# Automated Detection of Chondrocytes in Growth Plate Images

Emily Beylerian, Brian de Silva, Ben Gross, and Hannah Kastein
Mentors: Dr. Maria-Grazia Ascenzi and Hayden Schaeffer

**Abstract**

Analysis of chondrocyte development in bone growth plates yields data concerning foundational genetic structure. Previous research on chondrocyte organization and alignment has required manual detection of cell sizes and locations. This paper proposes an algorithm for automated detection of chondrocytes within bone growth plate images. The method aims to aid biological research by increasing the efficiency and consistency of collecting data from images of growth plate regions. Due to indistinct background cells and variations between regions of growth plates, classical techniques of clustering, segmentation, thresholding, and filtering fail to accurately identify cell boundaries. We propose an automated algorithm that incorporates the methods of Retinex, anisotropic diffusion and various thresholding operations in order to detect locations and sizes of chondrocytes. Our results and analysis demonstrate the effectiveness of the proposed numerical method as applied to various growth plate images.

# 1 Introduction

Longitudinal growth of bones is the result of a process involving cell division, migration, then ossification that occurs in growth plates which are located at both the proximal and distal ends of the bone. In healthy growth plates the cells, called chondrocytes, arrange in columns along the direction of growth. Analysis of the distribution of chondrocytes provides insight into the developmental effects of repressing various genes, the lack of which can lead to bone growth disorders [2]. Detecting chondrocytes in the growth plate is a vital first step to analyzing the distribution of these cells. Rather than manually determining cell sizes and locations in each zone of the growth plate, an automated process of cell-detection and zone approximation would aid biological research by increasing efficiency and consistency.

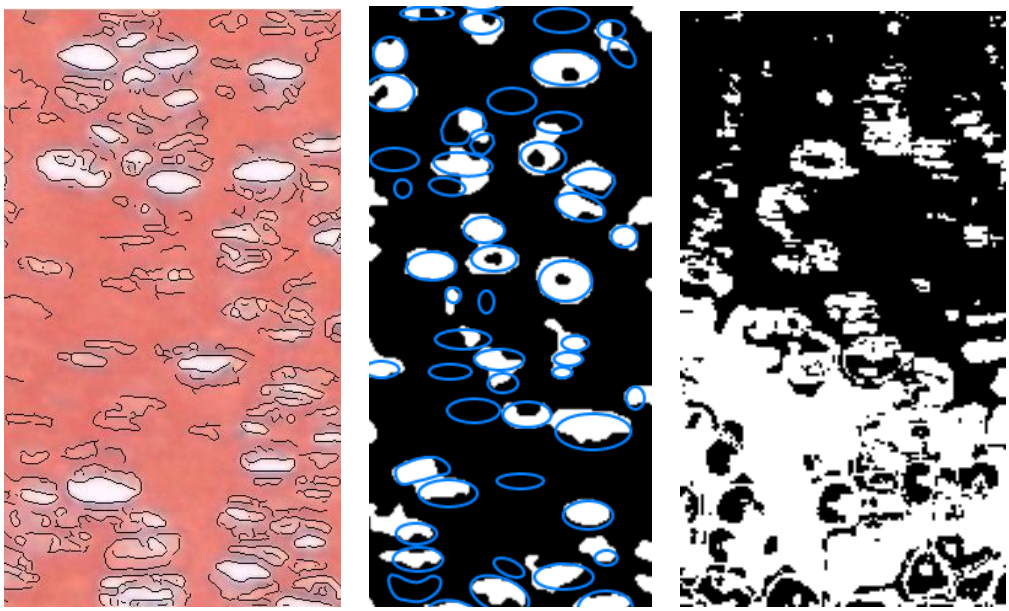
Classical methods of segmentation and image processing are ineffective on growth plate images for multiple reasons. Within one growth plate, the characteristics of chondrocytes vary greatly by region. In the resting zone, cells are small, circular, densely packed, and have distinctly visible nuclei. Upon entry into the proliferating zone, cells elongate horizontally and undergo division by mitosis. Each division results in one mother cell and one daugther cell. In this region, mother and daughter cells are closely spaced, often overlapping or sharing a border, but may be far apart from other cells in the region. Only some nuclei are visible in the chondrocytes. In the hypertrophic zone of a growth plate, cells are relatively larger than in the rest of the growth plate, are very round, and are packed closely to one another with few to no visible nuclei. The region is marked by the ossification of the chondrocytes [3]. Throughout the growth plate, chondrocytes positioned beneath the plane of the slide appear faintly in the background of the image. This complicates detection of cells on the plane of focus of the slide. Furthermore, the intensity of stain penetration in the background varies throughout the images. Edge detection methods pick up intensity changes in background cells and staining variations as edges, which result in false positive detections. Pre-existing cell detection software packages such as ImageJ [14], CellProfiler [4], and HT-HCS [6] do not take these constraints into account.

Released by the National Institutes of Health in 1997 as open source image processing software compatible with multiple platforms, ImageJ includes many filtering, thresholding, segmentation and clustering methods as tools and plugins [5]. A recent implementation of ImageJ for evaluation of apoptotic cell death [7] has similar cell detection goals. In this process, however, detection of every cell on a slide is desired, leading to simpler separation from the background than is required for our project. CellProfiler is a free open source program designed for cellular image analysis [4]. Unfortunately, when applied to our images, this program overestimates the number of cells, resulting in many false detections. In [6] an automated cell segmentation algorithm for high-throughput (HT) and high-content screening (HCS) is proposed. This HT-HCS algorithm first detects nuclei locations, then uses these as seed points for cell locations. The seeding process

assumes that each cell has a nucleus and that there is a one-to-one correspondence between local maxima in pixel intensity values and nuclei locations. The proceeding cell detection method employs histogram thresholding and k-means clustering. As detailed above, we cannot assume that our class of growth plate images contains a well-defined nucleus for every cell. Furthermore, we will describe the shortcomings of k-means clustering as a cell detection method in our images.

Attempting to segment the chondrocytes in an input image before pre-processing results in false detections of cells beneath the plane of focus. Only applying segmentation also assigns edges to changes in the intensity of stain penetration, as shown in Figure 1(a). In addition, the irregularities and inconsistencies of growth plate slides also render many standard processing techniques ineffective. Motivated by the ideas in [11] of using weaker norms and which [16] extended, cartoon-texture decomposition interprets small-scale features in an image as texture and separates them from the large-scale cartoon components. We use a model based on the work in [13,18] where ideally, correct parameterization would identify the chondrocyte boundaries as the texture component and the remainder of the image as the cartoon component. In practice, the drastic cell enlargement between the resting zone and hypertrophic zone prevents the use of a single set of parameters for an entire image. Even when confined to a single region of the growth plate, the texture component does not accurately detect cells on the plane of focus. Figure 1(b) shows an example of cell-detection according to cartoon-texture decomposition compared to manual detection. The k-means clustering algorithm is based on work in [10], which takes a set of $n$ data points and clusters them into $k$ groups, based on the similarity of each point. The desired output of k-means was to have three separate clusters: background, cells, and nuclei. Unfortunately, changes in background stain penetration result in inconsistent clustering of background pixels and cell pixels, as shown in Figure 1(c).

The wide variety in cell size, shape, and density, as well as oscillatory pixel intensity values mentioned above, demonstrate the need for a new automated cell detection algorithm for this class of growth plate images. In this paper, we propose an automated algorithm that incorporates the methods of Retinex, anisotropic diffusion and various shape thresholding operations in order to detect location and size of the chondrocytes existing on the plane of focus of the slides, while dividing the growth plate into appropriate regions. In Section 2 we describe the origins and implementation of the methods used in our algorithm. In Section 3 we give an outline of the final algorithm. In Section 4 examples of final results of chondrocyte detection in various growth plates are displayed. Section 5 describes the methods of error analysis used to asses the accuracy of our process, followed in Section 6 by a conclusion of our findings.



(a) Canny Segmentation (b) C-T Decomposition (c) K-Means Clustering

Figure 1: Results from standard segmentation and processing algorithms. Canny segmentation applied directly to an input growth plate image in (a) results in false positive cell-detection and incomplete edges. The texture component of the cartoon-texture decomposition in (b) is overlaid with blue outlines of manual cell detections. Results of k-means clustering in (c) show unwanted clustering of background pixels.

# 2 Methods

## 2.1 Retinex

Retinex theory was developed in an effort to imitate and describe human color perception [8]. Because the original Retinex method was excessively complicated, many different interpretations and implementations of the method have been developed [9]. When applied to images, Retinex algorithms typically smooth together subtle shading variations as well as enhance effects created by the human visual system. Thus, a Retinex algorithm has the potential to remove subtle differences in shading in our images. It could cause cells to have similar intensity values and make the background more homogeneous. It should be noted that Retinex methods will never improve the overall quality of an image as they purposely remove information. However, our images inherently contain extraneous data in the form of cells outside the plane of focus and inconsistencies in the shading and in the luminosity of the background.

It was recently shown in [12] that it is possible to formalize the original Retinex algorithm as a discrete partial differential equation. The primary task of the algorithm then becomes solving a version of the Poisson equation with Neumann boundary conditions. Given an initial image, $f$, we want to find a reconstructed image, $u$, such that

$$-\Delta u_{i,j} = F_{i,j}, \tag{1}$$

where

$$\Delta u_{i,j} := u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} \tag{2}$$

is the discrete Laplacian at pixel $(i, j)$. $F_{i,j}$ is defined as

$$F_{i,j} = T(f_{i,j} - f_{i+1,j}) + T(f_{i,j} - f_{i-1,j}) + T(f_{i,j} - f_{i,j+1}) + T(f_{i,j} - f_{i,j-1}), \tag{3}$$

and $T$ is the hard thresholding function such that

$$T(x) = \begin{cases} 0 & \text{if} \quad |x| \leq \tau \\ x & \text{if} \quad |x| > \tau \end{cases} \tag{4}$$

where $\tau$ is a thresholding parameter. Thus the algorithm first considers every pixel in the image and replaces small gradients (those with magnitude less than the parameter $\tau$) with zero, thereby creating a vector field. The Poisson equation then constructs the image whose gradient most closely matches said vector field. These steps can then be reapplied to the resultant image iteratively.

There are multiple well-known numerical methods that are often employed to solve the Poisson

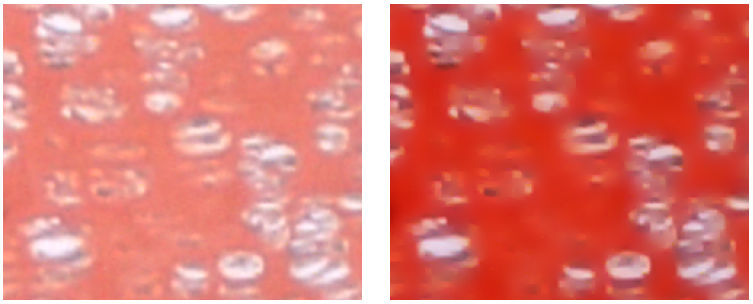


Figure 2: An image of a cropped region of a bone growth plate before the Retinex algorithm has been run (on the left) and an image of the same cropped region after an application of the algorithm (on the right).

equation, for example, inversion of the discrete Laplacian in the Fourier domain (see [12]). For our application, the Retinex Poisson equation is used for jointly denoising and removing small variations in image luminosity. As opposed to shadow removal, the anomalous variations that exist in our set of images are on a small scale. That is they are local variations rather than global ones. Hence a semi-implicit scheme is more than sufficient for our purposes. The scheme is as follows:

$$u_{i,j}^{n+1} = \frac{1}{4}(u_{i-1,j}^{n} + u_{i+1,j}^{n} + u_{i,j-1}^{n} + u_{i,j+1}^{n} + F_{i,j}). \tag{5}$$

Since the method considers only local variations, the iterative scheme (5) converges to the steady state rapidly.

In Figure 2 we have an example of the effects of our Retinex algorithm on an image of a small region of a bone growth plate. Note how the background in the second image has been made more homogeneous than in the first. This is a result of the smoothing effect Retinex algorithms often create. Also notice that the faint cells outside the plane of focus in the first image have been faded considerably, if not eliminated entirely. Our Retinex method has performed in the exact manner as expected. It has lessened the prominence of cells outside the plane of focus. The algorithm has similar effects across all of our growth plate images.

Applying our Retinex algorithm to the images of bone growth plates is an early step in our overall processing method. Once we have eliminated some of the superfluous detail in the images with a Retinex algorithm, other methods to locate the cells themselves can be applied.

### 2.2 Anisotropic Diffusion

Filters are processes which, given some input, decompose that input into both the desired output and the undesired extras. In general, filtering is an averaging process, where the intensity value at

each pixel is replaced by a weighted average of neighboring values. These methods can be linear, such as Gaussian and Laplacian filtering, or nonlinear, such as total variation and anisotropic diffusion filtering.

### 2.2.1 Theory

The foundation of anisotropic diffusion lies in Gaussian filtering,

$$u = G_\sigma * f, \tag{6}$$

where the filtered signal u is defined as the convolution between the Gaussian filter $G_\sigma$ and the original signal $f$. The Gaussian filter $G_\sigma$ is related to solving $\frac{\partial u}{\partial t} = \triangle u$ in finite time, where the standard deviation, $\sigma$, is related to time through the relationship $\sigma = \sqrt{2t}$. The Laplacian, $\triangle u$, has too high of a diffusivity and thus over-smooths the edges. Anisotropic diffusion is a member of a different family of diffusivity functions,

$$\frac{\partial u}{\partial t} = \operatorname{div}\left(g\left(|\nabla u|\right)\nabla u\right), \tag{7}$$

where $u(t = 0) = f$. The above family of diffusivity functions deals with edges differently than it approaches areas which are distant from edges. This is because, in general, as $z$ goes to infinity, $g(|z|)$ goes to zero. However, as $z$ goes to zero, $g(|z|)$ approaches one. These boundary conditions come from the following definition of $g(|z|)$.

$$g\left(|\nabla u|\right) = \frac{1}{|\nabla u|^p}, \tag{8}$$

where varying the value of $p$ distinguishes the methods within the diffusivity function family, as shown in Equation 7. With the value of $p = 0$, the high diffusion levels of the Laplacian are reached. When $p = 1$, total variation flow has been created; and as $p$ extends past one to $p = 2$, Perona - Malik Diffusion is generated. In general, with anisotropic diffusion, we employ $0 < p < 1$. The effect the equation defining $g(|z|)$ has regarding diffusivity in an image is as follows: where there is a large gradient, such as across an edge, only small amounts of diffusion will occur. Where there are small gradients, due to noise or small features, diffusivity is high. Thus, while diffusion is inhibited across edge boundaries, it is still encouraged along those boundaries. This can occur due to the directionality of the gradient being preserved in addition to its magnitude.

$$u = \arg\min_u \int \Psi\left(\nabla u, \nabla u^T\right) dx, \tag{9}$$

where $\Psi$ is always choosen so its derivative is the diffusivity function $g$,

$$\Psi' = g. \tag{10}$$

The order of vector multiplication between $\nabla u$ and $\nabla u^T$ in Equation (9) creates for anisotropic diffusion a tensor structure rather than a scalar value. The tensor structure is where the directionality of the gradient is preserved.

### 2.3 Threshold

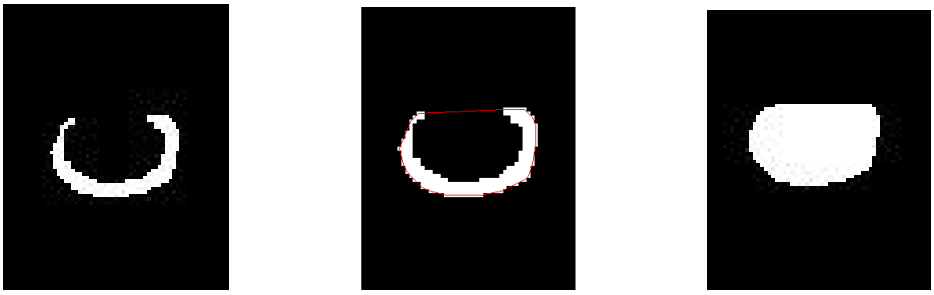
From the cartoon-like solution, edge information is extracted through a gradient detector. The output is a matrix containing gradient values scaled between 0 and 255. In order to segment the cells from the background and clean the image, morphological functions will be applied in future steps. However, in order to apply morphological functions, a binary representation of the gradient image is needed. To obtain the binary representation of the gradient image, hard thresholding is applied. For the definition of hard thresholding see Equation 4. The value of the set threshold value is chosen as to maximize membrane separation for cells within clusters while minimizing cell membrane pixel loss over the entire growth plate region. However, loss of pixels that represent chondrocyte perimeters is prevalent. The next step in the pipeline is to apply the convex hull operation. The algorithm will act on the binary representation of the gradient image obtained through hard thresholding to linearly bridge the cell perimeter discontinuities.

### 2.4 Convexification of Cell Perimeters

Convex hull is a morphological operation which we apply to a binary representation of the gradient image. After hard thresholding to obtain the binary representation, there is a loss in the quantity of cell perimeter pixels. Thus, incomplete cell perimeters result. In order to fill and identify cells, discontinuities in cell boundaries must be eliminated. Mathematically, to perform a convex hull operation means to find the smallest convex set that encloses the given points. The convex hull operation was applied to the binary representation using the *convhull* command in MATLAB. The MATLAB algorithm identifies the smallest convex set of points that fully encloses the perimeter of each cell. Then the points are connected, forming a filled polygon representation of each cell.

### 2.5 Biological Restrictions

The next processing steps are size thresholding and shape thresholding. These operations are based on our knowledge of cell biology. The cells are thresholded in order to conform to expected cell



(a) Incomplete cell perimeter produced by thresholding the gradient.

(b) Convex hull overlaid on the incomplete cell perimeter

(c) Filled polygon represented cell produced.

Figure 3: Convex hull operations steps.

geometries, of which we specifically utilize restrictions on area and convexity. The thresholding is applied to the output of the convex hull operation. Size thresholding eliminates large polygons which are often created during the convex hull operation due to boundary points of the growth plate region. See Figure 4. Polygons with areas larger that a set maximum pixel value are eliminated, as well as polygons with areas smaller than a set minimum pixel value. In addition to the elimination of polygons which do not conform to size expectations, elimination of cells with nonconforming convexities is also applied. Our shape thresholding is based on the isoperimetric inequality. For each polygon we compute the isoperimetric ratio, $C$, where

$$C = \frac{4\pi A}{L^2}. \tag{11}$$

In the ratio equation, $A$ is the area of the polygon and $L$ is the length of the perimeter of the polygon. Polygons with isoperimetric ratios that demonstrate geometries that do not conform with cell biological knowledge are eliminated.

### 2.6 Zone Approximation

In order to perform biological analysis on the bone growth plates captured in our images it is important to be able to identify which zone a given cell lies in. There are three major zones present in the majority of our images; the resting zone, the proliferating zone, and the hypertrophic zone. The cells in the resting and hypertrophic zones are, in general, much more circular than the cells in the proliferating zone. The cells in this region are much flatter than those in the other two. Furthermore, the cells immediately undergo drastic changes in size and shape upon entering the proliferating zone from the resting zone and again upon leaving the proliferating zone for the hypertrophic zone. We use this predictability in cell shape to estimate the three zones.

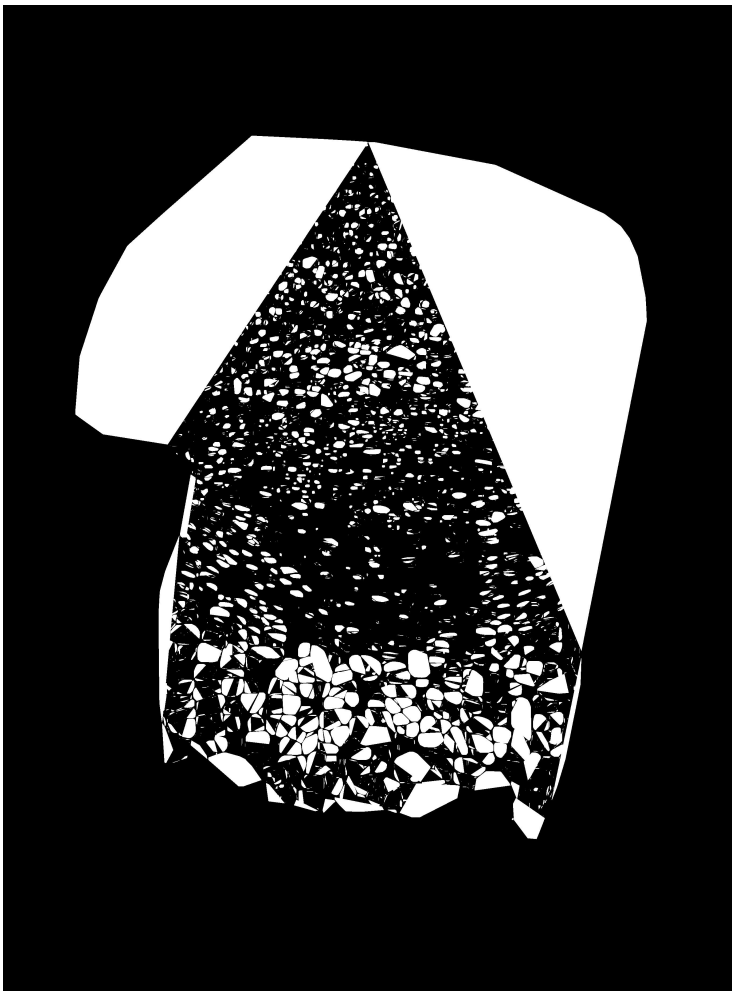


Figure 4: Image of growth plate region after convex hull operation. Polygons are present that do not correspond to cells.

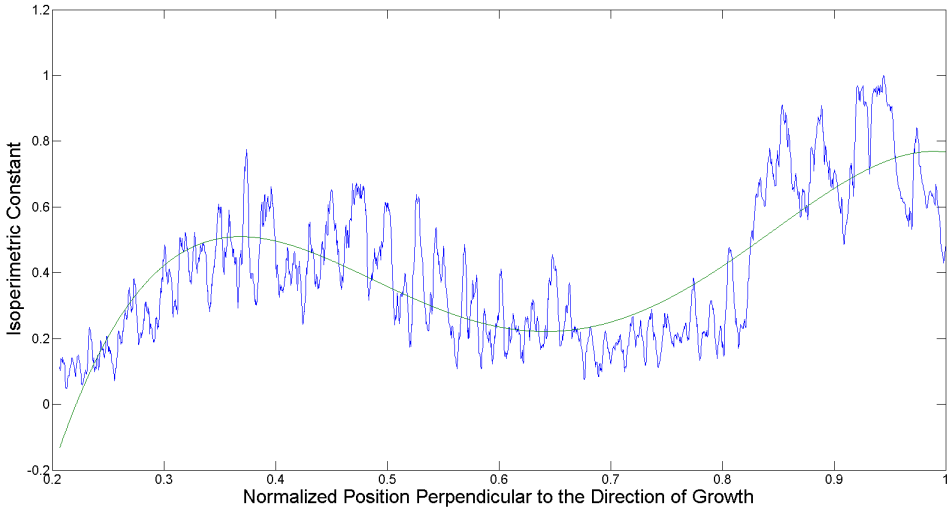
We start by classifying each cell based on its isoperimetric constant, a good measure of its shape for our purposes. Next we compute the average isoperimetric constant along strips that lie perpendicular to the direction of cell migration in the growth plates. For the majority of our images, the direction of cell movement is downward, and so in most cases we consider horizontal strips across the image. Next we plot the averages in MATLAB. Since the cells in the resting and hypertrophic zones are much more circular than those in the proliferating zone and because the proliferating zone separates the other two zones, we expect two peaks in this graph. Thus we use a fourth degree polynomial to approximate it. We then use the two inflection points of the polynomial as estimations of the boundaries between the zones.

As is shown in Figure 5, the inflection points of the fourth degree polynomial provide satisfactory estimations of both the boundary between the resting and proliferating zones and the boundary between the proliferating and hypertrophic zones.

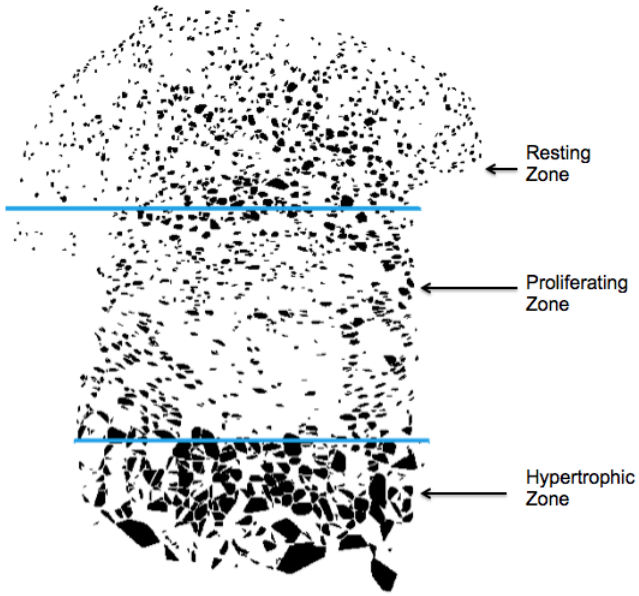
# 3 Algorithm

Our final algorithm was implemented entirely in MATLAB. It expects only two inputs: a cropped image of a bone growth plate with a white background, and the same image with a green background (the background should have the uniform RGB value (0,255,0)). The output is a binary image of the same size as the initial images, where the cell pixels have been colored white and the remaining pixels of the image black.

First, the Retinex algorithm specified in 2.1 is applied separately to each color channel of the image of the bone growth plate with the white background. The results from each color channel are then recombined to form a new image. The output image from the Retinex process is then passed to an anisotropic diffusion algorithm. Anisotropic diffusion is applied in order to calculate the gradient of the image. The gradient of the image contains information regarding diffusion along and across cell boundaries. We are interested in the gradient output of the anisotropic diffusion process. We use our knowledge of expected chondrocyte geometry to analyze this image and compute a binary representation of the gradient image using a hard threshold so that we may then apply morphological functions. Discontinuities in cell boundaries occur as an unavoidable result of the thresholding conversion process. Thus, a convex hull operation is applied. The resulting output of filled polygon representations of cells is cleaned using size thresholding, based on the minimum and maximum number of pixels in a cell and shape thresholding based on the calculation of the isoperimetric ratio for each polygon. Finally, we approximate the resting, proliferating, and hypertrophic zones of the bone growth plate using the isoperimetric constants of the objects. The product of this entire process is a black and white image with two blue lines drawn along the estimated boundaries of the resting and proliferating zones and of the proliferating and hypertrophic

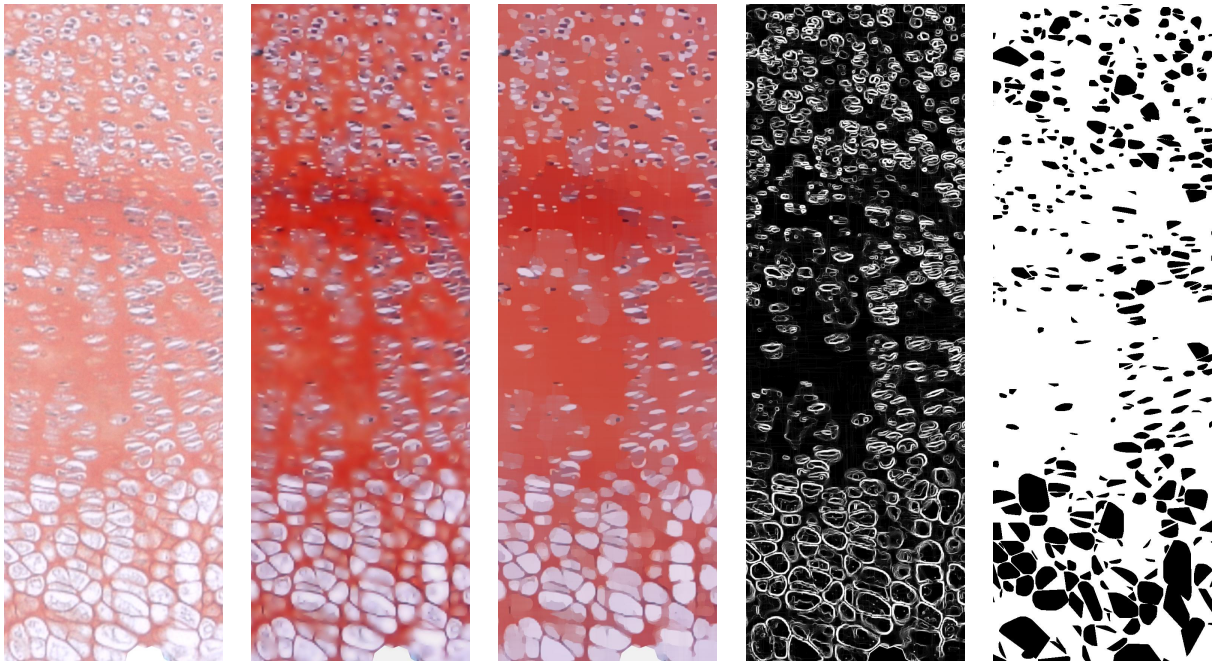



(a)




(b)

Figure 5: In (a), a fourth degree polynomial approximates the graph of the mean isoperimetric constant for horizontal strips down a growth plate image. In (b), the zone boundaries predicted by inflection points of the polynomial accurately divide the growth plate.



(a) Input Image (b) Retinex (c) Cartoon Component (d) Gradient (e) Output Image

Figure 6: A cropped section of a growth plate image at each step in the algorithm.

zones.

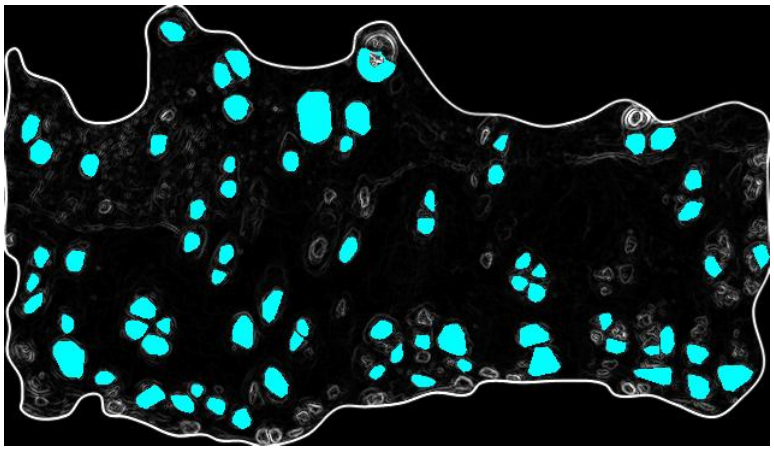
## 4 Results

Figure 6 shows a growth plate image after each step of the final algorithm. Figure 6(a) is a section of an input image and 6(b) is the result of the Retinex portion of the algorithm. Figures 6(c) and 6(d) are the cartoon component and gradient of the image, respectively. Finally, Figure 6(e) is the result of thresholding and the convex hull operation, and is the final output of the algorithm. This growth plate excerpt includes chondrocytes in the resting zone, proliferating zone, and hypertrophic zone, demonstrating the effectiveness of the algorithm across regions with different characteristics.

Figure 8 shows the results of our algorithm applied to a type of growth plate different than the growth plates used to develop and test the algorithm. The cell detections overlaid on the gradient image in Figure 8(b) demonstrate the accuracy of our method, even when applied to different classes of growth plates.

Figure 7: Output of algorithm when applied to entire growth plate image.

(a) Input Image



(b) Overlay of Cell Detections

Figure 8: Algorithm results from a tibia growth plate image.

# 5 Error Analysis

In order to create ground truths for cell detection, we manually fit ellipses and polygons to chondrocytes which fit our criteria for detection. The manual detection we used as the basis for comparison. Our algorithm was then able to be tested using well known clustering metrics and statistics.

In this section we define the classes $\{L_j\}$ in our ground truth image S=$\{l_1, \ldots, l_N\}$ and clusters $\{C_i\}$ in our segmented image $S^{'}$=$\left\{l_1^{'}, \ldots, l_N^{'}\right\}$.

## 5.1 Clustering Purity

Purity is a simple way of measuring the accuracy of a clustering algorithm [1]. It is defined as:

$$P = \sum_i \frac{|C_i|}{N} \max_j \frac{|C_i \cap L_j|}{|L_j|}. \tag{12}$$

The main drawback of this statistic is that it does not penalize segmenting a class into multiple clusters nor clusters which cover multiple classes. The trivial segmentations of every pixel as its own cluster and segmenting the whole image as one cluster both maximize the purity of the image.

## 5.2 Rand Index

One of the most well known clustering metrics, the Rand Index, looks at the percentage of pixel pairs that are mapped from either the same class to the same cluster, or from different classes to different clusters. [15]

$$R(S, S^{'}) = \frac{1}{\binom{N}{2}} (|A| + |B|) \tag{13}$$

where $A = \left\{(i,j)|i \neq j, I_i = I_j, I_i^{'} = I_j^{'}\right\}$ and $B = \left\{(i,j)|i \neq j, I_i \neq I_j, I_i^{'} \neq I_j^{'}\right\}$.

## 5.3 Adjusted Rand Index

The Adjusted Rand Index is a normalization of the Rand Index, which accounts for the expected Rand Index of a random clustering compared to our images. This expected value of the Rand Index, becomes our normalized zero-value, giving us an estimate of how much better our clustering is compared to a randomly chosen clustering [15].

$$AR(S,S') = \frac{R(S,S') - \mathbb{E}(R(S,S'))}{Max(R(S,S')) - \mathbb{E}(R(S,S'))} = \frac{\sum_{i,j}\binom{|C_i \cap L_j|}{2} - \frac{\sum_i |C_i| \sum_j |L_j|}{\binom{N}{2}}}{\frac{\sum_i |C_i| + \sum_j |L_j|}{2} - \frac{\sum_i |C_i| \sum_j |L_j|}{\binom{N}{2}}} \tag{14}$$

The equivalence of the above expressions can be algebraically derived.

### 5.4 Normalized Information Distance

Lastly, we can use entropy,

$$H(S) = -\sum_i \frac{|C_i|}{N} \log\left(\frac{|C_i|}{N}\right), \tag{15}$$

and conditional entropy,

$$H(S|S') = -\sum_i \sum_j \frac{|C_i \cap L_j|}{N} \log\left(\frac{|C_i \cap L_j|}{L_j}\right), \tag{16}$$

to measure the accuracy of our clusterings. The mutual information $I(S,S')$=$H(S)$-$H(S|S')$, measures how much the the uncertainty in one variable is decreased by knowing the other, essentially how dependent the distributions of the two are. Using these statistics we can calculate the Normalized Information Distance metric for these images: [17]

$$d_{max} = 1 - \frac{I(S,S')}{max(S,S')} \tag{17}$$

### 5.5 Results of Error Analysis

The above algorithms were programmed into MATLAB, and run on a selected rectangle of our processed image (the same rectangle used to create our ground truth image). This subsection of the image was then compared to our ground truth and the results are tabulated below.

| Error Measure | Rand Index | Adj. Rand Index | Purity | Norm. Info. Dist. |
|---|---|---|---|---|
| Maximum Value | 0.7434 | 0.3447 | 0.8867 | 0.6771 |

As we can see, The purity is the highest of these statistics, indicating that the algorithm is identifying most of the desired cell pixels. The Rand Index having a slightly lower value than this is due in part to the splitting of cells in some of our clusters, which is heavily penalized under this measurement. Similarly, the comparatively lower Adjusted Rand Index and Normalized Information Distance indicate that there is a moderate correlation between the images. Some of this error results from the ambiguity and inaccuracy of our ground truth image, since ellipses do not perfectly

approximate the cells in our original image. There is also ambiguity as to which cells are in the foreground of the image.

# 6 Conclusion

We have developed a specialized set of algorithms implemented in MATLAB which can be used for the automated detection of chondrocytes in growth plate images. The data extracted from these images can be used to analyze the position and shape of the chondrocytes as needed for the biological analysis. This algorithm results in an accurate segmentation of the input image, as well as accurate identification of the resting zone, proliferating zone, and the hypertrophic zone. The results enable simplification and expedition of the process of cell detection in biological research.

# 7 Acknowledgements

We would like to acknowledge the Nonlinear Diffusion Toolbox created by Frederico D'Almeida, which is available through MathWorks.com. The authors would also like to thank Dr. Andrea Bertozzi.

# A Manual Detection Using Xara X 1.1

This is a step by step guide for operating the program Xara X 1.1 for the purpose of processing images of cells in epiphyseal growth plates. The end result will be a Xara file consisting of multiple image layers representing the original image and its desired characteristics identified.

1. Open Xara X 1.1.
2. Go to *File*, and then from the drop down menu choose *Import...*
3. The *Import File* box should appear. Locate the folder on the computer where the image you wish to edit is stored. Select the desired image by clicking once on it and pressing the *Open* button.
4. Be sure the image is selected on the page. If not, select it by clicking once on the image. Along the top menu bar, locate the $X$ and $Y$ coordinate measurements. These display and control the coordinates of the lower left corner of the image. Change the values for both $X$ and $Y$ to 0.0 pix.

5. Go to *Utilities*, then *Galleries*, then select *Layer Gallery* to turn it on. The Layer gallery should appear as a box on the screen. There should be one layer with the generic name*Layer 1*. Right click on *Layer 1* in the *Layer gallery*. Select *Properties....* When the *Layer properties* box appears, rename the layer as *Original Image*.

6. Within the *Layer gallery*, click *New...* in order to create a new layer. A prompt will appear querying the name of your new layer. Create layers until there is a total of five layers with the following names: *Original Image, Background, Rectangle, Cell Membranes,* and *Nuclei*.

7. In order to edit any of the layers, you need to go into the *Layer gallery* and make sure that only the layer you wish to edit is currently selected. The first column of checkboxes allows you to visibility of the different layers. The second column of checkboxes allows you to make different layers available for editing. Whenever editing a layer, that layer needs to be the only one that is editable - there should be a checkmark next to it in the second column. No other layers should have a check in the second column.

8. Begin by editing the *Background* layer. Change your mouse from the *Selector Tool* to the *Shape Editor Tool*. Both tools can be found in the vertical toolbar located on the left side of the screen. Begin by

# B Cropping Images Using Xara X 1.1

This is a step by step guide for operating the program Xara X 1.1 for the purpose of specifically cropping images of epiphyseal growth plates. The end result will be a TIFF file consisting of a version of the orginal image in which the non-growth-plate-region has been cropped.

1. Open Xara X 1.1.

2. Go to *File*, and then from the drop down menu choose *Import...*

3. The *Import File* box should appear. Locate the folder on the computer where the original growth plate image you is stored. Highlighting the desired file, then click the *Open* button.

4. Be sure you select the image on the page. If it is not selected, do so by clicking once on the image. Along the top menu bar, locate the $X$ and $Y$ coordinate measurements. These display and control the coordinates of the lower left corner of the image. Change the values for both $X$ and $Y$ to 0.

5. Begin by changing your mouse from the *Selector Tool* to the *Shape Editor Tool*. Both tools can be found in the vertical toolbar located on the left side of the screen. Change the line width to 2.0 in order to make the lines you will create easier to see. Click and create points

around the edges of the growth plate region. Make sure to complete your shape by connecting your final point to your first point. If you have done this correctly, the shape will be complete and the entire shape will fill with a solid color. Go to the color toolbar at the bottom of the window and select *No Fill*.

6. Next switch back to the *Selector Tool*. Click outside the image. Next select *both* the original image and your drawn outline, using an operation such as *Ctrl + A*. Right click and go to *Combine Shapes...*, then select *Intersect Shapes*. The result will be the growth plate region cropped out of the original image.

7. Since you have obtained your desired image, the next step is to save it so you can run your algorithm on it. Saving by just using the general *Save* button will result in a .xar file type. Thus, rather than saving, instead go to *File*, then select *Export*. Complete the dialogue box by entering your desired file name, and most importantly, setting the *File Type* to be TIFF. Then click the button to export your image. You now have a cropped version of your growth plate region image ready to process.

# C Metamorph

In order to run the Metamorph Software to collect cell data from our images, certain procedures (detailed below) must be followed in order to guarantee an accurate and consistent analysis.

## C.1 For Images in Xara X Files

1. Open the Xara X (.xar) file containing the image
2. Make sure only the appropriate layer is selected
3. Select [File>Export] and export as a .tif file with 300 dpi and no compression
4. Open the exported .tif file in Metamorph [File>Open]
5. Select [Measure>Calibrate Distance>Pixels>Apply]
6. Select [Measure>Set Color Threshold>(Select a Color Range)>Apply]. For our Images we either selected a 168-255 or 243-255 range for the Red, Green, and Blue thresholding values to get a good bounds on the cell size. Alternatively if the cells are entirely composed of the *exact* same color pixels, we can easily identify all cell pixels in Metamorph by selecting [Measure>Set Color Threshold>Set by Example>(Clicking on a cell pixel in Metamorph)>Apply]. In either case, the selected pixels will turn orange.

7. We can then collect the data by selecting [Measure>Integrated Morphology Analysis>(Select the boxes for Area, Perimeter, X-Centroid, Y-Centroid, and Orientation)>Measure]. Once the data is collected the cell pixels should turn green.

8. To export the data to an Excel Spreadsheet select [Open Log>Dynamic File>(input desired sheet name)]. This should open an excel sheet and establish a connection between the programs. To transfer the data select the Log Data button.

### C.2 For Images not in Xara X

If the image is not in Xara X it can either be opened in Xara X, in which case the above procedure can be followed from Step 2. If the file format is compatible (e.g. .tif, .jpeg) the file may be opened directly in Metamorph, in which case the above procedure can be followed from Step 5.

If the image opened includes pixels outside of the growth plate we want to analyze, we can crop this out in Metamorph by creating a region. Select [Trace Region Tool>(Click Around Perimeter of the Desired Area)> (Double Click on Starting Point to Close Region)]. To undo a segment created, right click with the mouse. After the region is closed the perimeter should flash. Then select [Measure>Region Measurements]. From this point we can follow the above procedure from Step 5.

## References


[1] E. Amigó, J. Gonzolo, and J. Artiles. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12, May 2009.

[2] M-G Ascenzi, C. Blanco, I. Drayer, H. Kim, R. WIlson, K.N. Retting, K.M. Lyons, and G. Mohler. Effect of localization, length and orientation of chondrocytic primary cilium on murine growth plate organization. *Journal of Theoretical Biology*, 285(1):147–155, September 2011.

[3] M-G Ascenzi, M. Lenox, and C. Farnum. Analysis of the orientation of primary cilia in growth plate cartilage: a mathematical method based on multiphoton microscopical images. *Journal of Structural Biology*, 158(3):293–306, June 2007.

[4] A.E. Carpenter, T.R. Jones, M.R. Lamprecht, C. Clarke, I.H. Kang, O. Friman, D.A. Guertin, J.H. Chang, R.A. Lindquist, J. Moffat, P. Golland, and D.M. Sabatini. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(10), 2006.

[5] T.J. Collins. Imagej for microscopy. *BioTechniques*, 43, July 2007.

[6] D. Fenistein, B. Lenseigne, T. Christophe, P. Brodin, and A. Genovesio. A fast, fully automated cell segmentation algorithm for high-throughput and high-content screening. *Cytometry Part A*, 73A:958–964, 2008.

[7] I.M. Helmy and A.M. Abdel Azim. Efficacy of imagej in the assessment of apoptosis. *Diagnostic Pathology*, 7(15), 2012.

[8] E.H. Land. The retinex. *American Scientist*, 52(2):247–264, 1964.

[9] E.H. Land and J.J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61:1–11, 1971.

[10] S.P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982.

[11] Y. Meyer. *Oscillating patterns in image processing and nonlinear evolution equations: the fifteenth Dean Jacqueline B. Lewis memorial lectures.* American Mathematical Society, 2001.

[12] J.M. Morel, A.B. Petro, and C. Sbert. A pde formalization of retinex theory. *IEEE Transactions on Image Processing*, 19(11):2825–2836, November 2010.

[13] H. Schaeffer and S. Osher. A low patch-rank interpretation of texture. CAM Report, November 2011.

[14] C.A. Schneider, W.S. Rasband, and K.W. Eliceiri. Nih image to imagej: 25 years of image analysis. *Nature Methods*, 9:671–675, 2012.

[15] R. Unnikrishnan and M. Hebert. Measures of similarity. *Seventh IEEE Workshop on Applications of Computer Vision (WACV)*, pages 394–400, January 2005.

[16] L.A. Vese and S. Osher. Modeling textures with total variation minimization and oscillating patterns in image processing. *Journal of Scientific Computing*, 19(1-3), December 2003.

[17] N.X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(2837-2854), 2010.

[18] W. Yin, D. Goldfarb, and S. Osher. Image cartoon-texture decomposition and feature selection using the total variation regularized l1 functional. *Variational, Geometric, and Level Set Methods in Computer Vision. Lecture Notes in Computer Science*, 372:73–84, 2005.