Segmenting Gene Expression Patterns of Early-stage Drosophila Embryos

Min-Yu Huang[1,6], Oliver Rübel[1,2,3,6], Gunther H. Weber[3], Cris L. Luengo Hendriks[4,6], Mark D. Biggin[5,6], Hans Hagen[2], and Bernd Hamann[1,3,6]

[1] Institute for Data Analysis and Visualization, University of California, Davis, 1 Shields Avenue, Davis CA 95616, USA
{myhuang,oruebel,bhamann}@ucdavis.edu
[2] International Research Training Group "Visualization of Large and Unstructured Data Sets," University of Kaiserslautern, Germany
hagen@informatik.uni-kl.de
[3] Visualiztion Group, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley CA 94720, USA ghweber@lbl.gov
[4] Life Sciences Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley CA 94720, USA clluengo@lbl.gov
[5] Genomics Divisions, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley CA 94720, USA MDBiggin@lbl.gov
[6] Berkeley Drosophila Transcription Network Project, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley CA 94720, USA
http://bdtnp.lbl.gov/Fly-Net/

Summary. To make possible a more rigorous understanding of animal gene regulatory networks, the *Berkeley Drosophila Transcription Network Project* (BDTNP) has developed a suite of methods that support quantitative, computational analysis of three-dimensional (3D) gene expression patterns with cellular resolution in early *Drosophila* embryos.

Defining the pattern of gene expression is an essential step toward further analysis in order to derive knowledge about the characteristics of gene expression patterns and to identify and model gene inter-relationships. To address this challenging task we have developed an integrated, interactive approach toward pattern segmentation. Here, we introduce a ridge-detection-based 3D gene expression pattern segmentation algorithm. We compare this algorithm to common 2D pattern segmentation methods, such as thresholding and edged-detection-based methods, which we have adapted to 3D pattern segmentation. We show that such automatic strategies can be improved to obtain better segmentation results by user interaction and additional post-processing steps.

Key words: three-dimensional gene expression, pattern segmentation, gene expression pattern, ridge detection, edge detection, thresholding

1 Introduction

Intricate spatial and temporal patterns of gene expression are responsible for determining the shape of a developing animal embryo. Research of these patterns is typically based on visual inspection or computer-assisted analysis of two-dimensional (2D) photomicrographic images. Animal embryos comprise dynamic 3D arrays of cells, however, and thus analysis of 2D images cannot capture the full complexity of a developing embryo. To overcome this challenge, the BDTNP has developed image processing methods from 3D image data [7][3] to extract information about gene expression from imaging data using early *Drosophila melanogaster* embryos as model organisms. Stacks of confocal images of blastoderm stage *Drosophila* embryos are converted into matrices specifying the position of individual nuclei and the amount of mRNA or protein (expression levels) of genes around each nucleus. The resulting novel datasets, termed *PointClouds*, promise to be an invaluable resource for studying animal development.

The purpose of the data generated by the BDTNP is to understand the gene regulatory network. A subset of genes, the so-called transcription factors, regulate the expression of all genes. That is, they cause a gene to be expressed or not in a particular cell. Each of these transcription factors regulates, and is regulated by, other transcription factors. These interactions form the transcription network. This network in combination with the initial conditions in the egg given by maternally transcribed factors yields the expression patterns we analyze here. Modeling the regulatory network is a common approach to gain understanding of the integrated behavior of genes and their regulatory interactions. Despite its limitations, the binary gene regulatory model, which takes the gene to be either on or off in each cell, allows examination and creation of very large systems (several thousand genes) and is mathematically the most tractable network modeling approach. However, the algorithm chosen to define the binary pattern of the genes' expression will influence the results obtained from this network model. Therefore we propose a segmentation algorithm that accurately follows pattern edges, as given by locations of rapid change in expression level, rather than a globally chosen threshold value. Additionally, binarization yields a high level of abstraction for gene expression patterns. Based on such binary gene expression patterns, dedicated user interactions for similarity-based pattern queries can be defined to allow searches for patterns that have, for example, seven stripes or for genes that are expressed in specific regions. And, concentrating only on those cells that show expression of a specific gene or gene combination, more complex analysis becomes possible.

In the field of image processing(IP), there are several approaches which can segment grey-value images into binary patterns. Many of these segmentation algorithms require data to be on a rectangular grid or on a 2D manifold. However, even in the very simple blastoderm *Drosophila* embryos, which are mostly a single layer of cells, our *PointCloud* expression data cannot be seg-

mented by these approaches because the locations of the nuclei form irregular grids and do not necessarily form a 2D manifold, especially near the posterior pole. Furthermore, we require that our segmentation algorithms be applicable to later-stage embryos, where the cells are packed in 3D volumes. Since we cannot record expression values with an absolute metric, an algorithm which will not be affected by scaling is desired here. Intrinsic properties such as local maxima, local minima, and inflection points are scale-invariant and thus particularily suitable to be used to define the gene expression pattern.

A gene expression pattern can be defined as regions with high expression values enclosed by loci of inflection points. To define the pattern of a gene in 3D space, we have developed a segmentation method based on ridge region detection. Using a fully automatic approach does not always produce results of sufficient quality. In order to make use of biologists' knowledge, we have developed an integrated interactive approach toward pattern segmentation that significantly improves the accuracy of the final results.

Thresholding and edge detection are two common techniques used to specify gene expression patterns. We compare our method to thresholding and edge-detection-based segmentation methods, which we have extended and applied to define 3D gene expression patterns.

2 Related Work

Until last year, scientists did not have access to 3D gene expression data at cellular resolution for a whole multi-cellular organism. Kumar et al. [5] segmented gene expression patterns from 2D image data by using a simple thresholding approach. For each gene stained in the image, a specific threshold is suggested by a histogram-based algorithm. Nuclei with expression values larger than the threshold are considered to be in the pattern. This binarizes the image by defining the *pattern* and the *background*. The choice of a good or "correct" threshold has always been an open question since there are several algorithms designed for different purposes.

While many algorithms choose the threshold based on the histogram of the data, some algorithms segment data using spatial information and do not depend on histogram information. In the field of 2D image processing and computer vision, edge detection and ridge detection are two commonly used techniques that use spatial information to extract patterns from images. An edge detection algorithm, such as Canny edge detection [1], can locate pixels where the luminous intensity changes sharply. When such an algorithm is applied to gene expression data, it can tell us at which nuclei expression values change the most in a local neighborhood. If the expression value changes sharply, it is likely that this nucleus is at the edge of an expression domain. On the other hand, a ridge/valley detection algorithm can extract skeletons of watershed/watercourse patterns in an image [2]. Because ridges usually occur

along the center of elongated patterns, they can provide compact representations for these patterns, especially when the patterns in the data have soft edges.

Several ways exist to define a ridge mathematically and different algorithms are designed accordingly. Two major ridge detection categories evaluate ridges by computing principal curvatures [9] and by constructing separatrices [11]. Besides approaches mentioned above, Peng et al. [12] assumed that gene expression patterns in a 2D image can be described by a Gaussian mixture model (GMM), and that gene expression patterns can be segmented and extracted by decomposing Gaussian elements.

Although 2D histogram-based algorithms can be easily applied directly on 3D gene expression data with little, or no modification, algorithms which depend on spatial information, such as GMM, usually cannot be easily modified because 3D expression data are stored on irregular grids. We modified the algorithm proposed in [6] to extract ridge patterns in 3D expression data.

3 Segmenting Gene Expression Patterns

Segmentation techniques, such as Canny edge detection and ridge detection, require gradients of expression values for segmentation. Hence, we have to estimate gene expression gradients before we can apply those techniques.

3.1 Expression Gradient Estimation and Canny Edge Detection

If we treat gene expression values as a function $\mathcal{E} : \mathbb{R}^3 \rightarrow \mathbb{R}$, the gradient vector for nucleus ρ, which is located at position $\vec{\rho} = (x, y, z)$, is defined as $\nabla\mathcal{E}(x, y, z) \equiv \frac{\partial\mathcal{E}}{\partial X}\hat{X} + \frac{\partial\mathcal{E}}{\partial Y}\hat{Y} + \frac{\partial\mathcal{E}}{\partial Z}\hat{Z}$ which defines the direction in which the expression value increases maximally. (Here $\hat{X}$, $\hat{Y}$, and $\hat{Z}$ represent unit vectors in X, Y, and Z direction respectively.) In a *PointCloud*, we only have spatially discrete gene expression values stored at the locations of nuclei. Assume the nucleus ρ_0 has n neighboring nuclei $\rho_i, i = 1 \dots n$. We can choose nearst n neighbors or natural neighbors of ρ_0 (*i.e.* neighbors in the Voronoi diagram) to be ρ_i. Because the expression function can be linearly approximated by the first-order terms of its Taylor expansion $\mathcal{E}(\vec{\rho}) \approx \mathcal{E}(\vec{\rho}_0) + \nabla\mathcal{E}(\vec{\rho}_0) \cdot (\vec{\rho} - \vec{\rho}_0) = e + a(x - x_0) + b(y - y_0) + c(z - z_0)$, where the gradient vector is $\nabla\mathcal{E}(\vec{\rho}_0) = (a, b, c)$, after considering known values $(x_i, y_i, z_i, i = 0 \dots n)$, we have the following $n + 1$ equations:

$$\begin{cases} \mathcal{E}(\vec{\rho}_0) = [e\ a\ b\ c]\ [1 & 0 & 0 & 0\ ]^T \\ \mathcal{E}(\vec{\rho}_1) = [e\ a\ b\ c]\ [1 & (x_1-x_0) & (y_1-y_0) & (z_1-z_0)\ ]^T \\ \quad\vdots \\ \mathcal{E}(\vec{\rho}_n) = [e\ a\ b\ c]\ [1 & (x_n-x_0) & (y_n-y_0) & (z_n-z_0)\ ]^T \end{cases}, \tag{1}$$

where unknown variables in the matrix $[e\ a\ b\ c]$ can be determined by solving a least-square fitting problem. Please note that there also exist other approaches to estimates these gradient vectors. The one we present here can be replaced by another method without influencing the segmentation algorithm too much.

Figure 1 shows the mRNA expression level of a gene called *even-skipped* (*eve*), and the corresponding gradient vectors. Once we have gradient in-

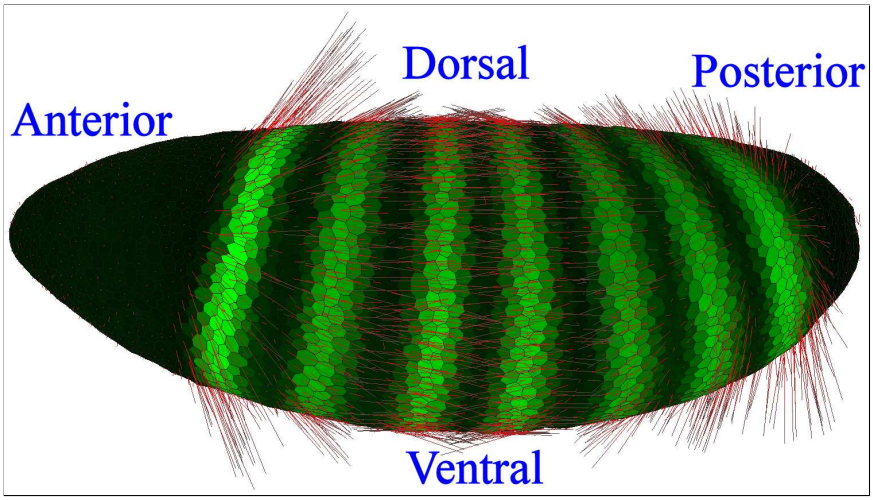
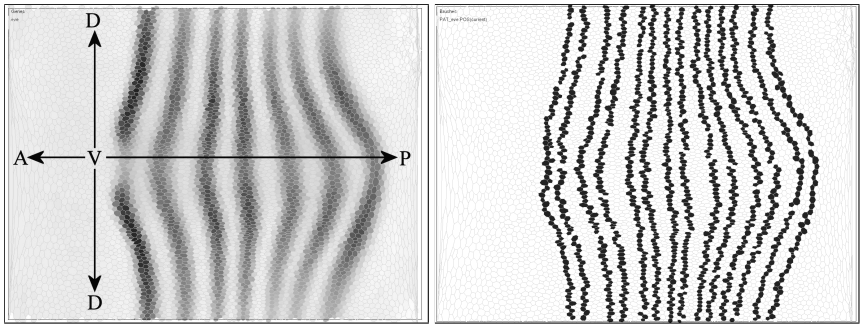


Fig. 1. An example of the mRNA expression levels of the gene *eve*(shown in green) and corresponding gradient vectors. The embryo is shown in a 3D view described in [15]. The center of mass of nuclei define a surface that is partitioned using Voronoi tessellation. A cell is represented by a polygon-like structure and it shape does not reflect the real cell size. Gradient vectors start from red ends and terminate at gray ends. Note that some vectors point into the inside of the embryo.

formation, the Canny edge detection algorithm can be applied to the gene expression data with little modification. Figure 2(b) shows the Canny edge detection result for *eve* mRNA expression levels. In this example, edges of the seven expression stripes are detected. However, this approach cannot directly provide the full regions of expression, and as we show later, does not always usefully segment gene expression patterns. Thus, we also adapted another segmentation method, ridge region detection(see section 3.2), to obtain region information for gene expression patterns and to provide an alternative approach that may be more successful in segmenting gene expression patterns when edge detection fails.

3.2 Segmenting by Ridge Region Detection

Ridge detection can be done by building separatrices or by evaluating curvatures. Separatrix-based algorithms are easier to implement. However, the



(a) *eve* gene mRNA expressions (b) Edges of *eve*

Fig. 2. *eve* edges detected by Canny's algorithm. The embryo is shown in a cylindrical projection as described by Luengo Hendriks et al. [7] with cell-like structures to provides a better overview of the blastoderm surface. The cylindrical projection is only used for displaying the data, the actual algorithms we used in this paper are performed in 3D. The top and bottom of each panel corresponds to the dorsal side (D) of the embryo; the middle corresponds to the ventral side (V); the left side corresponds to the anterior (A); and the right side corresponds to the posterior (P). Higher expression values are shown in darker gray levels.

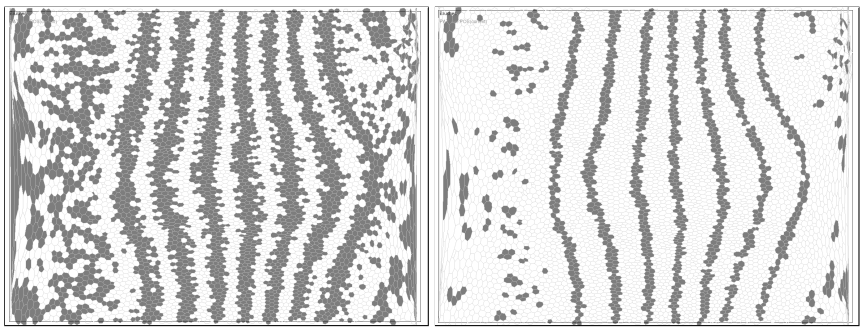
results are usually found not to be satisfying. An edge in the result could be sometimes both a ridge line and a valley line [14]. Another problem of separatrix-based algorithms is that they are global algorithms, *i.e.* the result in a local area can be influenced by changes far away. This is not the case for curvature-based algorithms, which are local.

Traditionally, curvature-based algorithms detect ridges by *level set extrinsic curvatures* (LSEC) [9]. Higher-order derivatives have to be evaluated to compute curvatures. López et al. [6] proposed their *multi-local level set extrinsic curvature* (MLSEC) algorithm which only needs gradient vectors and is more accurate when only discrete data are available. Their basic idea is that the most obvious characteristic of a ridge pattern is that the gradient vectors near the ridge are all pointing toward it. They proved that the divergence of normalized gradient vectors is mathematically equivalent to LSEC. The first reason why we choose their algorithm is that it also works for irregular grids and thus would require less modification. The second reason is that López' algorithm can produce not only ridge lines, which are less useful in our application, but also ridge regions(regions with high values enclosed by loci of inflection points), while Separatrix-based algorithms cannot produce ridge regions.

In López' algorithm, the ridge evaluator $\widetilde{\kappa}_d$ is defined as

$$\begin{aligned}\widetilde{\kappa}_d &= -\mathrm{div}(\hat{\omega}) \\ &= -\lim_{V\to 0}\frac{1}{V}\oint_{\partial V}\hat{\omega}\cdot d\vec{A} \\ &\approx -\frac{d}{r}\sum_{k=1}^{r}\hat{\omega}_k\cdot\hat{n}_k\ ,\end{aligned} \tag{2}$$

where $\hat{\omega}$ is the normalized gene expression gradient vector, d is the number of dimensions, r is the number of neighboring nuclei, $\hat{\omega}_k$ is the normalized gene expression gradient vector at the k-th neighboring nucleus, and $\hat{n}_k$ is the unit normal vector on ∂V for the k-th neighboring nucleus, which can be estimated as $\vec{\rho}_k - \vec{\rho}_0$ in our application. The MLSEC ridge evaluator $\widetilde{\kappa}_d$ is bounded by $[-d, +d]$. Higher positive values of $\widetilde{\kappa}_d$ imply more ridge-like behavior and lower negative values of $\widetilde{\kappa}_d$ indicate more valley-like behavior. Figure 3 shows the MLSEC ridge detection result of *eve* mRNA gene expression.



(a) Positive $\widetilde{\kappa}_d$ (b) High $\widetilde{\kappa}_d$

Fig. 3. MLSEC ridge region detection results of *eve* mRNA expression shown in cylindrical projection. False ridge regions still remain in the anterior area and the posterior area even when we use a high threshold for $\widetilde{\kappa}_d$.

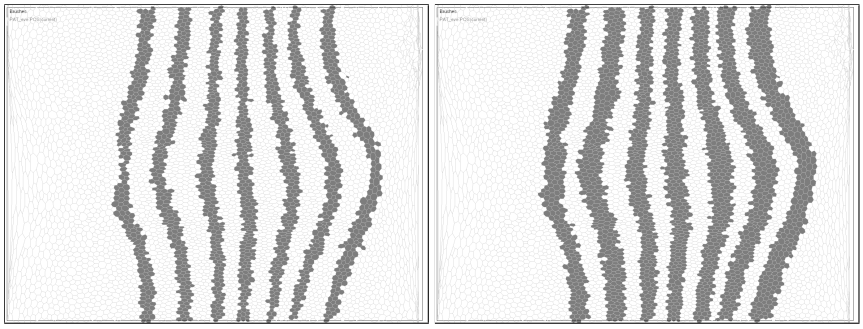
Mathematically, edges of patterns should be composed of nuclei whose $\widetilde{\kappa}_d$ is zero, or very close to zero, since edges also represent loci of inflection points in gene expression. In other words, a ridge region is an area enclosed by edges and has higher expression values than its neighboring regions. To binarize gene expression, the ridge regions are marked "pattern", the rest is "background".

Any measured gene expression data generally contain noise which will introduce small artefactual ridges and valleys that interfere ridge region detection. As Figure 3 shows, undesirable ridge regions still remain in the anterior area and the posterior area, in which *eve* expression values are very low, even when we choose a large threshold for $\widetilde{\kappa}_d$ 3(b). Since in MLSEC $\widetilde{\kappa}_d$ is evaluated by divergence of *normalized* gradient vectors, only directional changes of

gradient vectors contribute to $\widetilde{\kappa}_d$. To get rid of ridges caused by fluctuation of noise, one possibility is to also consider magnitudes of gradient vectors along with their directions, since expression fluctuation caused by noise usually only has a small divergence. Thus, we can define a new ridge evaluator κ_d as

$$\begin{aligned} \kappa_d &= -\mathrm{div}(\vec{\omega}) \\ &\approx -\frac{d}{r}\sum_{k=1}^{r}\vec{\omega}_k \cdot \hat{n}_k \ , \end{aligned} \tag{3}$$

where, $\vec{\omega}$ and $\vec{\omega}_k$ are the unnormalized versions of gradient vectors. Again, higher positive κ_d values still indicate more ridge-like behavior, and lower negative κ_d values indicate more valley-like behavior. However, we note that κ_d is not bounded while $\widetilde{\kappa}_d$ is. Figure 4(a) shows the resulting image of *eve* mRNA expression levels by using our new ridge evaluator κ_d and a low positive threshold obtained by Rosin's unimodal [13] thresholding method, as described in section 4.1.



(a) Positive κ_d (b) fluctuation-tolerant algorithm

Fig. 4. Ridge regions detected in *eve* mRNA expression with our new ridge evaluator κ_d. A low positive threshold is used to remove false ridge regions caused by noise. With our fluctuation-tolerant strategy, this new ridge region evaluator generates better and more accurate results.

The new ridge evaluator κ_d gets rid of the problem of small fluctuations in expression generated by noise; However, it also generates a new problem. Compared to Figure 2(a), ridge regions in Figure 4(a) are in general thinner than expected. This is due to the fact that the use of a threshold to remove nuclei with small κ_d removes not only false ridge regions caused by small fluctuation (noise) but also nuclei that are near, or on edges and hence have smaller κ_d. To address this problem, we propose a fluctuation-tolerant strategy to recover nuclei in real ridge regions. This strategy uses the following three rules:

(1) When a nucleus' κ_d value is larger than the positive threshold T, the nucleus is part of the ridge region.
(2) When a nucleus' κ_d value is smaller than $-T$, the nucleus is not part of the ridge region.
(3) If a nucleus' κ_d value is within the range $[-T, +T]$, the nucleus is part of the ridge region only when at least one of its neighboring nuclei fulfills rule (1); otherwise, it is not in the ridge region.

Here the positive threshold T needs to be defined by the user. In our tool, *PointCloudXplore* [15], we offer several thresholding options which are described in more detail in the next section. An example result of the fluctuation-tolerant strategy is shown in Figure 4(b).

4 Interactive Segmentation

Since no fully automatic segmentation algorithm is applicable in every situation, a good segmentation tool should provide a user interface to allow users to customize parameters used in the algorithms and to perform post-processing to improve the results.

4.1 Thresholding

Thresholding is required in all three approaches discussed above: simple thresholding, edge detection, and ridge detection. It is difficult to design a universal algorithm to pick a good threshold automatically for every case. By considering a data histogram, one can either chose a threshold manually, or pick an appropriate automatic thresholding method, see Figure 5. Depending on the structure of the histogram, one can choose from several automatic thresholding options: *Rosin's unimodal* [13], *2-Gaussian mixture* [10], *2-Mean clustering* [8], and *above one standard deviation*. Other options, such as RATS(robust automatic threshold selection) techniques [4][16], will be added here in the future. What histogram information is shown in the UI depends on the segmentation algorithm used: When simple thresholding is used, the histogram of the original expression values is shown. For edge detection, the histogram of the gradient magnitude is used. For ridge detection, the divergence of the gradient is used.

4.2 Post-processing

Noise and outliers often exist in the data and can generate small false ridge regions. Since an objective quality measure is not easy to define in this application, we just provide two basic post-processing methods here to aid biologists more easily to edit the results to generate the final pattern. Pattern filtering and splitting are the two main pattern post-processing features supported to

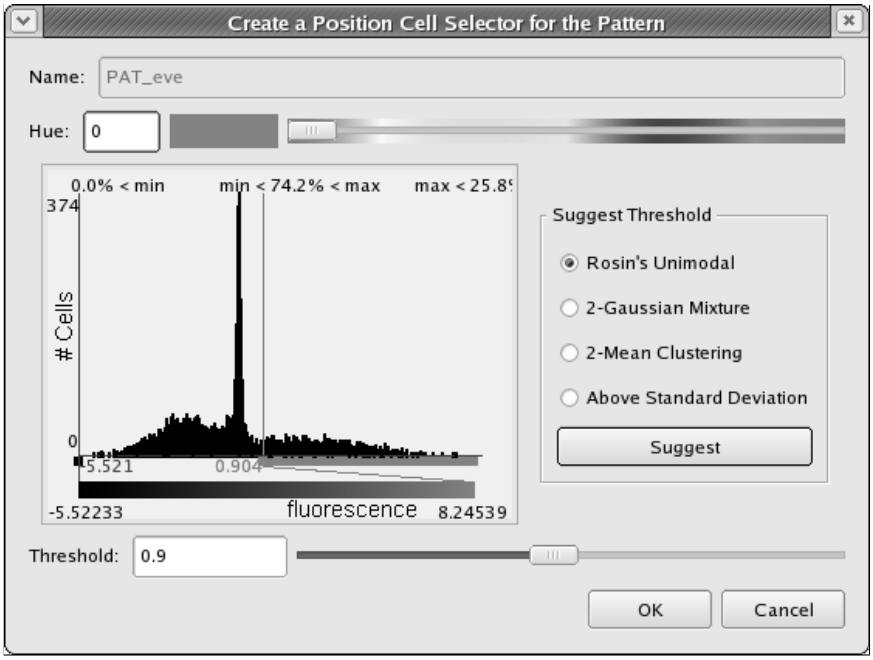
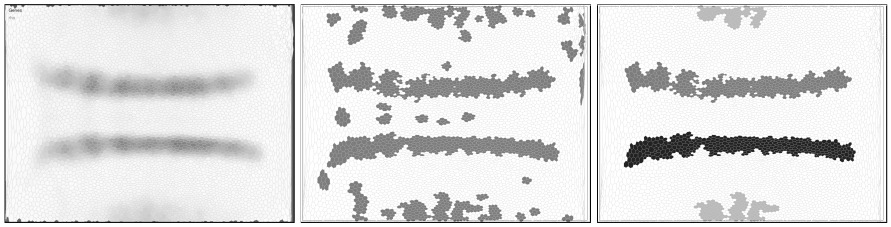


Fig. 5. The UI shows data histogram and provides thresholding options, including algorithms which can suggest a threshold.

improve the quality of segmentation results. False pattern regions could be eliminated by keeping only the first few largest regions, or by filtering out regions smaller than a given number of nuclei. A pattern can also be split into its spatially independent components to allow detailed analysis of independent expression domains. An example of filtering and splitting is shown in Figure 6. In *PointCloudXplore*, segmentation results are in general stored in so-called *Cell-Selectors* (*brushes*) to make further analysis on the generated patterns possible [15].

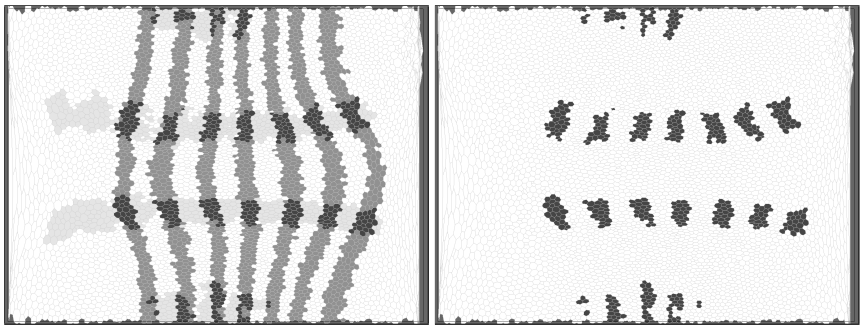
4.3 Segment Editing and Comparison

Segments stored in *Cell-Selectors* can be shown and edited interactively in all physical views in *PointCloudXplore*. This allows the user to correct or fine-tune the segmentation results manually. The user can also combine *Cell-Selectors* with logical operations to create new data patterns [15]. An example is shown in Figure 7.



(a) Original expression (b) Before post-processing (c) After post-processing

Fig. 6. This example uses the ridge regions of the mRNA expression pattern for transcription factor gene *rhomboid* (*rho*). The left image shows original expressions of *rho*. The middle image shows segmentation results before post-processing. The right image shows the results after post-processing by requesting the largest three regions and splitting.



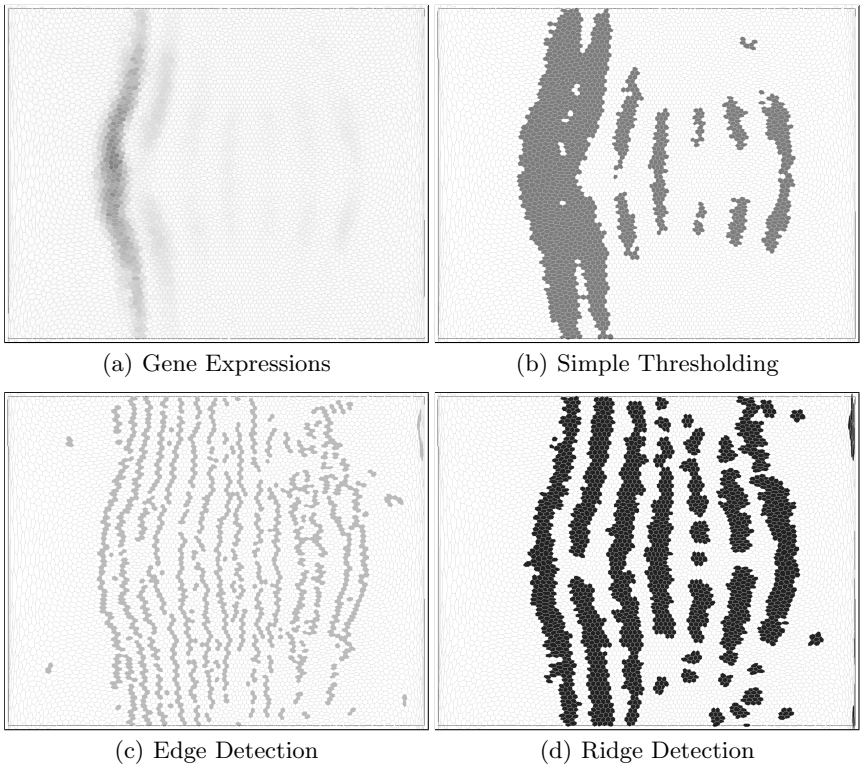
(a) Individual Cell-Selectors (b) Combined with logical-AND

Fig. 7. In the left image, *eve* ridge segments are shown in green while *rho* ridge segments are shown in red. Their logical-AND results are shown in yellow in both images highlighting those nuclei where both genes are expressed.

5 Results and Discussion

Simple thresholding is the easiest way to segment gene expression patterns. It is easy to understand and implement. However, using only one threshold value sometimes does not produce satisfactory segmentation results. For example, as shown in Figure 8(a), there are seven stripes in the *paired* (*prd*, a transcription factor) mRNA expression pattern. The first two stripes have high expression levels, but the remaining five have very low expression levels and can hardly be observed in the image. When using simple thresholding, we have to pick a very low threshold in order to see low-expression stripes. Unfortunately, such a threshold results in poor segmentation results as shown in the first two stripes to merge together, as can be seen in Figure 8(b). On the other hand, as shown in Figure 8(c) and 8(d), both edge detection and ridge detection can

capture all seven stripes because these two approaches perform segmentation based on geometric properties.



(a) Gene Expressions (b) Simple Thresholding

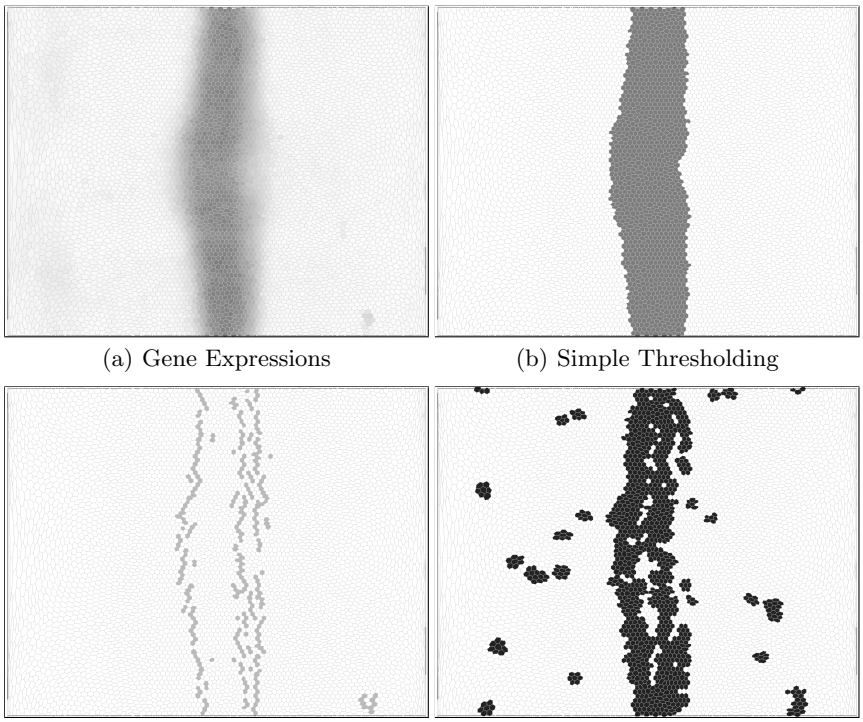
(c) Edge Detection (d) Ridge Detection

Fig. 8. mRNA expression of gene *prd*(a), segmented using simple threshold (b), edge detection (c), and ridge detection (d).

The edge detection approach identifies those nuclei where gene expression changes most rapidly. One way to define a gene pattern is finding the regions enclosed by these edges. However, due to noise, detected edges are usually not continuous, as can be seen in Figure 8(c). Due to these discontinuities, no closed regions are defined and further processing is required before a binarized pattern can be produced. On the other hand, the ridge detection approach yields binarized patterns directly.

In some cases, ridge detection fails to segment gene expression patterns. For example, the "soft edges" or gradual slopes lack strong gradients. These low gradient values then get overpowered by the noise, invalidating both the edges detected and the estimated divergence used by the ridge region detection algorithm. In this case, the simple thresholding approach yields a segmentation result that is less fragmented and more consistent with human perception.

Figure 9 shows such an example. Hence, in *PointCloudXplore*, we provide all three approaches mentioned above to help users define gene patterns. They can choose the most suitable one and later edit the results interactively with our tool if necessary.



(a) Gene Expressions (b) Simple Thresholding

(c) Edge Detection (d) Ridge Detection

Fig. 9. mRNA expression of transcription factor gene *Krüppel* (*Kr*)(a), segmented using simple threshold (b), edge detection (c), and ridge detection (d).

6 Conclusion and Future Work

Defining the pattern of the gene expression is a challenging task. Previous work in this field (e.g.[5][12]) has concentrated on segmenting 2D images in order to extract the expression pattern of a gene. We have presented an interactive semi-automatic approach for 3D gene expression pattern segmentation based on ridge region detection. We have compared our method to standard thresholding and edge-detection-based segmentation techniques commonly used in 2D image analysis, which we have adapted to the problem.

Even though the data we show here in this paper mainly distributed on a 2D manifold, our algorithms do all the computations disregarding this manifold. In the future, we will apply these algorithms to later stage embryos, in which most organs are formed by 3D packing of cells.

Gene expression patterns are not static but show dynamic variation over time. One focus of our future work will therefore be development of analysis methods which take spatio-temporal variation of gene expression into account. Understanding how gene patterns evolve over time is essential in order to understand the complex relationships between genes. Current pattern segmentation methods take only the expression of one gene at a time into account. Development of new analysis techniques which incorporate the pattern of several genes at a time is likely to provide deeper insight into gene relationship.

Acknowledgements

This work was supported by the National Institutes of Health through grant GM70444, by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, by the National Science Foundation through award ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and a large Information Technology Research (ITR) grant; and by the LBNL Laboratory Directed Research Development (LDRD) program;

We thank the members of the Visualization and Computer Graphics Research Group at the Institute for Data Analysis and Visualization (IDAV) at the University of California, Davis; the members of the BDTNP at the Lawrence Berkeley National Laboratory (LBNL) and the members of the Visualization Group at LBNL.

References

1. J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 8(6):679–698, 1986.
2. G. Fu, S.A.Hojjat, and A. Colchester. Integrating watersheds and critical point analysis for object detection in discrete 2D images. *Medical Image Analysis*, 8(3):177–185, Sep 2004.
3. S. V. E. Keränen, C. C. Fowlkes, C. L. Luengo Hendriks, D. Sudar, D. W. Knowles, J. Malik, and M. D. Biggin. Three-dimensional morphology and gene expression in the Drosophila blastoderm at cellular resolution II: Dynamics. *Genome Biology*, 7(12):R124, 2006.
4. J. Kittler, J. Illingworth, and J. Föglein. Threshold selection based on a simple image statistic. *Computer Vision, Graphics, and Image Processing*, 30(2):125–147, 1985.

5. S. Kumar, K. Jayaraman, S. Panchanathan, R. Gurunathan, A. Marti-Subirana, and S. J. Newfeld. BEST: A novel computational approach for comparing gene expression patterns from early stages of Drosophila melanogaster development. *Genetics*, 162(4):2037–2047, Dec 2002.
6. A. M. López, D. Lloret, and J. Serrat. Multilocal creaseness based on the level-set extrinsic curvature. Technical Report 26, Centre de Visió per Computador, Dept. d'Informàtica, Universitat Autònoma de Barcelona, Spain, 1997.
7. C. L. Luengo Hendriks, S. V. E. Keränen, C. C. Fowlkes, L. Simirenko, G. H. Weber, A. H. DePace, C. Henriquez, D. W. Kaszuba, B. Hamann, M. B. Eisen, J. Malik, D. Sudar, M. D. Biggin, and D. W. Knowles. Three-dimensional morphology and gene expression in the Drosophila blastoderm at cellular resolution I: Data acquisition pipeline. *Genome Biology*, 7(12):R123, 2006.
8. J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkely, CA, USA, 1967. University of California Press.
9. J. A. Maintzy, P. A. van den Elsen, and M. A. Viergever. Evaluation of ridge seeking operators for multimodality medical image matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(4):353–365, Apr 1996.
10. G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, 2000.
11. L. R. Nackman. Two-dimensional critical point configuration graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6(4):442–449, 1984.
12. H. Peng and E. W. Myers. Comparing in situ mRNA expression patterns of Drosophila embryos. In *RECOMB '04: Proceedings of the eighth annual international conference on Resaerch in computational molecular biology*, pages 157–166, New York, NY, USA, 2004. ACM Press.
13. P. L. Rosin. Unimodal thresholding. *Pattern Recognition*, 34(11):2083–2096, 2001.
14. P. L. Rosin, A. C. F. Colchester, and D. J. Hawkes. Early image representation using regions defined by maximum gradient paths between singular points. *Pattern Recognition*, 25(7):695–711, 1992.
15. O. Rübel, G. H. Weber, S. V. E. Keränen, C. C. Fowlkes, C. L. Luengo Hendriks, N. Y. Shah, M. D. Biggin, H. Hagen, D. W. Knowles, J. Malik, D. Sudar, and B. Hamann. PointCloudXplore: Visual analysis of 3D gene expression data using physical views and parallel coordinates. In T. Ertl, K. Joy, and B. Sousa Santos, editors, *Proceedings of the EUROGRAPHICS - IEEE VGTC Symposium on Visualization 2006*, pages 203–210, Lisbon, Portugal, May 2006.
16. M. H. F. Wilkinson. Optimizing edge detectors for robust automatic threshold selection: coping with edge curvature and noise. *Graph. Models Image Process.*, 60(5):385–401, 1998.

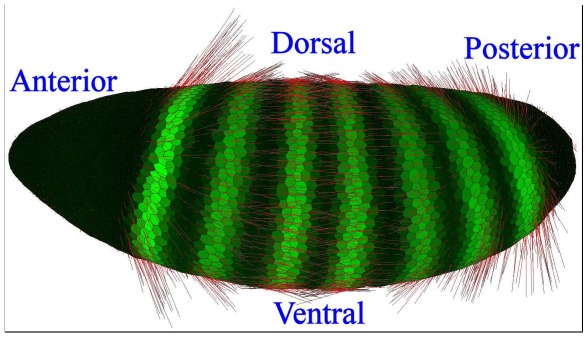


Figure 1

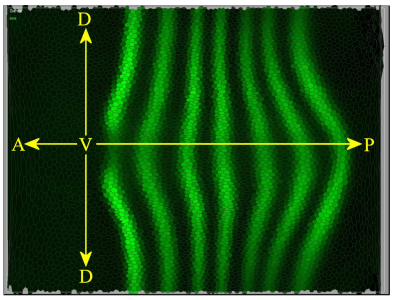


Figure 2(a)

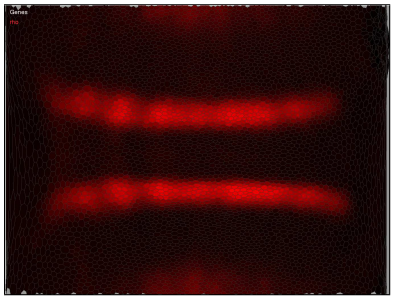


Figure 6(a)

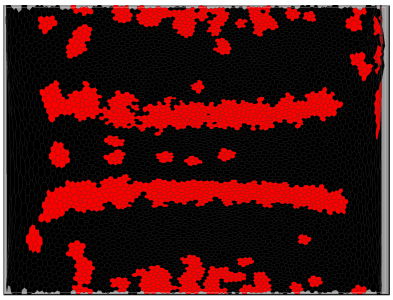


Figure 6(b)

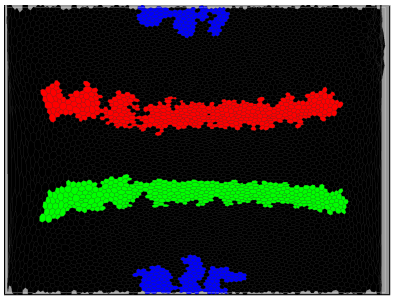


Figure 6(c)

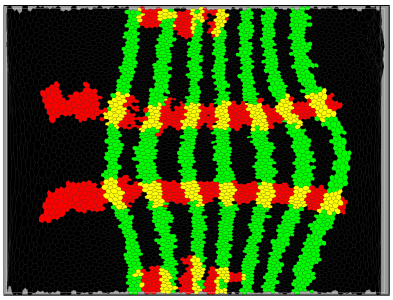


Figure 7(a)

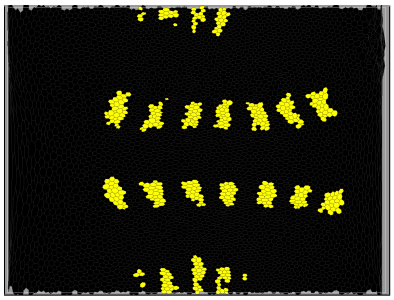


Figure 7(b)