# Segmentation of Parchment Scrolls for Virtual Unrolling

Oksana Samko
O.Samko@cs.cardiff.ac.uk
Yu-Kun Lai
Yukun.Lai@cs.cardiff.ac.uk
David Marshall
Dave.Marshall@cs.cardiff.ac.uk
Paul L. Rosin
Paul.Rosin@cs.cardiff.ac.uk

School of Computer Science & Informatics
Cardiff University
Cardiff, UK

**Abstract**

In this paper we introduce a framework for the segmentation of scanned scrolled parchments, based on a novel graph cut based approach with an additional shape prior, in combination with anisotropic diffusion and geometry-constrained postprocessing. This problem has not been investigated by the computer vision community properly yet due to the parchment scanning technology novelty, and is extremely important for effective data recovery from historical scrolled documents whose content is inaccessible due to the deterioration of the parchment. To date, parchment segmentation has required user interaction, which is very time consuming for such data. We demonstrate with real examples how our algorithm is able to solve the major problem for scrolled parchment analysis, namely segment connected layers, and process the data without user interaction.

## 1 Introduction

Digital document restoration has become an increasingly active area of research over the last few years. Brown and Seales in [3] proposed a general de-skewing algorithm for arbitrary warped documents based on 3D shape. Doncescu et al. in [7] reported a similar method, where a laser projector is used to project a 2D light network on the document surface to capture 3D shape, and then 2D distortions of the surface are corrected with a two-pass mesh de-warping algorithm. Cao et al. in [5] presented an algorithm to rectify the warping of a bound document image: they built a general cylindrical model, and then used the skeleton of horizontal text lines in the image to estimate the model parameters. Pilu in [16] introduced a novel method for distorted document restoration which is based on physical modelling of paper deformation with an applicable surface. Yamashita et al. in [22] introduced a shape reconstruction method by using a two-camera stereo vision system. Except for Cao's work [5] and a few others [24, 25], most of the current approaches require special setup to assist in 3D shape discovery. Also they can only handle smooth distortion of the image surfaces.

The most related work was undertaken by the EDUCE project [10], which attempted to read a scrolled document from a 3D scan. However, very few results on document unrolling


BMVC 2011 http://dx.doi.org/10.5244/C.25.37

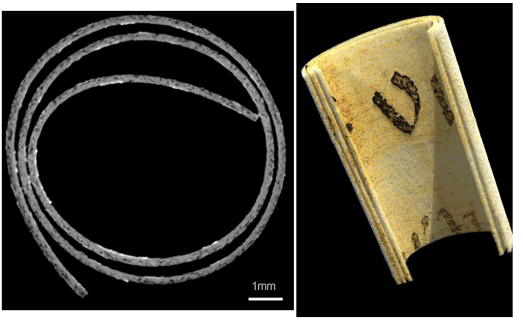


Figure 1: A small cut sample from a historical parchment scanned with the high definition XMT scanner. Left: cross-sectional tomographic slice with contrast enhanced to show ink on the surface of the parchment. Right: volume rendered cutaway view with pseudo-coloring.

have been reported [13, 19]. The results were only shown on small contrived samples and would not scale up to real parchment with many layers, which are tightly stuck together. The segmentation stage of that work was performed semi-manually [13]. Apart from that, no more results on virtual parchment unrolling have been reported. X-ray scanning technology is typically deployed for medical data analysis [4, 8, 23], except boundary extraction is not of such concern in such applications as for the parchment ink recovery task.

There is a critical need to access the valuable information in historical scrolls that cannot be read by conventional means. In some cases, their physical deterioration is at such an advanced state that any attempt to unravel the document manually would cause catastrophic fragmentation, destroying the internal information. Use of X-ray microtomography, a new direction in digital document analysis [11], provides a digital copy of a scrolled parchment as a 3D volumetric object, see Fig. 1. Clearly, the parchment layers need to be separated to perform a virtual unrolling and apply further digital document restoration methods.

Parchment is essentially animal skin and therefore has an irregular sponge-like structure, also its thickness may vary across a document surface. As a result of degradation over time, parchment may convert to its entropic form, gelatin, making the boundary between its layers difficult to observe even with the human eye. Image noise, low contrast and scanning artifacts may lead to even more indistinct parchment structure boundaries. As can be seen in Fig. 2, it would be difficult to handle the parchment segmentation task satisfactorily. A general algorithm can destroy damaged areas because of parchment's latent texture (oversegment), and not split tightly connected layers with zero gradient (undersegment) at the same time. The topology of parchment smoothly changes from slice to slice, but can differ significantly across the whole scroll. The parchment ink thickness is only a few voxels deep (represented by the light pixels close to the parchment boundary), so it is very important to carefully process the boundary to avoid losing important information due to incorrect segmentation.

This paper describes the segmentation process utilised in our system for virtual unrolling of parchments, whose goal is to extract the scrolled parchment surface from the volumetric data, and to separate its touching layers. We exploit both geometric and pixel intensity features from the data. Our algorithm consists of three main steps: data filtering, segmentation, and postprocessing using geometric constrains. The first step, anisotropic filtering, makes the parchment structure more homogeneous, simultaneously preserving the parchment's layer boundaries. At the next step we introduce the main segmentation routine, based on Graph

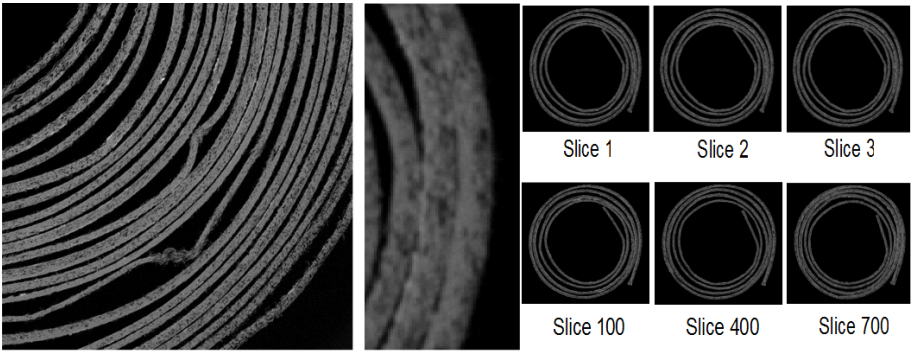


Figure 2: Parchment data examples: fragments and several slices of the same scroll

Cut [1], with a novel shape prior optimisation. Here we incorporate parchment layer thickness information together with the traditional pixel intensity. This makes the segmentation more robust, however a few very tight connections between layers may still be retained. The reason for such incorrect connections is that the local boundary features may not exist, or it may be difficult to detect them reliably using the global optimisation. Therefore instead of involving time-consuming user interaction which also requires great accuracy from the user, in the last step of our algorithm we employ local geometric constraints to automatically separate such connections. We evaluate our algorithm by applying it to segment three different parchment data sets, which vary in the parchment's condition, size and number of layers. Our experiments indicated that we were able to fully separate these parchments' layers, outperforming traditional segmentation methods both visually and numerically.

## 2 Parchment segmentation framework

We treat the data, volumetric images from an X-ray scanner, as a set of slices, as in Fig. 1, because the parchment volumetric model is very large and constructing volumetric segmentation will take up large amounts of time and resources. Using the fact that the parchment structure is only changing slightly from slice to slice, we use the segmentation for one slice as the initialisation for the next slice in the set. Our algorithm consists of filtering, segmentation itself, and postprocessing. We describe these steps below.

### 2.1 Data filtering

We use Coherence-Enhancing Diffusion filtering (CED) as a segmentation preprocessing due to its property of completion of interrupted lines [20]. CED uses a nonlinear diffusion process whose diffusion tensor allows anisotropic smoothing by acting mainly along the preferred structure direction. This so-called coherence orientation is determined by the eigenvector of the structure tensor with the smallest eigenvalue [20]. Using the CED filter enables us to preserve the topology of the parchment layers, while the internal variation caused by the parchment's sponge-like structure is diminished.

A gray-scale image $u(x,y)$ can be treated as a surface corresponding to the mass concentration (the grey level). The equation describing the diffusion of the mass concentration

is

$$d_t u = div(D \cdot \bigtriangledown u), \tag{1}$$

where $\bigtriangledown u$ is the concentration gradient, and $D$ is the diffusion tensor, a function of local image structure. As in [20], we define $D$ using the regularised structure tensor matrix $J_\rho(\bigtriangledown u_\tau)$:

$$J_\rho(\bigtriangledown u_\tau) = N_\rho * (\bigtriangledown u_\tau \otimes \bigtriangledown u_\tau), \tag{2}$$

where $\rho$ is the integration scale, and $u_\tau$ is the regularised image of $u$ obtained by convolution with a Gaussian $N_\tau(x,y) = (2\pi\tau^2)^{-1}\exp(-\frac{x^2+y^2}{2\tau^2})$. The eigenvectors of $J_\rho$ give the preferred local orientations, and the corresponding eigenvalues denote the local contrast along these directions.

For a given parchment image $u(x,y)$, we have three parameters to define: time step $t$, local scale $\tau$ and integration scale $\rho$. We set $\tau = 1.5$ for our framework; this small value gives us a uniform blurring over the whole object. The integration scale $\rho$ reflects the characteristic size of the texture and is defined individually for each parchment. Correctly adjusted, it plays an important role in "reconstructing" a parchment's damaged areas, whilst preserving boundaries. Small $\rho$ (1,2) does not lead to the visually dominant coherence orientation and gives a rough approximation of the parchment's boundary. For the relatively small parchment data we got good results with $\rho = 4$. Generally larger parchments require a bigger $\rho$ value; but excessive $\rho$ values may deform the parchment boundary into a shapeless mass. Finally, the time step $t$ is also defined for each parchment individually. Larger $t$ values produce increased blurring. This parameter depends on the image resolution and parchment condition. If the image resolution is small, a large time scale may dissolve the parchment boundary. If the parchment condition is poor (damaged, many pores), large $t$ is preferable. In our experiments we set $t = 4$ for the small parchment, and $t = 12$ for the damaged parchment. Fig. 3 shows the example of the original parchment image, and the results of applying CED with different parameters to it. Note that the effect of a large $\rho$ is most significant in very damaged areas, which contain boundaries that we also need to preserve.

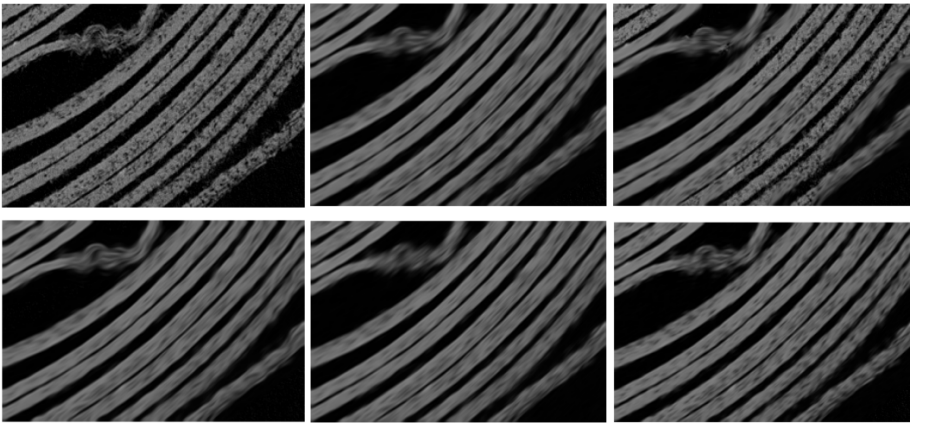

Figure 3: From left to right: original parchment fragment; the CED filtered images with parameters ($\tau$, $\rho$, $t$): (1.5, 10, 12) - optimal, (10, 10, 12) - nonuniform blurring, (1.5, 4, 12) - small $\rho$, (1.5, 20, 12) - large $\rho$, (1.5, 10, 4) - small $t$

## 2.2 Graph Cut with shape prior

We use a Graph Cut based optimisation for our main segmentation step. Boykov and Jolly [1] formulated Graph Cut segmentation as a binary labelling problem, i.e. each pixel in the image has to be assigned a label from the label set $\{0,1\}$, where 0 and 1 stand for the background and the object, respectively. The labelling corresponding to the minimum energy is chosen as the solution.

The Graph Cut energy is formulated as a function of the pixel assignment:

$$E(f) = \sum_{p \in P} D_p(f_p) + \lambda \sum_{(p,q) \in N} V_{pq}(f_p, f_q) \tag{3}$$

Here $P$ is the set of image pixels, $f_p$ is the binary label assigned to pixel $p$, $N$ is the set of neighbourhood pixel pairs, $D_p$ is the data term, $V_{pq}$ is the smoothness (boundary) term for two neighbouring pixels, parameter $\lambda \geq 0$ specifies the relative importance between these two terms. To avoid user interaction, we initialise these models through GMM learning. We take $D_p(f_p)$ as the negative log likelihoods of the constructed background/foreground models. The smoothness term counts the weighted sum of discontinuities in $f$:

$$V_{pq}(f_p, f_q) = w_{pq} \varsigma(f_p, f_q) = \exp\left[-\frac{(I_p - I_q)^2}{2\sigma^2}\right] \frac{1}{dist(p,q)} \varsigma(f_p, f_q) \tag{4}$$

Here $\varsigma(f_p, f_q)$ is 0 if $f_p = f_q$ and 1 otherwise, $w_{pq}$ is the weight, $I_p$ is the intensity of pixel $p$, and $\sigma$ is the intensity variance. To minimise the energy from Eq. 3, Boykov and Jolly in [1] used the minimum cut on the constructed graph. Later in [2], an efficient algorithm for computing the minimum cut was presented, which we apply in our framework.

Fig. 4 (left) shows the result of applying Graph Cut to the parchment data from Fig. 1. It can be seen that after the segmentation we still have many interlayer connections. Incorporating prior shape information, based on the parchment thickness, should make the segmentation more robust. This is performed as follows: at the initialisation stage, we set region $U$ as a dilation of the foreground, and define its neighbourhood pixel pairs as $N_u$. To get local thickness we separate the image into background and foreground, detect parchment boundaries, and calculate distances $d_x$ for each pixel $x$ from $U$ to its closest boundary, taking into account the edge's orientation. Define $m$ as the parchment thickness parameter, which can be estimated as the mean distance between opposite boundaries of parchment layers. Using these settings, we define our shape prior energy, and rewrite the energy function from Eq. 3 as:

$$E(f) = \sum_{p \in P} D_p(f_p) + \lambda \sum_{(p,q) \in N} w_{pq} \varsigma(f_p, f_q) + \mu \sum_{(p,q) \in N_u} s_{pq} \rho(f_p, f_q) \tag{5}$$

Here $\rho(f_p, f_q)$ is 0 if $f_p = f_q$ and 1 otherwise, $\mu$ is the shape parameter to control the relative importance of the shape term, $s_{pq}$ is the shape weight. In our parameter settings we always bias intensity over shape due to the parchment structure and irregularity in thickness. The shape weight controls the layer thickness, we use the following form:

$$s_{pq} = \exp\left[-\frac{\left(\frac{d_p + d_q}{2} - km\right)^2}{2\sigma_u^2}\right] \frac{1}{dist(p,q)} \tag{6}$$

where $k$ is the parchment layer counter, $k = 1 \dots M$, $M = max_{d_x} \lceil \frac{d_x}{m} \rceil$, $\sigma_u$ is a parameter which we estimate as $\sigma_u \approx m+1$. To define $k$, we calculate $\frac{d_p+d_q}{2}$. If $0 < \frac{d_p+d_q}{2} < \frac{3m}{2}$, then $k = 1$, and generally $\frac{(2k-1)m}{2} < \frac{d_p+d_q}{2} < \frac{(2k+1)m}{2}$ for $k = 2 \dots M$.

The energy based on our shape prior will be low for the neighbouring pixels $p$, $q$ which are close to the estimated parchment boundary and have different labels. Assuming that the shape prior energy is 0 outside $U$, and considering all the above, we can conclude that our shape term $S_{pq}(f_p, f_q) = s_{pq}\rho(f_p, f_q)$ satisfies the following property

$$S_{pq}(0,0) + S_{pq}(1,1) \leq S_{pq}(1,0) + S_{pq}(0,1) \tag{7}$$

and therefore according to [12] can be minimised using Graph Cut. Fig. 4 (right) illustrates how our Graph Cut with the shape prior works. We get much less interlayer connections in comparison with the original Graph Cut, did not get any extra holes inside the parchment layers, and retained the ink at the surface.

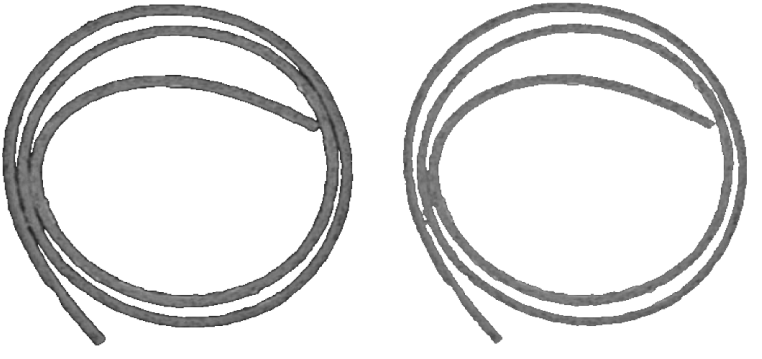

Figure 4: Segmentation using Graph Cut (left), and Graph Cut with shape prior (right)

## 2.3 Postprocessing

Our segmentation result in the right part of Fig. 4 indicates that there are still some false connections between the parchment layers. The problem appears in the areas which are stuck together such that it is impossible to even see the boundary between them, as also demonstrated in Fig. 5. Many popular segmentation methods rely on user corrections during the segmentation process [9, 17], but this is time consuming, and they still require the presence of boundary features. We use a purely geometrical approach, based on the parchment's local features, to separate such areas.

Figure 5: Examples of segmentation containing false connections between parchment layers

The idea is to recreate a missing boundary from the preserved boundary of the opposite side of the same layer, otherwise from the closest preserved boundary. The basic stages of our postprocessing algorithm are:

1. Detect a layer's false connections using existing boundary information. We extract the parchment boundary, and apply simple rules based on the patterns in a moving $7 \times 7$ window to detect and disconnect these connections.

2. Endpoint linking. For each detected endpoint, we find its pair and the opposite side of their layer. Between these endpoints we construct links parallel to the opposite parchment side, adjusting it if necessary to the local parchment thickness. Next we similarly reconstruct the boundary of the joined layer using the closest (reconstructed) boundary, maintaining the distance. We perform analysis on a layer by layer basis, starting from the outermost layer.

3. If necessary, adjust the obtained boundary to avoid connections with the previous slice.

Note that we give priority to the inner parchment surface, because this side of the parchment contains ink. It is not necessary to perform all these steps for all slices; since the parchment topology changes smoothly, we postprocess the first slice using the described algorithm, adjust the result for the second slice, and so on. Also note that depending on the parchment condition, postprocessing may not be required. Fig. 6 illustrates the stages of our postprocessing algorithm.

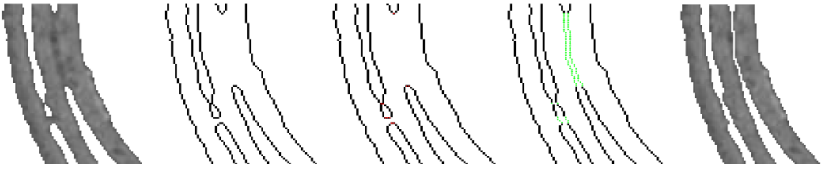

Figure 6: From left to right: segmented parchment fragment; its contour; endpoints detection (red); links construction (green); postprocessed fragment

## 3 Experiments

We demonstrate how to apply our framework to segment three real parchment examples, which range in size and complexity. The data were obtained through tomographic development in the School of Medicine and Dentistry at QMUL [6].

The size of the first parchment (shown in Fig. 1, 4) is $430 \times 430 \times 708$. It has fairly good condition, excluding a few areas where layers are stuck together. Fig. 7 illustrates the stages of applying our algorithm to this data. Here we set $m = 12$, $\lambda = 0.45$, $\mu = 0.3$. The segmentation results are quite similar for all slices of the same scroll, so our pictures represent the algorithm's performance for the whole data set.

Fig. 8 demonstrates segmentation results with snakes [21], the original Graph Cut, and our algorithm. Since the original slice does not have tight connections our Graph Cut with the shape prior was able to separate its layers without any postprocessing, while the original Graph Cut and snakes both failed. Although we give more penalty to the parchment boundaries, the result is not oversegmented because the ink is always considered as foreground.

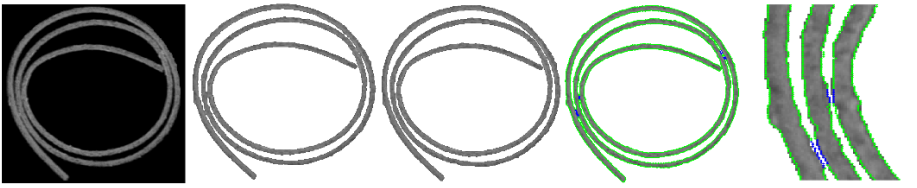

Figure 7: From left to right: original slice; segmented by Graph Cut with the shape prior; final result; postprocessed slice and contour of its neighbour slice (green, blue marks postprocessing areas); scaled fragment

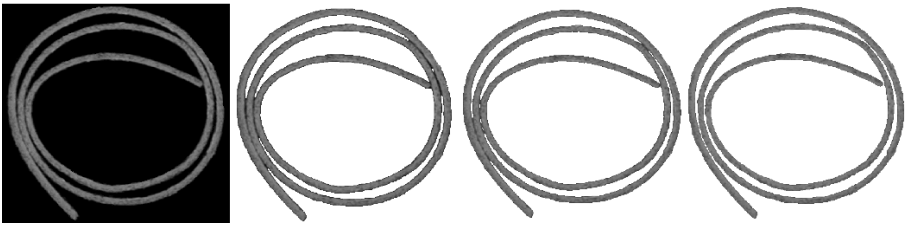

Figure 8: From left to right: original slice; segmented by snakes; segmented by Graph Cut; segmented by our method

The next parchment example is more complicated, more damaged and has more topological variation. It consists of 708 slices, $530 \times 530$ pixels each. We set $m = 11$, $\lambda = 0.5$, $\mu = 0.2$ for this data, and due to the variation in layer thickness we reduce our shape penalty to have less effect on very thick layers. Fig. 9 demonstrates our segmentation, and the result obtained with the original Graph Cut. With Graph Cut, we got undersegmentation (joined layers) simultaneously with oversegmentation (a large hole at the bottom left) even after the filtering. Our Graph Cut with shape prior was able to extract the parchment without holes, by giving them more penalty in comparison to the parchment boundary. Also we were able to separate most of the layers without postprocessing. In comparison to the previous example, this parchment needed more postprocessing to completely separate its layers.

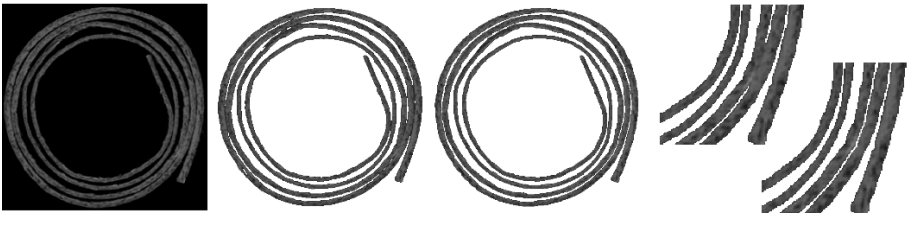

Figure 9: From left to right: original slice; segmented by Graph Cut; segmented by our method; a fragment before and after postprocessing

Finally we consider a large scroll of size $1702 \times 1732 \times 423$, containing two interleaved parchments, shown in Fig. 10. With this example we demonstrate how our method can help to process parchment with highly damaged layers. For the previous (less damaged) examples, we detected the boundaries of our initialised models and calculated the distances $d_x$. In this case we do not have a solid boundary in some areas, see the top middle of Fig. 10.

| | Example 1 | | | Example 2 | | | Example 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Our method** | **Graph Cut** | **Snakes** | **Our method** | **Graph Cut** | **Snakes** | **Our method** | **Graph Cut** | **Snakes** |
| PRI | 0.9445 | 0.5887 | 0.5893 | 0.9592 | 0.6414 | 0.6415 | 0.9073 | 0.5953 | 0.6142 |
| VI | 0.3182 | 1.6193 | 1.5772 | 0.2664 | 1.3628 | 1.3605 | 0.7020 | 3.0737 | 2.7298 |
| P | 0.9911 | 0.9695 | 0.9518 | 0.9687 | 0.9452 | 0.9353 | 0.8942 | 0.8855 | 0.8758 |
| R | 0.9634 | 0.9374 | 0.9385 | 0.9402 | 0.9372 | 0.9309 | 0.9585 | 0.9446 | 0.9328 |
| F | 0.9771 | 0.9532 | 0.9436 | 0.9542 | 0.9412 | 0.9331 | 0.9252 | 0.9141 | 0.9034 |

Table 1: Numerical evaluation scores

The third layer there is very weak even after the filtering to detect its boundary with the initialisation. Therefore we “reconstruct" a boundary in the following manner: we dilate the image to fully include the weak layer, and define the obtained region as $U$. After that we assign the parchment boundaries as one pixel away from the obtained mask boundary, and calculate distances $d_x$ to these “reconstructed" boundaries. As a result we get more joined layers which we separate using our shape prior, but also we save the weak boundaries. The right part of Fig. 10 shows a fully segmented slice of the scroll. The parameters here are $m = 12$, $\lambda = 0.4$, $\mu = 0.35$. Our Graph Cut with the shape prior was able to separate most of the data, we needed only minimal (automatic) postprocessing for the areas with very tight and thin layers.

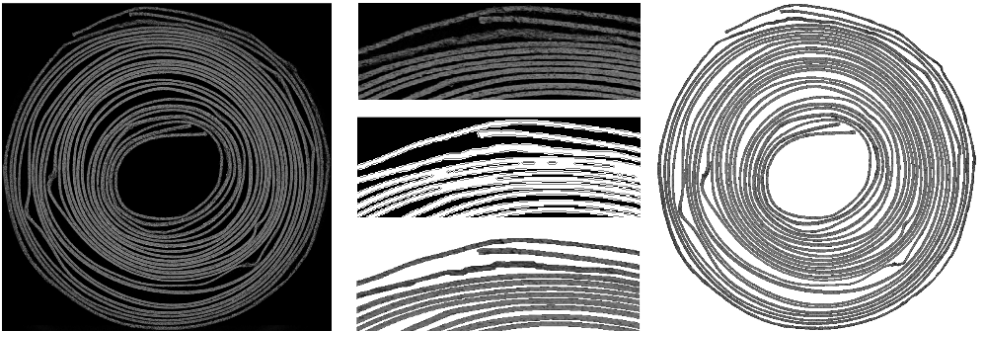

Figure 10: From left to right: original slice; fragments: original, thresholded with marked assigned boundary (gray), result; segmented by our method

To provide a numerical evaluation of our algorithm, we use multiple benchmark criteria: Probabilistic Rand Index (PRI) [18], Variation of Information (VI) [15], as well as boundary-based precision-recall framework measures Precision (P), Recall (R), and F-measure (F) [14]. We manually segment an image from each set to obtain ground truth, perform a connected component labelling on the segmented images (on both foreground and background), and compare the obtained labellings so as to emphasise the importance of correct topology. The results are presented in Table 1. The results show that our proposed method obtained high scores even for complicated data, and outperforms both snakes and standard Graph Cut.

# 4 Conclusion

We have presented a novel method for parchment segmentation which will subsequently be used for virtual unrolling and visualisation of the parchment. Technologies for the effective scanning of parchment are under active investigation, creating a demand for methods of analysing scanned scrolls. Our algorithm is the first attempt to solve the segmentation problem automatically for such data. The presented method does not require user interaction and incorporates parchment thickness information to separate the layers. We illustrated the performance of our algorithm with three different real scrolls, and were able to process very damaged areas and tightly stuck together layers. Our algorithm may be useful for medical data applications, such us vessel extraction. In our future work we plan to involve local parchment thickness in the Graph Cut shape prior to make it more robust for the data with very large thickness variation.

# References

[1] Y. Boykov and M.P. Jolly. Interactive graph cuts for optimal boundary and region segmentation. In *Proceedings of ICCV*, volume 1, pages 105–112, 2001.

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimisation in vision. *IEEE Trans. PAMI*, 26(9):1124–1137, 2004.

[3] M.S. Brown and W.B. Seales. Document restoration using 3d shape: a general deskewing algorithm for arbitrarily warped documents. *Proceedings of ICCV*, 2:367–374, 2001.

[4] P. Campadelli, E. Casiraghi, and G. Lombardi. Automatic liver segmentation from abdominal CT scans. In *Proceedings of the 14th International Conference on Image Analysis and Processing*, ICIAP '07, pages 731–736, 2007.

[5] H. Cao, X. Ding, and C. Liu. A cylindrical model to rectify the bound document image. *Proceedings of ICCV*, 2:228–233, 2003.

[6] G.R. Davis and J.C. Elliott. High definition X-ray microtomography using a conventional impact X-ray source. *J. Phys. IV*, 104:131–134, 2003.

[7] A. Doncescu, A. Bouju, and V. Quillet. Former books digital processing: image warping. *Proc. International Workshop Document Image Analysis*, pages 5–9, 1997.

[8] B. Fischl, D.H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, A. Montillo, N. Markis, B. Rosen, and A.M. Dale. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33(3):341–355, 2002.

[9] L. Grady. Random walks for image segmentation. *IEEE Trans. PAMI*, 28(11):1768–1783, 2006.

[10] http://viscenter.com/educe. Enchanced digital unwrapping concervation exploration (EDUCE) project.

[11] C.J. Kennedy and T.J. Wess. The use of X-ray scattering to analyse parchment structure and degradation. *Physical Techniques in the Study of Art, Archaeology and Cultural Heritage*, (1):151–172, 2006.

[12] V. Kolmogorov and R. Zabih. What energy function can be minimised via graph cuts? *IEEE Trans. PAMI*, 26(2):147–159, 2004.

[13] Y. Lin and W.B. Seales. Opaque document imaging: buildingimages of inaccessible texts. *Proc. ICCV*, 1:662–669, 2005.

[14] D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. PAMI*, 26:530–549, 2004.

[15] M. Meila. Comparing clusterings: an axiomatic view. In *Proc. of ICML*, pages 577–584, 2005.

[16] M. Pilu. Undoing page curl distortion using applicable surfaces. *Proc. of CVPR*, 1: 67–72, 2001.

[17] C. Rother, V. Kolmogorov, and A. Blake. Grabcut-interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[18] A. Savitzky and M.J.E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, (36(8)):1627–1639, 1964.

[19] W.B. Seales and Y. Lin. Digital restoration using volumetric scanning. *Proc. joint ACM/IEEE Conference on Digital Libraries*, pages 117–124, 2004.

[20] J. Weickert. Coherence-enhancing diffusion filtering. *Journal of Computer Vision*, 31: 111–127, 1999.

[21] C. Xu and J.L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Trans. Image Processing*, 7(3):359–369, 1998.

[22] A. Yamashita, A. Kawarago, T. Kaneko, and K.T. Miura. Shape reconstruction and image restoration for non-flat surfaces of documents with a stereo vision system. *Proceedings of ICPR*, pages 482–485, 2004.

[23] P.A. Yushkevich, J. Piven, C.H. Cody Hazlett, C. R. Gimpel Smith, C.S. Ho, J.C. Gee, and G. Gerig B. User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. *NeuroImage*, 31:1116–1128, 2006.

[24] Z. Zhang and C.L. Tan. Correcting document image warping based on regression of curved text lines. *Proceedings of ICDAR*, pages 589–593, 2003.

[25] Z. Zhang, C.L. Tan, and L.Y. Fan. Restoration of curved document images through 3D shape modeling. *Proceedings of CVPR*, 1:10–16, 2003.