DYNAMIC BASED CONTOUR CLUSTERING

A Thesis Presented

by

Xiao Zhang

to

The Department of Electrical and Computer Engineering

in partial fulfillment of the requirements

for the degree of

Master of Science

in

Electrical Engineering

in the field of

Communication and Signal Processing

Northeastern University
Boston, Massachusetts

August 2014
© Copyright 2014 by Xiao Zhang

All Rights Reserved
Abstract

Contour provides very fundamental and important information of objects from images, which is very useful in object detection, classification, recognition and retrieval. A wide range of computer vision tasks benefit from the improvement at contour detection and clustering. And contour clustering is a necessary and crucial step of contour based object recognition.

In this thesis, we present a novel approach of clustering contours based on comparing the dynamic distances between them. A significant portion of this work is inspired by the excellent performance of human activity recognition using this method. The main idea is to hypothesize contours extracted from images as the output trajectories of unknown linear dynamic systems with unknown initial conditions. To avoid the complex task of system identification, Hankel matrices are built to encapsulate the dynamic properties of contour trajectories in the feature space. Then we use dynamic based dissimilarity metric to compare Hankel matrices and calculate the dynamic distances between them. With a matrix consisting of dynamic distances of each possible pair of contours, Normalized Cuts is applied to classify contours into different clusters. In real application, contour trajectories are composed of a sequence of discrete pixels. Rank minimization is required to clean the data and reduce the rank of Hankel matrices. And contour trajectories are also needed to be chopped at corners into segments. The primary contribution of the thesis is proposing a robust dissimilarity metric combing the dissimilarity score function used in human activity recognition with the order information of dynamic systems, the rank of Hankel matrices. We also use cumulative angles as the feature of contours instead of velocities, the derivative of positions of contours.
Acknowledgements

First of all, I would like to express my deepest gratitude to my research advisor, Prof. Octavia Camps. Her guidance and support is of great importance to me. Her constant efforts, great patience and brilliant ideas have inspired me a lot and kept my research improving. I indeed benefited from Prof. Camps’ extensive knowledge and professional teaching of Computer Vision courses.

I would also express my great sense of gratitude to Prof. Mario Sznaier, who offered impressive theoretical support in the area of dynamic systems and mathematics, and Richard Moore, who read my thesis and gave me valuable suggestions.

Finally, I would like to thank these PhD students in the Robust System Lab, especially for Mengran Gou, Xikang Zhang and Yin Wang. They provided a number of beneficial suggestions about how to solve the problems in the experiments. Without their help, I could not complete this work.
# Table of Contents

Abstract \hfill II

Acknowledgements \hfill III

List of Figures \hfill VI

1 Introduction \hfill 1

1.1 Overview \hfill 1

1.2 Related Work \hfill 2

1.3 Thesis Organization \hfill 5

2 Contour Detection and Extraction \hfill 6

2.1 Evolution of Contour Detection \hfill 6

2.2 Contour Cut vs Structure Forest \hfill 7

3 Hankel Matrix and Dynamic Distance \hfill 11

3.1 Hankel Matrix \hfill 11

3.2 Dynamic Distances \hfill 14

3.3 Rank Minimization \hfill 15
4 Method of Contour Clustering

4.1 Features of Contour

4.1.1 Slope, Angle and Angular difference

4.1.2 Cumulative Angles

4.2 Corner Detection

4.2.1 Dynamic based Corner Detector

4.2.2 Cumulative Angle based Corner Detector

4.3 Clustering Methods

4.3.1 K-means

4.3.2 Normalized Cuts

5 Experiments and Analysis

5.1 Data Set

5.2 Measurement of Dynamic Distances

5.3 Clustering of the Synthetic Data

5.3.1 Clustering by dissimilarity score function

5.3.2 Clustering by dynamic based dissimilarity metric

5.3.3 The Influence of Noise on Clustering

5.4 Clustering of Contours Extracted from Images

6 Conclusions and Future work

Bibliography
List of Figures

Figure 2.1: Contour detection by Contour Cut and Structured Forest................................. 8

Figure 2.2: Contour extracted by Contour Cut and Structured Forest........................................... 9

Figure 3.1: An example of comparison between input and output of rank minimization ....16

Figure 3.2: The tangent angles extracted from data of contours before and after rank minimization ...................................................................................................................... 17

Figure 4.1: Flow chart of the dynamic based contour clustering ..............................................19

Figure 4.2: Contour trajectory........................................................................................................ 21

Figure 4.3: The MATLAB function atan2 ................................................................................... 21

Figure 4.4: Angular Function........................................................................................................ 22

Figure 4.5: The MATLAB function atan and map it into $[0, 2\pi]$ .............................................. 24

Figure 4.6: The angle and the cumulative angle ........................................................................... 25

Figure 4.7: Flow chart of dynamic based corner detection........................................................27

Figure 4.8: The outputs of dynamic based corner detection of the synthetic image .......... 28

Figure 4.9: The cumulative angles of contours in the synthetic image ................................. 30

Figure 4.10: The derivative of cumulative angles of contours in the synthetic image .......... 30
Figure 4.11: The output of cumulative angle based corner detector........................................ 31

Figure 5.1: Dynamic distances calculated by dissimilarity score function ..........................37

Figure 5.2: Dynamic distances among the circular arcs on the same circle ....................... 39

Figure 5.3: Synthetic data for contour trajectories ............................................................... 40

Figure 5.4: The output of contour clustering on the synthetic data .................................... 42

Figure 5.5: The utilization of order information in contour clustering ..................................43

Figure 5.6: Contour clustering using dynamic based dissimilarity metric .........................44

Figure 5.7: Clustering based on the dissimilarity score function and its distance matrix .... 45

Figure 5.8: Clustering based on dynamic based dissimilarity metric and the distance matrix ..........................47

Figure 5.9: Singular values of cumulative angles of ellipse and sinusoid ..........................48

Figure 5.10: The best result of contour clustering in the experiments...............................48

Figure 5.11: The influence of noise on contour clustering ............................................... 49

Figure 5.12: Clustering of contour segments by our approach .........................................50

Figure 5.13: Better clustering with straight lines detection firstly .................................... 51

Figure 5.14: Clustering of contour segments from BSDS500 .............................................52
Chapter 1

Introduction

1.1 Overview

Contour detection and analysis have attracted much attention in the field of computer vision and cognitive perception. Contour provides very fundamental and important information of objects from images, which is very useful in object detection, classification, recognition and retrieval. It can represent the shape information of object, an intrinsic feature for image understanding. It’s also robust to illumination and variation in object color and texture. A wide range of computer vision tasks benefit from the improvement at contour detection and clustering due to the key role of contour. To better recognize objects with different contours, we need to group contours into different clusters. So contour clustering is a necessary and crucial step of contour-based object classification and recognition.

In this thesis, we present a novel approach of contour clustering based on the theory of dynamic systems. Dynamic systems have been used in a variety of computer vision applications, including texture recognition, target tracking and activity recognition. A significant portion of this work is inspired by the excellent performance of human activity recognition using this method [1]. The main idea is to hypothesize contours extracted from images as the output
trajectories of unknown linear dynamic systems with unknown initial conditions. That means we transform contours into time-series signals. To avoid the prior knowledge of dynamics and the complex task of system identification, Hankel matrices are built to capture the dynamic properties of systems corresponding to contour trajectories in the feature space. Then we use dynamic based dissimilarity metric to compare the Hankel matrices normalized by Frobenius norm and calculate the dynamic distances between these contours. With a distance matrix consisting of dynamic distances of each possible pair of contours, the Normalized Cuts method is applied to classify contours into different clusters. The primary contribution of the thesis is proposing a dynamic based dissimilarity metric combining the dissimilarity score function introduced by Li et al. [1] with the order information of dynamic system, the rank of Hankel matrices. We also use the cumulative angles as the feature of contours instead of the velocities, the derivative of positions of contours. Experiments demonstrate that cumulative angles are more robust than velocities for dynamic based contour clustering.

1.2 Related Work

Contour clustering plays important role in object recognition. And it’s most related to shape clustering. Finding good descriptors in the feature space and dissimilarity measurements of contours are the central issues in these applications. Shape context is a very popular descriptor, presented by Belongie et al. [2]. It captures the shape of an object by a finite set of points on contour of the object. A histogram is defined to be the shape context of objects in log-polar space. It describes the distribution of positions of these points with respect to a given referent point on the shape. The log-polar space makes the descriptor more sensitive to positions of nearby sample points on contour than to those of points farther away. Shape context enable us to solve the problem of finding correspondences between points on any two shapes. Given the correspondences of points, a transformation is estimated to best align the two shapes.
Accordingly, the dissimilarity between two shapes is calculated as a sum of matching errors between corresponding points, plus with a term measuring the magnitude of the aligning transformation. In the framework of nearest-neighbor classification, shape contexts are used to match and recognize objects.

Daliri et al. [3] utilized the tangential vectors along the contours to compute the curvature, one of the most important features to describe a contour of shape. Different curve parts can be distinguished according to the curvature. For instance, the values of straight lines are close to zero and the sharp angles or corners have high values. Then the curvatures of contour segments have been transformed into a symbolic representation by using a predefined dictionary based on the value, sign and linkage of the curvatures. From the symbolic representation, an invariant high-dimensional feature space is created. And the most relevant lower dimensions of feature space are extracted by principal component analysis (PCA). Last, support vector machine (SVM) is employed to classify contour segments extracted from silhouettes in the feature space.

In contour and shape clustering, we can also describe contour in the frequency domain. The most common descriptors in the frequency domain are Fourier descriptors, wavelet descriptors and wavelet-Fourier descriptors. Contour representation using Fourier descriptors are easy to compute by Fourier transform. They are usually obtained from one dimensional function derived from contours, including centroid distance, position function, curvature and cumulative angles. The lower frequencies contain general features of contours and the higher frequencies involve more details of contours. Direkoğlu et al. [4] introduced a multiscale Fourier-based shape descriptor by applying a low-pass Gaussian filter (LPGF) and a high-pass Gaussian filter (HPGF) separately in 2-D space. Actually, this is a region-based shape descriptor rather than a contour-based shape descriptor. The output of LPGF represents the inner and central part of object. On the other hand, the output of HPGF represents the exterior and contour part of object. The combination of different outputs of the filters at various scales can increase discrimination power and the accuracy of classification.
Yankov et al. [5] demonstrated the ability to attract similar shapes together in the nonlinear projection algorithm such as Isomap. This suggests that the data of shape are isometric to some nonlinear embedding of the original shape space. Therefore, it can group shapes by using a relative low dimensional representation. Besides, the skeleton structure of shape is also an important characteristic in shape clustering and classification [6] [7]. Bai et al. [8] combined both information of contour and skeleton locally and globally for shape analysis and derive an effective classifier on the shapes with large intra-class variation and inter-class similarity.

In general, a perfect complete contour is very hard to detect in the real images. The contour segments extracted from images probably contain the meaningless edges due to the texture and noise. We can group contour segments belonging to the same object by clustering them with multi-feature similarity measurement [9]. The variance for gray value besides contour segments in a certain size of square area is computed and combined with the features of contours themselves. The multi-feature grouping cue is more reliable and robust compared with single-feature and it can improve the performance of contour detection and image segmentations.

Furthermore, contour clustering has brought advantages to various industrial applications. Govindaraju et al. [10] defined a log-arithmetic distance between two contours in the feature space. The agglomerative clustering algorithm is exploited to group contours extracted from postal parcel images and to locate the postal address blocks. This work can help increasing efficiency of postal service and intelligent transportation. J. Zhang et al. [11] extracted the explicit topological relationships of contours of buildings from LIDAR data, which represent the structure information of buildings. Then contour clustering is based on the topological relationship and similarity analysis. Using clusters of contours, the detailed structure of buildings are detected to model and reconstruct them. This technique provides a sound foundation for reconstructing buildings with multiple layers and complex shapes.
1.3 Thesis Organization

The thesis will focus on the contour clustering algorithm by using dynamic based dissimilarity metric in the feature space of cumulative angles. Chapter 2 introduces the development of contour detection algorithms and makes a comparison between Contour Cut method and Structured Forest edge detector. And it shows how to extract contours before contour clustering in our approach by Structured Forest, a more accurate and faster contour detector. Chapter 3 expounds the theory of dynamic systems and Hankel matrices. Our dynamic based dissimilarity metric is proposed in this chapter. Additionally, a method of rank minimization is introduced to clean the contour trajectories and reduce the rank of Hankel matrices. Chapter 4 demonstrates the main procedure of our contour clustering algorithm. We concentrate on the cumulative angles in the feature space and propose two corner detection techniques: dynamic based and cumulative angle based corner detectors. The clustering method of Normalized Cuts is compared with K-means. Chapter 5 characterizes the experiments of measuring dynamic distances between contours and clustering contours based on our dynamic based dissimilarity metric. Our approach has been applied on three kinds of data, including synthetic data and contour trajectories extracted from both synthetic images and natural images. And Chapter 6 summarizes the dynamic based contour clustering algorithm and discusses its application in the area of object recognition in the future work.
Chapter 2

Contour Detection and Extraction

2.1  Evolution of Contour Detection

Contour detection has been a fundamental problem in computer vision for a long time. It’s related to edge detection but with higher level information to infer the boundary of objects. Early approaches concentrate on the respond of sharp discontinuities in image brightness by convolving grayscale images with local derivative masks, including Roberts, Sobel and Prewitt edge detection operators. The most popular edge detection technique, Canny detector, also detect sharp discontinuities of brightness. To get thinner and more complete boundaries, non-maximum suppression and hysteresis threshold are added in the Canny detector.

More recent approaches take account of brightness, color and texture multi-channel information and extend them in globalization to get better results. Martin et al. [12] measured the difference of all three channels by defining gradient operators in local image. With a large data set of human-labeled contour ground truth in natural images, the combination of the three channels has been converted to a supervised learning problem. And the posterior probabilities of boundaries (Pb detector) at image pixels are predicted by using these measurements as input of a logistic regression classifier.
Arbeláez et al. [13] extended these measurements in multiscale levels and combined all levels in the globalization framework as the final globalized probability of boundary (gPb detector). The globalization machinery strengthens the Pb detector by a reduction of clutter edges and incomplete contours in the output. It also improves the performance of contour detection based on the Precision-Recall curve on the Berkeley Segmentation Dataset benchmark. Ren et al. [14] made a further improvement on gPb detector by computing the sparse code gradients with the Orthogonal Matching Pursuit (OMP) algorithm and K-VSD algorithm. K-VSD is a dictionary learning algorithm derived from generalizing K-means method to learn codewords from unsupervised data. Experiments show that sparse code gradients can effectively measure local contrast and find contours in natural images.

2.2 Contour Cut vs Structure Forest

In the thesis, good contour detection is an essential preprocessing procedure of contour clustering. We make a contrast between two of the state-of-the-art contour detection methods: Contour Cut algorithm and Structured Forest edge detector. Contour Cut was proposed to detect the salient contour in images by Kennedy et al. [15]. This algorithm transforms contour detection into the problem of searching for cycles in directed weighted graphs. It defines each image edge as a graph node and connects them with weighted graph edges. And the weights are given by directed collinearity energy function which calculates the relative angles of image edges. So image edges with similar angles will be strongly connected and form cycles in graph for both closed and open contours. Then graph circulation is used for ensuring a natural random-walk representation of the contour cut cost function to detect cycles in graph. Kennedy proved that this method could be solved by calculating a family of Hermitian eigenvalue problem.

Dollár et al. [16] introduced a generalized structured learning approach called Structured Forest edge detector, using the labels in structured information of edge patches to train random decision
forests which map structured labels into a discrete space. Then the random forest framework with the captured structured labels predicts the probability of the final edge map in the detection process in real-time, which is huge progress in efficiency and faster than other competing state-of-the-art methods.

The results of contour detection by the two methods respectively are illustrated in Figure 2.1. Subfigure (a) is the original image from the data set BSDS500; (b) is the output of Contour Cut method; (c) is the probability map of boundary obtained by Structured Forest edge detector; (d) is the outcome of (c) after binaryzation with a threshold.

Figure 2.1: Contour detection by Contour Cut and Structured Forest
After contour detection, we extract contours from top-left to bottom-right in these binary boundary images. In contour extraction, we treat each contour in binary images as a trajectory. Each trajectory is formed among contours following clockwise direction. The trajectories with length shorter than 15 pixels, regarded as some noise or meaningless boundaries, should be removed. And we also break the trajectories at crossing so that the extracted contours look like clean in Figure 2.2.

**Figure 2.2:** Contour extracted by Contour Cut and Structured Forest
In Figure 2.2, the contours in left column subfigures are detected and extracted by Contour Cut; the right column subfigures are extracted by Structured Forest. According to these contours, Structured Forest can produce more accurate contours whereas contours given by Contour Cut are more detailed and smoother. It takes less than 0.3 second for Structured Forests to detect contour in an image. However, it will spend about two minutes for Contour Cut to do the same work because of the computational cost of Pb detector and graph cut. Considering the performance and remarkable efficiency of Structured Forest, we choose it as our contour detection and extraction method for natural images in BSDS500.

For synthetic images produced by computers, it’s easy to detect complete closed contours by Canny edge detector since a majority of synthetic images belongs to clean background and the boundaries of objects are obviously perceptible. Nevertheless, it’s hard for Structured Forest to extract complete contours from synthetic images. The corners normally exist on the contours of objects in the synthetic images, such as triangles, squares, and hexagons. The sharp discontinuities on the boundary of an object separate the contour extracted by Structured Forest into segments. Therefore, it’s better to choose Canny edge detector as contour detection approach for synthetic images.
Chapter 3

Hankel Matrix and Dynamic Distance

Recently, dynamic systems play an important and powerful role in widespread computer vision tasks, including object tracking, human activity recognition and dynamic texture recognition. We can transform the problems into analyzing and predicting the temporal evolution of the data which is a measurement vector $y_k \in \mathbb{R}^n$ as a function of state vector $x_k \in \mathbb{R}^d$ with relatively low dimension and changing over time in dynamic systems. In this thesis, we assume contours extracted from images as the output trajectories of unknown linear dynamic systems with unknown initial conditions. Then Hankel matrices are built to capture the dynamic information and to calculate the dynamic distances between contour trajectories for clustering.

3.1 Hankel Matrix

For contour clustering, we model each contour trajectory as the output of a linear time invariant (LTI) system, which is the simplest dynamic model. Given a sequence of the measurement vector $y_k \in \mathbb{R}^n$ and a relatively low dimensional state vector $x_k \in \mathbb{R}^d$, the form of LTI system is made as followed:
\[ y_k = Cx_k + w_k \]
\[ x_k = Ax_{k-1}, \quad x_0 \text{ given} \]  

(3.1.1)

where both equations of \( y_k \) and \( x_k \) are linear, the matrices \( C \) and \( A \) are constant over time and where \( w_k \sim N(0, Q) \) is uncorrelated zero mean Gaussian measurement noise. The dimension of the state vector \( d \) represents the order of the system and it measures the complexity of the system [17].

Unfortunately, there is a non-negligible limitation for the model in the practical computer vision applications. We cannot avoid estimating the dimensions and values of the triples \((A, C, x_0)\) which are not unique given a finite number of measurements \( y_k \). This limitation leads to a time-consuming computational problem. To avoid the intricate task of system identification, we build the Hankel matrices to capture the dynamic information of the systems instead of estimating their parameters.

Hankel matrix is an upside-down Toeplitz matrix. Given a sequence of output measurements \( y_0, y_1, \ldots, y_{rs} \), the Hankel matrix \( H_{y}^{s, r} \) is:

\[
H_{y}^{s, r} = \begin{bmatrix}
y_0 & y_1 & y_2 & \cdots & y_r \\
y_1 & y_2 & y_3 & \cdots & y_{r+1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
y_s & y_{s+1} & y_{s+2} & \cdots & y_{r+s}
\end{bmatrix}
\]

(3.1.2)

where the columns of \( H_{y}^{s, r} \) match the overlapping subsequences of \( y_k \), shifted by one, and the anti-diagonals of \( H_{y}^{s, r} \) are constant as defined in the formula 3.1.2.

In addition, the order information \( n \) of dynamic systems can also be acquired by computing the rank of Hankel matrices. Singular value decomposition (SVD) is applied on Hankel matrices to get the singular values of \( H_{y}^{s, r} \) \( (H = U \Sigma V^*) \). \( \Sigma \) is a \( s \times r \) rectangular diagonal matrix with singular values on it. Then we extract the dominant singular values by principal component analysis to measure the value of \( n \).
We sum the normalized singular values in accordance with the descending order until the following inequality is satisfied.

\[
\sum_{i=1}^{\min(s, r)} \frac{\Sigma_{i,i}}{\min(s, r)} \geq t
\]  

(3.1.3)

where \( n \) is the rank of \( H_{s,r}^{s,r} \), which is also the order of the corresponding dynamic system.

And \( t \) is the threshold between 0 and 1, for example, \( t = 0.95 \).

In practice, the output measurements are a sequence of features extracted from contour trajectories, including positions, velocities and angular information. And the Hankel matrix containing these features is rewritten as:

\[
H_{(x,y)}^{s,r} = \begin{bmatrix}
H_{s}^{s,r} \\
H_{y}^{s,r}
\end{bmatrix}
\]  

(3.1.4)

\[
H_{\theta}^{s,r} = \begin{bmatrix}
\theta_0 & \theta_1 & \theta_2 & \ldots & \theta_r \\
\theta_1 & \theta_2 & \theta_3 & \ldots & \theta_{r+1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\theta_s & \theta_{s+1} & \theta_{s+2} & \ldots & \theta_{r+s}
\end{bmatrix}
\]  

(3.1.5)

where \( H_{(x,y)}^{s,r} \) and \( H_{\theta}^{s,r} \) respectively denote the Hankel matrix with positions in \((x, y)\) coordinates and angular information.
3.2 Dynamic Distances

According to the dynamic subspaces angles (DSA) theory proved in [17], the principal angles between the subspaces spanning the columns of a Hankel matrix are zero, which means that the columns of Hankel matrix correspond to the output trajectories of the same dynamic system with different initial conditions. So we can compare the dynamic distance between two contour samples by calculating the principal angles between columns of the two Hankel Matrices involving features from respective contours. Unfortunately, the approach is easy to be corrupted by noise and thus requires more precisely estimation of these subspaces. To avoid this problem, Li et al. developed a dissimilarity score function to compare Hankel matrices in a smart way [1].

For two contour trajectories \( p \) and \( q \), we use the Frobenius norm to normalize two corresponding Hankel matrices \( H_p \) and \( H_q \):

\[
\hat{H}_p = \frac{H_p}{\|H_p H_p^T\|_F^{1/2}}, \quad \hat{H}_q = \frac{H_q}{\|H_q H_q^T\|_F^{1/2}}
\]  

(3.2.1)

Then the dissimilarity score function is applied to calculate the dynamic distance between them:

\[
d\left(\hat{H}_p, \hat{H}_q\right) = 2 - \left\|\hat{H}_p \hat{H}_p^T + \hat{H}_q \hat{H}_q^T\right\|_F
\]  

(3.2.2)

where \( d \geq 0 \) since Hankel matrix is normalized. If \( d\left(\hat{H}_p, \hat{H}_q\right) = 0 \), the two corresponding Hankel matrices belong to the same dynamic system. The function decreases the effect of noise in data with a computationally efficient way.

To improve the performance of contour clustering, we propose a novel dynamic based dissimilarity metric combing the dissimilarity score function with the order information of dynamic systems. When calculating the dynamic distance between two contours, we firstly compare their rank of Hankel matrices. If their ranks are equal, we still use the dissimilarity score function. If their ranks are not equal, we add a relatively large value on this function.
Then the dissimilarity metric function is written as:

\[
d(H_p, H_q) = 2 - \left\| \hat{H}_p^T \hat{H}_p + \hat{H}_q^T \hat{H}_q \right\|_F
\]

\[
\hat{d}(\hat{H}_p, \hat{H}_q) = \begin{cases} 
    d(H_p, H_q) & \text{if order}(p) = \text{order}(q) \\
    1 + \lambda d(H_p, H_q) & \text{if order}(p) \neq \text{order}(q)
\end{cases} \quad (3.2.3)
\]

where normally \( \lambda = 0 \) or 1 in the experiment.

The function sets the dynamic distance between the contour \( p \) and \( q \) to be a larger number when the orders of \( p \) and \( q \) are not equivalent. The larger value of distance helps us to better discriminate these contours with different orders.

### 3.3 Rank Minimization

Note that generally two contour trajectories \( p \) and \( q \) are corrupted by severe noise owing to the discretization of sampling contours by pixels. The effect of discretization results in full rank of the corresponding Hankel matrices \( H_p \) and \( H_q \). Hence the angle between the subspaces of the two noisy Hankel matrices is zero no matter whether the contours \( p \) and \( q \) correspond to different dynamic system. In this scenario, we cannot group contours into clusters precisely. To overcome the problem, a method of rank minimization is utilized to clean the data of contour trajectories and reduce the rank of Hankel matrices.

The rank minimization problem (RMP) arises in diverse areas of engineering applications, from statistics to control theory. It can be expressed as a problem of selecting the simplest model among the set of feasible models with convex constraints. For example, the features of contour can be embedded in a relatively low dimensional space. However, the rank minimization problems are known to be NP-hard and computationally intractable. One of the best solutions to RMP is characterized by M. Ayazoglu et al. [18].
The general RMP can be defined as:

\[
\begin{align*}
\text{minimize} & \quad \text{Rank}(X) \\
\text{subject to} & \quad X \in C
\end{align*}
\]  

(3.3.1)

where \(X \in \mathbb{R}^{m \times n}\) is the optimization matrix and \(C\) is the convex set of constraints [19] [20]. In our work, this problem can be translated to minimizing the nuclear norm of a Hankel matrix, subject to appended structural and sparse constraints on its elements. And the problem is solved by a fast algorithm for structured robust principal component analysis (SRPCA) based on the statements in [18]. Figure 3.1 shows an example of this algorithm on a simplistic curve sampled by pixels. In the light of the output, cleaning the curve by the algorithm looks like a curve fitting problem based on rank minimization. It smooths the curve at the corners and reduces the order of dynamic system related to the curve.

**Figure 3.1:** An example of comparison between input and output of rank minimization

To exhibit the importance and necessity of rank minimization in the practical feature extraction of contours, we plot the extracted features both before and after rank minimization. In Figure 3.2, the top row is the input contour data of rank minimization; the middle row is the tangent angles extracted from original discrete data; the bottom row is the tangent angles extracted from the
output data after rank minimization. Considering circles in the first column, there are a lot of noises on the output angles calculated by discrete pixels, leading to huge errors in computing the dynamic distances. After cleaning the circle by rank minimization, we can almost remove the noises and get more accurate values of angles. However, the discontinuities of angles makes the values oscillating between $-\pi$ and $\pi$. It causes much trouble to obtain precise tangent angles for bottom sides of triangle and square. The problem can be overcome by using cumulative angles as features assembled in the Hankel matrices, which will be described in next chapter. And the cumulative angles, cleaned by Ayazoglu’s approach of rank minimization, are robust and reliable features for contour clustering and corner detection.

**Figure 3.2:** The tangent angles extracted from data of contours before and after rank minimization
Chapter 4

Method of Contour Clustering

For an input image, we firstly detect and extract contours as the cell array of trajectories. For natural images, we use Structured Forest edge detector. For synthetic images, we use Canny edge detector. To reduce the effect of discretization in image pixels, the rank minimization method of structured robust principal component analysis (SRPCA) is adopted. After rank minimization, the contour trajectories become much cleaner and the orders of them have been reduced. A cumulative angle based corner detection technique finds the corners on the contour trajectories. Afterwards, these trajectories are chopped at corners into segments. We calculate the cumulative angles of each contour segment as the feature assembled into the Hankel matrix to encapsulate the dynamic information of the corresponding system. With the specialty of Hankel matrix, the dynamic distances between contour segments are computed by the dynamic based dissimilarity metric introduced in Chapter 3. Then a similarity matrix is derived from the logarithm transformation of the distance matrix composed of the dynamic distance between every possible pair of contour segments. Lastly, we treat the similarity matrix as the input of Normalized Cuts method to cluster contour segments into different groups. Figure 4.1 gives the main procedure of our dynamic based contour clustering algorithm presented in the thesis.
The input image

Contour detection and extraction

Rank Minimization to decrease the effect of discretization

Chop contour trajectories at corners into segments

Build Hankel Matrices assembling features of contour segments

Calculate dynamic distances between contours using dynamic based dissimilarity metric

Contour segments clustering based on Normalized Cuts

Natural Images
Structured Forests

Synthetic Images
Canny detector

Structured robust principal component analysis (SRPCA)

Cumulative angle based corner detection technique

γ(s): cumulative angles
(x, y): coordinates (positions)
(Δx, Δy): derivatives (velocities)
Δy/Δx: tangent slopes
θ: tangent angles
Δθ: differences of angles

Combining the order information of system with dissimilarity score function

Figure 4.1: Flow chart of the dynamic based contour clustering
4.1 Features of Contour

After contour detection and extraction, we get the cell array of contour trajectories. Each contour trajectory is in the form of [row, column]:

\[ C_{xy} = \begin{bmatrix} y_1 & x_1 \\ y_2 & x_2 \\ \vdots & \vdots \\ y_L & x_L \end{bmatrix} \]  \hspace{1cm} (4.1.1)

where \( y_k \) and \( x_k \) represent the index of row and column of image matrix respectively. \( L \) is the length of each contour trajectory. Then we can use the features of contours to build the Hankel matrices to capture the dynamic properties of them.

Clearly, the basic feature of contours in the form of (4.1.1) is the positions, the Cartesian coordinates \((x, y)\). Before building the Hankel matrices, we must remove the mean of each contour to counteract the effect of translation:

\[
\begin{align*}
    x &= \frac{1}{L} \sum_{k=1}^{L} x_k, \\
    y &= \frac{1}{L} \sum_{k=1}^{L} y_k \\
    x_k &= x_k - \bar{x}, \\
    y_k &= y_k - \bar{y}
\end{align*} \hspace{1cm} (4.1.2)

And we can also get the normalized gradients or derivatives \((\Delta x, \Delta y)\) from positions:

\[
\Delta \hat{p}_k = (\Delta \hat{x}_k, \Delta \hat{y}_k) = \frac{\Delta p_k}{\sum_{j=1}^{L-1} \| \Delta p_k \|} \hspace{1cm} (4.1.3)
\]

\[ \Delta p_k = (\Delta x_k, \Delta y_k), \quad \Delta x_k = x_{k+1} - x_k, \quad \Delta y_k = y_{k+1} - y_k \]

The derivatives of contour trajectories in 2D image are similar to the velocities of human activity trajectories between frames in the video data. Besides, the angular information is also a set of important characteristics for the shape of contours.
4.1.1 Slope, Angle and Angular difference

![Figure 4.2: Contour trajectory](image)

For the contour trajectory in the Figure 4.2, we firstly compute the derivative \((\Delta x, \Delta y)\). For each point \((x_k, y_k)\) on the trajectory except the initial and end point, we have \((\Delta x_k, \Delta y_k)\):

\[
\Delta x_k = x_{k+1} - x_{k-1}, \quad \Delta y_k = y_{k+1} - y_{k-1}
\]

Then the ratio \(\Delta y_k / \Delta x_k\) represents the tangent slope at the point \((x_k, y_k)\). And the arctangent of the ratio \(\Delta y_k / \Delta x_k\) represents the tangent angle \(\theta_k\). The derivative \(\Delta \theta_k\) is called angular difference.

\[
\theta_k = \arctan2(\Delta y_k, \Delta x_k), \quad \theta_k \sim [-\pi, \pi], \quad \Delta \theta_k = \theta_{k+1} - \theta_k
\]

![Figure 4.3: The MATLAB function atan2](image)
Unfortunately, all of the slope, angle and angular difference have the discontinuities in the value, which can highly increase the rank of Hankel matrix and lead to significant errors of dynamic distances and contour clustering. If we use them as the features of contours, the results of clustering are not predictable and reliable. So the cumulative angle is introduced in next subsection to solve this problem.

### 4.1.2 Cumulative Angles

The cumulative angle at a point is defined as the amount of angular change from the starting point. So it represents the summation of the angular difference (the derivative of angles) to each point [21]. To get the cumulative angle, the angular function $\varphi(s)$ is defined and measures the tangential angular direction as a function of arc length, which was used to obtain the set of Fourier descriptors. Figure 4.4 illustrates the angular function at a point on a closed contour. However, there is an undesirable property that the angular function has discontinuities when it increases to more than $2\pi$ or decreases to less than zero since it’s bounded from zero to $2\pi$.

![Angular Function](image)

**Figure 4.4:** Angular Function
This problem is eliminated by computing the summation of angular change for each point as the cumulative angles. We assume the angular difference corresponds the curvature $\kappa(s)$. And the cumulative angle is defined as

$$\gamma(s) = \int_0^s \kappa(r) \, dr - \kappa(0)$$

where the parameter $s$ takes values from zero to $L$ (the length of the contour trajectory). For a closed contour, the initial and final values of the function are $\gamma(0) = 0$ and $\gamma(L) = 2\pi$.

The cumulative angle avoids the discontinuity of angle. But it still has two problems. First, it has a discontinuity at the end. Second, its value depends on the length of contour. These problems can be solved by defining the normalized function:

$$\gamma^*(t) = \gamma\left(\frac{L}{2\pi} t\right) + t$$

where $t$ takes values from 0 to $2\pi$. It makes that $\gamma^*(0) = \gamma^*(2\pi) = 0$ only for closed contours.

Here is how we calculate the cumulative angle in the MATLAB. We used the function atan instead of atan2.

$$\theta_k = \text{atan}\left(\frac{\Delta y_k}{\Delta x_k}\right), \quad \theta_k \sim [-\pi/2, \pi/2]$$

And we map the $\theta_k$ into the interval $[0, 2\pi]$ in the Figure 4.5. But the $\theta_k$ still have the discontinuity in $[0, 2\pi]$. Then we use the following formula to calculate the cumulative angle.

$$\gamma_k = \begin{cases} 
\gamma_{k-1} - (\theta_k - \theta_{k-1}), & \text{if } |\theta_k - \theta_{k-1}| \leq \pi \\
\gamma_{k-1} - (\theta_k - \theta_{k-1} + 2\pi), & \text{if } \theta_k - \theta_{k-1} < -\pi \quad \text{with initial condition } \gamma_1 = 0 \\
\gamma_{k-1} - (\theta_k - \theta_{k-1} - 2\pi), & \text{if } \theta_k - \theta_{k-1} > \pi
\end{cases}$$
Figure 4.5: The MATLAB function atan and map it into $[0, 2\pi]$.

Figure 4.6 shows the output angle, cumulative angle and its normalized value of line, circle, ellipse and sinusoids. It also demonstrates the singular values of Hankel matrices with the feature of the cumulative angle and the normalized cumulative angle. In the first row subfigures, contours are sampled by setting any two neighboring points on curve with equal distance. We call it the equally interval sampling. And the longer contour comes with more points. The fourth row and sixth row display the singular value of the cumulative angle and the normalize one respectively. For the straight line, the angle is a constant value, so its cumulative angle is zero and its rank is just one. For the circle, its cumulative angle is a straight line with slope and comes with the rank two though the second singular value is very small. According to the singular values, for the unnormalized and normalized cumulative angle, we can hardly find out which one is a better feature since they are similar visually. However, in the experiments, the cumulative angle makes better clustering than the normalized value. For the normalized cumulative angle, a linear incremental component has been added on the value of cumulative angle for the open contours, which can change their dynamics and interfere with contour clustering. So we suppose that the cumulative angle is a more suitable feature for contour clustering.
Figure 4.6: The angle and the cumulative angle
4.2 Corner Detection

In the theory of dynamic system, corners never exist in the output trajectories. Therefore, the part of contour at corners cannot be modeled as the output of dynamic systems. We need to chop contour trajectories at corners into segments. So a good corner detector is essential for our task. The precision of the common Harris corner detector [22] can’t satisfy the requirement of our method. In the thesis, we propose two approaches for corner detection: dynamic based corner detector and cumulative angle based corner detector.

4.2.1 Dynamic based Corner Detector

Dynamic based corner detector uses the change on the ranks of Hankel matrices to find out where corners lay on the contour trajectories. The corners have high order information and increase the rank of Hankel matrix. Firstly, we build a $15 \times 15$ Hankel matrix assembling the features of each contour trajectory from the starting point and calculate its rank as the initial order. The $15 \times 15$ Hankel matrix involves the features of first 29 points on the contour. Then the feature of the 30th point is added into the $15 \times 16$ Hankel matrix. We add the features of next points one by one into the Hankel matrix until its rank is increased. That’s the position where a corner occurs. After a corner is detected, we repeat the above steps from the points behind the corner to detect other corners. The flow chart of this corner detector is displayed in Figure 4.7.

According to the positions of the output corners in Figure 4.8, there is a little shift about three or five pixels on the real position of every corner. It results from the delay of the influence of high order information in the output of dynamic system. We also find out that it’s difficult to select an appropriate threshold in the formula (3.1.3) to compute the ranks and detect corners precisely for all of contours in the synthetic image, even though we clean the data by rank minimization at first.
The input contour trajectory

Build a n x n Hankel matrix from starting point and calculate its rank

Does it arrive at the end of data?

Add the feature of next point into the Hankel matrix

Is its rank increased?

Corner occurs and the position of the corner is recorded

The output corners

Build a n x n Hankel matrix from the point behind the detected corner and calculate its rank

Figure 4.7: Flow chart of dynamic based corner detection
Figure 4.8: The outputs of dynamic based corner detection of the synthetic image
4.2.2 Cumulative Angle based Corner Detector

To get the positions of corners more precisely, we propose another corner detection technique, called cumulative angle based corner detector. For the contours with unique indexes in Figure 4.8, we plot the cumulative angles of them in Figure 4.9, except the index 10 and 11. The corners produce salient discontinuities on the output curve of cumulative angles. Circles and ellipses don’t have corners on the contours and produce smooth output without discontinuities. Thus we suppose that the corners can be discerned by finding out where the discontinuities happen on the output of cumulative angles.

For computational convenience, we take the derivative of cumulative angles and discover the impulse responses. These impulses are detected by finding the local extrema on the derivatives. It’s easy to realize the method in Figure 4.10. Those red stars denote the position of impulses on the derivative of cumulative angles, and they also indicate the positions of corners on the contours. Based on the derivatives plot of ellipse in the ninth subfigure, the thinner ellipse with larger ratio of major / minor axis will probably produce obvious local extrema and impulse responses. A suitable threshold is required to distinguish the points near the major axis of ellipse from the real corners.

In the top subfigure of Figure 4.11, all of corners are successfully detected by this approach. The little circle on each contour represents its starting point. The corners at the starting points are discarded since the trajectories of contour are not exactly closed and completed. In the bottom subfigure, the contours are chopped at these corners into contour segment. For a triangle, it is composed of three contour segments, three straight lines. Owing to non-existence of corners on circles and ellipses, each of them has only one contour segment, the whole contour trajectory of itself. Consequently, we take advantage of the cumulative angle based corner detector to remove the corners on the contours.
Figure 4.9: The cumulative angles of contours in the synthetic image

Figure 4.10: The derivative of cumulative angles of contours in the synthetic image
Figure 4.11: The output of cumulative angle based corner detector
4.3 Clustering Methods

Clustering is an unsupervised learning method which groups a set of data into different clusters. With a measurement of distance metric, its objective is maximizing the inter-class dissimilarity and minimizing the intra-class dissimilarity. In the thesis, we treat each contour as a point in the feature space and group them according to the dynamic distances. We focus on two methods of clustering: K-means and Normalized Cuts.

4.3.1 K-means

K-means is one of the simplistic unsupervised learning algorithms. It aims to partition a given data set into a certain number of clusters. After clustering on the data set, each observation from the data set belongs to the cluster with the nearest mean.

Firstly, we choose k points in the data set as k initial group centroids. These centroids are better placed as much as possible far away from each other since different initial location of centroids cause different results. The next step is to assign each observation to the group which has the nearest centroid. After that, the initial clustering has been done. We need to recalculate k new centroids as the centers of the clusters resulting from the previous step. Then we generate a loop repeating steps for assigning each observation and recalculate new centroids until no more changes are done on the position of centroids, which means the centroids do not move any more.

Finally, the algorithm intends to minimize the objective function, a squared error function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2$$

(4.3.1)

where $\| x_i^{(j)} - c_j \|^2$ is the distance measurement between one observation $x_i^{(j)}$ and the centroid of the cluster $c_j$. The measurement is an indicator of the distance of $n$ observations from their
respective cluster centroids.

However, the K-means algorithm does not always find the most optimal solution matching the minimum value of the global objective function. And it’s so significantly sensitive to initial conditions randomly selected that we have to run multiple times to reduce the effect. To avoid this sensitivity to initial conditions, we apply the Normalized Cuts algorithm in the experiments.

### 4.3.2 Normalized Cuts

The Normalized Cuts algorithm is most related to the formulation of grouping in the graph theory. Given an image and a weighted undirected graph $G = (V, E)$, the contours are detected and extracted as the nodes $V$ of the graph. The edge $E$ is formed between each pair of nodes. And the weight of every edge, $w(i, j)$, is a measurement of the similarity between nodes $i$ and $j$ in the feature space of contours. Then we can transform the contour classification problem into the graph partitioning problem seeking to partition the set of nodes $V$ into the disjoint sets $V_1, V_2, ..., V_k$. According to the measurement $w(i, j)$, the nodes in a subset $V_i$ have high similarity and the nodes in different subsets $V_i, V_j$ have low similarity. So we need an efficient method to partition the graph nodes based on the similarity metric.

Assume $A$ and $B$ are two disjoint subsets of the graph $G = (V, E)$, $A \cup B = V$, $A \cap B = \emptyset$. This can be done by simply removing edges connecting the two parts. The measurement of dissimilarity between $A$ and $B$ is computed by summing the weights of the edges that have been removed. In the graph theory, it is called the *cut*:

$$\text{cut}(A, B) = \sum_{u \in A, v \in B} w(u, v)$$

(4.3.2)

The minimum cut of the graph is necessary to be found out to solve the problem. A new dissimilarity measurement in defined in [23] [24], called the *normalized cut* (ncut):
\[ ncut(A, B) = \frac{cut(A, B)}{cut(A, V)} + \frac{cut(A, B)}{cut(B, V)} \]  

(4.3.3)

It calculates the cut cost as a fraction of the edge connections of all the nodes in the graph. In the same spirit, a normalized association to intra-groups is defined as:

\[ nassoc(A, B) = \frac{cut(A, A)}{cut(A, V)} + \frac{cut(B, B)}{cut(B, V)} \]  

(4.3.4)

For any cut \((S, \bar{S})\) in \(G\) in the normalized cuts approach, \(ncut(S, \bar{S})\) measures the similarity between different groups, and \(nassoc(S, \bar{S})\) measures the total similarity of nodes in the same groups. Due to \(ncut(S, \bar{S}) = 2 - nassoc(S, \bar{S})\), a cut \((S, \bar{S})\) minimizes \(ncut(S, \bar{S})\) while maximizes \(nassoc(S, \bar{S})\). And this problem can be formed into a generalized eigenvalue problem:

\[
\min_{(S, \bar{S})} ncut(S, \bar{S}) = \min_y \frac{y^T (D-W)y}{y^TDy} \\
\text{subject to} \begin{cases} 
   y_i \in \{1, -b\} \text{ for some constant } -b \\
   y^TD1 = 0
\end{cases}
\]  

(4.3.5)

where \(D\) is a \(n \times n\) diagonal matrix with \(d\) on the diagonal, and \(d(i) = \sum_j w_{ij}\). \(W\) is a \(n \times n\) symmetric matrix with \(W_{ij} = w_{ij}\). To make the problem tractable, the constraints on \(y\) are relaxed and allowed to be the real values. Then the relaxed problem can be solved by solving the generalized eigenvalue problem \((D-W)y = \lambda Dy\). We take the second smallest eigenvalue to partition the graph.

In the thesis, Normalized Cuts algorithm is used to cluster contours. We calculate how many kinds of orders in the systems corresponding to them. And the value is assumed to be the adaptive number of clusters. So we don’t need to set the number of clusters before N-cuts operating.
Chapter 5

Experiments and Analysis

In this chapter, experiments for measuring dynamic distances between contours and clustering contours based on the metric are characterized and analyzed. First, we introduce the data set used in the experiments, including synthetic data directly produced by MATLAB, synthetic images drawn in the computer and natural images from the BSDS500. Second, the dynamic distances between some simplistic contour trajectories are computed in both ideal conditions and noisy circumstances. We make a comparison among the distances with different input features, which substantiates the robustness and reliability of cumulative angles.

Furthermore, we combine the dissimilarity score function described in [1] with the order information of dynamic model corresponding to contours, the rank of Hankel matrices, to cluster the contour trajectories in the synthetic data. The results in the experiment have high accuracy by using this new dissimilarity metric based on dynamics. Last but not least, we apply the dynamic based contour clustering method on the contour segments chopped by corners in both synthetic images and natural images.
5.1 Data Set

In the experiments, the contour clustering method proposed in the thesis is applied on the synthetic data of contour trajectories directly produced by MATLAB commands and two kinds of images, synthetic images and natural images. Synthetic images are drawn in the Microsoft Office software, including lines, curves, circles, triangles, rectangles and some more complex graphs. And the natural images come from the BSDS500. It’s the Berkeley Segmentation Data Set consisting of 500 natural images, ground-truth data and benchmarks [25]. In the BSDS500, 300 images are used for training or validation and 200 fresh images are added for testing. It also provides the way of performance evaluation with measuring Precision / Recall on detected boundaries and other region-based metrics, which is a benchmark for comparing different contour detection and image segmentation algorithms.

5.2 Measurement of Dynamic Distances

The measurement of distances is very important factor in clustering task, which mostly determines the performance of clustering methods. To analyze the results of our contour clustering algorithm, we need to find out the dynamic distances without the influence of noise and discretization. To get the clean contour trajectories, we directly use the MATLAB commands to produce them. Then the normalized Hankel matrix (3.2.1) and dissimilar score function (3.2.2) are employed to calculate the dynamic distances in the ideal condition.

Figure 5.1 exemplifies the dynamic distances between some simplistic contour trajectories with different input features. There are twelve contour trajectories with different indexes in the figures. The index 1, 2, 3, 4 are circular arcs with the same radius and the radii of 5 and 10 are different from them. The index 6, 7 and 8 represent straight lines with different orientations. The corner and sinusoidal curves are displayed respectively by 9, 11 and 12.
Figure 5.1: Dynamic distances calculated by dissimilarity score function
The input features of contour in the subfigures are arranged as the following order: coordinates or positions in top-left; gradients or derivatives of positions (velocities) in top-right; tangent slopes in middle-left; tangent angles in middle-right; differences of angles in bottom-left; cumulative angles in bottom-right.

Positions and velocities are two dimensional vectors in the feature space. According to the top subfigures, the ways of normalization by both removing the mean of positions and computing the derivatives can eliminate the interference of translation. As a result, the dynamic distance between 1 and 3 should be zero since they are exactly the same circular arcs except translation. We also find out that the dynamic distances with the features of positions and velocities depend on the orientation of contour trajectory. And the velocities can take into account the rank information of Hankel matrix to some extent. This property makes the velocities more reliable than positions when computing the dynamic distances. For instance, the distance between 7 and 12 computed by the velocities is much larger than that computed by positions. Then we can easily classify straight lines and sinusoidal curves with the larger value of distances.

However, there is a critical problem about these two dimensional features. The distances mainly depend on the orientations of contours rather than the rank of Hankel matrices. Hence the distance between two circular arcs on the same circle (index 1 and 2) is not zero. And the distance between horizontal line and vertical line (index 6 and 7) is even larger than that between straight line and sinusoidal curve (index 7 and 12). To decrease the influence of orientation, we think about the angular information, one dimensional features.

We need to compute the derivatives at first to get the angular information of contour trajectories. Therefore the angular information has the translation invariance. With the input features of tangent slopes and angles, we only have the rotation invariance on the lines as no angular change occurs on them. The distances with angular information among all the lines are zero. For the angular differences and cumulative angles, we extend the rotation invariance to all the graphs. In the bottom subfigures, the distances between different circular arcs are zero excluding the index
The index 10 has a discontinuity in the value of angles from $-\pi$ to $\pi$ and only cumulative angles can avoid the problem. Moreover, the distance between the line and circular arc (index 1 and 6) is zero in the bottom-left subfigure. This may lead to that circular arc can be hardly distinguished from lines when the input feature is the angular differences. Accordingly, cumulative angles work best while taking the angular information as the input features.

Figure 5.2: Dynamic distances among the circular arcs on the same circle

Figure 5.2 manifests our assumption about the dynamic distances with the six kinds of features we talk about in the above statement. We extract the contour trajectory of a circle from the synthetic images by the Canny detector and rank minimization. And the contour trajectory is resampled as multiple contour segments with the same length, consisting of a cluster of circular arcs. Each arc has a unique index from 1 to 18 following the clockwise sampling direction. We compare each arc with the index 1 highlighted as the referent contour and get the dynamic distances with different features.
On the basis of these plots, the dynamic distances on the top row in Figure 5.2 are periodic following the orientation of arcs. And the problem of discontinuities is apparently clarified in the subfigures of angles and angular differences. The values of distance with cumulative angles are nearly zero owing to some noises, which matches what we expect.

5.3 Clustering of the Synthetic Data

To cluster contours in an environment without noise, we use MATLAB commands to produce the synthetic data of clean contour trajectories in Figure 5.3. These simple contour trajectories cover sixteen straight lines with varied orientation and translation, four circles with different radii, four ellipses with various rotations and several sinusoids with different frequencies and magnitudes.

Figure 5.3: Synthetic data for contour trajectories
5.3.1 Clustering by dissimilarity score function

A distance matrix is created to hold the dynamic distances between each possible pair of trajectories. These dynamic distances are computed by the dissimilarity score function in different feature spaces and represent one of dissimilarity metrics. But the Normalized Cuts method handles graph partitioning task in the context of similarity metrics. The distance matrix is required to be transformed into the similarity matrix by the formula $w = e^{-d}$, where $d$ is the dynamic distance and $w$ is the weight estimating the similarity. The larger distance comes with the smaller value of similarity.

Figure 5.4 shows the result of contour clustering by using Normalized Cuts. We plot the clustering outputs in accordance with the same sequence of features in Figure 5.1. In every subfigure, each color stands for one cluster of contours. The colors between inter-subfigures are unrelated since the output label of lines, circles and other curves might be random. In top subfigures, the results of clustering definitely rely on the orientation of trajectories. Straight lines and ellipses with the same orientation are categorized into one class. And circles are classified into single group as they are same in all orientations. For sinusoids, if clustered in the feature space of positions, they’ll be gathered with horizontal lines because their horizontal components take over the dominant factor in clustering. In the feature space of velocities, the increased dynamics resulting from the frequency factors can distinguish different sinusoids. Consequently, velocity is a more appropriate feature than position in our dynamic based contour clustering method.

Considering the middle and bottom subfigures, the input features are the angular information. It is hard to see which kind of feature is better since there is no big difference between the outputs. They successfully group all straight lines, whereas they are incapable of discriminating ellipses from circles even for the cumulative angles regarded as the best feature in the angular characteristics. Then another experiment has been made to explain why it didn’t work.
Figure 5.4: The output of contour clustering on the synthetic data
For a synthetic image including squares, triangles and circles, the contour trajectories are detected and displayed in the top-left subfigure of Figure 5.5. Afterwards, they are resampled as contour segments with the same length (\( L = 30 \)) and denoised by the rank minimization method. The top-right subfigure shows the cumulative angles of these contour segments. As we know, the zero responses of cumulative angles correspond with the straight lines and the monotonous decreasing linear outputs imply the circular arcs. For the corners, their outputs are the smooth step responses. Actually, there are immense visual disparities on the cumulative angles of lines, arcs and corners. However, we cannot categorize them correctly by employing the cumulative angles. Therefore, we assume that the dynamic distance calculated by dissimilarity score function is unable to make a distinction between contours without the same order.

**Figure 5.5:** The utilization of order information in contour clustering
5.3.2 Clustering by dynamic based dissimilarity metric

To improve the performance of our contour clustering method with cumulative angles, we combine the order information of dynamic system, the rank of Hankel matrices, with the dissimilarity score function described in the Chapter 3. When calculating the dynamic distance between two contours, we firstly compare their rank of Hankel matrices. If their ranks are equal, we still use the dissimilarity score function. If their ranks are not equal, we add a relatively large value on this function. This new metric is written as (3.2.3), called dynamic based dissimilarity metric. It attaches large value on the dynamic distances between contours with different orders. The larger value of distance helps us to better discriminate lines, arcs and corners in the bottom-right subfigure of Figure 5.5. The clustering of contours in Figure 5.3 is also improved by our new metric in Figure 5.6.

![Figure 5.6: Contour clustering using dynamic based dissimilarity metric](image)

To determine whether the clustering method based on dynamic distances has scaling invariance, more ellipses with different sizes and aspect ratios are added in the Figure 5.3. We firstly plot the result of clustering based on dissimilarity score function and its distance matrix in Figure 5.7.
Figure 5.7: Clustering based on the dissimilarity score function and its distance matrix

All of the contour trajectories in the top subfigure are indexed with unique numbers from 1 to 37. And the symbol from ① to ⑧ denotes the indexes of contour clusters. In the bottom-left subfigure, the distance matrix is composed of the dynamic distance of each possible pair of two contour trajectories. There are different distances among the circles with various radii based on the plot of distance matrix. As the outputs of cumulative angles for circles are straights lines, the distances between different circles are supposed to zero. But the slopes of the outputs depend on
the length of contour, which means the outputs of smaller circles come with larger slope and larger distance with lines. Consequently, the relative small circles (index 19 and 20) have been separated from big circles. Actually, the distances among lines, circles and ellipses are very small. So we cannot distinguish the ellipses from circles.

Furthermore, one sinusoid (index 33) is grouped into the class of lines since the distance between them is very small. And the outputs of the cumulative angles for lines are all zeroes. We suppose that the zero responses represent nothing about the information of the dynamic system. Thus the distances between lines and other curves, calculated by *dissimilarity score* function, are not robust and reliable.

The distance matrix in the bottom-right subfigure expresses the dynamic distances between intra-cluster and inter-cluster contour trajectories. For the diagonal of the matrix, each value is taken by computing the maximum distance between intra-cluster contours. And the maximum value is also regarded as the radius or the range of the cluster. The other values in the matrix stand for the distances between inter-cluster contours. We take the minimum value of the distances between contours belonging to different clusters. For example, we have the contour cluster $A$ and $B$. The radii of $A$ and $B$ are $\max \left\{ d\left( A, A_i \right) \right\}$ and $\max \left\{ d\left( B, B_j \right) \right\}$ respectively. And the distance between $A$ and $B$ is $\min \left\{ d\left( A_i, B_j \right) \right\}$. Based on the matrix, the cluster one (lines) is close to the cluster two (circles and ellipses), which matches the distance matrix in the bottom-left subfigure.

Figure 5.8 exhibits the result of contour clustering based on our dynamic based dissimilarity metric compared with the result in Figure 5.7. The cluster index from ① to ⑧ follows the value of orders in an incremental direction. That means the cluster ⑧ has the highest order in the graph. The new metric has the scaling invariance on the circles since the orders of all of them are two. Nevertheless, the ellipses with different scales are separated. If we change the aspect ratio of ellipses and make them fatter, they will be clustered in the group of circles. Besides, we find that the order information of sinusoids is related to the frequency of them. The higher
frequency the sinusoid is, the higher order it indicates.

Figure 5.8: Clustering based on dynamic based dissimilarity metric and the distance matrix

The result looks much better than that in the Figure 5.7 except that one sinusoid (index 33) is grouped with the ellipses. We check the singular value of both the ellipse and the sinusoid in the Figure 5.9. \( H_e \) and \( H_{sin} \) are the Hankel matrices of ellipse and sinusoid respectively. \( [H_e \ H_{sin}] \) is the matrix concatenating \( H_e \) with \( H_{sin} \) horizontally. And we find out that they are almost in the same subspace.
Hence, we assume that we can hardly classify them by using the new metric. But if we increase the number of clusters from 8 to 9, then we can solve the problem though the distance between the ellipse and sinusoid is still relatively small. Figure 5.10 shows the best result of contour clustering by using the new metric. The class index from ① to ⑨ still follows the value of orders in an incremental direction.

Figure 5.10: The best result of contour clustering in the experiments
5.3.3 The Influence of Noise on Clustering

According to the experiments, we can cluster the contour trajectories with high accuracy by using the new metric in an ideal condition. We wonder what happens if the noise is added on the original data. In the Figure 5.11, we add the Gaussian white noise with zero mean on the original synthetic data. The standard deviation in the left column is 0.01 and the value is 0.05 for the right column. In the top row, the result of classification is bad since the output of cumulative angle for lines are zero and very sensitive to the noise. Then we remove all the lines in the bottom row, the influence of noise has been significantly decreased. In the noisy environment and realistic application, we should extract the straight lines at first in the preprocessing step to guarantee the accuracy of clustering.

![Figure 5.11: The influence of noise on contour clustering](image)

49
5.4 Clustering of Contours Extracted from Images

In real application, the contour trajectories extracted from images are composed of a sequence of discrete pixels. If the discrete data is directly used to compute cumulative angles, it will cause enormous errors in the output and full rank of Hankel matrices. We have to clean the contour trajectories by rank minimization before clustering them. Generally, the corners exist on contours, especially for those extracted from synthetic images. The part of contours around corners cannot be modeled as the output trajectories of dynamic systems. Thus, a contour trajectory should be chopped at corners into segments. The dynamic distances between segments are calculated by our dynamic based dissimilarity metric in the feature space of cumulative angles. And Normalized Cuts method is used to group these contour segments into clusters.

![Figure 5.12: Clustering of contour segments by our approach](image)

In Figure 5.12, the segments in Figure 4.11 are clustered by our approach. However, for such simple contour segments, our approach cannot group all straight lines into the same cluster no matter how many clusters preset. The cumulative angles are oscillating near the corners even
though contour trajectories are cleaned by rank minimization at first. The oscillation greatly increases the rank of Hankel matrices for straight lines since their cumulative angles are zeroes and very sensitive to noise. So we should detect all straight line segments in the first place and set their rank to ones or zeroes. As their cumulative angles are closed to zeroes, the values of their 2-norm are supposed to be very small. To detect line segments in the figure, we just need an appropriate threshold. This preprocessing step before clustering will group all straight lines together and give a desired clustering result.

Normally, the number of clusters is predefined in clustering tasks. In our experiments, we adopt an adaptive technique to define the number by calculating how many kinds of dynamic systems with different orders exist. With the adaptive technique, it is unnecessary to predefine the number of clusters before our dynamic based contour clustering method. Figure 5.13 shows better contour clustering than that in Figure 5.12. Moreover, our approach is also applied on natural images from BSDS500 and the results are displayed in Figure 5.14. The detected straight lines are still plotted in blue color.

![Figure 5.13: Better clustering with straight lines detection firstly](image)

Figure 5.13: Better clustering with straight lines detection firstly
Figure 5.14: Clustering of contour segments from BSDS500
Chapter 6

Conclusions and Future work

In this thesis, we propose a new distance metric of contour clustering based on the dynamics. Firstly, the structure forest edge detector is used to detect and extract contours from images. We model these contours as the output trajectories of some dynamic systems. The cumulative angles working best in the experiments is calculated for each contour in the feature space. Then we build the Hankel matrices to encapsulate the dynamic information and compare them by the new metric, combing the dissimilarity score function with order information. After getting the distance matrix, we transform it to similarity matrix and use Normalized Cuts method to cluster the contours. In real application, it’s necessary to clean the contour trajectories by rank minimization to remove the effect of discretization. These trajectories should be also chopped at corners into segments before clustering.

Our dynamic based contour clustering approach exhibits the ability to group contours extracted from synthetic images into clusters accurately. In the future work, we need to find a better way to clean the data and extract features of contours from real images. We can train one classifier by applying contour clustering on a large data set. Then good contour classification can be achieved by this classifier. With contours recognized by the approach and structure information extracted from images, object recognition tasks will become much easier and more efficient.
Bibliography


International Conference on Computer Vision (ICCV), pp. 1841-1848, 2013


