

Real-Time 2D Shape Tracking with MOVels and Affine Transformations

Giancarlo Iannizzotto, Lorenzo Vita

Abstract

In this paper we present a new real-time algorithm for shape tracking over a sequence of 2D images. This algorithm is able to track a shape even in presence of substantial perspective transformations and to detect changes in the shape due to changes in the object structure. The core of the system is a chain made of a sort of autonomous software agents (MOVels), which acts like an active contour on the sequence of frames. The performances and experimental results are encouraging, even if much is still to be done as regards implementation and optimization of the code.

I. INTRODUCTION

Shape tracking is one of the main issues in real-time computer vision. Most applications of real-time computer vision deal with detection, analysis and eventually recognition of objects in dynamic scenes, i.e. scenes with moving or variable parts, or varying parameters. A particular case occurs when the vision system is able to move: in this evenience, it is often necessary to identify in the scene some kind of features which can be considered as fixed or, at least, having a predictable behaviour (*fiducial features*). By tracking those features over time it is possible to understand how the system is moving and, for example, avoid obstacles or follow a precise path while moving. Tracking objects over the time is also extremely useful in automatic surveillance, traffic control, factory automation. Other applications include on-line digital video characterization in the field of continuous media delivery (Video on Demand, etc), human-machine interaction (human body visual tracking), medical surveillance, medical analysis, and some particular surgery techniques.

Differently from off-line (batch) video characterization, which aims to extract from videos information to be used for database indexing, traffic shaping (setting communication networks parameters in order to obtain adequate Quality of Service for media delivery) or shots mosaicing (“the representation of an entire (video) shot in a single frame, which includes all its static and dynamic information without redundancy...”[1]), real-time video analysis is subject to strict temporal constraints (*deadlines*) which, if missed, may conduct to quality of service degradation (soft deadlines) or even to some kind of failure (hard deadlines). Most common applications require a rate of 10 to 30 frames per second, according to the dynamics of the process to control or to observe.

Very often real-time shape tracking algorithms heavily rely on *active contours*. There are two main reasons for this convergence:

1. since the objects to be tracked can change in shape, position, orientation and distance from the camera, and since the image perceived from the vision system is substantially a projection of the 3D scene onto a 2D plane, it is necessary to adopt a dynamic model to represent the shapes of the objects;
2. active contours have proved to be stable and reliable, and, moreover, it is relatively easy to implement an active contour with appropriate properties to deal with almost any particular application [2] [3].

Active contours are models representing open or closed curves that, having a very high ability of morphological adaptation, can accurately reveal contours of any shape. These models are called *active* because they automatically respond to specific characteristics of the points of the image to which are applied, by changing their shape consequently. For example, an active contour can respond to the edgeness values of the points of the image by changing its shape till it fits all of the points characterized by the (locally) highest values of edgeness (*edgeness local maxima*). A particular type of active contour is the *snake*: it responds to the characteristics of the points of the image through the minimization of a quantity called external energy

G. Iannizzotto is with the Department of Mathematics, University of Messina, C.da Papardo, Salita Sperone, 98166, Messina - Italy. E-mail: ianni@ingegneria.unime.it .

L. Vita is with the Istituto di Informatica e Telecomunicazioni, University of Catania, Viale A. Doria 6, 95025 Catania - Italy. E-mail: vita@iit.unict.it

and, moreover, responds to specific internal laws ruling shape and way of deformation, tending to minimize a quantity called internal energy [4] [5]. An interesting variation to the theme of snakes is the *snake spline*, which uses a spline parametric representation of the shape modeled [6]. Many other active, deformable models have been developed, most of them being very effective (see, for example: [7]). A unifying approach to the described techniques can be found in [8], where a class of constrained clustering algorithms for boundary extraction (as a generalization of known algorithms) is introduced.

Real-time applications of active contours have been explored since the end of the 80's, and very interesting results using snakes and affine invariance can be found, among the others, in [9]. However, due to the need for calculating internal and external energy, it is our opinion that snakes suffer from an intrinsic high computational complexity which could slow down the tracking process. Moreover, they usually introduce a *smoothing* of the tracked shape, which sometimes can be undesired. Following this opinion we developed a different active contour, initially deemed for fast image segmentation and subsequently applied to real-time shape tracking, with encouraging results.

In this paper we present **Real-time MOVels**, a new technique to realize a real-time active contour which is characterized by intrinsically low computational complexity and no smoothing of the tracked contours, and is easy to parallelize due to the locality of the information used during the tracking process. In the next sections we describe the Real-time MOVels paradigm and present some experimental results and some conclusions and future work plans.

II. IMAGE SEGMENTATION WITH MOVELS

Image segmentation is the first, essential step for quite any shape tracking algorithm. Unless we consider a preliminar, operator-driven initialization of the shape to track, it is in fact mandatory to identify this shape, i.e., extract the boundary contour of the object and represent it in a suitable way [9]. Image segmentation can be defined as partitioning the image in disjoint areas, one representing the image background and the others representing distinct objects. Since an image area is always surrounded by a closed contour, and since the most relevant information about the shape of a planar area is embedded in its contour, finding closed contours is exactly what we have to do if we are interested in 2D object shapes. We developed an active contour for this task which is based on a continuous, closed chain of autonomous agents that act together on the image plane like a sort of cellular tissue, detecting the contours of the objects as closed curves and thus segmenting the image [10][11][12]. For reasons of space we won't describe the segmentation algorithm: more informations about it can be found in the cited papers and in our web site¹.

We apply our segmentation algorithm to extract the shape of the object to track from the first frame of the sequence: this process requires about 2 seconds according to the complexity of the image and the number of object it contains, but this relatively long time is spent just for this first frame, and can be regarded as a "setup time" for the system. After this initialization, the system knows the original shape, position and orientation of the object and will track it over the time at a rate which can be estimated at about 10 frames per second².

III. SHAPE TRACKING WITH REAL-TIME MOVELS

The problem of tracking a real-world (i.e. 3D) object by using just a sequence of 2D digital images is not trivial. During the initialization phase, a 2D shape is identified for the object, which is the *perspective projection* of the 3D object onto the image plane. As the object moves, its projection changes not only in position and orientation, but also in its shape and size, according to the laws of projective geometry. In order to track the shape of the object in the 2D image we therefore need a model for this projective transformation: without such a model the system will loose the right shape of the object, being misled by noise, other objects in the scene, changes in the illumination, and so on [9]. However, it is also useful to have some way to

¹<http://www.medialab.unict.it/~ianni>

²As will be better explained in section IV, our algorithm is currently implemented in Matlab, so it is definitely slower than this estimated speed; nevertheless, we can assert that once correctly ported in C and optimized, the tracking process will easily run at this speed.

detect substantial changes in the shape of the object which are not imputable to projective transformations. Examples of such cases are:

- The real object can be composed of different parts which could move one respect to the other during the tracking process.
- During the initialization phase, the segmentation process could have been misled by occlusions between distinct objects, shadowing or illumination effects, perspective effects and so on, thus producing a 2D model which is not the model of the object to track, but some kind of *merging* of multiple objects. This case can be identified only when the object moves, as it will probably move in a different way than the other objects in the image.
- During the tracking process, the system can be “distracted” from the right object to track, due to causes similar to those just mentioned above; this is always possible, even when we impose perspective-compliant constraints to the model.

What we need, so far, is a 2D model which is able to follow the changes in shape caused by perspective transformations, detect substantial changes in the shape not originated by perspective transformations, adapt to changes in the object’s 3D shape. And, the model has to be *fast*. In off-line image analysis, the most interesting strategy to deal with 3D-to-2D transformation is currently full perspective (and projective) modeling, which is actually the only way to “capture the true perspective distortion that turns planar parallelograms into trapezoids in the 2D image”[13]. In the real-time computer vision context, *affine models* (i.e. weak perspective and paraperspective) are still more popular, due to their faster computability. Following this tendency, we currently adopt an affine model to constraint our active contour. Moreover, according to the need for speed, we have decided to avoid the computational overload due to the calculation of internal/external energy which characterizes the snakes and present an alternative solution based on MOVels. MOVels never calculate energies or gradients: they just move according to a few simple rules and to a low-cost, brightness contrast function, which permits the identification of the new matching position for each point of the object’s contour in the new frame. This allows to track the shape of the object over the sequence of frames. In the rest of this paper we will assume that the initial segmentation has already been performed (as stated above, we use our MOVels-based segmentation algorithm for this phase), producing a closed chain of points, in form of a $2 \times n$ coordinate matrix, where n is obviously the number of points in the chain. Each of those points will become a Real-Time MOVel in the following steps.

In figure 1 is described the main tracking algorithm: matrices $A^{(2 \times 2)}$ and $B^{(1 \times 2)}$ are the transformation matrices for the cartesian coordinates of each MOVel in the image plane, according to equation (1), and matrix I_m is the $m \times m$ identity matrix.

$$X_{i+1} = A_i X_i + B_i \tag{1}$$

$$X_i = [x, y] \tag{2}$$

Main Routine	
start:	Initialize $A = I_2$, $B = [0, 0]$; $i = 1$;
loop:	Repeat for each chain
	Transform the MOVels coordinates according to eqn. (1);
	Place the chain in the frame $i + 1$;
	Let the MOVels move in the image (i.e. Call MOVE_Chain function);
	Calculate the global <i>position spread index PSI</i> ;
	If $PSI > threshold$ then recalculate A and B ;
	GOTO loop ;

Fig. 1. Main algorithm definition

The algorithm is repeated for each chain because, due to the splitting mechanism described below, even starting with a single chain could lead to multiple chains. The predictive model applied is of *constant velocity*,

i.e., we suppose that the affine parameters (A and B) don't change or change very slowly between two consecutive frames. The function `MOVE_Chain` moves each `MOVEl` looking for a point in the image which matches, according to the local brightness contrast and the pointwise brightness value, the point on which the `MOVEl` was in the previous frame. In short, the `MOVEl` has been placed in the new frame in a position which is calculated according to equation (1); starting from this position it looks for its best-matching position. If the object has moved according to the affine transformation defined by A and B , the `MOVEl` will not move at all; if the estimate, for some reason, is wrong, the `MOVEl` will have to look for its new position, moving around in the image. The search is constrained on the line perpendicular to the chain and limited by a predefined maximum distance. When the `MOVEl` stops, either because it has found its new position or because it has reached its maximum distance from its original position, the distance it has covered is recorded. The distances covered by all the `MOVEl`s of a chain have, at each loop, two distinct uses:

- The sum of all the distances (the global PSI) is used to decide whether recalculate A and B ;
- The single distances are used to clusterize the `MOVEl`s which have moved so much to suggest that some modification could have occurred to the shape of the object: if there is any section of the chain which is characterized by long distances while the rest of the chain is still following the model defined by A and B , it is possible that the object is changing its shape or, perhaps, that what we are tracking is not a single object. In this case, the chain *splits* into two or more chains, one for each section of the chain. Of course, each section is closed on itself during the splitting process, thus leading to distinct, closed contours. The clustering algorithm is based on a dual-threshold technique, similar to that used by Canny in his well known edge following algorithm [14]. The splitting algorithm is identical to that used in our segmentation system [12] [10].

It should be noted that while the `MOVEl`s move in the image, the distance between two consecutive `MOVEl`s can increase: since we adopted the hypothesis of *closed contour*, it is mandatory that in this case the space between the two `MOVEl`s is filled by interpolation. Similarly, when it happens that two `MOVEl`s, due to some shrinking, occupy the same image position, the first of them is eliminated. This mechanism is the same adopted in the segmentation algorithm [10].

The new values of A and B can be easily recalculated from any adequate choice of `MOVEl`s from the chain. The choice should be done ensuring a uniform distribution of the samples over the chain. As regards the number of samples, according to the number of `MOVEl`s compounding the chain, a choice of about the 20%, but no more than 30 of them, leads to a good estimate of A and B . There are many different methods to estimate A and B : we use a least-squares estimation, but different ways are possible.

IV. EXPERIMENTAL RESULTS

The work is still in progress: in particular, the current implementation is in Matlab and has not been optimized yet. Moreover, if we don't consider the most common visual approach to validate the results obtained, we haven't yet developed a method to objectively evaluate the real accuracy of the algorithm during the shape tracking. Nevertheless, due to the low computational complexity and relatively good speed of the algorithm (in Matlab, it needs about 2 seconds per frame to complete the computation), it is not difficult to estimate a speed-up of about 20 to 25 after its porting in C and after optimization. The hardware we currently use for Matlab tests is a common PC with Pentium II processor 300MHz, running Windows NT operating system. We plan to port the system in C under Linux as soon as the algorithm will be complete.

In figure 2 we present some frames of a test sequence. In the first frame the initial shape is represented, while the second frame represents the shape as it is returned by the segmentation algorithm. The shape is tracked by our Real Time `MOVEl`s over the sequence, and the position of the `MOVEl`s of the chain is marked, in each frame, in white. Some considerations about the testing environment: we currently don't use color information, so during the process the sequence is converted to 32 grey levels. Since we are interested in a low-cost implementation, a cheap webcam is used as input device. The illumination conditions are those of a typical real case: low contrast, not-omnidirectional light source, presence of shadows, and so on. As it can be seen, the system tracks correctly the shape over the frames, even in presence of evident deformations due to perspective transformations. The frame `f` shows a case of *splitting*. In this case, the second chain produced by the splitting process is coloured in black.



a: first frame



b: frame segmented



c: frame 11



d: frame 23



e: frame 40



f: frame 50

Fig. 2. Definition of Turning Angles

V. CONCLUSIONS AND FUTURE WORK

We have presented a new real-time algorithm for shape tracking over a sequence of 2D images taken from a 3D real-world scene. The system is still under development, so only preliminar results can be considered. In particular, the speed performances can be just estimated since the current prototype is not optimized and is implemented in Matlab. Nevertheless, as stressed above, the results are promising. Good tracking accuracy and relative speed have been obtained, and much can be done to further enhance them. Due to its prediction-correction strategy, under the hypothesis of smooth changes of velocity between consecutive frames, the algorithm can be usefully embedded in a soft or even real-time control environment. If, for any reason, the system is not able to complete the analysis of one or more frames, the predictor gives an estimate which can be accurate enough to avoid disasters in most cases. The analysis of the following frames will correct the model and conduct the system on the right way. Future work comprises the introduction of a parallel thread which will calculate a more complete prediction model in order to take account of sudden changes in the motion parameters of the object; the automatic selection of the thresholds according to the global image characteristics; use of color to better identify the matching points; further optimization for better speed performances.

REFERENCES

- [1] Alberto Del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann Publishers, 1999.
- [2] D. Metaxas and D. Terzopoulos, "Shape and nonrigid motion estimation through physics-based synthesis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15 (6), pp. 580–591, 1993.
- [3] R. Szeliski and D. Terzopoulos, "Physically-based and probabilistic modeling for computer vision," in *SPIE Proceedings of: Geometric Methods in Computer Vision*, July 1991, vol. 1570, pp. 140–152.
- [4] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, pp. 321–333, 1988.
- [5] K. F. Lai and R. T. Chin, "Deformable contours: Modeling and extraction," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1994.
- [6] M. Flickner, H. Sawhney, D. Pryor, and J. Lotspiech, "Intelligent interactive image outlining using spline snakes," in *Proceedings of 28th Asilomar Conference on Signals, Systems, and Computers*, 1994.
- [7] R. Malladi, J. A. Sethian, and B.C. Vemuri, "Shape modelling with front propagation," *IEEE Trans. on PAMI*, vol. 17(2), Feb. 1995.
- [8] R. Malladi and J. A. Sethian, "A unified approach to noise removal, image enhancement, and shape recovery," *IEEE Trans. on Image Processing*, vol. 5 (11), November 1996.
- [9] A. Zisserman, A. Blake, R. Curwen, "A framework for spatio-temporal control in the tracking of visual contours," in *Real-Time Computer Vision*, C. M. Brown and D. Terzopoulos, Eds. 1994, pp. 3–33, Cambridge University Press.
- [10] G. Iannizzotto and L. Vita, "Fast and accurate edge-based segmentation with no contour smoothing in 2-d real images," *to be published in IEEE Transactions on Image Processing*.
- [11] G. Iannizzotto and L. Vita, "A fast, accurate method to segment and retrieve object contours in real images," in *International Conf. on Image Processing*, Lausanne, Switzerland, 1996.
- [12] G. Iannizzotto and L. Vita, "Edge-based image segmentation with movels," in *ICMCS 99*, Firenze, ITALY, June 1999.
- [13] C. M. Brown and D. Terzopoulos, "Introduction," in *Real-Time Computer Vision*, C. M. Brown and D. Terzopoulos, Eds. 1994, pp. 3–33, Cambridge University Press.
- [14] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. on PAMI*, vol. 8(6):679-698, June 1986.