Interactive Visual Analysis of Images

BY

Tuan Nhon Dang
B.S. (with honor), Ho Chi Minh City University of Technology, 2006
M.S., Politecnico di Milano, 2008
M.S., University of Illinois at Chicago, 2009

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2014

Chicago, Illinois

Defense Committee:

Leland Wilkinson, Chair and Advisor
Andrew Johnson
Robert Kenyon
Gyorgy Turan, Mathematics, Statistics, and Computer Science
Marco Santambrogio, Politecnico di Milano
Copyright by

Tuan Nhon Dang

2014
This dissertation is dedicated to my mother, Bong Thi Phan. Her encouragement, devotion, and sacrifice have sustained me throughout my life.
ACKNOWLEDGMENTS

I would like to express my special appreciation and thanks to my PhD advisor Professor Leland Wilkinson. Without him, this dissertation would not have been possible. In fact, he is the first person who introduced me to Visual analytics and I immediately realized that is my research interest which I had been looking for since admitted to computer science. I love our working environment which encourages creativities. I am also very appreciative of his patience in teaching and guiding me to do research.

Also, I would like to thank my committee members: Andrew Johnson, Robert Kenyon, Gyorgy Turan, and Marco Santambrogio. I appreciate the value of their time and their feedback on my work.

I feel very lucky to meet and work with many people in graduate school who were always willing to share ideas and give constructive feedback on my solutions. Those I do thank in particular is my former colleague Dr. Anushka Anand who has been joy to collaborate with and also be a good friend in life. She was the first person whom I discussed with on new ideas. Her objective and constructive feedbacks are very important for my papers to be accepted for publication.

TND
CONTRIBUTION OF AUTHORS

Chapter 1 introduces image retrieval and summarizes the contributions of my thesis into this field. Chapter 2 describes some related work in feature extraction and image retrieval. Chapter 3 briefly reviews Scagnostics [1], which laid the foundations for our work. Chapter 4 represents a published manuscript © 2013 IEEE. Reprinted, with permission, from Tuan Nhon Dang, Anushka Anand, and Leland Wilkinson, TimeSeer: Scagnostics for High-Dimensional Time Series, Transactions on Visualization and Computer Graphics, 03/2013. I was the primary author and major driver of the research. My research advisor, Dr. Leland Wilkinson assisted me in writing the manuscript. Dr. Anushka Anand assisted me in citing related work. Chapter 5 represents another published manuscript © 2014 IEEE. Reprinted, with permission, from Tuan Nhon Dang and Leland Wilkinson, ScagExplorer: Exploring Scatterplots by Their Scagnostics, IEEE Pacific Visualization Symposium, 03/2014. For this paper, I was the primary author who conducted the research and wrote the manuscript. Dr. Leland Wilkinson assisted me in writing the manuscript and citing related work. Chapter 6 represents my own unpublished work in image retrieval research. This work extended Scagnostics to handle pixels in images. I anticipate that this line of research will be continued and will ultimately be published as part of a co-authored manuscript. Chapter 7 provides concluding discussions and future work.

The IEEE policies on reusing the above published manuscripts in this thesis are describes in Appendix.
# Table of Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Summary of Contributions</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Thesis Organization</td>
<td>2</td>
</tr>
<tr>
<td><strong>2. RELATED WORK</strong></td>
<td>4</td>
</tr>
<tr>
<td>2.1 Color Histogram</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Region-Based techniques</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Feature-based techniques</td>
<td>6</td>
</tr>
<tr>
<td><strong>3. SCAGNOSTICS</strong></td>
<td>8</td>
</tr>
<tr>
<td>3.1 Computing Scagnostic Measures</td>
<td>8</td>
</tr>
<tr>
<td>3.2 Summary</td>
<td>11</td>
</tr>
<tr>
<td><strong>4. VISUAL PATTERN DETECTION FOR LARGE ORDERED POINT SETS</strong></td>
<td>13</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>13</td>
</tr>
<tr>
<td>4.2 TimeSeer</td>
<td>14</td>
</tr>
<tr>
<td>4.2.1 The TimeSeer GUI</td>
<td>14</td>
</tr>
<tr>
<td>4.3 Examples</td>
<td>14</td>
</tr>
<tr>
<td>4.3.1 Variable Selection SPLOM</td>
<td>15</td>
</tr>
<tr>
<td>4.3.2 Lensing</td>
<td>16</td>
</tr>
<tr>
<td>4.3.3 Time Series Viewer</td>
<td>19</td>
</tr>
<tr>
<td>4.3.4 Filtering, Brushing, and Drill-down</td>
<td>21</td>
</tr>
<tr>
<td>4.3.4.1 Filtering</td>
<td>21</td>
</tr>
<tr>
<td>4.3.4.2 Brushing</td>
<td>23</td>
</tr>
<tr>
<td>4.3.4.3 Drill-down</td>
<td>23</td>
</tr>
<tr>
<td>4.3.5 Searching for Similar Patterns or Interesting Distributions</td>
<td>25</td>
</tr>
<tr>
<td>4.3.5.1 Automatic Search for Similar Distributions</td>
<td>25</td>
</tr>
<tr>
<td>4.3.5.2 Manual Search for Similar Distributions</td>
<td>28</td>
</tr>
<tr>
<td>4.4 Conclusions</td>
<td>31</td>
</tr>
<tr>
<td><strong>5. VISUAL PATTERN DETECTION FOR LARGE UNORDERED POINT SETS</strong></td>
<td>32</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>32</td>
</tr>
<tr>
<td>5.2 Related Work</td>
<td>32</td>
</tr>
<tr>
<td>5.3 ScagExplorer Overview</td>
<td>33</td>
</tr>
<tr>
<td>5.3.1 Quality Metrics</td>
<td>33</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3.2 Datasets</td>
<td>35</td>
</tr>
<tr>
<td>5.4 ScagExplorer Components</td>
<td>35</td>
</tr>
<tr>
<td>5.4.1 Clustering Algorithm</td>
<td>36</td>
</tr>
<tr>
<td>5.4.2 Displaying the leader scatterplots</td>
<td>38</td>
</tr>
<tr>
<td>5.4.3 Exploring similar scatterplots</td>
<td>42</td>
</tr>
<tr>
<td>5.4.4 Filtering Scagnostics</td>
<td>42</td>
</tr>
<tr>
<td>5.4.4.1 Filtering Scagnostics with Parallel Coordinates</td>
<td>46</td>
</tr>
<tr>
<td>5.4.4.2 Filtering Scagnostics with a Rose Controller</td>
<td>48</td>
</tr>
<tr>
<td>5.5 Conclusions</td>
<td>50</td>
</tr>
<tr>
<td>6 VISUAL PATTERN DETECTION FOR IMAGES</td>
<td>52</td>
</tr>
<tr>
<td>6.1 Gestalt Theory of Visual Perception</td>
<td>52</td>
</tr>
<tr>
<td>6.1.1 Principle 1 - Proximity</td>
<td>53</td>
</tr>
<tr>
<td>6.1.2 Principle 2 - Figure and ground</td>
<td>53</td>
</tr>
<tr>
<td>6.1.3 Principle 3 - Similarity</td>
<td>55</td>
</tr>
<tr>
<td>6.2 PixSearcher</td>
<td>56</td>
</tr>
<tr>
<td>6.2.1 PixSearcher Design</td>
<td>56</td>
</tr>
<tr>
<td>6.2.2 Quantizing Images</td>
<td>58</td>
</tr>
<tr>
<td>6.2.3 Computing Descriptors</td>
<td>63</td>
</tr>
<tr>
<td>6.2.4 Dissimilarity of Two Images</td>
<td>71</td>
</tr>
<tr>
<td>6.3 Testing</td>
<td>71</td>
</tr>
<tr>
<td>6.3.1 Benchmark databases for CBIR</td>
<td>71</td>
</tr>
<tr>
<td>6.3.2 Selection of number of quantizations</td>
<td>75</td>
</tr>
<tr>
<td>6.3.3 Selection of rescaled resolution</td>
<td>78</td>
</tr>
<tr>
<td>6.3.4 Comparisons of different CBIR algorithms</td>
<td>78</td>
</tr>
<tr>
<td>6.3.5 Image Retrieval</td>
<td>82</td>
</tr>
<tr>
<td>6.4 Image Cluster</td>
<td>83</td>
</tr>
<tr>
<td>6.5 Conclusions</td>
<td>88</td>
</tr>
<tr>
<td>7 CONCLUSIONS AND FUTURE WORK</td>
<td>91</td>
</tr>
<tr>
<td>CITED LITERATURE</td>
<td>93</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>110</td>
</tr>
<tr>
<td>VITA</td>
<td>112</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Characteristics of datasets for ScagExplorer Demonstrations</td>
</tr>
<tr>
<td>II</td>
<td>Characteristics of datasets used for evaluation PixSearcher and other CBIR algorithms</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
</tr>
<tr>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>22</td>
</tr>
<tr>
<td>9</td>
<td>24</td>
</tr>
<tr>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>11</td>
<td>27</td>
</tr>
<tr>
<td>12</td>
<td>29</td>
</tr>
<tr>
<td>13</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>36</td>
</tr>
<tr>
<td>15</td>
<td>41</td>
</tr>
<tr>
<td>16</td>
<td>43</td>
</tr>
<tr>
<td>17</td>
<td>44</td>
</tr>
<tr>
<td>18</td>
<td>45</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>19</td>
<td>University ranking data, filtering Scagnostics by parallel coordinates. 47</td>
</tr>
<tr>
<td>20</td>
<td>NRC university ranking data, filtering high Monotonic plots. 49</td>
</tr>
<tr>
<td>21</td>
<td>Filtering Clumpy on a Pie of Rose Controller. 50</td>
</tr>
<tr>
<td>22</td>
<td>Filtering scagnostics by Rose Controller: a) Monotonic ≥ 0.5, Outlying ≥ 0.5 b) Monotonic ≥ 0.7, Outlying ≥ 0.7 c) Scagnostic View. 51</td>
</tr>
<tr>
<td>23</td>
<td>Proximity principle: a) The thirteen squares are placed without proximity b) The thirteen squares are placed with close proximity to form two groups. The figure was drawn by the author. 53</td>
</tr>
<tr>
<td>24</td>
<td>Figure and ground principle: a) On the white background, the two black faces gain the better impression as the element in focus b) On the inverted image, the central region resembling a vase appears to be the figure. 54</td>
</tr>
<tr>
<td>25</td>
<td>Configural properties: (a) Convexity (b) Small Area (c) Wide base (d) Symmetry (e) Surrounded region (f) Lower region. 55</td>
</tr>
<tr>
<td>26</td>
<td>Examples of Gestalt principle of similarity. 56</td>
</tr>
<tr>
<td>27</td>
<td>Schematic overview of PixSearcher. 57</td>
</tr>
<tr>
<td>28</td>
<td>An example image in the Wang dataset described in Section 6.3.1. 59</td>
</tr>
<tr>
<td>29</td>
<td>Comparison of quantization results on RGB and HSB color space of a sample image (RGB color cube). The input image has been retrieved from (65). 61</td>
</tr>
<tr>
<td>30</td>
<td>Comparison of quantization results on RGB and HSB color space of a sample image (HSB cylindrical color wheel). The input image has been retrieved from (66). 62</td>
</tr>
<tr>
<td>31</td>
<td>Comparison of quantization results on RGB and HSB color space of image 426 in the Wang dataset described in Section 6.3.1. 64</td>
</tr>
<tr>
<td>32</td>
<td>Two low resolution versions with different orientations of the same image (image 886 in the Wang dataset described in Section 6.3.1; this image has been altered). The left image can be interpreted as beach and sea. The right image can be interpreted as mountain and blue sky. 66</td>
</tr>
<tr>
<td>33</td>
<td>Top image shows high Close and sparse distribution. Bottom image shows low Close and dense distribution. 69</td>
</tr>
<tr>
<td>34</td>
<td>Top image shows high Convex and low Skinny distribution. Bottom image shows low Convex and high Skinny distribution. 70</td>
</tr>
<tr>
<td>35</td>
<td>Top image shows high Clumpy distribution. Bottom image shows low Clumpy distribution. 70</td>
</tr>
<tr>
<td>36</td>
<td>One example image from each of the 10 classes of the Wang database together with their class labels under each image. 73</td>
</tr>
<tr>
<td>37</td>
<td>Six example images from the UW database with annotations under each image. 74</td>
</tr>
<tr>
<td>38</td>
<td>Eight example images from the UCID dataset. 75</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>39</td>
<td>ZuBuD image dataset: The left panel displays a query image (top left) from testing dataset and 5 similar images (of the same building) from training dataset.</td>
</tr>
<tr>
<td>40</td>
<td>Error rate (upper panel) and computation time (lower panel) of PixSearcher with different quantizations on the 4 image datasets.</td>
</tr>
<tr>
<td>41</td>
<td>Error rate (upper panel) and computation time (lower panel) of PixSearcher with different resolutions on the 4 image datasets.</td>
</tr>
<tr>
<td>42</td>
<td>Dotplots display the error rates for 20 algorithms for 4 image datasets (Wang, UW, UCID, and ZuBud). The last row is the average error rates of 4 image datasets.</td>
</tr>
<tr>
<td>43</td>
<td>Feature computation times for 6 algorithms for 4 image datasets (Wang, UW, UCID, and ZuBud).</td>
</tr>
<tr>
<td>44</td>
<td>Feature computation times for PixSearcher and color histogram for images of various resolutions. PixSearcher (in red) requires much less time than color histogram (in blue) to process high quality images.</td>
</tr>
<tr>
<td>45</td>
<td>Retrieving similar images to a textured image on two CBIR systems: SIMPLIcity system on the top, PixSearcher system at the bottom.</td>
</tr>
<tr>
<td>46</td>
<td>Retrieving similar images to a non-textured image on two CBIR systems: SIMPLIcity system on the top, PixSearcher system at the bottom.</td>
</tr>
<tr>
<td>47</td>
<td>Corel database: Clusters of leader images.</td>
</tr>
<tr>
<td>48</td>
<td>Corel database: (a) Leader images are aligned on X-axis based on their Hue (b) Leader images are aligned on both X-axis and Y-axis based on their Hue and Brightness.</td>
</tr>
</tbody>
</table>
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UIC</td>
<td>University of Illinois at Chicago</td>
</tr>
<tr>
<td>Scagnostics</td>
<td>Scatterplot Diagnostics</td>
</tr>
<tr>
<td>SPLOM</td>
<td>Scatterplot Matrix</td>
</tr>
<tr>
<td>MST</td>
<td>Minimum Spanning Tree</td>
</tr>
<tr>
<td>CBIR</td>
<td>Content-Based Image Retrieval</td>
</tr>
<tr>
<td>DTW</td>
<td>Dynamic Time Warping</td>
</tr>
<tr>
<td>DWT</td>
<td>Discrete Wavelet Transform</td>
</tr>
<tr>
<td>PAA</td>
<td>Piecewise Aggregate Approximation</td>
</tr>
<tr>
<td>SAX</td>
<td>Symbolic Aggregation Approximation</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>DOI</td>
<td>Degree Of Interest</td>
</tr>
<tr>
<td>VaR</td>
<td>Value and Relation</td>
</tr>
<tr>
<td>MSD</td>
<td>Multidimensional Scaling</td>
</tr>
<tr>
<td>NRC</td>
<td>National Research Council</td>
</tr>
<tr>
<td>V1</td>
<td>Primary Visual Cortex</td>
</tr>
<tr>
<td>RGB</td>
<td>Red, Green, and Blue</td>
</tr>
<tr>
<td>HSB</td>
<td>Hue, Saturation, and Brightness</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>CRT</td>
<td>Composite Region Template</td>
</tr>
<tr>
<td>MPEG</td>
<td>Moving Picture Experts Group</td>
</tr>
<tr>
<td>CCH</td>
<td>Contrast Context Histogram</td>
</tr>
<tr>
<td>KNN</td>
<td>K Nearest Neighbor</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
</tbody>
</table>
SUMMARY

Mining and visualizing huge image databases has become a daunting task for many application areas such as astronomy, medicine, geology, oceanography, and crime prevention. In this thesis, we introduce a new technique for feature extraction that obtains image signatures from pixels obtained by quantized images at different levels. We also present an application (PixSearcher) that retrieves and organizes huge images databases. In contrast to raster techniques that use the entire pixel raster for distance computations, our application uses a small set of descriptors to handle large image collections.

Our research program began in 2010, when we were involved with characterizing the 2D distributions of orthogonal pairwise projections on a set of points in multidimensional Euclidean space. These characterizations included measures such as density, skewness, shape, outliers, and texture. Working directly with these measures, we located anomalous or interesting distributions in large collections of pairwise projections. For example, we used these measures to locate unusual correlations between pairs of time series and we used them to reorganize scatterplot matrices to avoid clutter.

Although the original motivation was to characterize the 2D distributions of scatterplots, we soon realized the idea had more general implications. We extended this work to handle pixels in images and developed new descriptors that are appropriate for images. Using these descriptors led to algorithms that outperformed conventional techniques on speed and accuracy.
CHAPTER 1

INTRODUCTION

Image retrieval is the process of searching and retrieving images from a huge dataset \(^{(4)}\). One of the most popular image search techniques \(^{(5)}\) is based on human-supplied text annotations to describe image semantics (a concept-based approach) \(^{(6)}\). Text searching algorithms are then used to obtain similar images. However, there are problems with this technique. Different people may tag the same image differently due to emotive cues, context, and culture. Moreover, text annotations are not always available. There are recent efforts \(^{(7}; 8; 9)\) to generate text annotations automatically, but evaluating the accuracy of this process is as difficult as evaluating the accuracy of image retrieval itself.

To overcome the drawbacks of text-based image retrieval, content-based images retrieval (CBIR) was introduced \(^{(10}; 11)\). In contrast to concept-based approaches, content-based image retrieval (CBIR) approaches analyze images based on colors, shapes, textures in the images rather their annotations. In general, CBIR is more efficient, more accurate, and less expensive \(^{(6)}\). The approach proposed in this thesis belongs to this class.

Our algorithm involves several stages. We first rescale images. We then quantize these scales on three channels (Hue, Saturation, and Brightness) to obtain qualified pixels. Next, we compute descriptors of the configurations of these qualified pixels. Finally, we compute similarity of images based on their descriptors.
1.1 Summary of Contributions

The contributions of this thesis can be seen along the following dimensions:

- We devise a framework for extending feature descriptors based on scatterplots to the domain of time-varying data analysis. This induces a particular type of data reduction that allows for fast identification of interesting features such as outliers in high-dimensional data sets.

- We design an interactive system for visually mining doubly-multivariate data series using multiple visual metaphors in a novel combination.

- We implement a leader algorithm [12] to cluster similar scatterplots. We have developed a dynamic algorithm that leverages force-directed graph methods to cluster the leader scatterplots. This cluster layout provides a comprehensive summary of the 2D relations among variables in a dataset. Users can then select a cluster leader to investigate for further details.

- We propose a method for filtering features in order to refine subsets of scatterplots sharing common features.

- We develop new pixel distribution descriptors for characterizing images.

- We design an interactive environment for visualizing image clusters.

1.2 Thesis Organization

In Chapter 2, we introduce some related work in feature extraction and image retrieval.
In Chapter 3, we introduce a visually-based, graph-theoretic characterization of high-dimensional point sets that we call Scagnostics. These measures capture nine canonical features of statistical data typically observed by data analysts. This is a foundation for our work described in later chapters.

In Chapter 4, we introduce TimeSeer, an application using scagnostics as features to analyze large ordered point sets.

In Chapter 5, we introduce ScagExplorer, an application using scagnostics as features to analyze large unordered point sets.

In Chapter 6, we propose a set of image descriptors based on pixel distributions. We also discuss the performance of our descriptors on huge image databases in comparison to other well-known approaches in this field.

In Chapter 7, we present concluding discussions and future work.
CHAPTER 2

RELATED WORK

The number of image retrieval techniques has been increasing in the past decade. Some image retrieval techniques focus on a specific domain. For example, they can target the medical domain (13, 14) or cities and landscapes (15). Other image retrieval techniques work on general-purpose images. The general-purpose CBIR systems share one common procedure: descriptors are extracted from the query image. These descriptors are then compared to the images in a database using a selected distance measure such as Euclidean distance. Finally, the most relevant image (with the smallest distance to the query image) is returned. The accuracy of a CBIR system depends on how well these features characterize the target images and how suitable the distance measure is selected for these features.

The general-purpose CBIR systems can be classified into three categories: color histograms, region-based techniques, and feature-based techniques.

2.1 Color Histogram

Color histogram is a fundamental and commonly used technique (16, 17). This technique is popular because it is simple, intuitive to implement, and pretty accurate (18). The basic idea behind color histogram is learning how colors are distributed in images. To compare two color histograms, Euclidean distance is commonly used (16). Color histogram often use RGB (Red, Green, and Blue) color space for quantizing process. The two images are similar if they
have similar color histograms. But the reverse is not always true as depicted in Figure 1. In all four example images, 50% of pixels are red and 50% of pixels are blue. However, they are very different. A recent study [19] highlights the weaknesses and strengths of this model.

![Figure 1. Four different images result the same color histogram.](image)

### 2.2 Region-Based techniques

A region-based retrieval system decomposes an image into regions using image segmentation [20, 21]. Therefore, the accuracy of region-based techniques depends on the accuracy of automatic image segmentation. The techniques belonging to this category include NeTra [22], Blobworld [23]. These techniques present a user with the segmented regions of an image. The user selects regions and attributes to be matched. SIMPLIcity [24] is able to match all segmented regions automatically using integrated region matching. Due the the inaccuracy of image segmentation, the same object can be segmented into different numbers of regions for
different images. Integrated region matching tries to incorporate two region sets to compute similarity scores.

Another approach in this category is described in (25). This approach uses the dissimilarity function of Windsurf (26) for multi-region queries. Windsurf formulates the matching between two region sets as an optimization problem. Using efficient multi-step algorithms instead of a brute-force search enables this approach to work on larger image databases. Experiments on a database with more than 370,000 images have shown the multi-step approach outperformed the previous approaches.

2.3 Feature-based techniques

Tamura (27) proposed six textural features based on human perception. These textural features are coarseness, contrast, directionality, line-likeness, regularity, and roughness. Experiments have demonstrated that the first three features are more important. Since Tamura features aim to capture the textual properties of an image, they do not perform well on non-textured images (18).

The MPEG-7 standard includes color descriptors (such as the dominant color descriptor) and texture descriptors (such as the edge histogram texture descriptor) (28; 29; 30). These feature are computationally inexpensive. The best combination of these descriptors is color layout, dominant color, edge histogram, and texture browsing (31).

SIFT (Scale Invariant Feature Transform) features are invariant to image scaling, translation, and rotation (32). These features are used for object recognition. The recognition accuracy can be improved by incorporating other image aspects such as color and texture.
Gabor features (33) are inspired by text retrieval research. This approach uses an inverted file with more than 8,000 features per image. Weighting scheme in text retrieval is applied for frequency of occurrence of features. Relevant feedback is also used to improve the performance (precision and recall) of this approach.

The performances of CBIR techniques in this chapter are discussed in detail in an experimental study in 2008 ([18]). We will use the results in this study to compare to our proposed technique in chapter 6.

In the next chapters, we will present a method for assigning features to point sets on the plane. While this dissertation is concerned with images, the fundamental idea that motivates our treatment is to consider images, with suitable transformation, as point sets on the plane. Consequently, we will first discuss methods for characterizing point sets that can be extended and modify to handle images.
Scagnostics had been published \cite{1} several years before I was admitted to PhD program and joined this project in 2010. Scagnostics laid the foundation for my research during the four years of my PhD program. The following is a brief history of this topic.

In the mid 1980s, John and Paul Tukey proposed a method to characterize 2D scatterplots through a small set of measures. This method called Scagnostics. We can use these measures to detect interesting distributions of data points involving density, shape, trend, etc. We implemented the original Tukey idea through nine Scagnostics (Outlying, Skewed, Clumpy, Sparse, Striated, Convex, Skinny, Stringy, Monotonic) \cite{1}. We now review the Scagnostic algorithm, described in more detail in \cite{34,35}.

We compute all our measures on the binned points using the counts in each bin as weights. The Scagnostics measures depend on proximity graphs that are all subsets of the Delaunay triangulation: the minimum spanning tree (MST), the alpha complex \cite{36}, and the convex hull \cite{37}. Figure 2 shows an example of the three geometric graphs generated on the same set of data points.

3.1 Computing Scagnostic Measures

We now present the Scagnostic measures computed on the three geometric graphs. In the formulas below, we use $T$ for the minimum spanning tree, $A$ for the alpha hull, and $H$ for the
convex hull. We are interested in assessing three aspects of scattered points: density, shape, and association.

**Density Measures**

**Outlying:** The Outlying measures the proportion of the total edge length of the minimum spanning tree accounted for by the total length of edges adjacent to outliers.

\[ c_{outlying} = \frac{\text{length}(T_{outliers})}{\text{length}(T)} \]  \hspace{1cm} (3.1)

**Skewed:** This feature measures skewness in the distribution of edge lengths of the MST.

\[ q_{skew} = \frac{(q_{90} - q_{50})}{(q_{90} - q_{10})} \]  \hspace{1cm} (3.2)
**Sparse:** Sparse measures whether points in a 2D scatterplot are confined to a lattice or a small number of locations on the plane.

\[ c_{\text{sparse}} = q_{90} \] (3.3)

**Clumpy:** Clumpy is computed based on the MST: the RUNT statistic [38].

**Striated:** This measure is based on the number of adjacent edges in the MST whose cosine is less than -0.75. Let \( V^{(2)} \subseteq V \) be the set of all vertices of degree 2 in \( V \) and let \( I() \) be an indicator function. Then

\[ c_{\text{striate}} = \frac{1}{|V|} \sum_{v \in V^{(2)}} I(\cos \theta_{e(v,a)e(v,b)} < -0.75) \] (3.4)

**Shape Measures**

**Convex:** The convexity measure is based on the ratio of the area of the alpha hull and the area of the convex hull.

\[ c_{\text{convex}} = [\text{area}(A)/\text{area}(H)] \] (3.5)

**Skinny:** The ratio of perimeter to area of a polygon measures, roughly, how skinny it is.

\[ c_{\text{skinny}} = 1 - \sqrt[4]{\pi \text{area}(A)/\text{perimeter}(A)} \] (3.6)
**Stringy:** A stringy shape is a skinny shape with no branches. We count vertices of degree 2 in the minimum spanning tree and compare them to the overall number of vertices minus the number of single-degree vertices.

\[ c_{\text{stringy}} = \frac{|V^{(2)}|}{|V| - |V^{(1)}|} \]  

(3.7)

**Association Measure**

**Monotonic:** We use the squared Spearman correlation coefficient to assess monotonicity in a scatterplot.

\[ c_{\text{monotonic}} = r_{\text{spearman}}^2 \]  

(3.8)

### 3.2 Summary

Figure 3 shows an example that summarizes all nine scagnostics. In particular, the scatterplots with a low score on the associated scagnostic are on the left while the scatterplots with a high score on the associated scagnostic are on the right.

Before discussing applications in the image domain, we will present in the next two chapters applications we have developed to analyze point sets using scagnostics. Then, in the Chapter 6, we will show how these methods can be extended and modified to analyze images.
Figure 3. Nine scagnostics measures.
CHAPTER 4

VISUAL PATTERN DETECTION FOR LARGE ORDERED POINT SETS

Although the original motivation for Scagnostics was to locate interesting scatterplots in a large scatterplot matrix, we soon realized Scagnostics can be used for more applications. We have argued (34) that Scagnostics should be regarded as a type of projection that enables us to examine features in Scagnostics space and then make inferences about patterns that would not be apparent in the raw data space. Our platform for visual pattern detection for large ordered point sets rests on this fundamental principle.

4.1 Introduction

This chapter introduces an application, TimeSeer, for guiding interactive exploration through high-dimensional data. The application is based on nine characterizations of the 2D distributions of orthogonal pairwise projections on a set of points that have been introduced in Chapter 3. Working directly with these Scagnostic measures, we can locate anomalous or interesting distributions in large ordered point sets. In this chapter, we focus on point sets ordered by time (time series data). But the ordering here has more general meanings. For example, it can be the point sets showing the correlation of air pressure and temperature in different weather stations from North to South of the US East coast; or it can be the point sets showing the relationships between two variables along a DNA sequence.
4.2 TimeSeer

4.2.1 The TimeSeer GUI

The TimeSeer GUI incorporates two major systems: Variable Selection SPLOM and Time Series Viewer. The first enables the analyst to select Scagnostics and then pairs of variables. The second enables the analyst to view the selected pairs of variables in a full window. Figure 4 shows an example of TimeSeer GUI.

Figure 4. Two major systems of TimeSeer GUI in the US Employment data.

4.3 Examples

In this section, we use two different datasets to demonstrate the performance of TimeSeer. The first is a series of US Employment data and the second is a series of US Weather data.
The US Employment data comprise monthly employment statistics for 50 states over 22 years from 1990 to 2011. The data were retrieved from http://www.bls.gov/. There are 25 variables in the collected data. For these data, we have 78,600 scatterplots with 50 data points each to examine.

The Weather data comprise hourly meteorological measurements over a year from the Gulf of Maine in 2008. There are 17 variables represented in the dataset. Data and variable descriptions can be found at http://gyre.umeoce.maine.edu/buoyhome.php. For these data, we have 50,000 scatterplots with 24 data points (24 hours in a day) each to examine.

### 4.3.1 Variable Selection SPLOM

Here are the steps to do our analysis: 1) select the Scagnostic of interest, 2) select a criterion to order variables in SPLOM (mean or variance of the Scagnostic series), and 3) select a subset of the scatterplots, either by picking individual cells in the scatterplot matrix or by picking all frames corresponding to a single variable.

Figure 5(a) shows the scatterplot matrix for 25 variables in the US Employment data. We have selected the Convex measure and sorted the variables by their means. In particular, each plot (each pair of variables) is colored by its mean of the selected Scagnostic time series; the embedded small graph shows a thumbnail of the actual Scagnostic time series of each pair of variables. On the top of Figure 5(a) is the color legend for the mean of Convex Scagnostic time series. We use a Kelvin color temperature scale \([39]\) to encode the range of all possible Outlying mean values with red corresponding to high values of means and green corresponding to low values of means. This range (always within the 0 and 1 interval) is different when we
select a different Scagnostic feature. TimeSeer sorts the variables so that low Outlying series are at the bottom and high Outlying series are at the top. Consequently, most of plots with high Convex means are on the top and vice versa. Notice that we also color variable names to differentiate and group them by categories and subcategories.

TimeSeer offers two ways to select pairs of variables: single plot selection and single variable selection. Single plot selection is invoked by clicking on any of the panes in the scatterplot matrix as depicted in Figure 5(a). This selection mode allows the analyst to investigate specific Scagnostic series that show interesting patterns of behavior among the two featured variables. Single variable selection mode is invoked by clicking on the angled variable names to the right of the scatterplot matrix diagonal. Figure 5(b) shows Total Nonfarm selected. Black rectangles are used to denote selected plots. This selection mode allows the analyst to examine all variables paired with a specific variable of interest.

4.3.2 Lensing

Scatterplot matrices can easily run out of pixels when the number of variables increases. We use lensing to ameliorate this problem. TimeSeer offers two types of lensing. Figure 6(a) shows an implementation of Table Lens, in which a row/column is enlarged and the remaining rows/columns are reduced; Figure 6(b) shows smooth lensing on the top and in the corner, which involves a smooth lens so that distant rows/columns are reduced proportionally. Unlike the standard implementation of Table Lens, our smooth lensing offers a smooth transition when we move the mouse over different plots. Therefore, it is easier to keep track of the whole context (SPLOM) and the focused area.
Figure 5. Plot selection in SPLOM of 25 sorted variables in the US Employment data.
Figure 6. Lensing in scatterplot matrix of 25 sorted variables in the US Employment data.
4.3.3 **Time Series Viewer**

After we have selected pairs, we go to the Time Series Viewer. There are several ways to visualize multiple time series: small multiples or multiples superimposed with or without lensing. Figure 7 is an example. The series are built from ones we selected in the controller based on the US Employment data. We selected Monotonic as the Scagnostic for this example, and we chose 8 pairs of variables sorted by their means. Notice the slanted orientation of the second variable in each pair. This device helps the viewer to understand which variable is on the X axis and which variable is on the Y axis.

Figure 7(a) shows 8 small multiples corresponding to 8 pairs. The lensing in the Y-dimension is applied to the first three series, colored red, green, and blue respectively; the other pairs are colored gray and greatly reduced in size. We also employ a gradient on the lensed series to make the profiles more discernible and to coordinate highlighting with the scatterplots at the top of the window. The larger the series value, the lighter the coloring (like snow on mountains). This use of brightness also facilitates the highlighting of the scatterplots at the top. Each scatterplot corresponds to the appropriate colored Scagnostic series directly below it, and is highlighted with the same brightness as the point on the series.

For the arrangement we have selected, we notice that the Monotonic Scagnostic shows a distinct seasonal pattern with an annual cycle. This is consistent with what we would expect for variables related to farming.

In the US employment data, there are 262 data points on each Scagnostic time series. As we can see in Figure 7(a), however, the data points are crowded enough so that we cannot read
Figure 7. Visualization of US Employment data: a) Small multiples Overview with lensing on Y-dimension b) Small multiples with lensing on both X and Y-dimension c) Line graphs superimposed by plots with lensing on both X and Y-dimension
any details in a season. The common solution in this case is to display a selected season or interval. One limitation of this method is that when we select a season or interval, we lose the overall context of the time series. As a remedy, we chose to implement smooth lensing for the X-dimension. When we lens a season, we still can see what is going on in the other seasons throughout the entire time series. We can simply move the mouse to enlarge a different season. Moreover, smooth lensing allows continuous transition as we move the mouse.

Figure 7(b) shows a lens applied to the Scagnostic series. Vertical lensing is applied to three first pairs of variables and horizontal lensing are applied to two seasons (highlighted in Box A). The lensing works over the series as well as the scatterplots, so we are able to investigate individual scatterplots to see the configuration of points that led to the value of the Scagnostic shown in the series.

Figure 7(c) shows an alternate view of the same series. This view superimposes scatterplots on line graphs of the Scagnostics series. Such an arrangement allows investigation of individual scatterplots without anchoring or reference to a row of scatterplots elsewhere in the window. We believe this layout is useful once interesting segments are found in the series. In any case, toggling between views can be done in an instant.

4.3.4 Filtering, Brushing, and Drill-down

4.3.4.1 Filtering

We implemented range sliders on the left of each Scagnostic time series to allow users to filter only scatterplots with selected Scagnostic values in a specific interval (for example, with Monotonicity from 0.4 to 0.7). Figure 8(b) shows an example of filters applied to the Outlying
Figure 8. Visualization of the US Employment data: a) Lensing in interested area b) Filtering: Outlying ≥ 0.6.

series for the three pairs in the lensing area. We are looking for outliers. The filtered parts of time series are faded. In the pairwise view area, the filtered distributions are faded so viewers can focus on data distributions with high numbers of outliers.
4.3.4.2 **Brushing**

Brushing allow users to check out the data point in a scatterplot for further details. For example, we may want to see which state is the outlier in an Outlying distribution. Or, we may ask where is Illinois in the overall picture for 2010. Or, we may want to compare New York and California in 2012.

Figure 9(b) illustrates the use of a brush. When we brush a state, the state name is displayed in a tool tip, the same state is highlighted in other plots and a line appears connecting adjacent plots. This reveals, in effect, a spatial-temporal series.

4.3.4.3 **Drill-down**

Figure 10 shows an additional view invoked by a simple user action. The Scagnostic is Outlying, and the pairs of variables selected involve State Employment against \{Manufacturing, Leisure and Hospitality, Information, Other Services, Government, Service-Providing\}.

From the overview in Figure 10(a), we see a peak in several Scagnostic series. This suggests a time point in 2005 (highlighted in Box A) in which we would expect to see outliers in the relevant scatterplots. We lens this region in Figure 10(b). We see that a period in the Fall has an unusually high peak. This is the precise point where we expect to find an outlier. We get our details on demand, as shown in Figure 10(c), by clicking on the green scatterplots. Subsequently, the raw series of State Employment and Leisure and Hospitality (which are the 1-month net change in employment rate) are displayed in the lower graph on two separated scales (in cyan and pink). By brushing the outlier in the scatterplot of September of 2005, we see the actual line graph for that brushed state in yellow. In this case, the outlier is Louisiana. Hurricane
Figure 9. Visualization of the US Employment data: a) Lensing in interested area b) Brushing an outlier.
Katrina wreaked havoc on their employment and productivity figures (note the sudden drop in Louisiana employment rate and many industries). Notice that Louisiana is also the outlier in the scatterplot of December of 2005, even as it recovered.

4.3.5 Searching for Similar Patterns or Interesting Distributions

Upon finding an unusual distribution, one may want to look for similar ones. For example, one may wonder if there is another month having a similar distribution to the Katrina example in State Employment vs. Leisure and Hospitality in September of 2005. Others may want to see if other pairs of variables have similar distributions to the Katrina example in State Employment vs. Leisure and Hospitality in September of 2005.

TimeSeer offers several methods for discovering similar patterns in the Scagnostic series. The dissimilarity of two scatterplot \((S \text{ and } P)\) is computed by using the Euclidean distance on feature space.

4.3.5.1 Automatic Search for Similar Distributions

To perform a search, a user first selects a plot from the main screen. TimeSeer searches and plots the top 5 most similar scatterplots, as characterized by the selected Scagnostic (Outlying in this case). In the example depicted in Figure 11 taken from the US employment data, we have selected a plot with a high outlier on State Employment against Leisure and Hospitality in September of 2005. In the lower panel, the first plot on the left is the plot we selected, highlighted by a yellow rectangle. The five most similar plots are ordered over the nine Scagnostics (the smaller the index, the more similar). Again, the background of a plot is colored by the
Figure 10. Details on demand: a) Overview b) Lensing in interested area c) Requested actual data plots of an outlier (Louisiana).
time series containing that plot; saturation encodes the value of interested feature (the brighter the shade, the more salient the Scagnostic).

Figure 11. US Employment data, searching for distributions which are similar to State Employment against Leisure and Hospitality in September of 2005.

We also can see the time (on the top of each plot) for when the data distributions happen to be similar. The Scagnostics of the selected plot and top five plots are also grouped and ordered appropriately. From Figure 11 we note that it is interesting that Louisiana is the outlier in all six plots. Additionally, four out of five similar plots are in the same month (September of 2005).
This tells us that Hurricane Katrina affected Louisiana employment in the selected economy sectors.

Figure 12 shows a similar result for the Weather data. We have selected a plot with a high Stringy and Skinny Scagnostic value on barometric pressure vs. air temperature and searched for similar distributions in the same time series. This is a rather fascinating example of an unusual relation between variables that would not be evident in summary statistics such as the Pearson correlation. It is well-known that air temperature and barometric pressure are related, but these plots make clear that it is not a simple functional relationship. By searching for Stringy Scagnostics, we see that this dynamic relationship between barometric pressure and air temperature has little error (the strings/paths are quite smooth) but is highly nonlinear (they wind around instead of following a straight line).

4.3.5.2 Manual Search for Similar Distributions

We have devised an annotation that allows a user to search for similar plots manually. The user selects a plot from the main screen and TimeSeer computes the Scagnostic dissimilarity of each plot compared to the selected plot. It then displays this dissimilarity underneath the series. Figure 13(a) shows an example. We have selected an Outlying Scagnostic. We also selected the scatterplot in State Employment against Leisure and Hospitality in September of 2005. The similarity at each time point compared to the selected scatterplot is presented by the saturation (in purple) of the bar under it; the higher the saturation, the more similar the scatterplots. The slider at the bottom is used to filter similarity.
Figure 12. Weather data, searching for distributions which are similar to barometric pressure and air temperature on day 43.

The user can filter these plots to see only the most similar ones. Figure 13(b) shows an example. All plots with a dissimilarity greater than 0.6 have been filtered (by using the slider). The user can brush on the remaining dissimilarity purple bars or on the dot histogram to check the dissimilarity. When the mouse is over a purple bar or a plot in the dot histogram, a small window appears right below the purple bar. On this window, a new scatterplot and its Scagnostics is plotted next to the selected plot. In the example in Figure 13(b), we can see that the distribution of State Employment against Retail Trade in September of 2005 (and its Scagnostic histogram) is similar to the distribution of State Employment against Leisure and Hospitality (and its Scagnostic histogram) in the same time frame.
Figure 13. US Employment data, searching for distributions which are similar to State Employment against Accommodation and Food in September of 2005: a) Dissimilarity index for all selected pairs b) Filter applied: dissimilarity ≤ 0.6.
4.4 Conclusions

In this chapter, we presented an application for analyzing a doubly-multivariate time series. We normalize each variable independently to have the range 0 to 1 before computing Scagnostics measures. Consequently, TimeSeer is sensitive to how data is normalized. Variables that change the same amount in the normalized scale even with vastly different relative values, produce scatterplots that appear similar and Scagnostics do not help detect these changes especially if the changes in absolute values are small.

We have now seen that Scagnostics can be used to analyze large collections of ordered 2D point sets. In the next chapter, we will extend this capability to a larger class of unordered point sets. This methodology will be the basis for our derivation of a general method of computing similarity among images and analyzing large collections of images based on those similarities.
CHAPTER 5

VISUAL PATTERN DETECTION FOR LARGE UNORDERED POINT SETS

We have seen in the end of Chapter 4 the use of scagnostics to compute the similarity of point sets. This chapter will expand that capability to deal with unordered set of scatterplots in a very large corpus. TimeSeer has a moderate number of scatterplots (thousands of them). Now we are going to deal with millions of scatterplots.

5.1 Introduction

Previous researchers have applied data-driven approaches to moderate-sized collections of scatterplots (1, 34, 40). The scale of these efforts has been constrained by display limits and computational complexity. In this chapter, we develop an alternative approach in order to deal with the scalability problem for SPLOMs in terms of data sets larger than one hundred dimensions. Our goal is to be able to organize these plots into meaningful subsets in reasonable time and to present these plots to users in a rich exploratory environment.

5.2 Related Work

Seo and Shneiderman (41) computed statistical summaries (means, standard deviations, correlations, etc.) on univariate and bivariate distributions and then ranked them in order to identify similar distributions. Their Rank By Feature tool helps a viewer to navigate through a relatively large corpus of statistical data. Other researchers have developed scagnostics-type
measures for parallel coordinates, pixel displays, 3D scatterplots, and other graphics \num{42}, \num{43}, \num{44}, \num{45}, \num{46}, \num{47}.

As we have seen in Chapter \num{1}, TimeSeer \num{2} uses scagnostics for organizing multivariate time series and for guiding interactive exploration through high-dimensional data. TimeSeer consists of 2 systems: a SPLOM viewer and a time series viewer. The SPLOM viewer provides guidance for selecting interesting pairs of variables. Then, the time series viewer can graph scagnostics time series of up to 10 selected pairs of variables in a single display. A revision of TimeSeer \num{35} resolves this limitation by allowing a user to examine all time series in a corpus simultaneously.

5.3 ScagExplorer Overview

ScagExplorer is a platform for interactively visualizing scagnostics on multivariate distributions. ScagExplorer is designed to deal with a large scatterplot space containing thousands of scatterplots. While TimeSeer is devoted to ordered point sets, ScagExplorer is concerned with managing and exploring large collections of scatterplots.

5.3.1 Quality Metrics

A quality metric captures properties useful to the extraction of meaningful information about data. Quality metrics for detecting interesting patterns in high-dimensional data are described in \num{48}. We now briefly describe the quality metrics used in ScagExplorer.

What is measured. Clustering metrics measure the extent to which the data contain groupings. Our Clumpy measure captures this feature in a scatterplot. Correlation captures the extent to which systematic changes to one dimension are accompanied by changes in other
dimensions. Our Monotonic measure belongs to this category. Outlier metrics capture the extent to which the data segment under inspection contains elements that behave differently from the large majority of the data. Our Outlying measure is used to detect outliers in a scatterplot. Complex patterns metrics capture aspects that cannot be easily categorized as any of the classes described above. Other scagnostic measures, such as Stringy, Striated, and Skinny, belong to this category.

**Where it is measured.** Our quality metrics can be calculated in *data space* or *image space*. In the case of small-sized data (less than 200 data points in each scatterplot), scagnostics can be computed directly on normalized data. For large-sized data, scagnostics can be computed on binned data. This makes our quality metrics scalable.

**Analysis purpose.** ScagExplorer can be used for the following purposes. Projection aims at finding 2D projections in which interesting patterns reside. For example, Skinny and Stringy measures are used to identify unusual correlations of variables in data (see Section [5.4.4]). Abstraction aims at providing an overview by clustering scatterplots in entire dataset based on their scagnostic measures (see Section [5.4.1]). Visual mapping aims at coloring scatterplots by their measures or similarity to a selected plot. This helps a viewer to discern patterns of scatterplot distributions (see Section [5.4.2]).

**Interaction with the quality-metrics-based automation.** ScagExplorer offers both threshold selection and metrics selection. For example, users can filter only high Outlying plots or combine filtering on all nine scagnostic measures (see Section [5.4.4]).
TABLE I

Characteristics of datasets for ScagExplorer Demonstrations.

<table>
<thead>
<tr>
<th>Datasets</th>
<th># Instances</th>
<th># Attributes</th>
<th># Scatterplots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast Cancer</td>
<td>569</td>
<td>32</td>
<td>496</td>
</tr>
<tr>
<td>Sonar</td>
<td>208</td>
<td>60</td>
<td>1,770</td>
</tr>
<tr>
<td>Libras</td>
<td>360</td>
<td>91</td>
<td>4,095</td>
</tr>
<tr>
<td>Subway</td>
<td>423</td>
<td>104</td>
<td>5,356</td>
</tr>
<tr>
<td>Communities</td>
<td>1,994</td>
<td>128</td>
<td>8,128</td>
</tr>
<tr>
<td>Gas Sensor</td>
<td>3,600</td>
<td>129</td>
<td>8,256</td>
</tr>
<tr>
<td>Baseball MLB 2008</td>
<td>227</td>
<td>133</td>
<td>8,778</td>
</tr>
<tr>
<td>Musk</td>
<td>476</td>
<td>167</td>
<td>13,861</td>
</tr>
<tr>
<td>Isolet</td>
<td>1,559</td>
<td>167</td>
<td>13,861</td>
</tr>
<tr>
<td>Madelon</td>
<td>1,042</td>
<td>500</td>
<td>124,750</td>
</tr>
<tr>
<td>Secom</td>
<td>1,567</td>
<td>590</td>
<td>173,755</td>
</tr>
<tr>
<td>Arcene</td>
<td>100</td>
<td>2,000</td>
<td>1,999,000</td>
</tr>
<tr>
<td>Cancer</td>
<td>198</td>
<td>2,000</td>
<td>1,999,000</td>
</tr>
</tbody>
</table>

5.3.2 Datasets

We will illustrate the features of ScagExplorer mainly through examples. We use datasets retrieved from UCI Repository and other sources to demonstrate the performance of ScagExplorer. Table I summarizes prominent aspects of these datasets ordered by the number of attributes.

5.4 ScagExplorer Components

This section explains our approach in detail. Figure 14 shows a schematic overview:

1. **Processing:** Our approach computes nine scagnostics measures of each scatterplot in the input SPLOM. Then, scatterplots are clustered based on scagnostics space.
2. **Visualization:** The leader plots in each cluster are displayed in the force-directed layout.

3. **Interaction:** Users can select a leader to see all similar plots in that cluster or filter scatterplots by their features.

### 5.4.1 Clustering Algorithm

The scatterplot matrix is a useful tool for displaying the correlation between a pair of variables. However, it is easy to run out of space as the number of variables increases. We use the leader algorithm \([12]\) to cluster scatterplots. The input of this algorithm are the list of all scatterplots and an initial threshold \(r\) (radius around a clusters center, in which inputs must fall). Here is the summary of the algorithm:

1. We initialize the leader list \(L = \emptyset\).
2. For each scatterplot $S_i$, we find the nearest leader (center) in $L$ which has the squared Euclidean distance to $S_i \leq r$.

3. If we could not find a nearest leader satisfying this condition, we make $S_i$ a new leader and add $S_i$ into $L$.

4. Otherwise, we add $S_i$ to the follower (children) list of the nearest leader.

5. Repeat steps 2-4 for all scatterplots $S_i$.

6. Now we have a complete leader list $L$, we repeat step 2 and step 4 once for all scatterplots $S_i$ to avoid the mistakes of finding the nearest leader when $L$ is incomplete. We do not need to repeat step 3 because we are simply reassigning followers in case closer leaders emerged in the first pass through the data. This reassignment is similar to the iterative reassignments in the k-means algorithm \cite{[12]}, but we do only one more pass through the data.

Too many leaders makes the visualization too busy while too few leaders over-summarizes the data set. We limit the size of the leader list $L$ from 15 to 30 to avoid these problems. This means that we need to adjust the threshold $c$ and repeat steps 1-6 a few times to get the right number of leaders.

How to select the initial threshold $r$? The distribution of squared Euclidean distances is approximately a noncentral Chi-square variable, assuming the distances themselves are roughly normal \cite{[50]}. However, we chose an empirical method on real data in order to relax the normality assumption. From results on 20 different datasets of various sizes retrieved from UCI repository
The threshold $r$ that produced approximately 20 clusters varied from 0.5 (for small datasets with thousands of scatterplots) to 2.0 (for large datasets with millions of scatterplots) on a possible range of 0 to 9. Therefore, we initialize $r = 1.5$.

There are two ways to adjust $r$:

- **Binary search** is a quick way to get to the right threshold $r$ for 15 to 30 leaders. However, it is possible that large datasets may end up with fewer leaders than smaller datasets.

- To avoid this problem and since we already know the range of $r$, ScagExplorer increments $r$ by a small step (0.1 in this case). This method works well on real data because the initial choice of $r = 2$ almost always produces fewer than 30 clusters. In the rare cases where we have too many clusters, we search upward on $r$ to produce fewer clusters.

The complexity of this algorithm is $O(p)$ or $O(v^2)$ compared to $O(v^3)$ of Lehmann’s reordering algorithm (where $p$ is the number of scatterplots and $v$ is the number of dimensions). This also means that ScagExplorer can handle higher-dimension datasets.

### 5.4.2 Displaying the leader scatterplots

After having clusters and their leader plots, we now use the force-directed layout to place them on a 2D view. An alternative to this presentation is Multidimensional Scaling (MDS) to project nine-dimensional scagnostics space to positions in 2D space. The drawback of MDS is that it produces overlapped scatterplots for similar distance measures, which makes it difficult to visualize cluster sizes.

The advantages of force-directed layouts are intuitiveness, flexibility, and interactivity. The main disadvantage is high running time. Since for every plot, we have to compute the attraction
or repellant against all other plots, the running time at each iteration is $O(l^2)$ (where $l$ is the number of leader scatterplots). However, $l$ is much smaller than $p$ and we can control the clustering algorithm so that it generates a reasonable numbers of leader plots (less than 30 leader plots). This running time is not remarkable.

In the force-directed layout, we first put all leader plots randomly in the output panel and we then allow them to interact to find similar leader plots based on their scagnostics measures. Consequently, similar leaders are grouped together. This makes it easier to interpret the clustering results. Here is the summary of the force-directed algorithm:

1. For each pair of leader scatterplots $S_i$ and $S_j$, we compute a dissimilarity measure as the squared Euclidean distance in feature space.

2. We get dissimilarity cut $D$ as $c + 0.2$ where $c$ is the radius of the clustering algorithm. We then define $D_{ij} = Dissimilarity(S_i, S_j) - D$.

3. We compute $\vec{U}_{ij}$ as the unit vector from $S_i$ to $S_j$.

4. If $D_{ij} \leq 0$, $\vec{F}_{ij}$ is the attraction between $S_i$ and $S_j$ computed by the following equation:

   $$\vec{F}_{ij} = D_{ij} \cdot \vec{U}_{ij}$$  \hspace{1cm} (5.1)

5. If $D_{ij} > 0$, $\vec{F}_{ij}$ is the repulsion of $S_j$ on $S_i$.

   $$\vec{F}_{ij} = \frac{D_{ij} \cdot \vec{U}_{ij}}{Distance(S_i, S_j)}$$  \hspace{1cm} (5.2)
6. The force applied on $S_i$ is the sum of forces by all scatterplots on $S_i$ (l is the number of leader scatterplots):

$$\vec{F}_i = \sum_{i=1}^{l} \vec{F}_{ij} \quad (5.3)$$

7. Repeat steps 3-6 for all leader scatterplots $S_i$.

Figure 15 shows how we display the leader plots of three different datasets (Breast Cancer, Sonar, and Libras) in the forced-directed layout. In particular, the frames on the left summarize thousands of scatterplots in each dataset. The size of each leader plot is computed based on its cluster size (or the number of scatterplots in each cluster). The Kelvin temperature color scale is adopted to highlight Monotonicity (red plots are high Monotonic, blue plots are low Monotonic). In the frames on the right, we aligned the leader plots on X-axis based on their Monotonicity. This alignment reveals the density distribution of scatterplots on Monotonicity. The forced-directed layout makes transitions from left frames to right frames smoothly. We only have to tell the leaders where they should go, then the leader plots fit themselves into the display area and avoid overlappings between them.

Users can select a different scagnostics measure from a list box to highlight and/or align leader plots. Figure 16 shows the alignment of leader plots in larger datasets (Subway, Communities, and Gas Sensor Array Drift) on Outlying. The datasets respectively contain 104 attributes (5,356 scatterplots), 128 attributes (8,128 scatterplots), and 129 attributes (8,256 scatterplots). We now use the Kelvin temperature color scale to highlight Outlying (red plots are high Outlying, blue plots are low Outlying). As depicted in the right frames, the Subway
Figure 15. Visualizing the Breast Cancer, Sonar, and Libras data. Left frames: overview layout. Right frames: Leader plots are aligned on X-axis (Monotonic).
data are very high Outlying while the Gas Sensor data are very low Outlying. Notice that we have requested to display the number of scatterplots in each cluster on top of each leader plot. Similarly, Figure 17 shows how we display the leader plots of very large datasets (Madelon, Arcene, and Cancer). The datasets respectively contain 500 attributes (124,750 scatterplots), 2,000 attributes (1,999,000 scatterplots), and 2,000 attributes (1,999,000 scatterplots). Notice that we use circles instead of rectangles to present scatterplots since we characterize a scatterplot with orientation-independent features. This option creates a better effect on displaying density compared to rectangle shapes.

5.4.3 Exploring similar scatterplots

After having an overall idea of all scatterplots in the data, one may want to request the details in each cluster. This can be done by a simple click on a leader plot. Figure 18(a) shows an example for the Libras data. In particular, scatterplots are colored by their Monotonicity. The selected leader is in the center surrounding by scatterplots in its cluster. On the right, we link where they are in SPLOM. The same effect is shown in Figure 18(b) with a different leader. The linked SPLOMs reveals the monotonic pattern: the closer to the diagonal, the higher Monotonicity. This is because variables in input data have been ordered so that highly correlated variables locate close to each other.

5.4.4 Filtering Scagnostics

Another way to drill-down the large collection of scatterplots is filtering by scagnostics measures. ScagExplorer offers two ways to filter scatterplots: a Parallel coordinates controller and a Rose controller.
Figure 16. Visualizing the Subway, Communities, and Gas Sensor Array Drift data. Left frames: overview layout. Right frames: Leader plots are aligned on X-axis (Outlying).
Figure 17. Visualizing the Madelon, Arcene, and Cancer data. Left frames: overview layout. Right frames: Leader plots are aligned on X-axis (Outlying).
Figure 18. Expanding all scatterplots of a selected cluster in the Libras data.
5.4.4.1 Filtering Scagnostics with Parallel Coordinates

Figure 19 shows an example of filtering scagnostics by parallel coordinates. The data are NRC university rankings in Mathematics. The National Research Council (NRC) ranking data comprise university rankings in Mathematics in 2006 according to different criteria. There are 33 variables represented in the dataset: R-Rankings, S-Rankings, ranking factors and information on 127 universities in the US. For S-Rankings, programs are ranked highly if they are strong in the criteria that scholars say are most important. For R-Rankings, programs are ranked highly if they have similar features to programs viewed by faculty as top-notch. Overall, we have 528 scatterplots with 127 data points (127 universities) to examine. In particular, each coordinate corresponds to a scagnostic measure. Colors are used to differentiate scagnostics. All scagnostics are in a common range from 0 (left) to 1 (right). There are 528 scatterplots in the NRC university ranking data, and thus the same number of polylines in parallel coordinates. The symmetric graph on each coordinate shows the density distribution of scatterplots in the entire search space according to each measure. When we are filtering a measure, ScagExplorer updates the density graphs on other measures showing only the remaining plots satisfying the filtering conditions. This guides users on making interactive scagnostic selections.

Figure 20(a) shows an example when this filter is applied on Monotonicity (Monotonicity $\geq 0.5$). When we filter plots on one scagnostic, other graphs showing the distribution of selected scatterplots are updated. The forced-directed layout on the right shows 31 plots satisfying the condition. We can obtain correlated variables from these high Monotonic scatterplots.
Figure 19. University ranking data, filtering Scagnostics by parallel coordinates.
After viewing pairs of variables that are highly correlated, one can request to see the variable relationship graph. Figure 20(b) shows the variable relationship graph of 31 monotonic plots in Figure 20(a). Each node in this graph represents a variable and each edge exists if a pair of variables exists in Figure 20(a). Red edges connect highly correlated variables and blue edges connect uncorrelated variables (although blue edges have been filtered out in this figure). Notice that Research Activity is the main variable involved with both R Rankings and S Rankings. Other variables which have moderate influence on rankings are percent of faculty with grants, citation per publication, average number of PhD Students graduated, and publication per allocated faculty.

5.4.4.2 Filtering Scagnostics with a Rose Controller

Filtering scagnostics using a Nightingale Rose controller is similar to filtering scagnostics using parallel coordinates. To make our controller, we divide a circle into 9 sectors. We then divide each sector into 3 selection areas: outer, middle, and inner. Figure 21 shows an example of filtering Clumpy. By selecting the inner area (numbered 1 in the figure), we can control the lower bound of the range slider for Clumpy. By selecting the middle area (numbered 2), we can move both ends of the range slider at the same time. By selecting the outer area (numbered 3), we can control the upper bound of the range slider. This controller is directly related to the Scagnostic View. The filtering condition of Figure 21 is $0.45 \leq \text{Clumpy} \leq 0.75$.

Figure 22(a) shows an example of Communities data where we apply filtering on two measures: Monotonic $\geq 0.5$, Outlying $\geq 0.5$. In Figure 22(b), we filter out more plots by setting Monotonic $\geq 0.7$, Outlying $\geq 0.7$. Figure 22(c) shows the Scagnostic View of the filtered scat-
Figure 20. NRC university ranking data, filtering high Monotonic plots.
terplots. This view reveals how selected scatterplots conform to the filtering conditions. The text on the left shows the number of scatterplots that satisfy the selection criteria in the entire search space.

5.5 Conclusions

This chapter proposes a framework for visualizing high dimensional data where the number of scatterplots is too large to be visualized by a ordinary SPLOM. The greater runtime efficiency allows ScagExplorer to provide a quick and comprehensive summary of the input data. Then, we can drill-down in the data by inspecting a cluster or filtering using a target scagnostics measure.

Finally, we plan to investigate the use of ScagExplorer on large security databases to assess the gains we claim for its performance. In addition, we expect to investigate how ScagExplorer can be extended to temporal and spatial data. Time and space have similar statistical issues when modeling [52], so addressing them together makes sense.
Figure 22. Filtering scagnostics by Rose Controller: a) Monotonic $\geq 0.5$, Outlying $\geq 0.5$ b) Monotonic $\geq 0.7$, Outlying $\geq 0.7$ c) Scagnostic View.
CHAPTER 6

VISUAL PATTERN DETECTION FOR IMAGES

An image is a scatterplot. This rather odd sentence encapsulates how we consider an image in the context of this research. In particular, we transform an image into a set of low-level, point-based representations. Each representation includes a set of pixels satisfying certain filtering conditions. These representations incorporate different descriptive aspects of an image. For each presentation, we then compute its descriptors based on the filtered pixels. The descriptors include a subset of the Scagnostics and additional features that represent Gestalt concepts of visual perception. We begin with the Gestalt theory.

6.1 Gestalt Theory of Visual Perception

Gestalt refers to theories of visual perception developed by German psychologists in the 1920s. The key idea of Gestalt theory is when we see a group of objects in a complex scene, we perceive the entire scene before we see the constituent objects. Gestalt principles are laws of organizing perceptual scenes. These principles are proximity principle, figure-ground articulation, similarity principle, common fate principle, continuity principle, and closure principle. Our image descriptors rely on a subset of Gestalt principles, including proximity principle, figure-ground articulation, and similarity principle, that we will describe in the following section.
6.1.1 Principle 1 - Proximity

According to the law of proximity, elements that are closer together are perceived as more related (as belonging to the same group) than elements that are further apart (54). Figure 23 shows an example of proximity principle. In Figure 23(a), the thirteen squares are placed without proximity. They are perceived as separate shapes. In Figure 23(b), the thirteen squares are now perceived as two groups.

Figure 23. Proximity principle: a) The thirteen squares are placed without proximity b) The thirteen squares are placed with close proximity to form two groups. The figure was drawn by the author.

6.1.2 Principle 2 - Figure and ground

According to the law of figure and ground, we seem to have an innate tendency to perceive elements as either figure (the element in focus) or ground (the background on which the figure
The figure and ground relationship can be unstable. The most famous example of figure and ground ambiguity is the faces-vase drawing of psychologist Edgar Rubin depicted in Figure 24.

![Figure 24](http://en.wikipedia.org/wiki/File:Cup_or_faces_paradox.svg)

**Figure 24.** Figure and ground principle: a) On the white background, the two black faces gain the better impression as the element in focus b) On the inverted image, the central region resembling a vase appears to be the figure.

There are several factors that impact our perception in detecting the positive element (figure) in an image. Regions that are convex, smaller in area, wide base, symmetric, surrounded, or lower are more likely to be seen as figure than contiguous regions that are concave, larger in area, narrow base, asymmetric, surrounding, or upper. These properties are illustrated in Figure 25.
6.1.3 **Principle 3 - Similarity**

The similarity principle claims that elements tend to be integrated into groups if they are similar to each other [53]. As illustrated in Figure 26, elements are perceptually partitioned into three pairs as the elements in each pair share a common visual property such as lightness (Figure 26(a)), color (Figure 26(b)), size (Figure 26(c)), orientation (Figure 26(d)), or shape (Figure 26(e)).

This section does not mean to present complete research of perceptual theory in a few pages. Instead, we review the key principles of visual perception that are related to our work in the
next chapter. These principles can be considered as a background to build up the features of our content-based image retrieval system.

![Figure 26. Examples of Gestalt principle of similarity](http://www.scholarpedia.org/article/File:Todorovic-Gestalt_principles-Figure_3.jpg)

### 6.2 PixSearcher

#### 6.2.1 PixSearcher Design

[Figure 27](http://www.scholarpedia.org/article/File:Todorovic-Gestalt_principles-Figure_3.jpg) shows a schematic overview of our image retrieval approach. We design our system as close as possible to the way the human perceive objects. In fact for every stage in the bottom-up process (“retina image → features → patterns → visual objects” - Collin Ware [57], pages 10 to 12) , there is a corresponding stage in our approach. In particular, a query

---

1. [http://www.scholarpedia.org/article/File:Todorovic-Gestalt_principles-Figure_3.jpg](http://www.scholarpedia.org/article/File:Todorovic-Gestalt_principles-Figure_3.jpg)
image is preprocessed (rescaled and then quantized) to create low-level representation of the query image. Image descriptors are then computed on the quantized images. These descriptors collectively represent the query image. The set of features of the query image is compared to images in the training image dataset. Finally, PixSearcher outputs the most similar image in the training dataset. The most similar image has the smallest distance in the feature space to the query image.

Figure 27. Schematic overview of PixSearcher.
Humans are the users of CBIR systems who are able to access the validity of the output. Therefore, we design our image descriptors based on visual perception theory briefly reviewed in Section 6.1. These descriptors are extended from scagnostics and adapted to the image context based on the Gestalt principles. In other words for our design, we can think of Gestalt principles as the pull factors which draw our image descriptors close to the visual features of the bottom-up process. This is important for archiving a more accurate CBIR system. Here is an example of this reasoning. Color histogram divides RGB color space into a uniform grid of 8 by 8 by 8 then counts the number of pixels in the query image falling into each color cell. Using color histogram, we can come to the conclusion for the example image in Figure 28: 35 percents of the pixels are red and 65 percents of the pixels are green. However, there is a better way to describe the example image: 35 percents of the pixels are red and located in the center of the image; these red pixels are surrounded by 65 percents of green pixels in the background. The Section outlines our algorithms and how we can get as close as possible to these visual descriptions.

6.2.2 Quantizing Images

We begin by rescaling the query image into 40 by 40 pixel arrays. The choice of rescaling size is constrained by efficiency (too many pixels slow down calculations) and sensitivity (too few pixels obscure features in the images). We map each pixel \( p_{ij}, i = 1, \ldots, 40, j = 1, \ldots, 40 \) to the tuple \( (x_k, y_k), k = 1, \ldots, 1600 \), where \( 0 \leq x_k \leq 1 \) and \( 0 \leq y_k \leq 1 \).

Next, we quantize the 40 by 40 pixel images on Hue, Saturation, and Brightness (HSB). For each of these color dimensions, we produce \( q \) binary pixel arrays (the selection of \( q \) will be
justified later in this Chapter). In each of these arrays, a pixel takes the value 1 if its color dimension lies within a quantile and 0 otherwise. We then restrict our tuples to the set $X$ containing the tuples corresponding to quantized pixels with value 1.

Why do we choose to quantize the rescaled images on Hue, Saturation, and Brightness (HSB) instead of Red, Green, and Blue (RGB)? “The standard red-green-blue (RGB) color space is not very useful for color processing, as distances in RGB space have little meaning and there is no simple (even approximate) mapping from RGB coordinates to human color names. A hue-based space such as HSV is superior to RGB in these respects” (58; 59). Converting the color images from RGB into another color spaces such as HSB and processing these images gives better results (60; 61). For that reason, there are many more recent CBIR techniques that choose to work with HSB color space (62; 63). A 2013 survey and comparison between the RGB and HSB models has come to the same conclusion that HSB model accuracy is higher than that of the RGB model (64).
Figure 29 and Figure 30 show two examples of quantizing images on HSB and RGB color space. In particular, the rescaled versions of the input images are on the top of each figure. The first three rows display quantization results of the rescaled image on HSB color channels (Hue, Saturation, and then Brightness). The next three rows display quantization results of the rescaled image on RGB color channels (Red, Green, and then Blue). In these examples on each color channel, we apply 8 quantizations (0 to 0.125, 0.125 to 0.25, 0.25 to 0.375, 0.375 to 0.5, 0.5 to 0.625, 0.625 to 0.75, 0.75 to 0.875, and 0.875 to 1). The black dots in the results are the quantized pixels. We can see that the quantization results of the same image on RGB and HSB are completely different.

We compare the effectiveness of quantization on the two color spaces in Figure 31. In particular, the quantization results on Red, Green, and Blue (the last three rows) are pretty similar. That is, they contain redundant information of the input image (in this case, image 426 of the Wang dataset described in Section 6.3.1). In contrast, the quantization results on Hue, Saturation, and Brightness are different. In fact, they contain different aspects of the input image: the first quantized image on Hue contains the dinosaur; the second quantized image on Hue contains the floor; the first quantized image on Saturation contains only the background behind the dinosaur; the last quantized image on Brightness contains the surroundings of the dinosaur (floor and background). These components are all meaningful in understanding the content of the input image. Notice that none of the quantized images on RGB displays the full body of the dinosaur (all of the black pixels constitute the dinosaur). Our algorithm uses these
Figure 29. Comparison of quantization results on RGB and HSB color space of a sample image (RGB color cube). The input image has been retrieved from (65).
Figure 30. Comparison of quantization results on RGB and HSB color space of a sample image (HSB cylindrical color wheel). The input image has been retrieved from [66].
quantized black pixels to compute features of the input image, which we will describe in the next section.

6.2.3 Computing Descriptors

As mentioned above, our image descriptors are extended from our work on scagnostics. This does not mean that we bring the whole set of scagnostics to apply for pixels in images (notice that some scagnostics features are inappropriate to the image context, for example, Monotonic and Outlying). Instead, we select suitable scagnostics features and adapt them to the image context. These extended features are Close, Dense, Convex, Skinny, and Clumpy. Additionally, we develop some more features in accordance with the Gestalt principles reviewed in Section 6.1. Namely, the set of new features include Pixel level, Vertical symmetry, Horizontal symmetry, and Lower. We begin with the new features.

1. **Pixel level:** Since size is one important characteristic to compare/group two elements (according to Gestalt principle of similarity), two similar images should contain approximate numbers of pixels in each pair of quantized images. This is a necessary but not sufficient condition for the resemblance of two images. To compute the Pixel level feature, we first count the number of pixels satisfying the filtering conditions, namely, $|X|$. This number is then normalized by the maximum number of pixels in the rescaled image.

$$f_{\text{pixel level}} = |X|/(40 \times 40) \quad (6.1)$$
Figure 31. Comparison of quantization results on RGB and HSB color space of image 426 in the Wang dataset described in Section 6.3.1.
2. **Vertical symmetry:** This feature measures whether the data points are equally distributed around the horizontal central line. For each data point, we measure how far it is from the central line. That is, the bottommost pixels receive the value 20 and the topmost pixels receive the value -20. We then sum these distances $Sum_V$ and get the absolute value of the sum. In the worst case, all data points ($20 \times 40$ data points) are on top/bottom area; $|Sum_V|$ is 8000 ($20 \times 40 \times 10$, 10 is the average distance to the horizontal central line of all pixels). Finally, we normalize this feature by dividing $|Sum_V|$ to the score of the worst case.

$$f_{\text{vertical\_symmetry}} = |Sum_V|/(20 \times 40 \times 10) \quad (6.2)$$

3. **Horizontal symmetry:** This feature measures whether the data points are equally distributed around the vertical central line. For each data point, we measure how far it is from the central line. Specifically, the rightmost pixels receive the value 20 and the leftmost pixels receive the value -20. We then sum these distances $Sum_H$ and get the absolute value of the sum. Finally, we divide this value to the score of the worst case when 8000 data points lie on side (left or right) of the frame.

$$f_{\text{horizontal\_symmetry}} = |Sum_H|/(20 \times 40 \times 10) \quad (6.3)$$

The two symmetric features are accordance with Gestalt principle 2 (figure and ground) in perceiving the positive element (figure) in an image: Symmetrical rather than asymmetrical patterns tend to be perceived as figures [56]. Moreover, if an image is symmetric on both
dimensions, it tends to contain a focus object in the middle surrounded by the ground according to the Gestalt principle of figure and ground: Surrounded region tends to be perceived as figure while surrounding area is usually seen as ground (56).

4. **Lower:** According to Gestalt law of figure and ground, lower rather than upper region tends to be perceived as figure (56). The position of patterns/regions is important to how human interpret their meanings. For example, a blue area on the top of the left image can be quickly perceived as the sky while a blue region at the bottom of the right image can be quickly interpreted as the sea as depicted in Figure 32.

![Figure 32. Two low resolution versions with different orientations of the same image (image 886 in the Wang dataset described in Section 6.3.1; this image has been altered). The left image can be interpreted as beach and sea. The right image can be interpreted as mountain and blue sky.](image)

Although some CBIR systems (24; 67; 68) try to capture the meaning of an image independent of rotation, different rotations may be significant on how we interpret an image. “We know
that the pattern of resemblance depends both on stimulus geometry and also the perceptual reference frame that is organizing that geometry (i.e., the specification of figure/ground, the assignment of orientation, and so on). This point has been well documented in perception, and it seems highly likely that it will apply to imagery as well. It follows from these premises, then, that discoveries will flow easily from an image only if the discovery is compatible with both the depicted geometry and also the reference frame.” - Priti Shah and Akira Miyake (69), page 49 and 50.

Our Lower descriptor measures the average elevation of the quantized pixels. If 8000 pixels lie on the bottom half of the quantized image, then the Lower feature receive the value of 1. If 8000 pixels lie on the top half of the quantized image, then the Lower feature receive the value of 0. Clearly, this feature is rotation dependent. A Lower quantized image implies the vertically asymmetric distribution of quantized pixels but not the reverse. Similar to the Pixel level, Vertical symmetry, and Horizontal symmetry features, Lower requires linear time $O(n)$ to compute where $n$ is the number of quantized pixels.

We now outline the computation of five other descriptors which have been extended from scagnostics and adapted to the image context. These descriptors are computed based on proximity graphs that are subsets of the Delaunay triangulation. In the formulas below, we use $H$ for the convex hull, $A$ for the alpha hull, and $T$ for the minimum spanning tree (MST). Each of these proximity graphs is computed on the set $X$ of quantized image.

5. **Close:** Our Close descriptor is derived from the Outlying scagnostic. This scagnostic is used to detect the data points that are notable separated from the remaining vertices in the
MST. As opposed to the Outlying scagnostic, our Close descriptor measures the closeness of pixels in the quantized image. The Close descriptor is computed based on the proportion of the total edge length of the MST accounted for by the total length of edges connecting adjacent quantized pixels (edges of length 1).

\[ f_{\text{close}} = \frac{\text{length}(T_1)}{\text{length}(T)} \] (6.4)

6. Dense: Our Dense descriptor compares the area of the alpha shape to the area of the whole frame. Low values of this statistic indicate a sparse image. This descriptor addresses the question of how fully the quantized pixels fill the frame.

\[ f_{\text{dense}} = \frac{\text{area}(A)}{(40 \times 40)} \] (6.5)

Examples of the Close and Dense features are given in Figure 33.

7. Convex: According to the Gestalt principle of figure and ground, convex rather than concave patterns tend to be perceived as the focus element of an image (54). Our convexity measure is based on the ratio of the area of the alpha hull and the area of the convex hull. This ratio will be 1 if the alpha hull and the convex hull have identical areas.

\[ f_{\text{convex}} = \frac{\text{area}(A)}{\text{area}(H)} \] (6.6)
Figure 33. Top image shows high Close and sparse distribution. Bottom image shows low Close and dense distribution.

8. Skinny: The ratio of perimeter to area of a polygon measures, roughly, how skinny it is. We use a corrected and normalized ratio so that a circle yields a value of 0, a square yields 0.12 and a skinny polygon yields a value near one.

\[ f_{\text{skinny}} = 1 - \sqrt{\frac{4\pi \text{area}(A)}{\text{perimeter}(A)}} \]  

Examples of the Convex and Skinny features are given in Figure 34.

9. Clumpy: According to the Gestalt principle of proximity, elements close to each other are perceptually formed a group \(^{(54)}\). The Clumpy feature aims to detect clusters of data points. Different from the Clumpy scagnostic (which is based on the RUNT statistic \(^{(38)}\)), our Clumpy image descriptor is computed based on the number of alpha shapes \(|A|\). Examples of the Clumpy features are given in Figure 35.
Figure 34. Top image shows high Convex and low Skinny distribution. Bottom image shows low Convex and high Skinny distribution.

Figure 35. Top image shows high Clumpy distribution. Bottom image shows low Clumpy distribution.
6.2.4 Dissimilarity of Two Images

We apply 8 quantizations for each color dimension. Then, we compute 9 descriptors for each quantized image. Totally, each image is characterized by 216 features. The dissimilarity of two images \((S, P)\) is computed by using Euclidean distance on their feature space. The computation is given in the following equation:

\[
\text{Dissimilarity}(S, P) = \sqrt{\sum_{c=1}^{3} \sum_{q=1}^{8} \sum_{f=1}^{9} W_{c,q,f}(S_{c,q,f} - P_{c,q,f})^2}
\]

(6.8)

where \(S\) and \(P\) are two arrays of 9 features \((f)\) for 8 quantizations \((q)\) on 3 color dimensions \((c)\) of the two images and \(W_{c,q,d}\) is the weight of each descriptors (it is user input). ScagExplorer allows users to set weights for different descriptors/channels depend on images/application domains. In this chapter, we use the same weights for all descriptors.

6.3 Testing

6.3.1 Benchmark databases for CBIR

We use 4 benchmark image databases from different domains for evaluation. These data have been used in a survey paper in 2008 [18]. We were able to confirm some evaluation results in the paper. We will use these evaluation results to compare to PixSearcher’s performance. Table II summarizes prominent aspects of the 4 benchmark image datasets. These benchmark image databases are freely available.

The Wang dataset: The Wang database [70, 24] is a subset of the Corel image database. The Wang database contains 10 classes of 100 images each. The 100 images in the each class
TABLE II

Characteristics of datasets used for evaluation PixSearcher and other CBIR algorithms.

<table>
<thead>
<tr>
<th>Datasets</th>
<th># Images</th>
<th># Queries</th>
<th>Size</th>
<th>Average relation</th>
<th>Query mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wang</td>
<td>1,000</td>
<td>1,000</td>
<td>384 x 256</td>
<td>99.0</td>
<td>leaving-one-out</td>
</tr>
<tr>
<td>UW</td>
<td>1,109</td>
<td>1,109</td>
<td>756 x 504</td>
<td>59.3</td>
<td>leaving-one-out</td>
</tr>
<tr>
<td>UCID</td>
<td>1,338</td>
<td>262</td>
<td>512 x 384</td>
<td>3.5</td>
<td>leaving-one-out</td>
</tr>
<tr>
<td>ZuBuD</td>
<td>1,005</td>
<td>115</td>
<td>640 x 480</td>
<td>5.0</td>
<td>train and test datasets</td>
</tr>
</tbody>
</table>

have been carefully chosen to make sure that images in the same class are visually similar and they are different from images belonging to other classes. The size of these images is 384 by 256. Figure 36 shows one sample image for each class. We use leave-one-out cross validation to evaluate the performance of different CBIR algorithms: given a query image (in 1,000 images), we use a CBIR algorithm to find the most similar one in 999 images; if the query image and retrieved image belong to the same class, we have a correct classification. We repeat this process for all 1,000 images in the dataset. The Wang database is publicly available.

The UW dataset: The UW image database was created by the University of Washington (71, 72). This database consists of a 1,109 images. The size of these images is mostly 756 by 504. These images are annotated using keywords such as “buildings”, “tree”, and “clear sky”. Some example images with annotations are depicted in Figure 37. We also use leave-one-out cross validation to evaluate the performance of different CBIR algorithms on this image dataset.

http://wang.ist.psu.edu/docs/related/
Figure 36. One example image from each of the 10 classes of the Wang database together with their class labels under each image.

An retrieved image is considered to be relevant to a given query image if the two images have a common keyword in the annotation. The UW database is freely available\footnote{http://www.cs.washington.edu/research/imagedatabase/groundtruth/}

**The UCID dataset:** The UCID database \cite{73, 74} contains 1,338 vacation images. The size of these images is 512 by 384. Some example images are given in Figure 38. The images that are assigned to be relevant if they are very clearly related, e.g. for an image showing
Figure 37. Six example images from the UW database with annotations under each image.

American flag, images showing the same flag taken at different angles/times are marked as similar images. The UCID database is publicly available.[1]

The ZuBud dataset: ZuBuD[75] stands for “Zurich Buildings Database for Image Based Recognition”. The database consists of training and testing set. The training set contains 1,005 images of 201 buildings (5 images of each building taken from different viewpoints/rotations). The testing set contains 115 images. [Figure 39] shows two example buildings. In particular, six images of the same building are on the left panel; six images of a different building are on the right panel. In each group, the top left image is the test image; other five images belong to the training dataset. A correct classification is obtained if the retrieved image is one of the images.

five relevant images of the same building of the query image. The ZuBuD database is publicly available.

### 6.3.2 Selection of number of quantizations

How many quantizations to apply on each color channel? Figure 40 shows the test results of PixSearcher with different quantizations on the 4 image datasets. In particular, error rate test results (in the top panel) indicate that PixSearcher does not perform well with fewer quantizations. As we increase the number of quantizations, the error rates get smaller. The significant improvement is obtained with 8 quantizations (the smallest average error rates over four datasets). The results get worse if we keep increasing the number of quantizations. This is because we do not have enough pixels in each quantized image to characterize different aspects of an image. Therefore, we use 8 quantizations for PixSearcher.

Figure 39. ZuBuD image dataset: The left panel displays a query image (top left) from testing dataset and 5 similar images (of the same building) from training dataset.

The computation times are graphed in the lower panel of Figure 40. Overall, PixSearcher takes less time to compute the images descriptors as we increase the number of quantizations. Here is the proof. The proximity graphs of the Delaunay triangulation (convex hull, alpha shape, and minimum spanning tree) requires roughly $O(n \log(n))$ to compute. We also know that the more quantizations means the fewer average pixels in each quantized image. Let $q$ be the number of quantizations and $n$ be the average number of quantized pixels ($n = 1,600/q$). The feature computation time is $q \times n \log(n)$. If we double the quantizations $2q$, the average number of quantized pixels is $n/2$. The feature computation time is $2q \times (n/2) \log(n/2) = q \times n \log(n/2) = q \times n \log(n) - q \times n \log 2$. This amount is less than $q \times n \log(n)$. 
Figure 40. Error rate (upper panel) and computation time (lower panel) of PixSearcher with different quantizations on the 4 image datasets.
6.3.3 Selection of rescaled resolution

What is the best resolution to rescale original images? Figure 41 shows the test results of PixSearcher with different resolutions on the 4 image datasets. As depicted, in the top panel, we can get higher accuracy by increasing the resolution of rescaled images (increasing the details of quantized images). The improvements become insignificant with more than 40 pixels in each dimension (1). These improvements require the much higher computation time. The test results in the lower panel indicate that our feature computation time grows quadratically with the rescaled image dimension (2). For these two reasons (1) and (2), we choose 40 by 40 resolution to rescale original images.

6.3.4 Comparisons of different CBIR algorithms

Figure 42 shows the error rates for each algorithm for each image dataset. In particular, datasets are displayed vertically. On each dataset, we use dotplots to present the error rates. The lower error rates are closer to the left; the higher error rates are closer to the right. Red dots represent the error rates of PixSearcher; Blue dots represent the error rates of color histogram; Black dots represent the error rates of other algorithms. Color histogram outperforms other conventional algorithms (18). Compared to color histogram, PixSearcher has lower error rates on 3 out of 4 image datasets (UW, UCID, and ZuBuD image datasets). On average of the 4 image datasets, PixSearcher has lower error rate (21.2%) than color histogram (22.1%). The error rates of CBIR algorithms (except PixSearcher) are regenerated from an experimental comparison in 2008 (18).
Figure 41. Error rate (upper panel) and computation time (lower panel) of PixSearcher with different resolutions on the 4 image datasets.
Figure 42. Dotplots display the error rates for 20 algorithms for 4 image datasets (Wang, UW, UCID, and ZuBuD). The last row is the average error rates of 4 image datasets.

Figure 43 shows the feature computation time for each algorithm for each image dataset. In particular, PixSearcher is in red, color histogram is in blue, other 4 algorithms (gray histogram, Tamura texture histogram, global texture feature, and local feature) are in black. PixSearch is shown to be the fastest algorithm in this group. These are the CBIR algorithms that we were able to configure to run on our machine. All tests were performed on a 2.3 GHz Intel Core i5, Mac OS X Version 10.7.5, 4 GB RAM running Java 1.6 and Processing 1.5.1.

“Color histogram, often cited as a baseline in CBIR, is a reasonably good baseline for general color photographs” [18]. Compared to color histogram, our approach gets lower average error
Figure 43. Feature computation times for 6 algorithms for 4 image datasets (Wang, UW, UCID, and ZuBud).

rates (see Figure 42) and faster (see Figure 43). Similar to color histogram, we quantize input images on a color space. However, there are some basic differences. First, we use HSB instead of RGB for color quantizations. Second, we quantize images on one color dimension at a time instead of the combination of three color dimensions. Last but the most important, we try to learn the distributions of quantized pixels instead of simply counting them. This raises another question: Why is PixSearcher faster than color histogram as we spend more time to compute the descriptors on quantized images? The answer lies within the preprocessing stage: Original images are rescale to 40 by 40 pixel arrays before quantizations and feature computations. We
rescale original images because PixSearcher’s accuracy does not rely much on the high quality of images (see the test results in Figure 41). In fact, we can save computation time by working on the lower quality version which preserves most of the features of the original image.

To support this observation, we conducted a runtime comparison test of PixSearcher and color histogram on images of different resolutions. The image resolutions are in the range from 1 megapixels to 8 megapixels. Remember that images taken by an iPhone 5 have the resolution of 3264 by 2448 (equivalent to 8 megapixels) and can take up to 2 megabytes computer memory. Figure 44 shows the test results. As depicted, color histogram runtime is linearly dependent on image size. Color histogram takes 4 seconds to process an image taken by iPhone 4 and almost 8 seconds to process an image taken by an iPhone 5. In contrast, PixSearcher is much faster, especially for high quality images. The difference of processing times between 1-megapixel and 8-megapixel images is the difference in the times to read and rescale the input images. The PixSearcher’s feature computation times on rescaled images are similar.

The fast processing time on high quality images is an advantage of our approach since image sizes are getting larger and larger. Starting from 2011, modern digital camera resolutions are 80 megapixels (10320 by 7752) (77). For mobile phone as of 2014, Nokia 808 PureView can take high resolution pictures up to 41 megapixels (7728 by 5368) (77).

6.3.5 Image Retrieval

In this section, we compare the accuracy of the our system to SIMPLIcity system using the same Corel database (24). This is a general-purpose image database including 60,000 pictures, which are stored in JPEG format with size 384 by 256 or 256 by 384. We show the comparison
results for each query example. Due to the limitation of space, we show only the top 29 matches to each query on five rows. Figure 46 and Figure 45 respectively show examples on one textured image and one non-textured image. The query images are displayed on the left of first rows (query image ids are shown underneath). We leave the judgments on accuracy for the readers.

### 6.4 Image Cluster

Displaying huge images database, such as 60,000 images in the Corel database, is limited by screen pixels and computational power. Therefore, our solution is grouping similar images into clusters and presenting each cluster by a single selected image called leader image. We use
Figure 45. Retrieving similar images to a textured image on two CBIR systems: SIMPLIcity system on the top, PixSearcher system at the bottom.
Figure 46. Retrieving similar images to a non-textured image on two CBIR systems: SIMPLIcity system on the top, PixSearcher system at the bottom.
the leader algorithm (12) to cluster images based on Euclidean distance in feature space. The complexity of this algorithm is \( O(n) \) (where \( n \) is the number of images).

After having clusters and their leader images, we now use the force-directed layout to place them on a 2D view. The advantages of force-directed layouts are intuitiveness, flexibility, and interactivity. The main disadvantage is high running time. Since for every leader image, we have to compute the attraction or repellant against all other leaders, the running time at each iteration is \( O(l^2) \) (where \( l \) is the number of leader images). However, \( l \) is much smaller than \( n \) and we can control the clustering algorithm so that it generates a reasonable numbers of leaders (a few hundreds of leader images). This running time is not remarkable.

In the force-directed layout, we first put all leader images randomly in the output panel and we then allow them to interact to find similar leader images based on their descriptors. Consequently, similar leaders are grouped together. This makes easier to interpret the clustering results. Here is the summary of the force-directed algorithm:

1. For each pair of leader images \( S_i \) and \( S_j \), we compute a dissimilarity measure using

   \[ \text{Equation 6.8} \]

2. We get dissimilarity cut \( C \) as a user input. We then define \( D_{ij} = \text{Dissimilarity}(S_i, S_j) - C \).

3. We compute \( \vec{U}_{ij} \) as the unit vector from \( S_i \) to \( S_j \).

4. If \( D_{ij} \leq 0 \), \( \vec{F}_{ij} \) is the attraction between \( S_i \) and \( S_j \) computed by the following equation:

   \[ \vec{F}_{ij} = D_{ij} \times \vec{U}_{ij} \quad (6.9) \]
5. If $D_{ij} > 0$, $\vec{F}_{ij}$ is the repulsion of $S_j$ on $S_i$.

$$\vec{F}_{ij} = \frac{D_{ij} \cdot \vec{U}_{ij}}{\text{Distance}(S_i, S_j)}$$  \hspace{1cm} (6.10)

6. The force applied on $S_i$ is the sum of forces by all leaders on $S_i$ (l is the number of leader images):

$$\vec{F}_i = \sum_{i=1}^{l} \vec{F}_{ij}$$  \hspace{1cm} (6.11)

7. Repeat steps 3-6 for all leaders $S_i$.

The algorithm can be stopped manually when users feel happy with the configuration or automatically when there is no more improvement (all similar leaders are close to each other). Notice that in Equation 6.9, the attraction between $S_i$ on $S_j$ does not depend on their distance. This assures that similar images can come close to each other no matter where they are in the display.

Figure 47 shows how we display the leader images of Corel database in the forced-directed layout. In particular, the size of each leader image is computed based on its cluster size (or the number of images in each cluster). We also use circles instead of rectangles to present images since circles are more suitable for force-directed layout. As depicted, leader images are further grouped into clusters. This provides a summary for Corel database.

In Figure 48(a), we align the leader images on X-axis based on their Hue. This alignment reveals the density distribution of leader images on Hue. Furthermore, we can request to align the leader images on Y-axis based on their Brightness as depicted in Figure 48(b).
directed layout makes transitions from \textbf{Figure 48(a)} to \textbf{Figure 48(b)} smoothly. We only have to tell the leaders where they should go, then the leaders fit themselves into the display area and avoid overlapping.

\section{6.5 Conclusions}

In this chapter, we propose a novel image retrieval technique extended from our previous works on scagnostics. This technique is designed based on the visual perception theory. Since humans are the end users of CBIR systems who are able to access the validity of the output
Figure 48. Corel database: (a) Leader images are aligned on X-axis based on their Hue  
(b) Leader images are aligned on both X-axis and Y-axis based on their Hue and Brightness.
image, we try to implement our approach as close as possible to the bottom-up visual process\textsuperscript{(57)}. Gestalt principles are considered as the pull factors to make this happen.

Our approach can be summarized in the following steps. We first rescale the query image into 40 by 40 pixel image. We then quantize the 40 by 40 pixel image on Hue, Saturation, and Brightness to produce binary pixel arrays which are considered as data points to compute image descriptors. Features are extracted based on the pixels distribution. Finally, Euclidean distance on feature space is used to compare the similarity of the query image to images in the training dataset.

To justify our selections of the number of quantization and rescaled resolution, we have conducted the tests on 4 benchmark image databases from different domains. The same datasets have been used to compare the performance of our approach against other CBIR algorithms. Our approach is proven to be more accurate and faster, especially on high quality images which are becoming more and more popular.

For the visualization part, we cluster images based the small set of features using the leader algorithm. Instead of showing entire image collection of thousands of images, we display only a few hundreds of leader images in a force-directed layout. This provides an overview of the large image collection. The force-directed layout makes smooth transitions between different configurations of leader images.
CHAPTER 7

CONCLUSIONS AND FUTURE WORK

As the term “big data” becomes more and more popular, we need faster approaches to handle a large amount of data. In this dissertation, I introduce a new technique for exploring and visualizing huge databases using visual features. These visual features aim to characterize the 2D distributions of orthogonal projections on a set of points in a scatterplot or a set of pixels in an image. To compare two scatterplots/images, we use a small set of features instead of inspecting millions of data points/pixels. The visual features were first introduced in 2005 (1). We reviewed these visual features, called scagnostics (34), in Chapter 3.

We first used scagnostics to handle doubly-multivariate time series in Chapter 4. The data model that our application, TimeSeer (2), is designed to deal with is: \( t \) time points and \( p \) variables, resulting in \( p \)-multivariate time series. For each variable, we have \( n \) series, resulting in a doubly-multivariate design. Working directly with the scagnostics measures, we located unusual correlations between pairs of variables in time series, which are not obvious by looking at the raw time series.

We then used the scagnostics measures to cluster scatterplots in Chapter 5. In particular, we applied a simple clustering algorithm to obtain clusters and their exemplar scatterplots. For large datasets of thousands of dimensions (we cannot use scatterplot matrix to display all pairwise projections), our application, ScagExplorer (3), displays only exemplars (about 30 of
them) in a force-directed layout where similar exemplars are further grouped together. This provides a comprehensive overview of a high-dimensional dataset.

In Chapter 6 we extended scagnostics to handle pixels in images and developed new descriptors (based on visual perception principles) that are appropriate for images. In particular, the image descriptors are computed on pixels obtained by filtering the original images at different levels on Hue, Saturation, and Brightness. Our system, PixSearcher, outperformed conventional techniques on speed (Figure 43) and accuracy (Figure 42). Moreover, our model works even on black and white images, which the color histogram approach cannot. In this case, our model depends on Brightness to compare the similarity of two images. Some examples of black and white image databases in the medical domain can be found at http://ganymed.imib.rwth-aachen.de/irma/datasets_en.php.

For future work, we are planning to evaluate our approach on much larger image databases, for example the CoPhIR project (78). The CoPhIR database contains 100 million images that have been specifically developed for testing on the scalability of image retrieval algorithms. The database can be retrieved from http://cophir.isti.cnr.it/index.html.

For the visualization part, we are planning to apply this technique in more dynamic environments such as clustering new images posted on Facebook/Twitter, or thumbnails of newly-posted videos on Youtube.


56. Peterson, M. and Salvagio, E.: Figure-ground perception. [Online; accessed 11-May-2014].


168. Salvagio, E.: Figure-ground perception. http://www.scholarpedia.org/article/File:FigureGroundPerceptionFigure2NewConvexitySmallAreaSymmetryEnclosure.jpg, 2010. [Online; accessed 11-May-2014].

169. Salvagio, E.: Figure-ground perception. http://www.scholarpedia.org/article/File:FigureGroundPerceptionFigure4WideBaseLowerRegionProtrusion.jpg, 2008. [Online; accessed 11-May-2014].

APPENDIX

IEEE POLICIES ON REUSE LICENSE

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you must follow the requirements listed below:

Textual Material

Using short quotes or referring to the work within these papers, users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line 2011 IEEE. In the case of illustrations or tabular material, we require that the copyright line [Year of original publication] IEEE appear prominently with each reprinted figure and/or table. If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior authors approval.

Full-Text Article

If you are using the entire IEEE copyright owned article, the following IEEE copyright/credit notice should be placed prominently in the references: [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication] Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line. In placing the thesis on the author’s university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity’s name goes here]’s products or services. Internal
or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

The IEEE policies on reuse license can be found on IEEE website at http://www.ieee.org/publications_standards/publications/rights/permissions_faq.pdf
VITA

Name
Tuan Nhon Dang

Research Interests
Developing methods and tools for visual analytics - an integrated approach combining visualization, human factors and data analysis to derive insight from massive, dynamic, and ambiguous data.

Education
2010-Present
University of Illinois at Chicago, IL
Phd in Computer Science
Cumulative G.P.A: 3.64/4.0

2008-2009
University of Illinois at Chicago, IL
M.Sc. in Computer Science
Cumulative G.P.A: 3.42/4.0

2006-2009
Politecnico di Milano, Italy
M.Sc. in Computer Engineering
Final grade: 106/110

2001-2006
University of Technology, Ho Chi Minh City, Vietnam
B.Sc. (with honor) in Computer Science
Cumulative G.P.A: 8.1/10

Work Experience
August 2011 - Present
Research assistant in the Computer Science Department at University of Illinois at Chicago, IL

May 2011 - August 2011
Research assistant in the Learning Science Department at University of Illinois at Chicago, IL
January 2009 - May 2011
Research assistant in the National Center for Data Mining at University of Illinois at Chicago, IL

June 2005 - August 2005
Engineering Internship at Paragon Solutions Vietnam, Ho Chi Minh City, Vietnam

Achievements and awards

2006:
Caripro Scholarship of Politecnico di Milano, Milan, Italy

2006:
Ministry of Post and Telematics and Motorola Vietnam awards to top 24 Computer Science students from Vietnam’s leading universities in recognition of their academic excellence

2005:
Honored Award of American Chamber of Commerce for top students, Ho Chi Minh City, Vietnam

2004:
The fourth Award of honored program of University of Technology, Ho Chi Minh City, Vietnam

Publications


**Computer Skills**

- Programming: Java, J2EE, J2ME, Ruby on Rails, MS Visual C++, C on Linux

- Database: Oracle 9i, MySQL, MS Access

- Operating Systems: Mac, Windows, Linux