ML methods for Edge detection and Tracking for Drosophila Dorsal Closure

Swarna Ravindran
Department of Computer Science
Duke University
swarnakr@cs.duke.edu

We have developed fast and efficient methods to analyse Dorsal closure in Drosophila. We first detect edges inside and outside the "amnioserosa" region in Drosophila using Structured decision forests. This involves detecting faint and dark edges simultaneously, a scenario where traditional edge detectors are inefficient and noisy since it requires manual parameter tuning. The Structured decision forests model enables us to perform accurate and fast edge detection that is tailored to the kind of edges found in Drosophila.

Next, we track the vertices of the cells in the amnioserosa using the Kalman filter. This would help analyse the change in the area of the smaller cells in the amnioserosa and make predictions about the kinematics involved during Dorsal closure.

1 Introduction and Related Work

Biologists have analysed the change in the area of cells as it closes, or Dorsal closure for the larger understanding of the forces that drive cell shape changes[1, 2]. Kinematic models and dynamic models[1] have been developed to this end. Recently tools from Computer Vision have been used to analyse the problem[2].

However, these methods need the detection and tracking to be optimized over several variables and need explicit point correspondences between consecutive frames for the entire video sequence. Such methods are not only computationally intense but also slow.

We analyse the system from a purely visual perspective, using machine learning techniques such as structured decision forests and Kalman filter equations (which in fact bear a striking resemblance to biologically motivated state equations) that are very fast, cheap to compute and still yield accurate results. We hope that this intuitive framework can enable biologists to infer useful information about the system.

2 Method

We detect edges on the Drosophila amnioserosa using Structured Random Forests that are more efficient than traditional edge detectors such as Canny[3]. The Canny detector typically yields threshold-dependent noisy edges, where detecting faint and thick edges would require manual threshold specification and tuning. Structured Random Forests can be trained to yield edges required for the application at hand. They are also faster, more robust and capture the relationship between neighbouring pixels.

Once we detect edges, we detect the vertices connecting the edges using a corner detector such as Harris[4], and track them across frames in the video, using the Kalman Filter. Since the area of each cell in the amnioserosa changes to different extents, a separate velocity model is built for each vertex. The vertices and cells in the remaining frames can be automatically tracked using this model. A modest set of training images (first 30 frames of the video) is used to obtain the parameters of the velocity model.

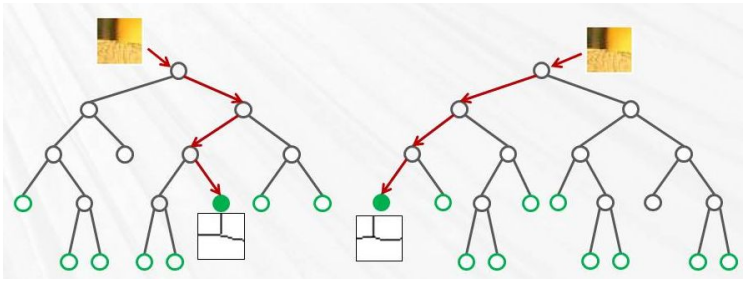


Figure 1: The relationship between pixels is captured in an image patch in a structured decision tree.

2.1 Structured Random Forests

A binary decision tree enables one to test on an attribute or feature at each one of its nodes, and branch to one of two outcomes depending on the value of the input and the test performed at the node. The test could be as simple as a threshold operation.

Random forests are a collection of decision trees, trained using a different parameters for each tree[5]. A typical decision tree, as opposed to a structured decision tree, takes scalar inputs and contains a scalar value at its leaf. For example, the leaf of a typical decision tree that performs edge detection contains the probability that the input pixel is an edge. A structured decision tree takes in structured input (patch of an image) and contains structured output (edge patch) at the leaf as shown in

Figure1[5]. This method also minimizes computations for each pixel since each leaf contains edge values of a collection of pixels (in the patch) and is thus much faster than its traditional non-structured version.

During training, we find parameters of the split function at each node that result in a good split of the data. This corresponds to maximizing the difference in values that land at the two branches of the node. This is defined using the information gain criterion of the form given in Eq1.

$$\Delta E = -\frac{|I_l|}{|I_n|}E_(I_l) - \frac{|I_r|}{|I_n|}E_(I_r) \tag{1}$$

where $|I_l|$ and $|I_r|$ correspond to the number of elements in the left and right branches respectively from the input set of elements in $|I_n|$. $E_(I_l)$ and $E_(I_r)$ correspond to the entropy of the elements in the left and right branches.

Thus, the threshold value to be applied on the input feature is determined at each node during training according to the information gain criteria[5].

Computing the information gain in the structured input scenario involves computations over high dimensions, ie 256 dimensions for a 16X16 patch. To perform this operation efficiently, a 2-means clustering of the input patches is done at each node as shown in Figure2.1. Thus, the input image at each node is sent to the left or right branch depending on the cluster it belongs to.

2.2 Kalman Filter

After obtaining the edge, we detect vertices on the first 30 frames using the Harris corner detector and associate them with the corresponding points to form the tracks or the measured values. Using these measured values in the Kalman filter equations below, we build a separate velocity model for each point.

$$x_{t+1} = Ax_t + Bu_t + w_t \tag{2}$$

$$y_t = Cx_t + v_t \tag{3}$$

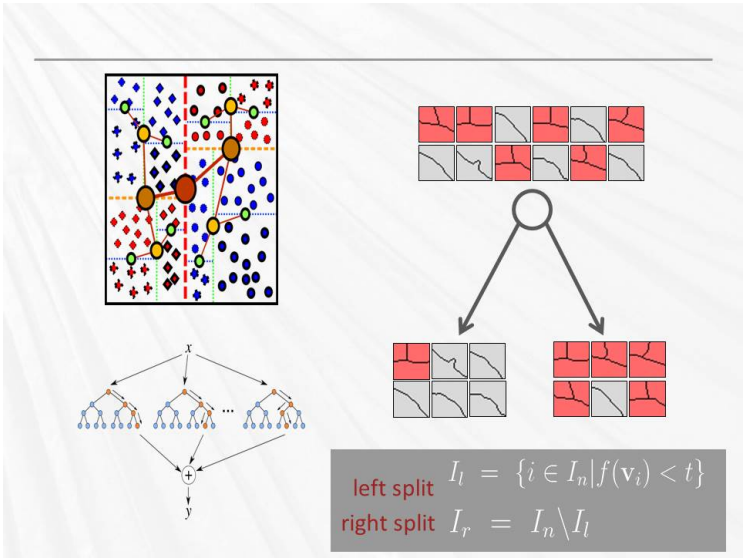


Figure 2: (left) Training and Testing modules of a decision tree (right) The information gain at each node

where $w_t \sim N(0, \Sigma_w)$, $v_t \sim N(0, \Sigma_v)$, and $x_0 \sim N\left(x_{0|-1}, P_{0|-1}\right)$. Note that $x \sim N(\mu, \Sigma)$ means

$$P(x) = \frac{1}{(2\pi)|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)\Sigma^{-1}(x-\mu)}. \tag{4}$$

We also have: $Ex = \mu$ and $E(x-\mu)(x-\mu) = \Sigma$.

After obtaining the parameters A, B, C, w and v, the velocity and therefore location of any point in the Drosophila cell can be predicted.

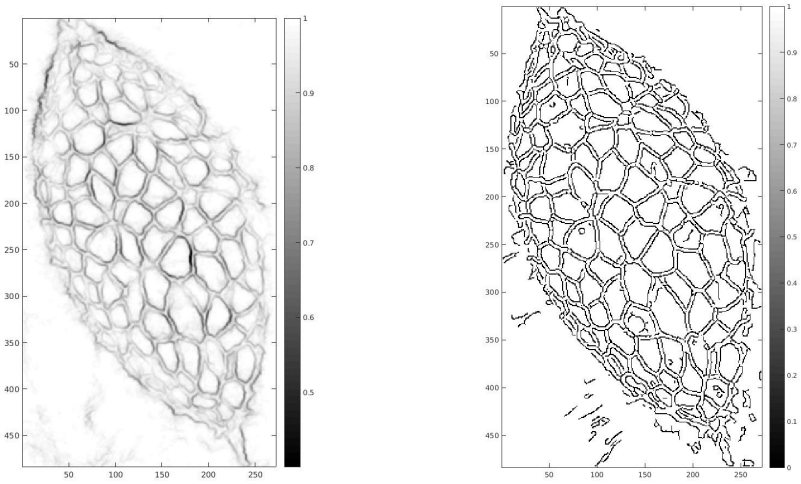
3 Results

4 Discussion

Figure 3 shows the comparison between the output edges obtained with training using Structured forests and that of the Canny edge detector. We observe that the edges are cleaner and more adapted to the Drosophila training images. Figure 4 shows edges detected on two frames after a time lag. This shows that our edge-detection is robust to the shrinkage in the cell and therefore conducive to tracking. Figure 5 shows a variation of our method where the edges are detected using smaller window sizes in the training and testing. This enhances the output in a manner similar to contrast normalisation.

We observe from these figures that the edges in Drosophila outside the amnioserosa can be detected effectively with low noise and high accuracy at very high speed by modifying the Structured Random forests.

We observe from Figure 6 that the Kalman filter can be used to predict the location of vertices inside the amnioserosa and track them efficiently.



(a) (b)

Figure 3: (a) General structured forest (b) Canny Edge detector

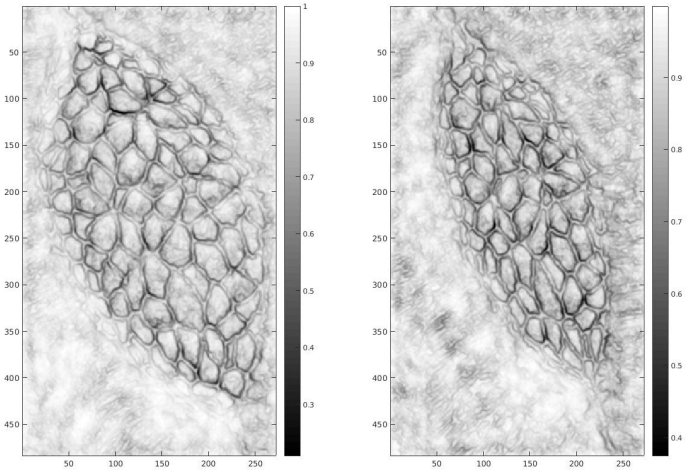


Figure 4: Edge detection on frames 5 and 100. The edges outside the amnioserosa overlap to a good extent and are trackable.

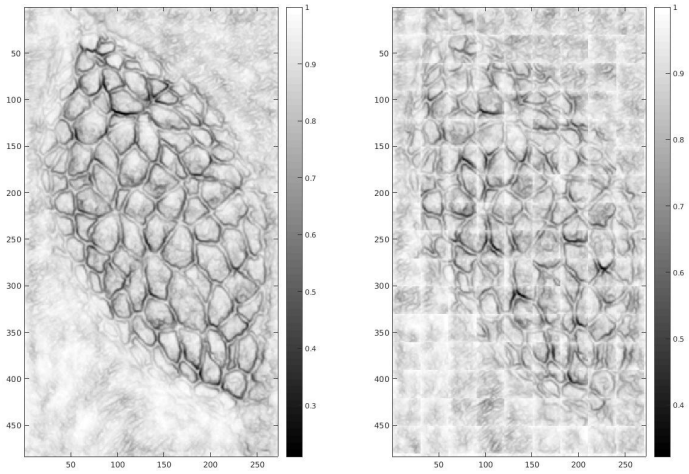


Figure 5: Two variations of our method, where even faint edges appear visible

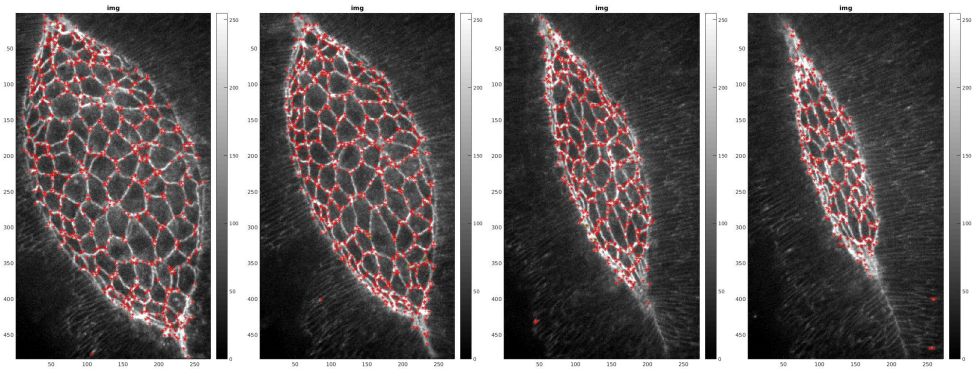


Figure 6: Tracking results at frames 1, 50, 180 and 280 as the cell closes

5 Conclusion

We have developed faster and more reliable methods to enable biologists to infer the kinematics of Dorsal closure in Drosophila using efficient and accurate image analysis aided by machine learning tools.

References

[1] Adam Sokolow, Yusuke Toyama, and Daniel P et al Kiehart. Cell ingression and apical shape oscillations during dorsal closure in drosophila. *Biophysical journal*, 102(5):969–979, 2012.

[2] Sabine C Fischer and et al Blanchard. Contractile and mechanical properties of epithelia with perturbed actomyosin dynamics. *PloS one*, 9(4):e95695, 2014.

[3] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

[4] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Citeseer, 1988.

[5] Piotr Dollár and C Lawrence Zitnick. Structured forests for fast edge detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1841–1848. IEEE, 2013.