Real Time Robust Template Matching

Frédéric Jurie and Michel Dhome
LASMEA - CNRS UMR 6602, Université Blaise-Pascal,
F-63177 Aubière - FRANCE
jurie@lasmea.univ-bpclermont.fr

Abstract

One of the most popular methods to extract useful informations from an image sequence is the template matching approach. In this well known method the tracking of a certain feature or target over time is based on the comparison of the content of each image with a sample template. In this article, we propose an efficient robust template matching algorithm that is able to track targets in real time. Special attention is paid to occlusions handling and illumination variations.

1 Introduction

Template matching is a major task for numerous computer vision applications. Two major categories of approaches are generally distinguished. *Feature-based* approaches uses local features like points, line segments, edges, or regions. With these techniques it is possible to localize the object in the current image and to predict the feature positions in subsequent ones, according to a motion model and an uncertainty model. Pose search techniques are naturally less sensitive to occlusions, as they are based on local correspondences. If several correspondences are missing the pose is still computable.

On the other hand, *global* or *template-based* approaches take the template as a whole. The strength of these methods lies in their ability to treat complex templates or patterns that cannot be modeled by local features. They are very robust and have been extensively used. They have also been called *sum-of-square-difference (SSD)* as they usually consist in minimizing the difference between a reference template and a region of the image. Historically brute force search was used. But this strategy is impractical in the case of transformations more complex than 2D translations, which involve higher dimensional parameter spaces.

Recently, a new efficient framework have been proposed. The tracking problem is posed as the problem of finding the best (in least squares sense) set of parameter values describing the motion and deformation of the target through the sequence. In this case, parameter variations are written as a linear function of a difference image (the difference between the reference target image and the current image). This approach is very efficient as motion can be easily deduced from the difference image. Cootes, Edwards and Taylor [2] use it to dynamically estimate the parameters of a face appearance model (2D model). Hager and Belhumeur [4] include it in a general framework for object tracking, under planar affine motions. Only a few works use this approach with projective transformations [3, 8, 7], because projective transformations are highly non-linear and because of

BMVC 2002 doi:10.5244/C.16.10

the size of the parameter space. In a recent article [6], we also have address the problem of tracking 3D surfaces viewed under projective transformations.

In this article, special attention is paid to occlusions handling and illumination variations. We propose an efficient solution to the problem of SSD tracking of *partially occluded* planar (or 3D) surfaces.

This article is made of three sections. In the first one, the problem of tracking templates with occlusions is posed and the principles of the proposed approach are briefly given. The next section is devoted to the detailed description of the proposed approach. In the last section, some experimental results are given.

2 Template matching and occlusion handling

2.1 Template based tracking

Let[1] $I(\mathbf{x},t)$ the brightness value at the location $\mathbf{x} = (x,y)$ in an image acquired at time t. Let $\mathcal{R} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ the set of N image locations which define a *target region*. $\mathbf{I}(\mathcal{R},t) = (I(\mathbf{x}_1,t), I(\mathbf{x}_2,t), \ldots, I(\mathbf{x}_N,t))$ is a vector of the brightness values of the target region. We refer to $\mathbf{I}(\mathcal{R},t_0)$ as the *reference template*. It is the template which is to be tracked; t_0 is the initial time ($t = 0$). These points are the projections of a set of 3D points $\mathcal{RO} = (\mathbf{X}_1, \ldots \mathbf{X}_N)$ belonging to an object surface.

The relative motion between the object and the camera induces changes in the position of the template in the image. We assume that these transformations can be perfectly modeled by a parametric *motion model*. In [7] we proposed a general motion model allowing any kind of planar transformations. In this article we assume that the transformation is applied to planar surfaces using homographies.

Let $\mathbf{X} = (X,Y,Z)$ be the coordinates of a point in the 3D object-centered coordinate system, and $\mathbf{x} = (x,y)$ its projection in the image. 3D homographies (as well as 3D rotations, translations and perspective projections) can be written with the standard homogeneous coordinates formalism as a matrix product: $\mathbf{x} = \mathbf{M}(\mu(t))\mathbf{X}$. In that case, we assume $\mathbf{x}$ and $\mathbf{X}$ to be written with homogeneous coordinates. Matrix $\mathbf{M}$ is representing the homography (or the composition of rotations/translations). $\mu(t) = (\mu_1(t), \ldots, \mu_P(t))$ is the set of parameters included in $\mathbf{M}$, depending on the relative position between the surface and the camera. There are six parameters ($P = 6$): three translation components and three Euler angles in case of projection of 3D surfaces (assuming the camera is calibrated), eight parameters ($P = 8$) in case of homographies. We call μ the motion parameter vector. At time t_0, the template position is known and parametrized by μ_0. The set of N image locations corresponding to the 3D points on the surface target are denoted $\mathcal{RO}$ and their projections at time t are denoted $\mathbf{M}(\mu(t))\mathcal{RO}$. With these assumptions, "tracking the object at time t" means "compute" $\mu(t)$ such that: $\mathbf{I}(\mathbf{M}(\mu(t))\mathcal{RO},t) = \mathbf{I}(\mathbf{M}(\mu_0)\mathcal{RO},t_0)$. The ground truth value, at time t_0, is supposed to be μ_0. The motion parameter vector of the target surface $\mu(t)$ can be estimated by minimizing the least squares following function:

$$O(\mu(t)) = \parallel \mathbf{I}(\mathbf{M}(\mu(t))\mathcal{RO},t) - \mathbf{I}(\mathbf{M}(\mu_0)\mathcal{RO},t_0) \parallel \quad (1)$$

This very general formulation of tracking have been used by several authors [1, 3, 4, 8].

[1]Bold fonts denote vectors and matrices.

2.2 Occlusion handling

2.2.1 Previous works

Equation (1) can be minimized by the straightforward estimation:

$$\mu(t) = argmin_{\mu(t)} \left(\sum_{\mathbf{X}_i \in \mathcal{RO}} \parallel \mathbf{I}(\mathbf{M}(\mu(t))\mathbf{X}_i, t) - \mathbf{I}(\mathbf{M}(\mu_0)\mathbf{X}_i, t_0) \parallel \right),$$

using brute force search [9] or optimization algorithms [1]. In that case, occlusions can be handled by introducing a robust error function ρ in the definition of the matching error, where ρ can be Huber's function [9] or other robust norms [10]. Black and Jepson [1] propose to treat some of the points as outliers. They define a binary "outliers vector", or "mask", $\mathbf{m}$. The mask is computed using a simple rule which set m_i as 0 or 1 depending on the value $\parallel \mathbf{I}(\mathbf{M}(\mu(t))\mathbf{X}_i, t) - \mathbf{I}(\mathbf{M}(\mu_0)\mathbf{X}_i, t_0) \parallel$.

On the other hand, other approaches minimize equation (1) using *difference images*. In that case, as shown in [3, 8, 4], the solution of equation (1) is $\mu(t+1) = \mu(t) + (J^t J)^{-1} J^t [\mathbf{I}(\mathbf{M}(\mu(t))\mathcal{RO}, t) - \mathbf{I}(\mathbf{M}(\mu_0)\mathcal{RO}, t_0)]$, where J is the Jacobian matrix of $\mathbf{I}$ with respect to μ. In case of occlusion, Hager et al. [4] propose to use an usual form of IRLS. In order to formulate the algorithm, they introduce the an "inner iteration" which is performed one or more times at each time step. They introduce a diagonal weighting matrix which weights the difference image. The inner iteration cycle consists in performing an estimation of this matrix step by step, while refining the motion estimation.

2.2.2 Proposed approach

None of the previously presented approaches really faces up to actual problems induced by occlusions. When using IRLS or robust norms the interpretation of the difference $\parallel \mathbf{I}(\mathbf{M}(\mu(t))\mathbf{X}_i, t) - \mathbf{I}(\mathbf{M}(\mu_0)\mathbf{X}_i, t_0) \parallel$ is ambiguous. There is no way to know if this difference is produced by an occlusion or by a movement of the object. There is an ambiguity between motion and occlusion. Strong differences (compared to the variance of grey level values without occlusions) can be easily discarded. On the other hand, weak differences (produced by shadows for example) can be interpreted as motions. We have experimentally observed that there are numerous cases where the previous approaches failed.

Handling occlusions is easier when using feature bases approaches, because robust estimation only concerns feature positions and not illumination. The geometric structure of objects cannot be changed by illumination and that is why in this case there is no ambiguity between occlusion and motion.

We strongly believe that robust estimation have to be applied on positions or local on transformations rather than on illumination. This is the key idea of this article. Unfortunately it is not directly applicable to template based approaches as templates are considered as being indivisible.

The proposed idea is to build a template representation which includes the original template plus sub-templates obtained by dividing it into several parts according to a quad-tree scheme. If there is l different levels, the representation will include $N_t = 1 + 4 + \cdots + 4^l$ different templates. Figure 1 illustrate this representation for $l = 3$.

In that case, there is not one but several templates that are to be tracked. First, it means that tracking must be very efficient because of the large number of templates.

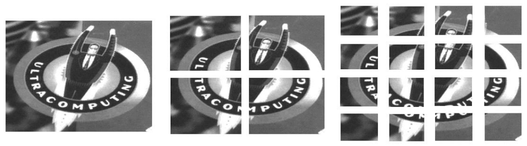

Figure 1: Template representation: it contains the original template plus sub-templates obtained by dividing the original one.

Second, each tracker only gives the motion of the corresponding sub-template. In case of occlusions, some of the trackers can provide erroneous motions. These local motions have therefore to be robustly combined in order to obtain the global and actual motion of the template. This task is not obvious because the largest sub-templates can be tracked with hight accuracy but are very sensitive to occlusions, while the smallest sub-templates produce less accurate but more robust motion estimations. As we want to suppress the ambiguity between illumination and motion, the only way to know if a tracker produces a correct motion is to robustly combine the set of motions produced by each one of these trackers.

If these two tasks - efficient sub-template tracking and robust motion estimation- can be handle this approach will be very powerful: when the template is not occluded the motion estimation is very accurate and can fully use the constraints brought by the whole template; in case of occlusion, the precision will decrease, proportionally to the amount of occlusion, because in that case the global motion will be computed from the motion of the smallest templates (those which are not affected by occlusions), which are less accurate.

The next section is devoted to the description of the two required tasks: efficient tracking and robust motion estimation.

3 Robust and efficient template matching

3.1 Efficient template matching - learning stage

Let us first see how we propose to efficiently track templates and sub templates. Each template and sub-template is tracked independently, using the following approach.

Our approach is based on two stages: an off-line learning stage and a tracking stage. This section describes the learning stage. During this learning stage a linear relation linking motion and illumination variations is established.

In this stage, we assume that the motion is computed from a difference between the initial position and a new position in the neighbourhood of the initial position. We will see later how to generalize this to any position of the template.

The key idea is to obtained a very straightforward and efficient computation of the motion parameters $\mu(t)$ by writing:

$$\mu(t_0+\tau) \quad = \quad \mu(t_0) + \mathbf{A}(t_0+\tau)[\mathbf{I}(\mathbf{M}(\mu_0)\mathcal{RO}, t_0) - \mathbf{I}(\mathbf{M}(\mu(t_0))\mathcal{RO}, t_0+\tau)]$$

where τ denotes the time between two successive images (see [7] for detailed explanations). We will see later how matrix $\mathbf{A}(t_0+\tau)$ can be obtained. If we write $\delta\mathbf{i}(t_0+\tau) =$

$\mathbf{I}(\mathbf{M}(\mu_0)\mathcal{RO}, t_0) - \mathbf{I}(\mathbf{M}(\mu(t_0))\mathcal{RO}, t_0 + \tau)$and $\delta\mu(t_0 + \tau) = \mu(t_0 + \tau) - \mu(t_0)$, the previous equation can be written:

$$\delta\mu(t_0 + \tau) = \mathbf{A}(t_0 + \tau)\delta\mathbf{i}(t_0 + \tau) \tag{2}$$

Equation (2) can be seen as the equation of P hyper-planes (P is the number of parameters of the transformation). An estimation of the matrix $\mathbf{A}$ can be done using the method proposed in [7].

3.2 Robust tracking

At this level, we assume that we know the variation $\delta\mu_i$ of the N_t templates, by using N_t different trackers working independently, as described in the previous section. Each tracker provides a motion vector $\delta\mu_i$, $i \in [1, N_t]$, relatively to the initial template position. From this set of motion vectors, we want to compute the real motion $\delta\mu$, in a robust way. When there is no occlusion, the vectors $\delta\mu_i$ are the same; in case of occlusion, those affected by occlusion will provide erroneous motion estimation. We have to keep in mind that these variations are variations from the initial position μ_0.

The key idea is to suppose that correct motion vectors will be concentrated in a unique area of the motion vector space, while erroneous ones will be disorganized, because the effect of occlusion on each tracker is different. We propose to see this problem as the problem of finding the correct "variation" of the template pose ($\delta\mu$ vector is the variation of the template pose) from N_t poses variations hypotheses ($\delta\mu_i$).

Furthermore, the fact that the different trackers do not have the same accuracy have to be taken into account to "weight" the hypotheses. In practice, the uncertainty will be modeled using Gaussian models.

Finally, the "best pose variation", i.e. the vector $\delta\mu$ that best satisfies the $\delta\mu_i$ hypotheses, can be computed by exploring the parameter space, taking into account the Gaussian contribution provided by each hypothesis. This "best pose" search will be done according to the strategy given in [5].

Modeling the uncertainty

During the learning stage, equation (2) allows to compute an estimation of matrix $\mathbf{A}$, for a given tracker. In a second time, one can compute the covariance matrix associated with this model. There is one covariance matrix per tracker. Let $\mathbf{Q}$ be this covariance matrix, obtained by: $\mathbf{Q} = (\mathbf{Y} - \mathbf{AH})^t(\mathbf{Y} - \mathbf{AH})$, using the notation defined in the previous section. Motion errors are assumed to be normally distributed.

Let $P(\delta\mu|\delta\mathbf{i})$ be the probability of $\delta\mu$ knowing $\delta\mathbf{i}$. In a P-dimensional parameter space, the P-dimensional normal probability density function with covariance matrix $\mathbf{Q}$ is:

$$P(\delta\mu|\delta\mathbf{i}) = (2\pi)^{-\frac{P}{2}}|\mathbf{Q}|^{-\frac{1}{2}}exp(-\frac{1}{2}\delta\mathbf{i}^t\mathbf{Q}^{-1}\delta\mathbf{i})$$

For purpose of simplification, in the rest of the paper we do not represent features in their original feature space [5]. We decompose the covariance matrix $\mathbf{Q}^{-1}$ as : $\mathbf{Q}^{-1} = \mathbf{UD}^{-1}\mathbf{U}^t$ where $\mathbf{U}$ is the orthogonal matrix of eigenvectors of $\mathbf{Q}$ and $\mathbf{D}$ the diagonal matrix of eigen values. The P eigen values will be denoted $\lambda_1, \cdots, \lambda_P$.

By projecting features in the eigenspace, the normal distribution for the model feature i is much simple:

$$P(\delta\mu|\delta\mathbf{i}) = (2\pi)^{-\frac{P}{2}} \prod_{j=1}^{j\leq P} \lambda_j^{-\frac{1}{2}} exp(-\frac{1}{2}\sum_{j=1}^{j\leq P}\frac{\epsilon_j^2}{\lambda_j})$$

$\Delta\mathbf{i}$ denotes the difference $\delta\mathbf{i}$ projected into the eigenspace ($\Delta\mathbf{i} = \mathbf{U}\delta\mathbf{i}$).

As explained before, there is one different covariance matrix per tracker.

Finding the best variation $\delta\mu$.

During the tracking stage, each tracker provides a motion hypothesis $\delta\mu_i$, which has to be interpreted as a Gaussian distribution in the motion parameter space. The best motion variation is defined as the variation satisfying the largest number of trackers. This variation $\delta\mu$ can be define as:

$$\delta\mu = argmax_{\delta\mu} \sum_{i\in[1,N_t]} P_i(\delta\mu|\delta\mathbf{i}) \qquad (3)$$

Justifications for using this definition are given in [5].

The idea is to find the best motion by recursively exploring the space of possible motions. Recursive subdivision consists in recursively splitting the space in two subspaces, alternating axes, and in evaluating the probability of each one of the two sub-spaces, and finally in exploring the ones having highest probabilities.

This process can be seen as a tree search. The root node corresponds to the initial box. Leaves are the smallest regions taken into account. We proposed in [5] a very efficient way to explore this space. The same strategy is used here.

At the end we get a small subset of the parameter space which best satisfied equation (3).

3.3 Tracking with hyperplane approximation

The proposed scheme is very inefficient, taken under its initial form. Direct computation of matrix $\mathbf{A}$ involves a least square minimization, which is to be repeated for each new position of the template. The matrix depends on the current position, orientation, etc. given by μ. The learning stage consists in computing a linear relation between a set of grey level differences and a correction of the parameter variation $\delta\mu$. This relationship is computed around the value μ_0 – known when the user selects an image region – and is not valid for other values of μ.

We explain in [7] how it is possible to extend this relation for any value of μ, without recomputing the matrix.

Let the region be defined as $\mathcal{RO} = (\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N)$ the set of N points in a local reference called the *region reference*. Matrix $\mathbf{M}(\mu)$ changes the coordinates of $\mathbf{X} = (X, Y, Z)$ in the reference region into $\mathbf{x} = (x, y) = \mathbf{M}(\mu)\mathbf{X}$ in the reference image. Time is suppressed to simplify the notations.

When the user defines a target region in the reference image, he defines a set of correspondences between points in the *reference region* and points in the *reference image* (for example the corners of a rectangular region). Knowing this set of correspondences, the computation of μ_0 such that $\mathbf{M}(\mu_0)$ aligns the *reference region* on the target (defined in the *reference image*) is possible.

The learning stage consists in producing small random disturbances $\delta\mu$ around μ_0. We denote such disturbances as ${\mu_0}' = \mu_0 + \delta\mu$.

In order to simplify the notations, we write: $\mathbf{M} = \mathbf{M}(\mu)$, $\mathbf{M}_0 = \mathbf{M}(\mu_0)$ and $\mathbf{M}'_0 = \mathbf{M}(\mu_0 + \delta\mu_0)$. We also suppose that matrix $\mathbf{M}_\Delta$ is such $\mathbf{M}'_0 = \mathbf{M}_0\mathbf{M}_\Delta$.

Disturbances produce a change of brightness $\delta\mathbf{i} = \mathbf{I}(\mathbf{M}_0\mathcal{RO}) - \mathbf{I}(\mathbf{M}'_0\mathcal{RO})$.

A set of N_p disturbances $\delta\mu$ are produced in order to obtain the linear model giving $\delta\mu = \mathbf{A}\delta\mathbf{i}$. During the training stage it is possible to estimate motion $\delta\mu$ knowing $\delta\mathbf{i}$.

If $\mathbf{x}'$ are the coordinates of $\mathbf{X}$ in the reference image under the transformation μ'_0 then $\mathbf{x}' = \mathbf{M}'_0\mathbf{X}$. Let $\mathbf{X}'$ be such that $\mathbf{x} = \mathbf{M}'_0\mathbf{X}'$. Assuming $\mathbf{M}$ is invertible, we obtain $\mathbf{X}' = \mathbf{M}'^{-1}_0\mathbf{M}_0\mathbf{X}$.

Therefore, knowing $\delta\mathbf{i}$, we can estimate μ'_0, and finally compute the displacement of the region expressed in the *region reference*. This displacement is only valid around μ_0.

At the beginning of the tracking stage, a prediction of the parameters is known and is denoted μ'. The tracking consists in estimating $\mathbf{M} = \mathbf{M}(\mu)$ such that

$$\mathbf{I}(\mathbf{M}\mathcal{RO}, t) = \mathbf{I}(\mathbf{M}_0\mathcal{RO}, t_0),$$

with the notation $I_0(\mathbf{M}_0\mathcal{RO}) = \mathbf{I}(\mathbf{M}_0\mathcal{RO}, t_0)$. Time t is removed to simplify the notations.

By computing

$$\delta\mu = \mathbf{A}\delta\mathbf{i} = \mathbf{A}[\mathbf{I}_0(\mathbf{M}_0\mathcal{RO}) - \mathbf{I}(\mathbf{M}'\mathcal{RO})], \tag{4}$$

where $\mathbf{M}' = \mathbf{M}(\mu')$, μ' being the predicted position of the template. We obtain a disturbance $\delta\mu$ that would have produced $\delta\mathbf{i}$ if the parameters vector had been μ_0. In that case, a location $\mathbf{X}$ of the region is transformed into $\mathbf{X}' = \mathbf{M}'^{-1}_0\mathbf{M}_0\mathbf{X}$, with $\mathbf{M}'_0 = \mathbf{M}_0\mathbf{M}_\Delta$.

The actual transformation turns $\mathbf{X}$ into $\mathbf{x}$: $\mathbf{x} = \mathbf{M}\mathbf{X}$. Introducing $\mathbf{X}' = \mathbf{M}'^{-1}_0\mathbf{M}_0\mathbf{X}$ in the relation $\mathbf{x}' = \mathbf{M}'\mathbf{X}'$ gives:

$$\mathbf{x}' = \mathbf{M}'\mathbf{X}' = \mathbf{M}'\mathbf{M}'^{-1}_0\mathbf{M}_0\mathbf{X} \tag{5}$$

This equation is fundamental for tracking: it gives the transformation aligning the region on the target at the current time, knowing a prediction μ' and a local disturbance $\delta\mu$. This local disturbance around the initial value μ_0 is obtained by mapping the current image on the reference region and computing the difference $\delta\mathbf{i} = \mathbf{I}(\mathbf{M}_0\mathcal{RO}) - \mathbf{I}(\mathbf{M}'\mathcal{RO})$. Equation (2) gives $\mu'_0 = \mu_0 + \mathbf{A}\delta\mathbf{i}$, and allows to compute $\mathbf{M}_\Delta = \mathbf{M}_0^{-1}\mathbf{M}'_0$.

The main idea is therefore to correct the transformation of the region in the reference region (acting as if the parameters were μ_0) and to transform this correction by applying $\mathbf{M}(\mu')$. This can be written:

$$\mathbf{M} = \mathbf{M}'(\mathbf{M}_0\mathbf{M}_\Delta)^{-1}\mathbf{M}_0 \tag{6}$$

Using this approach, it is possible to track more than 50 templates/sub-templates in real time, assuming these templates include from 100 (largest templates) to 20 points (smallest ones).

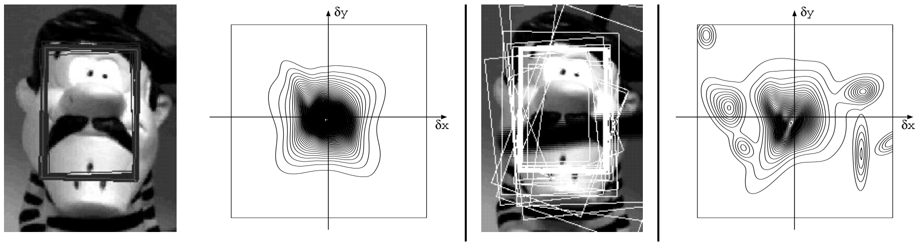

Figure 2: tracking with and without occlusions. Left: white rectangular area represent templates position provides by each tracker. Right: representation of the transformation space. The probability density is projected on the (tx, ty) plane, in order to represent the transformations. The square area represents translation of 10x10 pixels.

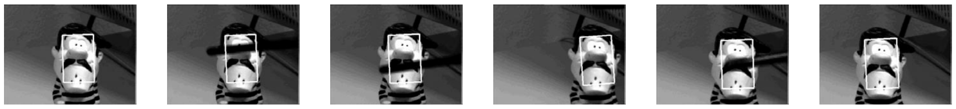

Figure 3: planar motions with occlusions. Six images taken from a video sequence.

4 Results

Tracking with translation, rotation and scaling In case of planar translation (tx, ty), planar rotation (θ) and scale (s), the relation between a point of coordinates (X, Y) in the reference region and the corresponding point in the image coordinates (x, y) is: $x = s\cos(\theta)X - s\sin(\theta)Y + tx$ and $y = s\sin(\theta)X + s\cos(\theta)Y + ty$.

It can be written with matrix products, using homogeneous coordinates.

For this experiment we use 3 levels to represent the template, leading to 21 different trackers. The largest template is made of 100 points; on the intermediate levels, sub-templates are made of 40 points; the 16 smallest sub-templates are made of 20 points. The learning stage take less than 0.5s; during this stage 1000 disturbances are produced.

The tracking (sub-template tracking and robust estimation) takes less than 30ms on a standard Silicon Graphics 02 workstation.

Figure 2 provides and illustration of our method. The same template is shown with and without occlusions. On the left, white rectangular areas represent template positions provided by each tracker. The transformation space is represented on the right. The probability density is projected on the (tx, ty) plane, in order to represent the transformations. The figure shown translations in an area of 10x10 pixels. If there is no occlusion, the trackers do not provide exactly the same results, because of the inaccuracy of the "smallest" trackers. Some trackers are more accurate than other, depending on the patterns included in the various sub-templates.

Figure 3 shows six images taken from a video sequence. It illustrates the ability of the algorithm to handle partial occlusions in real time.

Tracking homographic motions Homographic motions can be handle exactly in the same way. Figure 4 show six images taken from a video sequence processed in real time.

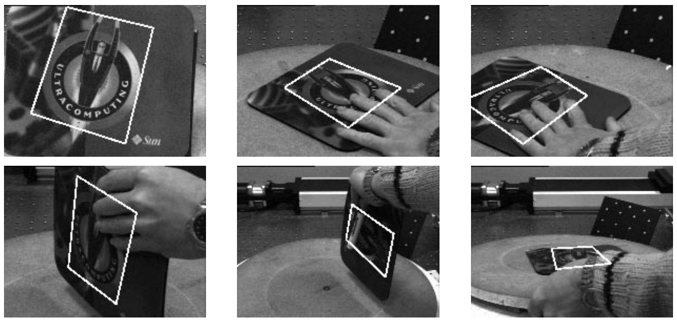
Figure 4: Tracking homographic motions with occlusions. Six images taken from a video sequence.

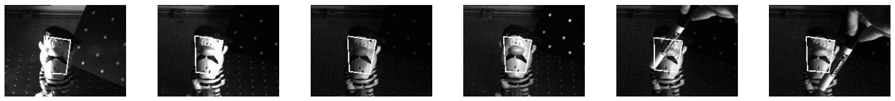
Figure 5: Tracking homographic motions with illumination changes. Six images taken from a video sequence.

Tracking with illumination variations Illumination variations can be treated in a much simpler way than occlusions. Occlusions locally change the local repartition of grey levels values. On the other hand, illumination changes can be easily compensated by locally norming the image. Instead of directly taking the grey levels values to obtain $\mathbf{I}(\mathbf{x})$, we compute the min and max value of the intensity in a small area around $\mathbf{x}$. In the previous equations, we substitute $\mathbf{I}(\mathbf{x})$ for $\frac{\mathbf{I}(\mathbf{x})-min}{max-min}$.

This simple normalization gives very reliable results in case of illumination variations as well as in case of shadows. Fig. 5 shows some images taken from a video sequence with occlusions shadows and illumination variations.

5 Conclusions

We have presented an original and efficient algorithm for handling occlusions and illumination variation in SSD trackers.

Using robust norms directly on intensity is ambiguous, as it is impossible to know if intensity variation is due to target motions or is due to occlusions. Our approach consists in representing the template as a pyramid of sub-templates, and in tracking independently each sub-pattern. The robust estimation is based on local motions rather than on intensities, suppressing ambiguities. The smallest sub-templates provide inaccurate but robust motions. The largest ones provide non robust but accurate motions. The search for the optimal global motion is computed using a recursive search in the motion space.

The presented experimental results show that our technique is efficient (real-time on low cost hardwares) and robust to occlusions and illumination changes.

References

[1] M.J. Black and A.D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation, January 1998.

[2] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. In *Proc. European Conference on Computer Vision*, pages 484–498, Freiburg, Germany, 1998.

[3] M. Gleicher. Projective registration with difference decomposition. In *CVPR97*, pages 331–337, 1997.

[4] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, October 1998.

[5] F. Jurie. Solution of the simultaneous pose and correspondence problem using gaussian error model. *Computer Vision and Image Understanding*, 73(3):357–373, 1999.

[6] F. Jurie and M. Dhome. Real time 3d template matching. In *Computer Vision and Pattern Recongition*, pages (I)791–797, Hawai, December 2001.

[7] F. Jurie and M. Dhome. Real time template matching. In *Proc. IEEE International Conference on Computer vision*, pages 544–549, Vancouver, Canada, July 2001.

[8] M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of textured-mapped 3d models. *PAMI*, 22(4):322–336, April 2000.

[9] H.T. Nguyen, M. Worring, and R. van den Boomgaard. Occlusion robust adaptive template tracking. In *ICCV01*, pages I: 678–683, 2001.

[10] Z. Zhang. Parameter-estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997.