A CONTOUR TRACKING ALGORITHM FOR ROTOSCOPY

Vincent Garcia, Éric Debreuve, Michel Barlaud

Laboratoire I3S, Université de Nice - Sophia Antipolis
2000 route des Lucioles, 06903 Sophia Antipolis, FRANCE
{garciav,debreuve,barlaud}@i3s.unice.fr

ABSTRACT

Rotoscopy is a crucial processing for cinema post-production. It corresponds to the segmentation of an object of interest in a video in order to apply a local processing. In the industry, rotoscopy is usually performed manually, frame by frame. Semi-automatic algorithms have been proposed to reduce the load of this task. However, they classically use contour-based information and consequently lack robustness in case of occlusion. Here, we propose a region-based contour tracking algorithm relying on feature points which are temporally matched to build trajectories used to estimate a global or local deformation between a distant reference contour and the current frame. Then, we propose a rotoscopy algorithm based on a forward and a backward contour tracking. The use of region information and distant reference contours allows to avoid drift and greatly reduce the influence of occlusions. The rotoscopy algorithm was applied to CIF and SD sequences.

1. INTRODUCTION

Rotoscopy is the term used in post-production for the task of isolating a video object by defining its contour in each frame of a video. The purpose is to apply a specific image processing to a particular object. In cinema post-production, this segmentation is usually performed manually and frame by frame by so-called animators. As a consequence, it is a long, repetitive and expensive task. This is why rotoscopy is a very active research topic of video processing.
The rotoscopy problem can be expressed as follows. Given the object contour (represented by a 2D closed curve, *e.g.*, polygon, spline, etc.) in the first and the last frame of the sequence, we want to compute the object contour as precisely as possible in the intermediate frames.
Semi-automatic rotoscopy is already proposed in some softwares but they are classically based on linear contour interpolation. If the global segmentation is not satisfying, the user can refine it by defining an intermediate keyframe. Unfortunately, linear interpolation is not precise enough and the segmentation has to be refined too many times.
Active contours [1, 2] can be used for rotoscopy. Luo and Eleftheriadis [3] propose a rotoscopy algorithm where the contour interpolation problem is decomposed into two directional contour trackings and a merging problem. In this paper, active contours are used for the tracking part. However, a drift can be observed because of error propagation from frame to frame. Agarwala *et al.* [4] present an algorithm which uses active contours to minimize an energy written as a linear combination of shape terms and image terms. Because these methods are too local both spatially and temporally, they are not robust to occlusions.
In this paper, we propose a rotoscopy algorithm based on a forward and a backward contour tracking. This tracking is performed by applying a global or a local deformation to a reference contour. The deformation is deduced from the 2D+t trajectories of feature points. The trajectories, also called *tracks*, are formed by feature point matching. This approach is region-based, as opposed to previously cited methods, because feature points are selected inside the contour region on not only at the contour vicinity. It allows to manage large occlusions.
The paper is organized as follows: Section 2 describes the proposed contour tracking algorithm. Section 3 describes the proposed rotoscopy algorithm. Section 4 shows and analyzes some rotoscopy examples. Finally, Section 5 concludes and gives some perspectives.

This work was partly supported by *"Le Conseil Régional Provence-Alpes-Côte d'Azur"*, France.

2. CONTOUR TRACKING ALGORITHM

The classical approach for contour tracking consists in computing a deformation from one frame to the next. The proposed method is based on the following steps: feature point extraction, occlusion-robust track building, search for reference frames and estimation of the reference contour deformation.

2.1. Feature point extraction

The first step of our contour tracking algorithm is the extraction of *feature points* on each frame. A "feature point" or "corner" is a two dimensional feature including for example corners and junctions. The corner extraction problem has been abundantly studied in image processing research [5, 6].
A feature point is a two dimensional position in a specific image. However, to distinguish two different feature points, we define for each feature point a descriptor which describes its neighborhood by a set of values grouped in a vector or a matrix. Thus, feature points defined on different frames and which correspond to the same corner in the video have theoretically the same descriptor. However, in practice, the descriptor may slightly change and the L^2 norm between two descriptors has to be used to distinguish two feature points. A classical descriptor is a block of luminance values centered on the feature point and every image information as chrominance values, gradient norm, gradient direction or multi-spectral values can be used.

2.2. Building tracks

We have now the possibility to verify if two feature points defined on two different frames correspond to the same image corner or not. The extracted feature points and the associated descriptors are used to build a set of *tracks*. A "track" is the 2D+t trajectory of a feature point. We consider that a track has the following properties:

- a track is a set of feature points defined on at least two different frames,
- a track can start after the first keyframe and finish before the last keyframe,
- the track *lifetimes* are independent of each other.

We explain below a short example of the construction of a link between two feature points.
Let F_m be the m^{th} frame of the sequence with $m > 1$. Let p_m^b be the b^{th} feature point defined in F_m and d_m^b its associated descriptor. We find now the feature point p_l^a defined in a previous frame F_l which minimizes $\| d_m^b - d_i^j \|_{L^2} \quad \forall i < m \; \forall j$. After we find the feature point p_n^c defined in a next frame F_n which minimizes $\| d_l^a - d_i^j \|_{L^2} \; \forall i > l \; \forall j$. If p_m^b and p_n^c are the same feature point (*i.e.* $m = n$ and $b = c$), the feature points p_l^a and p_m^b verify cross-validation (or are corresponding) and the link $\{p_l^a, p_m^b\}$ is created. Otherwise no link is created. Fig. 1 shows an example of failed and successful cross-validation.

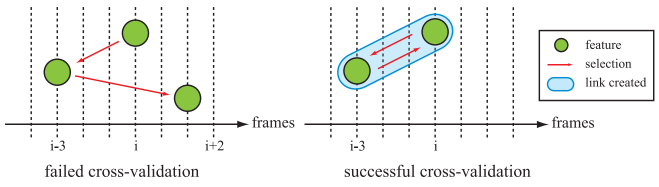

Fig. 1. Example of cross-validation.

This algorithm is used for all frames and all feature points and finally we have a set of links. Links are concatenated in order to build the set of tracks. Fig. 2 shows an example of tracks on ten frames of the sequence *Erik*.

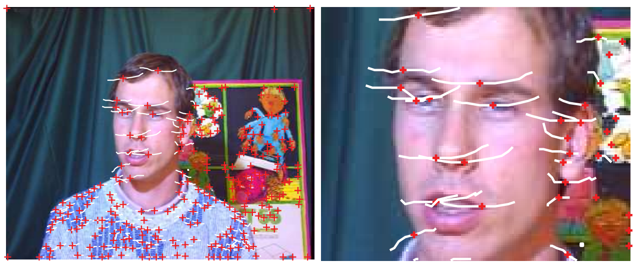

Fig. 2. Each curve is the 2D projection of a track to illustrate the 2D+t corner trajectory and the cross represents the actual position of the feature on the current frame.

2.3. Contour tracking

Let v be a video of n images or frames $F_1, \cdots, F_n$. Let c_1 be an initial contour (polygon, spline, B-spline, etc.) hand-edited on frame F_1 which segments an object. A contour tracking algorithm consists in computing from c_1 the contours $c_i, \forall i \in [2, n]$.
First we propose a global approach for contour tracking. However, global approach is not precise enough to represent the real contour object deformation. Therefore, we propose a local approach more adapted to our problem.

In both approaches, contour deformation is approximated by a local or a global matrix M of affine transformation with a variable number of parameters.

2.3.1. *Global approach*

Given the initial contour c_1, the contours $c_2, c_3, \cdots, c_n$ are computed frame by frame. Let us consider that contours $c_2, ...c_f$ have already been computed. c_{f+1} is computed from the tracks and the previous contours $c_1, ...c_f$ as follows:

- select all the tracks which exist on at least F_f and F_{f+1},
- among these tracks, keep all the tracks which were defined inside the previous contours $c_i, \forall i \in [1, f]$,
- find the reference frame F_r which is the first frame where all the selected tracks are defined,
- use the track positions on F_r and F_{f+1} to estimate the affine transformation matrix M,
- compute c_{f+1} by applying M to all the sample points of reference contour c_r.

In classical approach, contour c_{f+1} is always computed from c_f, and c_f from c_{f-1}, etc. Each computed contour is indirectly relative to the initial contour c_1. The miscalculations realized to compute c_f have repercussions on the computation of $c_{f+1}, \cdots, c_n$. To avoid this drift, we use in the proposed algorithm a reference contour which is not the very previous contour. Therefore, c_f is more directly relative to c_1.
However, there is still an issue. This contour tracking algorithm uses one reference frame and a global transformation matrix for the whole contour. As the transformation is the same for all the sample points of the contour, the contour transformation can only be an affine transformation. However, in a real video sequence, object transformation is generally more complex than a simple affine transformation. In addition, as object motion is non uniform, the estimation of the contour deformation can be biased as seen on Fig. 3. Therefore, we propose below a local approach.

2.3.2. *Local approach*

The main idea is to estimate one transformation matrix M for each sample point of the contour. For a given sample point, the matrix is estimated from the l^{th} closest tracks among the selected tracks used in the global approach. Likewise, we propose to use one local reference sample point and one local reference frame for each sample point. Thus, c_{f+1} is computed from the tracks and the previous contours $c_1, ...c_f$ for the local approach as follows:

- select all the tracks which exist on at least F_f and F_{f+1},
- among these tracks, keep all the tracks which were defined inside the previous contours $c_i, \forall i \in [1, f]$,
- find the global reference frame F_r which is the first frame where all the selected tracks are defined,
- for each sample point s_r^j of the contour c_r,
  - among the selected tracks, select a subset formed by the l^{th} closest tracks of s_r^j,
  - find the local reference frame $F_{\tilde{r}}$, with $\tilde{r} \leq r$, for s_f^j which is the first frame where all the tracks of the subset are defined,
  - use the subset track positions on $F_{\tilde{r}}$ and F_{f+1} to estimate the local affine transformation matrix M,
  - compute s_{f+1}^j by applying M to $s_{\tilde{r}}^j$.

2.3.3. Robustness to occlusions

The proposed algorithm is robust to occlusions because the track building can “jump” frames where there is occlusion.

3. ROTOSCOPY

Rotoscopy is a method to define a 2D+t segmentation for a chosen object. We propose in this section an algorithm of rotoscopy based on the previous contour tracking algorithm. First, we expose the proposed method with two keyframes. Second, we explain how user can refine the whole segmentation by adding new keyframes.

3.1. Two keyframes

The rotoscopy problem is the following. Let v be a video of n images. Let c_1 be the initial contour hand-edited on frame F_1 which segments an object and c_n be the final contour on frame F_n which segments the same object. Rotoscopy consists in computing the contours c_i $\forall i \in [2, n-1]$ from c_1 and c_n.
Basically, we divide the rotoscopy problem in a two directional contour tracking problem and a merging problem. Contour tracking algorithm can be applied in forward direction as seen before, but it can also be applied in backward direction because tracks are independent from temporal direction. The proposed rotoscopy algorithm is the following:

- compute the forward segmentation c_i^f $\forall i \in [2, n]$ by using contour tracking from initial contour $c_1^f = c_1$ in forward direction,
- compute the backward segmentation c_i^b $\forall i \in [1, n-1]$ by using contour tracking from final contour $c_n^b = c_n$ in backward direction,
- merge forward and backward segmentations by computing a weighted average segmentation C_i as follows:

$$C_i = \frac{n-i}{n-1}.c_i^f + \frac{i-1}{n-1}.c_i^b, \; \forall i \in [1, n]$$

Forward and backward segmentations are linearly weighted to verify the properties $C_1 = c_1$ and $C_n = c_n$.

3.2. Refinement by adding keyframes

If the computed segmentation is not satisfying, a *refinement* method consists in the user interaction on the computed segmentation. The user chooses a frame F_m (called *interframe*) among $F_2, \cdots, F_{n-1}$, where the computed contour is too far from the real object contour, and modifies it by pulling some sample points. F_m becomes a new keyframe and to compute the segmentation, rotoscopy is computed as before separately on intervals $[1, m]$ and $[m, n]$. Refinement can be used iteratively.

4. EXPERIMENTS

We first expose and analyze the differences and the influences of the two proposed contour tracking algorithms: the global and the local approach. Fig. 3 shows the difference between both approaches. The segmentation is more accurate with the local approach. Indeed, the parameter estimation is biased for the global approach by the non uniform motion of the tracks in the object.
In addition, contours are more directly relative to the initial contour. For example, on the sequence Bus seen on Fig. 6, the mean distance between current frame and reference frame is 5.3 frames for the global approach and 9.5 for the local one.

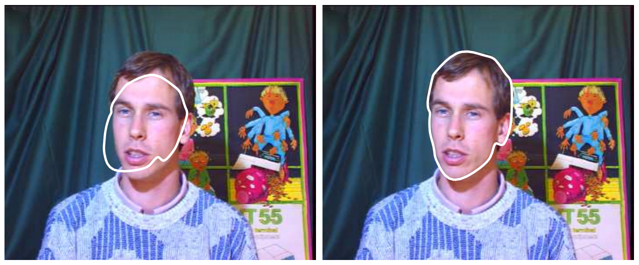
(a) Global approach (b) Local approach

Fig. 3. Comparison of the global and the local approach for contour tracking algorithm.

The proposed rotoscopy algorithm is tested on three sequences which have different features, one CIF (352x288 pixels) and two SD sequences (720x480 pixels). Fig. 4 shows the sequence *Erik* where the face is segmented with a good accuracy in spite of deformation and motion of Erik’s face. On the sequence *Skate* (Fig. 5), the contours of all frames are drawn. The object is well tracked in spite of the translation and the scaling of the object. The main result is about occlusion management which is a very difficult problem for object segmentation. Fig. 6 shows a sequence where a bus is largely occluded by a billboard. In spite of this occlusion and the contour deformation, the bus is precisely tracked. Finally, these figure shows that contour may be indifferently a polygon or a spline.

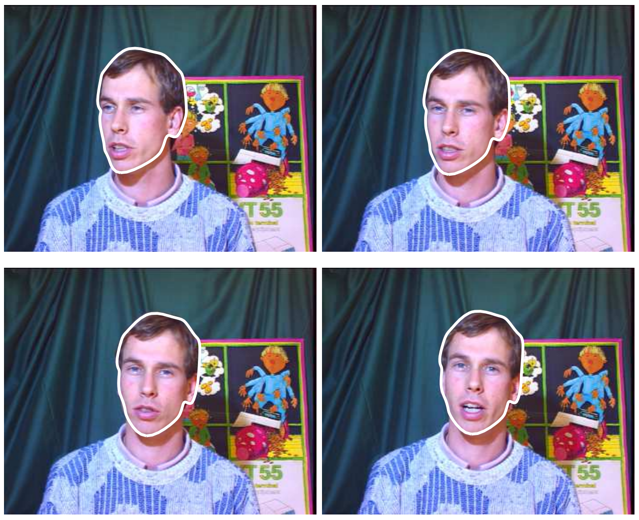

Fig. 4. Rotoscopy results on sequence *Erik* (CIF).

5. CONCLUSION AND PERSPECTIVES

We have proposed in this paper a contour tracking algorithm based on feature point tracking. The main particularity of this algorithm is that the object tracking is region-based instead of contour-based. Tracks are the key element of the system and local affine transformation increases the precision of the segmentation. This method is used

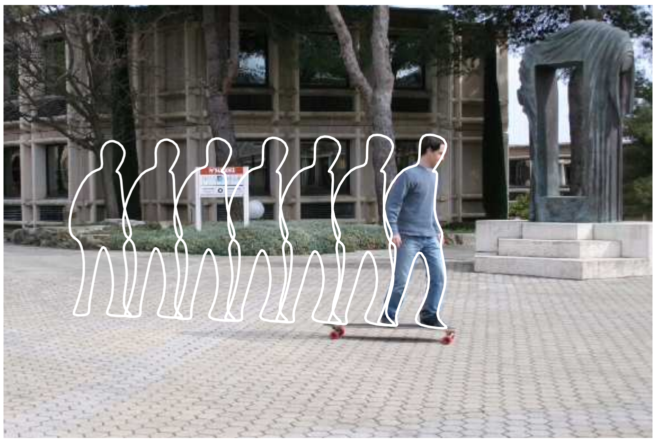

Fig. 5. Rotoscopy results on sequence *Skate* (SD).

to derive a rotoscopy algorithm by performing a tracking in forward and backward temporal direction.
According to the experimentation, this algorithm seems accurate and robust with occlusions. In addition, the proposed rotoscopy algorithm can be applied to high definition sequences (HD) in regard to the low computation times in standard definition sequences (SD).
As perspectives, we plan to test alternative robust parameter estimation algorithms [7, 8, 9].

6. REFERENCES

[1] G. Aubert, M. Barlaud, O. Faugeras, and S. Jehan-Besson, "Image segmentation using active contours: Calculus of variations or shape gradients?," *SIAM Applied Mathematics*, vol. 63, no. 6, pp. 2128–2154, 2003.

[2] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," in *Proc. Intern. Conf. on Computer Vision*, 1995, pp. 694–699.

[3] H. Luo and A. Eleftheriadis, "Spatial temporal active contour interpolation for semi-automatic video object generation," in *Proc. Intern. Conf. on Image Processing*, Kobe, Japan, october 1999, pp. 944–948.

[4] A. Agarwala, A. Hertzmann, D. H. Salesin, and S. M. Seitz, "Keyframe-based tracking for rotoscoping and animation," in *Proc. Intern. Conf. on Computer Graphics and Interactive Techniques*, Los Angeles, California, USA, august 2004.

[5] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. of The Fourth Alvey Vision Conference*, Manchester, UK, 1988, pp. 147–151.

[6] S. M. Smith and J. M. Brady, "Susan – a new approach to low level image processing," *Int. Journal of Computer Vision*, pp. 45–78, may 1997.

[7] P. J. Huber, *Robust Statistics*, John Wiley and Sons, 1981.

[8] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[9] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud, "Deterministic edge-preserving regularization in computed imaging," *IEEE Trans. Image Processing*, vol. 6, no. 2, pp. 298–311, 1997.

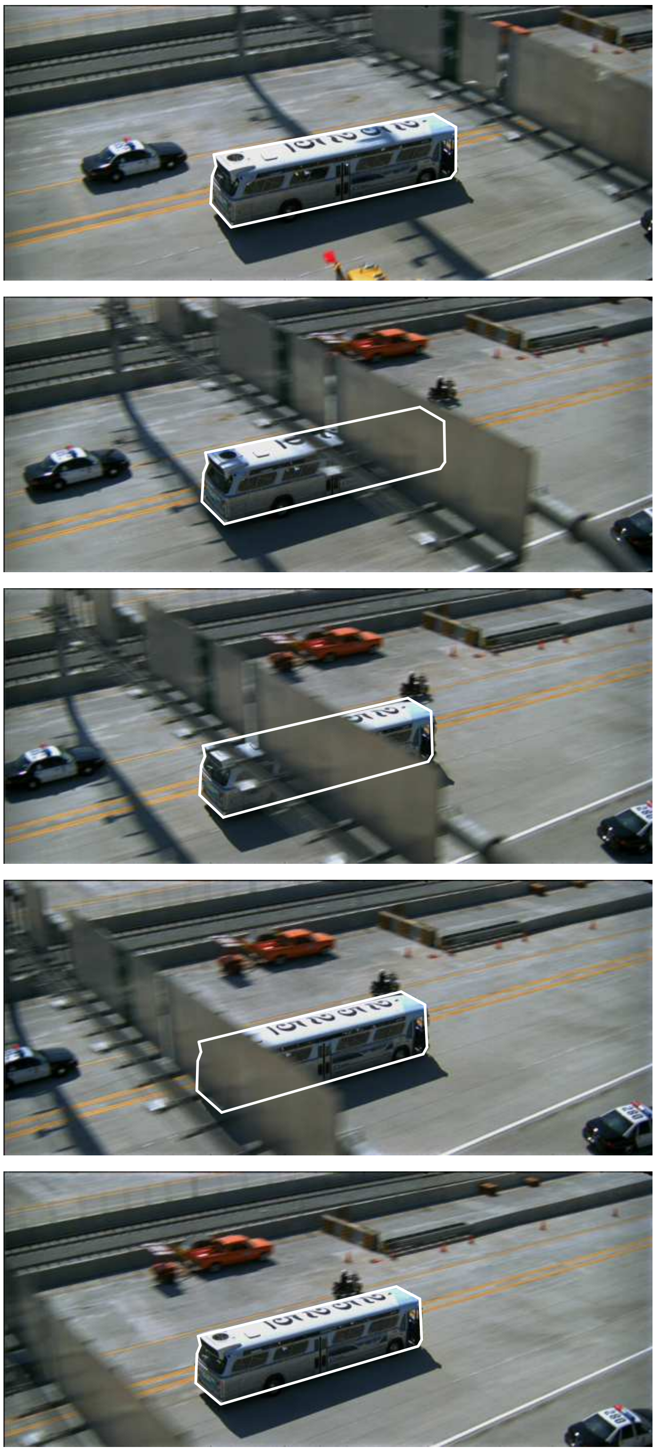

Fig. 6. Rotoscopy results on sequence *Bus* (SD).