Published in IET Computer Vision
Received on 25th March 2010
Revised on 9th November 2010
doi: 10.1049/iet-cvi.2010.0032

ISSN 1751-9632

Real-time tracking using A* heuristic search and template updating

E. Sánchez-Nielsen[1] M. Hernández-Tejera[2]
[1]Departamento E.I.O. y Computación, Universidad La Laguna, 38271 La Laguna, Spain
[2]Instituto de Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería, ULPGC, 35017 Gran Canaria, Spain
E-mail: enielsen@ull.es

Abstract: Many vision problems require fast and accurate tracking of objects in dynamic scenes. In this study, we propose an A* search algorithm through the space of transformations for computing fast target 2D motion. Two features are combined in order to compute efficient motion: (i) Kullback–Leibler measure as heuristic to guide the search process and (ii) incorporation of target dynamics into the search process for computing the most promising search alternatives. The result value of the quality of match computed by the A* search algorithm together with the more common views of the target object are used for verifying template updates. A template will be updated only when the target object has evolved to a transformed shape dissimilar with respect to the actual shape. The study includes experimental evaluations with video streams demonstrating the effectiveness and efficiency for real-time vision based tasks with rigid and deformable objects.

1 Introduction

Real-time and accurate template tracking is a critical task in many computer vision applications such as vision based interface tasks, visual surveillance, navigation tasks for autonomous robots, and virtual and augmented reality systems. The central challenge is to determinate the image position of a target in a dynamic vision context given one or several templates that represent the target, and adjusting the requirements of the computational cost of the template tracker to be as low as possible to make feasible real-time performance over general purpose hardware. This template tracking problem of objects in three-dimensional (3D) space from 2D images can be formulated in terms of decomposing the transformations induced by moving objects between frames into two parts: (i) a 2D motion, corresponding to the change of the target position in the image space, which is referred to as the template position-matching problem and (ii) a 2D shape, corresponding to a different aspect of the object becoming visible or an actual change of shape in the object, which is referred to as the template updating problem.

For the template-matching problem different algorithms involving searching through a space of transformations have been proposed. Some of them establish point correspondences between two shapes and subsequently find a transformation that aligns these shapes [1, 2]. The iteration of these methods involves the use of algorithms such as iterated closest points (ICP) [3, 4] or shape context matching [2]. Other approaches are based on an exhaustive search that works by subdividing the space of transformations to find the transformation that matches the template position into the current image based on the Hausdorff distance [5–7]. However, heuristic measures and target dynamics have not been combined in these search processes. This situation leads to an increase of the computational costs in the tracking process, limiting therefore, the efficiency of the search process for computing fast performance for real-time applications.

For template updating problem different template tracking approaches have been based on the underlying assumption that the appearance of the target object remains the same through the entire video [8–10]. This assumption is generally reasonable for a certain period of time and a naive solution to this problem is updating the template every frame [5, 7, 11] or every n frames [12]. This situation leads that small errors can be introduced in the location of the template each time the template is updated and establishing that the template gradually drifts away from the object [13].

In this paper, a template based solution for fast and accurate tracking of moving objects is proposed. The main contributions are: (i) an A* search algorithm that uses the Kullback–Leibler measurement as heuristic to guide the search process for efficient matching of the target position, (ii) dynamic update of the space of transformations using the target dynamics, (iii) updating templates only when the target object has evolved to a new shape change significantly dissimilar with respect to the current template in order to solve the drift problem and (iv) representation of illustrative views of the target shape evolution through a short-time visual memory (STVM).

As a result, the first two contributions provide a fast algorithm to apply over a space of transformations for computing target 2D motion, and the other two contributions provide robust tracking because accurate template updating can be performed. In addition to these contributions, the paper also contains a number of experimental evaluations on challenging sequences.

The structure of this paper is as follows: the problem formulation is illustrated in Section 2. In Section 3, the A* search algorithm for computing target position is described. The updating reference template problem is detailed in Section 4. Extensions to the basic tracking approach are described in Section 5. Experimental results are provided in Section 6 and Section 7 concludes the paper.

2 Problem formulation

For the sake of subsequent problem formulation, some definitions are introduced first:

Definition 1 (template): Let $T(k) = \{t_1, \dots, t_r\} \subseteq \mathbb{R}^2$ be a set of points that represent a template in step time k.

Definition 2 (image): Let $I(k) = \{i_1, \dots, i_s\} \subseteq \mathbb{R}^2$ be another set of points that denote an input image in step time k. It is assumend that each new step time k corresponds to a new frame k of the video stream.

Definition 3 (set of transformations): Let a translational transformation g be parameterised by the x displacement g_x and the y displacement g_y. That is, $g = (g_x, g_y)$. Let a bounded set of translational transformations be a set of transformations $\mathbb{G} = [g_{x\,\min}, g_{x\,\max}] \times [g_{y\,\min}, g_{y\,\max}] \subseteq \mathbb{R}^2$ and let $g^c = (g_x^c, g_y^c)$ denote the transformation that corresponds to the center of $\mathbb{G}$. It is defined as

$$g^c = \left(\left(\frac{1}{2}(g_{x\,\min} + g_{x\,\max})\right), \left(\frac{1}{2}(g_{y\,\min} + g_{y\,\max})\right)\right) \quad (1)$$

where the lower and upper bounds of $\mathbb{G}$ in x-and y-dimension are represented by (x min, x max) and (y min, y max).

2.1 Template position matching problem

Given a step time k, a template $T(k)$, an input image $I(k)$ and an error bound ε, the template position-matching problem can be viewed as the search process in the space of transformations $\mathbb{G}$ in order to find the transformation g_{opt} that satisfies the quality of match $Q(g)$.

The quality of match $Q(g)$ assigned to a transformation g is defined by the allowed error ε when template points are brought to image points using the transformation g. It is expressed as

$$Q(g) = h_l(g(T(k)), I(k)) < \varepsilon \quad (2)$$

where $h_l(g(T(k)), I(k))$ is the partial directed Hausdorff distance [14] between the translated points set of template $T(k)$ by the translation g (it is denoted as $g(T(k))$) and the input image $I(k)$. It is defined as

$$h_l(g(T(k)), I(k)) = \underset{t\in T}{L}{}^{\text{th}} \min_{i\in I} \|g(t(k)) - i(k)\| \quad (3)$$

where $\|.\|$ denotes the Euclidean distance. The partial directed Hausdorff distance ranks each point of the translated template T based on its distance to the nearest point in I and uses the lth quartile ranked point as the measure of distance.

2.2 Template updating problem

Once the new position of the target has been computed, the template is updated to reflect the change in its shape. Since the template-matching position problem determines translational motion between consecutive frames, all the non-translational transformations of the image motion and 3D shape change are considered to be a change in the 2D shape of the object.

Given $g_{\text{opt}}(T(k))$ and $I(k)$, new 2D shape changes between successive images are computed as the measure of the discrepancy between $g_{\text{opt}}(T(k))$ and $I(k)$ under a certain error bound δ. That is, the new template $T(k+1)$ is represented by all those points of input image $I(k)$ that are within distance δ of some point of $g_{\text{opt}}(T(k))$ according to the following expression

$$T(k+1) = \sum_{i\in I} \|g_{\text{opt}}(T(k)) - i\| < \delta \quad (4)$$

Fig. 1 illustrates a graphical summary of the problem formulation. The set of points that represent $I(k)$ and $T(k)$ are edge-based features extracted from real images using a Canny edge detector [15].

3 Search framework for computing template position

Formulation of problem solving under the framework of heuristic search is expressed through a 'state space-based representation approach' [16], where the possible problem situations are considered as a set of states. The start state corresponds to the initial situation of the problem, the goal state corresponds to problem solution, and the transformation between states can be carried out by means of operators. As a result, problem solving is addressed as the sequence of operators which transform the start state to the goal state.

According to the heuristic search framework described above, the A* search problem is formulated as: the search process oriented to find the transformation g_{opt} in $\mathbb{G}$ that satisfies the quality of match $Q(g)$. The elements of the problem solving according to this formulation are:

1. *Initial state:* corresponds to a bounded set of $\mathbb{G}$.
2. *State:* each search state n is associated with a subset of transformations $G_n \subseteq \mathbb{G}$. Each state is represented by the transformation g^c.
3. *Goal state:* is the transformation that best matches the current template $T(k)$ in the current image $I(k)$, according to $Q(g)$.
4. *Operators:* are the functional elements that lead the transformation of one state to another. For each current state n, the operators A and B are computed:

- *Operator A:* Each set of transformations from the current state is partitioned into four new states by vertical and horizontal bisections. Each one of these new states is represented by the transformation g^c.
- *Operator B:* This operator computes the quality of match (expression (3)) for each one of the four new states generated by operator A. It is denoted as $h_l(g^c(T(k)), I(k))$.

Splitting each current state into four new states leads to the representation of the search tree to be a quaternary structure, where each one of the four states computed are referred to as

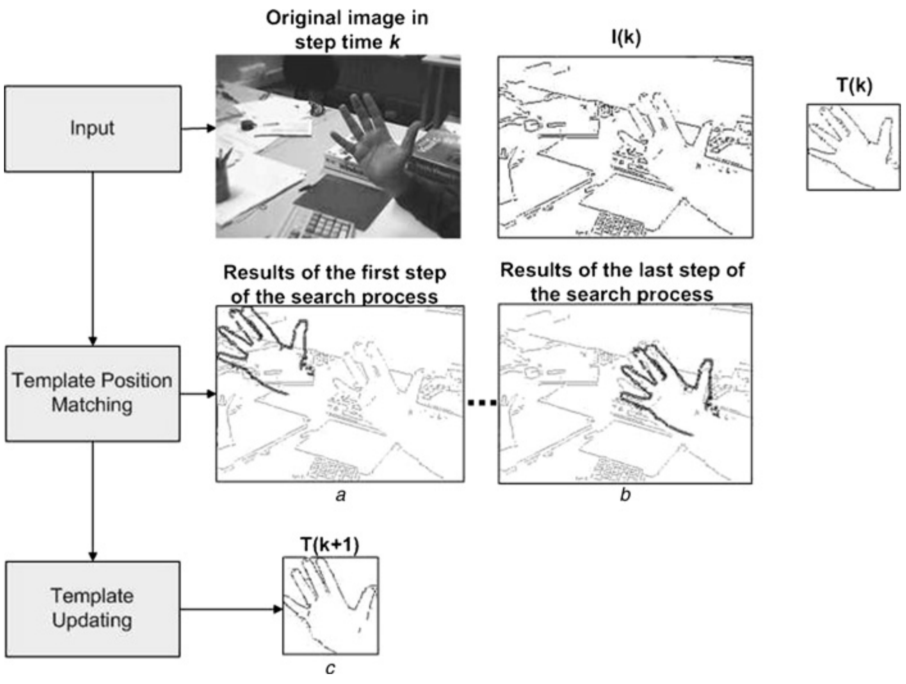
Original image in step time k
I(k)
T(k)
Input
Results of the first step of the search process
Results of the last step of the search process
Template Position Matching
a
b
T(k+1)
Template Updating
c

Fig. 1 *Schematic overview of the template tracking problem*

Template position matching consists of the search process in the space of transformations $\mathbb{G}$ in order to find the transformation g_{opt} that satisfies the quality of match $Q(g)$

a Illustrates the result of applying the initial transformation $g = (g_x, g_y)$ of the search process to every point in template $T(k)$, where $Q(g)$ is not satisfied

b Shows the result of applying the transformation that best matches the current template points $T(k)$ in the image $I(k)$, after the search process has been computed and where $Q(g)$ is satisfied

c Once the template-matching problem has been computed, the template is updated in order to represent the new view of the target such is illustrated

NW, NE, SE and SW cells. The heuristic search is initiated by the association of the set $\mathbb{G}$ with the root of the search tree, and subsequently the best state at each tree-level l is expanded into four new states. The splitting operation is finished when the quadrisection process computes a translational motion according to the quality of match $Q(g)$ or all the the different states have been partitioned in cells of unit size.

The best state to expand from NW, NE, SW and SE cells at each tree-level l is computed by the evaluation function of the A* search algorithm [16], which combines features of uniform-cost search and pure heuristic search. The corresponding cost value assigned to each state n by the evaluation function $f(n)$ is given as

$$f(n) = c(n) + h^*(n) \tag{5}$$

where $c(n)$ is the estimated cost of the path from the initial state s_0 to current state n, and $h^*(n)$ is the heuristic estimate of the cost of a path from state n to the goal state.

3.1 Heuristic evaluation function $h^*(n)$

The heuristic value of the cost of a path from state n to the goal is estimated evaluating the quality of the best solution reachable from the current state n. Desirability of the best state is computed measuring the similarity among the distribution functions that characterise the current state and the goal state.

Let P denote the distribution function that characterises the current state n and let Q denote the corresponding distribution function that characterises the goal state. Since the quality of match $Q(g)$ is denoted by the partial directed Hausdorff distance, the distribution function P can be approximated by a histogram of distances $\{H_{g^c}\}_{i=1\ldots r}$, which contains the number of template points $T(k)$ at distance d_j with respect to the points of the input image $I(k)$, when the transformation g^c of the current state n is applied on the current template $T(k)$.

Similarly, since the quality of match $Q(g)$ is denoted by the partial directed Hausdorff distance, the distribution function Q that characterises the goal state corresponds to the largest number of ranked template points $T(k)$ at distance zero, and closest to zero with respect to the points of the input image $I(k)$, when the transformation g^c of the current state n is applied on the current template $T(k)$. This distribution function Q can be modelled by an exponential distribution function $f(n) = ke^{-an}$, where parameter a checks the falling of the exponential function. Therefore if the quality of match $Q(g^c)$ assigned to the transformation g^c is satisfied, the distributions P and Q, respectively, will show an appearance similar to the illustration of Figs. 2*a* and *b*.

The similarity between distributions P and Q is measured using the Kullback–Leibler distance (KLD) [17, 18]

$$D(P\|Q) = \sum_{i=1}^{R} p_i \log\frac{p_i}{q_i} \tag{6}$$

where R is the number of template points. According to [17], $D(P\|Q)$ has two important properties:

1. $D(P\|Q) \geq 0$,
2. $D(P\|Q) = 0$ iff $P = Q$.

The values of KLD, will be non-zero and positive when the distributions P and Q are not similar because the template points do not match the input image points. Otherwise, the value of KLD is equal or close to zero when the template points match the input image points.

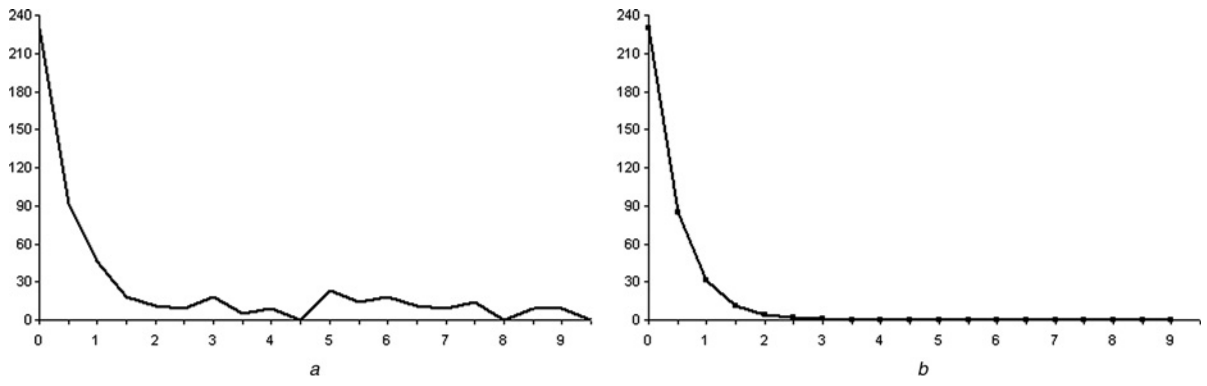

Fig. 2 *Distribution functions*

a Histogram of distances $\{H_{g^c}\}_{i=1\dots r}$ associated with a transformation g^c that verifies the quality of match $Q(g^c)$
b Distribution function $f(n) = ke^{-an}$ with parameter $a = 1$ that characterises the final state. The horizontal axis represents distance values and the vertical axis denotes the number of transformed template points with g^c at distance d_j with respect to the current points scene

3.2 Estimated cost function c(n)

An estimated cost function $c(n)$ is added to the evaluation function $f(n)$ in order to generate a backtracking when the heuristic function leads the search process towards no promising solutions. This depth term is based on the number of operators of type A applied from the initial search state to the current state n.

3.3 Initial state computation

The dimension $M \times N$ of $\mathbb{G}$ is dynamically computed as an adjustable size-based set of transformations in every frame k by means of incorporating an alpha–beta predictive filtering [19] into the A* search algorithm. It is focused on the assumption that there is a relationship between the size of the set of transformations and the resulting uncertainty of the alpha–beta predictive filtering. The basic hypothesis underlying this assumption is that the regularity of the movement of mobile objects is related to small uncertainties in the prediction of future target locations, which leads to a reduced set of transformations. Otherwise, the set $\mathbb{G}$ is increased when the motion trajectory exhibits any deviation from the assumed temporal motion model.

The dimension of $\mathbb{G}$ is computed from the parameters to be estimated by the filtering approach. The location vector and the corresponding velocity vector of the filtering approach [19] are jointly expressed as a state vector $\boldsymbol{x} = [\theta^T, \dot{\theta}^T]^T$. The dynamic state equation assuming a constant velocity model is formulated as

$$\boldsymbol{x}(k+1) = \Phi\boldsymbol{x}(k) + v(k) \tag{7}$$

The state vector estimation is computed as

$$\hat{\boldsymbol{x}}(k+1 \mid k+1) = \hat{\boldsymbol{x}}(k+1 \mid k) + v(k+1)\left[\alpha \frac{\beta}{\Delta T}\right]^T \tag{8}$$

where $\Phi = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$, Δt is the sampling interval, $v(k)$ denotes the process noise vector, $\hat{x}(k+1 \mid k+1)$ represents the updated estimate of x, $\hat{x}(k+1 \mid k)$ corresponds to the predicted value at time step $k+1$, and $v(k+1)$ denotes the residual error, called innovation factor. The parameters α and β, respectively, are used for weighting the target position and velocity in the update state computation.

Since the innovation factor $v(k)$ represents a measure of the error of $\hat{z}(k+1)$, a decision rule focused on the uncertainty measurement can be computed in order to determine the dimension of $\mathbb{G}$. Two criteria are considered in the decision rule design. The first one is that small values of the innovation factor indicate low uncertainty about its estimate and therefore, a reduced size of $\mathbb{G}$. The second criterion is that the dimension of $\mathbb{G}$ must be a $2^p \times 2^q$ value in order to assure that each terminal cell of $\mathbb{G}$ will contain only a single transformation after the last quadrisection operation of the A* search algorithm had been applied. Assuming these criteria, $M \times N$ is bounded by $2^{\min} \le v(k) \le 2^{\max}$. From this, the dimension $M \times N$ of $\mathbb{G}$ can be computed according to the following decision rule:

$$M, N = \begin{cases} 2^{\min}, & \text{if } w + 2^{\min} \le v(k) \\ 2^{\max}, & \text{if } w + 2^{\min} > v(k) \end{cases} \tag{9}$$

where w is computed according to the following expression

$$w = \phi(2^{\max} - 2^{\min}) \tag{10}$$

and ϕ is a parameter that weights the influence of the difference between both bounds in the selection of the appropriate value. Fig. 3 illustrates the integration of the the alpha-beta filtering with the computation of the adjustable set of transformations.

3.4 A* search algorithm

In this section, we describe the steps of the A* search algorithm for the template matching problem.

Input

$\mathbb{G}$: initial set of transformations.

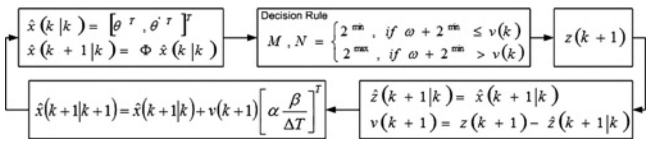
$\hat{x}(k|k) = [\theta^T, \dot{\theta}^T]^T$
$\hat{x}(k+1|k) = \Phi\hat{x}(k|k)$
Decision Rule
$M, N = \begin{cases} 2^{\min}, & if\ \omega + 2^{\min} \le v(k) \\ 2^{\max}, & if\ \omega + 2^{\min} > v(k) \end{cases}$
$z(k+1)$
$\hat{z}(k+1|k) = \hat{x}(k+1|k)$
$v(k+1) = z(k+1) - \hat{z}(k+1|k)$
$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + v(k+1)\left[\alpha \frac{\beta}{\Delta T}\right]^T$

Fig. 3 *Adjustable size-based set of transformations*

Computation of $M \times N$ dimension of $\mathbb{G}$

ε: distance error bound allowed.
$v(k)$: uncertainty measurement computed by the alpha–beta filtering approach.
ω: estimate which determines from what value the $2^{\max}$ or $2^{\min}$ bound value should be selected for computing $M \times N$ dimension of $\mathbb{G}$.
$2^{\max}$ and $2^{\min}$: upper and lower values for the innovation factor $v(k)$.
$D(P \| Q)$: value of Kullback–Leibler distance between the distribution functions that characterise the search state that is being evaluated, P, and the goal state, Q.
η: number of operators of type A applied from the initial search state to the current state n.

Output

$M \times N$ dimension of $\mathbb{G}$
$g_{\text{opt}} = [g_{x\text{opt}}, g_{y\text{opt}}]$

Algorithm

Step 1. Compute $M \times N$ dimension of $\mathbb{G}$:

$$M, N = \begin{cases} 2^{\min}, & \text{if } w + 2^{\min} \leq v(k) \\ 2^{\max}, & \text{if } w + 2^{\min} > v(k) \end{cases}$$

Step 2. Find g_{opt} such that $Q(g) = h_l(g(T(k)), I(k)) < \varepsilon$ is satisfied

While $Q(g) > \varepsilon$ or (all states are not made up by a single translation) Do

2.1. Split current state n into four new states $\{n\}_{i=1\cdots4}$
2.2. Compute $Q(g_c) \leftarrow h_l(g_c(T(k)), I(k))$ for each new state n_i
2.3. Expand the best state n_i according to the evaluation function $f(n) = c(n) + h^*(n)$:

2.3.1. $c(n) \leftarrow c(n-1) + \eta$
2.3.2. $h^*(n) \leftarrow D(P \| Q)$

End while

Step 3. Output $g_{\text{opt}} = [g_{x\text{opt}}, g_{y\text{opt}}]$ or no solution if all states conform a single translation and $Q(g^c) = h_l(g^c(T(k)), I(k)) > \varepsilon$ for each g^c.

A new A* search algorithm allowing weak solutions is computed with all the different views of the target that are stored in the STVM when the current A* search algorithm cannot provide a solution that satisfies $Q(g)$. This situation occurs when the target object has evolved significantly to a new shape in a sudden way. In particular, this happens in frames with deformable targets which introduce abrupt changes between frames such as turns, strechings, and bendings such as the Hand sequence shown in Fig. 5. A weak solution is a solution where the error bound ε of the quality of match $Q(g)$ is increased in one unit until a maximum value of 10. We denote this maximum error bound as $\varepsilon_{\max}$.

4 Updating framework for computing template changes

The new template is only updated when the target object has evolved significantly to a new shape. Since the A* search algorithm defines the 2D motion component, a solution g that satisfies $Q(g)$ with a error bound ε denotes that target object is moving according to a translational motion and no different aspect of the target is becoming visible. On the other hand, new 2D shape changes take place if the A* search algorithm computes a weak solution, that is, a solution g that satisfies $Q(g)$ with a maximum error bound $\varepsilon_{\max}$.

4.1 Short-time visual memory

The different templates that compose STVM represents the more common views of the target object. With the purpose of minimising redundancies, the problem about what representative templates should be stored is addressed as a dynamic approach that removes the less distinctive templates in order to leave space to the new ones. In order to determine the weight of every template, we introduce a relevance index, which is defined according to

$$R(k, i) = \frac{T_p(i)}{1 + k - T_s(i)} \tag{11}$$

where k represents the time step, i corresponds to the identification symbol template, $T_p(i)$ represents the frequency of use as template, and $T_s(i)$ corresponds to the time from the last instant it was used as the current template. A new template is inserted into STVM, when the quality of match $Q(g)$ ranges from ε to $\varepsilon_{\max}$. The template with the lower index of relevance is removed from STVM when the stack of templates is full and the new template is included into STVM.

4.2 Template updating algorithm

According to the value of $Q(g_{\text{opt}})$ computed by the A* search algorithm, each current template $T(k)$ is updated as $T(k+1)$ based on one of the steps of the following algorithm:

Step 1: If $Q(g_{\text{opt}}) \leq \varepsilon$, the new template in time step $k+1$, $T(k+1)$, is equivalent to the best matching of $T(k)$ in $I(k)$. That is, the edge points of $I(k)$ that are directly overlapping on some edge point of the best matching, $g_{\text{opt}}(T(k))$, represent the new template $T(k+1)$

$$T(k+1) \leftarrow \{i \in I(k) \mid \min_{t \in T(k)} \|g_{\text{opt}}(t) - i\| = 0\}$$

Step 2: If some template of STVM computes the best matching when the current template $T(k)$ cannot match the target object with an inferior or equivalent distance value ε, $Q(g_{\text{opt}}) > \varepsilon$, this template of STVM is selected as the new template $T(k+1)$. Otherwise, the current template, $T(k)$ is updated by incorporating the shape change by means of the partial directed Hausdorff distance measure according to expression (4) In this context, we denote $\text{STVM} = \{T(\text{STVM})_i\}_{i=1}^{N}$ as the different templates that integrate STVM, $Q(g; T(\text{STVM})_i, I(k), \varepsilon)$ as the best error bound distance ε computed for the ith template of STVM.

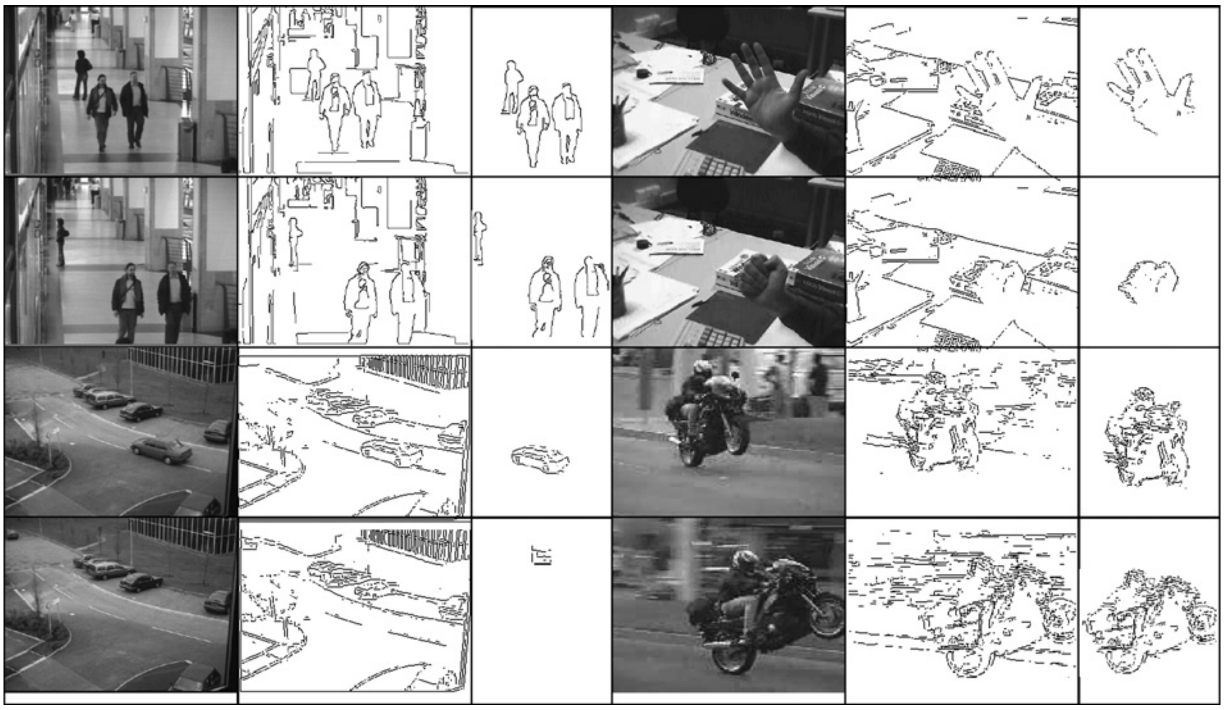

Fig. 4 *Indoor sequences: People sequence, Hand sequence (frames 150 and 250 are shown) and outdoor sequences: Car sequence (frames 150 and 250 are shown) and Motorcycle sequence (frames 10 and 50 are shown)*

The updating process is expressed as (see equation at the bottom of the page)

Step 3: If the best matching computed using the current template $T(k)$ and all templates of STMS is superior to the error bound distance $\varepsilon_{\max}$, no template is updated

$$T(k+1) \leftarrow \{\phi \text{ if } Q(g_{\text{opt}}; T(k), I(k), \varepsilon) \geq \varepsilon_{\max} \quad \text{and} \quad Q(g_{\text{opt}}; T(\text{STVM})_i, I(k), \varepsilon) \geq \varepsilon_{\max}\}$$

5 Extensions to the basic tracking approach

In this section, two extensions to the basic tracking approach described above are presented. The first of these is multiple object tracking. The second extension is searching for objects with a reduced size of edge points.

5.1 *Multiple object tracking*

The tracking approach described may work for sequences containing multiple moving objects. In such cases each target to be tracked is modelled as a visual tracking process. Using tracking processes, each target in image sequences such as the People sequence shown in Fig. 4 is associated with an individual process. In this way, we will have as many processes as targets to be tracked. The steps for each tracking process are as follows: (i) computing template position using the search framework, and (ii) computing updates of templates and STVM.

5.2 *Tracking of disappearing objects*

The tracking approach described thus far does not work well when the template being tracked becomes a template with a reduced set of edge points. For instance, frames in which much of the object is dissapearing from view such as the Car sequence shown in Fig. 4. One means of dealing with this problem is to combine our approach with more perceptual pathways such as the use of intensity information. In such cases targets are located by sliding intensity templates through the search intensity window when the bounding box area, which includes the template is below a margin error. Similarity between overlapped images is measured using a sum of squared differences (SSD), although any other measure can be used. After the intensity template is located, the heuristic search framework is computed in the best matching position. If the results of both approaches confirm a matching, a new template is updated according to the approach described in Section 4. Otherwise, the tracking approach enters in a ‘suspicious mode’ in which additional templates from the STVM are used in the best matching position computed by the SSD approach. Once the tracking process goes into suspicious mode, it stays that way for several frames, waiting for the reappearing of the target.

6 Experiments and results

This section illustrates the performance and average runtime of the tracking approach under a variety of circumstances, noting particulary the effects of the use of arbitrary shapes and the use of mobile and fixed acquisition cameras. All

$$T(k+1) \leftarrow \left\{ \begin{array}{ll} \{i \in I(k) \mid \min\limits_{t \in T(k)} \|g_{\text{opt}}(t) - i\| \leq \delta\}, & \text{if } Q(g_{\text{opt}}; T(k), I(k), \varepsilon) \leq Q(g_{\text{opt}}; T(\text{STVM})_i, I(k), \varepsilon) \\ T(\text{STVM})_i, & \text{if } Q(g_{\text{opt}}; T(\text{STVM})_i, I(k), \varepsilon) < Q(g_{\text{opt}}; T(k), I(k), \varepsilon) \end{array} \right\}$$

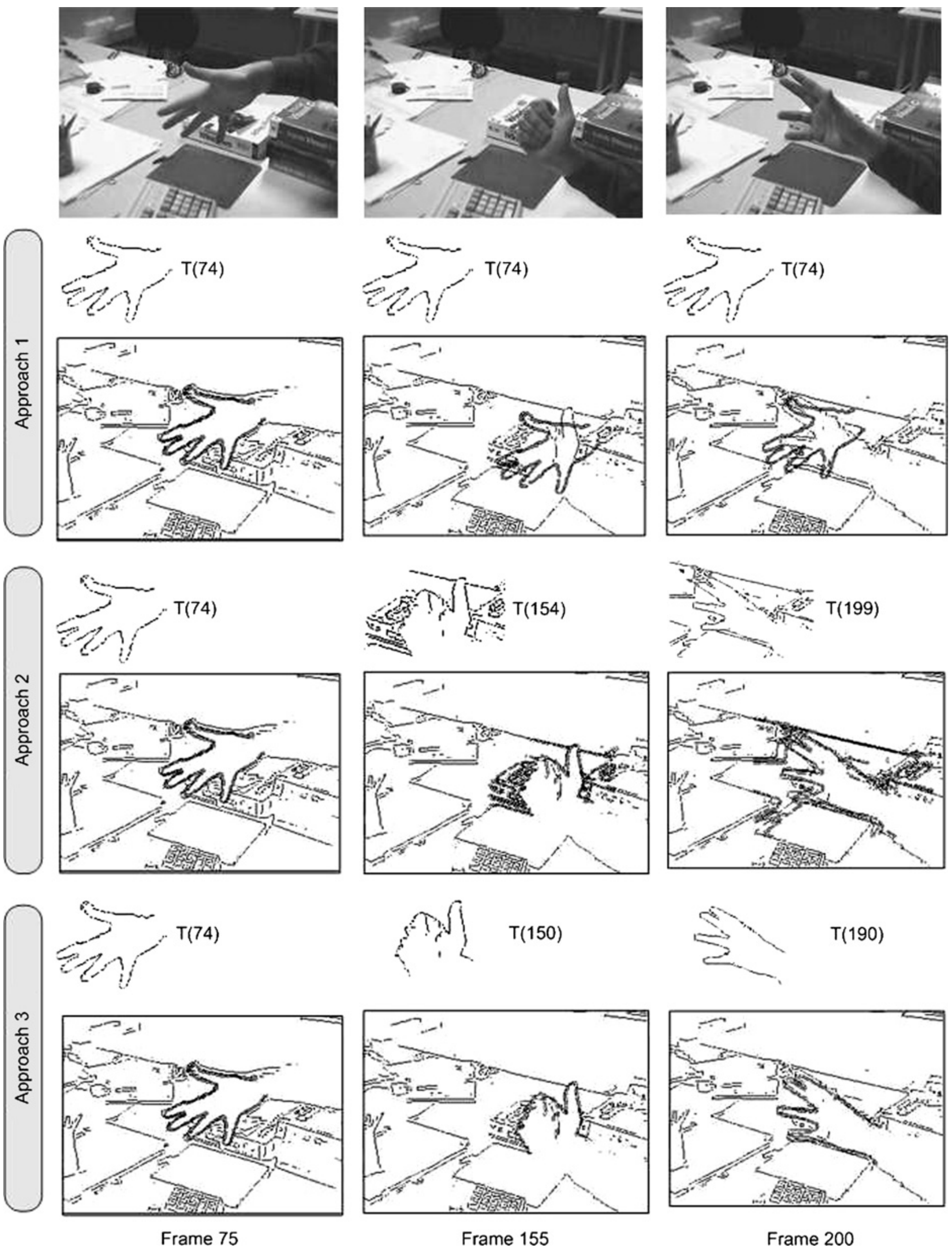

Fig. 5 *Qualitative comparison of template updating approaches 1, 2 and 3*

With approach 1, the template is not updated and the tracking process fails in frames 155 and 200, because the template does not reflect the change of the hand shape. With approach 2, the template is updated every frame. In this situation, the tracking process fails because the template has also been updated in those situations where the target hand has not been evolved to a new shape and therefore the template was constructed from background edges such as $T(154)$ and $T(199)$. With approach 3, the template is only updated when the target hand has evolved to a new shape. With this approach, the target hand is successfully followed and the template is appropriately updated across the sequence

experiments were performed by an Intel Core2 Duo 3.0 GHz. In order to compare the approach proposed with previous works, we use the same values of parameters that have previously been used. In the experiments performed, the goal state of the A* search algorithm is defined as the translation g that verifies that 80% (parameter $l = 0.8$) of template edge points are at maximum 2 pixels distance (error bound distance $\varepsilon = 2.0$) from the current explored image $I(k)$. The dimension of the short-time visual memory (STVM) is settled up to 6.

Fig. 4 illustrates four sample frames of indoor sequences (People and Hand sequence) and other four sample frames of outdoor sequences (Car and Motorcycle). People sequence is from the CAVIAR database [20]. This sequence shows a clip with multiple people in an inside corridor. The processing results of this sequence in the following sections include the data of the tracking of three people through the sequence. In Fig. 4, the first and fourth columns show original frames; the second and fifth columns depict the edge image; third and sixth columns show edge located template.

6.1 Comparative analysis between search strategies and average runtime

Table 1 shows the performance measured in seconds between the proposed A* search algorithm with the conventional blind search strategy [6] that works by subdividing the transformations space and no heuristic function is used to guide the search process, and the template updating process for each video stream. The results show that the A* heuristic search algorithm is faster than the blind search strategy in an average rate of four times better, and the time for processing each frame of sequences People, Hand, Car and Motorcycle is, respectively: 36, 17, 11 and 14 ms. The results of the procesing time for People sequence corresponds to the time for tracking three people through an inside corridor. These results confirm the adaptation of the tracker proposed to real-time restrictions (processing of each frame in a maximum time of 40 ms).

6.2 Comparative analysis between template-updating approaches

Fig. 5 illustrates a qualitative comparison of template updating approaches with the Hand sequence. This figure shows for each approach the original frame, the appearance of the template to be matched at $I(k)$, the result of the matching at $I(k)$, and the step time k from which was computed the template. For example, $T(74)$ denotes that the template was computed from step time $k = 74$.

In order to test the robustness of the template updating algorithm, we evaluate the performance when a short-time memory is incorporated and we evaluate the number of template updates for each sequence, testing: (i) our approach based on updating templates only when the target has evolved to a new shape, and (ii) the template updating approach used in [5, 6] and [11] based on continuous updating at every frame using the Hausdorff measurement.

Table 1 Comparative runtime between search strategies and total runtime in seconds for processing each process of each sequence (A* search strategy and template updating)

Sequence	People	Hand	Car	Motorcycle
N° of frames	745	512	414	70
seconds sequence	30	20	17	3
time required to process blind search strategy	64	21.98	15.8	2.6
time required to process A* search strategy	16.2	5.48	3.96	0.6
average rate between search strategies	4	4	4	4.3
time required to process template updating	10.6	3.29	0.7	0.4
total (s) to process the sequence	26.8	8.78	4.65	1.0

Table 2 summarises the results showing that the number of required updates is minimised in relation to other updating approaches based on Hausdorff measurement [5, 6] and [11]. No target object was drifted using the proposed approach; however, templates were drifted in sequences People, Hand and Motorcycle using continuous updating approach at every frame. Moreover, the use of a short-term memory avoids the loss of the target in certain situations such as illustrated in Table 2. These templates from STVM are used when: (i) an imprecision error of the edge detection process takes place (People and Hand sequence), (ii) disappearance and reappearance conditions of the target object along the video stream (Car and People sequence) and (iii) occlusion conditions of the target (People, Hand, Car, and Motorcycle sequence).

As comparisons to evaluate the effectiveness and efficiency of our tracking approach with template updating, we also show the quantitative performance evaluation measure between our approach and the robust fragments-based tracking using the integral histogram method (which is called FragTrack) without template updating of Adam *et al.* [21]. For the sequences evaluated we manually marked the ground truth, where the target centroid position error is calculated as the Euclidian distance between the centroid of the bounding boxes of the ground truth and the tracking results. Table 3 shows the results of the position error of the FragTrack tracker and our tracker. As can be seen from Table 3, the quantitative performance for People and Car sequence is similar; both methods work well with partial occlusions, pose changes and little scale changes. However, our method outperforms FragTrack tracker in two situations: (i) in Hand sequence, where the target objet is an deformable object with uniform in color (a flexible hand that twists, bends, and rotates on a desk with different objects around it) and (ii) in Motorcycle sequence, where the target changes its appearance greatly with fast moving in a traffic environment with a mobile camera with zoom features. The best performance of our method in these two situations is because of template updating when the target has evolved to a new shape and the no introduction of

Table 2 Template updating: number of template updates, total number of different templates stored in STVM during the tracking process for each sequence, total number of templates used from STVM during the tracking process and number of frames where the target was retrieved using STVM

Sequence	People	Hand	Car	Motorcycle
N° of frames	745	512	414	70
number of updates	380	300	200	60
number of different templates stored in STVM	124	19	6	6
number of templates used from STVM	36	12	6	4
number of frames where the target was retrieved using STVM	18	10	8	4

Table 3 Quantitative performance evaluation measure between the FragTrack method and our method

Sequence	People	Hand	Car	Motorcycle
position error – FragTrack	0.091	0.433	0.097	0.241
position error – our method	0.093	0.073	0.093	0.051

occluding objects into the template while this one is evolving to a new shape.

7 Conclusions and future work

This paper is concerned with fast and accurate tracking of arbitrary shapes in video streams without any assumption of the speed and trajectory of the objects. It is proposed an A* search framework in the space of transformations to compute efficient target motion. 2D shape change is captured with 2D templates that evolve with time. These templates are only updated when the target object has evolved to a new shape change. The representative temporal variations of the target shape are enclosed in a STVM. The proposed template-based tracking system has been tested and has been empirically proved that: (i) the A* search proposed is faster than previous search strategies in an average rate of four times better, allowing real-time performance on general purpose hardware, (ii) although abrupt motions cannot be predicted by an alpha-beta filtering approach, the tracker performance was well adapted to the non-stationary character of the person's movement, which alternates abruptly between slow and fast motion such as the People sequence, (iii) target shape evolution introduces, in certain situations, views that cannot be matched in the current image $I(k)$ using the A* heuristic search. These situations are presented when the target shape is represented by some sparse edge points and its size dimension is extremely reduced because of disappearance and reappearance conditions from the current image $I(k)$, and (4) updating templates using combined results focused on the value of the quality of match and the use of a short-term memory lead to accurate template based tracking.

This work leaves a number of open possibilities that may be worth further research, among others it may be interesting to propose a new tracking approach with the following steps: (i) divide the target template into non-overlapping fragments or patches after an edge-enhancement process, (ii) matching each image patch fragment with the corresponding fragment of the template-size section in the search region, (iii) computing the final similarity measure by comparing the values obtained for each fragment, and (iv) fragment level template updating to address the problems of strong clutter and greatly target changes. In order to generate more stable edge sequences and reduce the illumination sensitivity, edge-enhancement can be performed by means of the use of an anisotropic diffusion filter instead of the Gaussian filter, and replacing the hysteresis step of Canny detector by a dynamic threshold process for reducing blinking effect of edges during successive frames.

8 Acknowledgment

This work was supported by the Spanish Government through the Ministry of Science and Innovation under the project TIN2008-06570-C04-03.

9 References

1 Baker, S., Matthews, I.: 'Lucas Kanade 20 Years On: a unifying framework', *Int. J. Comput. Vis.*, 2004, **56**, (3), pp. 221–255

2 Belongie, S., Malik, J., Puzicha, J.: 'Shape matching and object recognition using shape context', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2002, **24**, (4), pp. 509–522

3 Besl, P.J., McKay, N.: 'A method for registration of 3-D shapes', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1992, **14**, (2), pp. 239–256

4 Chen, Y., Medioni, G.: 'Object modelling by registration of multiple range images', *Image Vis. Comput.*, 1992, **10**, (3), pp. 145–155

5 Parra, R., Devy, M., Briot, M.: '3D modelling and robot localization from visual and range data in natural scenes', (*LNCS*, **1542**), (Springer-Verlag, 1999), pp. 450–468

6 Rucklidge, W.J.: 'Efficient computation of the minimum Hausdorff distance for visual recognition', (*LNCS*, **1173**), (Springer-Verlag, 1996)

7 Schlegel, C., Illmann, J., Jaberg, H., Schuster, M., Worz, R.: 'Integrating Vision based bejaviours with an autonomous robot', (*LNCS*, **1542**), (Springer-Verlag, 1999), pp. 1–20

8 Comaniciu, D., Ramesh, V., Meer, P.: 'Real-time tracking of non-rigid objects using mean shift'. Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2000, vol. II, pp. 142–149

9 Hager, G.D., Belhumeur, P.N.: 'Efficient region tracking with parametric models of geometry and illumination', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1998, **20**, (10), pp. 1025–1039

10 Liu, T.-L., Chen, H.-T.: 'Real-time tracking using trust-region methods', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004, **26**, (3), pp. 397–401

11 Sánchez Nielsen, E., Hernández Tejera, M.: 'Tracking moving objects using the Hausdorff distance. A method and experiments', *Front. Artif. Intell. Appl.: Pattern Recog. Appl.*, 2001 IOSPRess, pp. 164–172

12 Reynolds, J.: 'Autonomous underwater vehicle: vision system'. PhD thesis, Robotic systems Laboratory, Department of Engineering, Australian National University, 1998

13 Matthews, I., Ishikawa, T., Baker, S.: 'The template update problem', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004, **26**, (6), pp. 810–815

14 Huttenlocher, D.P., Klanderman, G.A., Rucklidge, W.J.: 'Comparing images using the hausdorff distance', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1993, **15**, (9), pp. 850–863

15 Canny, J.: 'A computational approach to edge detection', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1986, **8**, (6), pp. 679–698

16 Pearl, J.: 'Heuristics. Intelligent search strategies for computer problem solving' (Addison-Wesley Series Artificial Intelligence Press, 1984)

17 Kullback, S.: 'Information theory and statistics' (John Wiley and Sons Press, New York, 1959)

18 Cover Thomas, M., Thomas, J.A.: 'Elements of information theory' (John Wiley and Sons, Inc., New York, 1991)

19 Bar-Shalom, Y., Xiao-Rong, Li.: 'Estimation and tracking: principles, techniques, and software' (Artech House Press, Boston, 1993)

20 Caviar datasets: http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/, accessed September 2010

21 Adam, A., Rivlin, E., Shimshoni, I.: 'Robust fragments-based tracking using the integral histogram'. Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2006