A Fast Level-Set Method for Accurate Tracking of Articulated Objects with An Edge-Based Binary Speed Term

Cristina Darolti, Alfred Mertins and Ulrich G. Hofmann

Institute for Signal Processing, Univ. of Lübeck, Lübeck, 23538, Germany

Abstract. This paper presents a novel binary speed term for tracking objects with the help of active contours. The speed, which can be 0 or 1, is determined by local nonlinear filters, and not by the strength of the gradient as is common for active contours. The speed has been designed to match the nature of a recent fast level-set evolution algorithm. The resulting active contour method is used to track objects for which probability distributions of pixel intensities for the background and for the object cannot be reliably estimated.

1 Introduction

One of the necessary steps in making computers see is to teach them how to decide which object in the image is the one of interest. In many cases the object is completely defined by drawing a contour around the object area. Tracking involves keeping a lock on the correct contour as the object changes its position, shape and context in a video stream.

In this paper we present a method for tracking objects using active contours. An active contour is a curve which evolves from a start configuration towards the boundaries of an object in an image whilst its motion is governed by image properties. The curve can be represented parametrically, for example as a spline curve [1–3], or non-parametrically [4, 5]. Usually faster and more robust to clutter, parametric curves cannot easily describe articulated objects. This can however be simply achieved by non-parametric curves for which the representation of choice is the zero level set of a distance function [6].

The method presented here is intended for tracking articulated objects, thus active contours represented as level sets are the more suitable framework. The motion of the curve in this framework is governed by one of three forces. The first two are a force depending on the curvature of the boundary and a force depending on the strength of the image edge at the boundary [2, 4]. A third force expressing the belief that a region along the boundary belongs to the tracked object has recently been added [7–9]. The region force is proportional to the joint probability of pixels in the region, assuming the probability distributions in the object and background are known.

Active contours can be used in tracking by allowing the curve to move in each frame till it finds the boundary of the object in the respective frame. Like in the case of single images, tracking makes use of region and/or edge information [7]. The feature distributions of background and object are both used in [10, 11]. In [12], the vector field obtained by computing the optical flow between two images is used to track the contour around the moving object. It has been suggested [13] that a statistical distance measure between the probability distribution in the object region and a model distribution may be used to track the object, but since a distribution is independent of the objects area, the algorithm needs very special conditions for tracking to work.

We intend to track objects for which the probability distributions of the intensities of pixels does not have an analytical form and where an approximation by a mixture of normal distributions is not practicable when considering time constraints. The assumption that the distributions are normal is also problematic when the distributions of object and background strongly overlap. Should we add to these characteristics an inhomogeneous texture, it becomes obvious that it is very difficult to reliably describe the region information. Methods which can eventually describe the complicated statistics of such an image exist, but they are computationally much too expensive to qualify for use in tracking.

For this problem, we introduce a new reliable binary speed term into the active contour framework with the goal of tracking the boundaries of smooth objects with properties as described above. An additional requirement is that the object boundary is detected with high accuracy, i.e. the computed boundary should be less than two pixels away from the real boundary as picked by the human eye. The method is utilized to track hands during articulated motion. Specifically, we are interested in measuring hand movements during precision work, for example as performed during surgical operations, without using markers; the detected boundaries need to be accurate so that they can lead to precise measurements.

To set the frame for our work, a short overview of active contours evolved using the fast level-set method is given in Section 2. In Section 3, we extend the well-known active contours method with a novel binary speed term that was designed to match the nature of the fast level-set algorithm. The binary speed is based on local nonlinear filtering with the SUSAN edge detector and mean-shift filter, unlike the established image-gradient-based speeds. The results of applying the binary speed to real videos of different surgeons performing suturing are to be found in Section 4. Finally, we complete our paper with conclusions and outlook in Section 5.

2 Active Contours by Level Sets

A geodesic active contour is a curve which moves in time; at every time step, the curve is associated with an energy that depends on the curvature of the boundary and the image edge strength at the boundary as introduced in [2]. If a new metric is defined on the scalar field of image edge magnitudes, one where distances are defined to be short when the path passes through points with large magnitudes, the curve's energy is written as [4]:

$$E(C(p)) = \int_0^L g(\nabla I(C(p)))|C'(p)|dp, \tag{1}$$

where $C(p) = (x(p), y(p))$ is a two-dimensional curve, L is the length of the curve, $C'(p)dp$ is the arc length of the curve and $g(|\nabla I)|) : [0, +\infty \rightarrow \mathbb{R}^+$ is a strictly decreasing function. The curve is considered to be optimal when its energy is minimal, which is equivalent to finding a smooth curve of minimal length passing through the strongest edges. Using the energy's Lagrangian, an equation of motion is derived which describes the displacement of the curve in the direction of its Euclidean normal:

$$C_t = g(I)k\boldsymbol{n} - (\nabla g \cdot \boldsymbol{n})\boldsymbol{n}, \tag{2}$$

where C_t denotes the curve's time derivative, I the image, k the Euclidean curvature and $\boldsymbol{n}$ the normal vector, each of these variables being computed for every point (x, y) on the curve. A framework was thus established where image features could be used to evolve a smooth curve. One can take into consideration edge features [4, 2], region features [9] or both [7]. Osher and Sethian [6] have published the level-set method for numerical evolution of curves which move along their normal. In the level-set method, a d-dimensional curve, with $d \in \{2, 3\}$, can be embedded as the zero level set of a $(d + 1)$-dimensional function φ, knowing the initial curve C_0 :

$$C(x(p), y(p)) = \{(x, y)|\varphi(x, y, t) = 0\}), \text{with } \varphi(x, y, 0) = C_0. \tag{3}$$

Osher and Sethian have shown that the curvature of C, its normal and the equation of motion (2) can be expressed in terms of the function φ. Furthermore, the equation can be generalized to the case where the force acting on a curve point has a curvature-dependent component F_k and an image-dependent component F_I. If φ_t is the time derivative of the function φ, $\nabla\varphi$ is its gradient, and the curvature is expressed as the divergence of the gradient of φ, a general equation of motion $C_t = \alpha F_I \boldsymbol{n} + \beta F_k k \boldsymbol{n}$ may be written

$$\varphi_t = \alpha F_I |\nabla\varphi| - \beta F_k \left(div \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right) \right) |\nabla\varphi|, \tag{4}$$

with α and β being regularization parameters which control the influence of each term.

In order to accomplish tracking with active contours, once the boundary of the object is found in a frame, the corresponding curve is used to initialize the active contour in the next frame; the position of the boundary is then updated by the active contour type law of motion such as to best match the measurements in the new frame [3, 14, 15] and this is the choice we make within this study. An alternative is to learn a motion model for the moving object and to reposition the contour with its help in the new frame such that the measurements best confirm it [8, 1, 7].

The Fast Level-Set Implementation

Although very powerful, the numerical scheme for the level-set method is computationally intensive; a fair amount of research has been made to improve on its speed, for example in [16, 7]. The fast level-set method described in [17] is two orders of magnitude faster than its predecessors; its distinguishing feature is that the algorithm implementing the curve motion works entirely in the integer domain and the computation of boundary curvature is simplified to integer operations whilst the computation of the normal is omitted altogether.

The boundary of the object is considered to lie between pixels. Its position is specified by listing the object pixels bordering the curve in a list of interior pixels, called L_{in}, and by listing the background pixels bordering the curve in a list of exterior pixels, called L_{out}. The level-set function is piecewise constant, with values of -3 in the interior, -1 at pixels in the interior list L_{in}, 1 at pixels in the exterior list L_{out}, and 3 in the exterior. For every list pixel, the image-dependent speed F_I and curvature-dependent speed F_k from Eq.(3) are computed, but only the sign is retained. By choice, the curve's normals point outwards; logically, if at an exterior pixel the speed is negative the curve will be pushed inward, otherwise it will be left in place. To push the curve outward the curve is moved at interior pixels positive speed; the curve always advances at a speed of one pixel per iteration. To advance the curve outward at a pixel x from the list of exterior pixels, pixel x is deleted from L_{out} and the level set at x is set to the value for interior boundary pixels given by $\varphi(x) = -1$. If for any of the four connected neighbors y of pixel x, $\varphi(y) = 3$, y is added to the exterior list by setting $\varphi(x) = 1$; the procedure is called the switch procedure. When switching, it may also happen that one of the neighbors y now only has neighbors which belong to the interior of the curve, all having negative values in the level-set function; if this is true y is not an interior boundary pixel anymore, its corresponding level set value is set to -3 and it is deleted from L_{in}. The procedure is called the clean procedure, and together with the switch procedure it occurs in the pseudo code of the algorithm in Fig. 1.

In [17] the clean procedure is executed after the list of exterior pixels has been iterated through, but this may leave a neighborhood temporarily incoherent; although cleaning at every step necessitates four extra comparisons, we choose to execute this operation to keep the list coherent at every step. The symmetric process is used to advance the curve inward at a pixel x from the list of interior pixels. The motion stops when changes for none or a very small percentage of the list pixels have to be made .

Finally, an alternative to computing curvature, is to smooth the curve by convolving the level-set function with a Gaussian kernel converted to integer. It has been shown in scale space theory that this operation is equivalent to computing the Laplacian of an image; for an implicit function its Laplacian is equal to its curvature. The size of the Gaussian kernel controls the amount of smoothing. The position of the curve is thus updated by evolving it first according to the image-dependent speed for a number of iterations and subsequently evolving it according to the curvature-dependent speed for a number of iterations. It becomes obvious that this algorithm moves the curve exactly one or zero pixels per step. Thus, one needs not compute the magnitude of speeds F_I and F_k for the fast level-set implementation. The sole information needed here is binary in nature and is equivalent to the answer to the question: is the list pixel an edge pixel or not and/or does it belong to object region or not. The equation of motion can be rethought in terms of a binary speed, as discussed in the next section.

3 SUSAN Edge-Based Term

Curve evolution based on region and edge features has an additive form

$$C_t = \underbrace{\log\left(\frac{p_{in}(v(C(x)))}{p_{out}(v(C(x)))}\right)}_{\text{Region term}} + \underbrace{\alpha F_I \boldsymbol{n} + \beta F_k k \boldsymbol{n}}_{\text{Edge and smoothing term}} , \tag{5}$$

where the edge term has already been introduced in the previous section. The variables p_{in} and p_{out} denote the probability distribution of the feature vector v on the inside, respectively on the outside region of the object's boundaries; the new region term causes the curve to expand when $p_{in} > p_{out}$, otherwise causing the curve to shrink. In general, the region and edge terms are computed independent of each another. We observe that the region term needs to be computed solely at pixels located on the curve, which means that the region information for pixel x is gathered from its neighborhood only. The expansion of the function F_I reads as $F_I(I(C(x)))$, so the same is true for the edge term. As stated in the introduction, we intend to track objects for which discriminative distributions p_{in} and p_{out} cannot be estimated in a useful time. Since this is the case, we decide to use filters by which a pixel and its neighborhood can be analyzed to concomitantly describe region and edge properties. A simple binary speed is defined to categorize the result of filtering as follows

$$F_{sw} = \begin{cases} 1, & \text{if} \quad result(filter(x)) \quad \text{is of type “object“} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Correspondingly, the energy and the equation of motion are:

$$E(C) = \int_{\Omega} F_{sw} ds + \beta \int_{C} ds \qquad\qquad C_t = (F_{sw} + \beta k)\boldsymbol{n} \tag{7}$$

where Ω denotes the object's interior region, β is a regularization parameter which controls the strength of the smoothing and ds is the arc length. The binary speed F_{sw} is chosen to be binary in order to match the nature of the fast level-set algorithm.

Armed with this simple framework, we search for filters which can best characterize the boundaries of the sort of objects we wish to track. Because of problems in describing object regions, we choose an edge-based approach. Most edge-based active contours measure the edge as a function of the image gradient [4, 15, 7]. Thresholding gradient images to obtain binary edge images, like the one needed for the previously defined speed, bears well known problems, as will be discussed in the results section. We choose a nonlinear filter to analyze the intensities of neighboring pixels when deciding if a pixel is an edge pixel or not. More precisely, the similarity between a pixel and every other pixel in its neighborhood $N(x)$ is computed, and their sum

$$us(x) = \sum_{y \in N(x)} e^{-\left(\frac{I(x)-I(y)}{t}\right)^6} \tag{8}$$

yields a similarity score over the neighborhood, known as USAN [18] and denoted here by us; the parameter t specifies how large the difference between pixel intensities may be before they start to be dissimilar. The larger the us value, the more similar neighboring pixels are to the center of the filter mask. On the other hand, the us values will be smallest (Smallest USAN) when half of the pixels or less will be similar to the mask, a situation which occurs when the pixel lies on an edge or a corner. Multiple responses around the edges are eliminated by searching the minimum us value perpendicular to the edge direction; the direction vector $\boldsymbol{d}$ is obtained by computing the position of the center of gravity of the similarity responses within the mask. We may define the binary function in the simple motion equation (7) to be

$$F_{sw} = \begin{cases} 1, \quad \text{if} \quad us(x) > sim \quad \text{and} \quad \sum_{y \in N(x)} F_{sw}(y) = 1 \\ \qquad\quad \text{or} \quad us(x) \neq min\{us(y) | y \text{ is on } \boldsymbol{d}\ \} \\ \qquad\quad \text{or} \quad \sum_{y \in N(x)} F_{sw}(y) > no \\ 0, \quad \text{otherwise.} \end{cases} \tag{9}$$

The threshold sim denotes the smallest us value for which it can be stated with certitude that most pixels in the filter mask are similar to the center pixel; it can normally be set at 3/4 of the largest possible us value. The function is adjusted to fill in missing edges in the neighborhood $N(x)$ of a pixel and to stop zigzagged edges from causing a leak; to increase speed, this is done by simply setting an edge if the pixel has more than no neighbors which are edges. Also singleton edges are deleted if there are no other edge pixels in the neighborhood.

Looking at Function (9) one may notice that we have chosen to evolve the curve only outward. For most cases in object tracking it is possible to learn about the object and design an algorithm which finds a blotch in its interior. The boundary of this blotch is assumed to be the curve's initial position. In the next frame, the curve is evolved from its last known position to determine the blotch in the current frame. Tracking is achieved by expanding the blotch to the new correct boundary.

3.1 Mean Shift Local Filter for The binary Function

The USAN-based term defined in the previous section has the disadvantage of stopping at false edges if they form a smooth structure. Some may be eliminated by analyzing the probability distribution of pixel features in the neighborhood of an edge. For regions small enough, the probability distribution is well described by its mode since the number of samples is small enough. Let an image feature vector $\boldsymbol{x}$ be composed of spatial coordinates and the intensity value of a pixel. The mode is than determined by using the mean shift procedure [19, 14] on a three-dimensional variable. Consider the d-dimensional parametric Epanechnikov kernel density estimator K_E over n data points with bandwidth $h = (h_{spatial}, h_{intensity})$:

$$\boldsymbol{v} = \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \quad f(\boldsymbol{x}) = \tfrac{1}{nh^d} \textstyle\sum_{i=1}^{n} K_E(\boldsymbol{v}) \quad K_E(\boldsymbol{x}) = \begin{cases} c(1 - |\boldsymbol{x}|^2), \ |\boldsymbol{x}| \le 1 \\ \hfill 0, \text{ otherwise.} \end{cases} \tag{10}$$

The constant c ensures that the p.d.f. integrates to 1. The mode can then be found by looking for stationary points of the estimator function. The gradient of the estimator function is proven to be proportional to the mean shift vector

$$m_h(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g(\boldsymbol{v})}{\sum_{i=1}^{n} g(\boldsymbol{v})} \qquad\qquad g(\boldsymbol{x}) = \begin{cases} 1, & |\boldsymbol{x}| \le 1 \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

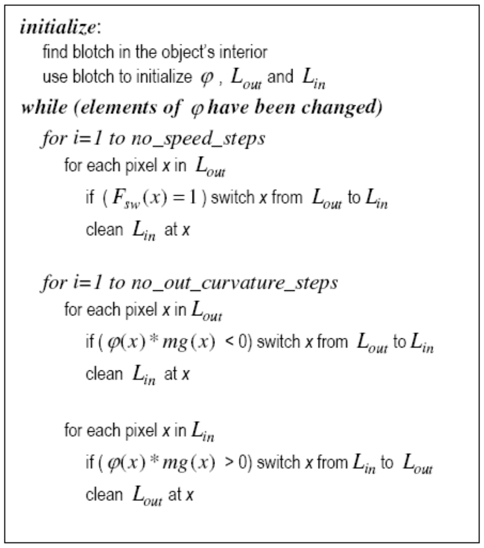
Two pixels that start the mean-shift procedure and converge to similar modes are considered to belong to the same probability distribution. In order to avoid a direct thresholding, and since a comparatively superior term for measuring the similarity between pixels has already been defined, the USAN score on the mean-shift filtered neighborhood of an edge pixel is computed. The speed for edge pixels with a similarity score larger than a minimum score, denoted as $msmin$, is reset to one. The new binary function is

$$F_{sw} = \begin{cases} 1, \quad \text{if} \quad us(x) > sim \quad \text{and} \quad \sum_{y \in N(x)} F_{sw}(y) = 1 \\ \qquad\quad \text{or} \quad us(x) \ne min\{us(y) | y \text{ is on } \boldsymbol{d}\ \} \\ \qquad\quad \text{or} \quad \sum_{y \in N(x)} F_{sw}(y) > no \\ \qquad\quad \text{or} \quad us(m_h(x)) > msmin \\ 0, \quad \text{otherwise.} \end{cases} \tag{12}$$

The algorithm implementing the above speed is summarized in Fig. 1.

4 Results

The binary speed based on SUSAN and mean-shift filtering in its fast level-set implementation is used to track hand motion. The main motivation is tracking the precise motions performed by surgeons during the suture procedure. The accurate contour is used to determine feature points, like the middle of the arm and wrist; these are useful in computing the position and trajectory of the hand with a stereo computer vision system. The suture motion can thus be analyzed or the surgical skill of the person can be measured.



```
initialize:
  find blotch in the object's interior
  use blotch to initialize $\varphi$ , $L_{out}$ and $L_{in}$
while (elements of $\varphi$ have been changed)
  for i=1 to no_speed_steps
    for each pixel x in $L_{out}$
      if ( $F_{sw}(x) = 1$ ) switch x from $L_{out}$ to $L_{in}$
      clean $L_{in}$ at x

  for i=1 to no_out_curvature_steps
    for each pixel x in $L_{out}$
      if ( $\varphi(x) * mg(x) < 0$) switch x from $L_{out}$ to $L_{in}$
      clean $L_{in}$ at x

    for each pixel x in $L_{in}$
      if ( $\varphi(x) * mg(x) > 0$) switch x from $L_{in}$ to $L_{out}$
      clean $L_{out}$ at x
```

Fig. 1. Pseudocode for the level set algorithm based on binary speed.

It has been mentioned, in Section 3, that a blotch in the objects interior is to be found first. In order to obtain such a blotch, an average background image is computed. The background image is subtracted from the current frame and the result is segmented with a double threshold. The binary image is processed with the fast level-set method with the initial curve at its last position in the previous frame to obtain two blotches. The size of the hands can only vary as restricted by cameras depth of field. It may be possible that the curve does not find the real boundary in a frame. Should the curve not stop in a maximum number of iterations, it is assumed that tracking in the current frame has failed and the blotches are re-initialized in the next frame after background subtraction and segmentation.

In the following, we observe some image properties of a typical frame from a recording of a suture operation; the frame in question is shown on the top left of Fig. 5. For this frame, Fig. 2 shows the histograms for the hand region and for the background region. The histograms were generated using the result of object/background segmentation, also shown in Fig. 5. It can be observed that the histograms overlap in the interval 25-60; pixels from shadowed parts of the hand and patches from the sleeves have many pixels with intensities in this interval, making this part of the image difficult to segment accurately. Because of the overlap, the result of segmenting the background subtracted image with an adaptive threshold, shown on the left in Fig. 4, is also unsatisfactory.

Visually, the hands appear to have strong edges, it should be thus possible to find the boundaries of the object using this information. We have tested three well known edge detectors: the Sobel, the Canny and the Laplacian-of-

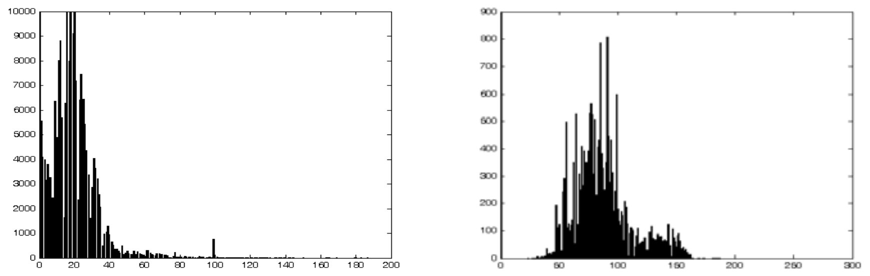


Fig. 2. Histogram for the background (left) and for the hands (right).

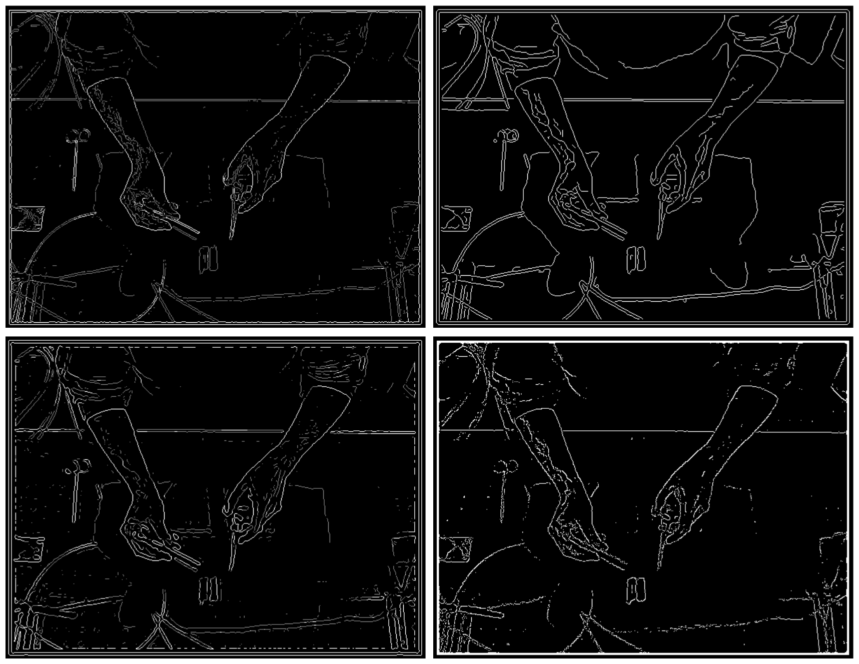


Fig. 3. Result of running edge detection on the top left frame from Fig. 5. Sobel edge detector with higher threshold – white edges – and lower threshold – gray edges – (top left). Canny edge detector (top right). Laplacian-of-Gaussian edge detector (bottom left). SUSAN edge detector(bottom right).

Gaussian methods, and their effect on the filtered frame can be observed in Fig. 3. The Sobel detector either does not find the boundaries of the upper shadowed hand parts - see the edges depicted in white - or introduces too many spurious edges on the hand surface - see the edges depicted in gray. The Canny edge detector reliably finds the correct edges, but introduces a few smooth ones on

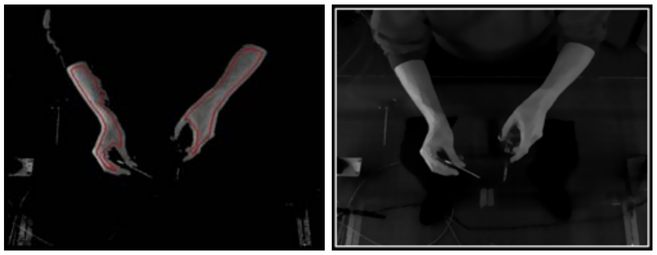


Fig. 4. The result of background subtraction and adaptive threshold segmentation (left) and mean shift filtering (right).

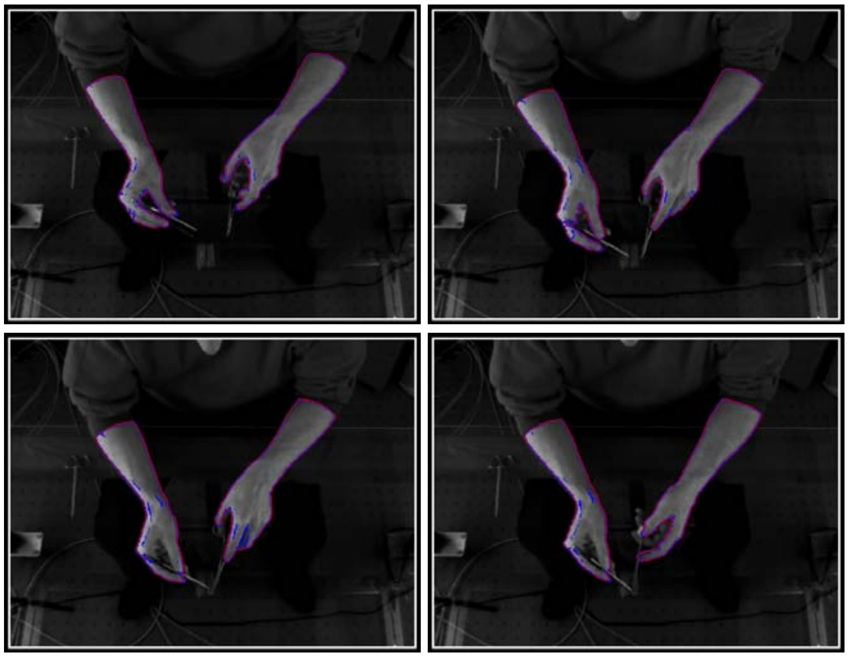


Fig. 5. Frames 1, 14, 23 and 40 from a recording showing a surgeon performing suture

the hand surface and these in turn are smooth enough to make the active contour stop; additionally, because of the edge thinning and gap-closing step, the Canny edge detector is slow compared to the SUSAN edge detector. The Laplacian-of-Gaussian is also comparatively slow and displays both the problems of the Canny detector and of the Sobel detector.

The SUSAN edge detector is computed with a 37 pixel circular mask and a value of 6 for the threshold t. It also introduces spurious edges, as it is obvious from Fig. 3(bottom right). To eliminate some of them a local mean-shift filtering is performed and analyzed with the USAN similarity measure on a 3×3

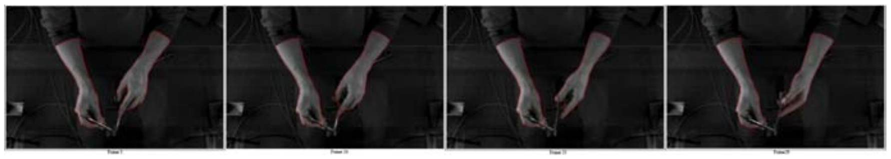


Fig. 6. Frames 4, 13, 18 and 20 from a recording showing a surgeon performing suture

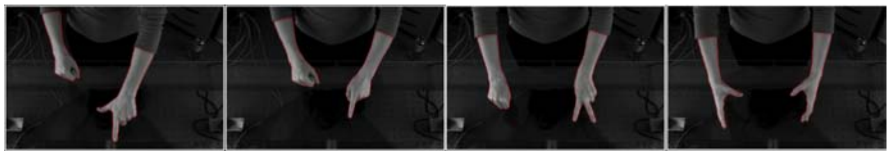


Fig. 7. Selection gestures (first two) and positioning gestures (last two) in a 3D medical visualization

neighborhood with the same threshold as the one used for the original image. To convey an impression of the effects of the filter, the result of filtering a frame with $(h_{spatial}, h_{intensity})$ =(5,10) is shown on the right in Fig. 4. Finally, Fig. 5 show in blue the edges which remain after removal of edge pixels with the help of the mean-shift operation, for four different frames of a video. In the same image, the position of the final contour is shown in red.

The hands of two different surgeons were tracked during suturing, as can be observed in Fig. 5 and 6. The algorithm implemented in C++, takes on average 0.18 seconds to process a frame on a desktop PC; the shortest processing time per frame was 0.1 seconds, the largest 0.2, but it is our belief that the implementation can be improved by parallelizing the code. The method was also employed to track hand motion when navigating a 3D medical visualization. Fig. 7 shows frames from a video where the user makes selection-by-pointing and positioning gestures.

5 Conclusions and Future Work

A novel binary speed based on SUSAN similarity scores between a pixel and its neighboring pixels and on probability density mode detection by the mean shift procedure has been presented. The speed is designed to match the nature of the fast level-set implementation. The hands of surgeons performing suture have been tracked at an average of 0.18 seconds per frame. Some pieces of the tracked boundaries are not accurate according to our definition. Also, in the frames with no boundary found, the curve leaked through a very local misdetection of edges. In the future, more of the information from neighboring pixels will be integrated in the binary speed. Finally, we propose to use shape templates to cope with large pieces of misdetected boundary.

References

1. Isard, M., Blake, A.: Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In: ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume I, London, UK, Springer-Verlag (1998) 893–908
2. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision **1(4)** (January, 1988) 321–331
3. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models – their training and application. Comput. Vis. Image Underst. **61**(1) (1995) 38–59
4. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. Int. J. Comput. Vision **22**(1) (1997) 61–79
5. Paragios, N., Deriche, R.: Geodesic active contours for supervised texture segmentation. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99). Volume 2., Los Alamitos, CA, USA, IEEE Computer Society (1999) 2422
6. Osher, S., Sethian, J.A.: Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. Journal of Computational Physics **79** (1988) 12–49
7. Paragios, N., Deriche, R.: Geodesic active contours and level sets for the detection and tracking of moving objects. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(3) (2000) 266–280
8. Ecabert, Thiran, O.: Variational image segmentation by unifying region and boundary information. In: 16th International Conference on Pattern Recognition. (2002)
9. Chan, T., Vese, L.: Active contours without edges. IEEE Trans. Image Processing **10** (2001) 266–277
10. Mansouri, A.R.: Region tracking via level set pdes without motion computation. IEEE Trans. Pattern Anal. Machine Intell. **24(7)** (2002) 947–961
11. Yilmaz, A., Li, X., Shah, M.: Contour-based object tracking with occlusion handling in video acquired using mobile cameras. IEEE Trans. Pattern Anal. Machine Intell. **26(11)** (2004) 1531–1536
12. Roy, T., Debreuve, É., Barlaud, M., Aubert, G.: Segmentation of a vector field: dominant parameter and shape optimization. Journal of Mathematical Imaging and Vision **24**(2) (2006) 259–276
13. Freedman, D., Zhang, T.: Active contours for tracking distributions. Image Processing, IEEE Transactions on **13(4)** (April 2004) 518– 526
14. Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using mean shift. In: IEEE Conf. Computer Vision and Pattern Recognition. (2000)
15. Shi, Y., Karl, W.C.: Real-time tracking using level sets. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), Washington, DC, USA, IEEE Computer Society (2005) 34–41
16. Sethian, J.: Level Set Methods and Fast Marching Methods. Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science. Cambridge University Press (1999)
17. Shi, Y., Karl, W.: A fast level set method without solving pdes. In: IEEE International Conference on Acoustics, Speech, and Signal Processing. (2005)
18. Smith, S.M., Brady, J.M.: Susan–a new approach to low level image processing. Int. J. Comput. Vision **23**(1) (1997) 45–78
19. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Analysis Machine Intell., **24** (2002) 603–619