Rapid 3D tube reconstruction from nearby views

Lee, Won-Sook — MIRALab, University of Geneva, Switzerland — Wonsook.Lee@cui.unige.ch

Tim Poston — CIeMed, ISS, National University of Singapore — tim@iss.nus.sg

Abstract We reconstruct a 3D tube (catheter or artery) from nearby X-ray views. Construct 2D skeleton curves from the image edge profiles by wave propagation, thin and smooth them, then reconstruct a 3D curve by epipolar correspondence, resolving side-point singularities by differential-geometric models. The resulting curve is highly accurate on data from a simulated catheter, whose 'true' 3D curve is known. Finding it from edges takes less than a second, sufficient for 'on the spot' reconstruction, and uses only the moving views normally created by the angiographer, rather than constrained and costly arrangements such as biplanar cameras.

1 Introduction

In angiography the full 180° range of views that would allow a CT reconstruction is problematic for reasons of time; the bolus of contrast agent that shows the arterial net moves and changes too much to sustain the inverse Radon hypothesis that the input images are projections of the same 3D opacity density. Human vision reconstructs 3D structures well, using a more readily available view that rolls briefly over a small range of angles.

Most 3D reconstruction schemes from 2D projections constrain the projection angles; we use the angle information from arbitrary nearby projections. Each time the radiologist rolls the point of view with the X-ray live, produces a narrow range of closely-spaced views, which our scheme can use. (Images are stamped with camera positions as a standard feature of the new C-Arm technology. Alternatively, include a reference X-opaque object from which the camera position can be deduced as in, *e.g.*, [Chang92].)

Absent lines, perspective and shading, 3D reconstruction must use views from multiple (relative) camera positions. Nguyen and Sklansky[Nguyen94] reconstructed arteries using a single-viewpoint sequence of images of the moving heart, constructing a successive-frame correspondence using a similarity measure based on angular changes along a fourth or fifth order polynomial-fit curve. More commonly, the camera moves or is doubled. Biplane angiography provides two views (usually orthogonal), and can be used to compute the 3D structure. However, its correspondence problem is hard (even for interacting humans, who solve it well for the small-eye-angle differences we are wired for), and has resisted automatic solution. Many systems solve it manually by identifying corresponding landmarks[Kitamura88] or by user input[Parker87]. Some require that the projections of an elliptical cross subsection fall on the same row in each projected image[Fessler91], constraining the movement of the camera. Wahle *et al.*[Wahle95] require the user to select striking points.

We simplify correspondence by using nearby views. Known correspondences can be triangulated more precisely with orthogonal views, but the small angle approach gains more from the similarity of neighboring ones, and from the possibility of longer-sequence refinement using a single-view camera with negligible time gap. A fully automated epipolar line correspondence allows the camera arbitrary movement.

This can be applied for many kinds of object, but here we address the case where one expects to see a simple tubular structure (notably a catheter). We reconstruct its 3D skeleton, with diameter estimates, from 2D profiles of views 2° apart, a typical spacing with the C-Arm. From edges extracted from two images, we use the distance waves to construct 2D tube-image center curves where tubularity is plausible, getting radius

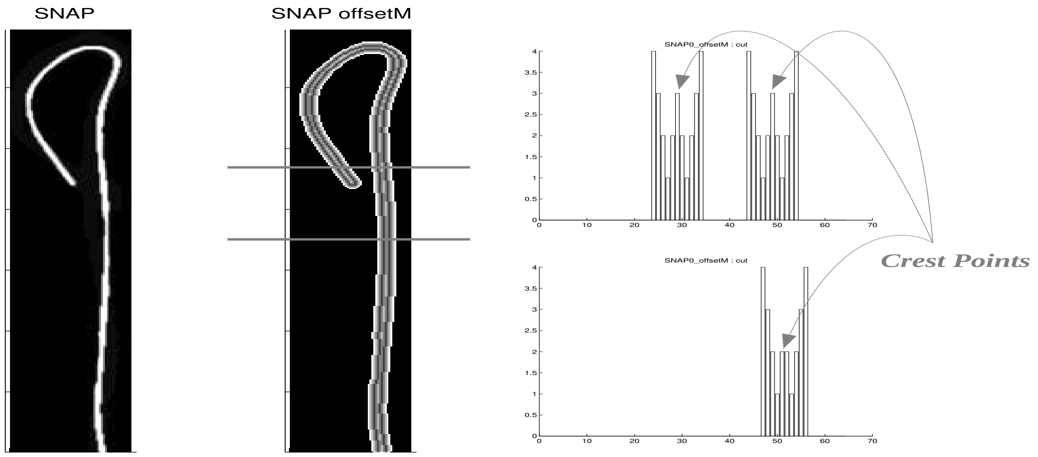

Figure 1: A catheter image (left), with distance curves after four wave advances (middle) and two lines of the resulting $\tau(x)$ values (right). These have value 1 on edge points and increase away from them up to distance 4, beyond which they are 0. The upper slice has two sharp crest points, with value 3, and the lower has one flat crest point with value 2.

data as a by-product. After connecting these into splines, we use epipolar reconstruction techniques to construct a 3D curve. Several rapid steps with lowpass and median filters remove singularities and smooth the curves.

The reconstruction scheme was developed on synthetic catheter images from the daVinci simulation project[daVinci96].

2 Spline-based smooth skeleton detection of a tube

2.1 Previous work

Much research has been done in tube edge detection, mainly of arteries, with excellent results[Brien94, Kutka96, Sun95, Sunka95]. Sun *et al.*[Sun95] obtained impressive results for a catheter image within the heart by line improvement. O'Brien *et al.*[Brien94] segmented vessels well by outward search from an initial point on the vessel structure, seeking a 'slightly darker' elongated region occupying less than a certain fraction of a moving square window, with 'slow' changes in pixel densities and (away from bifurcations and crossings) in width. The best choice of filtering and segmentation depends strongly on the data. Our aim here is efficient reconstruction, once we have a clear enough image to apply (for instance) the Sobel operator to get a good set of edge points.

2.2 Skeletonizing thin tubes

A thin-tube image has two very close edge curves, as sets of edge pixels, after edge detection. Rather than reconstruct these as, say, splines, we replace them by a single centre curve: a more useful identifier for the tube's location, and more robust against pixel level error. We build a distance-from-edge function by propagation between neigbouring pixels.

Large structures are best treated, as in [Mikheev94], by a mask that marks points as 'front' or 'passed since last step' (using a discrete circle and disc), but for tubes a few pixels wide we create a simple 3×3 mask $w_t = \begin{bmatrix} 0 & t+1 & 0 \\ t+1 & t+1 & t+1 \\ 0 & t+1 & 0 \end{bmatrix}$ at each time step t, and use it to propagate the wave. We initialize the wave's 'time of arrival' $\tau(x)$ to 1 for edge points, 0 elsewhere. To carry the wave N steps, initialize $t = 1$, and repeat N times

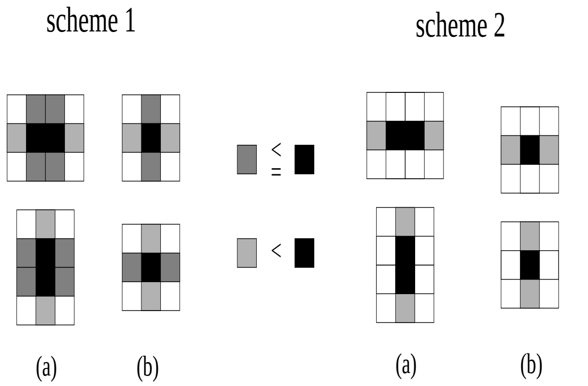

Figure 2: A pixel x is a crest point if it occurs as a black square in one of these patterns, with τ equal to the τ at any black neighbour, greater than or equal to τ at every dark grey pixel, and strictly greater than at every light grey one. Ignore τ at white points, and relations between τ at non-black points. (a) Two kinds of flat ridge (crest points). We select the middle point of two black points as a point in the skeleton. (b) Two kinds of sharp ridge.

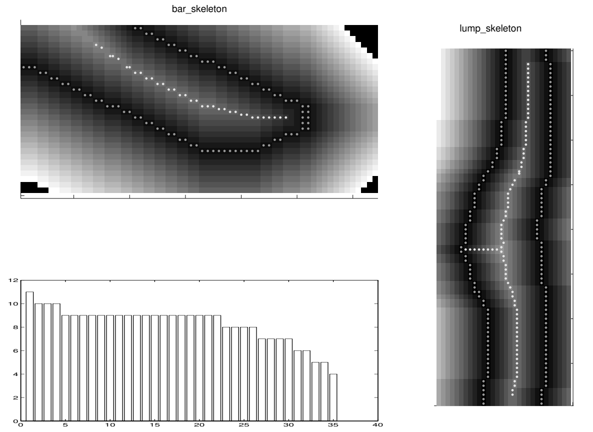

Figure 3: Edge points (grey) and crest points (white), against τ values shown by black for low τ, white for high. Scheme 2 selects the tip of the image at upper left and a curve in the lump at right, though they are not part of the desired skeleton. The graph shows τ values along the ridge above it, with the rapid drop-off typical of such lump cores, by which they can be detected and removed. With scheme 1, this kind of local self-collision rarely occurs.

```
construct w_t
list the points with τ = t + 1 . For each point [i][j] of these
    for all [x][y] with |x − i| ≤ 1 and |y − j| ≤ 1
        τ[x + i][y + j] ← pseudomin(τ[x + i][y + j], w_t[i][j])
increment t
```

where the pseudomin() function takes the smallest strictly positive value of a pair.

This algorithm assigns 'remoteness from edge' values τ at every point reached, with ridge crests along the skeleton of the thin-tube image. The crest τ values are lower where the tube is thinner, so we must detect them by 'crestness' rather than by a fixed value. Uncrested wavefront points with $\tau = N$ are detectable by having neighbours with $\tau = 0$; crest points as in Fig. 2.

We discard any point if at least one of its eight neighbours has $\tau = 0$, since this is an uncollided wavefront point, and any point with value 1, since it is an edge point. Next we check whether points are ridge-like (Fig. 1) in the x or y direction. There are two possible schemes (Fig. 2). Scheme 1 selects fewer points than scheme 2, but is much safer for local self collision (Fig. 3). As skeleton points we choose any sharp ridge point, and the midpoint of points flat in one or both directions. We retain these points' τ values, which are distances from the nearest edge points, and hence tube radii; a useful byproduct of skeleton construction.

Next, choose a minimal separation *Gap*. Iteratively from a seed point, select crest points with least pixel distance $>$ *Gap* from previous selections, to thin the skeleton, separating points and reducing their number. With $Gap = 1$, we can reduce the number of points by almost half.

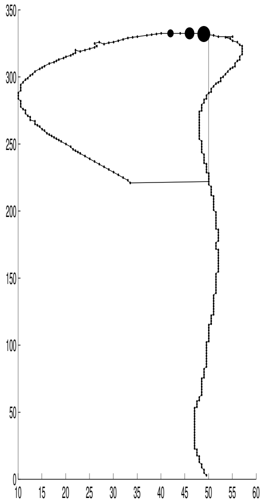

Figure 4: The seed point (biggest black dot) in list building is not usually an end point of the piece in the image. The median and smallest dots are the third and fifth points.

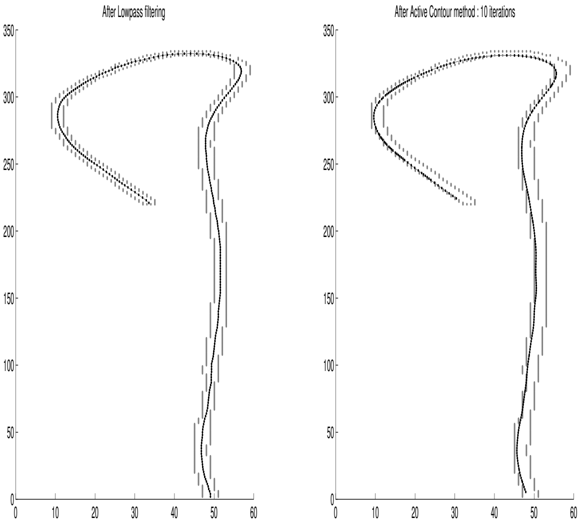

Figure 5: Left: point list (black) after reconnecting and applying a length 7 low pass filter, against the edge data (grey) to show its fit as a middle curve. With 237 points these steps took $< 0.004 sec.$ on a Sun SparcStation20. Right: edge spline (black) found more slowly by a snake method from the same points, against the edge.

The resulting set must then be organized as a chain. We use a 'cut' criterion of points being more than αGap apart, where α is typically around 4, and a 'reconnect' criterion of being within βGap, with a similar value for β.

1. Choose a seed point.
2. Make a linked list $\{p_i\}_{i=1\cdots N}$ by iteratively choosing the nearest point, saving the squared distance d_i^2 between p_i and p_{i+1}.
3. Cut into sublists wherever $d_i^2 > \alpha^2\, Gap^2$. (By the thinning process, the distance to the visually natural next point is typically just above Gap.)
4. Discard very short linked lists of points (< 4, say), which typically arise from noise.
5. Examine sublist ends. Cut and discard if τ rises sharply from the end, since this is not a tube-centre curve but a local self-collision of the wave, arising from a curvature maximum on the curve of edge points (Fig. 3). On a narrow, smooth object like a catheter, discarding such side branches restricts attention to the collisions of waves from opposite sides, and hence to a plausible center curve.
6. Reconnect nearest sublist ends less than $\beta^2\, Gap^2$ apart; usually this rejoins a break at the seed point. Other joins occur where the initial list building exhausted one component and jumped to the nearest point on another (Fig. 4).

2.3 Smoothing to reduce pixel level error

We assume a 2D profile to be smooth. Edge detection by pixel methods followed by linking into contours suffers from upward error propagation, with results not smooth at pixel level. Much work has used snake methods[Kass88] to get smooth edges, for instance using energy minimization methods to converge to a high edge-intensity path with low elastic bending energy[Lawton94]. This long computation with a lot of derivatives, and expensive integration over wide areas beyond the edges, is impractical for rapid results.

Snake methods catch a low-spatial-frequency image which expresses overall shape, rather than high frequencies which show detail and pixel level error. Our crest point

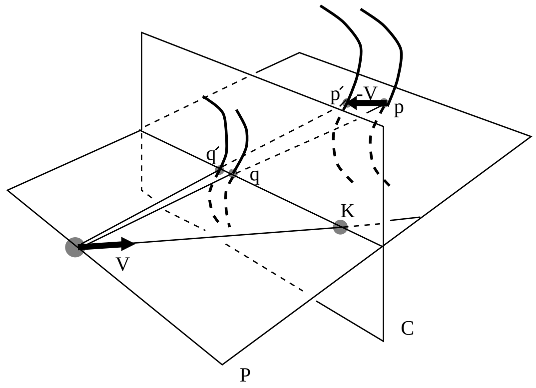

Figure 6: The planes **C** of the camera screen, and **P** of the triangle V_1V_2q, meet in the epipolar line l, in a V-fixed reference frame. The line of movement of V in direction $\mathbf{V} = dV/dt$ meets **C** in a point **K**, independent of **P** and q.

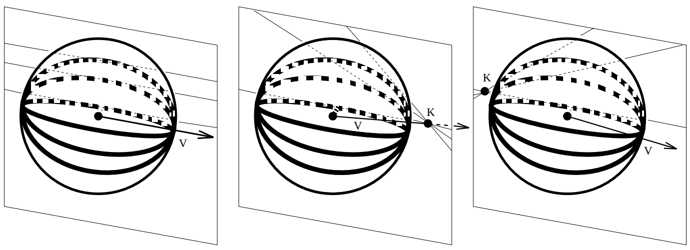

Figure 7: Three cases for different movements of view position V. The left one shows **V** parallel to the screen, the middle and right ones show motion towards it and away. The three lines on each screen are epipolar for the three cases.

set corresponds very well to the skeleton at pixel level, so a low pass filter suffices to smooth it fast to a low-frequency image. For example, the diagonal of a 400×300 image thins with $Gap = 1$ to about 400 points, a small set for filtering. The best choices of filter length and interpolation for smoothing is strongly related to *Gap*. For large *Gap* we connect with higher order splines, such as cubics: with $Gap = 1$, it suffices to use line segments (a spline of degree 1). Fig. 5 compares the result with an edge found using snakes.

Collecting skeleton points as in Section 2.2 also yields data for the tube radii. We smooth both skeleton points and radii, which (for a catheter) should also be smooth.

3 Smooth depth finding

When a camera moves, the image changes more for nearby features than far parts. From this we can compute distances. Human stereo depth perception, best with two views about 6° apart, is a special case, but any pair of viewpoints with non-zero separation suffice.

Several authors[Giblin87, Poston91] have approached theoretically the finding of 3D data from profile data. We follow here the notation of [Poston91] that treated singularities in surface reconstruction from profiles; similar issues arise in reconstructing curves.

A 3D curve S, or the outline of a viewed 3D surface (Fig. 6), projects to a 2D curve on a 2D camera screen. To triangulate for depth, we must identify corresponding points. We use epipolar line correspondence, with unrestricted camera motion; say, from V_1 to V_2.

We have a finite, reasonably uniformly distributed set Σ_1 of points on the first image curve. For each $q \in \Sigma_1$ we find a corresponding point q' on the second curve, from whose direction we can triangulate in the plane P containing the line $\overline{V_1V_2}$ and the unit vector $\mathbf{q_1}$ from V_1 to q. (Since q' is typically not one of the listed nodes Σ_2 of the second curve, we must interpolate using the spline structure.) The corresponding point q' is the intersection of this plane and the second projected curve. The lines $\overline{V_1q}$ and $\overline{V_1q'}$, being coplanar, meet in a point on S. The projection of 3D curve to 2D camera is not spherical projection but camera-screen plane projection. We must express the above *epipolar correspondence* in these terms for implementation.

We need special treatment near **side points**[Poston91] $s \in S$ where S is tangent to the plane through V_1, V_2 and s, where the correspondence becomes geometrically and

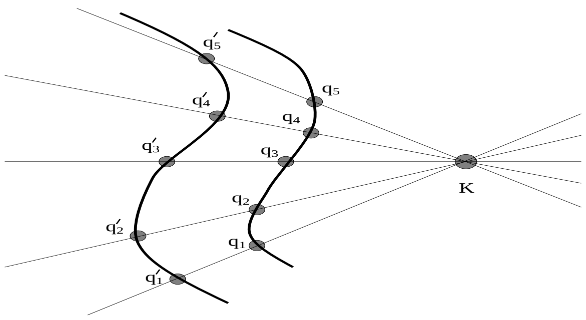

Figure 8: Epipolar lines in the plane $\mathbf{C}$ of Fig. 6 for five pairs of corresponding points.

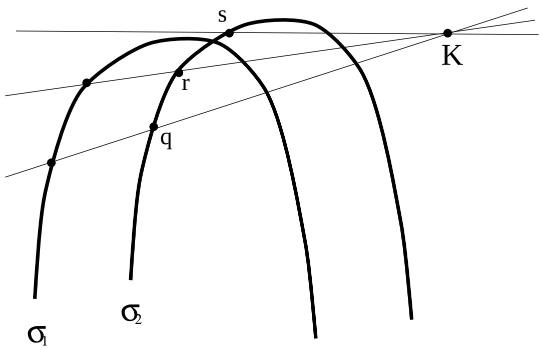

Figure 9: For a point like q on Σ_1, the corresponding point on Σ_2 is the nearest intersection of $\overline{q\mathbf{K}}$ with Σ_2. For points like r or s, this selection rule fails.

numerically unstable. A naive implementation can give absolutely wrong points. With image processing methods we remove singularities and smooth the reconstructed S.

Within the plane $\mathbf{P}$, if $\mathbf{q}_1$ and $\mathbf{q}_2$ are the unit vectors toward points q and q' coplanar with V_1 and V_2, for V_1 and V_2 near enough we can estimate the distance from V_1 to the intersection of the lines through them by[Poston91]

$$\lambda = -\frac{(\mathbf{q_1} \times \mathbf{q_2}) \cdot (\mathbf{q_2} \times (\mathbf{V_2} - \mathbf{V_1}))}{\|\mathbf{q_1} \times \mathbf{q_2}\|^2} \;. \tag{1}$$

3.1 Correspondence

We use three planes: the camera screens in the first and second view positions, and the plane $\mathbf{P}$ through V_1, V_2 and q. A point q in the intersection of the first camera screen and the plane $\mathbf{P}$ corresponds to a point q' in the intersection of the second and $\mathbf{P}$.

To simplify the calculation we project the first screen to the second, or *vice versa*, and get an epipolar line l for each point q, and q' as the (nearest) intersection point of l with the other curve. Fig. 7 shows three cases, corresponding to three different movement of view position. The first shows parallel epipolar lines, the second is epipolar lines with an intersection point on the right (movement toward the camera plane) and the last with it on the left (movement away). Fig. 8 shows the epipolar lines of the plane $\mathbf{C}$ for five pairs of corresponding points, all meeting in $\mathbf{K}$.

In many cases there is more than one point, or none, in the second curve's intersection with l (Fig. 9). Tentatively we select the point which is nearer to q after projecting the second screen on the first screen, but this can require correction (Fig. 9) at side points as intersection points come together.

Fig. 10 shows a good 'nearest point' correspondence close to $s_2 = s_1$ between the curve parameters of Σ_1 and Σ_2, away from the singularity. In the limit as V_1 and V_2 approach each other, the enlargement in Fig. 10 would show the dark points with smoothly changing shape, not the slight jump seen here.

3.2 Singularity removal and smoothing

At this stage we have two major kinds of large error. One is from completely wrong data, usually found near end points; the other is from singularities, that produce large local variation in estimated depth λ. Controlling these avoids singularities on the reconstructed

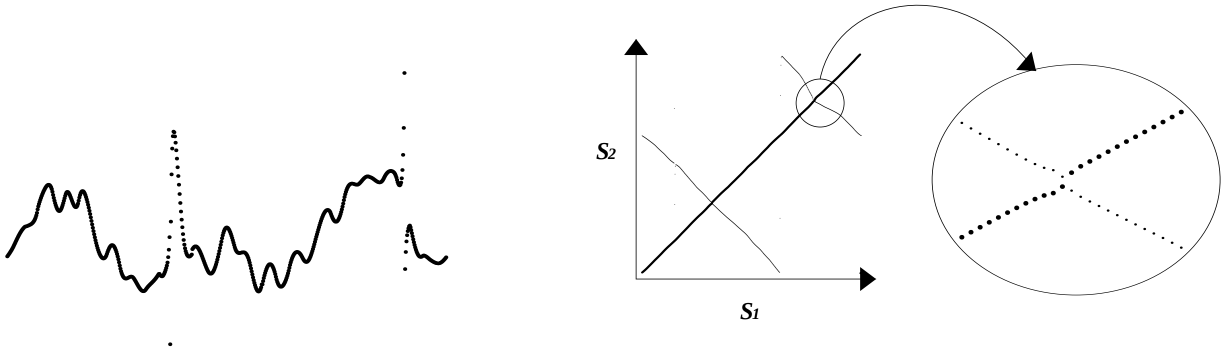

Figure 10: At left is an uncorrected λ curve, singular in two places and unsmooth everywhere. At right are possible matches between parameter s_1 for Σ_1 and s_2 for Σ_2. The thick line shows the subset of correspondences needed, and the thin line is for inappropriate ones. We magnify the intersection of the thick and thin curves, at a singularity.

object. We use a median filter for λ, forcing points with distinct λ to be more like their neighbors and eliminating spikes that are isolated in the area of the filter mask.

The appropriate filter size depends on the size of interval affected by singularities. Here, for instance, we use an odd number M about $\frac{1}{4}$ of the number of points in the curve. For a curve segment with ends, we repeat about $M/2$ points near end points.

We use a smoothing filter as in Section 2.3, this time to smooth the depth estimate λ. At this stage we use an odd filter size about $\frac{1}{8}$ the number of points in a curve segment. We then smooth with use one more lowpass filter, half the size of the previous one. We repeat some boundary points, for the same reasons as with median filters.

4 Experiment with a synthetic catheter

We chose the curved end of a synthetic catheter because it produces a mathematically interesting singularity for most directions of camera movement around it. From two pictures whose difference is $2°$ horizontally (Fig. 11), we found edges with the Sobel operator provided by Matlab. Then we used our distance-wave collision calculation to find their skeletons. We did not thin at this stage, since we needed as many points as possible for better reconstruction. (We could connect thinned points using higher degree splines: this experiment used 1-dimensional splines, which connect points with straight segments.) We had 458 points for the first picture and 462 points for the second. In building the skeletons, we cut if the distance was bigger than 3 and re-connected when ends were nearer than 10, after discarding pieces with length less than 3. Each image in Fig. 11 gave four pieces after cutting the first spline, three after discarding a short fragment, and one after final connection. We used a lowpass filter of length 11 to smooth the skeleton. Skeleton detection using colliding distance waves curves took 0.2935 *sec.* for the first image and 0.3032 *sec.* for the second, on a SparcStation20. With the 458 points for the first image and 462 points for the second, we found the depth λ for each point on the first curve, and repeated 57 boundary points to avoid boundary problems with the median filter (length 115) and the lowpass filter (length 57). To make λ smoother, we used one more lowpass filter of length 29. Finding the correspondence and distance took 0.1340 *sec.*. With the image and edges found (a task we do do not address in this paper, since it depends strongly on the image and we can suggest no improvement on available packages), the total reconstruction took about 0.7307 *sec.* on a SparcStation20.

Of our radius estimates, 92.35% were between 2.5 and 3 pixel widths, which is close

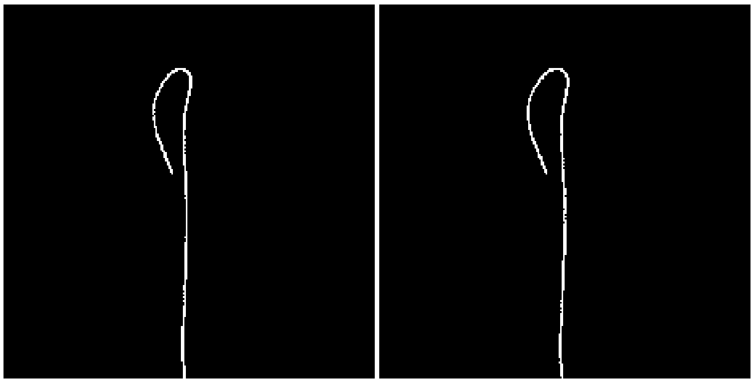

Figure 11: Two parallel snapshots of a synthetic catheter from 2° apart, used as input data for 3D reconstruction.

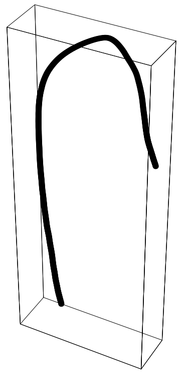

Figure 12: 3D figure reconstructed from the images in Fig. 11.

to constancy: appropriate for the parallel views used. (Perspective would require radius adjustment using the computed depth λ.) The catheter in our data had constant radius, so only the consistency of the radius results is interesting; but the result indicates the potential for detecting arterial stenosis (narrowing) by computed reduction in radius.

Throughout these processes, we reconstructed 3D data points for the catheter from the two snapshots in Fig. 11. To show 3D validity, we give several 3D views. Grey points in Fig. 13 show the reconstruction, white shows the 'true' centre curve. Different view angles give different views of the catheter. The white 'true' curves are used only for this comparison, not in the reconstruction.

5 Conclusions and further research

From two good edge images 2° apart of a narrow tube, we can quickly reconstruct a good 3D curve by wave collision centre-curve finding, appropriate filtering, and epipolar correspondence, usable for arbitrary nearby views. The time cost of filtering, since we apply it only to 1D sets, is minor.

One possible use of this technique is to reconstruct 3D catheter positions as the radiologist rolls the viewpoint. The current less-than-a-second running time of the program when used on catheter edge data may be termed 'on the spot' reconstruction, offering the radiologist a 3D catheter view shaded, or stereo, or with other depth cues as convenient. ('Real time' reconstruction would appear to keep pace with the rolling view.) This information-enhanced view could include also such data as centimeter marks, making it easy to judge (for instance) how long a stent is needed in 3D reality to span a foreshortened separation between two points in an image.

In ongoing work, this provides the basis for the reconstruction of a network of arteries from patient data. It is easily extensible to complexes of tubes such as arteries, with branches and (in projection) crossings. These require multiple views simply because of occlusion, but a range of 20° or 30° will be ample; we will not need the 180° needed for CT reconstruction, with the resulting radiation load and collection time (longer than contrast agent density stays near-constant, without steady continuing injections and their attendant dangers). Where tubes meet in a 2D image, we can distinguish crossing from branching by nearby depth estimates. With this correct topology for the 2D curves, we can complete the 3D reconstruction. By its natural radius estimation, the method is highly suitable for capturing stenosis. Profile information classified as 'non-tube' by the

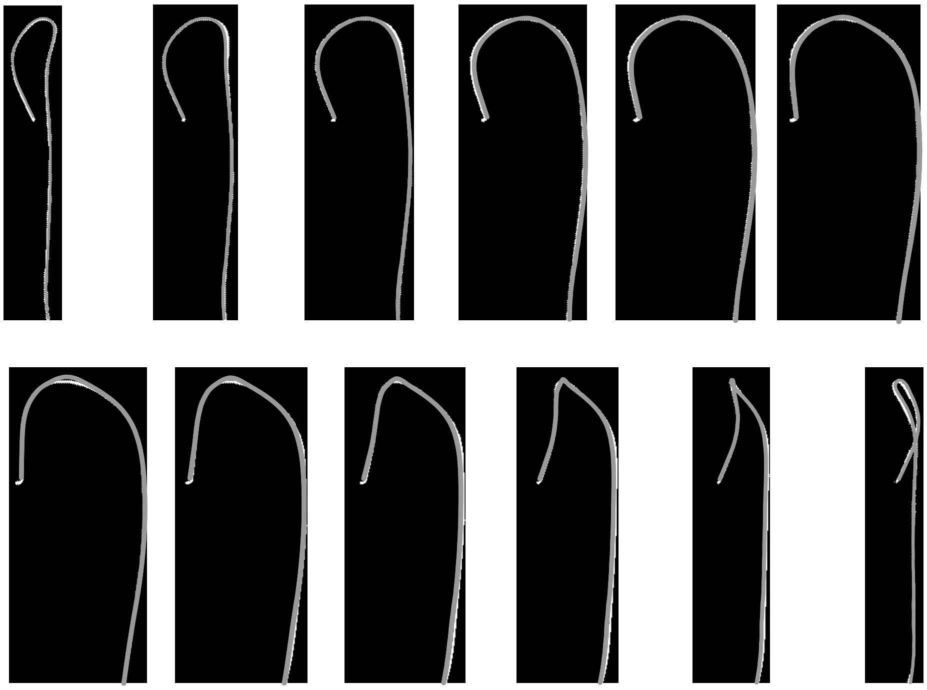

Figure 13: A reconstructed catheter (grey), from several views (in steps of 15°), from the two images in Fig. 11. We overlay it on the corresponding images of the catheter (white), which are *not* used as input for reconstruction. It takes about 0.2935*sec.* to detect the skeleton for the first edge image, about 0.3032 *sec.* for the second. Reconstruction takes about 0.1340 *sec.* So starting from image and (Sobel) edges (which we do not calculate inside our work since there are optimized commercial packages), the total reconstruction takes about 0.7307 *sec.* on a SparcStation20.

absence of wave collisions will provide at least partial information about the 3D geometry of aneurysms and other pathologies, where a tube description of arterial geometry becomes inadequate. In these cases we will use the edge curves directly, to reconstruct ribbons on the 3D surface.

The advantages of this approach to 3D reconstruction are that we require no constraints on camera movement, that by using very nearby views we can image even a moving object like a heart or a bolus of contrast agent, and that the similarity of nearby views makes epipolar correspondence effective even for complex images.

References

[daVinci96] J. Anderson, W. Brody, C. Kriz, Y. Wang, C.K. Chui, Y.Y. Cai, R. Viswanathan & R. Raghavan, daVinci: a vascular catheterization & interventional radiology-based training & patient pretreatment planning simulator, *Society of Cardiovascular and Interventional Radiology 21st Annual Meeting*, Seattle, USA, March 1996.

[Chang92] H D. Chang, T. Poston and K-I. Kim, A closed form 3D self-localization for a mobile robot using a single camera and a rectangular guide-mark, Proceedings, *Second International Conf. on Automation, Robotics, and Computer Vision, (ICARCV '92)*, Singapore, 16-18 September 1992, *CV*-12.3.1 – *CV*-12.3.5.

[Fessler91] J.A. Fessler and A. Macovski, Object-based 3D reconstruction of arterial trees from magnetic resonance angiograms, *IEEE Transactions on Medical Imaging* Vol. 10, No. 1, 25-39, Mar. 1991.

[Giblin87] P.J. Giblin and R. Weiss, Reconstruction of surfaces from profiles, *Proc. 1st Int. Conf. on Computer Vision*, 136-144, London, 1987.

[Kass88] M. Kass, A. Witkin, and D. Terzopoulos, Snakes: Active Contour Models, *International Journal of Computer Vision*, pp. 321-331, 1988.

[Kitamura88] K. Kitamura, J. Tobis, and J. Sklansky, Estimating the transverse areas of coronary arteries from biplane angiograms, *IEEE Transactions on Medical Imaging* Vol. 7, No. 3, pp. 173-187, Sept, 1988.

[Kutka96] R. Kutka and S. Stier, Extraction of Line Properties Based on Direction Fields, *IEEE Transactions on Medical Imaging*, Vol. 15, No. 1, February 1996.

[Lawton94] W. Lawton, Mathematical Methods for Active Geometry Proceedings *Int. Conf. Computer Aided Geometric Design*, Penang, Malaysia, July 1994.

[Mikheev94] A. Mikheev, M. Nozik and J. Rubinstein, Computation of Offset Curves by the Huygens Principle, *Computer Graphics Forum*, 249-252, Vol. 13 No. 4 1994.

[Nguyen94] T.V. Nguyen and J. Sklansky, Reconstructing the 3-D Medical Axes of Coronary Arteries in Single-View Cineangiograms, *IEEE Transactions on Medical Imaging*, 61-73, Vol. 13, No. 1 1994.

[Brien94] J.F. O'Brien and N.F. Ezquerra, Automated Segmentation of Coronary Vessels in Angiographic Image Sequences Utilizing Temporal, Spatial and Structural Constraints, *SPIE*, Vol. 2357, pp. 25-37, 1994.

[Poston91] T. Poston and C.-N. Lee, Singularities in Shape-from-Contour Estimation, *proceedings, First Korea-Japan Joint Conference on Computer Vision*, 152–158, 1991.

[Parker87] D.L. Parker, D.L. Pope, R.V. Bree, and H.W. Marshall, Three dimensional reconstruction of moving arterial beds from digital subtraction angiography, *Comput. and Bio. Res.* Vol. 20, 166-185, 1987.

[Sunka95] M. Sonka, M.D. Winniford, and S.M. Collins, Robust Simultaneous Detection of Coronary Borders in Complex Images, *IEEE Transactions on Medical Imaging*, Vol. 14, No. 1, March, 1995.

[Sun95] Y. Sun, R.J. Lucariello, and S.A. Chiaramida, Directional Low-Pass Filtering for Improved Accuracy and Reproducibility of Stenosis Quantification in Coronary Arteriograms, *IEEE Transactions on Medical Imaging*, Vol. 14, No. 2, June 1995.

[Wahle95] A. Wahle, E. Wellnhofer, I. Mugaragu, H.U. Sauer, Helmut Oswald, and Eckart Fleck, Assessment of Diffuse Coronary Artery Disease by Quantitative Analysis of Coronary Morphology Based upon 3-D Reconstruction from Biplane Angiograms, *IEEE Transactions on Medical Imaging*, Vol. 14, No. 2, June 1995.