journal of Computing and Information Technology*, 4(1):9–16, 1996.
4. R.E. Kalman. A new approach to linear filtering and prediction problems. *Trans., ASME, Basic Eng.*, 82:35–45, 1987.
5. J. Shi and C. Tomasi. Good features to track. In *CVPR.* IEEE, June 1994.
6. D. Terzopoulos and R. Szeliski. *Eds.: Blake, A. and Yuille, A. Active Vision*, chapter Tracking with Kalman Snakes, pages 3–20. Artificial Intelligence. MIT Press, Cambridge, Ma, 1992.
7. A. Tesei, G. L. Foresti, and C.S. Regazzoni. Human body modelling for people localization and tracking from real image sequences. In *Fifth International Conference on Image Processing and its Applications*, number 410 in 4, pages 806–809, Edinburgh, UK, July 1995. IEE.