2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Visual Sensing of Continuum Robot Shape Using Self-Organizing Maps

Jordan M. Croom, *Student Member, IEEE*, D. Caleb Rucker, *Student Member, IEEE*,
Joseph M. Romano, *Student Member, IEEE*, and Robert J. Webster III, *Member, IEEE*

***Abstract*— Shape control of continuum robots requires a means of sensing the the curved shape of the robot. Since continuum robots are deformable, they take on shapes that are general curves in space, which are not fully defined by actuator positions. Vision-based shape-estimation provides a promising avenue for shape-sensing. While this is often facilitated by fiducial markers, sometimes fiducials are not feasible due to either the robot's application or its size. To address this, we present a robust and efficient stereo-vision-based, shape-sensing algorithm for continuum robots that does not rely on fiducials or assume orthogonal camera placement. The algorithm employs self-organizing maps to triangulate three-dimensional backbone curves. Experiments with an object with a known shape demonstrate an average accuracy of 1.53 mm on a 239 mm arc length curve.**

I. Introduction

Continuously flexible robots offer a number of potential advantages over traditional, rigid-link manipulators in certain applications [21], [23]. They are particularly useful for working in obstacle-filled environments in a manner similar to an elephant trunk or octopus tentacle. Example continuum robots applications include subsea manipulation, industrial inspection, and jet-washing [21], as well as minimally invasive surgery (see e.g. [4], [6], [7], [26], [28]).

Our primary motivation for the vision-based, shape-sensing algorithm described in this paper is medical continuum robotics, particularly the concentric-tube continuum robot design also known as an active cannula, shown in Fig. 1 [7], [22], [26]. This particular robot is among the thinnest continuum robots developed to date (typically 1-2.5 mm in diameter), and can be constructed in diameters ranging from 200 μm-10 mm, a range defined by the commercially available diameters of the Nitinol tubes from which they are made. While shape-sensing is challenging for all continuum robot designs, it is particularly challenging for very thin robots.

Several methods exist for encoding the shape of curved, flexible devices. One involves embedding strain sensors along the length of the device, as implemented in the robotic cockroach antennae in [15]. Fiber optic sensors have also been developed for measuring strain in flexible objects [10], [19], [20]. However, while strain gages and optical fibers may be options for larger-scale continuum robots, they are prohibitively bulky for integration in medical continuum robots, which are designed to be as thin as possible.

One efficient and straightforward method of real-time shape-sensing for continuum robots is the use of fiducial markers on the robot. Hannan and Walker employed this approach using a high-speed monocular camera system to observe fiducial bands on an elephant-trunk manipulator [11]. They then fit a series of circular arcs to the fiducial markers to describe the shape of the robot. Limitations of this method include dependence on fiducials and the approximation of the shape by circular arcs, which may not be accurate in all cases. This method was subsequently applied to set point regulation, where the orientations of various reference planes along the trunk were controlled [5]. Proof-of-concept end-point control based on stereo vision has also been accomplished with an active cannula by Webster, et al. [27] using a tip-mounted fiducial. It is useful to extend these concepts to control an entire general robot curve, which requires sensing of that curve. It is also desirable to accomplish this without relying on fiducial markers, since reliably attaching and segmenting many fiducials is challenging – particularly with very thin robots with telescoping backbones. These factors motivated the algorithm described in this paper, which triangulates shape without fiducials.

Some prior work has been done in the area of shape sensing for medical robots and devices. Webster, et al. used monocular [24] and stereo [25] optical cameras to sense the shape of a steerable needle embedded in transparent phantom tissue by post processing images. The system was subsequently used for closed-loop control by tracking the advancement of the needle tip [12]. Lee and Poston used two cameras to triangulate curve points on a catheter using epipolar geometry [16]. While accurate and robust, this method requires a brute force search of all points in one image for every point in the other using epipolar geometry

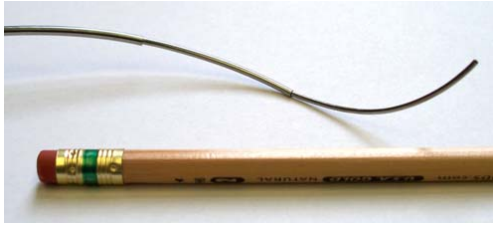

Fig. 1. Picture of an active cannula, a miniature surgical continuum robot actuated by the translation and rotation of concentric precurved tubes.

J.M. Croom, D.C. Rucker, and R.J. Webster III are with Vanderbilt University, Nashville, TN 37235, USA (e-mail: jordan.m.croom, daniel.c.rucker, robert.webster@vanderbilt.edu).

J.M. Romano is with University of Pennsylvania, Philadelphia, PA 19104, USA (e-mail: jrom@seas.upenn.edu).

This material is based upon work supported in part by National Science Foundation grant 0651803 and in part by National Institute of Health grant R44 CA134169. Manuscript received September 15, 2009.

to establish point correspondences. The algorithm proposed in this paper is similar in concept, but reduces the search space significantly through the use of Self-Organizing Maps.

Camarillo et al. proposed a "voxel-carving" algorithm to extract the position of their manipulator by projecting segmented pixels from three orthogonal views into a 3D voxel space to determine which voxels are occupied by the manipulator [4]. Using this 3D point cloud, the data was ordered in one of the views using an angle parameter from a fixed point outside the data set, and then smoothed. Though good accuracy and speed were obtained with this method, the use of three orthogonal cameras may not be feasible in all medical applications. In many medical applications we envision having at most two cameras, as would be the case in biplane fluoroscopy or binocular endoscopy. Furthermore, the method used for ordering the data may not be well-suited to more complex curves. General methods also exist for extracting camera parameters and reconstructing a 3-D shape from a number of uncalibrated cameras [2], but these methods are unnecessarily complex for simpler cases in which the fixed transformations between cameras and the world frame can be known a priori.

The Self-Organizing Map (SOM) structure was originally developed as a way to find lower dimensional patterns in complex, high-dimensional data. Kohonen [13] provides an informative overview of how and why SOMs are implemented, including possible variations in the algorithm. In this paper we use SOMs to find a simplified curve which approximates a 3D shape of a continuum robot. This is similar in nature (though different in implementation) to 3D surface reconstruction with SOMs, which has been extensively studied (see e.g. [1], [14], [18]). Kumar et al. addressed the problem of extracting a 3D curve from a 3D point cloud [14], and their work inspired our application of SOMs to continuum robots.

In this paper, a shape-sensing algorithm is presented which eliminates the need for fiducials, multiple orthogonally-placed cameras, and assumptions that the robot be circularly or near-circularly curved.

In this algorithm, a set of 3D backbone reference points, ordered according to arc length along the robot, are iteratively generated in such a way that their projections in the 2D camera views accurately approximate the 2D point distributions. The algorithm presented in this paper is an extension of previous advances in the use of self-organizing maps in 3D reconstruction in that we eliminate the need for a 3D point cloud as input and rely only on segmented images from stereo cameras. The segmented images are used in tandem to "train" representative reference points to fit the silhouette of the robot, order the points according to arc length, and then project them into 3D. The resulting ordered set of 3D backbone curve points can then be fit using a parametrically-defined curve in Cartesian coordinates.

II. SOM Continuum Robot Shape Sensing

Our algorithm assumes that one begins with two binary images, each containing pixels labeled as either containing the robot or not containing the robot. We explain how these are obtained via image processing in our proof-of-concept experimental system in Section III. Our algorithm works directly with these binary images, without first triangulating a 3D point cloud from them, so we refer to it as stereo SOM. Before addressing specifics of our stereo SOM algorithm, we provide a general overview of reconstruction of a parametric space curve from a large 3D point cloud using the conventional SOM algorithm.

A. Conventional SOM

For an extensive treatment of conventional SOM algorithms for 3D reconstruction we refer the reader to Kumar et al. [14], whose notation we follow in this paper. We begin by letting $S = \left\{P_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, ..., N\right\}$ be a 3D point cloud representing the physical presence of a continuum robot in space. We then choose a set of k reference vectors (3D points that the algorithm will iteratively adjust to approximate the centerline, or backbone curve of the robot), $M = \{Q_l = (x_l, y_l, z_l), l = 1, ..., k\}$. The initial positions of the reference vectors can be chosen in any way, for instance by choosing a random sample of points from S, or by sampling the curve estimate from a previous algorithm output. Let $Q_c(P_i)$ and $Q_{c'}(P_i)$ be the nearest and second nearest (in the euclidean sense) reference vectors, respectively, to each $P_i \in S$.

Using these definitions, one can then establish an adjacency matrix as illustrated in Fig. 2 for M on the basis of the nearest and second-nearest reference vectors for each point P_i as,

$$\begin{aligned} A &= \{a_{mn}\}, \quad \text{where} \\ a_{mn} &= \sum_{i}^{N} \begin{cases} 1, \text{ if } Q_m = Q_c(P_i) \quad \wedge \quad Q_n = Q_{c'}(P_i) \\ 0, \text{ else} \end{cases} \\ &\text{for} \quad m \leq k, n \leq k. \end{aligned} \tag{1}$$

Thus, the matrix A provides a sense of the "order" of the reference vectors because it shows which reference vectors are close to the same data points and thus each other. If $a(m, n) = 0$, there are no reference vectors with Q_m and Q_n as their best matching and second best matching reference vectors, respectively. Therefore Q_m should not be adjacent to Q_n. If S represents an ordered set of points along a curve, A will necessarily be constructed such that the only

	1	2	3	4	5
1	0	2	0	0	0
2	1	0	1	0	0
3	0	1	0	2	0
4	0	0	1	0	1
5	0	0	0	2	0

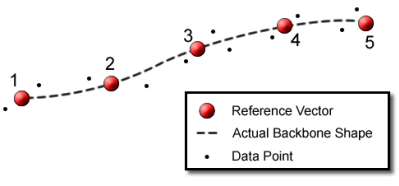

Fig. 2. Illustration of the structure of the adjacency matrix for correctly ordered reference vectors. The rows indicate the nearest reference vector and the columns indicate the second nearest reference vector to a given data point. Thus, for example, there are two data points nearest to reference vector 1, for whom the second nearest reference vector is 2. Therefore, the 1,2 element of the matrix is 2.

non-zero elements of the matrix lie on either side of the main diagonal. Thus, the correct ordering of the points in M will result in this form. The sparse reverse Cuthill-McKee ordering of matrix A can be used to find a permutation matrix which reorders A to have its nonzero elements concentrated near the diagonal. In our experiments in Section III, we use Matlab's `symrcm` function to implement this operation. This results in the correct ordering of M such that there is little or no topological error. In the context of a 3D curve, zero topological error would mean having the reference vectors in order of increasing arc length along the curve, as depicted in Figure 2. More information on the ordering technique can be found in [9].

After M is ordered, the algorithm "trains" the reference vectors. In the algorithm presented here, a batch-training construct was chosen for both its simplicity and speed when compared to other methods (see [13] for an overview of various training methods). In batch-training, each reference vector (Q_l) gets updated to a new position (Q_l^*) based on the average location of the points in its own neighborhood and the neighborhoods of the two closest reference vectors according to,

$$Q_l^* = \frac{\sum_{j \in U_l} |N_j| \, \overline{N_j}}{\sum_{j \in U_l} |N_j|} \tag{2}$$

where the neighborhood of Q_j is $N_j = \{P_i \,|Q_j = Q_c(P_i)\}$, $U_l = \bigcup_{b=l-n}^{l+n} Q_b$, and n is the range of neighborhoods to include in each training calculation. The batch-training algorithm is simply a weighted average. This form of the equation saves computation time because it allows precomputation of both the averages and number of elements in each N_j for the batch-training algorithm, both of which appear numerous times in overlapping neighborhoods. The training loop is performed for multiple iterations, quickly drawing reference vectors toward the trajectory of the curve, r.

B. Algorithm Modification for Stereo Images

The method of [14] above assumes that one has a 3D point cloud to begin. In the case of stereo vision for a continuum robot without fiducial markers, it can be computationally expensive to obtain such a 3D point cloud from 2D images, because a large number of point correspondences between the two camera images must be established to triangulate the 3D points. These point correspondences can be established by a search algorithm using epipolar geometry as in [8], but this is generally computationally expensive.

Thus, instead of first generating a large 3D point cloud and then applying SOM, we have adapted the SOM algorithm to use only the two separate 2D image views. We do so by training the 2D projections of the 3D reference vectors in parallel using the data in the two camera views, and then ordering both sets according to one set's adjacency matrix. We subsequently triangulate the projections into 3D and then project back into the camera views prior to the next training step. The triangulation of pairs of 2D reference vector projections into 3D is only performed once for each reference vector pair during a single iteration of the training loop, and the point correspondences are automatically maintained by the shared ordering of the reference vectors in both views. In this way, we can reconstruct a 3D curve from stereo camera data at least as fast as the conventional SOM algorithm presented in II-A can approximate a 3D curve from a 3D point cloud. Thus, by circumventing the step of generating a large 3D point cloud, our algorithm gains efficiency.

C. Stereo SOM

The algorithm begins with segmenting the images from each camera. The occupied pixels from each image are then stored in two separate matrices (one for each camera) of pixel coordinates. An initial set of reference vectors, each of which is denoted $Q3D_l$, must now be chosen to initialize the SOM algorithm. For the experiments described in Section III, the reference vectors were initialized as a straight line of points originating at the known robot base frame in the direction of deployment (the positive z-axis). Choosing an appropriate number of reference vectors is important to the performance of the algorithm as we will discuss in Section III. In general, the optimal number of reference vectors to choose will depend on the diameter, length, and potential shapes of the robot.

Next, $Q3D$ is projected into each camera frame, then converted from points in the camera frame to pixel coordinates. These pixel coordinates will hereafter be referred to as Q's. Next, the function "bestQs" finds $Q_c(P_i)$ and $Q_{c'}(P_i)$ for all P_i. This information is then used to sort the P_i into groups corresponding the neighborhoods of each Q. Thus, there will be one group of P_i's for each reference vector, although it is possible that some neighborhoods will be empty. In this case, the Q with an empty neighborhood will be removed in the training step.

We then choose one image and use its adjacency matrix Q's to sort the Q's in both images. This assures us that the Q's from each image are in the same order so that point correspondences are known before triangulation into 3D. It is desirable to use the image which is least likely to contain topological ambiguity or self intersections, so we use the projection that contains the largest range of Q's in either the x or y direction.

After reordering both sets of Q's, the $Q3D$'s can be triangulated using the point correspondences. As defined by [17], the triangulation algorithm uses the fundamental epipolar matrix F and the normalized camera coordinates of each pair of Q's, Q_n and $Q_{n'}$ (the corresponding Q's from each camera) to calculate $D = Q_{n'}^T F Q_n$, which is a measure of how well each pair of Q's correspond (how close their epipolar lines are to intersecting). If any of the D's exceeds some threshold value D_{max}, the loop will repeat by reprojecting the triangulated $Q3D$'s back onto the image planes and beginning a new training iteration. In our experiments, the algorithm usually did not need to repeat. The first training step is robust enough to pull even poorly initialized points very close to the backbone curve in a single iteration. Simply increasing the number of training epochs, or maximum number of times the training loop itself is

Algorithm 1 Biplane SOM

```
loop
  Get Camera Frames
  Segment Images
  Get Pixel Matrices
  Get Deployed Arc Length
  Initialize Q3D
  while D ≥ D_max and Iter≤MaxIter do
    Project Q3D into Camera Pixel Coordinates
    Run bestQs() and sortPoints()
    batchTrain()
    Add Q's if necessary
    if Q's added then
      batchTrain()
    end if
    Calculate Largest Spread
    Run aMatrix() according to largest spread
    Find Projected Q3D's from 2D Q's
    Calculate D
  end while
  if Q3D is reversed then
    Flip order
  end if
  Fit Parametric Curve to Q3D
  Reparametrize w.r.t. Arc Length
end loop
```

repeated, usually avoids repetitions of the entire SOM loop. For the experiments in this paper, a value of 10 was used for the maximum number of training epochs. Once the $Q3D$'s have converged, a parametric curve can be fit to them to create a backbone representation. Initializing $Q3D$ for future approximations of the backbone (e.g. if the robot is moving) can then be done by finding the position of various points along the previously determined curve.

III. Experiments

In order to evaluate the accuracy and speed of the algorithm proposed in Section II, experiments were performed using a mock-up of a miniature continuum robot, shown in Fig. 4, with a known backbone shape. The mock-up was rapid-prototyped with a circular cross section with an outer diameter of 6.4 mm. The backbone reference curve was defined by a parametric polynomial approximating a helix with linearly increasing radius. In our experiments, we reconstruct the 3D backbone of this mock-up from stereo camera images using the stereo SOM algorithm, and evaluate the accuracy by comparing the results against the known curve.

A. Image Acquisition and Preprocessing

As shown in Fig. 4, images of the mock-up were taken by a custom stereo camera system consisting of two Sony XCD-X710 digital cameras, capable of capturing 8-bit grayscale images at a resolution of 1024×768 pixels with a rate of 30 frames per second. The cameras are mounted on an adjustable truss structure so that they are capable of viewing the mock-up from a variety of overhead angles. The image background is covered by matte white felt to simplify image preprocessing for experimental validation.

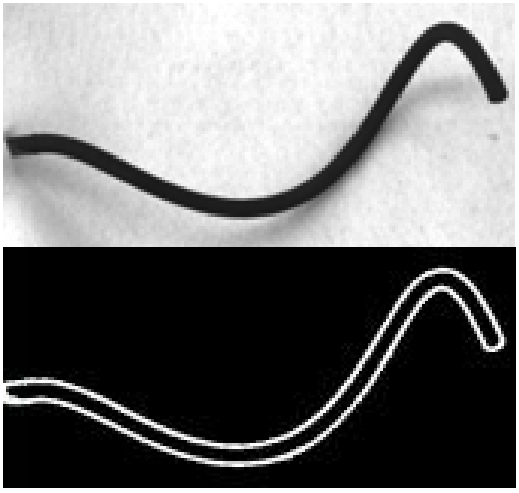

Fig. 3. The initial image from a single camera (top) and the final segmented image after processing (bottom).

Prior to beginning the experiments, we fixed the two cameras rigidly in place and used Matlab's Camera Calibration Toolbox [3] to calibrate them. To acquire images and perform several preprocessing steps, we used the OpenCV library for C++. We employed a standard edge-detection algorithm as follows. First, we convolved our original images with a 3×3 symmetric Gaussian kernel in order to filter the image, removing any background noise. Next, we enhanced edges via convolution with a 3×3, 2-dimensional Laplacian. Finally, we thresholded the resulting image. Adjusting the threshold appropriately to account for lighting conditions, we obtained a set of pixels almost solely along the robot edges, as seen in

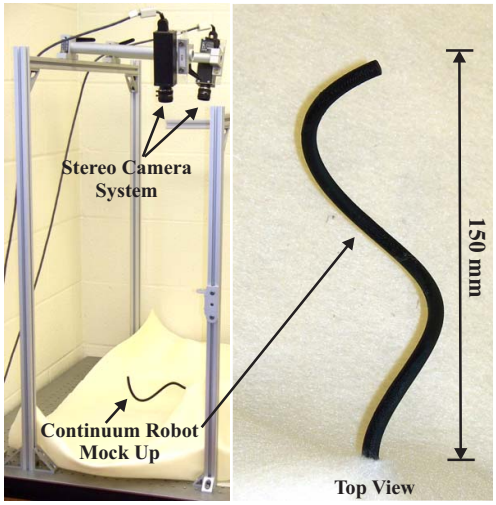

Fig. 4. A photo of the experimental setup. The continuum robot mock-up was actuated to various orientations while images were taken by our stereo camera system.

TABLE I

EXPERIMENTAL RESULTS FOR ACCURACY, PRECISION, AND SPEED

Number of Q's	Mean Error (mm)	Std. Dev. Error (mm)	Mean Time (s)
20	3.14	0.68	0.188
30	1.94	0.42	0.195
40	1.73	0.29	0.232
50	1.53	0.31	0.248
60	1.59	0.27	0.311
70	1.60	0.28	0.365
80	1.60	0.43	0.636
90	1.78	1.00	0.724
100	1.88	0.93	0.993

Fig. 3. We note that in practical implementation – particularly in the surgical setting – more advanced image pre-processing will be necessary. However, this simple method is sufficient to demonstrate the stereo SOM algorithm and perform the experiments that follow.

B. Experimental Procedure

The robot mock up was gripped at its base using a rotary positioning stage (Velmex model A5990TS C029338), and rotated and translated to 20 different poses. Images were taken and processed into two dense 2-D point distributions of edge points in image coordinates as described above in Section III-A. For each pose, the SOM algorithm developed in Section II was used to reconstruct the 3D backbone curve based on 800 data points randomly selected from the collection of edge pixels in each image. A set of initial reference vectors was established extending out 150 mm (the length in the z-axis of the mock-up) from the robot base in a straight line.

For each of the 20 experimental poses, the SOM algorithm was initialized with a number of reference vectors equally spaced along a 150 mm straight line from the origin along the positive z-axis. For numbers of initial reference vectors ranging from 20 to 100 in increments of 10, the time required to obtain the ordered 3D backbone points from the original 2D images was measured using Matlab's built-in `tic` and `toc` functions. The average run times are shown in Table I.

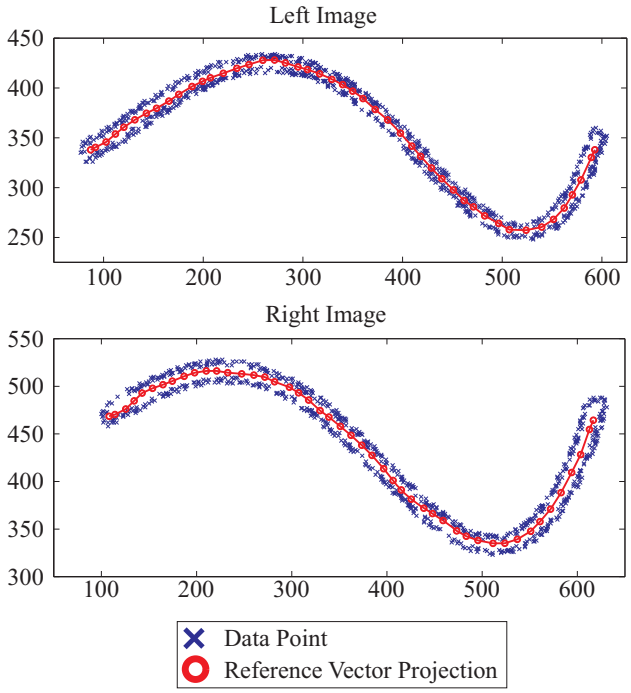

Fig. 5. The 2D data sets in the left and right camera views with the 2D projections of the 3D reference vectors for a case with 50 initial reference vectors.

C. Results and Discussion

The accuracy of the algorithm over the 20 positions was also evaluated for each number of initial vectors. After each run of the algorithm, a 6^{th} degree polynomial curve was fit to the resulting 3D backbone points (the ground truth backbone curve was also of degree 6). This experimental curve was rigidly registered to the ground truth backbone curve by comparing 100 points at equal increments along the arc lengths of each curve. The mean euclidean error in these points is shown in Table I Plots of the 3D points output by the algorithm, the curve fit to these points, and the ground truth curve are shown together in Figure 6. On average, each of the 20 orientations showed similar levels of error.

Table I summarizes the speed, accuracy and precision of the SOM algorithm in our experiments. In terms of accuracy and consistency, there is an optimal number of initial reference vectors around 50. The reason for this is illustrated by Fig. 6. If the number of reference vectors is too small, the error will increase because the curve is sparsely sampled, and the polynomial fit will not necessarily match the actual curve. If the number of reference vectors is too large, the model will start to follow any noise or random clumping in robot pixel positions, rather than the backbone of the device. As the number increases and reference vector neighborhoods become smaller, the ability of the training algorithm to pull reference vectors to the center of the curve

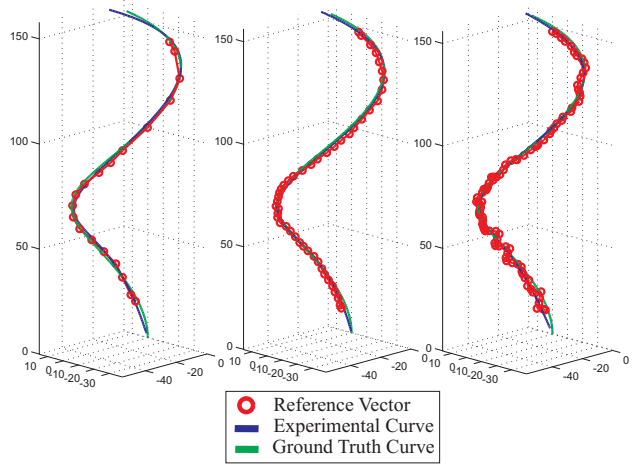

Fig. 6. A comparison of the reconstruction with 20 (left), 50 (middle), and 100 (right) initial reference vectors. Too few reference vectors lead to inaccuracy, while too many lead to increased run time, increased sensitivity to noise, and the potential for topological error.

is reduced. Another negative effect with large numbers of reference vectors is the potential for topological ambiguity (points not in a clear order), which causes an inaccurate curve fit and is again due to the noise inherent at this scale. This is the reason for the large standard deviation for 90 and 100 initialized Q's. Increasing the number of reference vectors also makes the algorithm more computationally intensive and thus increases the processing time.

In Figure 6, it is apparent that the reference vectors do not reach to the ends of the curve. This is due to the effect that batch-training has of pulling reference vectors into the middle of the curve – the outermost 2D reference vectors are pulled slightly away from the ends of the curve as shown in Figure 5.

Overall, the algorithm is relatively robust in the range of 40 to 70 initial reference vectors. The optimal number of 50 gives a mean centerline accuracy of 1.53 mm – which is relatively small considering that our mock up had an outer diameter of 6.4 mm – and a mean run time of 0.248 s. Thus, the the algorithm is capable of running at 4.0 Hz in Matlab 7.3.0 on a 2.13 GHz PC with 2.0 GB of RAM under the Linux Gnome Red Hat operating system, excluding the computation involved in image preprocessing.

IV. Conclusion

In this paper we have presented an algorithm based on the concept of Self-Organizing Maps that can be used to reconstruct a continuum robot's 3D backbone curve from segmented 2D stereo images. Bypassing the generation of a large 3D point cloud makes the algorithm efficient, and our experiments demonstrated a reconstruction accuracy of 1.53 mm at 4.0 Hz with a known robot curve seen from a number of different angles. This algorithm will be useful for encoding the shape of continuum robots, which is a necessary prerequisite to closed-loop shape control for the new class of miniature continuum robots being developed for minimally invasive surgery. In the future, we envision our algorithm being implemented with image feedback from a variety of medical imaging sources including biplane fluoroscopy and stereo endoscopy.

References

[1] A. Baader and G. Hirzinger, "A self-organizing algorithm for multisensory surface reconstruction," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 81–88, 1994.

[2] R. Berthilsson and K. Aström, "Reconstruction of 3-D curves from 2-D images using affine shape methods for curves," *Computer Vision and Pattern Recognition 1997 Proceedings*, pp. 476–481, 1997.

[3] J.-Y. Bouguet, "Camera calibration toolbox for matlab," June 2008. [Online]. Available: *http* : *//www.vision.caltech.edu/bouguetj/calib_doc/index.html*

[4] D. B. Camarillo, C. F. Milne, C. R. Carlson, M. R. Zinn, and J. K. Salisbury, "Mechanics modeling of tendon-driven continuum manipulators," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1262–1273, 2008.

[5] V. K. Chitrakaran, A. Behal, D. M. Dawson, and I. D. Walker, "Setpoint regulation of continuum robots using a fixed camera," *Robotica*, vol. 25, pp. 581–586, 2007.

[6] P. Dario, M. C. Carrozza, M. Marcacci, S. D'Attanasio, B. Magnami, and G. M. O. Tonet, "A novel mechatronic tool for computer-assisted arthroscopy," *IEEE Transactions on Information Technology in Biomedicine*, vol. 4, no. 1, pp. 15–29, 2000.

[7] P. Dupont, J. Lock, and E. Butler., "Torsional kinematic model for concentric tube robots," *IEEE International Conference on Robotics and Automation*, pp. 2964–2971, 2009.

[8] O. Faugeras, Q.-T. Luong, and T. Papadopoulo, *The Geometry of Multiple Images: The Laws that Govern the Formation of Multiple Images of a Scene and Some of Their Applications*. Boston, MA: The MIT Press, 2004.

[9] A. George and J. Liu, *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, 1981.

[10] D. T. George and D. K. Bogen, "A low-cost fiber-optic strain gage system for biological applications," *IEEE Transactions on Biomedical Engineering*, vol. 38, no. 9, pp. 919–924, September 1991.

[11] D. M. Hannan and D. I. Walker, "Vision based shape estimation for continuum robots," *Robotica*, vol. 23, no. 5, pp. 645–651, 2005.

[12] V. Kallem and N. J. Cowan, "Image guidance of flexible bevel-tip needles," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 67–78, 2009.

[13] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, pp. 1–6, 1998.

[14] G. S. Kumar, P. K. Kalra, and S. G. Dhande, "Curve and surface reconstruction from points: an approach based on self-organizing maps," *Applied Soft Computing*, vol. 5, no. 1, pp. 55–66, 2004.

[15] J. Lee, S. N. Sponberg, O. Y. Loh, A. G. Lamperski, R. J. Full, and N. J. Cowan, "Templates and anchors for antenna-based wall following in cockroaches and robots," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 130–143, 2008.

[16] W.-S. Lee and T. Poston, "Rapid 3D tube reconstruction from nearby views," in *Fifth International Conference in Central Europe in Computer Graphics and Visualization*, 1997, pp. 262–271.

[17] G. Mariottini and D. Prattichizzo, "EGT: a toolbox for multiple view geometry and visual servoing," *IEEE Robotics and Automation Magazine*, vol. 3, no. 12, 2005.

[18] R. M. Palenichka and M. B. Zaremba, "Multi-scale model-based skeletonization of object shapes using self-organizing maps," *ICPR*, pp. 143–146, 2002.

[19] Y.-L. Park, K. Chau, R. Black, and M. Cutkosky, "Force sensing robot fingers using embedded fiber bragg grating sensors and shape deposition manufacturing," *IEEE International Conference on Robotics and Automation*, pp. 1510 – 1516, 2007.

[20] R. Posey Jr., G. Johnson, and S. Vohra, "Strain sensing based on coherent rayleigh scattering in an optical fibre," *Electronics Letters*, vol. 36, no. 20, pp. 1688–1689, 2000.

[21] G. Robinson and J. B. C. Davies, "Continuum robots – a state of the art," *IEEE International Conference on Robotics and Automation*, pp. 2849–2854, 1999.

[22] D. Rucker and R. J. Webster III, "Parsimonius evaluation of concentric-tube continuum robot equilibrium conformation," *IEEE Transactions on Biomedical Engineering Letters*, vol. 56, no. 9, pp. 2308–2311, 2009.

[23] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Applied Bionics and Biomechanics*, vol. 5, no. 3, p. 99 117, 2008.

[24] R. J. Webster III, J. S. Kim, N. J. Cowan, G. S. Chirikjian, and A. M. Okamura, "Nonholonomic modeling of needle steering," *International Journal of Robotics Research*, vol. 25, no. 5/6, pp. 509–526, May/June 2006.

[25] R. J. Webster III, J. Memisevic, and A. M. Okamura, "Design considerations for robotic needle steering," *IEEE International Conference on Robotics and Automation*, pp. 3599–3605, 2005.

[26] R. J. Webster III, J. M. Romano, and N. J. Cowan, "Mechanics of precurved-tube continuum robots," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 67–78, 2009.

[27] R. J. Webster III, J. P. Swenson, J. M. Romano, and N. J. Cowan, "Closed-form differential kinematics for concentric-tube continuum robots with application to visual servoing." *11th International Symposium on Experimental Robotics 2008, Springer Tracts in Advanced Robotics*, pp. 485 – 494, 2008.

[28] K. Xu and N. Simaan, "An investigation of the intrinsic force sensing capabilities of continuum robots," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 576–587, 2008.