12-2010

Appearance Based Stage Recognition of Drosophila Embryos

Gopi Chand Nutakki
Western Kentucky University, gopichand.nutakki407@topper.wku.edu

Follow this and additional works at: http://digitalcommons.wku.edu/theses

Part of the Computational Biology Commons, Genetics Commons, and the Numerical Analysis and Scientific Computing Commons

Recommended Citation
http://digitalcommons.wku.edu/theses/217

This Thesis is brought to you for free and open access by TopSCHOLAR*. It has been accepted for inclusion in Masters Theses & Specialist Projects by an authorized administrator of TopSCHOLAR*. For more information, please contact topscholar@wku.edu.
APPEARANCE BASED STAGE RECOGNITION OF DROSOPHILA EMBRYOS

A Thesis
Presented to
The Faculty of the Department of Mathematics and Computer Science
Western Kentucky University
Bowling Green, Kentucky

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

By
Gopi Chand Nutakki
December 2010
APPEARANCE BASED STAGE RECOGNITION OF DROSOPHILA EMBRYOS

Date Recommended  December 1, 2010

Dr. Qi Li
Director of Thesis

Dr. Guangming Xing

Dr. James Gary

Signature: Richard A. Brooks  Jan 6, 2011
Dean, Graduate Studies and Research  Date
ACKNOWLEDGEMENTS

It was a great pleasure working under Dr. Qi Li for the success of this Thesis. I would like to thank Dr. Qi Li for the immense trust and patience he has over me.

I would like to thank Dr. Guangming Xing and Dr. James Gary for their valuable suggestions that helped me improve this Thesis.

I would like to thank my family for their never ending support, specially my brother who always sets the benchmarks for me and pushes me to the next higher level.
# TABLE OF CONTENTS

1 INTRODUCTION 3

2 IMAGE STANDARDIZATION 6

2.1 Embryo Extraction ............................................. 6
2.2 Isolating the main embryo ..................................... 6
2.3 Image registration ............................................. 8

3 FEATURE EXTRACTION 10

3.1 RGB Intensity Image Features ................................. 10
3.2 Orientation histogram ....................................... 12
3.3 Gabor Filter Features ....................................... 13

4 DIMENSION REDUCTION 19

4.1 Principal Component Analysis ............................... 19
  4.1.1 Covariance ............................................... 22
  4.1.2 SVD of Covariance ..................................... 23
  4.1.3 Relation with Eigen-Analysis ........................... 27
4.2 Liner Discriminant Analysis ................................. 28

5 EXPERIMENTS 30

5.1 Classifier ...................................................... 30
5.2 RGB Intensity ................................................. 32
5.3 Orientation Histogram ....................................... 35
5.4 Gabor Filter features .................................................... 38
5.5 RGB vs Orientation Histogram vs Gabor ............................. 41
   5.5.1 LDA on Orientation histogram, RGB and Gabor features .... 42

6 CONCLUSION AND FUTURE WORK ............................... 44
LIST OF FIGURES

2.1 BDGP embryo images varying in scales and orientations, which are to be registered to a standard scale and orientation. ........................................... 7

2.2 Respective Registered images of the embryos shown in Figure 2.1 ........ 8

3.1 Demonstration of a block of $4 \times 4$ sub-blocks used to compute an orientation histogram, where an arrow indicates a gradient vector. ................. 14

3.2 An orientation histogram is a 8-dimensional vector, where each dimension represents a bin associated with a specific directional degree, from $0^\circ$ to $270^\circ$ with the step of $45^\circ$. ...................................................... 14

3.3 Gabor function visualization from $0^\circ$ to $135^\circ$. ............................... 16

3.4 Gabor features extraction from the image using the Gabor Mask. .......... 18

4.1 Spacial Distribution of the Data ......................................................... 21

4.2 Principal Components, Z1 and Z2 ..................................................... 21

4.3 Projection of the data on to the first PC ............................................ 21

4.4 Visualization of Eigen Embryos for 8 different BDGP Embryo images. .. 23

5.1 Visualization of a BDGP embryo at different stages of its development. ... 31

5.2 The RGB Features reduced by PCA. Accuracy for the number of Principle Components from 2 to 20 and Nearest Neighbors from 1 to 5. ............. 33

5.3 The optimal RGB Feature Accuracies for Nearest Neighbors from 1 to 5. ... 33

5.4 Accuracy comparison of RGB Image features, PCA optimal values when classification is performed under 5 Nearest Neighbors. Accuracy for PCs $= 2$ to 20. .................................................. 33
5.5 The Orientation histograms reduced by PCA. Accuracy for the number of Principle Components from 2 to 20 and Nearest Neighbors from 1 to 5. . . . 36

5.6 The Orientation histograms for Nearest Neighbors from 1 to 5. . . . . . . . 36

5.7 Accuracy comparison of Orientation histograms, PCA optimal values when classification is performed under 1 Nearest Neighbors. Accuracy for PCs = 2 to 20. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37

5.8 The Gabor Features reduced by PCA. Accuracy for the number of Principle Components from 2 to 20 and Nearest Neighbors from 1 to 5. . . . . . . . 39

5.9 The Gabor Features for Nearest Neighbors from 1 to 5. . . . . . . . . . . . 39

5.10 Accuracy comparison of Gabor Image features, PCA optimal values when classification is performed under 5 Nearest Neighbors. Accuracy for PCs = 2 to 20. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39

5.11 Accuracy comparison of RGB, Orientation histogram, Gabor . . . . . . . . 42


<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Accuracy comparison of RGB, Reduced RGB with PCA at the optimal number of Principal Components for number of neighbors from 1 to 5</td>
<td>34</td>
</tr>
<tr>
<td>5.2</td>
<td>Time cost comparison of RGB features, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features</td>
<td>34</td>
</tr>
<tr>
<td>5.3</td>
<td>Accuracy comparison of orientation histogram, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5</td>
<td>37</td>
</tr>
<tr>
<td>5.4</td>
<td>Time cost comparison of orientation histogram, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features</td>
<td>37</td>
</tr>
<tr>
<td>5.5</td>
<td>Accuracy comparison of Gabor, PCA of Gabor features at the optimal number of Principal Components for number of neighbors from 1 to 5</td>
<td>40</td>
</tr>
<tr>
<td>5.6</td>
<td>Time cost comparison of Gabor features, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features</td>
<td>40</td>
</tr>
<tr>
<td>5.7</td>
<td>Accuracy comparison of RGB, orientation histogram, Gabor</td>
<td>42</td>
</tr>
<tr>
<td>5.8</td>
<td>Time cost comparison of RGB, Orientation histogram, Gabor</td>
<td>42</td>
</tr>
<tr>
<td>5.9</td>
<td>Accuracy comparison of RGB, orientation histogram, Gabor after applying LDA</td>
<td>43</td>
</tr>
</tbody>
</table>
Stages in Drosophila development denote the time after fertilization at which certain specific events occur in the developmental cycle. Stage information of a host embryo, as well as spatial information of a gene expression region is indispensable input for the discovery of the pattern of gene-gene interaction. Manual labeling of stages is becoming a bottleneck under the circumstance of high throughput embryo images. Automatic recognition based on the appearances of embryos is becoming a more desirable scheme. This problem, however, is very challenging due to severe variations of illumination and gene expressions. In this research thesis, we propose an appearance based recognition method using orientation histograms and Gabor filter. Furthermore, we apply Principal Component Analysis to reduce the dimension of the low-level features, aiming to accelerate the speed of recognition. With the experiments on BDGP images, we show the promise of the proposed method.
CHAPTER 1: INTRODUCTION

The role of Drosophila in explicating the function and interconnection of animal genes has established the species as a major model organism [8, 13, 16]. In situ hybridization is a recent technique to document gene expression pattern of individual embryos along their different developmental periods [14]. These documents, a set of embryos images contain rich information on the spatio-temporal patterns that are extremely valuable for the study of gene-gene interaction. Given two standardized images of embryos at the same developmental stage, the interaction strength of two genes can be quantified by computing the similarity of expression patterns, e.g., the ratio of overlapping expression regions of the images [7]. Stages in Drosophila development denote the time after fertilization at which certain specific events occur in the developmental cycle. Embryogenesis is traditionally divided into a series of consecutive stages manually distinguished by morphological markers [2]. The manual labeling of stages is becoming a bottleneck with the dramatically increasing data, e.g., the dataset contributed by The Berkley Drosophila Genome Project (BDGP) [14].

Appearance based stage recognition recently received attention [17] because of its potential in accelerating the pattern discovery of gene-gene interaction. There are two main strategies for appearance based recognition [9]: i) global appearance based, and ii) local appearance based. The former strategy assumes that an embryonic image is standardized with the same orientation and same scales and an image is represented as a single feature vector used to the similarity measure of images. The latter strategy does not have this assumption, but it requests an interest point detector [9] to localize a set of distinct local features that usually correlate with certain geometric/structural information. An important observation of embryonic images is that their appearances are dominated by textural information, which challenges the local appearance based strategy. The recognition method proposed in [17] is under the strategy of global appearance. In this
thesis, we will also focus on this strategy.

An important step in the global appearance based recognition is image standardization [7] that aims to establish pixel-to-pixel correspondence among embryonic images so that the automatic results from the remaining steps can be interpreted statistically and biologically. Raw embryonic images contain imaging variations, including illumination, orientation, scale, pose/view and the existence of touching embryos (certain parts of the boundary of an embryo is touched by neighboring embryos). After image standardization, background and neighboring embryos will be removed, and the orientation and the size the embryo of interest will also be normalized as same degree and dimension, respectively. Image standardization can be performed manually or automatically. We manually standardize embryonic images in the study. It is worth noting that image standardization is also valuable for gene-gene interaction study because the established pixel-to-pixel correspondence can help quantify the spatial relationship (e.g., overlapping regions) among gene expression patterns in embryos in the same developmental stage.

With standardized images, a simple method to construct low-level feature vectors for stage recognition is to use RGB intensity. However, there are two serious limitation of this method: 1) it is sensitive to illumination variation, and 2) it is sensitive to the localization error. It has been known that a single pixel shift of an image may lead to significant change of the Euclidean distance between the image and a reference image. Orientation[10] and Gabor[33] feature are two features that are robust with respect to illumination change. In this thesis, we design orientation and Gabor filter based methods to construct robust low-level features for stage recognition.

A representation of low-level features is generally in a high-dimensional space. High-dimensional representations may contain information that is irrelevant to a specific task (thus called noise). More seriously, high-dimensional representation will cause high computational cost for a recognition task that involves intensive computation of distance
Linear projection is a popular scheme to reduce the dimension of low-level features [36]. By applying matrix decomposition techniques to statistical information of training data (i.e., a collection of low-level feature representation with known information of stages), a linear projection method can construct a low-dimensional space to data representation. Many linear projection methods have been proposed, based on different criteria. In this thesis, we will study two most popular ones: i) Principal Component Analysis (PCA) [19, 20, 21, 22, 37, 38] that aims to maximize the variance of training data, and ii) Linear Discriminant Analysis (LDA) [40] that aims to maximize between-class and minimize within-class distance of training data.

The rest of the thesis is organized as follows: Chapter 2 describes image standardization that is an important step for global appearance based stage recognition. Chapter 3 presents three low-level features: i) RGB intensity, ii) orientation histogram, and iii) Gabor filter. Chapter 4 presents two linear projection methods to reduce the dimension of the low-level feature representation. Chapter 5 presents the experimental study on the effectiveness of low-level features in terms of classification accuracy and the effectiveness of dimension reduction methods in terms of computational time. Chapter 6 gives a conclusion and a future direction of this research.
CHAPTER 2: IMAGE STANDARDIZATION

Image standardization is an important step in order to provide a reliable and dimensionally synchronized image database[7] for the classification of the stage of embryonic images. To deal with issues such as noise, occlusion, and inconsistent orientation in the embryo images. The embryonic images are captured in the liquid environment, which can exaggerate the effect of illumination variation.

2.1 Embryo Extraction

With very few exceptions, the embryos and the background have significantly different local texture properties. Embryos have a rougher texture with high local variance, while the background, a watery solution has smooth tonal variations, which means pixels with low local variance. The variance of pixel intensity in a window of a given size (say 3 × 3), centered at each pixel of the image is calculated and the pixel is set as foreground if the value is above a fixed threshold value. It is quite common to have embryo-pixels assigned as background, mainly at the center region of them. Thus, after obtaining the binary image, a morphological binary operator is applied to “fill the holes” inside the embryo’s region. In our research we manually isolated the embryo by extracting the embryo using graphical tools and recreating an embryo by following few steps.

2.2 Isolating the main embryo

An embryo image taken during a typical embryogenesis experiment usually contains a few dozen embryos, with the most interesting embryo located at the center of the image. The embryos in an image may be touching the main embryo, or even occluding it. To extract the main embryo, we can use the well known, “watershed transform” to partition the foreground region of the binary image. However, to the noisy borders and concave
Figure 2.1: BDGP embryo images varying in scales and orientations, which are to be registered to a standard scale and orientation.

shapes of the embryos, the watershed approach with a bad initial state tends to “over-segment” the embryos. We can perform a shrink-expand processing of the foreground region, first the region is continuously eroded until we find two separated regions. The two partitions of the foreground region are then the initial state for the watershed flooding algorithm. The algorithm “grows” back the regions, until they are tough again, creating a watershed. For an image with more than two embryos the “shrink-expand” algorithm is recursively applied over the foreground region, keeping only the center-most region at each recursion step until the “shrink-expand” algorithm gives only one region. For our research we applied a straight forward manual method to extract the main embryo, we used the MATLAB image processing tool to set the boundaries of the embryo and then extract the region that is highlighted. Points are set along the boundary of the embryo forming an enclosed polygon which is close to the shape of an ellipse.
Figure 2.2: Respective Registered images of the embryos shown in Figure 2.1

2.3 Image registration

Embryos extracted could have different position, orientation, scale and shape in the image. For a better comparison between patterns in the extracted embryos, we can perform an image registration step to transform the images, so that the comparison can be performed regardless their original position, orientation, scale and shape. Since our research is specifically designed, ignoring the orientation of the embryo and considering all the embryos to be at a standard scale it is very crucial to have all the embryos pre-normalized with respect to the major axis and the head and tail. Once the embryo is highlighted in the original image, a new image with required dimensions having a white background is created and the extracted embryo is placed in the center of the new image with the major axis along the horizontal. The head and tail normalization is not automated.

Given that embryos at the same developmental stages do not show considerable differences on shape, special care has to be taken with the orientation of head and tail, head should be on the left side of the image and tail being on the right side of the image. We have to take care of the method so that after registration, each embryo will have an
ellipsoidal shape, predefined size, and with its major axis aligned horizontally. The final product of our image processing are gray-level images with $320 \times 240$ pixels. It is very important to realize that precise registration of the images is not always achieved by automated process. One challenge task in the consecutive step of feature extraction is to design features robust with respect to imprecise registration.
CHAPTER 3: FEATURE EXTRACTION

This chapter contains three parts. In the first part we present a RGB intensity based feature extraction method. In the second part, we present an orientation histogram method as the low-level features invariant to illumination. The orientation histogram we implemented here is basically the same as the one used in SIFT (Scale Invariant Feature Transform) [10], but note that the input embryonic images for stage recognition are expected to be standardized in the context of the global appearance based recognition. In other words, there is no scale variation. To be focused on the illumination variation, we use the term, orientation histogram, instead of SIFT. In the third part we present another set of features obtained using the Gabor filter, this technique follows the same setting as the orientation histogram, a Gabor mask slides over the image obtaining the co-efficients which populate the image features, the co-efficients are not accumulated. In the next chapter, we present a statistical learning method to reduce the dimension of the RGB features, orientation histogram and the Gabor features. The classification of the embryos is based on these features, a given embryo is classified by finding the nearest embryo using the k-Nearest Neighbor algorithm.

3.1 RGB Intensity Image Features

RGB intensity is the value of a given pixel which refers the percentages of Red, Green, Blue in the complete spectrum of the pixel color. The RGB feature of an image is based on the intensity values alone. It is a straight forward resemblance check of a given image to another image.

Each pixel in the image is processed to obtain the feature of the image. The image is subdivided into a number of blocks and each block is processed individually to obtain a semi feature which becomes a part of the complete image feature. In our experiments we divided the image in to $4 \times 4$ blocks. Each block is then processed and is standardized into
a 15 × 15 sized window. The window is a matrix of the given dimension. The matrix is then converted into a liner vector which form the feature of the image sub block. The length of the sub block image feature will be 225. All the sub block image features can be concatenated to obtain a single image feature for the complete image.

When the image is initially read, the image is converted into a double matrix which is of equal dimensions as that of the image. The double value in the matrix denotes the RGB intensity value of a given pixel. It is very important to capture as much detail as possible to attain better results. Starting from the first block of the image, the block of pixels is standardized to the window. During the standardizing procedure the pixels will overlap, in that case the intensity is accumulated and then finally the image feature is normalized to obtain a unit vector which is then compared with the other unit vectors of the images from the database.

The embryo images are of dimensions 320 × 240, this image will be divided into 4 × 4 blocks which will result in 16 blocks of size 80 × 60 each. Each block will be standardized to a window of 15 × 15. We start from the first block, go to each point/pixel of the image and based on the location of the pixel in the block, we find the corresponding location in the window. Since the block is of dimension 80 × 60, the first pixel of the top row of the block will go to the first location of the top row of the window and the pixel at the 80 location will go to the location 15 of the top row of the window. All the pixels in between will be adjusted in the window across 1 and 15.

Each block will be represented by a window of dimension 15 × 15 when converted to a linear vector it will have a length of 225. This 225 length vector represents the block. For the total image divided into 4 × 4 we get 16 such vectors. All the vectors are concatenated one at the end of the other and we obtain a single long vector of length 16 × 225. This vector is again normalized to obtain a unit vector for comparisons with other image vectors.
The images are now represented in a vector format. This vector is then crosschecked with the other vectors to find the closest or the most resembling vector thus finding the image which looks similar. When the images are labeled with their developmental stage numbers, the developmental stage of the image would be the stage number of the most closest image.

3.2 Orientation histogram

Orientation histogram works very close to the way the SIFT algorithm works. In order to preserve the maximum possible detail of the embryo, instead of considering only the keypoints as SIFT, the orientation histogram processes the entire image pixels and stores the information in the histogram.

Orientation histogram is a low-level statistical representation of a local region. The motivation of using statistical representation comes from the texture dominant appearance of embryos. One implementation of orientation histogram is based on gradient vectors, as shown in Figure 3.1. The upper $4 \times 4$ window is a zoom-in illustration of a sub-block in the lower window. A rectangle in the upper window represents a pixel, and the associated arrow denotes the gradient vector of the pixel. Note that the direction of the gradient vector represents the local orientation of the pixel, and the length of the vector represents the magnitude of the local variation of pixel values.

A way to construct an orientation histogram is to accumulate the gradient magnitudes in the same direction, as illustrated in Figure 3.1 (the lower window), and Figure 3.2. In the lower window of Figure 3.1, an arrow represents the accumulation of gradient magnitude of the same direction. Similar to [10], we discretize the angular space from 0° to 360° by the step of 45°. Therefore, we have 8 bins, as shown in Figure 3.2, to accumulate the gradient magnitudes. The algorithm proceeds with taking the gradient and orientation of each pixel of the image and accumulating the gradient magnitude to the
respective orientation bin. Each block will have 8 bins representing the 8 angles which are populated with the gradient magnitudes. Each image is divided into $16 \times 16$ blocks each block will give out 8 bins. For each pixel in a given block, the gradient magnitude and the orientation of the pixel are calculated.

The gradient is the square-root of the squares of the pixel value difference around a given block pixel. The orientation is the arc-tan of the vertical and horizontal pixel value differences. Once the gradient magnitude and the orientation for a pixel is calculated, the gradient magnitude of all the pixels is added to the corresponding bin of the block to which the pixel belongs to. All the bins from all the blocks are then concatenated to obtain the complete image feature which has a length of 8 times the number of blocks. The feature is extracted using each pixel of the image which gives an impression that most of the image detail is preserved in the feature. The more the information preserved the more distinct is the feature to the other image features. The feature is finally normalized to convert the feature into a form of unit vector which helps in obtaining the similarity between the feature vectors. Once the feature vector for an image is obtained, the vector dot product is applied with the other image features.

The strategy used in the construction of SIFT [10], concatenating multiple orientation histograms of multiple regions, compared with using a single orientation histogram, has the following two advantages: i) it is more robust to inaccurate localization, and ii) it tends to preserve more information. But the second advantage is somehow subtle, since using a larger number of sub-regions not only increases the dimension of the low-level feature, but also involves more noisy information.

### 3.3 Gabor Filter Features

Gabor filter has significant application in the signal processing [28] and it is also proven in the recent past that it can be applied in image processing for head pose
Figure 3.1: Demonstration of a block of $4 \times 4$ sub-blocks used to compute an orientation histogram, where an arrow indicates a gradient vector.

Figure 3.2: An orientation histogram is a 8-dimensional vector, where each dimension represents a bin associated with a specific directional degree, from $0^\circ$ to $270^\circ$ with the step of $45^\circ$.

Identification[29], scene analysis[30], human identification[31], etc. It is also applied in bio-medical physics and geophysics to better understand the signals[32].

Texture is an important feature of an image. Spatial frequencies and their orientations are important characteristics of textures in images. The frequency characteristics of images can be analyzed using spectral decomposition methods like Fourier analysis[33]. Recently the multichannel Gabor decomposition has become very popular for texture analysis. Gabor filter resembles the characteristics of simple visual cortical cells and is widely used to extract texture features from images for either texture segmentation or image classification and for image retrieval algorithms. The most successful results has shown that image retrieval using Gabor features outperforms other
popular techniques like pyramid-structured wavelet transform (PWT) features[34],
tree-structured wavelet transform (TWT) features[34] and multi-resolution simultaneous
autoregressive model (MR-SAR) features. It is also adopted by MPEG-7 as one of texture
descriptors. A number of parameters are used in the Gabor Filter. We used a number of
scales and orientations. In our experiments numbers run for 1 to 3 scales and 8
orientations. Moreover, there is no research found on how to select filter mask size so far.
In this paper, we investigate the accuracy of classification by extracting the image feature
concatenated by the coefficients obtained by the convolution. We also adopted constant
Gabor filter parameters on texture retrieval. In practice, it is a compromise to choose
number of filters.

Gabor filters have been used in many applications, such as texture segmentation,
target detection, fractal dimension management, fingerprint matching, edge detection,
image coding and image reconstruction. Gabor filter is a linear filter that is created by
modulating a sinusoid with a Gaussian. Figures: 3.3(a)- 3.3(d) show the visualization of
the Gabor function under a constant variables except the orientation varying from $-\pi$ to $\pi$.

$$g(x, y; \lambda, \theta, \phi, \sigma, \gamma) = e^{-\left(\frac{x'^2 + y'^2}{2\sigma^2}\right)} \cos\left(2\pi \frac{x'}{\lambda} + \theta\right)$$

where

$$x' = x\cos(\theta) + y\sin(\theta)$$

$$y' = -x\sin(\theta) + y\cos(\theta)$$

where the arguments x and y specify the position of a light impulse in the visual field and
$\sigma, \gamma, \lambda, \theta$ and $\phi$ are parameters as follows:

- $\sigma$ is the standard deviation of the Gaussian factor and determines the (linear) size of
  its receptive field.
- $\lambda$ specifies the wavelength of the cosine factor of the Gabor filter.
• $\theta$ specifies the orientation of the normal to the parallel stripes of the Gabor filter.

• $\phi$ is the phase offset of the cosine factor and determines the symmetry of the Gabor filter.

• $\gamma$ is called the spatial aspect ratio and specifies the ellipticity of the Gaussian factor.

The Gabor filter we implemented follows the same path as the Orientation Histogram in obtaining the features. Generating the Gabor filter image feature involves convolution of the image sub blocks with the Gabor mask. The Gabor mask is a simple matrix of a given dimension. A Gabor mask is created and it slides over the image regions obtaining a coefficient for each block. The coefficients are placed at a given position in the image feature. The complete image is convoluted for a total of $S \times O$, where $S$ is the number of scales considered and $O$ is the number of orientations considered. The mask is populated using the Gaussian and the COS functions. The center point of the mask is given more weightage than the points closer to the edges.

Gabor Filter feature is also a low-level statistical representation of a local region and the motivation of using statistical representation comes from the texture dominant
appearance of embryos. Our implementation of the Gabor filter features consists of constructing a Gabor mask and sliding the mask over the image.

A 2D Gabor filter can be realized as a sinusoidal plane wave of some frequency and orientation within a two dimensional Gaussian envelope. Its spatial extent, frequency and orientation preferences as well as bandwidths are easily controlled by the parameters used in generating the filters. However, there is an "uncertainty relation" associated with linear filters which limits the resolution simultaneously attainable in space and frequency. 2D Gabor filters are members of a class of functions achieving optimal joint resolution in the 2D space and 2D frequency domains. They have also been found to be a good model for two dimensional receptive fields of simple cells in the striate cortex[52].

Figure 3.3(a) is the visualization of the Gabor function for the orientation of 0° and the other parameters as constants. The Gabor function is obtained as a convolution of the Gaussian and the COS functions. The resulting Gabor function depicts the dominating, orientation and areas of the image.

When constructing a Gabor filter features we consider multiple scales and orientations. For each scale and orientation combination we create a mask and use the mask to obtain a co-efficient by the convolution of the mask with the image block which is of the same size as the mask. The mask is a matrix of size 16 × 16 and is populated with a double value obtained from the product of the Gaussian function and COS function for a given scale and orientation value. The scale varies from 1 to any positive integer. We considered only 3 scales 1, 2, 3. The orientation varies from 0° to 360° by steps of 45° which gives out 8 different orientations the image is sampled with. The image is totally sampled the number of scales times with the number of orientations. Gabor filter effectively samples the image with different scale and orientation combinations and does it for all the given scales and orientations, thus obtaining a very distinctive image feature for a given image.
Figure 3.4: Gabor features extraction from the image using the Gabor Mask.

Once the mask for a given set of scale and orientation is populated the mask is slid over each block of the image resulting in a set of co-efficients from the convolution of the mask with the image block. All the coefficients resulting from the convolution for all the combinations of scales and orientations are concatenated to obtain the Gabor filter features. Since Gabor filter is very close to the way the human vision system works, the image features obtained using this procedure is highly distinctive for a given image and so the classification results are better than orientation histogram.

Figure 3.4 shows the setup of extracting the image features using the Gabor mask. The image is divided in to number of sub-blocks, each of size equal to the Gabor masks which are of $16 \times 16$ pixel size. A coefficient is resulted when the image block is convoluted with the Gabor mask, this coefficient is used to populate the Gabor image feature. The complete image is convoluted for all the given Gabor mask each obtained from a specific combination of scale and orientation. The features are populated with the coefficient obtained from the convolution of the image block with the Gabor mask.
CHAPTER 4:  DIMENSION REDUCTION

High-dimensional datasets present many mathematical challenges[35] as well as some opportunities, and are bound to give rise to new theoretical developments. One of the problems with high-dimensional datasets is that, in many cases, not all the measured variables are “important” for understanding the underlying phenomena of interest. While certain computationally expensive novel methods can construct predictive models with high accuracy from high-dimensional data, it is still of interest in many applications to reduce the dimension of the original data prior to any modeling of the data.

Dimension reduction aims to reduce the dimensionality of data representation in order to reduce the computational cost in recognition (or other higher level application)[36]. Principal Component Analysis (PCA)[37] is a popular method for dimension reduction. PCA transforms the data into a new coordinate system such that the greatest variance by any projection of the data comes to lie on a subspace consisting of so-called Principal Components that maximize the data variance. PCA is theoretically the optimum transform for given data in least square terms. PCA is unsupervised, and convenient to use in many applications. Singular Vector Decomposition(SVD)[39] is used for both PCA and Linear discriminant analysis[40].

4.1 Principal Component Analysis

The Principal Component Analysis (PCA)[38] is one of the most successful techniques that have been used in image recognition and compression. PCA is a statistical method under the broad title of factor analysis. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to the smaller intrinsic dimensionality of feature space (independent variables), which are needed to describe the data economically. This is the case when there is a strong correlation between observed variables. PCA can be used for prediction, redundancy removal, feature extraction, data
compression, etc. Because PCA is a classical technique which can do something in the linear domain, applications having linear models are suitable, such as signal processing, image processing, system and control theory, communications, etc.

The central idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of a large number of interrelated variables, while retaining as much as possible of the variation present in the data set. This is achieved by transforming to a new set of variables, the principal components (PCs), which are uncorrelated, and which are ordered so that the first few retain most of the variation present in all of the original variables.

Principal component analysis is appropriate when you have obtained measures on a number of observed variables and wish to develop a smaller number of artificial variables (called principal components) that will account for most of the variance in the observed variables. The principal components may then be used as predictor or criterion variables in subsequent analyses. Principal component analysis is a variable reduction procedure. It is useful when you have obtained data on a number of variables (possibly a large number of variables), and believe that there is some redundancy in those variables. In this case, redundancy means that some of the variables are correlated with one another, possibly because they are measuring the same construct. Because of this redundancy, you believe that it should be possible to reduce the observed variables into a smaller number of principal components (artificial variables) that will account for most of the variance in the observed variables.

Specifically, given the data in m-dimensional space. PCA describes the location and space of the m-dimensional data cloud. There are 2 steps involved, translation and rotation of the data cloud to and about the origin respectively. Translation is done by mean clustering the data. If the data is not mean-centered, then the PC axes describes not only the shape of the data but also the location. Rotation is done by aligning the first PC axis along with the longest axis through the data set. A principal component can be defined as
a linear combination of optimally-weighted observed variables.

Figure 4.1 shows the visualization of the spacial distribution of a sample data along
$X_1$ and $X_2$. Each point represents a vector in the space. The data is distributed in the form of an ellipse to better demonstrate the Principal Components. The idea is to find the axis passing through the majority of the data, such axis is called the major axis and we also define a minor axis perpendicular to the axis, which covers lower amounts of the data. As the major axis passes through the majority of data, the data is projected on to the majority axis thus representing the complete data. Figure 4.2 shows the visualization of the majority and minority axes namely $Z_1$ and $Z_2$. Figure 4.3 shows the visualization of the projects of the data points on to the majority axis.

### 4.1.1 Covariance

The standard deviation and variance measures are purely 1-dimensional. Data sets like this could be heights of all the people in the room, marks for an exam etc. Many data sets have more than one dimension, and the aim of the statistical analysis of these data sets is usually to see if there is any relationship between the dimensions. For example, we might have our data set containing both the height of all the students in a class, and the marks they received for a paper. We could then perform statistical analysis to see if the height of a student has any effect on their marks. Standard deviation and variance only operate on 1 dimension, so that you could only calculate the standard deviation for each dimension of the data set independently of the other dimensions. However, it is useful to have a similar measure to find out how much the dimensions vary from the mean with respect to each other. Covariance is such a measure.

Covariance is always measured between 2 dimensions. If we calculate the covariance between one dimension and itself, you get the variance. So, if we had a 3-dimensional data set $(x,y,z)$, measuring the covariance between and $x,y$ or $y,z$ and $z,x$ would find the variance of the $x,y,z$ dimensions respectively. The formula for covariance is:

$$cov(X) = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(X_i - \bar{X})^T}{(n - 1)}$$
23

Figure 4.4: Visualization of Eigen Embryos for 8 different BDGP Embryo images.

The covariance is always measured between 2 dimensions. If we have a data set with more than 2 dimensions, there is more than one covariance measurement that can be calculated. A useful way to get all the possible covariance values between all the different dimensions is to calculate them all and put them in a matrix.

4.1.2 SVD of Covariance

Let \( x_i, i = 1, \ldots, n \) be a set of training data points, where \( n \) be the number of data points. Matrix \( X = [x_1, x_2, \ldots, x_n] \) is the data matrix. Let \( \bar{x} \) be the centroid of the training data points. By subtracting each data point by the centroid (i.e., the translation), we get the zero-mean data matrix \( X \) given as:

\[
X = [x_1 - \bar{x}, x_2 - \bar{x}, \ldots, x_n - \bar{x}]
\]

Then we construct the covariance matrix

\[
C = XX^T
\]

where \( T \) denotes the transpose of a matrix. SVD is now applied to the covariance matrix \( C \) to obtain the eigen-decomposition of \( C \). A projection matrix \( P \) consists of a set of eigenvectors associated with largest eigenvalues. With the projection matrix \( P \), we can
obtain the subspace representation of a data point \( x \) in the original \( m \)-dimensional space by \( P_x \).

Figure 4.4 shows the eigen-embryos obtained from the SVD of the covariance matrix. A set of images are taken, and are converted to to a linear vector. The average of the considered vectors is computed and a different set of vectors are computed which are equal to the difference between the average vector and original image vectors. All the vectors are again grouped as a single matrix. The SVD is then applied on the matrix. By reshaping the resulting matrix rows which are vectors into the dimensions of the original image we get the eigen embryos.

The Singular value decomposition (SVD) is a powerful technique in many matrix computations and analyses. Using the SVD of a matrix in computations, rather than the original matrix, has the advantage of being more robust to numerical error. Additionally, the SVD exposes the geometric structure of a matrix, an important aspect of many matrix calculations. A matrix can be described as a transformation for one vector space to another. The components of the SVD quantify the resulting change between the underlying geometry of those vector spaces. The SVD is employed in a variety of applications, from least-squares problems to solving systems of linear equations. Each of these applications exploit key properties of the SVD, its relation to the rank of a matrix and its ability to approximate matrices of a given rank. Many fundamental aspects of linear algebra rely on determining the rank of a matrix, making the SVD an important and widely used technique. The ability of SVD to eliminate a large proportion of the data is a primary reason for its use.

The purpose of singular value decomposition is to reduce a dataset containing a large number of values to a dataset containing significantly fewer values, but which still contains a large fraction of the variability present in the original data. Often in the atmospheric and geophysical sciences, data will exhibit large spatial correlations. SVD analysis results in a more compact representation especially with multivariate datasets and
can provide insight into spatial and temporal variations exhibited in the fields of data being analyzed. There are a few caveats one should be aware of before computing the SVD of a set of data. First, the data must consist of anomalies. Secondly, the data should be de-trended. When trends in the data exist over time, the first structure often captures them. If the purpose of the analysis is to find spatial correlations independent of trends, the data should be de-trended before applying SVD analysis.

The first structure is the single pattern that represents the most variance in the data. The structures are the elements of the eigenvectors of the variance-covariance matrix of the data. In the data library, the eigenvectors are also known as EOF’s. The first eigenvector (EOF) points to the direction in which the data vectors jointly exhibit the most variability. Essentially, a new coordinate system is created, with each axis aligned along the direction of maximum joint variability.

The second structure is the pattern that describes the second largest amount of variance, calculated the same way as the first structure. A very important property of the second structure is that it is completely uncorrelated with the first structure, as well as all other following structures. The second eigenvector is perpendicular to the first eigenvector, which is perpendicular to the third eigenvector and so on. This property is what led Lorenz[49] to call the technique empirical orthogonal function analysis. All structures are mutually uncorrelated.

The variance of the nth principal component is the nth eigenvalue. Therefore, the total variation exhibited by the data is equal to the sum of all eigenvalues. In the data library, eigenvalues are normalized such that the sum of all eigenvalues equals 1. A normalized eigenvalue will indicate the percentage of total variance explained by its corresponding structure. Structures have also been normalized so that the root mean square equals 1. This way, the structures can be expressed in terms of standard deviation.

Singular values are equal to the square root of the eigenvalues. Since eigenvalues are
automatically normalized in the data library, they do not easily provide information into the total amount of variance they explain. However, you may calculate the total variance explained by each EOF by squaring the singular values.

In the data library there is a time series associated with each structure. These time series are the principal components. The first time series is calculated by projecting the data matrix onto the first eigenvector of the variance-covariance matrix of the data, the second time series by projecting onto the second eigenvector, and so on. The time series values indicate the amount of the given structure needed to complete the data field. It follows that the structure (dimensionless) multiplied by the time series value at a single point in time (units of the data), summed over all structures, yields the original data at that point in time.

Mathematically, there are as many eigenvectors as there are elements in the vector data set. The first few eigenvectors will point in directions where the data jointly exhibits large variation. The remaining eigenvectors will point to directions where the data jointly exhibits less variation. For this reason, it is often possible to capture most of the variation by considering only the first few eigenvectors. The remaining eigenvectors, along with their corresponding principal components, are truncated.

Using a superscript T to denote the transpose of a vector or matrix, we say two vectors $x, y$ are orthogonal if

$$x^T y = 0$$

In two or three dimensional space, this imply means that the vectors are perpendicular. Let $A$ be a square matrix such that its columns are mutually orthogonal vectors of length 1, i.e.

$$x^T x = 1$$

The $A$ is an orthogonal matrix and

$$A^T A = I$$
the identity matrix. For simpler notation, assuming that a matrix $A$ has at least as many rows as columns ($M \geq N$).

A singular value decomposition of an $M \times N$ matrix $A$ is any factorization of the form

$$A = UDV^T$$

where $U$ is an $M \times M$ orthogonal matrix, and $D$ is an $M \times N$ diagonal matrix with all the elements except in the diagonal of the matrix are zeros.

### 4.1.3 Relation with Eigen-Analysis

Eigenvalue decomposition is defined only for square matrices as only square symmetric matrices have real-valued eigenvalues whereas SVD is defined for all matrices. Given a matrix $M$, we consider the eigen decomposition of the correlation matrices $MM^T$ and $M^TM$. SVD is the eigenvectors of

$$MM^T \times P \times E$$

where, $P$ is the Positive square roots of eigenvalues of $MM^T$ and $E$ is the eigenvectors of $M^TM$.

Both $MM^T$ and $M^TM$ are symmetric (they are correlation matrices). They both will have the same eigenvalues. Unless $M$ is symmetric, $MM^T$ and $M^TM$ are different so, in general their eigenvectors will be different (although their eigenvalues are same). Since SVD is defined in terms of the eigenvalues and vectors of the Correlation matrices of a matrix, the eigenvalues will always be real valued (even if the matrix $M$ is not symmetric). In general, the SVD decomposition of a matrix $M$ equals its eigen decomposition only if $M$ is both square and symmetric.

Any symmetric matrix $R$ can be decomposed in the following way through a
diagonalization, or eigenanalysis:

\[ Re_i = \lambda_i e_i \]

\[ RE = LE \]

Where \( E \) is the matrix with the eigenvectors \( e_i \) as its columns, and \( L \) is the matrix with the eigenvalues \( \lambda_i \), along its diagonal and zeros elsewhere. The set of eigenvectors, \( e_i \), and associated eigenvalues, \( \lambda_i \), represent a coordinate transformation into a coordinate space where the matrix \( R \) becomes diagonal. The square of the eigenvalue from the SVD is equal to the eigenvalue from the eigen analysis of the covariance matrix.

### 4.2 Liner Discriminant Analysis

LDA is another very popular dimensionality reduction concept. Suppose there are \( C \) classes in the training data. LDA performs dimensionality reduction while preserving as much of the class discriminatory information as possible. It seeks to find directions along which the classes are best separated. It does so by taking into consideration the scatter within classes but also the scatter between classes. It is also more capable of distinguishing image variation due to identity from variation due to other sources such as illumination and expression. Let there are \( m \) number of classes \( C_1, C_2, C_3 \ldots C_m \) are the local centroids of Class-1 Class-2 Class-3 \ldots Class-m. Let \( C \) be the global centroid. LDA computes a transformation that maximizes the between-class scatter while minimizing the within-class scatter.

\[
M_b = (C_1 - C, C_2 - C \ldots C_m - C)_{d \times m}
\]

\[
M_w = (P_{11} - C_1, P_{12} - C_2 \ldots P_{1m} - C_m)_{d \times n}
\]

Between-class Scatter is given as:

\[ M_b M_b^T \]
and the Within-class Scatter is given as:

\[ M_w M_{wd*n}^T \]

The SVD is then applied on:

\[ (M_w M_w^T)^{-1} M_b M_b^T \]

LDA is a supervised technique, the class information is provided along with the training data. PCA is an unsupervised technique. The next chapter shows few experimental results we performed on the data reduced using the LDA and PCA techniques.
CHAPTER 5: EXPERIMENTS

5.1 Classifier

K Nearest Neighbor (KNN from now on) is one of those algorithms that are very simple to understand but works incredibly well in practice. Also it is surprisingly versatile and its applications range from vision to proteins to computational geometry to graphs.

KNN is an non parametric lazy learning algorithm, it means that it does not make any assumptions on the underlying data distribution. This is pretty useful, as in the real world, most of the practical data does not obey the typical theoretical assumptions made (e.g. Gaussian mixtures, linearly separable etc). Non parametric algorithms like KNN are of a great use here.

It is also a lazy algorithm, it does not use the training data points to do any generalization. In other words, there is no explicit training phase or it is very minimal. This means the training phase is pretty fast. Lack of generalization means that KNN keeps all the training data. More exactly, all the training data is needed during the testing phase. This is in contrast to other popular techniques like Support Vector Machines where you can discard all non support vectors without any problem. Most of the lazy algorithms, especially KNN makes decision based on the entire training data set (in the best case a subset of them). The dichotomy is very obvious here. There is a non existent or minimal training phase but a costly testing phase. The cost is in terms of both time and memory. More time might be needed as in the worst case, all data points might take point in decision. More memory is needed as we need to store all training data.

KNN assumes that the data is in a feature space. More exactly, the data points are in a metric space. The data can be scalars or possibly even multidimensional vectors. Since the points are in feature space, they have a notion of distance. This need not necessarily be Euclidean distance although it is the one commonly used. In our experiments we used the
Euclidean distance. Each of the training data consists of a set of vectors and class label associated with each vector. In the simplest case, it will be either + or - (for positive or negative classes). But KNN, can work equally well with arbitrary number of classes. We can also give a single number "k". This number decides how many neighbors (where neighbors is defined based on the distance metric) influence the classification.

In this chapter, we test the performance of the proposed features for stage recognition. Our dataset contains 900 BDGP images in 6 developmental stages. Figure 5.1 shows an example of embryonic images. These images have been standardized with respect to orientation and scale, and the dimension of the images are $320 \times 240$. The classifier we used is K nearest neighbor (K-NN). We use 2-Fold cross validation to estimate the recognition accuracy. We will compare the recognition accuracy of RGB features, orientation histogram, the subspace representation of orientation histogram and
the Gabor features. We will also study their recognition efficiency.

The experiments are performed on the MATLAB platform using the Image Processing toolkit. To attain higher performance the classification algorithms were written in C language and compiled under MEX compiler, which enables using the MATLAB tools inside the C programs.

### 5.2 RGB Intensity

RGB intensity based algorithm uses a straightforward pixel to pixel type matching. This technique uses the image block standardization and tries to find the most similar image from the opposite set. Figure 5.3 shows the trend of the classification accuracies for different number of Nearest Neighbors. The optimal accuracy occurs at K=5. In order to reduce the time cost and possible gain the memory space and accuracy, we applied the Principal Component Analysis dimensionality reduction algorithm on the RGB Intensity features. The optimal accuracy for PCA was at N=2 and KNN for 5 nearest neighbors. When considering a single nearest neighbor the optimal accuracy for PCA was at N=2 as well. N denotes the number of Principal Components. Figure 5.2 shows the trend of the optimal PCA values for different N. Figure 5.4 shows the trend of the PCA at optimal PCs at 5 nearest neighbors.

The RGB intensity values are obtained by combining the R, G, B components of a pixel. The results will be better if the images are under a steady illumination. Higher illumination may cover up critical areas of the embryo which might affect the accuracies, so it is very important to find the correct level of illumination under which the images will be photographed. The images are standardized and registered manually for our experiments. The registration involves a procedure where the embryo is first highlighted and then extracted from the original image and then placed parallel to the horizontal axis on a white background, resulting in an image of required dimensions. SVD can be used to
determine the major axis and then align the embryo with respect to the horizontal axis. The images are not aligned with respect to the head and tail in our experiments.

Table 5.1 depicts the accuracy of the RGB feature based classification. Table 5.2
Table 5.1: Accuracy comparison of RGB, Reduced RGB with PCA at the optimal number of Principal Components for number of neighbors from 1 to 5

<table>
<thead>
<tr>
<th>K</th>
<th>RGB</th>
<th>PCA on RGB features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>61</td>
<td>24,(2 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>43</td>
<td>25,(7 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>39</td>
<td>23,(2 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td>24,(6 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>37</td>
<td>25,(2 PCs)</td>
</tr>
</tbody>
</table>

Table 5.2: Time cost comparison of RGB features, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features

<table>
<thead>
<tr>
<th>K</th>
<th>RGB</th>
<th>PCA of RGB features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12389</td>
<td>10245,(2 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>12324</td>
<td>11324,(7 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>12198</td>
<td>12056,(2 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>12234</td>
<td>10908,(6 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>12985</td>
<td>11200,(2 PCs)</td>
</tr>
</tbody>
</table>

depicts the trends of the time taken to perform the classification on the complete 900 BDGP images.

The RGB image features are extracted in a straightforward manner, no extra processing of the data is done other than the standardization of the image sub block to the window matrix. The features length being $225 \times (16 \times 16)$ the feature length is generally longer than the Orientation histogram. The features are then reduced to a length of 2 to 20 using the PCA which should generally take the same time when the orientation histogram and Gabor features are reduced to the same length. The classification occur at reduced amount of the image details, resulting in less number of accurate matches. The small number of training data which is small in the test also effects the classification accuracy.

Figure 5.2 shows the visualization of the RGB features under different nearest neighbors and the reduced feature accuracies. The images are reduced from a length of
16 × 16 × 225 to a length of principle components we choose which varies from 2 to 20. The accuracies after the reduction are lower than the original unreduced feature accuracies as the image detail is reduced, the dimension of the reduced features are drastically lower than the original features. Figure 5.3 shows the optimal reduced feature accuracies under different number of nearest neighbors. Figure 5.4 shows the visualization of the accuracies of the optimal reduced features under different number of nearest neighbors.

Table 5.1 shows the readings of the accuracies of the original features against the reduced features under different k values of the KNN algorithm. Table 5.2 shows the time cost readings to finish the classification using the original features against the reduced features under different k values of the KNN algorithm.

5.3 Orientation Histogram

The orientation histogram builds the features in a very close fashion to the SIFT features. Figure 5.6 shows that the orientation histogram classification accuracy was very close for all the Nearest Neighbor values, the optimal accuracy was 70 under K-NN with K=1. The orientation histograms are reduced using the PCA with PCs varying from 2 to 20. Figure 5.6 shows the optimal classification accuracy for different N. The PCA run time is considerably lower than the orientation histogram as the length of the features are reduced. Though PCA is reliable and a robust technique the dimension reduction of feature from 4 × 4 × 128 to less than 20 has resulted in a fewer number of accurate matches when compared to that of the original orientation histogram. The classification occur at reduced amount of the image details, resulting in fewer number of accurate matches. The number of training data, which is small, in the test also effects the classification accuracy.

Figure 5.5 shows the visualization of the orientation histogram under different nearest neighbors and the reduced feature accuracies. The images are reduced from a
length of $4 \times 4 \times 128$ to a length of principle components we choose which varies from 2 to 20. The accuracies after the reduction are lower than the original unreduced feature accuracies as the image detail is reduced, the dimension of the reduced features are significantly lower than the original features. Figure 5.6 shows the optimal reduced feature accuracies under $K=1$ nearest neighbors. Figure 5.7 shows the visualization of the accuracies of the optimal reduced features under different numbers of nearest neighbors.

Table 5.3 shows the readings of the accuracies of the original features against the reduced features under different $k$ values of the KNN algorithm. Table 5.4 shows the time cost readings to finish the classification using the original features against the reduced features under different $k$ values of the KNN algorithm.
Figure 5.7: Accuracy comparison of Orientation histograms, PCA optimal values when classification is performed under 1 Nearest Neighbors. Accuracy for PCs = 2 to 20.

Table 5.3: Accuracy comparison of orientation histogram, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5

<table>
<thead>
<tr>
<th>K</th>
<th>Orientation histogram</th>
<th>PCA on Orientation histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>70</td>
<td>59, (12 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>51</td>
<td>58, (12 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>52</td>
<td>57, (12 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
<td>58, (12 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>52</td>
<td>56, (12 PCs)</td>
</tr>
</tbody>
</table>

Table 5.4: Time cost comparison of orientation histogram, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features

<table>
<thead>
<tr>
<th>K</th>
<th>Orientation histogram</th>
<th>PCA of Orientation histogram</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1002</td>
<td>11245, (12 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>1134</td>
<td>10264, (12 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>1156</td>
<td>10426, (12 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>1242</td>
<td>10918, (12 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>1211</td>
<td>10290, (12 PCs)</td>
</tr>
</tbody>
</table>

shows the runtime costs for the orientation histogram and the reduced orientation histogram.

The Orientation histogram generally takes slightly longer time to classify the images. Orientation histogram involves calculation of the Gradient magnitude and
obtaining the orientation for each pixel and then populating the histogram in the correct slot. Each image sub block will populate its 8 bins, which are part of the bigger complete image histogram. When the orientation histogram is reduced from 128 to a length varying between 2 and 20 using the PCA algorithm, the time cost remains neutral as the feature lengths are the same. The orientation histogram outperforms the RGB features as the orientation histogram obtains the features which distinguishes the image not only with the appearance but also the orientation of the pixels. Since there is more distinct information fed into the histogram, the orientation histogram outperforms the RGB features which depend solely on the appearance.

5.4 Gabor Filter features

The Gabor features builds the features in a very different fashion to the RGB and the orientation histogram, but in a very effective manner. Figure 5.9 shows that the Gabor feature classification accuracy was very close for all the Nearest Neighbor values, the optimal accuracy was 76 under K-NN with K=1. The Gabor features are reduced using the PCA with PCs varying from 2 to 20. Figure 5.8 shows the classification accuracy of the reduced features for different N for K values from 1 to 5. The PCA run time is considerably lower than the Gabor features as the length of the features are reduced. Dimension reduction is applied on the feature and they are reduced to a length varying between 2 and 20. This has resulted in a smaller number of accurate matches when compared to that of the original orientation histogram. The classification occur at reduced amount of the image details, resulting in fewer accurate matches. The number of training data, which is small, in the test also effects the classification accuracy.

Table 5.5 shows the details about the accuracy results for different N values of the KNN. Table 5.6 shows the runtime costs for the Gabor features and the reduced Gabor features.

Figure 5.10 shows the visualization of the Gabor features under different nearest
Figure 5.8: The Gabor Features reduced by PCA. Accuracy for the number of Principle Components from 2 to 20 and Nearest Neighbors from 1 to 5.

Figure 5.9: The Gabor Features for Nearest Neighbors from 1 to 5.

Figure 5.10: Accuracy comparison of Gabor Image features, PCA optimal values when classification is performed under 5 Nearest Neighbors. Accuracy for PCs = 2 to 20.

neighbors and the reduced feature accuracies. The images are reduced from a length of $20 \times 15 \times scales \times orientations \times blocks$ to a length of principle components we choose
Table 5.5: Accuracy comparison of Gabor, PCA of Gabor features at the optimal number of Principal Components for number of neighbors from 1 to 5

<table>
<thead>
<tr>
<th>K</th>
<th>Gabor</th>
<th>PCA of Gabor features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>76</td>
<td>39,(5 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>49</td>
<td>40,(4 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>47</td>
<td>39,(5 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>39,(5 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>48</td>
<td>37,(5 PCs)</td>
</tr>
</tbody>
</table>

Table 5.6: Time cost comparison of Gabor features, PCA at the optimal number of Principal Components for number of neighbors from 1 to 5. Time costs measured in milliseconds, including the time taken to generate image features

<table>
<thead>
<tr>
<th>K</th>
<th>Gabor</th>
<th>PCA of Gabor features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>60234</td>
<td>13245,(17 PCs)</td>
</tr>
<tr>
<td>2</td>
<td>60244</td>
<td>15324,(11 PCs)</td>
</tr>
<tr>
<td>3</td>
<td>61039</td>
<td>12456,(12 PCs)</td>
</tr>
<tr>
<td>4</td>
<td>62372</td>
<td>13958,(04 PCs)</td>
</tr>
<tr>
<td>5</td>
<td>61231</td>
<td>13290,(13 PCs)</td>
</tr>
</tbody>
</table>

which varies from 2 to 20. The accuracies after the reduction are lower than the original unreduced feature accuracies as the image detail is reduced, the dimension of the reduced features are drastically lower than the original features. Figure 5.9 shows the optimal feature accuracies under different number of nearest neighbors. Figure 5.10 shows the visualization of the accuracies of the reduced features under different number of nearest neighbors.

Table 5.5 shows the readings of the accuracies of the original features against the reduced features under different k values of the KNN algorithm. Table 5.6 shows the time cost readings to finish the classification using the original features against the reduced features under different k values of the KNN algorithm.

The Gabor features involves obtaining the features based on the convolution of the image blocks with the Gabor mask. The mask is created using the Gaussian and the COS
functions. The features are more distinct than the orientation histogram and the RGB features. The feature length is considerably larger than both the orientation histogram and the RGB features. The Gabor feature based classification outperforms the Orientation histogram. Gabor filtering is a proven method closely associated with the human vision mechanism. The computation cost is very high when compared to the Orientation histogram. Not only the feature length is very large but the computation of the Gabor mask for a given set of Scale and Orientations involves high computation costs. When the Dimension reduction is applied on the Gabor features the computation costs come down but still maintain an edge over the reduced orientation histogram.

5.5 RGB vs Orientation Histogram vs Gabor

The time cost analysis shows that there is a considerable amount of time saved for an image set of 900 individual images. The PCA will result in much better results as the size of the image set increases. Reducing the dimensions of the features to less than 20 makes the computation cost come down. Comparing the RGB Intensity algorithm with the orientation histogram and Gabor features shows that the Gabor features takes longer time to run but yields much better accuracies. The computation costs for the RGB Intensity algorithm is low as the process of populating the feature is straight forward. The RGB intensity values are concatenated and used to populate the standardizing window which is converted into the image feature. The orientation histogram involves obtaining the Gaussian magnitude and the orientation of a given pixel which is an extra computation needed to be performed with respect to the RGB. The Gabor features take the highest computation time as it involves sampling the image with a set of scales and orientations, each iteration involves generating a Gabor mask and performing the convolution of the Gabor mask with the image sub block which gives a coefficient and this coefficient is used to populate the Gabor feature.
Table 5.7: Accuracy comparison of RGB, orientation histogram, Gabor

<table>
<thead>
<tr>
<th>K</th>
<th>RGB</th>
<th>Orientation histogram</th>
<th>Gabor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>61</td>
<td>70</td>
<td>76</td>
</tr>
<tr>
<td>2</td>
<td>43</td>
<td>51</td>
<td>49</td>
</tr>
<tr>
<td>3</td>
<td>39</td>
<td>52</td>
<td>47</td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td>52</td>
<td>48</td>
</tr>
<tr>
<td>5</td>
<td>37</td>
<td>52</td>
<td>48</td>
</tr>
</tbody>
</table>

Table 5.8: Time cost comparison of RGB, Orientation histogram, Gabor

<table>
<thead>
<tr>
<th>K</th>
<th>RGB</th>
<th>Orientation histogram</th>
<th>Gabor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12389</td>
<td>1002</td>
<td>60234</td>
</tr>
<tr>
<td>2</td>
<td>12324</td>
<td>1134</td>
<td>60244</td>
</tr>
<tr>
<td>3</td>
<td>12198</td>
<td>1156</td>
<td>61039</td>
</tr>
<tr>
<td>4</td>
<td>12234</td>
<td>1242</td>
<td>62372</td>
</tr>
<tr>
<td>5</td>
<td>12985</td>
<td>1211</td>
<td>61231</td>
</tr>
</tbody>
</table>

Figure 5.11: Accuracy comparison of RGB, Orientation histogram, Gabor

Table 5.7 shows the comparison of all the different features. Table 5.8 shows the computation costs of all the different features.

5.5.1 LDA on Orientation histogram, RGB and Gabor features

Linear discriminant analysis is one of the very popular dimension reduction techniques we used for testing the classification. The results are much lower when
Table 5.9: Accuracy comparison of RGB, orientation histogram, Gabor after applying LDA

<table>
<thead>
<tr>
<th>K</th>
<th>RGB</th>
<th>Orientation histogram</th>
<th>Gabor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td>32</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>33</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>32</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>34</td>
<td>15</td>
</tr>
</tbody>
</table>

compared to the accuracies of the dimensionally reduced features using PCA. Table 5.9 shows the accuracy comparison when the image features are reduced using the LDA.

The effects of embryo images not being aligned correctly and the normalization of the embryo respect to the head and tail clearly plays a very major role in the classification of the embryos. Considering the alignment, an image not properly aligned produces an image feature varying in values to that of an image properly aligned. As the features are populated with respect to the image blocks, any varying effects the location of the vector in the space thus giving an inappropriate result. An image feature incorrectly normalized with respect to head and tail will produce a feature which is a reverse of what the actual correctly normalized image would have produced.
CHAPTER 6: CONCLUSION AND FUTURE WORK

Using the orientation histogram features alone provides a decent rate of accuracy. The Gabor features outperform the orientation histogram. The performance can further be improved greatly by implementing the PCA dimensionality reduction technique on the orientation histogram features and the Gabor features. Training the algorithm with enough number of image samples may improve the accuracy of classification using the reduced orientation histogram features. The RGB Intensity algorithm computation cost is lower than the orientation histogram and Gabor feature algorithms. The subspace representation of orientation histogram features considerably reduces the computation cost.

In the future, we plan to study stage classification under unconstrained localization (i.e., without the assumption of orientation and scale normalized embryonic images).
BIBLIOGRAPHY


[29] Shen, L.L. and Bai, L. and Fairhurst, M. *Gabor wavelets and general discriminant analysis for face identification and verification* Image and Vision Computing


[38] Jolliffe, I. *Principal component analysis* Wiley Online Library


[44] Danaher, DJ and others *Application of singular value decomposition (SVD) in high energy [gamma]-ray astronomy (E0 0.1 TeV)* Astroparticle Physics

[45] Gerbrands, J.J. *On the relationships between SVD, KLT and PCA* Pattern Recognition


[49] Lorenz, E.N. and Air Force Cambridge Research Center (US) Geophysics Research Directorate *Empirical orthogonal functions and statistical weather prediction* Massachusetts Institute of Technology, Dept. of Meteorology

[50] North, G.R. *Empirical orthogonal functions and normal modes* Journal of the Atmospheric Sciences
