Published in final edited form as:
IEEE Int Conf Robot Autom. 2015 May 26; 2015: 4710–4716. doi:10.1109/ICRA.2015.7139853.

Automatic Initialization and Dynamic Tracking of Surgical Suture Threads

Russell C. Jackson, **Rick Yuan**, **Der-Lin Chow**, **Wyatt Newman**, and **M. Cenk Çavuşoğlu**
Department of Electrical Engineering and Computer Science (EECS) at Case Western Reserve University in Cleveland, OH, USA

Abstract

In order to realize many of the potential benefits associated with robotically assisted minimally invasive surgery, the robot must be more than a remote controlled device. Currently using a surgical robot can be challenging, fatiguing, and time consuming. Teaching the robot to actively assist surgical tasks, such as suturing, has the potential to vastly improve both patient outlook and the surgeon's efficiency. One obstacle to completing surgical sutures autonomously is the difficulty in tracking surgical suture threads. This paper proposes an algorithm which uses a Non-Uniform Rational B-Spline (NURBS) curve to model a suture thread. The NURBS model is initialized from a single selected point located on the thread. The NURBS curve is optimized by minimizing the image match energy between the projected stereo NURBS image and the segmented thread image. The algorithm is able to accurately track a suture thread as it translates, deforms, and changes length in real-time.

I. Introduction

Robotic surgical systems used in Minimally Invasive Surgery (MIS) present an opportunity to pair human surgical expertise with the precision and repeatability of robots. Surgical suturing is one task that can benefit from the synergy of surgeons and surgical robots. Currently, the robot is controlled through teleoperation. Requiring the surgeon's complete control of the robotic system can make certain tasks during MIS, such as suturing, challenging, fatiguing, and time consuming. This is largely due to the lack of haptic feedfack, a limited robotic workspace, and a narrow field of view from the laparoscopic camera system. Autonomous robotic manipulation and control of the suture needle and thread would allow the robot to complete the suture while relying on minimal guidance from the surgeon. Once the surgeon specifies key aspects of the suture, the rest of the task can be completed by the robot. The surgeon then plans the next step of the procedure while supervising the quick and precise autonomous completion of the suture.

In order to complete autonomous tasks during MIS, it is important to track task critical elements (e.g. suture needle, thread, and tissue for autonomous suturing). This paper focuses on the thread tracking problem. In the proposed method, the suture thread is initialized as a 3-dimensional NURBS curve. The NURBS suture thread model is projected into a stereo

The corresponding author M. Cenk Çavuşoğlu can be reached at cavusoglu@case.edu.

image pair and the projected models are optimized using an energy based image improvement algorithm. The updated models are then recombined into a 3-dimensional NURBS update. This approach allows the suture thread to be tracked in real-time as it is deformed and the method is robust during changes in thread length, self intersection, and knot formation.

The outline of the paper is as follows, section II summarizes previous related research in the literature. Section III discusses how the suture thread is segmented. Section IV describes how the shape of the suture thread is modeled. Section V explains how the NURBS model is seeded from a single point. Section VI discusses how the suture thread model is updated in real-time. Section VII demonstrates the capabilities of the suture thread initialization and tracking algorithm. Finally, section VIII presents the discussion and conclusion.

II. Literature Review

The problem of autonomous surgical suturing encompasses many facets of robot control and perception. An autonomous surgical suture requires tracking the thread, tissue, and needle in order to plan and execute the robot motion. This must be completed using techniques that are robust and adaptable to the dynamic surgical environment.

Iyer et al. demonstrates that it is possible, using tissue markers, to complete an automated circular needle drive [1]. This drive is completed while tracking the needle. In this implementation, the needle was not attached to any thread. This contrasts to actual suture needle kits where the thread and the needle form a single unit as shown in Fig 1. Once the needle is driven, the suture thread must be pulled through and tied off. Since suture thread can be 70 cm long, a single suture kit can be used for multiple sutures.

Detecting and manipulating the suture thread is an important component of completing a suture autonomously. While it is possible to circumvent suture thread tracking using specialized hardware [2], it may not always be practical to use. Previous works have demonstrated the feasibility of detecting a suture thread as well as similar objects. Javdani et al. [3] demonstrated that it is possible to model suture threads as Dynamic One-dimensional Objects (DOO). Each suture thread is then modeled as a collection of connected bones. The incident and twisting angles between the bones are modeled as an internal deformation energy of the thread. By minimizing the total internal energy coupled with the energy of the image matching, the suture thread is identified. In this example, the suture thread is segmented from the image using Canny edge detection [3]. This approach requires knowledge of the suture thread material model. An alternative approach by Schulman et al. [4] used a point cloud to help identify various deformable objects (including rope). Similar to [3], actively deforming a physical model increases the internal energy so that the energy of the image matching can be minimized. The final goal is to minimize the sum of the internal energy with the image match energy. Currently, MIS robots do not employ the same RGB plus depth imaging device that is required to generate a point cloud. Consequently, it is more practical to rely exclusively on the stereo imaging systems that are deployed in existing robotic surgical systems. Padoy et al. used multiple approaches to detect curvilinear structures, [5], [6]. One of which, [5], used NURBS curves to describe the shape of the detected suture. The control points of the curve were optimally positioned by using a Markov random field model that was iterated using a fastPD optimization algorithm [7]. In addition to using Markov random fields, active contours (Snakes) can be used for image based curve optimizations [8]. This includes optimizing closed NURBS curves with active contour modeling [9].

Previous algorithms for suture thread detection assumed that the end points were held by tracked manipulators during initialization [5]. While this allows for tracking algorithms to be tested, it neglects the problems that might be associated with tracking the suture end points. In an actual surgical environment, the suture thread may only have one end grasped. The thread may not even be grasped at all. Additionally, the end point constraint is not directly addressed. Due to the nature of the image energy functions in previous works [3],[5] mismatches are not penalized if the image of the suture thread model is a subset of the actual suture thread image. This allows for the suture thread model to 'shrink' with no apparent penalty. To mitigate this, a penalty was introduced where the cost would increase as the length of the model changed [5]. This can cause problems if a portion of the suture thread is occluded and is slowly exposed causing the thread to lengthen.

This paper presents an algorithm that can be used to initialize and then accurately track the suture thread from an initial seed point. The tracking algorithm is capable of tracking the suture thread as it moves, deforms, changes length, and tightens knots.

III. Suture Detection

Reliance on automatic suture thread detection requires top level knowledge of the surgical procedure. There may be multiple suture threads in the image. Some sutures might even have already been tied off. Detecting the suture thread of interest requires a directed pruning of irrelevant sutures. Supervisory selection will maximize the performance of the robot while keeping the surgeon in control of the procedure. Identification of the target suture thread can be quickly and easily performed by the surgeon provided that there exists an adequate user interface that enables the surgeon to communicate his/her intentions.

There are many methods that a surgeon can utilize to input the point of the suture thread. The surgeon can enter the point by area selection, three-dimensional projection, gripper based initialization, or even using an external laser pointer. The method used can be selected to complement the surgeon's existing user interface. In this implementation, a 3-dimensional space mouse was used to input the seed point. The seed point, when coupled with the segmented image, allows the initial thread position to be identified.

A. Suture Thread Segmentation

In the proposed method, the suture thread is segmented from the image using a thin feature enhancement algorithm developed by Frangi et al. [10]. The algorithm is modified to include directionality information that was explored by Steger et al. [11]. During thin feature enhancement, a gray scale image representation is convolved with a two dimensional Gaussian distribution second derivative. The variance of the Gaussian function corresponds to the scale space size of the image filter. The result is a scale space Hessian matrix of the original gray image. That is to say each image pixel is now a set of 4 pixels that comprise a 2 × 2 symmetric matrix. This 2 × 2 real symmetric matrix has real eigenvalues ($|\lambda_1| < |\lambda_2|$) and orthogonal eigenvectors $v_1 \perp v_2$. The magnitude and ratio of these eigenvalues are used to generate the output image magnitude. A large magnitude eigenvalue indicates that the local image region aligns well with the second derivative of a Gaussian function in the corresponding eigenvector direction. If one eigenvalue is large and the other is small, the local image region looks like a ridge (or a section of suture thread). The output vector for each pixel location $V_r(i, j)$ can be computed directly from the eigenvalues of the corresponding Hessian matrix.

$$\begin{aligned} \| V_r \| &= e^{-\frac{R_B^2}{2\beta^2}}\left(1-e^{-\frac{S^2}{2c^2}}\right) \\ V_r &= \| V_r \| \, v_1 \end{aligned} \quad (1)$$

Here $R_B = \lambda_1/\lambda_2$, $R_B{=}\lambda_1/\lambda_2, \quad S{=}\sqrt{\lambda_1^2{+}\lambda_2^2}$, and the parameters β and c are user defined parameters. The normalized eigenvector corresponding to the smaller eigenvalue is considered to be the local thread direction (v_1). Since the eigenvector can be arbitrarily scaled, the angle (θ) of the thread direction is wrapped into the domain $\theta \in [0, \pi)$. The result is that each pixel from the gray scale image is now a vector in $V_r(i, j) \in \mathbb{R}^2$. As part of the post processing, the image is smoothed using a low pass gaussian kernel, G. Due to the natural wrapping of the angle domain, the gaussian blur is computed as follows.

Algorithm 1 This algorithm blurs local pixel vectors together in order to maximize the output vector magnitude

```
1:  procedure GAUSSIAN ANGLE BLUR($V_r, V_o, G$)
2:      for $\forall V_r$ do
3:          $\boldsymbol{v}_t \leftarrow G(0,0)V_r(i,j)$
4:          for $(l,s) \in G \setminus (0,0)$ do
5:              $d \leftarrow \boldsymbol{v}_t \cdot V_r(i+l, j+s)$
6:              if $d > 0$ then
7:                  $\boldsymbol{v}_t \leftarrow \boldsymbol{v}_t + G(l,s)V_r(i+l,j+s)$
8:              else
9:                  $\boldsymbol{v}_t \leftarrow \boldsymbol{v}_t - G(l,s)V_r(i+l,j+s)$
10:             end if
11:         end for
12:         $V_o(i,j) = \boldsymbol{v}_t$
13:     end for
14: end procedure
```

An example comparing an unsegmented image to a segmented image is shown in Fig. 2. The purple suture thread is highlighted from the background. The segmented image is falsely colored to indicate direction.

IV. Suture Thread NURBS Model

In the proposed tracking approach, the thread is modeled as a 3-dimensional NURBS curve. The suture thread is defined by a set of n+1 control points $C = [c_0, \ldots, c_n]^T$. The order of the curve, p, indicates the polynomial degree that is used to smooth together the control points. A NURBS curve also requires a set of $m + 1$ knots where $m = n + p + 1$. In this case, the set of knots, U , are distributed in the range [0, 1]. The end knots have a multiplicity of $p + 1$. This ensures that the end points of the curve fall on the points c_0 and c_n, respectively. NURBS curves have been used previously in vision research because they generate smooth curves using a relatively low dimensional space [12], [5]. This naturally allows for the NURBS curve to have a reduced internal energy compared to point wise curve definitions. The NURBS curve is parametrically defined as in [13]

$$\boldsymbol{f}(u) = \frac{\Sigma_{i=0}^{n} N_{i,p}(u)\, w_i \boldsymbol{c}_i}{\Sigma_{i=0}^{n} N_{i,p}(u)\, w_i}, \quad (2)$$

where $u \in [0, 1]$ is the parameter used to generate the points on the curve. The term $N_{i,p}(u)$ is a B-spline basis function of order p. In this study, the weight w_i is defined to be 1 for all indices. This allows the NURBS curve to be easily projected from 3-dimensional space to 2-dimensional images. Additionally, unity weights reduce the space of optimized parameters, reducing the overall optimization time. The basis functions are defined over the knot vector of the NURBS curve as defined above. After the curve point set $P(u)$ is found, it is projected into the stereo images resulting in the point sets $P_l(u)$ and $P_r(u)$.

V. Initializing the Suture Thread

The curve initialization function generates a thread model including loops and dead ends when given an arbitrary point along the thread. The implementation of this algorithm is built on previous work by Kaul et al. [14]. The user selects the initial point using a projected 3-dimensional location with a spacemouse. The point grows outwards from the initial point according to a predefined cost function. The initially selected point has a cost of 0. This is similar to Dijkstra's tree growth algorithm. The cost of adjacent pixels to the current is calculated using the magnitude of the segmented image pixels.

$$L(i+i_x, j+j_y) = L(i,j) + \frac{\|\,(i_x, j_y)\,\|}{\|\,V_o(i+i_x, j+j_y)\,\|^{\rho}} \quad (3)$$

Here, the i, j represent the current image location while $i_x, j_y \in [0, 1, -1]$. The parameter ρ is used to improve the delineation between the thread and noisy background. This represents the 8-connected neighborhood around a pixel. The lowest net cost for each pixel is stored. The lowest cost pixel is then used to compute its neighbor's cost. The growth of this path is shown in Fig. 3a. When the cost of a pixel is calculated, it is added to a set of tracked pixels which grows outwards at each iteration. After the front of the pixel growth reaches a certain distance away from the origin point, preset as the step size, a thread point is marked at that location. A corresponding thread point is found in the other stereo image, and the stereo point is added to the current point list. When a new stereo point is found, the list that it belongs to must be identified. The new point is compared to the current point lists. If the point matches one of the lists, then it is appended to that list. Otherwise, the point seeds a new list. The end result is that there is a list of thread segments representing every continuous non-branched segment of thread. These thread segments are then combined into the final thread model. Endpoints of the thread segments that are close to each other and have similar angles are connected. The merging of matching segments is shown in Fig. 3b.

The resulting list of thread points is then used to generate a curve. The NURBS curve is calculated using a least square approximation of the generated 3-dimensional point list. The result is a list of NURBS control points $C = [c_0, \ldots, c_n]^T$. Once the thread has been initialized, the NURBS model is updated using an iterative tracking algorithm.

VI. NURBS Curve Iteration

After completing the suture thread initialization, the curve is defined as a set of control points, $C = [c_0, \ldots, c_n]^T$. Previous NURBS improvement techniques include using a Markov random field to optimize the set of control points [5]. This approach has the disadvantage of not inherently tracking the ends of suture thread as they change. It randomly updates the control points and checks to see if this improves the image match. Optimizing a closed NURBS curve also circumvents the need to deliberately track end points [9]. Optimizing the control points using the projected images of the curve allow the algorithm to update itself directly to the images while tracking the ends of the suture thread.

The proposed algorithm tracks the deforming suture thread as it moves in the surgical environment by iteratively updating the control points of the suture thread model. This includes removing and inserting control points as needed. The goal is to minimize the image energy while preventing the internal energy from being too high. Evenly spacing the NURBS control points helps to minimize the internal energy.

A. Image Energy

The energy of the NURBS curve image is based on both the image magnitude and image alignment. Not only should the curve image match the peak of the segmented image, they should share the same tangential direction as well. The image energy equation actually represent a pair of image energies, one for the left and one for the right image. Without loss of generality, the l subscript will stand for the pair l and r unless otherwise noted.

$$E_l \quad = \quad -b\int_{u=0}^{u=1} \parallel V_{ol}(u) \parallel \frac{(t_l(u) \cdot V_{ol}(u))^2}{\parallel V_{ol}(u) \parallel^2} \mathrm{d}u. \quad (4)$$

Here $t_l(u)$ is the unit tangent to the curve at point in image space while $V_{ol}(u) = V_o(p_l(u)) = V_{ol}(i, j)$ is the segmented image output vector and direction at the curve image point $p_l(u)$.

The spatial gradient of the image energy is used to generate forces on the curve image points. This gradient is approximated as,

$$\boldsymbol{f}_l(u) \quad = \quad -b\nabla \parallel V_{ol} \parallel \frac{(\boldsymbol{t}_l(u) \cdot V_{ol}(u))^2}{\parallel V_{ol}(u) \parallel^2} \quad (5)$$

It is assumed that the spatial derivative of $\frac{(\boldsymbol{t}_l(u) \cdot V_{ol}(u))^2}{\|V_{ol}(u)\|^2}$ is small. That is to say, the alignment between the segmentation direction and the curve tangent does not change quickly. This gradient force is then projected into the image curve normal $n_l(u)$. The point offset is then given by

$$\Delta \boldsymbol{p}_l(u) = (\boldsymbol{f}_l(u) \cdot \boldsymbol{n}_l(u))\, \boldsymbol{n}_l(u). \quad (6)$$

This prevents the points from sliding along the segmented curve. If the local energy gradient is small, then the local image region is actively searched for a potential point. This increases the image basis of the NURBS optimization algorithm. Optimizing the end points requires that they are given their own energy cost term.

B. End Point Energy

A difficulty in tracking the thread arises due to the special case of the thread endpoints. Other works have used fixed end points or penalized changing the length of the suture thread in order to assist in tracking the endpoints [5] [3]. In this work, the end of the thread is tracked using the local segmented image information as it aligns to the thread model.

The end points have their own calculated energy term.

$$\begin{aligned} E_{l,end} &= \left(\frac{(\boldsymbol{t}_l(0)\cdot V_{ol}(0))^2}{\|V_{ol}(0)\|} - \| V_{cl} \|\right)^2 \\ &+ \left(\frac{(\boldsymbol{t}_l(1)\cdot V_{ol}(1))^2}{\|V_{ol}(1)\|} - \| V_{cl} \|\right)^2 \end{aligned} \quad (7)$$

The parameter V_{cl} is the cutoff magnitude of the image. The cutoff magnitude is the average of the foreground mean magnitude and the background mean magnitude of the segmented image. This attracts the curve ends to the thread end. The end point forces acts on both ends

$$\boldsymbol{f}_{l,end}(0) = -2\nabla \| V_{ol}(0) \| \left(\frac{(\boldsymbol{t}_l(0)\cdot V_{ol}(0))^2}{\| V_{ol}(0) \|} - \| V_{cl} \|\right) \quad (8)$$

$$\boldsymbol{f}_{l,end}(1) = 2\nabla \| V_{ol}(1) \| \left(\frac{(\boldsymbol{t}_l(1)\cdot V_{ol}(0))^2}{\| V_{ol}(1) \|} - \| V_{cl} \|\right) \quad (9)$$

As in equation 5, the gradient of the segmentation NURBS alignment is neglected. The positive sign at $u = 1$ is due to the fact that the tangent of the nurbs curve points from $u = 0$ to $u = 1$. Consequently it must be flipped at the end point where $u = 1$. This force is projected to be along the end point tangent direction. The point offset is then given by

$$\Delta \boldsymbol{p}_{l,end}(0) = (\boldsymbol{f}_{l,end}(0) \cdot \boldsymbol{t}(0))\, \boldsymbol{t}_l(0). \quad (10)$$

The equation is similar for when $u = 1$. The forces are formulated such that the internal points are updated only in directions normal to the curve while the end points are additionally updated in directions that are tangential to the curve.

C. Pointwise Update

The NURBS curve is defined to be a linear combination of the control points. Since it is impractical to compute the curve for every $u \in [0, 1]$, only a finite number of points are actually computed. If the curve is generated for a list of parameters values $u_j \in u$ where $0 \leq j$

$\leq k$, then the basis function $N_{i,p}(u)$ can be precomputed and, as such, a matrix A(u) $\in \mathbb{R}^{(k+1)\times(n+1)}$ can be constructed

$$A(\boldsymbol{u}) = \begin{bmatrix} N_{0,p}(u_0) & N_{1,p}(u_0) & \dots & N_{n,p}(u_0) \\ N_{0,p}(u_1) & N_{1,p}(u_1) & \dots & N_{n,p}(u_1) \\ \vdots & \vdots & \ddots & \vdots \\ N_{0,p}(u_k) & N_{1,p}(u_k) & \dots & N_{n,p}(u_k) \end{bmatrix}. \quad (11)$$

The denominator from (2) is not present because it always sums to 1. The vector $u = [u_0, \dots, u_k]$ is defined such that the points are evenly spaced throughout the entire NURBS curve. The NURBS curve is now formulated as a linear combination of the control points

$$P = A(\boldsymbol{u})C. \quad (12)$$

Here $P = [p_0, \dots, p_k]^T$ is the set of points on the NURBS curve, C is the set of control points as defined previously, and $A(u)$ is defined in (11). By retaining a precomputed copy of the matrix $A(u)$, the NURBS curve can be quickly recomputed if the control points are updated without changing the parameter vector (u). After the curve finite point set P is found, it is projected into the stereo images resulting in the point sets P_l and P_r. The precomputed discrete points of the NURBS curve greatly simplify the force computation on the set of NURBS curve points. This results in an offset that is applied to each set of projected NURBS points Δp_{il}. Once the left and right image updates are found, they must be deprojected into 3-dimensional space.

D. 3-Dimensional Deprojection

Now that the update vector has been found for the set of image points Δp_{il} and Δp_{ir}, they must be mapped from stereo image space into 3-dimensional space. This can be accomplished by analyzing the projection of the orthonormal curve basis $t, n, b,$ (tangent,normal,binormal). Assuming that the 3-dimensional optimization vector is of the form $\Delta p_i = \alpha_i t_i + \beta_i n_i + \gamma_i b_i$ where α_i, β_i, and γ_i are update gains. Then, provided that $\|p_i\|$ is small, the projected vector Δp_{il} approximates the sum $\alpha_i t_{il} + \beta_i n_{il} + \gamma_i b_{il}$. This can be reduced to the following over determined matrix equation.

$$\begin{bmatrix} x_{til} & x_{nil} & x_{bil} \\ y_{til} & y_{nil} & y_{bil} \\ x_{tir} & x_{nir} & x_{bir} \\ y_{tir} & y_{nir} & y_{bir} \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{bmatrix} = \begin{bmatrix} x_{\delta il} \\ y_{\delta il} \\ x_{\delta ir} \\ y_{\delta ir} \end{bmatrix} \quad (13)$$

The least squares solution of (13) gives the values of α_i,β_i, and γ_i. Since the offset of each curve point Δp is found in 3-dimensional space, the update is constrained to the epipolar lines of the stereo camera system [15]. These values can then be used to generate the three dimensional offset vector $\Delta P = \Delta p_0, \dots, \Delta p_n{}^T$. This offset matrix ΔP in the curve point space is mapped into the control point space using the matrix $A(u)$ as defined in (11).

$$\Delta C = A^T(\boldsymbol{u})\,\Delta P \quad (14)$$

The end points are updated directly using the computed offset. This is done because the end points are defined to be the end control points (i.e. $\Delta c_0 = \Delta p_0$). The matrix D ∈ $\mathbb{R}^{k+1\times 3}$ is a denominator matrix that is used as a normalization matrix to normalize the force offsets on the matrix ΔC.

$$D = A^T(\boldsymbol{u})\begin{bmatrix} 1 & 1 & 1 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 1 \end{bmatrix} \quad (15)$$

This serves to normalize the effects of the curve point forces. The point set at the next time step is then simply the previous points plus the scaled offset point set.

$$C_{t+1} = \Delta C \oslash D + C_t \quad (16)$$

The operator $\oslash$ indicates that matrix ΔC is element wise divided by denominator matrix D

1) Control Point Post Processing: Once the control points have been updated, they are added and removed based on inter control point distance. Points are pruned if they are too close together. Conversely points are inserted where there is a large gap between neighboring points. The points are inserted such that the curve does not change shape. The goal is to keep the set of interpoint distances $||c_k - c_{k-1}||$ and $||c_k - c_{k+1}||$ within a certain range. This mitigates the possibility that control points are bunched up while at the same time ensures that there won't be any gaps in the thread model. While it is true that straight segments of the curve do not need to be modeled with as many points, the presence of the points enables the model to respond more quickly to a local changes in the thread shape.

Fig. 4 shows the composite result of fitting a NURBS curve to a suture thread. The quality of fit allows for a robotic gripper to grab the thread (See attached video).

VII. Experimental Validation

The goal of the experimental validation is to evaluate the performance of the suture thread tracking algorithm. The following aspects are validated: initialization, motion tracking, and distortion tracking. In order to test the algorithm, a pair of calibrated stereo cameras with a resolution of 640 × 480 pixels were mounted above a motorized X-Y linear stage that allows the thread to be tracked as it moves. The experimental stage is shown in Fig. 5. The orange and red marbled paper was used in an attempt to mimic colors and patterns that might be found in a surgical setting. The computer completing the image processing contains an Intel Core 2 Quad processor running at 3GHz with 8 GB of installed RAM. The segmentation algorithm was implemented using CUDA and ran on a Nvidia GTX 650 Ti with 2 GB of GDDR5 RAM. During the experiments, the segmentation algorithm as well as the tracking algorithm ran in parallel threads. The segmentation algorithm runs with a loop speed of (15

Hz). The suture tracking thread runs with a loop speed of (20 Hz). Since the suture thread tracking and the segmentation loop operate in parallel, the thread model lags behind the segmentation model in the video display thread.

A. Quantative Accuracy

In order to detect the accuracy of the thread tracking algorithm, the proposed algorithm was used to estimate the length of three pre-cut suture threads. This was done in order to avoid marking the thread during imaging. Marking the thread would have interfered with the segmentation. The lengths of the suture threads were 107 mm, 175 mm, and 200 mm. Each precut segment was positioned on the work space, initialized, and tracked. This includes tracking the thread as it moves and deforms. One instance of each thread was measured in this way. The thread was twisted and kinked in the different images so that it did not lie flat on the workspace. Additionally, one side of the suture thread was affixed to an immobile section of the background paper. This was done to ensure that the thread would deform significantly during tracking. This is illustrated in Fig 5. The table moved its attached thread endpoint in a figure 8 pattern with a maximum speed of 11 mm/s. The length of the thread model during deformation was logged and the mean and variance of the logged length are summarized in Table I. The term ℓ represents the actual length of the suture thread, while $\sim \ell$ is the average estimated length of the suture thread. The standard deviation of the measurements (σ) and the percent error (%) are also listed.

As the table indicates, the algorithm is capable of tracking the length of the suture thread within a few percent as well as a few millimeters. The largest error was with the 175mm thread length. This indicates that the error is not a fixed value, nor is it dependent on length. It is most likely that the shape of the thread itself introduces the most errors and consequently is the source of most of the variance.

B. Qualitative Tracking Results

The video submitted with this paper contains several components that are meant to illustrate the thread tracking algorithm. The video also shows how the tracking looks in the segmented image space as well as in the raw images.

The first video demonstrates that the initialization algorithm is able to generate a successful point list even when the thread self intersects numerous times. Once the NURBS model is initialized, the tracking algorithm is demonstrated during thread motion, deformation, changing lengths, and knot tying.

In order to test the suture tracking algorithm, the linear stage was used to move the suture thread. One end of the thread was fixed to the moving stage, while the other end of the thread was fed through a hole in a immobile background. As the stage moves, the suture thread is pulled through the hole. This causes the thread to move, lengthen, and deform simultaneously. The thread is successfully tracked while the stage moves with a velocity of 9.7 mm/s. The video submitted with this paper shows how the thread is tracked.

The video also includes how the tracking algorithm is capable of tracking a suture knot as it is drawn tight. Key images from the loop tightening are shown in Fig 6. As the overhand knot loop is pulled tight, the control points get closer and closer together. Eventually, they are so close that the control points causing the loop are deleted. The NURBS model then continues to track the suture as it moves and deforms.

The included video also demonstrates how a robotically controlled surgical gripper can reorient and pick up a suture thread that was localized and tracked using the proposed algorithm. The suture thread was threaded through a tissue phantom (part number SCS-10 by Simulab Corp.). Once the user entered the suture thread point, the thread was traced and a grasp point near the end was identified. In addition to providing a grasp point, the NURBS curve model also provides a tangent to the suture thread. This allows the gripper to reorient as needed to grasp the suture. The robot uses visual feedback to translate and reorient until it is able to grasp the suture. The robot gripper coordinates were identified using colored circle markers attached to the gripper. The gripper in the video was designed in the lab [16] and is mounted to an IRB140 (ABB Ltd. Zurich, Switzerland) industrial robotic assembly arm. Once the arm grabs the suture thread, the thread is pulled to show that the gripper grasped the thread successfully.

VIII. Discussion and Conclusions

This paper presents a novel method that can be used to track a complete surgical suture thread online in real-time using a calibrated stereo vision system. In the proposed method, the suture thread model is segmented and initialized from a single seed point on the thread. Once the suture thread is initialized, a NURBS model tracks the thread as it moves in real-time. The segmentation operates at 15 Hz while the actual NURBS tracking operates at 20 Hz. The algorithm is robust against thread deformations, translations, and dynamic thread length. The algorithm is well suited to track a thread end as the thread is pulled through a tissue sample. The method was validated on bench top experiments using actual suture thread while the test bench environment was colored to emulate more realistic surgical environments.

The current limitations of the initialization algorithm include some sensitivity to undesired segmented pixels. It also is currently unable to detect when the thread intersects itself for a finite length (e.g. overhand knot).

The main limitation with the thread tracking algorithm is that it does not add or remove control points based on local thread shape. This can result in reduced measurement accuracy of the actual thread since some bends might be poorly approximated. Another deficiency is that when the thread is moving, it can lead the changes in the NURBS model. This also is due to the parallel processing threads. One way around this might be to incorporate a velocity model into the NURBS curve iteration.

Further work will focus on actively tracking the suture thread as it is manipulated by surgical grippers. This will allow the thread to be tracked as it is tied into a suture knot. Additionally, the NURBS fitting algorithm can be improved to support tracking the thread using previous velocity information.

Acknowledgments

This work was supported in part by the National Science Foundation under grants, IIS-0905344 and CNS-1035602, and National Institutes of Health under grant R21 HL096941 and R01 EB018108.

REFERENCES

[1]. Iyer, S.; Looi, T.; Drake, J. A Single Arm Single Camera System For Automated Suturing; Robotics and Automation (ICRA), 2013 IEEE International Conference on; 2013. p. 239-244.

[2]. Leonard, S.; L. Wu, K.; Kim, Y.; Krieger, A.; Kim, PCW. Smart Tissue Anastomosis Robot (STAR): A Vision-Guided Robotics System for Laparoscopic Suturing; Biomedical Engineering, IEEE Transactions on; Apr.. 2014 p. 1305-1317.

[3]. Javdani, S.; Tandon, S.; Tang, J.; O'Brien, JF.; Abbeel, P. Modeling and perception of deformable one-dimensional objects; Robotics and Automation (ICRA), 2011 IEEE International Conference on; 2011. p. 1607-1614.

[4]. Schulman, J.; Lee, A.; Ho, J.; Abbeel, P. Tracking Deformable Objects with Point Clouds; Robotics and Automation (ICRA), 2013 IEEE International Conference on; 2013. p. 1122-1129.

[5]. Padoy, N.; Hager, GD. 3D thread tracking for robotic assistance in tele-surgery; Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on; 2011. p. 2102-2107.

[6]. Padoy, N.; Hager, G. Deformable Tracking of Textured Curvilinear Objects; Proceedings of the British Machine Vision Conference; BMVA Press; 2012. p. 5.1-5.11.

[7]. Komodakis, N.; Tziritas, G.; Paragios, N. Fast, Approximately Optimal Solutions for Single and Dynamic MRFs. Computer Vision and Pattern Recognition, 2007. CVPR '07; IEEE Conference on; 2007. p. 1-8.

[8]. Kass M, Witkin A, Terzopoulos D. Snakes: Active contour models. International Journal of Computer Vision. 1988; 1(4):321–331. [Online]. Available: http://dx.doi.org/10.1007/BF00133570.

[9]. Meegama RGN, Rajapakse JC. NURBS snakes. Image and Vision Computing. 2003; 21(6):551–562. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0262885603000660.

[10]. Frangi, A.; Niessen, W.; Vincken, K.; Viergever, M. Multiscale vessel enhancement filtering. In: Wells, W.; Colchester, A.; Delp, S., editors. Medical Image Computing and Computer-Assisted Intervention — MICCAI'98. Springer Berlin Heidelberg; 1998. 1496. p. 130-137.ser. Lecture Notes in Computer Science[Online]. Available: http://dx.doi.org/10.1007/BFb0056195

[11]. Steger G. An unbiased detector of curvilinear structures. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998; 20(2):113–125.

[12]. Martinsson, H.; Gaspard, F.; Bartoli, A.; Lavest, J. Reconstruction of 3D Curves for Quality Control. In: Ersbøll, B.; Pedersen, K., editors. Scandinavian Conference on Image Analysis. Springer Berlin / Heidelberg; 2007. p. 760-769.

[13]. Piegl, L. The NURBS book. 2nd. Tiller, W., editor. Springer-Verlag New York, Inc.; New York, NY, USA: 1997.

[14]. Kaul V, Yezzi A, Tsai YJ. Detecting curves with unknown endpoints and arbitrary topology using minimal paths. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2012; 34:1952–1965. [PubMed: 22201054]

[15]. Cham, T-JCT-J.; Cipolla, R. Stereo coupled active contours; Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 1997.

[16]. Liu, T. Design and Prototyping of a Three Degrees of Freedom Robotic Wrist Mechanism for a Robitic Surgery System. Case Western Reserve University; Cleveland, OH: 2010. Master's thesis

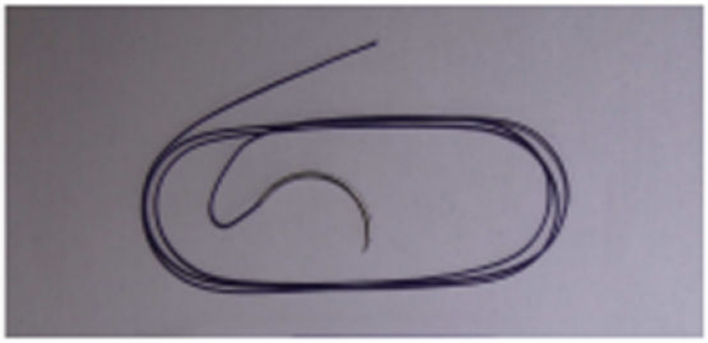

Fig. 1.
A sample suture needle kit has a semi-circular needle crimped onto a violet suture thread. This particular kit was manufactured by Ethicon (Cincinnati, Ohio) (part number J341).

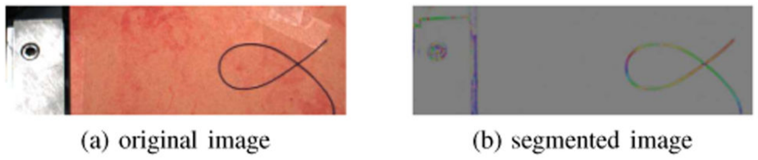

Fig. 2.

A sample segmented image (b) together with the corresponding original image (a). The segmented image uses false color which denotes the direction of the pixel estimated by the algorithm. In addition to the suture thread, other image edges are also segmented. This is because, in the gray scale image, the border appears to be a contrasting location to the surrounding edges. The thread self intersection can be identified from the direction information as can be seen from the different colors of the intersecting thread segments. Suture thread initialization and tracking must be robust against falsely segmented regions.

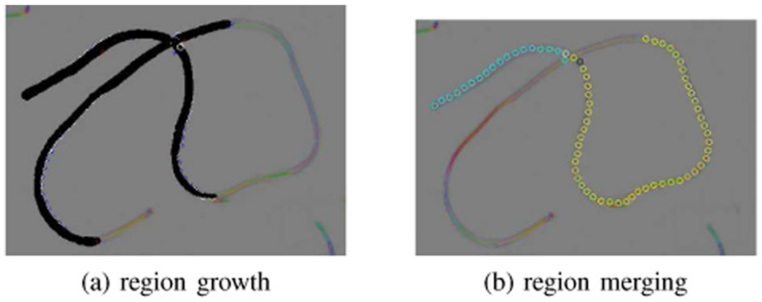

Fig. 3.
As the segmented suture thread region grows, it will branch as in (a). Once the regions are done growing, they are connected together based on their alignment. This is illustrated in (b).

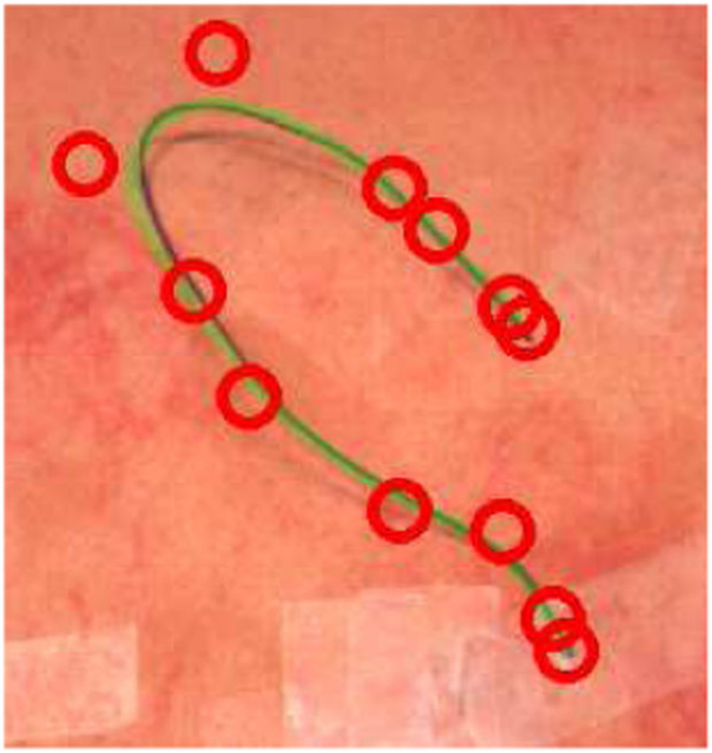

Fig. 4.

This is an image of the thread model overlaid onto the camera image. The semi translucent green curve is the NURBS curve, while the red circles are control points.

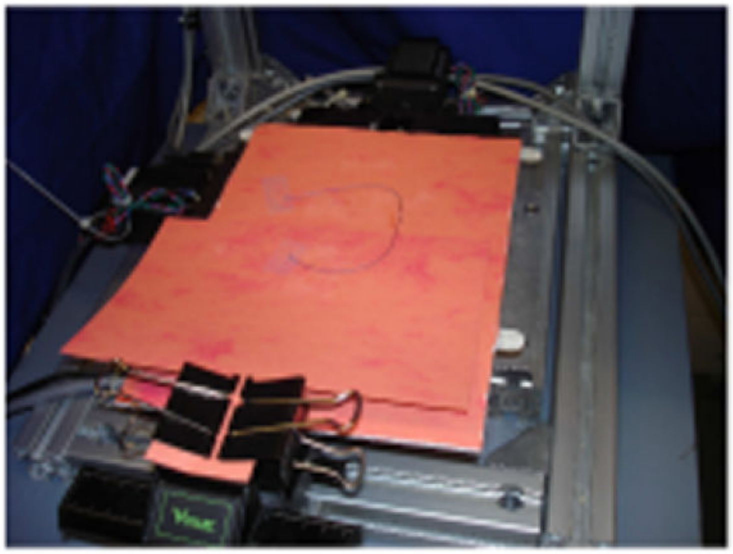

Fig. 5.

The X-Y linear stage with a sample of thread. The orange and red patterned construction paper is meant to emulate some of the colors and patterns that might be present in an actual surgical environment. One end of the suture thread is affixed to the immobile piece of paper while the other is affixed to the linear stage. This allows the thread to deform as it is tracked.

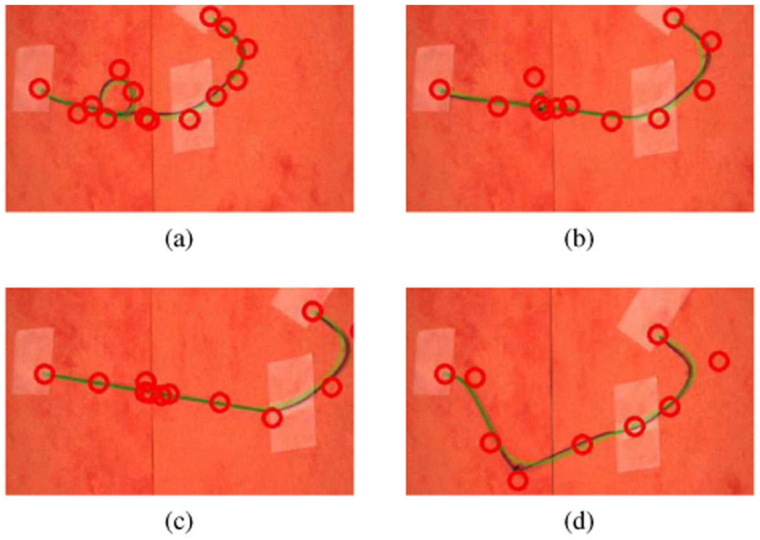

Fig. 6.

As the suture knot is pulled tight in (a), the control points are drawn closer together. Eventually, the control points converge into a narrow area (b). When the area is small enough, the control points are pruned (c). When the tight knot is moved, it is then tracked by a single control point (d).

TABLE I

The measured (ℓ) and detected ($\sim \ell$) length of the suture threads. The standard deviation (σ) and the percent error (%) are also reported.

ℓ(mm)	$\sim \ell$(mm)	σ (mm)	%
107	108.8	2.41	1.68
175	181.4	2.65	3.66
200	203.53	2.77	1.76