

High Quality Image Retrieval System Using Computational Geometry

著者	Kaothanthong Natsuda
学位授与機関	Tohoku University
学位授与番号	11301甲第15926号
URL	http://hdl.handle.net/10097/58701

Acknowledgements

First of all, I would like to express the deepest appreciation to my supervisor, Prof. Takeshi Tokuyama, who has been extraordinarily tolerant and supportive to my study. Without his guidance and persistent encouragement this dissertation would not have been possible. I am deeply grateful to Assoc. Prof. Jinhee Chun for her generous supports for both academic and daily's life. Her supports were invaluable.

Besides, I am deeply grateful to my thesis committee: Prof. Ayumi Shinohara and Prof. Kentaro Inui of Tohoku University for their insightful comments and suggestions.

I would like to thank Assoc. Prof. Akiyoshi Shioura, Assoc. Prof. Sonoko Moriyama of Tohoku University, and Asst. Prof. Yota Otachi of Japan Advanced Institute of Science and Technology for their advice and comments which were an enormous help to me.

I would like to offer my special thanks to all of my co-authors (Takashi Horiyama, Takehiro Ito, Ryousei Kasai, Matias Korman, Cam Tu Nguyen, Hirotaka Ono, Yoshiki Ota, Xuan-Hieu Phan, Hiromi Takahashi, Ryuhei Uehara and Takeaki Uno,) for their discussions and scientific supports.

My grateful thanks are also extended to Prof. Dorothea Wagner, Dr. Martin Nöllenburg, and the members of the Institute of Theoretical Informatics of Karlsruhe Institute of Technology; Assoc. Prof. Hee-Kap Ahn and the members of the Geometric Algorithms Laboratory of Pohang University of Science and Technology; and Asst. Prof. Sangwon Bae of Kyonggi University for the warm welcome, generous support, and many interesting discussions during

my visit.

I would like to express my gratitude to Monbukagakusho (MEXT) scholarship, Japan Society for the Promotion of Science (JSPS) Research Fellowship for Young Scientists, and Grants-in-Aid for Japan Society for the Promotion of Science (JSPS) Fellows for their financial support.

Special thanks to all members of Tokuyama Laboratory for the continuous support and friendships. To all my dear friends, who I could not name all of them since I am afraid of missing someone, I sincerely appreciate their friendships and unconditional supports.

Finally, I wish to thank my family for their endless support and encouragement throughout my study.

Abstract

In this dissertation, we study an image retrieval system. The two most important properties of a retrieval system are: 1. Short time processing of queries; and 2. Quality of the retrieved result that must be as good as human does. To accomplish the quality expectation, we focus on an image retrieval system that uses the shape feature as a similarity measure. A major drawback of applying shape feature in the image retrieval application is the time complexity for computing the similarity since one query needs to be compared against each object in the database. Therefore, the retrieval time is linear in the database size and high degree polynomial in the size of each object.

To process a query efficiently, the system consists of two parts; preprocessing and similarity assessment. In the preprocessing part, a search structure of the shape features of each object in the database is prepared in advance. This process includes the image segmentation, which separates objects from an image, and the modeling of the shape feature into the search structure. In the similarity assessment part, given a shape feature of a query, a set of similar objects is retrieved via the precomputed search structure based on a high quality shape similarity measure.

We propose an algorithm called *room-edge region* for image segmentation. Given an $N = n \times n$ pixel grid \mathcal{G} , the time complexity for computing the optimal room-edge region is $O(N^{1.5})$. To deal with the complicated shape of objects, an image is subdivided into small subrectangles and the rooms-edge region is applied independently. We propose two

algorithms for subdividing a pixel grid: 1. Using a set of baselines and 2. Using a quadtree structure. Given a pixel grid \mathcal{G} and a number of k baselines, the optimal location of the k baselines can be computed in $O(N^{2.5})$ time. On the other hands, given a depth d of the quadtree, the optimal decomposition of a pixel grid \mathcal{G} can be computed in $O(N^{3.5})$ time.

To support an efficient query, the shape feature of the images in the database is organized in the search structure. We propose an effective shape descriptor called *Distance Interior Ratio* (DIR) that can be integrated to a search structure efficiently and is scale, rotation, and translation invariant.

In order to improve the time complexity for computing similarity measure of two shapes, we present a framework that reduces the matching cost by selecting a set of corresponding points for estimating the optimal transformation for computing the modified-Hausdorff distance. Moreover, we proposed a local shape descriptor called *Local Distance Interior Ratio* (LDIR) for solving the correspondence.

Our image retrieval framework consists of two steps. Firstly, it applies the search structure of the DIR descriptor for screening the relevant objects from the database. Next, relevant objects are reranked using the framework for computing the modified-Hausdorff distance. The time complexity for processing one query is sublinear in the database size, that is $O(|\mathbf{C}|\mathcal{K}^2m^2)$, where m is the number of boundary points, \mathcal{K} is a set of correspondence, and \mathbf{C} is a set of objects retrieved from the search structure. The collection \mathbf{C} is significantly smaller than the database size O . This improves the naive $O(Om^6)$ time complexity, which uses the modified-Hausdorff distance solely to compare the shape similarity of a query to each object in the database.

Contents

1	Introduction	1
1.1	Problem Statement and Motivation	3
1.1.1	Image segmentation	4
1.1.2	Shape-based Image Retrieval	5
1.2	Contributions	7
1.2.1	Image Segmentation	7
1.2.2	Shape-based Image Retrieval	9
1.3	Conclusion	12
1.4	Organization of Thesis	14
2	Preliminary	15
2.1	Image Retrieval	15
2.1.1	Query using keywords	15
2.1.2	Query using an image	17
2.1.3	Preliminary Works	18
2.2	Image Segmentation	22
2.2.1	Definition	23
2.2.2	Segmentation Methods	24
2.2.3	Preliminary	28

2.2.4	Previous Work	30
2.3	Shape-based Image Retrieval	33
2.3.1	Geometric Pattern Matching	34
2.3.2	Shape Descriptor	35
2.3.3	Spatial-based approach	38
2.3.4	Preliminary	39
2.3.5	Previous Work	44
3	Image Segmentation using Maximum Weight Base-monotone Region	51
3.1	Introduction	51
3.1.1	Summary of the Result	53
3.2	Room-Edge Maximum Weight Region	53
3.2.1	Algorithms for the room-edge MWRP	55
3.2.2	Image Segmentation Application	63
3.2.3	Experiment	63
3.3	Improving Segmentation Result of Room-Edge MWRP	68
3.3.1	Grid Decomposition	69
3.3.2	Optimal location of k baselines	73
3.3.3	Quadtree Decomposition	77
3.4	Experimental Result	80
3.4.1	Setting of the Experiment	80
3.4.2	Grid Decomposition: k -Baselines	81
3.4.3	Grid Decomposition: Optimal k -Baselines	89
3.4.4	d -depth Quadtree Decomposition	93
3.4.5	Segmentation Result Comparison	100
3.5	Discussion and Conclusion	111

4	Distance-based Shape Descriptor	113
4.1	Introduction	113
4.2	Summary of the result	114
4.3	Distance Interior Ratio	115
4.3.1	Distance Features	116
4.3.2	Structural Histogram	118
4.3.3	Similarity Measurement	120
4.3.4	Descriptor Properties	122
4.3.5	Computational Complexity	124
4.3.6	Experimental Result	124
4.4	Local Distance Interior Ratio	128
4.4.1	Distance Feature	129
4.4.2	Similarity Measurement	131
4.4.3	Computational Complexity	132
4.4.4	Experimental Result	132
4.5	Discussion and Conclusion	136
5	Framework for Shape-based Image Retrieval	139
5.1	Introduction	139
5.2	Summary of the result	140
5.3	Computing shape distance using correspondence	141
5.3.1	Solving Correspondence	142
5.3.2	Computing shape distance	144
5.3.3	Experimental Result	146
5.4	Shape-based image retrieval framework	149
5.4.1	Preprocessing Stage	150
5.4.2	Screening Stage	152

5.4.3	Refining Stage	154
5.4.4	Analysis and Time Complexity	156
5.4.5	Experimental Result	158
5.5	Discussion and Conclusion	160
6	Conclusion	165
6.1	Future Work	167

Chapter 1

Introduction

The increase in data, e.g., text, image, and video data, being exchanged on the internet asks for effective tools for retrieving information, and thus has gained the attention of researchers. An *information retrieval* system is a system for finding a set of objects in a large-sized database that are similar to an input query.

In this research, we focus on an image retrieval system. Given a query image, the system identifies its similarity with respect to image content of the database. Such a system is called a *Content-Based Image Retrieval* (CBIR). It has been utilized in many applications, such as face detection [49], scene understanding [58], and object detection [38].

A CBIR system is composed of two main parts [55]: 1. Preprocessing and 2. Retrieving.

1. **Preprocessing:** To retrieve images efficiently, the visual contents of the images in the database are extracted, compressed into feature vectors, and stored in a search structure.
2. **Retrieving:** To retrieve a set of similar images to a given query, the feature vector of the query image is extracted and processed using the same method as in the preprocessing. Utilizing this feature vector and the precomputed search structure, the system is able to browse for similar images in the database efficiently. The retrieved images

are ranked with respect to the similarity of their feature vectors to the feature vector of the query object.

Many mathematical models have been proposed to assist computers to compare visual image contents [32]. The image features can be grouped into the following types [44]: color [30], texture [65], local [67], and shape features [92].

A color feature is a good baseline for retrieving general photographs since it can be transformed into a feature vector using a histogram of the distribution of the color pixels in an image. The similarity of two images is measured by the difference of the color feature vectors. A texture feature is used for finding important regions of an image. It outperforms the color feature in terms of accuracy but the computation cost is much higher than for the color feature [34]. A local feature is any feature of a subimage and it is widely used for recognizing objects in an image [67]. One limitation of local features is that the system requires huge training data. Shape is one of the basic features used for describing image content [95, 91]. Combining a shape feature with some other feature, such as a color feature, can significantly improve the performance of a CBIR system [66, 92, 62].

In addition, defining similarity of two shapes is the main component of a retrieval system. One possibility is to embed shapes in a metric space and use a distance function to measure similarity between them. The selection of the distance function has a big influence on the performance of the system [34]. Examples of distance functions are: Minkowski L_p norm, Fréchet distance, Hausdorff distance, and Euclidean distance. The major drawback of using such functions is that it is not easy to be combined with an n -nearest neighbor search method. A solution is to compute shape information and express it in form of a histogram. An example of shape information that is easy to compute is a pairwise distance among boundary points. However, feature vectors obtained from histograms are not enough to uniquely represent a characteristic of shapes [85, 15].

When considering shape feature as a shape similarity measure in a retrieval system, noise

in an image has a big impact on the similarity measure [77]. In contrast to a color or a texture features which can be processed directly from an image, a shape feature essentially requires an additional preprocessing in order to allow a retrieval system to unambiguously obtain and handle shape information from an image.

Image segmentation is an important tool for assisting the CBIR system to handle shape information. Given an image, it is separated into multiple regions using predefined criteria, such as continuity of either a color feature or a texture feature [81]. The boundary of the main region is extracted in such a way that it is simple and accessible to similarity measuring algorithms. Many image segmentation approaches have been proposed in the past [59, 81, 48]. However, obtaining an accurate boundary from an image with a complex background or an image with noise is still challenging [62, 34].

In this research, we target a CBIR system that employs shape features as a similarity measure. The expectation toward the system is to provide efficient processing time of a single query and accurate results. An image segmentation method and a shape similarity measure are two important components. One limitation of the current shape similarity measures is the trade-off between time complexity and accuracy. Moreover, the quality of the image segmentation has a big impact on the accuracy of the retrieved result.

1.1 Problem Statement and Motivation

We divide the problem into two main tasks. The first one is the shape feature extraction process that mainly focus on the image segmentation. The second task is a mathematic representation of a shape that must give an accurate similarity measure and minimizing processing of a single query.

1.1.1 Image segmentation

Image segmentation is used for locating the main region and separating the foreground from the background [77, 62]. A variety of either supervised or unsupervised methods have been proposed in the past, such as curve evolution [54], energy diffusion [19], homogeneous region [2, 37, 28], and graph partitioning [18]. Since images are often corrupted with noise, defects, and occlusion; robust segmentation tools are required [34].

The daily increase in the images being uploaded on the internet is huge, a requirement of a fully-automatic segmentation tools attracts the attention of many researchers since segmenting general images is an unsolved problem [32, 34, 96]. Some methods require additional information on the location of the main content [54, 81] or a sample foreground and background pixels [17, 37, 30].

The desired properties of a good segmentation can be summarized as follow [48]:

- Regions of a segmented image should be *uniform* and *homogeneous* with respect to color or texture characteristic.
- Region interiors should be simple and without many small holes.
- Adjacent regions should have significantly different values with respect to their characteristics.
- Boundaries of each region should be simple and spatially accurate.

Achieving all the desired properties is difficult because strictly uniform and homogeneous regions are typically full of small holes and unconnected components, which causes ragged boundaries. Such effects, for instance, occur if segmenting a complex background image [62]. In addition, determining the quality of the final result automatically is not easy [96]. Moreover, high accuracy segmentation is computational intensive.

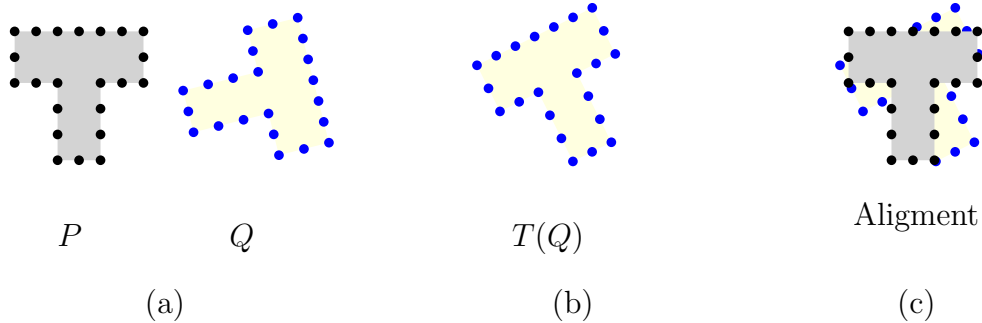


Figure 1.1: (a) A set of points P and Q on the shape contour of two polygons. (b) A rigid motion T applied to Q . (c) An alignment of P and $T(Q)$.

1.1.2 Shape-based Image Retrieval

The shape of an object is a very useful geometric feature for human beings. It is also one of the basic features used for describing image content [95]. It has been utilized in many applications such as character recognition and fingerprint recognition. However, shape information is often difficult to use in more general domains. Especially, image retrieval is a major internet technology, but the current retrieval methods are not really considering shape information: Commercial systems, such as Google image search, use texts and labels attached to images [92, 91].

Similarity assessment

Many methods are proposed for shape similarity assessment, which can be summarized into three approaches: 1. Geometric pattern matching, 2. Shape descriptor, and 3. Spatial matching.

1. **Geometric pattern matching:** Given two objects in terms of boundary points as shown in Figure 1.1, their similarity is measured by computing the shape distance using some similarity metric, such as the bottleneck distance [36] or the Hausdorff distance [50]. In order to obtain an optimal similarity measure, we need to find the optimal

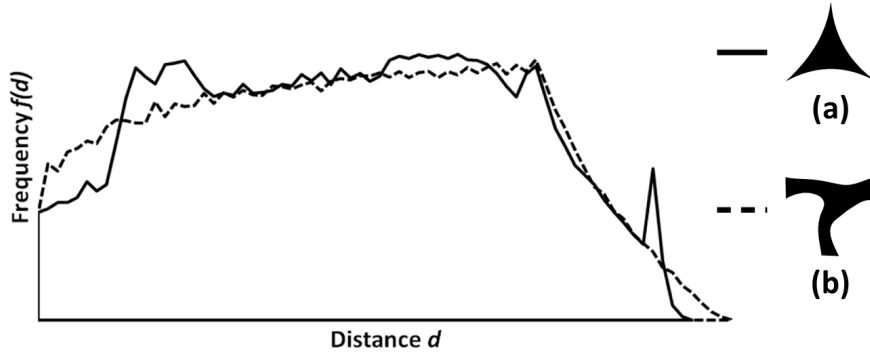


Figure 1.2: The distance distributions of the boundary of a “Triangle” and “T”.

rigid motion T that maps one shape onto the other such that the shape distance is minimized.

The advantage of this approach is that it can also be used to partially map objects onto each other [90]. However, the time complexity for solving the optimal rigid motion depends on the number of boundary points.

2. **Shape descriptor:** Given a set of boundary points P , a *shape invariant* is a function $F(P)$ that transforms the shape to a feature vector. A shape invariant must be such that if P resembles to Q then $F(P) \approx F(Q)$; and also invariant under rigid motions and robust against small perturbations [70, 84, 88, 94].

Considering the distribution of the pairwise distances among the points on the shape boundary is natural. A histogram of the pairwise distances is computed to obtain a feature vector $F(P)$. In this way, shapes can be compared in a vector space. A distance-based shape invariant is a basic one which has been applied for pattern matching of three dimensional objects [70, 53]. However, a major drawback is the existence of *homometric pairs* [85, 15], which are objects with different shape but similar distance distribution as shown in Figure 1.2.

3. **Spatial matching:** Shape features containing local information is attached to some

points of the shape boundary points [13, 86, 63, 47, 23]. By local information, such as texture, color, or shape information, we refer to structural information obtained from the perspective of a specific point [25]. By comparing the local information, the corresponding points of two shapes can be identified [12]. Such pairing points can be used to find the rigid motion to align the two shapes.

Retrieval time

A geometric pattern matching approach is used as a similarity metric in this work. One limitation of employing such a method in a retrieval system directly is the time complexity for processing one query.

Suppose that a query image is modeled by a set of arbitrary m points on a shape contour Q , and we want to find the images in the database whose shape contour is similar to Q . Using a geometric pattern matching, we need to find the optimal rigid motion T that maps each object in the database onto Q such that the shape distance is minimized. Suppose that the similarity of two shapes can be measured in $g(m)$ time, and there are O objects in the database, the most similar images can be retrieved in $O(Og(m))$ time. Since O is often quite huge, such a method is not practical.

To cope with the huge retrieval time, retrieving similar objects through an efficient search data structure is one solution. Unfortunately, the search space of a similarity measure based on shape distance is not easy to organize in order to support fast query.

1.2 Contributions

1.2.1 Image Segmentation

We propose a semi-image segmentation algorithm that is formulated as a combinatorial optimization problem. Assume that either a positive or a negative weight is assigned to

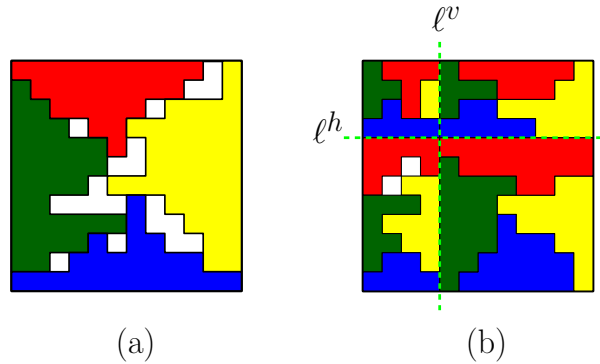


Figure 1.3: (a) A room-edge region with a four coloring. (b) A room-edge region computed of four subgrids obtained by the lines ℓ^v and ℓ^h .

each pixel. We consider the *Maximum Weight Region Problem* (MWRP), which is to find a region satisfying certain properties such that the total weight of the pixels in the region is maximized. Two methods are considered:

1. We propose an algorithm for computing a region, called a *room-edge* region, that is the union of four disjoint base-monotone regions, where each region is computed from a boundary edge of a pixel grid as shown in Figure 1.3 (a). Each color in the figure corresponds to a base-monotone region.

The room-edge region can be computed in $O(N^{1.5})$ time using $O(N^{1.5})$ space, where N is the number of pixels in the pixel grid. We also present a framework for applying the room-edge region to an image segmentation application.

2. An algorithm for improving the segmentation of images that contain many objects. The idea is to decompose a pixel grid into small subgrids and compute the room-edge region separately for each subgrid as shown in Figure 1.3 (b).

To achieve a result of high quality when segmenting an image containing a sophisticated object or many objects, the subdivision into subgrids plays an important role for decomposing the pixel grid. For comparison, see Figure 1.4. We propose two methods:

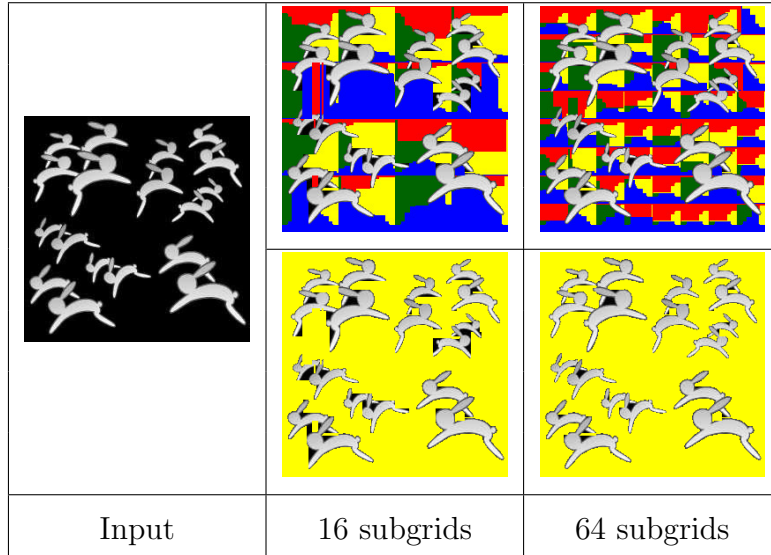


Figure 1.4: Segmentation of a picture of rabbits using 16 and 64 subgrids.

a grid decomposition algorithm, which can be solved in $O(N^{2.5})$ time, and a quadtree decomposition algorithm, which can be solved in $O(N^{3.5})$ time.













































1.2.2 Shape-based Image Retrieval

We present two main results. The first result aims to cope with the shape similarity assessment when using a distance-based shape descriptor. The second result targets the retrieval time when applying a shape distance as a similarity measure.

Similarity measurement

Two distance-based shape descriptors that are able to deal with homometric pairs of objects are proposed. The first descriptor describe an entire shape in form of a feature vector. The second descriptor is the same as the first except that it conveys local information of the shape from the perspective of some of the boundary points. The details are as follows:

Table 1.1: Comparison of the five closest retrieved objects. We use Kimia’s dataset [80].

Input	Method	1	2	3	4	5
	Approx. Mod.					
	Haus.					
	Approx. Mod.					
	Haus.					
	Approx. Mod.					
	Haus.					
	Approx. Mod.					
	Haus.					

1. A novel shape descriptor, called *Distance Interior Ratio* (DIR), is proposed. The shape contour is described using the relative distribution of the pairwise distances and the interior ratios among the boundary points. The descriptor can be computed efficiently.

We demonstrate the accuracy of the proposed descriptor and its ability to be integrated into a search structure. The Bull’s Eye Percentage (BEP) of the proposed descriptor is 73.73%, which is higher than the BEPs of the Shape Context [14] and the Inner Distance Shape Context [63] descriptors.

2. We extend the idea of the DIR descriptor in order to improve the accuracy. Another shape descriptor, called a *Local Distance Interior Ratio* (LDIR), is proposed that describes the shape contour using local information.

The LDIR descriptor outperforms the DIR descriptor, it achieves a BEP of 78.27%.

Retrieval time

We aim to reduce the time complexity when using the modified-Hausdorff distance, which is less sensitive to noise than the Hausdorff distance, as a similarity measure. A method for improving the time complexity for a single comparison and one for improving the overall retrieval time of the CBIR system are proposed.

1. To improve the time complexity for finding the optimal rigid motion to map two shapes, we employ the LDIR descriptor for selecting a set of corresponding points of the two shapes. Basically, we look for a bipartite matching between the boundary points of one shape and the boundary points of the other. The Hungarian method can be applied, which is efficient.

The time complexity for computing the modified-Hausdorff distance is $O(\mathcal{K}^2m^2)$, where \mathcal{K} is the number of the corresponding points and m is the number of the boundary points.

Due to the time complexity when computing the optimal modified-Hausdorff distance, an approximate modified-Hausdorff distance is applied in the experiment. Our experimental result suggests that for computing the approximate modified-Hausdorff distance, using the corresponding points obtained from our method yields a good result. A few examples are shown in Table 1.1 in which “Approx. Mod. Haus.” shows the retrieved result using the approximate modified-Hausdorff distance and “LDIR+” refers to the result of the proposed method.

2. To improve the overall time complexity of a CBIR system using the above method as similarity measure, we present a three-stage framework to avoid a linear scan of objects in a database that consists of a preprocess, screening, and refining stage as illustrated

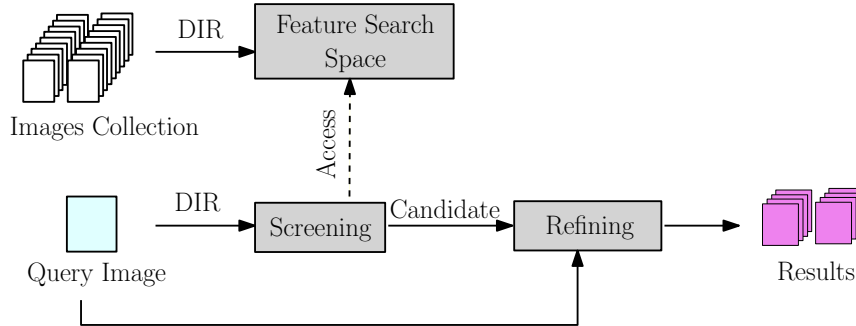


Figure 1.5: Overview of the proposed 3-stage shape-based image retrieval system.

in Figure 1.5.

Image in a database are preprocessed that includes extracting shape boundary, compressing the boundary to a feature vector, and storing the feature vector in a search structure. Given a query, a set of candidate objects is retrieved through a precomputed search structure; these candidate objects are screened and selected with respect to the DIR shape descriptor similarity. The method explained above is applied in order to compute the similarity measure between the query and each candidate. The most similar candidates are taken and ranked with respect to the shape distance.

The time complexity for retrieving result is sublinear in the size of the database. Computing the similarity measure using the proposed framework, the time complexity is $O(|\mathbf{C}|\mathcal{K}^2m^2)$, where \mathbf{C} is the candidate set. The total time complexity for one query is $O(\mathcal{Q} + |\mathbf{C}|\mathcal{K}^2m^2)$, where \mathcal{Q} is the time complexity for screening the candidates.

1.3 Conclusion

We propose a semi-automatic image segmentation algorithm that formulates the problem as a combinatorial optimization problem. The experimental results illustrate the quality of segmentation of an image containing objects with sophisticated shape. Compared to state-

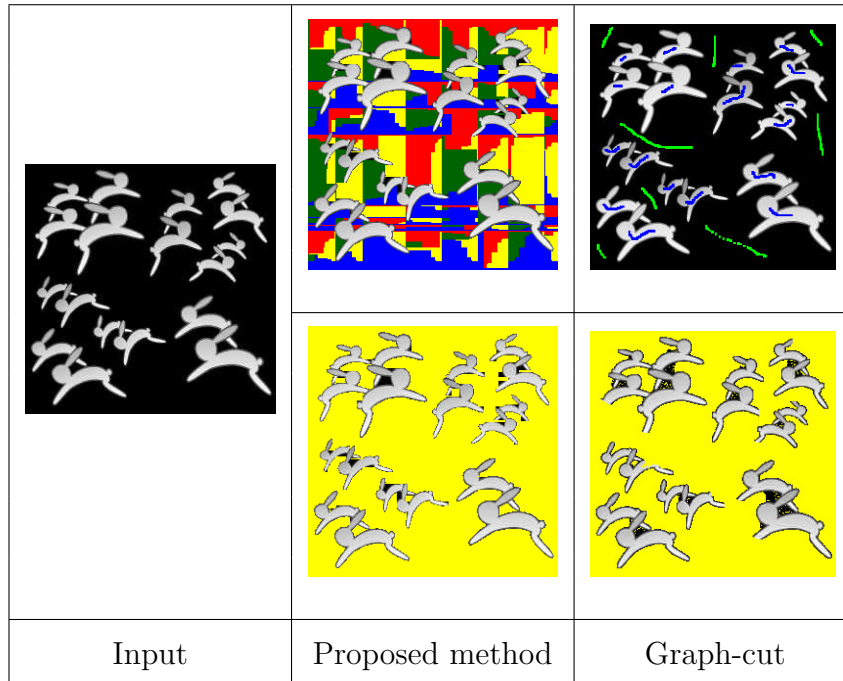


Figure 1.6: Comparison of a picture of rabbits using our proposed segmentation algorithm and the graph-cut method. The proposed method uses 64 subgrids. A set of green pixels is a constraint of background region. Note that the graph-cut method requires addition information (blue and green marks in the top-right picture).

of-art methods, such as the graph-cut method [17], our method does not require additional information about image content provided by the user. A comparison of the segmentation result is given in Figure 1.6.

The proposed retrieval framework allows similar images to be retrieved in sublinear time complexity in the database size. In addition, the combination of the LDIR descriptor and the Hausdorff distance for computing the similarity measure yields a better result than applying the Hausdorff distance only. Furthermore, the computation time for a pair of shapes is 0.391 seconds on average for the former approach while it is 1.506 seconds for the latter.

1.4 Organization of Thesis

The rest of this thesis is organized into 6 chapters:

- Chapter 2 presents our preliminary works on image retrieval, which are image annotation and shape-based image retrieval. General methodologies being applied in image segmentation and shape similarity measurement are presented.
- Chapter 3 proposes an algorithm for computing a room-edge maximum weight region. A framework for transforming the maximum weight region problem to an image segmentation problem is presented. To achieve a high quality segmentation result, when segmenting sophisticated regions, two methods are proposed.
- Chapter 4 proposes a novel shape descriptor, called Distance Interior Ratio, that is a relative distribution of the Euclidean distances and the interior ratios among the boundary points of a shape. Also a local shape descriptor, called Local Interior Ratio, which describes a shape from the perspective of a specific point is proposed.
- Chapter 5 adapts the Local Interior Ratio shape descriptor to reduce the time complexity for finding the optimal rigid motion to compute the Hausdorff distance. Also a CBIR framework for improving the time complexity for processing a single query is proposed.
- Chapter 6 concludes the dissertation and outlines future work.

Chapter 2

Preliminary

2.1 Image Retrieval

Image retrieval is the process of browsing and identifying images (objects) in a large database that are similar to a given image [95]. Two approaches of image retrieval system are classified in accordance with a format of an input query: (1) using document attached to an image; (2) using image content such as color, texture, and shape features.

2.1.1 Query using keywords

Basically, associated metadata such as keywords, tag, and attached document are used to retrieve images. Examples of the metadata are filename, title of an image, and surrounding document which are illustrated in Figure 2.1. One problem of utilizing the metadata is the reliability of the image content and the surrounded document. As shown in Figure 2.2, the word “jaguar” is ambiguous since it can refer to either a car or an animal. To improve the retrieved result using this method, we need to assign the keywords which are relevant to the image content. However, assigning the keywords to a huge number of images manually is not practical.

To cope with a vast size of images, an image annotation is used to automatically assign keywords according to image content. One difficulty is the semantic gap between low level feature of image content and high level feature such as keyword is the major problem for image annotation.

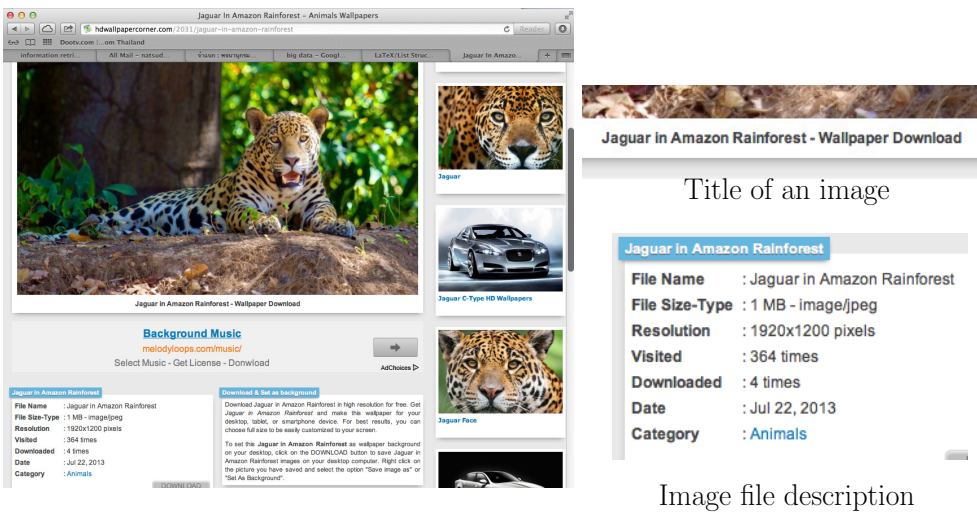


Figure 2.1: An image of jaguar and the description of the image file.

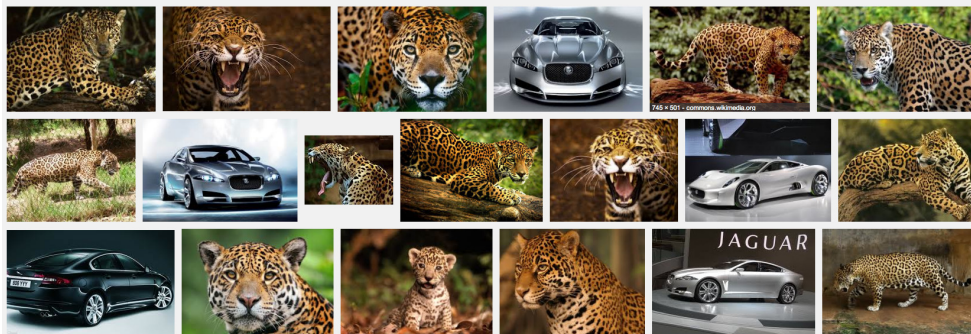


Figure 2.2: Retrieved result using “jaguar” keywords.

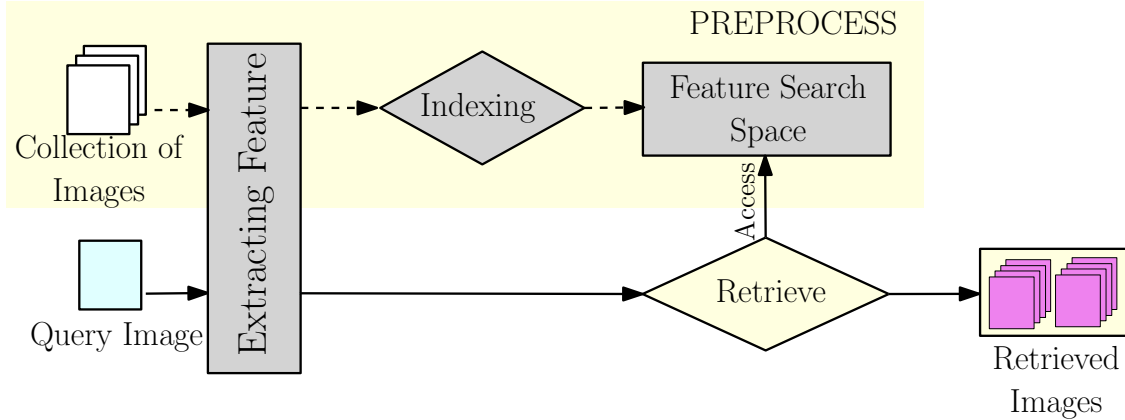


Figure 2.3: Overview of the image retrieval system.

2.1.2 Query using an image

When the text feature is absent or incomplete, a *Content-Based Image Retrieval* (CBIR) is used to achieve higher accuracy of retrieved images according to the content such as color, texture, shapes, or any information extracted from the image content. So far, CBIR has been only applied for specific professional areas such as architecture design, art and craft museums [21], airline-related images [3], and medical imaging [66, 69].

The CBIR is the process of retrieving desired images from a large image database on the basis of features extracted from the image [55]. The overall system is illustrated in Figure 2.3. The image features in the database are extracted and described by multi-dimensional feature vectors. The indexing data structure is applied to store the feature vectors and also for support an efficient search for the image in the collection. Recent CBIR systems incorporate users relevance feedback [32] to improve the retrieved result.

To construct a database index, feature extraction and representation are formulated to perform image clustering and classification for efficient retrieval. In CBIR, images are indexed by their visual content, such as color, texture, shapes.

- Color is feature often cited as a baseline in CBIR [34] because it gives a reasonable

good baseline for retrieving general photograph. Moreover, the representation, that is a color histogram, is easy to compute and compare. Color space such as RGB, LUV, is partitioned into ranges. For each partition, the pixels with a color within its range are counted, resulting in a representation of the relative frequency of the occurring color. Time complexity for computing similarity metric of the two color histogram is efficient since it depends on the size of the partitioned ranges.

- Texture features such as Scale-Invariant Feature Transform (SIFT) [65] is a salient attribute which is often used for representing important regions of images in real world [66]. Although texture feature outperforms the color feature, the computation cost is much higher. Moreover, none of the current descriptor conveys a complete description of the texture properties of an image.
- Shape is an important visual feature and also one of the basic features used to describe image content. The combination of shape feature and other features such as color and texture can significantly improve the image retrieval performance [34]. However, computing shape representation and description is a difficult task [95, 68] as compare to color and texture features. One reason is due to the inaccuracy of segmentation. Moreover, it is a grand challenge to attain two trade-off requirements of efficiency and accuracy when employing shape feature in CBIR.

2.1.3 Preliminary Works

Feature-Word Topic Model for Image Annotation

In our preliminary research, we present a novel method for image annotation to improve semantic gap between image content and words. The proposed a method is based on Multiple Instance Learning (MIL) and topic modeling. By using topic modeling, word is allowed to take associations into image annotation. The main idea of framework is to guess the scene settings or the story of the picture to obtain better image annotation.

The method proposed is divided into two parts: (1) Training and (2) Annotation. The detail of each part is as follow:

1. **Training Step:** This step consists of two stages. The first stage is to estimate the feature-word distribution by considering the probability of features vector and a set of label assigned to an image as shown in Figure 2.4. The image features being used are color and texture. The second stage is to estimate word-topic distributions. A set of labels assigned to an image in the training set is grouped in order to relate a set of words to a topic. Then, the probability of a word given a topic and the probability of a topic given a feature vector are generated.
2. **Annotation Step:** In this step, a Feature-Word-Topic model which is generated in the training process is applied to annotate a testing image. First the candidate words are selected according to the values of the feature-word distribution. Then, the topic is estimated from the candidate words. The final tags are selected according to value of the probability of words given topic and probability of selected topic of the image.

The experimental result shows that our proposed Feature-Word-Topic model improves the Supervised Multiclass Labeling model (SML) which is state-of-the-art method considerably 29%. An example of the comparison of assigned labels between the SML method and our proposed method is shown in Figure 2.5.

Shape-Based Image Retrieval

In this preliminary work, we apply a geometric pattern matching which gives high quality result as a similarity measurement. One limitation of applying this method is the time complexity for processing one query.

Suppose the query image is modeled by a geometric shape Q , and we want to find shapes P in the database resembles to Q . Given a pair P and Q of geometric objects, we can

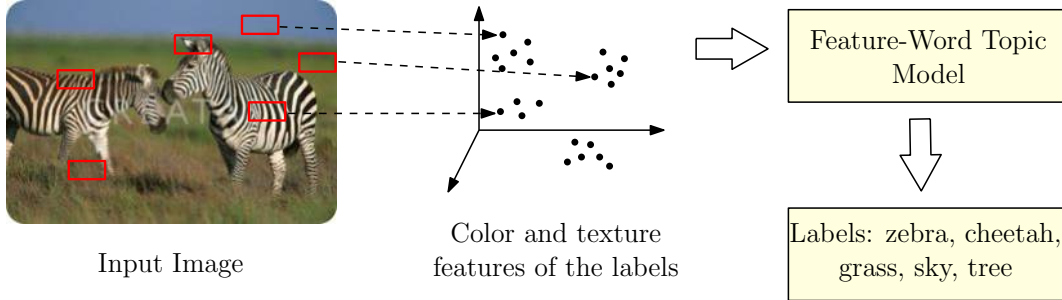


Figure 2.4: A feature space F which x -axis represents the Euclidean distances and y -axis represents the DIR. F is equally divided into 4×4 blocks.


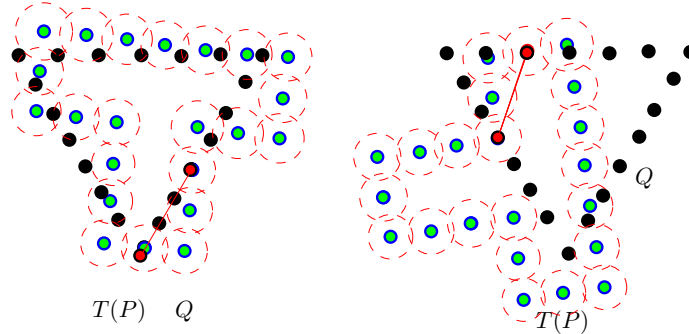
	SML: zebra, cheetah, grass, ocean, house
	Our method: zebra, cheetah, grass, sky, trees

Figure 2.5: Example of annotations using Supervised Multiclass Labeling (SML) and our method.

give an objective function to describe their similarity. The computation of the similarity of two objects is widely studied as geometric pattern matching. In our preliminary work, the objective function is the number of common points of the two boundary point sets [4], and P can be transformed by rotation, translation, reflection, and scaling to $T(P)$. Here, we need to find the transformation T to maximize the number of coincide points between $T(P)$ and Q . Figure 2.6 shows examples of the coincide points of the two transformation. The common points are the black vertices which fall in the red circles. We can see that the transformation on the left results in larger number of coincide points than the right one.

One serious drawback of this method is that the lack of scalability. Suppose we have N shapes in the database. If we need to count the number of coincide points between the query shape Q and all the shapes in database, it needs huge query time and cannot be accepted as



Number of coincide points between $T(P)$ and Q according to $T[r, r'; s, s']$

Figure 2.6: Example of coincide points between the two sets of boundary points.

a tool in information retrieval. Therefore, we need a methodology to avoid this huge query time to employ this approach in an information retrieval system.

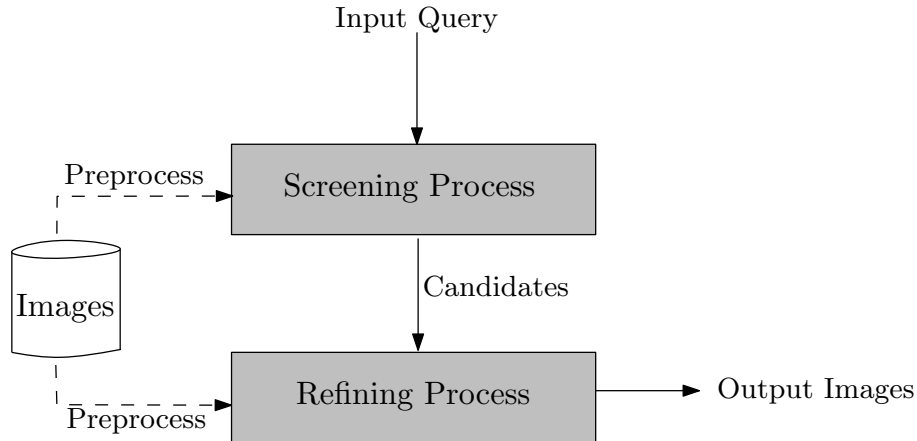


Figure 2.7: Examples of retrieved result using the similarity of the input keyword and metadata.

In a preliminary work, we propose a framework for an image retrieval based on shape information. Our main idea is to compress the size of data by applying the feature search, and then apply the pattern matching. In this work, the shape feature are preprocessed and stored in an indexing data structure, i.e. a hash table. We use a shape descriptor to describe

the shape in a compact fashion. Moreover, it should be invariant to rigid motion, scaling, and noise. The proposed algorithm for retrieving a set of similar shapes consists of two processes, (1) *Screening Process* and (2) *Refining Process* as illustrated in Figure 2.7.

1. **Screening Process:** In this process, we use a pairwise distance among points on the boundary to describe the shape. An indexing data structure, which is preprocessed, is used for retrieving a set of candidate images which have the same shape signature.
2. **Refining Process:** We employ a computational geometric algorithm to the image retrieval in order to remove the irrelevant images in the candidate set.

Using the proposed framework, the time complexity for computing pattern matching becomes independent of the database size. The cardinality of the candidate set, $Y(Q)$, to be sent to the refining process is $C = \delta N$, where N is the total number of objects in the data set. The time complexity for pattern matching is t . The time complexity for our query is $O(Kt + |X(Q)|)$, assuming that the shape signature is described in the statistical information of a constant size. The preprocessing time is $O(Nn^2)$ time, where n is the number of sample points uniformly taken from each object in the database.

2.2 Image Segmentation

Image segmentation is one of the important parts of the image processing and pattern recognition. It is a process for separating a digital image into multiple regions [28]. The ultimate goal of this process is to simplify the image in order to get more meaningful for analysis [77]. It is also used for locating the interested region in an image. Figure 2.8 depicts a segmentation result which the interested region is a swan in the image. The increased in the number of images causes a demand for automatic image segmentation. However, fully-automatic segmentation for general images is an unsolved problem.

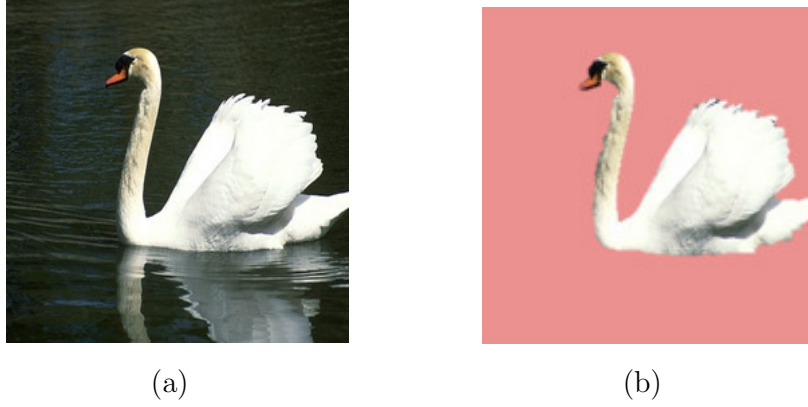


Figure 2.8: Example of the segmentation regions. (a) A given image. (b) Segmented region.

2.2.1 Definition

A formal definition of image segmentation is as follow [44]: Let \mathcal{G} represent the entire spatial region occupied by an image \mathbf{I} . An image segmentation is viewed as a process that partitions \mathcal{G} into s subregions, denoted by R_1, R_2, \dots, R_s , such that

- (a) $\bigcup_{i=1}^s R_i = \mathcal{G}$. The segmentation must be complete that is every pixel must be in a region.
- (b) R_i is a connected set, $i = 1, 2, \dots, s$. Pixels or points in a region are connected in some predefined sense.
- (c) $R_i \cap R_j = \emptyset$ for all $i = 1, 2, \dots, s$. The regions must be disjoint.
- (d) $Q(R_i) = TRUE$ for $i = 1, 2, \dots, s$. The heterogeneous properties that must be satisfied by the pixels in a segmented region, e.g. $Q(R_i) = TRUE$ if all pixels in R_i have the same intensity level.
- (e) $Q(R_i \cup R_j) = FALSE$ for any adjacent regions R_i and R_j . The two adjacent regions R_i and R_j must be different in the sense of predicate Q .

where $Q(R_k)$ is a logical predicate defined over the points in set R_k , and \emptyset is the null set.

2.2.2 Segmentation Methods

Generally, there are two basic properties of being used by the segmentation algorithms: (1) Discontinuity; (2) Similarity.

In the first category, the image is partitioned based on abrupt changes in intensity. In other words, the boundary detection is based on local discontinuity in intensity of the boundaries of regions which are sufficiently different from each other and also from the background. The principle approach in this category is an edge-based segmentation.

In the second category, the image is partitioned into regions in which each region is homogenous according to a set of predefined criteria such as intensity level. The principle approach in this category is a region-based segmentation.

Edge-based Segmentation

Edge-based segmentation has been extensively used in grey-level image segmentation. Many techniques were proposed such as high-emphasis spatial frequency filtering and active contour. See [81, 76] for survey.

1. **Edge Detection:** The idea of the high-emphasis spatial frequency filtering is to detect points with abrupt changes in grey level. One technique to detect sharp changes in intensity is to apply the gradient operator over the image. A small window which also called a mask is convolved the image. A discontinuity which is a distinct change in grey level between two adjacent pixels is defined as *edge*.

A 3×3 mask is generally used for edge detection which is shown in Figure 2.9. The weight w_i of each cell is assigned according to the direction of the edge to be detected. Figure 2.9 (b) and (c) depicts examples of the mask for detecting vertical and horizontal edges, respectively. The value of a parameter K in the mask depends on the type of mask such as $K = 2$ for Sobel mask and $K = 1$ for Prewitt mask.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

(a)

1	K	1
0	0	0
-1	K	-1

(b)

-1	0	1
K	0	K
-1	0	1

(c)

Figure 2.9: (a) A 3×3 mask. (b) A mask for convolving the image in y -axis. (c) A mask for convolving the image in x -axis.

The limitation of the edge detection using difference operator is an ill-defined edges that are formed by a gradual change in grey level across the edge. Moreover, it is susceptible to noise because of the small size of the mask.

2. **Contour Detection:** On the other hand, an active contour technique is used to find salient image contour such as edges, lines, and subjective contours. In the model proposed by Kass *et al.*[54], the connectivity of the contours and the presence of corners affect the energy function. This method is a semi-automatic method. A user has to place an initial contour, also called *snake*, at an intended contour. The snake is actively placed to the desired contour according to the energy minimization.

The limitation of the active contour is that it lacks scalability. Since, it requires an expert user to push a snake to an intended location, segmenting an intensively large size of the image database is inefficient.

Region-based Segmentation

An image is partitioned into regions in which each region is homogenous according to a set of predefined criteria such as intensity level. Examples of this approach are histogram and thresholding, clustering, watershed, region growing, region splitting and merging, and graph-based approach.

1. **Histogram and Thresholding:** Histogram and thresholding is a widely used technique [28, 77, 59]. A histogram of grey level of an image represents the number of pixels which have a specific grey value. It can be normalized according to the ranges of the grey value.

Suppose that an image that contains a bright object and a dark background is given and our task is to segment an object. The histogram of such image has two peaks as shown in Figure 2.10 where each peak corresponds to a dark region and a bright region. From the histogram, we can choose a threshold value which is a valley between two adjacent peaks for clustering the pixels. A common problem with the histogram-based technique is the frequency of the histogram is affected by noise which causes the segmentation ambiguities. Moreover, peaks of the histogram are not always nicely separated by a valley.

The histogram is also applied to other features such as region size [30]. The pixel information such as intensity value and color components is combined with the region size. Instead of comparing the similarity using the histogram computed from the whole image, the histogram is based on a specific region such that the intensity level of the pixels is similar.

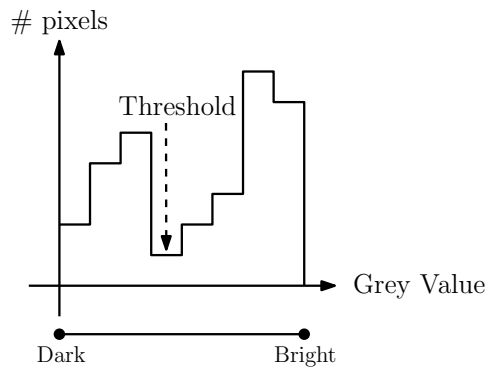


Figure 2.10: A histogram of grey level of an image.

2. **Region Growing:** A region growing technique [2, 37], an image is segmented with respect to a set of given q seeds. From a seed S_i for $1 \leq i \leq q$, a correspondence region R_i is expanded if its neighboring pixels satisfy the homogeneity criteria such as grey level. The pixels in the same region are labeled by the same label and the ones in the other regions are assigned differently.

The main advantage of this technique is the connectivity and compactness of the region. Moreover, it is robust, rapid, free of running parameters, and can be applied to large variety of images. However, this technique suffers from the problem of choosing suitable seed pixels automatically and also homogeneity criteria.

3. **Split-and-Merge:** This technique is contrast to the region growing technique. The subdivision of the image is defined by the feature homogeneity. The homogeneity can be established by determining the different between the largest and the smallest intensity levels among pixels in the quadrant. If the difference is small enough, then the region do not need to be split [78].

Initially, the whole image is considered as one region. If the feature in the region does not satisfy the predefined homogeneity criteria, the region is split into four quadrants. Each quadrant is tested in the same way, until every divided sub rectangular regions satisfy the homogeneity. A particular technique being used for splitting an image is so-called *quadtree* that is a tree in which each node has exactly four descendants. The whole image is considered as a root and the leaves of the tree are the homogeneous regions.

After the splitting phase, the region boundaries obtained tend to be squarish and artificial. Moreover, the adjacent quadrants need to be joined rather than remain separated. In this way, the merging phase is applied in order to combine the neighboring regions which are homogeneous.

2.2.3 Preliminary

Pixel grid and weight matrix

Let \mathcal{G} be an $n \times n = N$ pixel grid, and consider a family $\mathcal{F} \subset 2^{\mathcal{G}}$ of pixel regions. A pixel of \mathcal{G} is a unit square $g(i, j) = [i - 1, i] \times [j - 1, j]$ where $1 \leq i \leq n$ and $1 \leq j \leq n$.

The pixel g at position (i, j) in the grid has a real value $W(g) = W(i, j)$ called the weight of \mathcal{G} . The weight of the pixel grid is regarded as a real-valued $n \times n$ matrix $W = (W(i, j))_{1 \leq i, j \leq n}$, where the row indices are counted from top to bottom and column indices from left to right. For illustration, see Figure 2.11. For convenience, we extend W to an $(n+2) \times (n+2)$ -matrix by defining $W(0, j) = W(n+1, j) = W(j, 0) = W(j, n+1) = 0$ for all $j = 0, \dots, n+1$.

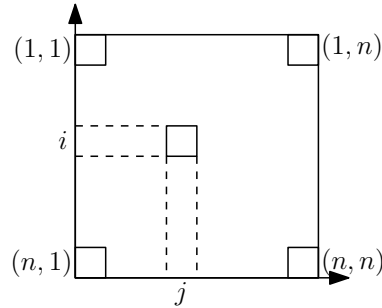


Figure 2.11: A weight matrix W of a pixel grid \mathcal{G} .

Base-monotone Region

A baseline of a pixel grid \mathcal{G} is a vertical line $x = s$ or a horizontal line $y = s$ where $0 \leq s \leq n$. For a given horizontal baseline $l : y = s$ and a pair of functions $a, b : [0, n+1] \rightarrow [0, n+1]$ with $a(i) \leq s \leq b(i)$ for all i , the corresponding *base-monotone region* is a union of column segments intersecting the baseline which is the region $\{g(i, j) : 1 \leq i \leq n, a(i) \leq j \leq b(i)\}$. In other words, it is a union of column segments intersecting the baseline. It is a special case of a x -monotone region, where the intersection of the region with each column can be any connected segment. If the i th column segment is empty, $a(i) = b(i)$, we do not assume

connectivity of the region. The vertical baseline is defined analogously. A base-monotone region with the baseline $x = 0$ (reap. $y = 0$) is often called *base x -monotone* (resp. *base y -monotone*) region.

A base-monotone region with a horizontal baseline l is subdivided into the base-monotone regions of the upper and lower half plane of l . The upper region is defined as $\{g(i, j) : 1 \leq i \leq n, s < j \leq b(i)\}$. The lower region is defined as $\{g(i, j) : 1 \leq i \leq n, a(i) < j \leq s\}$. The family of the upper and the lower base-monotone region of the half planes l are denoted by $\mathcal{U}(l)$ and $\mathcal{D}(l)$ respectively. Similarly, for a vertical baseline m , the families of the left and right half planes of m is denoted as $\mathcal{L}(m)$ and $\mathcal{R}(m)$ respectively. The families of the four regions are depicted in Figure 2.12.

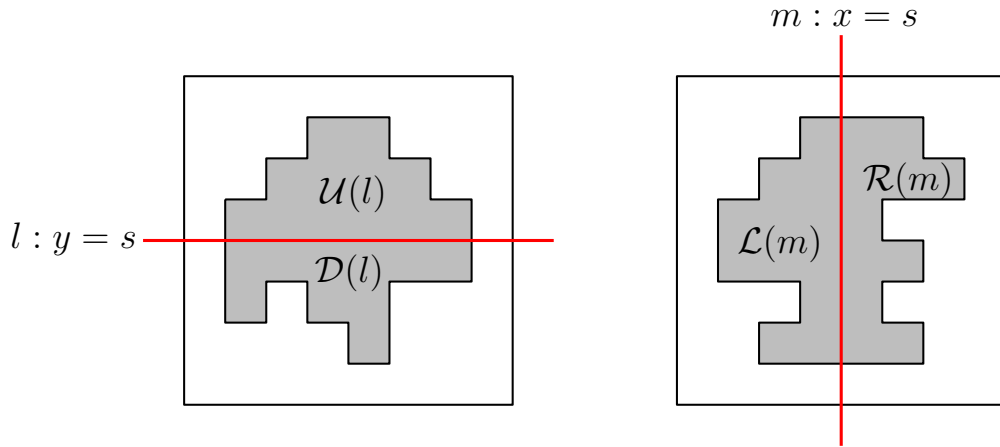


Figure 2.12: (a) The MWRP of a horizontal line. (b) The MWRP of a vertical line.

Quadtree structure

The quadtree data structure was proposed by Finkel *et al.*[39] to handle queries of the two-dimensional keys. A quadtree is a rooted tree in which every internal node has four children as shown in Figure 2.13 (b). Every node in the quadtree corresponds to a square as depicted in Figure 2.13 (a). Let us denote these quadrants as R_1 , R_2 , R_3 , and R_4 . If a node v has

children, then their corresponding squares are the four quadrants of the square of v . This implies that the squares of the leaves form a subdivision of the square of the root. This subdivision is called the *quadtree subdivision* [33].

The quadtree can be used to store different types of data such as region or point. When a predefined condition such as the maximum number of points within R_i (for $1 \leq i \leq 4$) is reached, R_i splits into four quadrants. The subdivision of each quadrant is continued until the predefined conditions satisfy.

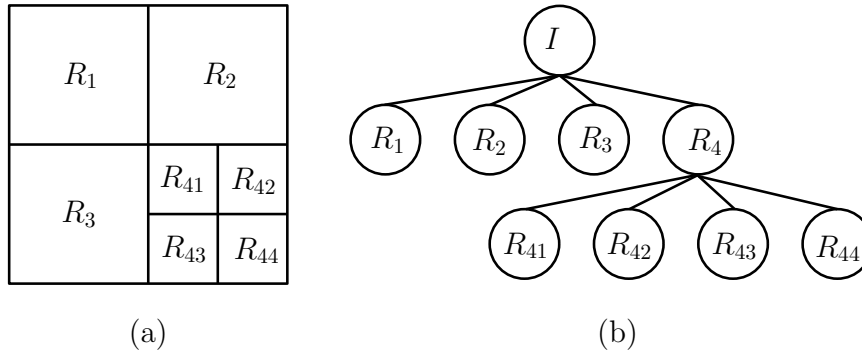


Figure 2.13: (a) Partitioned image I using quadtree structure. (b) Corresponding quadtree of the decomposition. I represents the entire image region.

2.2.4 Previous Work

Graph-Cut

Minimum cut/maximum flow algorithms [19, 56, 17, 18] on graphs, which is an optimization method, has been emerged as an increasingly useful tool for exacting or approximating an energy minimization in low-level vision.

A graph $G = \langle V, E \rangle$ is used to represent a pixel grid \mathcal{G} where V is a set of nodes v_i for $1 \leq i \leq N$ and a set of edge E where $e_{i,j} = (v_i, v_j)$ for $1 \leq i \leq N$ and $1 \leq j \leq N$. Each pixel $g \in \mathcal{G}$ is represented by a node v and the edge $e_{i,j} = (v_i, v_j)$ represents the weight

between the two nodes v_i and v_j . Two sets of pixels are given as a hard constraint in order to indicate the object \mathcal{O} and background \mathcal{B} region as shown in Figure 2.14 (a). The minimum cut/maximum flow algorithm is applied for separating the pixels into two regions such as a green line as shown in Figure 2.14 (b).

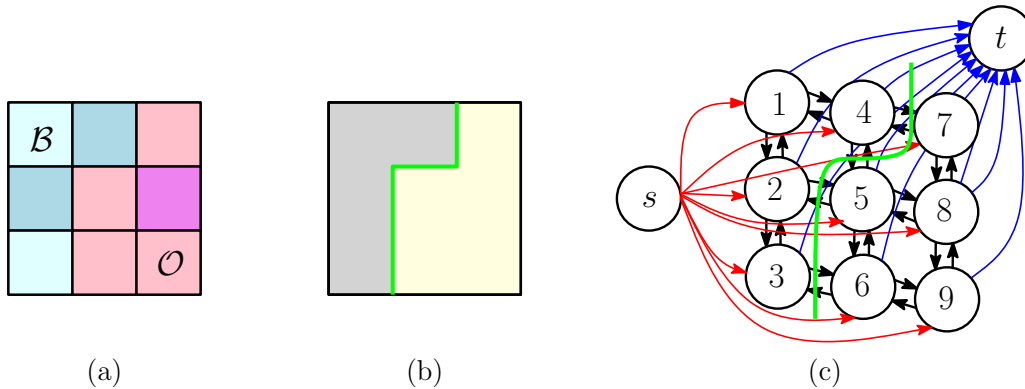


Figure 2.14: (a) An input pixel grid with \mathcal{O} and \mathcal{B} as a hard constraint. (b) The separation of pixel regions. (c) Edges of the nodes and two terminal nodes. A green line is a cut that separates the nodes into two disjoint sets.

Two types of edge \mathcal{E} are defined: (1) *n-link* denoted by \mathcal{N} connects each pair of pixel $\{p, q\}$ where $pq \notin \mathcal{O} \cup \mathcal{B}$; (2) *t-link*: connects nodes to the terminal nodes s and t which are *source* and *sink* nodes respectively. Each pixel p has two t-links that are $\{p, s\}$ and $\{p, t\}$ as shown in Figure 2.14 (c). The red and the blue edges indicate the t-link that connects to the source and the sink nodes respectively. The black edges are the n-link.

A suitable capacity is assigned to each edge in order to represent the similarity of intensity level of adjacent pixels according to the given constraints \mathcal{O} and \mathcal{B} . If the intensity of the adjacent nodes is similar, the edge capacity is assigned with a large value. Otherwise, a smaller value is assigned [16].

The minimum cut is applied on the graph in order to find a set of edges that has the minimum flow from the node s to the node t . It separates the graph into two disjoint subsets

\mathcal{S} and \mathcal{T} such that the node $s \in \mathcal{S}$ and $t \in \mathcal{T}$. A set of such edges is called a *cut* as represented in the green line in Figure 2.14 (c) which separates an image into regions.

The segmentation result using the graph-cut approach depends on the hard constraints \mathcal{O} and \mathcal{B} . An automatic method for selecting hard constraints are presented [24, 75]. However, an automatic evaluation of the accuracy of the segmentation is not an easy task [96].

Maximum Weight Base-monotone Region

Assume that the output object is denoted by S_0 and $S_1 = \mathcal{G} - S_0$ is the background. Asano *et al.* presented two methods for separating the object S_0 from the background S_1 which are: (1) using single monotone chain; and (2) using two monotone chains.

1. **Single Monotone Chain:** The segmentation is computed using one x -monotone chain in which the segmented region must be connected. Assume that the optimal region is below the separating chain, shown in blue line in Figure 2.15(a).

\mathcal{G}_m is defined to be the m -th column of \mathcal{G} , and $\mathcal{G}_{\leq m} = \cup_{i \leq m} \mathcal{G}_i$. For illustration, see Figure 2.15(a). Let $F(k, m)$ be the maximum of the optimal region $U(S_0)$ under the conditions that $|S_0| = k$, $S_0 \subseteq \mathcal{G}_{\leq m}$, and $S_0 \cap \mathcal{G}_{\leq m} \neq \emptyset$. The last condition indicates that the optimal region must be connected. Naturally, the maximum value of $U(S_0)$ is defined as $F(k) = \max_{1 \leq m \leq n} F(k, m)$. For all k and m , $F(k, m)$ can be computed using recursion.

For each $m = 1, 2, \dots, n$ and for any interval I of integers in $[1, n]$, the weight is defined as $f_m(I) = \sum_{i \in I} p_{im}$. The recursion formula of $F(k, m)$ is defined as:

$$F(k, m) = \max_{1 \leq t \leq \max\{n, k\}} \{F(k - t, m - 1) + f_m([1, t])\}$$

2. **Two Monotone Chain:** The segmentation is computed using two x -monotone chains as shown in Figure 2.15(b). The condition of the region and the notations are the same as the above method.

For $0 < k \leq N$, $F(k, t, m)$ is defined to be the maximum of $U(S_0)$ under the conditions that $|S_0| = k$, $S_0 \subseteq P_{\leq m}$, and S_0 contains the pixel (t, m) of \mathcal{G} . For every (t, m) , $F(0, t, m) = 0$. The maximum value of $U(S_0)$ is $F(k) = \max_{m=1, \dots, n} \{ \max_{t=1, \dots, n} F(k, t, m) \}$. For all k, t , and m , where $k > 0$, $F(k, t, m)$ can be computed using recursion which is defined as.

$$F(k, t, m) = \max_l \{ \max_{I \ni t, l} \{ F(k - t, l, m - 1) + f_m(I) \} \}$$

in which the maximum is taken over all the intervals I containing both t and l as shown in Figure 2.15(b).

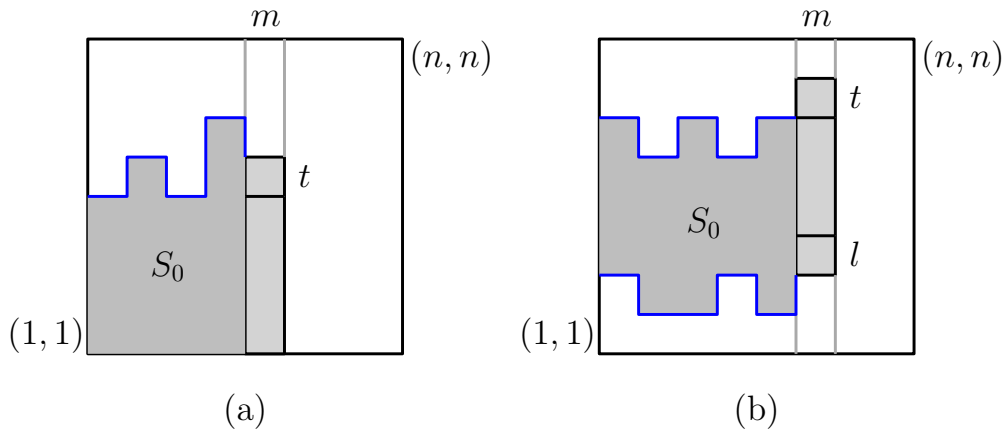


Figure 2.15: (a) Single monotone chain. (b) Two monotone chain.

Apart from the segmentation application, the x-monotone MWRP has been extended for detecting and localizing the object with arbitrary shape in an image [27]. Comparing to other methods such as rectangle bounding box, the admissible region gives a global optimal result.

2.3 Shape-based Image Retrieval

Shape is an important visual feature and also one of the basic features used for describing image content. Computing similarity of two shapes is a fundamental task in shape-based

image retrieval.

Assume that image segmentation has been done. Each object in the database is represented by a set $P = \{p_1, \dots, p_m\}$ of m arbitrary points on the boundary of a simple polygon which is extracted from an image. Many methods for shape similarity assessment have been proposed, which can be summarized into three approaches as discussed in the following subsection.

2.3.1 Geometric Pattern Matching

The similarity measurement is computed from the boundary points of the two shapes.

Hausdorff Distance

Hausdorff distance is a classical shape matching method which has been used for locating objects in an image and measuring similarity between shapes [51, 8]. The similarity between shapes is measured using point-to-point matching which every point on the shape is treated as a feature point.

Given two point sets of the two simple polygon that is $P = \{p_1, \dots, p_m\}$ and $Q = \{q_1, \dots, q_m\}$, the (bidirectional) minimum Hausdorff distance is defined as:

$$H(P, Q) = \max(h(P, Q), h(Q, P)),$$

where $h(P, Q)$ is a one-way minimum Hausdorff distance and can be defined as,

$$h(P, Q) = \max_{p \in P} \min_{q \in Q} \|p - q\|$$

$h(Q, P)$ can be defined analogously.

To measure the similarity of the two point sets, we need to find a transformation T (translation and rotation) in order to overlay one point set on the other. Therefore, the Hausdorff distance is $H(P, Q) = \max(h(P, T(Q)), h(T(Q), P))$.

In a naive method, the Hausdorff distance can be computed in $O(m^3n^3)$ time by considering all combination of points for estimating an optimal transformation. Chew *et al.*[29] proposed an algorithm that improves the time complexity for computing the Hausdorff distance under rigid motion to $O(m^3n^2 \log^2 mn)$ time for one direction and $O(m^3n^3 \log^2 mn)$ time for bidirectional minimum Hausdorff distance.

Geometric Hashing

A geometric hashing [60] is a voting-based method that utilizes a hash table in order to speed-up the algorithm. A set of points is randomly selected from both shapes. Each point gives a “vote” for a shape that are in the same hash table. The shape that receives the maximum vote is retrieved.

The transformation of each model (objects) in the database is precomputed and stored in a hash table. A basis which is a pair of points from a model shape is selected. It is aligned with coordinate $(0,0)$ and $(0,1)$ and invariant coordinates of the rest of points are computed. Figure 2.16(b) illustrates the invariant coordinates. The invariant coordinates of the transformed points are stored in a hash table which reveals in Figure 2.16(c). To compute the shape similarity, a basis of an observed, say Q , shape are selected. The invariant coordinates of Q that falls into the same bucket as P cast a vote to such model, i.e., P . The geometric hashing is able to improve the time complexity; however, the space complexity for storing the models is high.

2.3.2 Shape Descriptor

Shape similarity can be measured using a shape descriptor which represents important features of the shape. The important properties of the shape descriptor is that it must be able to effectively and perceptually represent important shape features based on either shape boundary information or boundary plus interior content. For two-dimensional shape de-

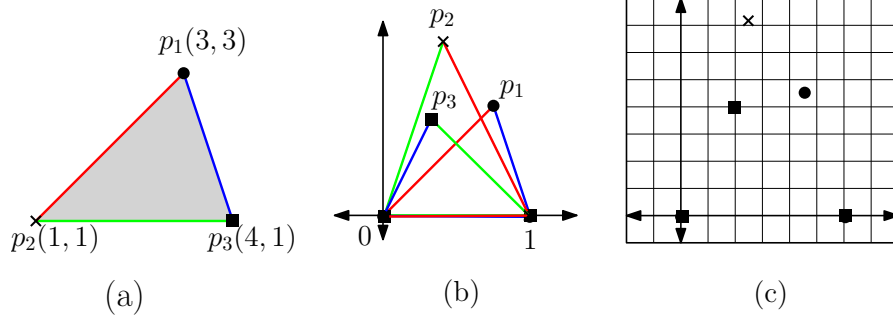


Figure 2.16: Precomputed model for voting: (a) A given polygon P . (b) Invariant coordinates and the similarity transform of P , a point p_3 is an invariant coordinate using points p_1 and p_2 as a basis. (c) Storing invariant coordinates data into bins of the hash table.

descriptor [11], it should be insensitive to the transformation, i.e. translation and rotation, and also to the scale changes of the object.

There has been a lot of effort to efficiently describe a shape using a compact representation. For recent survey, refer to [68]. The representation should be able to describe the shape as accurately as possible; on the other hand, it should be as compact as possible for indexing and retrieval. Generally, we can categorize recent shape representation using shape contour into two families: (1) Region-based shape representation and (2) Contour-based shape representation.

1. **Region-based shape descriptor:** The information of the shape are extracted from pixels. Common simple descriptor are area, medial axis [80], convexity of a shape [79, 98]. In general, such kind of descriptors is not suitable as standalone shape descriptors. For example, the two similar shapes as shown in Figure 2.17(a) and Figure 2.17(c) have totally different area of their convex hull.

Lowe [65] presented a scale invariant feature transfer (SIFT) descriptor for describing the gradient distribution in the detect region. Given a query image, the SIFT descriptor has been used as an interested region detector for retrieving the similar images. It has

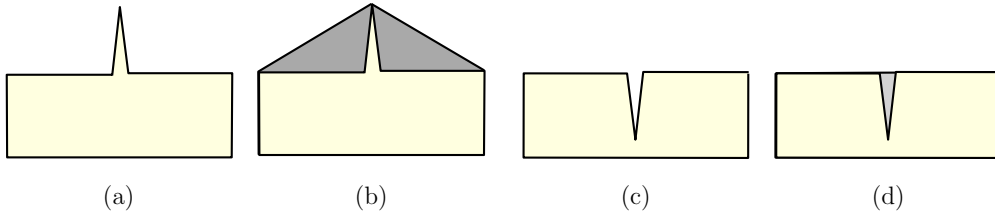


Figure 2.17: Examples of convex hull of two similar shape. (a) A shape with small peak, (b) A convex hull of shape (a), (c) A shape with small concave, (d) A convex hull of shape (c).

also been reported as the best local descriptor [67]. However, one restriction is the size of the descriptor.

2. **Contour-based shape descriptor:** The information used for describing the shape is extracted from the shape contour. It may either describe the shape by: (1) decomposing the boundary into primitive or part of the contour; or (2) using the information obtained from the whole shape. The former method is called a structural-based approach and the latter called a global-based approach.

In this research, the contour-based shape descriptor is discussed. The detail of the structural and the global-based approached will be discussed in detailed in the following.

Contour-Based Shape Descriptor using Structural approach

For the structural approach, shapes are broken down into boundary segments called *primitive*. Common methods are based on polygonal approximation, curvature decomposition and curve fitting [74]. The shape contour is encoded into a string of the general form $E = e_1, e_2, \dots, e_n$ and it is used directly as a shape description. A substring, e_i , can be an element of a polygon such as chain code [40], turning angle of the edges [88], an arc, a spline, etc. The similarity of the two shapes is directly matched using the encoded string of the two shapes.

Recently, the shape skeleton is extracted using medial axis. The skeleton is represented as a rooted tree. The similarity of the shapes is computed using edit distance to compare the trees [80]. Another method is to decompose the shape contour into convex polygons. The tree of the decomposition is used to describe the shape [64].

Contour-Based Shape Descriptor using Global approach

The global approach considers the shape feature extracted from the whole shape contour. Examples are such as Fourier descriptor [94] and statistical-based descriptors such as centroid distance [84], and distance distribution [70]. Comparing the descriptor size of the discrete approach, this approach is more compact. Moreover, it can be integrated to an indexing data structure. The advantage of this approach is that it supports very fast retrieval since it allows the comparison of two feature vectors. However, they are very inaccurate shape descriptor because the local properties of the shapes are discarded. Therefore, the global-base descriptor limits the ability to discriminate the shapes in the collections. One solution is to combine it with other descriptors to create practical shape descriptor.

2.3.3 Spatial-based approach

A spatial-based approach is used for solving the matching of the two shapes. The advantage of this method is that it can be solved two problems: (1) finding corresponding points on the two shapes; and (2) finding an alignment transformation of the two shapes. A spatial data is a descriptor attached to each point on the shape contour. The descriptor can be either a structural-based descriptor [23, 86], a global-based descriptor [14, 47, 63], or a geometric approach [7] for solving an approximate matching.

For a global-based approach, one example is a shape context descriptor [14] which locally describe the shape using the relative distribution of pairwise distance and orientations among the points on the contour. On the other hand, the correspondence are established based on

part primitives of the shape using the structure of points and lines [23].

2.3.4 Preliminary

Polygon Representation

Given an input image \mathbf{I} , assume that the image segmentation has been applied. An object is modeled as a set $P = \{p_1, \dots, p_m\}$ of m arbitrary points uniformly selected on the contour of a simple polygon extracted from \mathbf{I} .

Transformation Class

To apply the exact matching, the transformation of one point set to the other is important. There are three transformation classes: (1) Translations, (2) Rigid motion, and (3) Similarity map.

1. **Translation:** The most basic transformation is the translation. Two points determine a unique translation mapping one point to the other. Let T be a transformation and let $T^v = (v_x, v_y)$ be a translation vector. Every point of the pre-image is moved the same distance in the same direction to form the image. A mapped point $T(p) = q$ where $p \in P$ and $q \in Q$ can be computed by $T(p) = p + T^v$, where $T^v = (v_x, v_y)$. Figure 2.18 shows the translation of the original image to the right and up directions.
2. **Rigid Motion:** A rigid transformation is one in which the transformed image is translated and rotated to align with the original image. The scale of the transformed image is unchanged. A transformation space is three-dimensional which consists of translation and rotation. Two points $p_1, p_2 \in P$ and $q_1, q_2 \in Q$ and tangent $\theta(p_1)$ and $\theta(q_1)$, which is the direction of the tangent line at each point, determined a unique translation and rotation without altering its shape or size. See Figure 2.19(b). Let $T^r = (\alpha, v_x, v_y)$ be a transformation vector defined by the two points which α is a

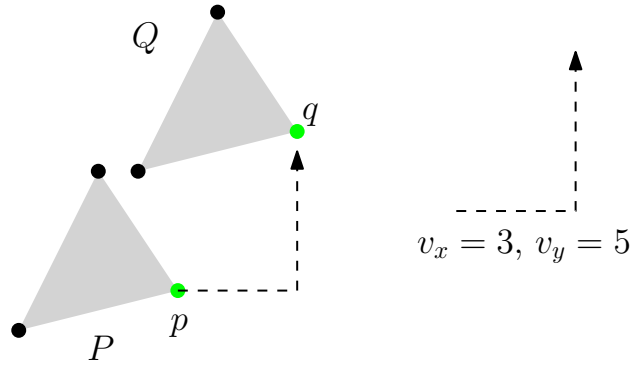


Figure 2.18: (a) An original point p and a target point q . (b) A translation vector T^v and distance (v_x, v_y) .

rotation angle such that $\theta(p_1) + \alpha = \theta(q_1)$ and $T^v = (T_x^v, T_y^v)$ is a translation vector such that $T(p_1) = q_1$. A mapped point $T(p_1) = q_1$ can be computed by $T(p_1) = Mp_1 + T^v$, where M is a rotation matrix and defined as:

$$M = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix}$$

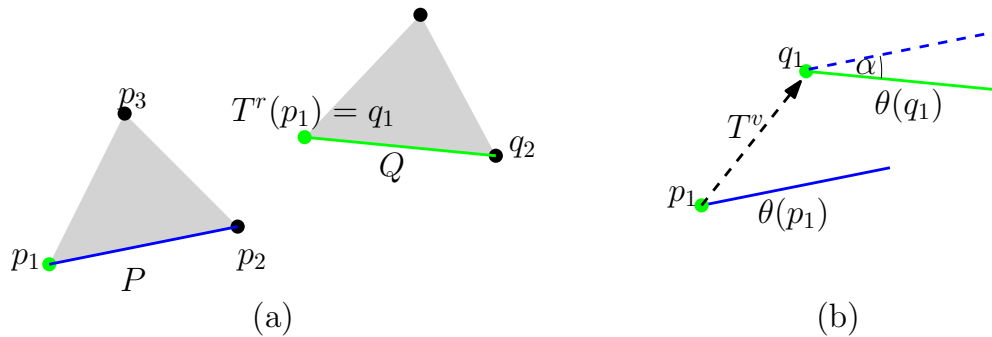


Figure 2.19: (a) An original point p_1, p_2 and its tangent $\theta(p_1)$; and a target point q_1, q_2 and its tangent $\theta(q_1)$. (b) A translation vector T^v , a rotation angle α .

3. **Similarity map:** Similar to the rigid transformation except that the scale of the transformed image is changed. A transformation space is four-dimensional. A similarity map $T^s = (\alpha, k, v_x, v_y)$ is defined by a rotation angle α , scaling factor k , and a

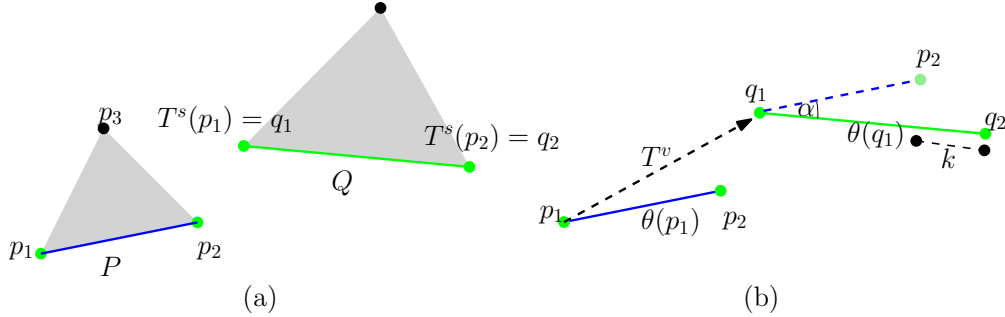


Figure 2.20: (a) An original point a and its tangent θ_a ; and a target point a' and its tangent $\theta_{a'}$. (b) A translation vector v and a rotation angle α , and a scaling factor k .

translation vector $T^v = (v_x, v_y)$. Four points $(p_1, p_2) \in P$ and $(q_1, q_2) \in Q$ determine a unique similarity transformation T^s and a scaling factor k for mapping point p_1 to p_2 and point q_1 to q_2 as depicted in Figure 2.20(b). In other words, the size of the original shape is altered according to the target. A mapped point $T^s(p_1) = q_1$ can be computed by $T^s(p_1) = Mp_1 + T^v$, where M is a rotation matrix defined as:

$$M = \begin{pmatrix} k \cos \alpha & -k \sin \alpha \\ k \sin \alpha & k \cos \alpha \end{pmatrix}$$

Modified-Hausdorff Distance

One important property of a distance measure for object similarity is that it should be tolerated to noise. However, the original Hausdorff distance is not robust to noise. A modified-Hausdorff distance was proposed for the purpose of the object matching [35].

Let $P = \{p_1, \dots, p_m\}$ and $Q = \{q_1, \dots, q_m\}$ be two arbitrary points on the boundary of simple polygon P and Q respectively. Assume that the distance between two points p and q is defined as the Euclidean distance $d(p, q) = \|p - q\|$. The modified-Hausdorff distance is defined as:

$$H_{mod}(P, Q) = \max(d(P, Q), d(Q, P)) \quad (2.1)$$

where,

$$d(P, Q) = \frac{1}{m} \sum_{p \in P} \min_{q \in Q} \|p - q\|$$

$d(Q, P)$ can be defined analogously.

Hungarian Algorithm

Hungarian algorithm is a method for solving the assignment problem [57]. The assignment problem can be described as follow: Given a set of workers, a set of jobs, and a set of score indicating the performance of each worker on each job; determine the best possible assignment of workers to jobs such that the total score is maximized. The assignment problem is also known as the maximum weight bipartite matching problem which sets of workers and jobs are represented by sets of node V_1 and V_2 of a graph G respectively, and a set of weighted E indicates the score w_{ij} between nodes v_i^1 and v_j^2 .

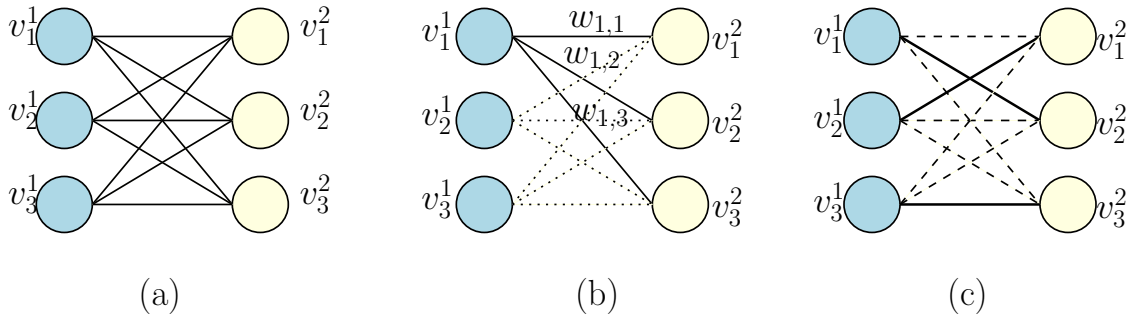


Figure 2.21: (a) A bipartite graph of a set of workers and a set of jobs. (b) A weight w_{ij} indicating the score between a worker v_i^1 and a job v_j^2 . (c) The assignment of the nodes of the two sets.

Hash table

An objective of an information retrieval application is to return objects which are similar to a given query. This problem can be derived as the nearest neighbor problem which is defined

as follow: given a collection of \mathbf{n} points, build a data structure which is able to report the data points that is closest to the given query point.

A hash table object consists of buckets that contain the elements of the collection. A bucket is a virtual subgroup of elements within the hash table, which makes searching and retrieving easier and faster. Each bucket is associated with a hash code, which is generated using a hash function and is based on the key of the element. To retrieve similar objects, an approximate k -nearest neighbor search algorithm is used to find a set of candidates that probably lies close to the query feature. One hash-based method is a Locality Sensitive Hashing (LSH) which is an approximate similarity search based on hashing. The LSH has been utilized in many computer vision application [46, 41].

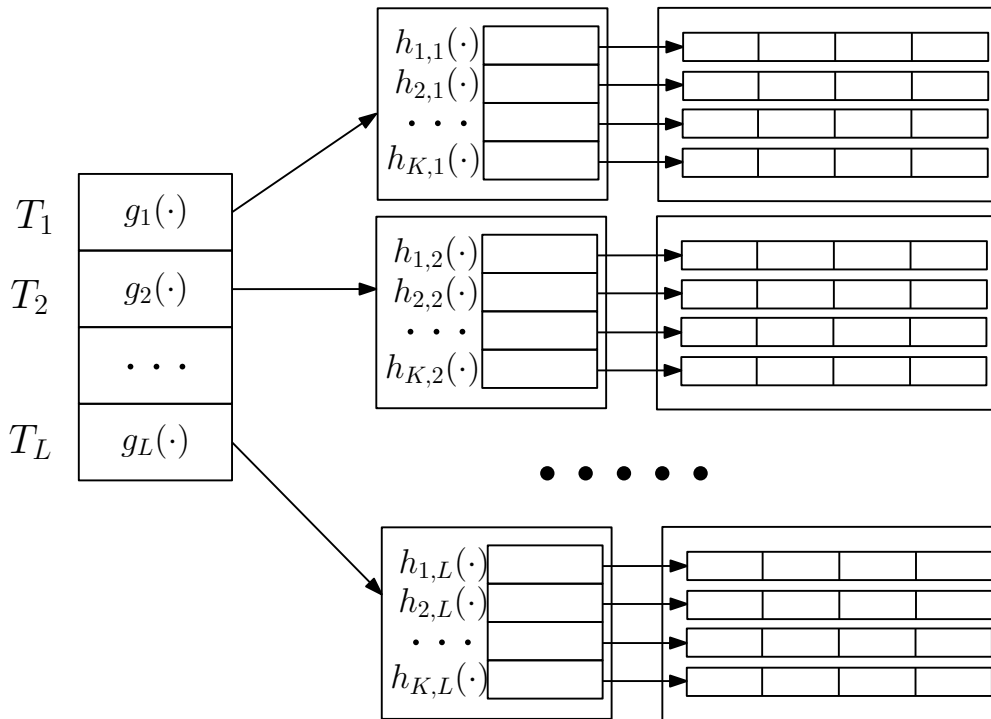


Figure 2.22: The overview of the locality sensitive hashing.

Locality Sensitive Hashing

A *Locality Sensitive Hashing* (LSH) [43] is an efficient algorithm for finding pairs of similar (or highly correlated) objects in a database without enumerating all pairs of such objects. The basic idea is to store points which are similar in the same buckets and those that are different are stored far apart. In other words, the probability of collision of the two points is closely related to the distance between them that is the larger the distance the smaller the probability. In order to increase the collision probability of the similar points, each point p in the collection are assigned to the bucket using two level hash. For illustration purpose, see Figure 2.22.

Given two positive integer L and K , the LSH function maps a point p to a bucket $g_i(p)$ of a hash table Tb_i , which is the first-level hash. Then, the content of each buckets is mapped using a standard hash functions $h_{i,1}(p), \dots, h_{i,K}(p)$. To retrieve a set of similar points of a query q , first retrieve the union of the matches from L tables which is denoted by X . Then, the distance between q and each point in X is calculated and the C nearest points are returned. The strong point of the LSH is that the collision probability and the size of the set X can be controlled using the optimal value of L and K .

2.3.5 Previous Work

Distance-base shape descriptor

Pairwise distance distribution is a contour-based descriptor that has been widely used not only for describing two-dimensional objects but also three dimension [70]. Given a set of arbitrary points on the shape contour P , we compute all $m(m-1)/2$ pairwise distances among the points as shown in Figure 2.23 (c). The distance distribution can be compressed using a statistical method, i.e., a histogram as illustrated in Figure 2.23 (d). In this way, a compact descriptor can be organized and stored in an efficient search structure. In addition

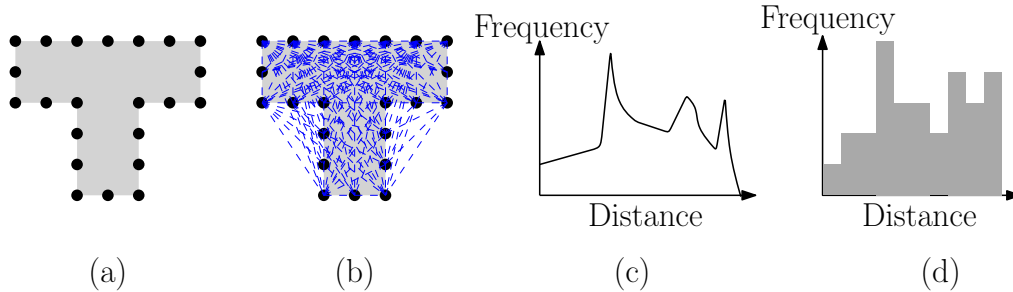


Figure 2.23: (a) A set of arbitrary points on the shape contour. (b) Pairwise distance among the points. (c) Distance distribution. (d) A histogram of a distance distribution.

to the compactness, it is invariant to rotation, transformation, and scaling.

The compactness and the computation simplicity of the pairwise distance distribution have gained an attention from researchers in order to improve the retrieval accuracy. Ip *et al.*[53] proposed a simple distance-based description for representing a three-dimensional object. An object is described using three pairwise distance histograms of the classified line segments. A line segment defining the distance is classified according to its position on the shape boundary as depicted in Figure 2.24. The detail of each class is as follow:

1. **Interior:** A line segment is within the shape boundary which is labelled as (A) in Figure 2.24.
2. **Exterior:** A line segment is outside the shape boundary which is labelled as (B) in Figure 2.24.
3. **Mixed:** A line segment is both inside and outside the shape boundary which is labelled as (C) in Figure 2.24.

Given a set of histogram, i.e., Interior, Exterior, and Mixed; of two shapes P and Q , the shapes are compared using dissimilarity measures. The dissimilarity between the two histogram can be computed using L_1 norm. The dissimilarity of the two shapes is obtained by averaging the dissimilarity between the histograms.

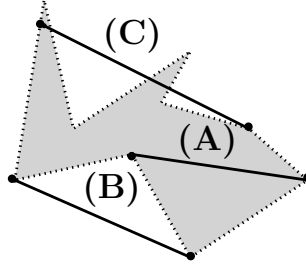


Figure 2.24: Classified distances [53] (A) Interior, (B) Exterior, and (C) Mixed

Shape Context

Shape Context (SC) [13, 14] is a descriptor that can be used for finding correspondences of the two shapes boundary. The shape is described using the relative shape information of a feature points and the boundary points. In this way, if the two shapes are similar then SC descriptor of the feature points are similar.

For each feature point, a log-polar histogram, which describes the distribution of the Euclidean distance and the tangent to the boundary points, is attached. The advantage of SC is that the feature points can be any arbitrary points on the shape contour.

Given a set of n arbitrary points on the shape contour P , the shape is described using the distribution over relative positions. The histogram which is defined to be the shape context of a point p_i is a coarse histogram h_i of the relative coordinate of the remaining $m - 1$ points p_j for $1 \leq j \leq m_1$. The histogram of the k bin for $1 \leq k \leq BIN$ is computed as $h_i(k) = \#\{p_j \neq p_i : (p_j - p_i) \in (bin(k))\}$. The log-polar space is applied in order to make to descriptor becomes more sensitive to positions of nearby sample points than farther away points. For illustration purpose, see Figure 2.25 (d) and Figure 2.25 (e).

Consider a point p_i on the shape P and a point q_j on the shape Q . The matching cost is comprised of two terms: the shape context and the local appearance. The shape context is the distance between the two histograms. Let SC_{ij} denote the cost of matching a point p_i of the shape P and a point q_j of Q . The cost SC_{ij} is defined as

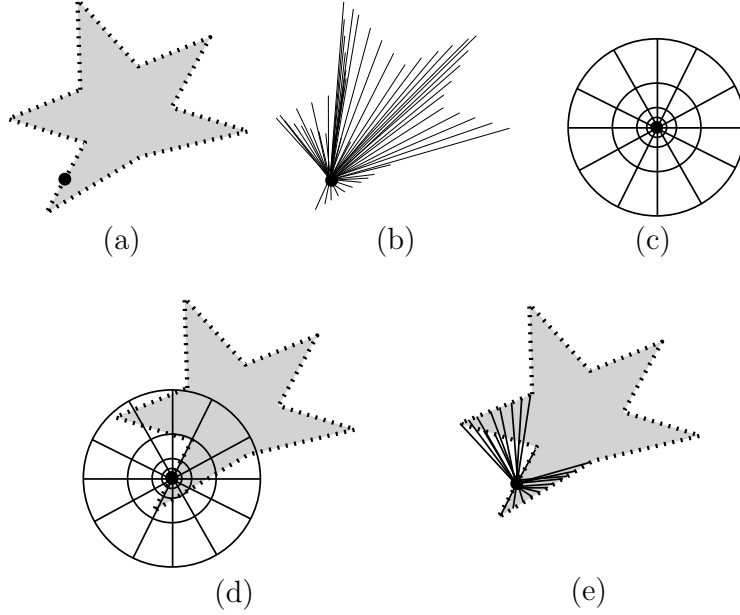


Figure 2.25: (a) A point on the boundary. (b) A line segment from a point p to every other boundary points. (c) A log-polar histogram. (d) A log-polar histogram centered at a feature point. (e) Line segments from the feature point to the points in the log-polar histogram.

$$SC_{ij} = \frac{1}{2} \sum_{k=1}^{\mathcal{K}} \frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)} \quad (2.2)$$

where $h_i(k)$ and $h_j(k)$ denote the k -bin normalized histogram at p_i and p_j respectively.

The local appearance, denoted by La , is a measure of tangent angle dissimilarity. The cost La of the two shapes is defined as

$$La_{ij} = \frac{1}{2} \left\| \begin{pmatrix} \cos(\theta_1) \\ \sin(\theta_1) \end{pmatrix} - \begin{pmatrix} \cos(\theta_2) \\ \sin(\theta_2) \end{pmatrix} \right\|$$

The combined matching cost is computed via a weighted sum. It is

$$cost(p_i, p_j) = (1 - \beta)SC_{ij} + \beta La_{ij} \quad (2.3)$$

, where $\beta = 0.1$.

Given the cost $cost(p_i, p_j)$ for $1 \leq i \leq m$ and $1 \leq j \leq m$, the correspondence of each points which is a one-to-one matching is computed using the Hungarian method. The result from the Hungarian method is a permutation $\pi(i)$ such that the sum $\sum_i cost(p_i, q_{\pi(i)})$ is a minimum.

Using the obtained correspondences of the two shapes, the shape distance is computed using a regularized Thin Plate Spline (TPS). The shape distance is estimated as the weighted sum of three terms: shape context distance, image appearance distance, and bending energy.

Belongie *et al.*[14] applied the Thin Plate Spline (TPS) for estimating a plane transformation for mapping boundary points from one shape to the other. TPS is commonly used for representing flexible coordinate transformation. It is also used for modeling changes in biological forms.

To compute the shape distance, three iterations are applied for shape context matching and a regularized TPS re-estimation. The shape distance is measured between a shape P and P as the symmetric sum of shape context matching cost over the best matching points.

The shape context cost becomes:

$$dist_{sc}(P, Q) = \frac{1}{m} \sum_{p \in P} \operatorname{argmin}_{q \in Q} cost(p, Tps(q)) + \frac{1}{m} \sum_{q \in Q} \operatorname{argmin}_{p \in P} cost(p, Tps(q)) \quad (2.4)$$

where $Tps(\cdot)$ denotes the estimated TPS shape transformation of each iteration.

Inner Distance Shape Context

One problem of using the Euclidean distance in the shape description is a homometric pair problem [85]. This is because the Euclidean distance does not consider whether or not the line segment defining the distance cross the shape boundary. Ling *et al.*[63] proposed a shape descriptor using the similar idea as the Shape Context excepts that the Euclidean distance is replaced by an inner distance.

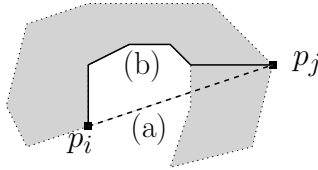


Figure 2.26: (a) A line segment from a point p to q . (b) A shortest path from a point p to q . This path is used for defining the inner distance of point p and q .

An inner distance of a pair of points $p_i, p_j \in P$ for $1 \leq i, j \leq m$ denoted as $dist_{inner}(p_i, p_j; P)$, is defined as the length of the shortest path connecting p_i and p_j within P . For comparison with the line segment, see Figure 2.26. Given a set of m boundary points, the inner distance can be computed as follow. The locations of all pair of points are checked whether or not it lies within the shape boundary. Base on the obtained location, a visibility graph is built. The inner distance of all pair of points is computed by using the shortest path algorithm.

The time complexity for checking all the line segments in $O(m)$. The visibility graph can be constructed in $O(m^3)$ time. The path of all pair of points is computed based on the shortest path algorithm which takes $O(m^3)$ time. The total time complexity for computing the inner distance of all pair of points on the boundary is $O(m^3)$ time.

Chapter 3

Image Segmentation using Maximum Weight Base-monotone Region

3.1 Introduction

We extend the idea proposed by Asano *et al.*[10] which the image segmentation problem is formulated as a combinatorial optimization problem. Given an $n \times n$ pixel grid \mathcal{G} , the proposed framework is a method to compute the segmentation problem provided that the following key problem can be solved efficiently: compute the region $R \in \mathcal{F}$ maximizing $\sum_{g \in R}(f(g) - \theta)$. Once the parameter value θ is fixed, the weight $W(g) = f(g) - \theta$ is assigned to each pixel g . In this way, the problem is transformed into the *Maximum Weight Region Problem* (MWRP). The optimal value of the parameter θ is found using hand-probing method.

Assume that the weight of each pixel, $W(g)$, is either a positive or a negative values. Figure 3.1 shows examples of the families of R which are (a) Disconnected region; (b) Connected region; and (c) x -base monotone region. Without considering the connectivity of the region, the maximum weight region can be computed by selecting only the positive-weighted pixels

as shown in Figure 3.1 (a). On the other hand, if the connectivity is given as a constraint, the problem become NP-hard [10] as shown in Figure 3.1 (b). The connected x -base monotone region computed from the red baseline in Figure 3.1 (c) can be computed in polynomial time [10].

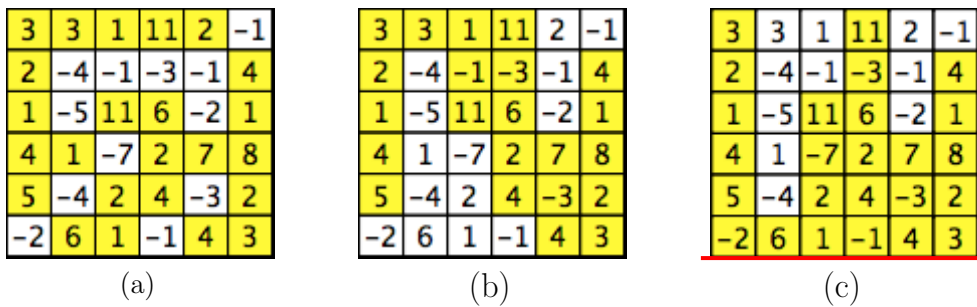


Figure 3.1: A weight matrix W of a pixel grid \mathcal{G} .

A major problem of the framework in [10] is that the method can only handle cases in which the image contains a single object with a very elementary shape as shown in Figure 3.2 (a) and Figure 3.2 (b). The segmentation result as in Figure 3.2 (d) shows that some black pixels cannot be removed by the two monotone chain algorithm.

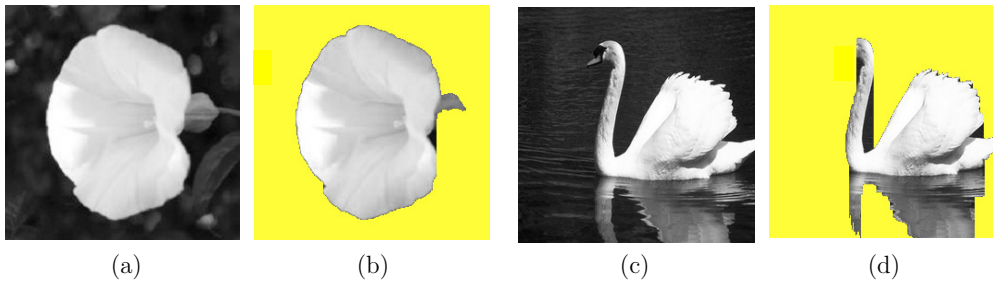


Figure 3.2: (a) A grey-scale input image. (b) The segmentation result of the image (a). (c) A grey-scale input image. (d) The segmentation result of the image (c).

3.1.1 Summary of the Result

The objective of this work is to improve the quality of the segmented results of the complex region family \mathcal{F} using MWRP. We consider the problem for solving the maximum weight region of a region $R \in \mathcal{F}$ such that $W(R) = \sum_{g \in R} W(g)$ is maximized. We present two main results.

1. An algorithm for computing a special case for base-monotone MWRT called a *room-edge* region. The room-edge MWRP is a disjoint union of four base-monotone region where each region is computed from an edge of \mathcal{G} as shown in Figure 3.3 (a).

Given an $N = n \times n$ pixel grid \mathcal{G} , the room-edge MWRT can be computed in $O(N^{1.5})$ time. We also present a framework and experimental result when applying the room-edge MWRP to an image segmentation application.

2. An algorithm for improving the segmentation when applying the room-edge MWRP is proposed. The idea is to decompose \mathcal{G} into sub grids and the MWRP is computed separately as shown in Figure 3.3 (b). To achieve a high quality of the result when segmenting an image containing a sophisticated object or many objects, the location of each baseline plays an important role for decomposing a pixel grid. We proposed two methods: (1) Grid Decomposition which can be solved in $O(N^{2.5})$ time; (2) Quad tree Decomposition which can be solved in $O(N^{3.5})$ time.

3.2 Room-Edge Maximum Weight Region

To segment a sophisticated object, the MWRP region is constructed as an aggregation of more than one elementary region. In this research, a base-monotone region is considered as an elementary region.

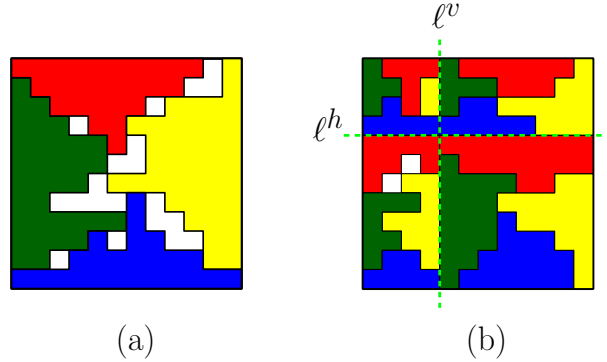


Figure 3.3: (a) A room-edge region with four coloring. (b) A room-edge region of computed from the baselines ℓ^v and ℓ^h .

We consider a special case called a *room-edge MWRP*. The room-edge MWRP is a union of two or more base-monotone regions in which the four boundary edges of \mathcal{G} are considered as the set of baselines.

The room-edge MWRP of the weight matrix W in Figure 3.1 is shown in Figure 3.4. Each region is a base-monotone region computed from the edge of the pixel grid. We abbreviate the families of base-monotone regions as \mathcal{U} , \mathcal{D} , \mathcal{L} , and \mathcal{R} . Each base-monotone region corresponds to upper, lower, left, and right half planes which are computed from bottom, top, right, and left edges of \mathcal{G} . For example, a base-monotone region \mathcal{R} computed from the left edge as a baseline and grows rightward.

A region \mathcal{S} is called *room-edge feasible* if it can be decomposed into pairwise disjoint regions \mathbf{U} , \mathbf{D} , \mathbf{L} , and \mathbf{R} such that $\mathbf{U} \in \mathcal{U}$, $\mathbf{D} \in \mathcal{D}$, $\mathbf{L} \in \mathcal{L}$, and $\mathbf{R} \in \mathcal{R}$. Let assume that each region is painted with blue, red, yellow, and green respectively. Feasible means that the region can be painted with the four colors with condition that there must be no overlapping and mixing of colors.

In an image segmentation application, we utilize the room-edge region for removing background region by using the four edges of \mathcal{G} as the baselines. The segmented result is the four-color painted room-edge region S^* which $W(S^*)$ is maximized. Figure 3.5 shows the

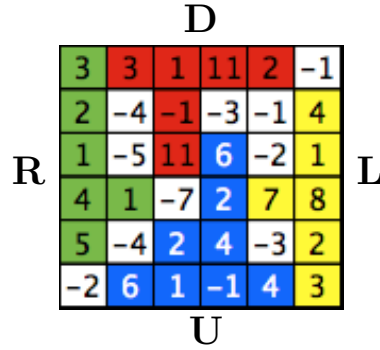


Figure 3.4: An example of a union of the base-monotone regions of the pixel grid. Each color represent a base-monotone region; i.e. **R**, **D**, **L**, and **U**; computed from each edge of the pixel grid.

segmentation using the room-edge region. From the result, the object region is a complement of the four base-monotone regions using the four edges of a pixel grid \mathcal{G} as a baseline.

3.2.1 Algorithms for the room-edge MWRP

Given a matrix W of a pixel grid \mathcal{G} where $W(g) = W(i, j)$ denotes a weight of a pixel g , an algorithm for computing room-edge MWRP of \mathcal{G} with $k \in \{2, 3, 4\}$ colors is recursively based on an algorithm for painting with $k - 1$ colors. For an explanation purpose, we utilize W in Figure 3.6 (a).

Painting with one color

Painting using one color can be computed in linear time [10]. Suppose we would like to paint a region $\mathbf{D} \in \mathcal{D}$ such that $W(\mathbf{D})$ is maximized. Note that the base-monotone region $\mathbf{D} \in \mathcal{D}$ is computed from top edge of \mathcal{G} .

To pain the base-monotone region with one color, it is suffice to consider the prefix sum

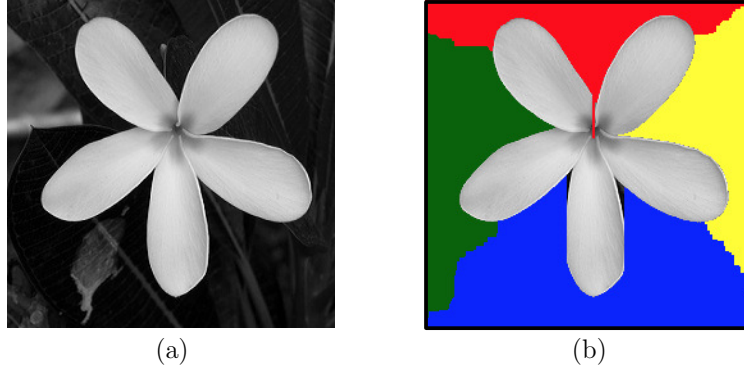


Figure 3.5: (a) Input region and baselines. (b) A base-monotone feasible region. (c) A union of four base-monotone region computed from four boundaries.

for each column j . The prefix sum of the position (s, j) is defined as:

$$\text{prefix}(s; j) = \sum_{k=0}^s W(k, j)$$

Figure 3.6 (b) shows the prefix sum of W . The maximum prefix in the j -th column to the i -th row is computed as

$$D(i, j) = \max_{0 \leq s \leq i} \text{prefix}(s; j)$$

. For illustration, see Figure 3.6 (c). The weight of the region \mathbf{D} is

$$W(\mathbf{D}) = \sum_{1 \leq j \leq n} D(n, j)$$

and the region \mathbf{D} is obtained as $\{(i, j) : D(i + 1, j) < D(n, j)\}$. The region \mathbf{D} is painted in red as shown in Figure 3.6 (c).

Painting with two color

Using two colors, an image is painted from either two opposite edges or two adjacent edges. Painting from the two opposite edges is easy since each column (resp. row) is separately considered. The optimal split which separates the region into disjoint opposite column segments takes $O(n)$ time per column.

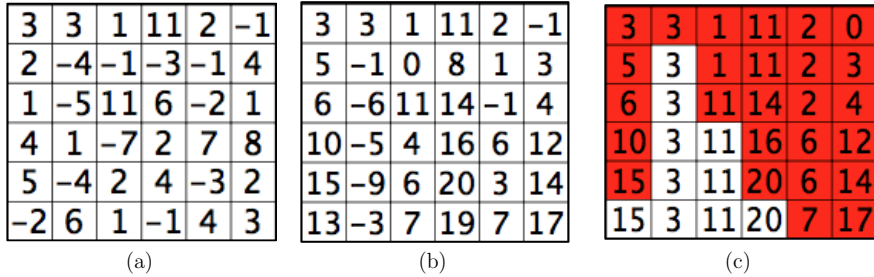


Figure 3.6: (a) A weight matrix W . (b) The prefix sum of W . (c) The array D . The region \mathbf{D} is painted in red.

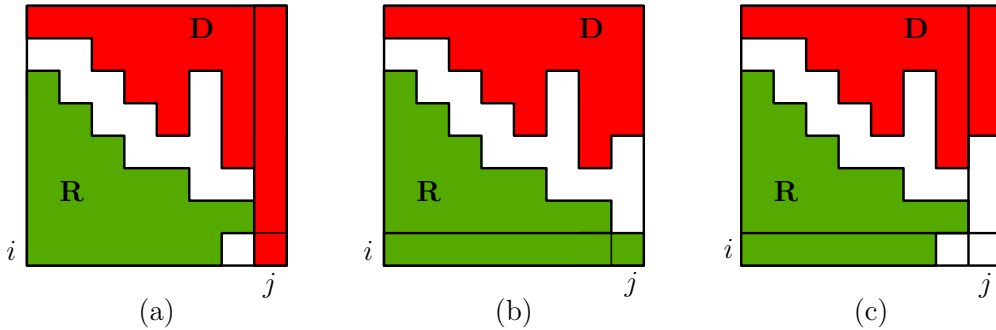


Figure 3.7: Three cases to be considered when painting with two colors: (a) Case 1. (b) Case 2. (c) Case 3.

We paint two adjacent base-monotone regions which are \mathbf{D} and \mathbf{R} . Note that $\mathbf{D} \in \mathcal{D}$ and $\mathbf{R} \in \mathcal{R}$ are computed from top and left edges respectively and each region is painted using red and green colors respectively.

The table $DR(\cdot, \cdot)$ can be obtained using dynamic programming classifying the optimal painting of $DR(i, j)$ into three cases based on the status of a pixel (i, j) as shown in Figure 3.7.

1. **Case 1:** If the pixel $(i, j) \in \mathbf{D}$ then $DR(i, j) = DR(i - 1, j) + D(i, j)$. See Figure 3.7(a).
2. **Case 2:** If the pixel $(i, j) \in \mathbf{R}$ then $DR(i, j) = DR(i - 1, j) + R(i, j)$. See Figure 3.7(b).

3. **Case 3:** The pixel (i, j) is belonged to neither \mathbf{D} nor \mathbf{R} . See Figure 3.7(c). In this way, $DR(i, j)$ is either $DR(i - 1, j) + D(i, j)$ or $DR(i - 1, j) + R(i, j)$.

From the above three cases, the optimal painting of $DR(i, j)$ is:

$$DR(i, j) = \max\{DR(i, j - 1) + D(i, j), DR(i - 1, j) + R(i, j)\}$$

Using the precomputed tables D and R for painting with one color, $DR(\cdot, \cdot)$ can be computed in linear time. Example of the DR table is shown in Figure 3.8 (a). The red color represents the case 1, the green color represents the case 2, and the white color represents the case 3. The optimal region \mathbf{DR} is obtained by backtracking using the precomputed tables D and R and $DR(\cdot, \cdot)$. Two color painting can be computed in $O(N)$ time. The result is shown in Figure 3.8 (b).

3	6	7	18	20	20
5	8	9	20	22	25
6	9	20	34	36	40
10	14	25	41	47	59
15	19	30	50	56	70
15	23	35	55	64	81

(a)

3	3	1	11	2	-1
2	-4	-1	-3	-1	4
1	-5	11	6	-2	1
4	1	-7	2	7	8
5	-4	2	4	-3	2
-2	6	1	-1	4	3

(b)

Figure 3.8: (a) Table DR obtained from the dynamic programming. The painted color indicates the three cases: red color indicates Case 1, green color indicates Case 2, and white color indicates Case3. (b) The region \mathbf{DR} .

Painting with three color

Suppose we want to compute a union of three base-monotone regions, i.e., \mathbf{UDR} , in which the regions are painted from bottom, top, and left edges of \mathcal{G} to maximizes $W(\mathbf{U}, \mathbf{D}, \mathbf{R})$. We apply precomputed tables that are UR , UD , and DR . The table $UDR(j)$ is defined as the maximum weight of coloring the pixels in the first j columns using three colors.

The $UDR(\cdot)$ table can be obtained using dynamic programming classifying the optimal painting into two cases based on the status of the j -th column as shown in Figure 3.9.

1. **Case 1:** If the j -th column of the region $UDR(j) \notin \mathbf{R}$, then the column j is painted from top and bottom edges. See, Figure 3.9(a). Thus, we have $UDR(i) = UDR(j - 1) + UD(j)$, where $UD(j)$ is the maximum weight painting of the j -th column from top and bottom.
2. **Case 2:** If the j -th column of the region $UDR(j) \in \mathbf{R}$, then the region \mathbf{R} is divided into upper half and lower half which are \mathbf{DR} and \mathbf{UR} respectively. See, Figure 3.9(b). Thus, we have $UDR(j) = \max_{0 \leq j \leq n} \{DR(i, j) + UR(i + 1, j)\}$.

According to the above two cases, the optimal painting of $UDR(j)$ is:

$$UDR(j) = \max\{UDR(j - 1) + UD(j), \max_{0 \leq j \leq n} \{DR(i, j) + UR(i + 1, j)\}\}$$

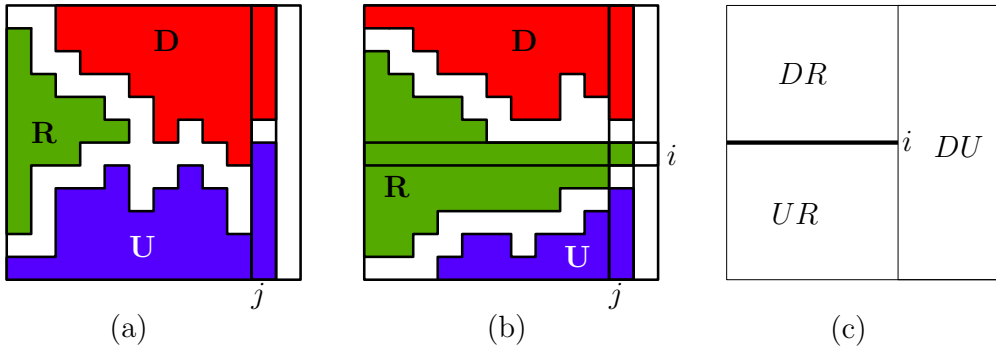


Figure 3.9: Two cases to be considered when painting with three colors: (a) Case 1. (b) Case 2. (c) Backtracking three colors

After computing $UDR(j)$ for $1 \leq j \leq n + 1$, the backtracking is started from the column $j = n$. If $UDR(j) \in \mathbf{DU}$ then only column j -th is painted. Otherwise, the rest region is separated into two regions **DR** and **UR** at row i -th as shown in Figure 3.9(c). The region

DR is painted from (i, j) and **UR** is from $(i+1, j)$, in which the two split regions are painted using the backtracking method as presented in painting with two colors.

Since we prepared the tables UR, UD , and $DR, UDR(j)$ can be computed in $O(n)$ time for each j . Hence, each new entry of UDR can be computed in $O(n)$ time by taking the maximum of the above two cases, and the table UDR is computed in $O(n^2) = O(N)$ time.

Painting with four color

Using four colors, an image is painted from four edges. There are two cases to consider:

1. **Case 1:** \mathcal{G} is decomposed into two instances of the three-color paintings using either a horizontal line or a vertical line.
2. **Case 2:** \mathcal{G} is decomposed into four instances of the two-color paintings.

For the first case, the problem is optimally decomposed using a partition line, i.e. a horizontal line and a vertical line. A horizontal line $y = i$ separates the pixel grid into two regions. Each region is solved using the precomputed three-color table, i.e., RLD and RLU . For illustration, see Figure 3.10 (a). The optimal partition line $y = i$ is a row i such that $\max_{0 \leq i \leq n} \{RLD(i) + RLU(i + 1)\}$. On the other hand, the vertical line $x = j$ can be computed using UDR and UDL as depicted in Figure 3.10 (b). The optimal the partition line $x = j$ is the column j that $\max_{0 \leq j \leq n} \{UDR(j) + UDL(j + 1)\}$.

For the second case, the problem cannot be decomposed using a single partition line because every color is blocked by another painted region as shown in Figure 3.11 (a). In this way, the pixel grid is decomposed into five rectangles where four are the instances of the two color region, i.e. **DR, DL, UR, and RL**. The decompositions are illustrated in Figure 3.11 (b) and Figure 3.11 (c). Using precomputed table DR, DL, UR , and UL , the optimal value is obtained by combining the four optimal solutions in $O(1)$ time for each possible partition. Since there are $O(n^4) = O(N^2)$ possible partition patterns, the room-edge MWRP can be solved in $O(N^2)$ time using $O(N)$ space.

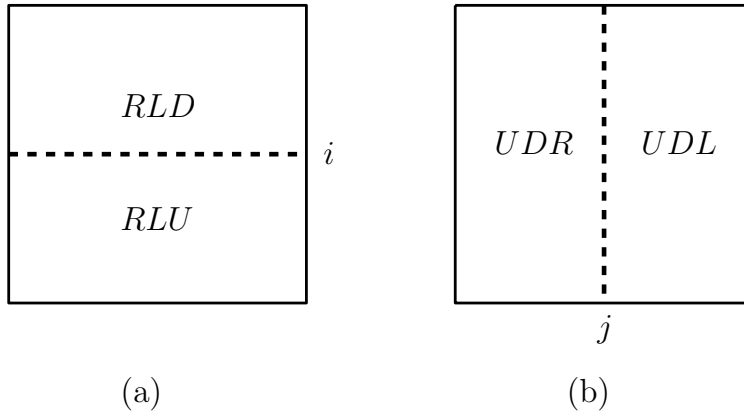


Figure 3.10: Two cases when \mathcal{G} is painting with two instances of three-color paintings

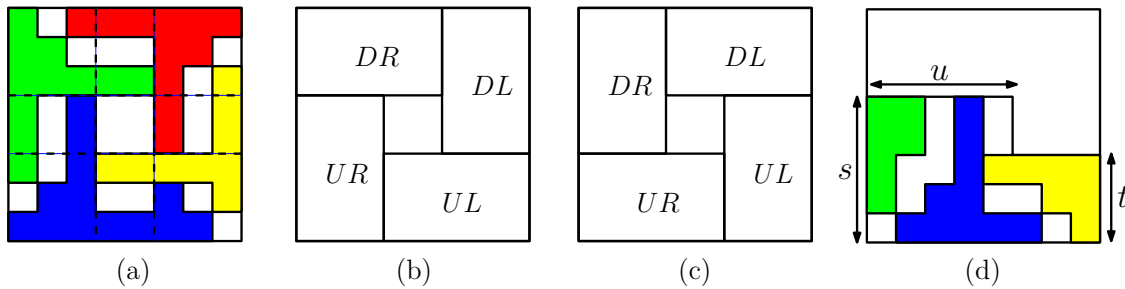


Figure 3.11: (a) A region which cannot be divided using neither a horizontal nor a vertical line. (b) The first decomposition of the four two-color regions. (c) The second decomposition of the four two-color regions. (d) A lower L -shaped region which is painted using three colors

The above time complexity can be reduced to $O(N^{1.5})$ time using another decomposition called L -shape. The L -shape region is the union of two rectangle where each contains a corner of the pixel grid. There are two types of L -shape regions that is a *upper* and a *lower*. Each region is painted with three colors. An example of a lower L -shape region containing the left-lower corner, where the tall rectangle has height s and width u , and the short rectangle has height t and width $|n - u|$ is shown in Figure 3.11 (d). The optimal three-color painting is defined by $F(s, t, u)$. There are three different cases for obtaining the maximum-weight

painting of the following recursive formula:

$$F(s, t, u) = \max\{F(s - 1, t, u) + R(s, u), UR(s, u) + UL(t, u + 1), F(s, t, u - 1)\}$$

The three cases are illustrated in Figure 3.12.

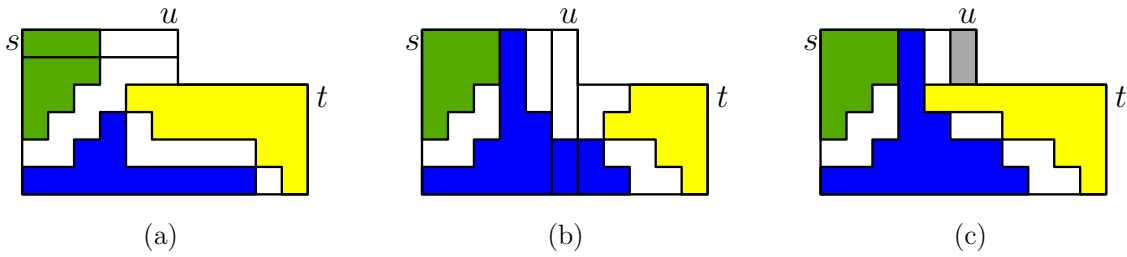


Figure 3.12: Three cases for painting using a special L -shaped region

1. **Case 1:** If row s is not penetrated by region \mathbf{U} , we have $F(s - 1, t, u) + R(s, u)$. See Figure 3.12 (a).
2. **Case 2:** If column u is not penetrated by region \mathbf{L} , we have $UR(s, u) + UL(t, u + 1)$. See Figure 3.12 (b).
3. **Case 3:** If row s is penetrated by region \mathbf{U} and column u is penetrated by region \mathbf{L} , the region is blocked by the three colors. Thus, the region between the u -th column and the s -th row remains uncolored. We have, $F(s, t, u - 1)$.

The optimal solution is decomposed into an L -shaped region at s -th row and t -th row and u -th column. The weight is $F(s, t, u)$ and two pieces UR and UL . Since UR and UL are separated at column u , each weight is $UR(s, j)$ and $UL(t, j)$ respectively.

Each entry of the table $F(\cdot, \cdot, \cdot)$ can be computed in $O(1)$ time using previous entries and the precomputed tables. The time complexity for computing the entire table F is $O(n^3) = O(N^{1.5})$. Fixing the height t , $F(s, t, u)$ is recursively computed for $t \leq s \leq n$ and $0 \leq u \leq n$. In this way, the optimal L -shaped region can be computed using $O(N)$ space.

Theorem 3.2.1. *Given a pixel grid \mathcal{G} of size $|N| = n \times n$, the room-edge MWRP can be solved in $O(N^{1.5})$ time and $O(N)$ space.*

3.2.2 Image Segmentation Application

The optimal room-edge region has an application in image segmentation. A given grey-scale $n \times n$ image shown on a pixel grid \mathcal{G} is defined as a function f on \mathcal{G} . $f(p)$ is an intensity value of range $[0, 255]$ of a pixel $g \in \mathcal{G}$. A $(n+2) \times (n+2)$ weight matrix W of \mathcal{G} is generated. The weight of the pixels at the boundary that is 0-th row and 0-th column is assign as follow: $W(0, j) = W(n+1, j) = W(j, 0) = W(j, n+1) = 0$ for all $j = 0, \dots, n+1$. For $W(i, j)$ such that $1 \leq i \leq n$ and $1 \leq j \leq n$, the weight is assigned as:

$$W(i, j) = \theta - f(g(i, j)) \quad (3.1)$$

, where $\theta = \mu\alpha$ in which θ is an average intensity level of \mathcal{G} and α is a user-defined value.

The optimal room-edge MWRP is used to a region R of the weight matrix W such that its complement denoted by $\mathcal{G} \setminus R$ is the image segmentation result.

3.2.3 Experiment

In this chapter, we consider two condition of an input image. The first one is when an image contains a clear single object. The other is when an image contains many objects and each object has a simple shape. We also demonstrate the efficiency when using the room-edge region to segment a noise image and a complicated image such as a Computer Tomography (CT).

Setting of the experiment

We used greyscale pictures of size 256×256 containing $N = n^2 = 65,536$ pixels. From the grey level $f(g)$ of the pixel $g = g(i, j)$, the weight matrix $W(\mathcal{G})$ is computed using

equation 3.1. Although a good choice of α may depend on statistic properties of the picture, $\alpha = (1.2)^{-1}$ is suitable for all pictures we tried.

Segmenting images with a clear single object

We consider an image that contains a clear single object surrounded by the background. The segmentation results are demonstrated in Figure 3.27. The result shows that the swan (and its shadow) is correctly segmented. However, the segmentation shows that the curve of the neck cannot be removed. Comparing to the graph-cut method, the background in the neck area is segmented using the giving set of constrains which are green pixels and blue pixels. From the overall result, we can see that the two methods give similar result.

We give 10% and 30% random white noise to each pixel of the input. The comparison of the segmentation result can be found in Figure 3.14. In the segmentation result where the white noise has been increased to 30%, the small curved textures appeared. We can notice from the result of the room-edge four coloring that the noise textures follow the borders of the regions. However, this texture can be removed by a post-process operation.

Segmenting images with many objects

We consider the case where there are many (possibly overlapping) objects in the input image, and each of them has a simple shape, e.g., oval-shaped. In this experiment, we apply a picture which background is not complicated. We set $\alpha = 1$ for both pictures.

Figure 3.30 shows the segmentation result. Using the room-edge MWRP cannot remove the background region. Though the segmentation from the graph-cut method is better, it requires a set of constraint pixels from each object in the picture. Figure 3.32 illustrates the segmentation of the CT image which consists of many heterogeneously shaped components. Using the room-edge MWRP solely, the segmented region is connected. However, we want to totally separate each part such as liver and kidneys from the region.

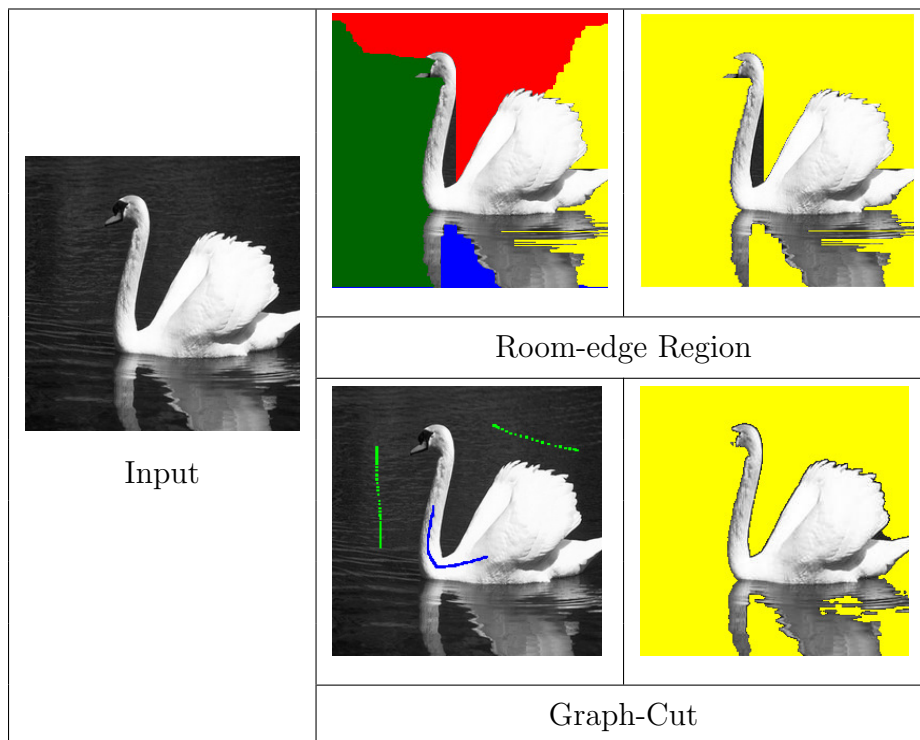


Figure 3.13: Comparison of the image segmentation of a picture of a swan using the room-edge MWRP and the graph-cut method.

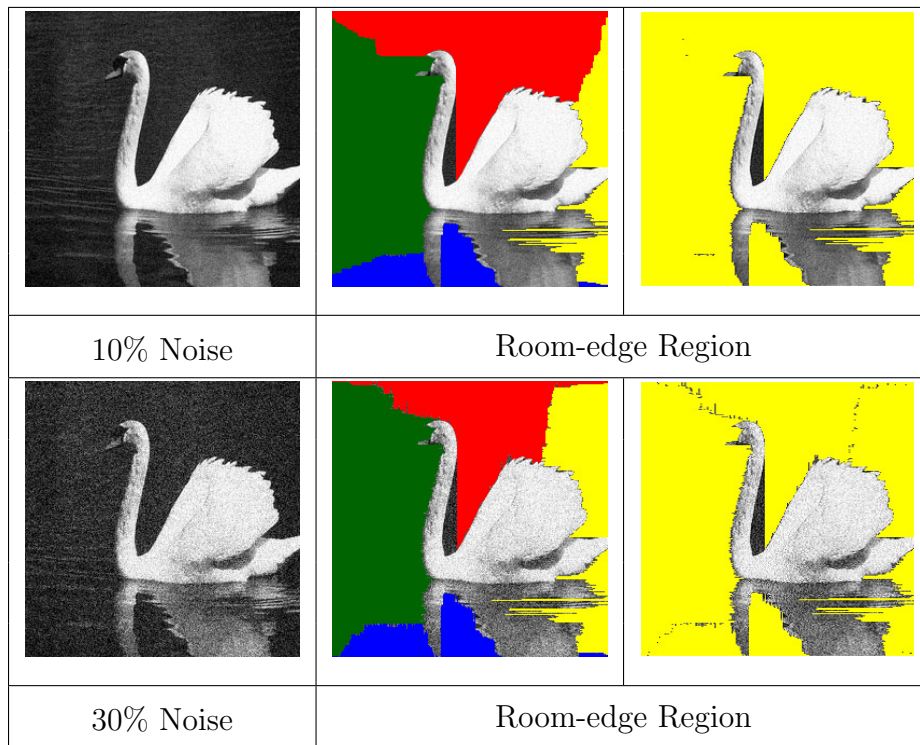


Figure 3.14: Segmentation of a picture of a swan with 10% and 30% noise using the room-edge MWRP region.

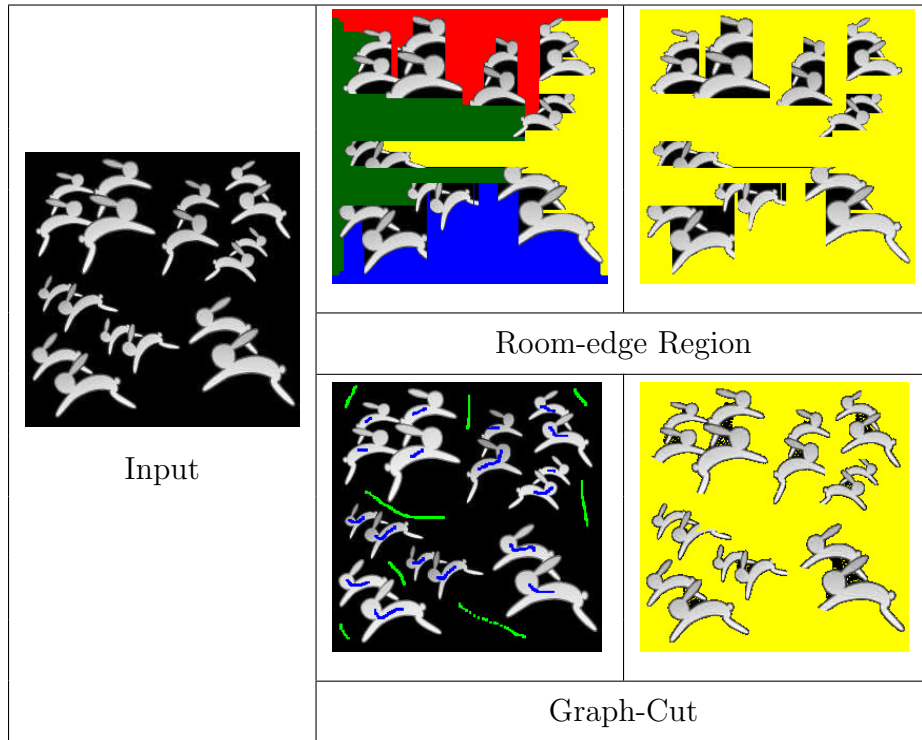


Figure 3.15: Segmentation of a picture with many objects.

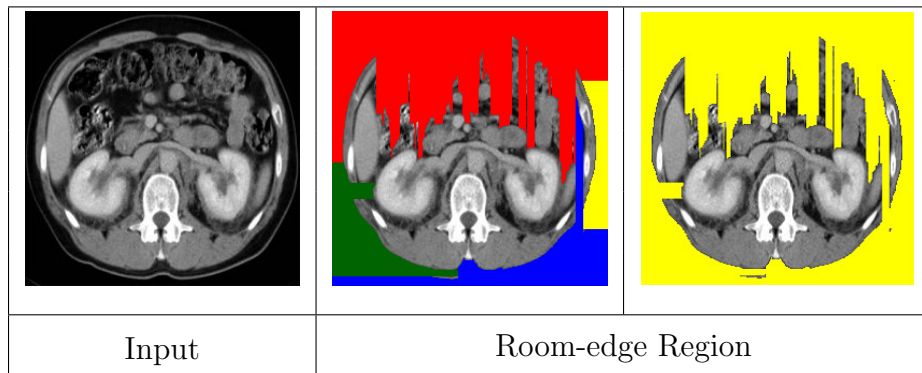


Figure 3.16: Segmentation of a CT picture using the room-edge MWRP region.

3.3 Improving Segmentation Result of Room-Edge MWRP

The segmentation of a picture which contains many objects is shown in Figure 3.17. From the result, we want to improve the quality of the segmentation when applying the room-edge region.



Figure 3.17: Segmentation of a picture with many objects using the room-edge MWRP region.

The idea is similar to how we use scissors to cut a paper: We cut the paper into rectangular pieces by a vertical and horizontal cut, and then cut the complicated figures from the boundary of each piece. In other words, we divide the pixel grid into rectangular sub grids and compute room-edge region for each sub grid. For illustration purpose, see Figure 3.18.

We consider two decomposition methods: (1) grid decomposition and (2) quadtree decomposition.

- Grid Decomposition:** The grid decomposition is obtained by cutting a pixel grid from top to bottom using k vertical partition lines and from left to right using k horizontal partition lines as shown in Figure 3.19 (a).
- Quadtree Decomposition:** The grid decomposition is obtained by cutting a pixel grid using a set of partition line segments. The placement of the partition line segments

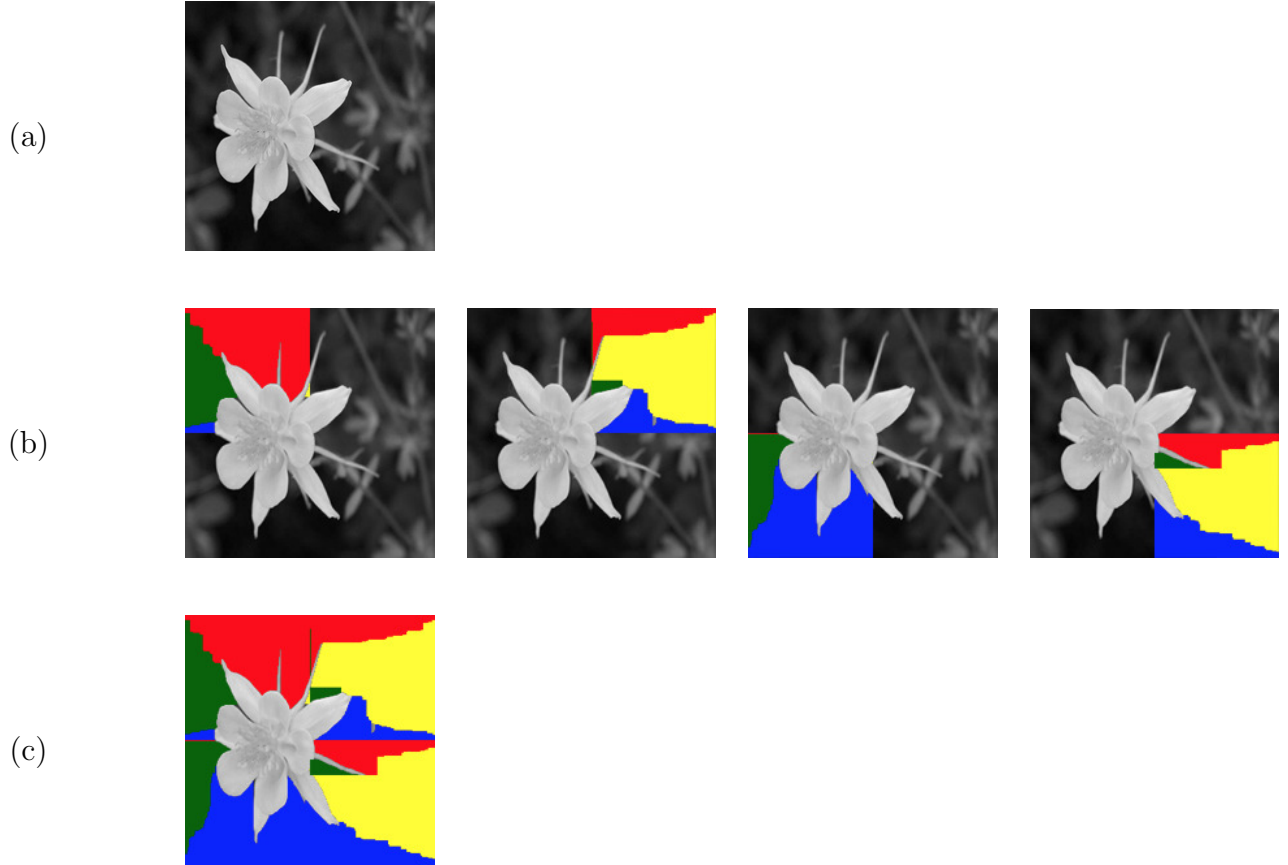


Figure 3.18: (a) An input image. (b) Segmentation results using the room-edge region of each sub grid. (c) Composition of the sub grids.

is decided using a depth d -quadtrees structure where d is given. Figure 3.19 (b) shows the decomposition of 3-quadtrees structure.

3.3.1 Grid Decomposition

The grid decomposition of a pixel grid \mathcal{G} is obtained by cutting it in two directions:

1. **Vertically:** The pixel grid is divided from the top edge to the bottom edge.

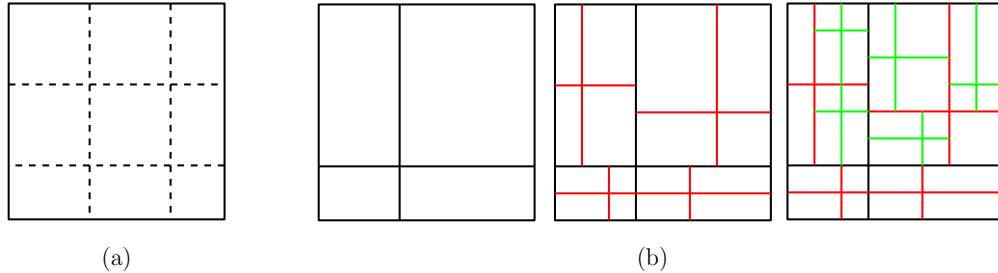


Figure 3.19: (a) A pixel grid decomposition using one vertical partition line and one horizontal partition line. (b) A pixel grid decomposition using quadtree structure using depth $d = 1, 2$ and 3 respectively.

2. **Horizontally:** The pixel grid is divided from the left edge to the right edge.

The pixel grid can be decompose several times under the condition that the number of cuts in each direction must be the same. If we cut k times, then \mathcal{G} is decomposed into $(k+1) \times (k+1)$ sub grids.

Allowing k baselines

We now consider the case in which we are given k (vertical or horizontal) baselines. We study the k base-line MWRP, where we look for the maximum-weight (possibly disconnected) region that can be decomposed into base-monotone polygons with respect to the given baselines. The arrangement of k baselines decomposes the pixel grid into $O(k^2)$ cells that are themselves rectangular sub-grids. The next lemma shows that we can solve the k base-line MWRP by solving each cell of the arrangement independently (using the room-edge MWRP approach of the previous subsection).

Lemma 3.3.1. *The union of the optimal solutions of the room-edge MWRP problem for each of the cells of the arrangement is decomposable into pairwise disjoint base-monotone regions of the given separating lines. Moreover this union attains the maximum weight of the k base-line MWRP.*

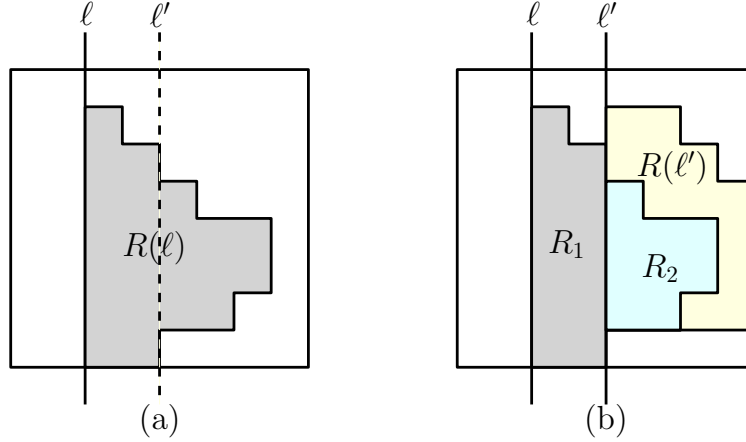


Figure 3.20: (a) A base-monotone region $R(\ell)$ with baseline ℓ . (b) A base-monotone region separated by ℓ' and a region $R(\ell') \cup R_2$.

Proof. Suppose that a base-monotone region $R(\ell)$ with baseline ℓ intersects another baseline ℓ' that is parallel to ℓ as shown in Figure 3.20 (a). The line ℓ' splits $R(\ell)$ into two regions $R_1 \cup R_2$, where R_2 is separated from ℓ by ℓ' . We replace $R(\ell)$ by R_1 and $R(\ell')$ by $R(\ell') \cup R_2$ to obtain a new set of base-monotone regions, whose union remains the same and which satisfies the additional property that the new $R(\ell)$ does no longer intersect ℓ' . For illustration purpose, see Figure 3.20 (b). Repeating this process for all such intersections, we can transform the decomposition in a way that no component intersects any other parallel baselines. Since baselines are either horizontal or vertical, all crossings between lines are orthogonal and all obtained cells will be rectangles. Thus, by partitioning the problem into separate cells and taking the union of the solutions of each of the obtained room-edge instances we get the global solution of the k base-line MWRP. \square

For each cell \mathcal{G}_t of the base-line arrangement consisting of N_t pixels, the room-edge problem inside the cell can be solved in $O(N_t^{1.5})$ time by Theorem 3.2.1. Since $\sum_{\mathcal{G}_t} O(N_t^{1.5}) = O(N^{1.5})$, the total time complexity of solving all the different room-edge problems is $O(N^{1.5})$. Thus, we have the following theorem.

Theorem 3.3.2. *The k base-line MWRP can be computed in $O(N^{1.5})$ time and $O(N)$ space.*

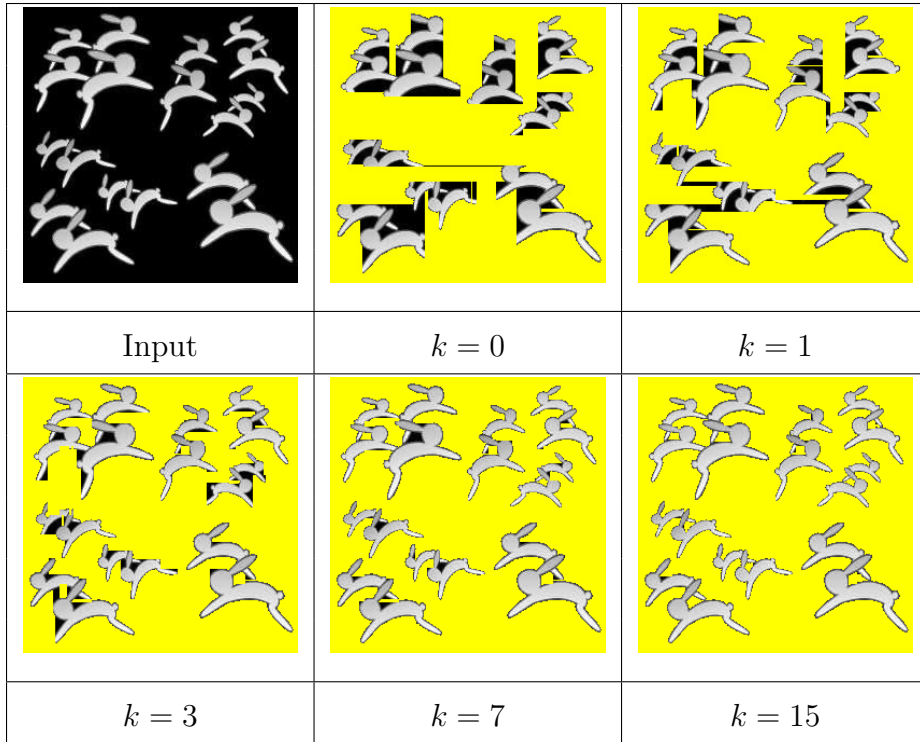


Figure 3.21: Segmentation of a image that contains many objects.

Equally partition a pixel grid using k baselines

The simplest for decomposing a pixel grid \mathcal{G} is to place the k baselines such that the space between the two baselines are equally placed. Using k vertical baselines and k horizontal baselines, the pixel grid is equally partitioned the grid into $(k + 1) \times (k + 1)$ square sub grids.

Example of the segmentation results using different number of cut k are shown in Figure 3.21. From the result, one may observe that the larger the value of k the better segmentation result obtained. Unfortunately, the segmentation result using larger value of k does not always give a better segmentation as evidence in Figure 3.22. Note that the input picture contains a complicated background. If $k = n - 1$, the optimal solution is the trivial set

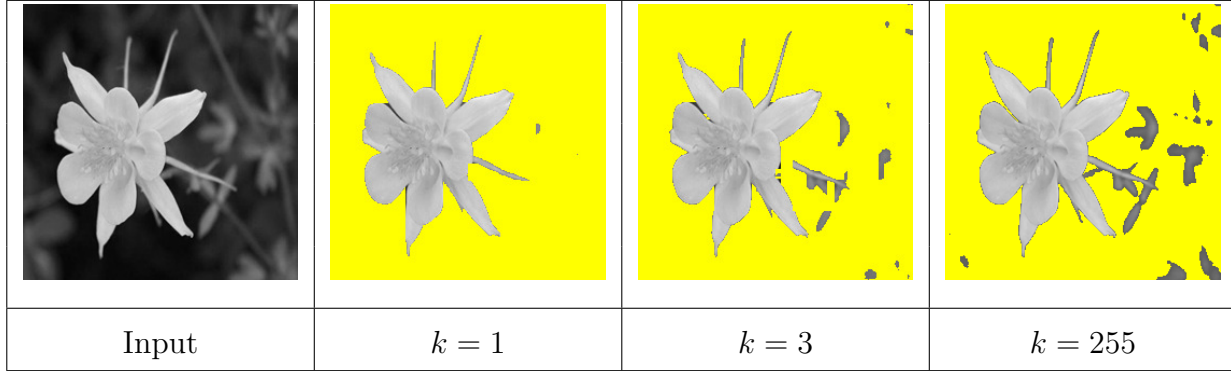


Figure 3.22: Segmentation of a flower image.

gathering all pixels with positive weights which includes some background region. Moreover, the unconnected regions affect the simplicity of the union of the optimal region.

3.3.2 Optimal location of k baselines

In addition to the number of the baselines, we learn from the segmentation result using k baselines that the quality of the segmentation result depends on the location of the baselines. If we control the space between separating lines optimally, the segmentation result is improved.

Given a pixel grid \mathcal{G} and a positive integer k , we want to find the location of the vertical baselines $\{\ell_1^v, \dots, \ell_k^v\}$, where $\ell_i^v < \ell_j^v$ and $i < j$, in \mathcal{G} that maximize the weight of an optimal base-monotone feasible region with respect to these baselines. The horizontal baselines $\{\ell_1^h, \dots, \ell_k^h\}$ is defined analogously. Unfortunately, we have the following negative result.

Theorem 3.3.3. *The baseline location problem is NP-complete in the strong sense.*

Despite of the negative result of placing k vertical baselines and k horizontal baselines, we present an approximate algorithm that computes the locations in each direction independently. In other words, we compute the optimal placement of the k vertical baselines.

Then, we utilize the obtained k vertical baselines to find the optimal k horizontal baselines separately.

Vertical k -baseline location problem

Without loss of generality, we assume that we can use the vertical baselines at position 0 and n of \mathcal{G} as baselines for free by adding new first and last columns to the grid and setting huge negative weights to the new cells.

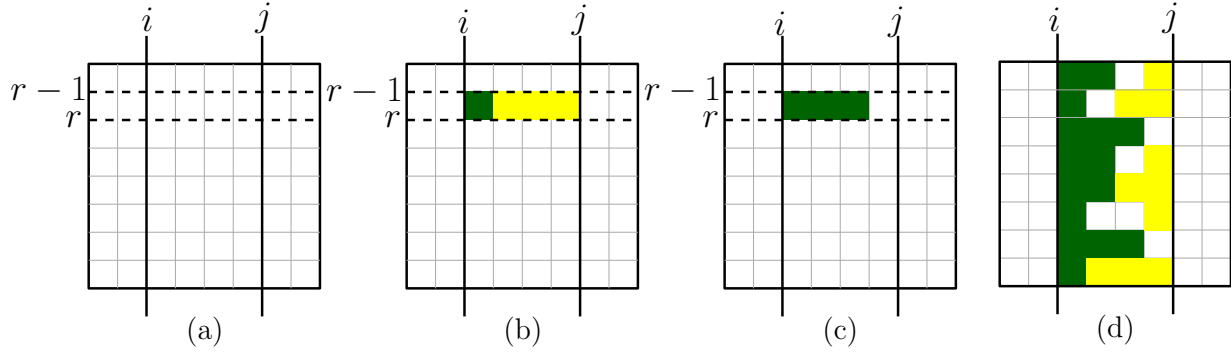


Figure 3.23: (a) A sub grid $[r - 1, r] \times [i, j]$ which is denoted by $\mathcal{G}_{i,j}^r$. (b) $\hat{A}_{i,j}^r$: The maximum weight of base-monotone region with two baselines. (c) $\hat{B}_{i,j}^r$: The maximum weight of base-monotone region with one baseline. (d) A sub grid $[0, n] \times [i, j]$ which is denoted by $\mathcal{G}_{i,j}$ and the maximum weight of base-monotone region with two baselines.

For $1 \leq r \leq n$ and $0 \leq i \leq j \leq n$, let $\mathcal{G}_{i,j}^r$ be the sub grid $[r - 1, r] \times [i, j]$ of \mathcal{G} as shown in Figure 3.23 (a). Let $\hat{A}_{i,j}^r$ be the maximum weight of base-monotone regions in $\mathcal{G}_{i,j}^r$ with the baselines at positions i and j . In other words, $\hat{A}_{i,j}^r$ can be solved using two colors painting algorithm with baselines at position i and j as depicted in Figure 3.23 (b). Similarly, let $\hat{B}_{i,j}^r$ be the maximum weight of base monotone regions in $\mathcal{G}_{i,j}^r$ with the baseline at position i only, which can be computed using one color painting as shown in Figure 3.23 (c). Clearly, we have $\hat{A}_{i,j}^r = \hat{B}_{i,j}^r = 0$ for $i = j$. For $i < j$, $\hat{A}_{i,j}^r$ and $\hat{B}_{i,j}^r$ can be obtained as follows:

$$\hat{A}_{i,j}^r = \max\{\hat{B}_{i,j-1}^r, \hat{A}_{i,j-1}^r + W(r, j)\}$$

and

$$\hat{B}_{i,j}^r = \max \left\{ \hat{B}_{i,j-1}^r, \sum_{j'=i+1}^j W(r, j') \right\}.$$

where $W(r, j)$ and $W(r, j')$ are the weight of the MWRP of the sub grid $[r-1, r] \times [i, j]$.

All entries of the three-dimensional table \hat{A} can be computed in $O(N^{1.5})$ time.

For $0 \leq i \leq j \leq n$, let $\mathcal{G}_{i,j}$ be the sub grid $[0, n] \times [i, j]$ of \mathcal{G} . Let $\hat{C}_{i,j}$ be the maximum weight of base-monotone regions in $\mathcal{G}_{i,j}$ with baselines at positions i and j as illustrated in Figure 3.23 (d). Since $\hat{C}_{i,j} = \sum_{r=1}^n \hat{A}_{i,j}^r$, all entries of the table \hat{C} can be computed in $O(N^{1.5})$ time.

Let $\hat{D}_{c,j}$ be the optimal value of the c^{th} baseline MWRP in $\mathcal{G}_{0,j}$ with respect to the vertical baseline at j , and other $c-1$ baselines in $\mathcal{G}_{0,j-1}$. It is easy to see that $\hat{D}_{1,j} = \hat{C}_{0,j}$ and that for $c \geq 2$,

$$\hat{D}_{c,j} = \max_{c-1 < i < j} (\hat{D}_{c-1,i} + \hat{C}_{i,j})$$

Recursively compute the table \hat{D} using a table \hat{C} , each $\hat{D}_{c,j}$ can be computed in $O(n)$ time.

The baseline ℓ^v is obtained while backtracking. The optimal placement of the k vertical baselines can be obtained using dynamic programming. Therefore, the table \hat{D} can be computed in $O(N^{1.5})$ time. The optimal value is

$$\max_{k < i < n} (\hat{D}_{k,i} + \hat{C}_{i,n})$$

The optimal placement of k horizontal baselines can be solved analogously. We conclude by summarizing the above result.

Theorem 3.3.4. *Given an $n \times n$ pixel grid \mathcal{G} and a positive integer k , the vertical k -baseline location problem can be solved in $O(N^{1.5})$ time.*

Horizontal k -baseline location for (fixed) given k vertical baseline locations

We show that the optimal placement of k horizontal baselines for fixed k vertical baselines can be solved in $O(N^{2.5})$ time. This algorithm is optimal with respect to fixed location of k

vertical baselines.

Similar to the vertical k -baseline location algorithm, we assume that we can use the horizontal lines at position 0 and n as baselines for free by adding new first and last rows to the grid and assigning huge negative weights to the new cells.

For $1 < c \leq k$ and $0 \leq i \leq j \leq n$, let $\mathcal{G}_{i,j}^c$ be a subgrid $[i, j] \times [\ell_{c-1}^v, \ell_c^v]$ of \mathcal{G} , where $1 \leq c \leq k$, and let (j, c) denotes the lower right pixel of the sub grid as shown in Figure 3.24 (a). Assume that $k = 2$ vertical baselines ℓ_i^v and ℓ_j^v are given, where $1 \leq i < j \leq k$ and thus $1 \leq \ell_i^v < \ell_j^v \leq n$. Let $\tilde{A}_{i,j}^c$ be the weight of a room-edge region in $\mathcal{G}_{i,j}^c$ computed using four edges of the sub grid $\mathcal{G}_{i,j}^c$ as baselines. For illustration purpose, see Figure 3.24 (b).

For each sub grid, the time complexity for computing the room-edge region is $O(N^{1.5})$. Since the number of subgrids is $O(N)$, the table \tilde{A} can be computed in $O(N^{2.5})$ time.

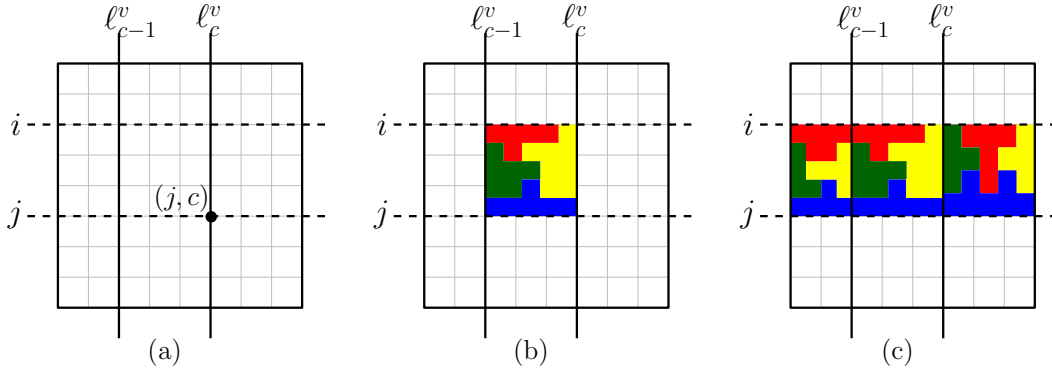


Figure 3.24: (a) A sub grid $\mathcal{G}_{i,j}^c$, also identified by (j, c) . (b) Room-edge region of a sub grid $\mathcal{G}_{i,j}^c$. (c) Room-edge regions of all subgrids on row $[i, j]$.

For $0 \leq i \leq j \leq n$, let $\tilde{C}_{i,j}$ be the sum of room-edge regions of the subgrids $\mathcal{G}_{i,j}^c$, $1 < c \leq k$ as depicted in Figure 3.24 (c). Each entry of table \tilde{C} , $\tilde{C}_{i,j}$ can be computed as:

$$\tilde{C}_{i,j} = \sum_{c=1}^k \tilde{A}_{i,j}^c$$

The optimal placement of the k horizontal baselines can be obtained using dynamic programming. We define $\mathcal{G}_{0,j}$, $1 \leq j \leq n$, differently from the last section. Let $\tilde{D}_{r,j}$ be the

optimal value of the r baselines for $1 \leq r \leq k$ in $\mathcal{G}_{0,j}$ with respect to the horizontal baseline at position j , and other $r - 1$ vertical baselines in $\mathcal{G}_{0,j}$. Clearly, we have $\tilde{D}_{1,j} = \tilde{C}_{0,j}$ and for $r \geq 2$,

$$\tilde{D}_{r,j} = \max_{r-1 < i < j} \left\{ \tilde{D}_{r-1,i} + \tilde{C}_{i,j} \right\}$$

The optimal location of k horizontal baselines can be obtained by backtracking. We conclude by summarizing the above result into the following theorem.

Theorem 3.3.5. *Given k vertical baselines, the optimal placement of k horizontal baseline problem can be solved in $O(N^{2.5})$ time and $O(N^{1.5})$ space.*

Given k horizontal baselines, the optimal placement of k vertical baseline can be defined analogously.

3.3.3 Quadtree Decomposition

The quadtree decomposition is given as follows: we divide the pixel grid G into four quadrants using one vertical and one horizontal line segments. By applying the division recursively for d times to each quadrant, a quadtree decomposition with depth d is obtained and it contains 4^d sub grids.

Our task is to optimally locate the partition line segments for a fixed depth d such that the summation of the weight of the optimal room-edge region of the sub grids is maximized. We consider the following problem: Given a grey picture I in a pixel grid \mathcal{G} , we consider a depth d quadtree decomposition D of \mathcal{G} , and find the optimal room-edge region in each sub grid. Let $W(D(\mathcal{G}))$ be the summation of those weights of room-edge regions. We want to find the decomposition D maximizing $W(D(\mathcal{G}))$.

Lemma 3.3.6. *Given a $w_q \times h_q$ subrectangular pixel grid \mathcal{G}_q in \mathcal{G} , we can compute the optimal room-edge region of \mathcal{G}_q in $O(w_q h_q^2)$ time.*

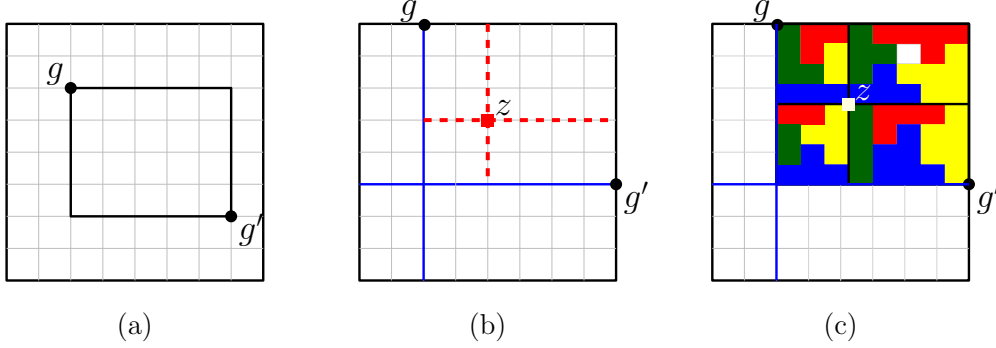


Figure 3.25: (a) A sub rectangular pixel grid $\mathcal{G}_q(g, g')$. (b) An intersection point z that subdivide a pixel grid into quadrants. (c) Room-edge regions of the sub grids decomposed at z .

We propose an algorithm for decomposing \mathcal{G} using the quadtree structure. Our algorithm is a dynamic programming algorithm consisting of d steps. For given two grid points $g = (g_x, g_y)$ and $g' = (g'_x, g'_y)$ such that $g_x < g'_x$ and $g_y < g'_y$, let $\mathcal{G}(g, g')$ be a sub rectangular pixel grid which has g and g' as a north-west and south-east corners respectively as depicted in Figure 3.25 (a). Let $W_i(\mathcal{G}(g, g'))$ be the weight of an optimal depth i -quadtree decomposition of $\mathcal{G}_q(g, g')$. To subdivide $\mathcal{G}(g, g')$ into four sub grids by a horizontal line and a vertical line, we define a point z as an intersection point such that $g < z < g'$ as shown in Figure 3.25 (b). Let $W_i(\mathcal{G}(g, g') : z)$ be the summation of the weight of the room-edge region of the four sub grids which can be computed as follow:

$$\begin{aligned}
 W_i(\mathcal{G}(g, g') : z) = & W_{i-1}(\mathcal{G}((g_x, g_y), (z_x, z_y))) + W_{i-1}(\mathcal{G}((z_x, g_y), (g'_x, z_y))) + \\
 & W_{i-1}(\mathcal{G}((g_x, z_y), (z_x, g'_y))) + W_{i-1}(\mathcal{G}((z_x, z_y), (g'_x, g'_y)))
 \end{aligned}$$

Thus, the decomposition is a point z such that the summation of the weight of the four sub grids is maximized. We have,

$$W_i(\mathcal{G}(g, g')) = \max_{z: g < z < g'} W_i(\mathcal{G}(g, g') : z)$$

Given d depth quadtree decomposition, the weight of an optimal solution for the depth d -quadtree decomposition is $W_d(\mathcal{G})$. For $i = 0$, we compute the maximum weight room-edge

region in each $\mathcal{G}_q(g, g')$. For fixed g , there are $O(n^2)$ entries. Thus, there are $O(n^4)$ entries in \mathcal{G} which can be solved in $O(n^7)$ time using $O(n^3)$ -time algorithm. For $i \geq 2$, each entry in $\mathcal{G}_i(g, g')$ can be obtained in $O(n^2)$ time using the computed value in level $i - 1$ and all entries can be computed in $O(n^6)$ time. Since $d \leq n$, the total time complexity is $O(n^7)$. We conclude by summarizing the above result into the following theorem.

Theorem 3.3.7. *Given a $n \times n$ pixel grid \mathcal{G} and a d -depth quadtree decomposition, the decomposition D maximizing $W(D(\mathcal{G}))$ can be computed in $O(n^7) = O(N^{3.5})$ time.*

Casual Formulation of the Quadtree Decomposition

Since $O(N^{3.5})$ time complexity prohibits the implementation of the algorithm for a fine resolution digital picture (in practice, $n = 10^6$ or more), the casual formulation of the $O(N^{3.5})$ algorithm is presented to support the proposed algorithm practically.

Given a $n \times n$ pixel grid, a depth d , and y guidelines, we divide \mathcal{G} equally into $y \times y$ grid which is denoted by Y . The grid is divided using the d -quadtree decomposition algorithm. For illustration purpose, see Figure 3.26. In this formulation, we restrict the location a point r which is an intersection point of a horizontal line segment and a vertical line segment to be placed according to the location of the y guidelines. Thus, the weight of an optimal solution for the depth d -quadtree decomposition is $W_d(Y)$. For fixed p , there are $O(m^2)$ entries of $W(Y(g, g'))$ in which there are $O(m^4)$ entries.

Theorem 3.3.8. *Given a $n \times n$ pixel grid \mathcal{G} and a positive integer y , let Y be a pixel grid of \mathcal{G} that is equally divided into $(y + 1) \times (y + 1)$ grid. Since $y < n$, the decomposition D maximizing $W(D(Y))$ can be computed in $O(y^4 n^3)$ time.*

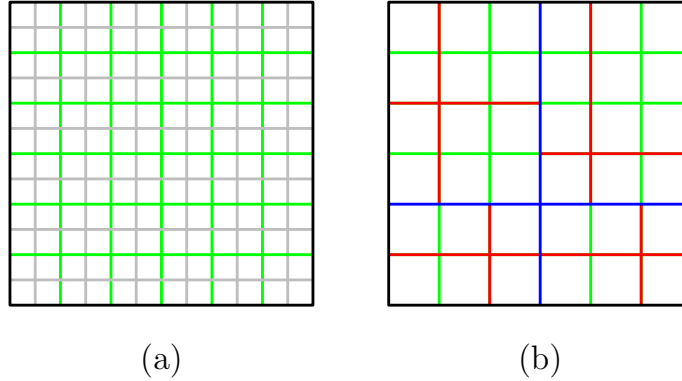


Figure 3.26: (a) A $y \times y$ pixel grid Y where $y = 5$. (b) A quadtree decomposition of Y .

3.4 Experimental Result

We show the experimental result of the proposed three decomposition methods: (1) k -baseline grid decomposition; (2) the optimal k -baseline grid decomposition; and (3) d -depth quadtree decomposition. We also shows the comparison of the propose methods with the graph-cut method.

3.4.1 Setting of the Experiment

In the experimental result using k -baselines method, we used greyscale pictures of size 256×256 containing $N = n^2 = 65,536$ pixels. For the segmentation using the optimal k -baseline location and the d -depth quadtree decomposition, pictures of size is 128×128 containing $N = n^2 = 16,384$ pixels.

The weight matrix $W(\mathcal{G})$ of a pixel grid \mathcal{G} is computed as follow: For $W(i, j)$ such that $1 \leq i \leq n$ and $1 \leq j \leq n$, the weight is assigned as:

$$W(i, j) = \theta - f(g(i, j))$$

where $f(g)$ is a gray level of the pixel $g = g(i, j)$, $\theta = \mu\alpha$ which μ is an average intensity level of \mathcal{G} , and α is a user-defined value. In this experiment, we set $\alpha = (1.2)^{-1}$, 1 and $(1.2)^2$.

3.4.2 Grid Decomposition: k -Baselines

Computation Time

Our algorithms are implemented using C^\sharp language, and executed on a standard desktop PC (3.16 GHz processor and 3GB memory). We applied our $O(N^{1.5}) = O(n^3)$ time algorithm for the room-edge problem. If we use k separating horizontal lines and k vertical lines, the time complexity is reduced to $O(\frac{n^3}{k})$ since we solve $(k+1)^2$ different problem instances (i.e. cells of the $k \times k$ grid) of size $n/(k+1) \times n/(k+1)$ each. The running time for the room-edge problem (i.e. $k = 0$) is 0.146, 1.36, and 16.0 seconds if $n = 64, 128$ and 256 , respectively on a picture of swan given in Figure 3.27. We note that the running times for the other images were very similar. The dependency on k (recall that n is fixed to be 256) is shown in Table 3.1, and it supports the theoretical analysis.

Table 3.1: The computation time of room-edge MWRP using $\{0, 1, 2, 7, 15\}$ baselines.

Value of k	0	1	3	7	15
Computation time T_k (sec)	16.0	5.72	2.12	1.40	0.681
Ratio of T_k to $T_0/(k+1)$	1	0.715	0.53	0.7	0.681

Thus, provided that the algorithm outputs a good image segmentation, it is faster and advantageous to use a larger k . However, we will see that if k is too large, the output quality is highly affected by noise and texture in the input pictures.

Result

The segmentation results of an image that contains a clear single object surrounded by the background are shown in Figure 3.27. The comparison of the segmentation result using the number of baselines as $k = 0, 1, 3$ and 7 are demonstrated in Figure 3.27. The result shows that the swan (and its shadow) is correctly segmented. We can clearly see that the curve of

the neck is removed when decomposing the pixel grid.

The comparison of the segmentation result of pictures with 10% and 30% noise are shown in Figure 3.28. In the segmentation result where the white noise has been increased to 30%, it is easy to see that the small curved textures appear. From the result of the room-edge four coloring, we can notice the noise textures follow the borders of the regions. However, this texture can be removed by a post-process operation.

Figure 3.29 shows an image with many goldfish and the resulting segmentations for $k = 3, 7$ and 15. From the result, we notice that a smaller k gives a poorer segmentation. Note that some of the goldfish are hard to see, even for a human being. If we set $k = 7$ or 15, all goldfish are captured. However, some components which are connected to each other, the algorithm would segment the two objects as one. The 10% white noise is added to the input and the segmentation with $k = 7$ is shown in Figure 3.29. We can see that the segmentation is not highly affected by noise. Figure 3.30 shows segmentation result of rabbits which some are overlapped. The result shows that the segmentation with $k = 15$ removes all background pixel while some small background region pixel cannot be removed using $k = 16$.

Figure 3.31 shows the segmentation result of a picture which is difficult to segment even for a human being. We applied the k -baseline method using $k = 3, 7$, and 15. The results show that the larger the value of k , the better segmentation is obtained. However, there is no big difference between $k = 7$ and $k = 15$. The lower half of Figure 3.31 shows the same figure, where 30% white noise has been added to the input. Notice that the effect of noise is visually milder than in Figure 3.28. This is caused by the fact that the objects of the image make texture curves, hence they are easier to detect.

The segmentation of an image that consists of many heterogeneously shaped components, such as the case of a computer tomography (CT) image of a human body is depicted in Figure 3.32. We set $\alpha = 1$, and execute our algorithm using $k = 1, 3, 7$, and 15 baselines. For large values of k , i.e., 7 or 15, all important parts to be examined by a doctor are detected. On the other hand, the smaller values of k give a poorer segmentation.

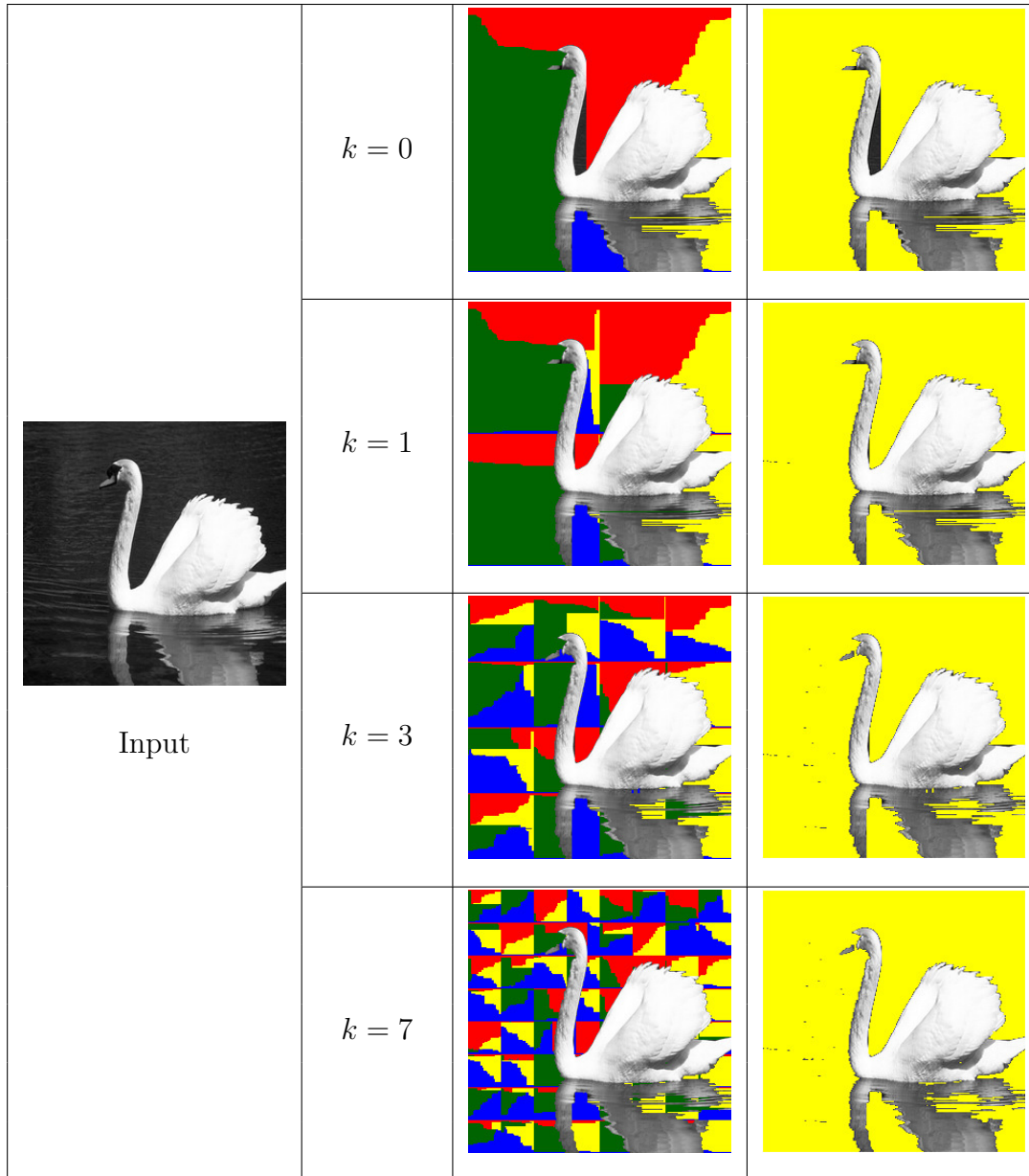


Figure 3.27: Image segmentation of a picture of a swan using k -baseline method where $k = 0, 1, 3,$ and 7 .

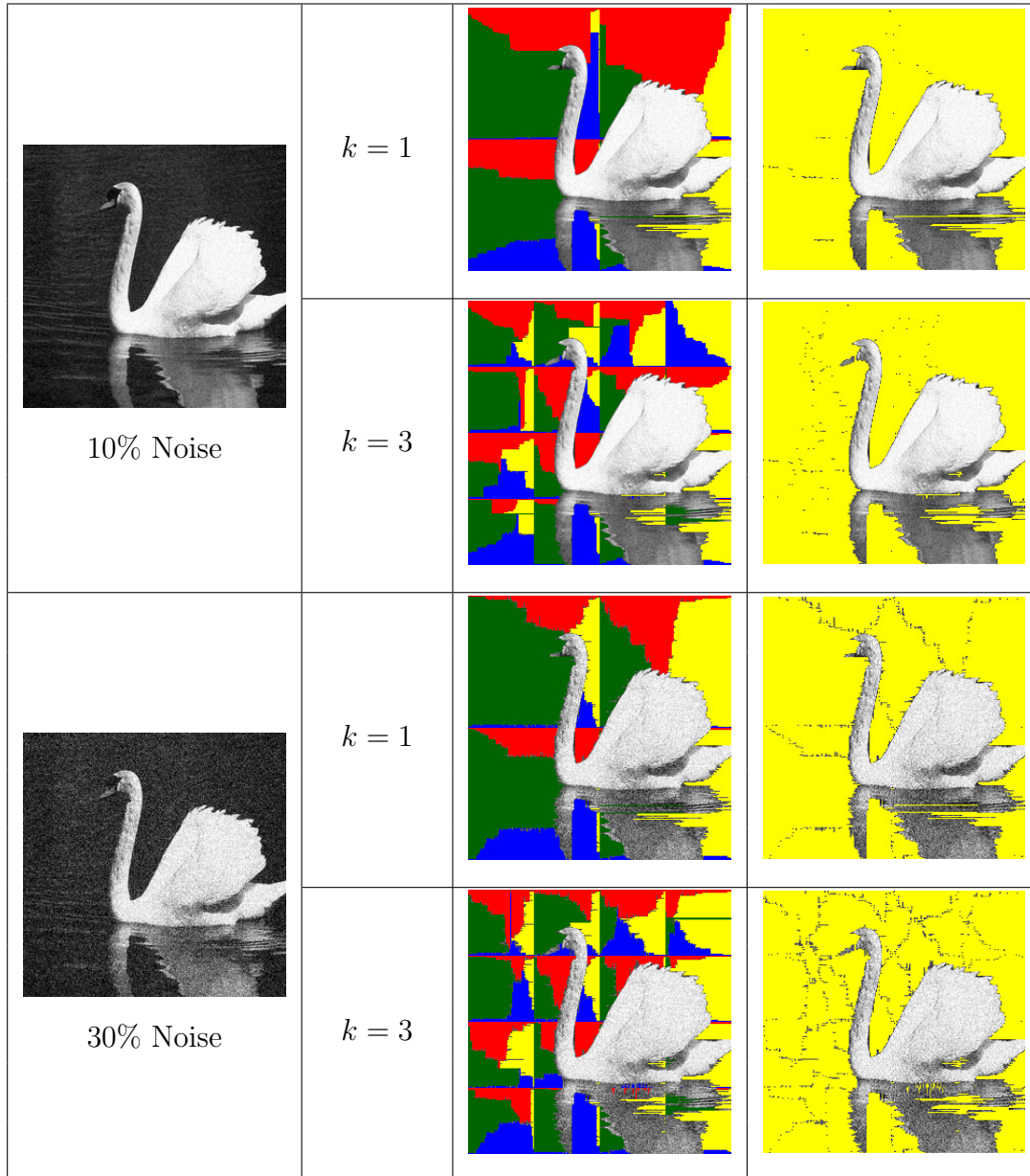


Figure 3.28: Segmentation of a picture of a swan with 10% and 30% noise k -baseline method where $k = 1$, and 3.

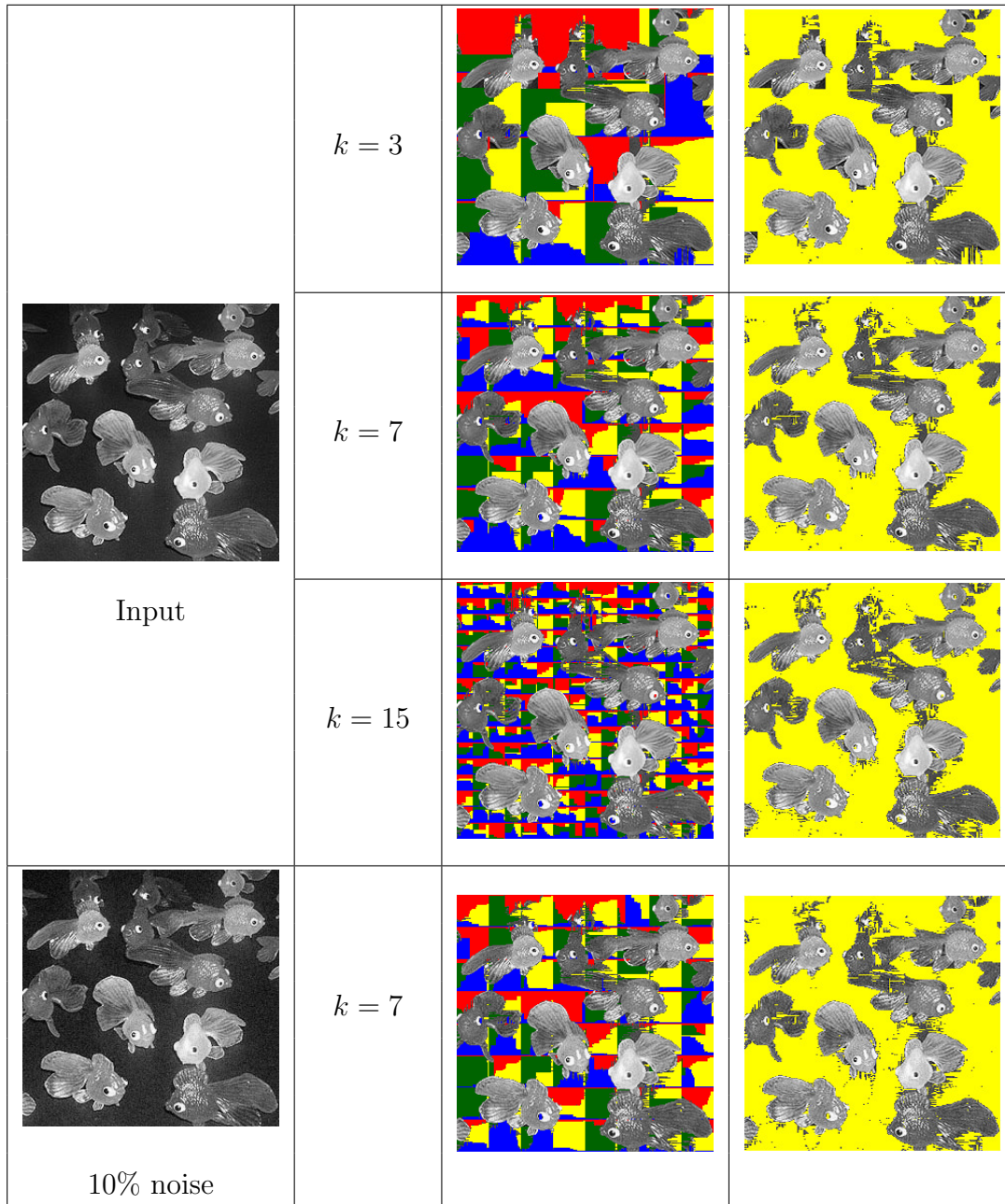


Figure 3.29: Segmentation of a picture of goldfish with k -baseline method where $k = 3, 7,$ and 15 and a picture with 10% noise with $k = 7$.

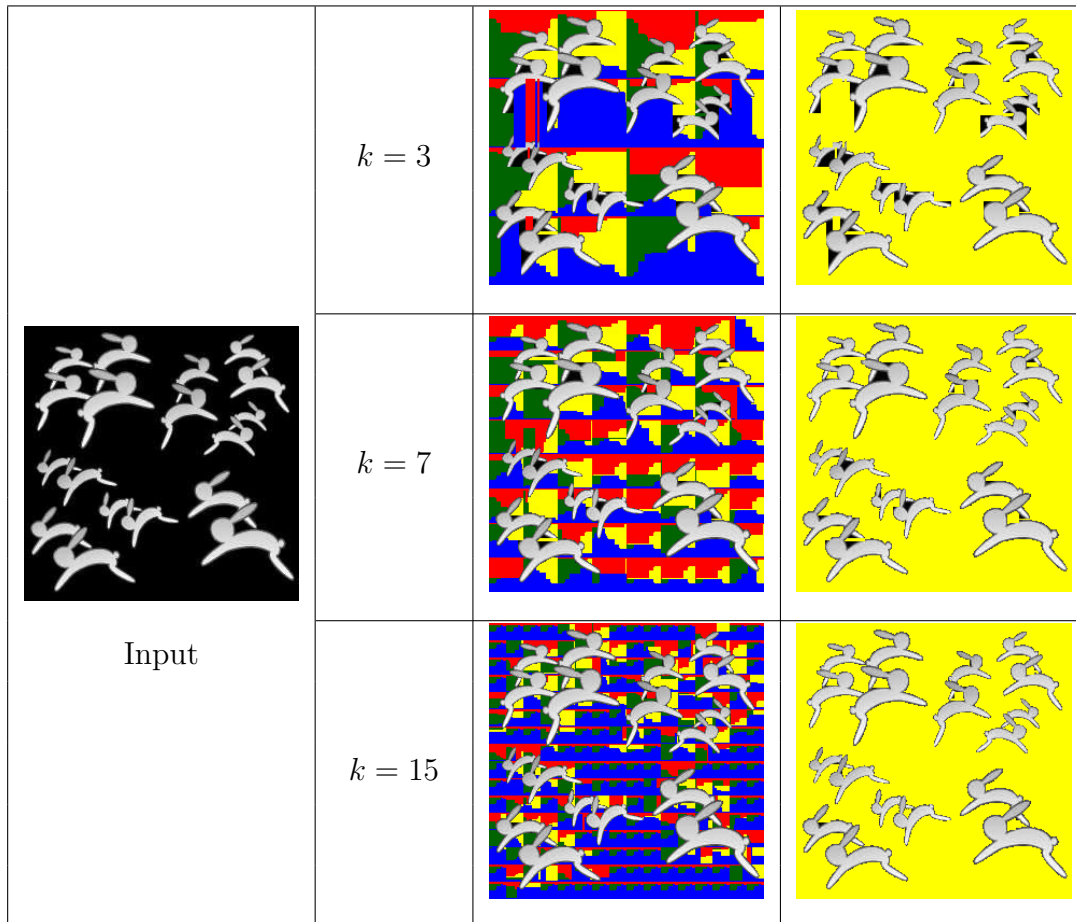


Figure 3.30: Segmentation of a picture of rabbits using k -baseline method where with $k = 3, 7,$ and 15 .

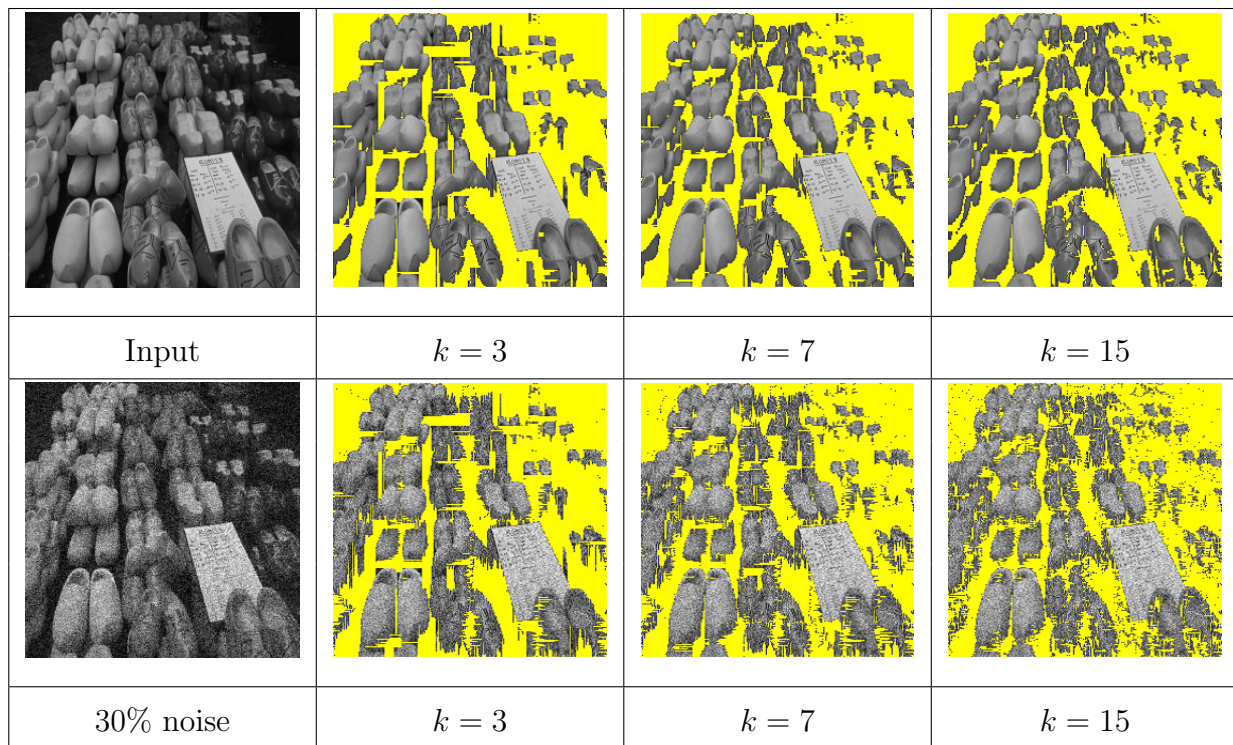
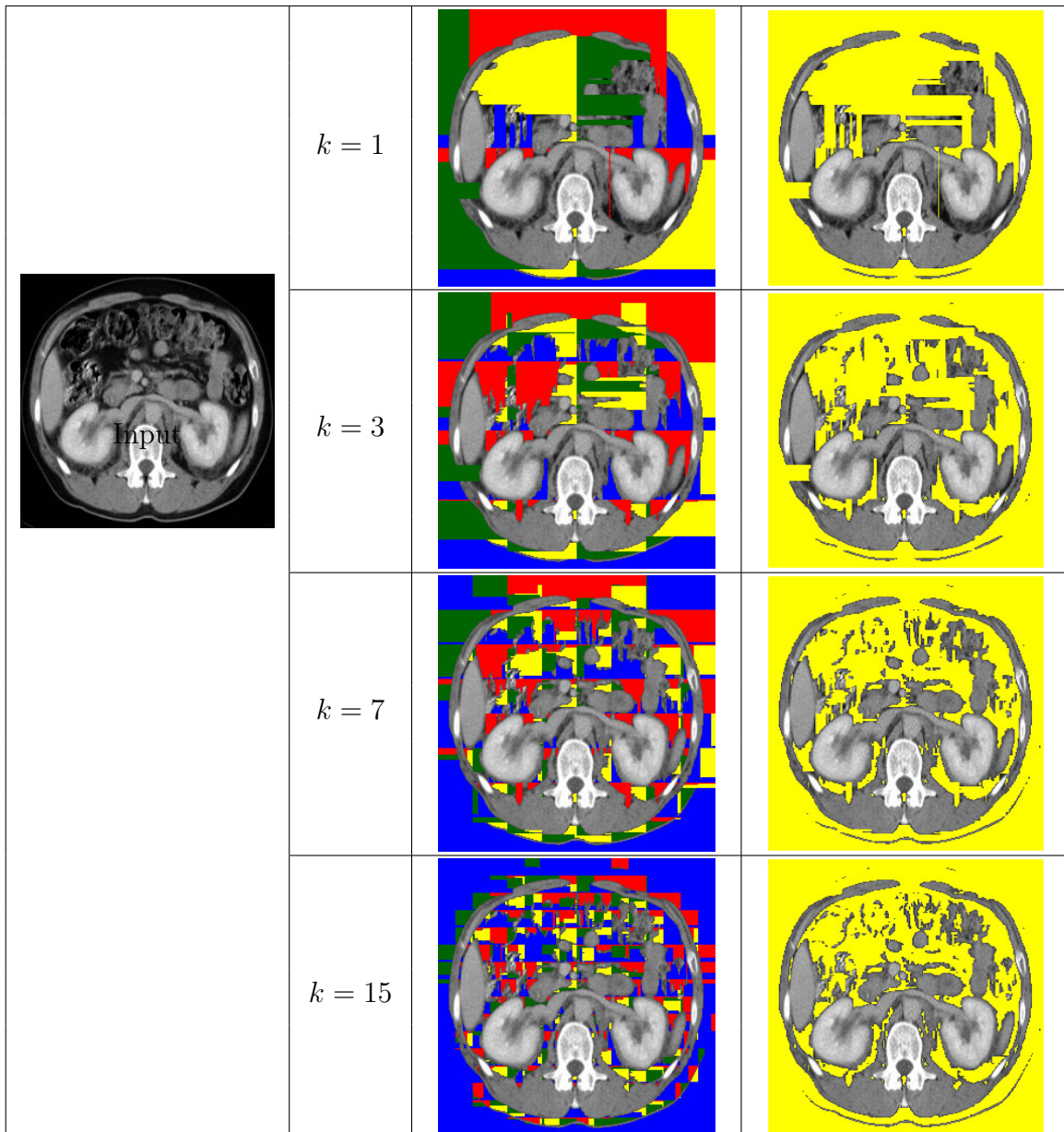


Figure 3.31: Segmentation of a picture of a shoes shop k -baseline method where $k = 3$, and 7.. The lower images show the case in which 30% random white noise has been added in the input image.

Figure 3.32: Segmentation of a CT image using k -baseline method where $k = 1, 3,$ and 7 .

3.4.3 Grid Decomposition: Optimal k -Baselines

The size of baselines being used are $k = 1, 3$, and 7 . We show two results for this method that are: (1) the room-edge region of the optimal vertical baselines; (2) the room-edge region with the horizontal baselines computed using the optimal vertical baselines.

Result

Figure 3.33 and Figure 3.34 show the segmentation result of the optimal k -baselines using $k = 1, 3$ and 7 . In Figure 3.33, it is easy to see that the baselines are placed closed to the neck area for both a picture with 30% noise and without noise. Similarly, the location of both vertical and horizontal lines in Figure 3.34 are placed closed to the flower which is the main object in the picture.

From the input image in Figure 3.35, there are many interested component and each has different shapes. The segmentation using $k = 3$ and 7 for the optimal k -baselines location can be found in Figure 3.35. The segmentation are similar, however, the result using $k = 7$ segments a smaller region in the upper left region while the one with $k = 3$ ignores it.

Figure 3.36 shows the segmentation result of a picture which contains many objects using $k = 3$, and 7 . For a picture of goldfish, we can easily see that the segmentation results are similar. However, if we consider the connectivity of the segmented region, the result of $k = 3$ is better. For a picture which objects are overlapped such as a picture of rabbits, using $k = 7$ can remove the background region more than $k = 3$. The segmentation of a sophisticated picture such as a picture of shoes store shows that the boundary of the segmentation region of $k = 7$ is more natural than using $k = 3$. Notice that the segmented region of $k = 3$ is rectangular.



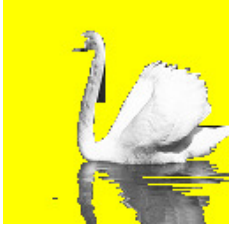
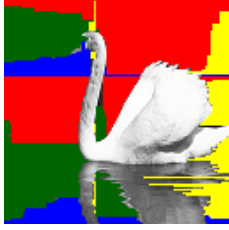





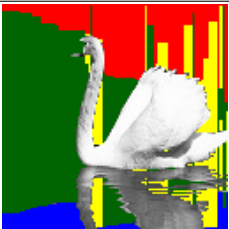
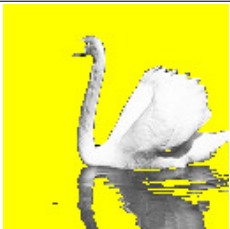
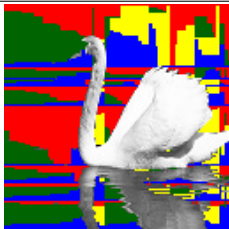
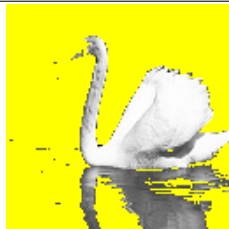





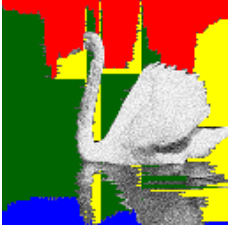
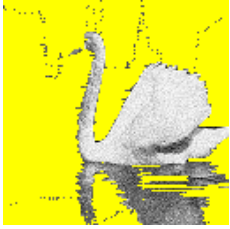
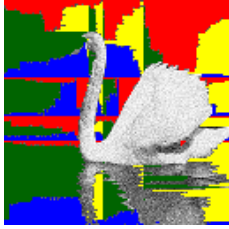
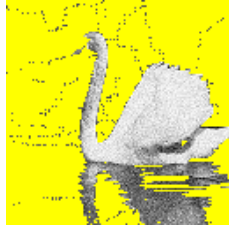
Input	k	Optimal vertical baselines		Vertical and horizontal baselines	
	1				
	3				
	7				
 30% noise	1				
	3				

Figure 3.33: Segmentation result of a swan picture and a flower picture using k optimal baseline location with $k = 1, 3$ and 7 .





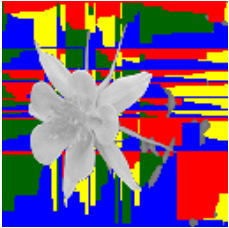
Input	k	Optimal vertical baselines	Vertical and horizontal baselines
	3		
	7		

Figure 3.34: Segmentation result of a flower picture using k optimal baseline location with $k = 3$ and 7.


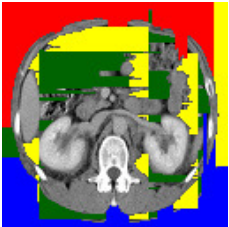
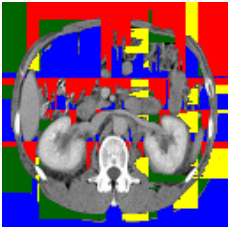
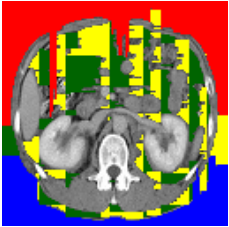
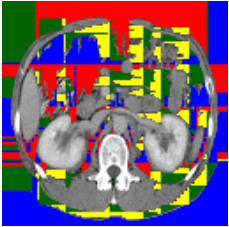
Input	k	Optimal vertical baselines	Vertical and horizontal baselines
	3		
	7		

Figure 3.35: Segmentation result of heterogeneous components using k optimal baseline location with $k = 3$ and 7.

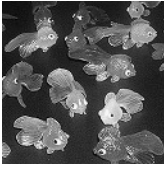
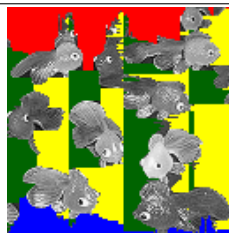
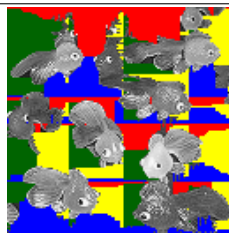
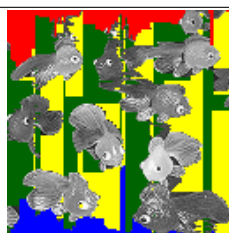
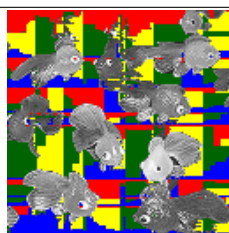




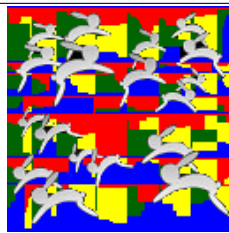


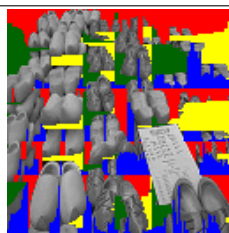
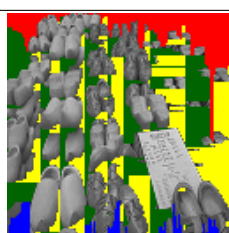
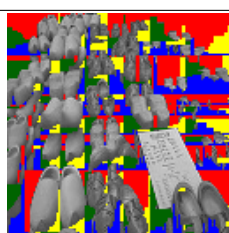
Input	k	Optimal vertical baselines	Vertical and horizontal baselines
	3		
	7		
	3		
	7		
	3		
	7		

Figure 3.36: Segmentation result of a picture of goldfish, rabbits, and shoes store using k optimal baseline location with $k = 3$ and 7.

3.4.4 d -depth Quadtree Decomposition

Due to the computation time of the quadtree decomposition, we apply a casual formulation in this experiment. In the casual formulation, the pixel grid \mathcal{G} is equally divided into $(y + 1) \times (y + 1)$ grid and denoted by Q . The d -depth quadtree decomposition algorithm applied on Q . In this experiment, we set $y = 1, 3$, and 7 and the depth of the quadtree is set to $d = 2, 3$, and 4 .

Figure 3.37 shows the segmentation of a picture containing a single object using $y = 7$ and 15 and $d = 2$ and 3 . From the result, it is easy to see that the segmentation results of a picture of swan are very similar. On the other hands, the baseline location of the picture with 30% noise is slightly different from the one without noise. From the two input picture, we notice that the baseline segments are placed at the shadow when y and d become larger. Figure 3.38 shows the result of a picture of a flower. The segmentation result of $y = 3$ and 7 using $d = 2$ are almost the same excepts that the location of the baseline segments of $y = 7$ and $d = 2$ is placed nearby the bright background region.

Figure 3.39 shows the segmentation of a computer tomography picture using $y = 7$ and 15 and $d = 3$ and 4 . From the result, we can see that using $d = 4$ gives a better result than the one using $d = 3$. The segmentation of a picture of goldfish can be found in Figure 3.40. We set the grid size $y = 7$ and 15 and the depth of the quadtree $d = 2, 3$ and 4 . We can see that the segmentation results are similar. The region of $y = 7$ and $d = 2$ is the most connected, however, boundary of the region is not as smooth as the one of $y = 15$ and $d = 3$. Figure 3.41 shows the segmentation results using $y = 7$ and 15 and $d = 3$ and 4 . We can see that the dark pixels are removed when using $y = 15$ and $d = 4$. The results of a picture of shoes with $y = 7$ and 15 and $d = 2$ and 3 are shown in Figure 3.42. Although, the segmentation regions are similar, the region boundary of the result using $y = 15$ and $d = 3$ is smoother than the others.


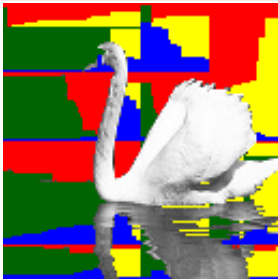

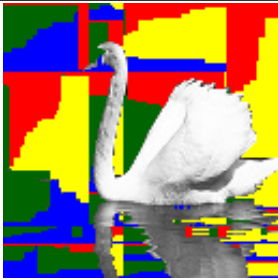


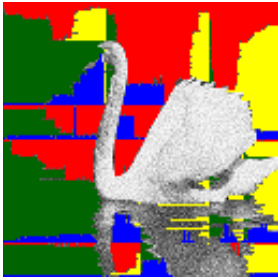

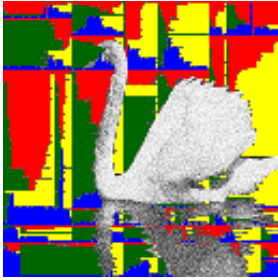
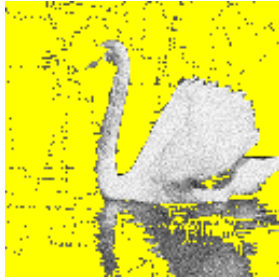
Input	Grid y , Depth d	Room-edge Region	Output
 Input	$y = 7$, $d = 2$		
	$y = 15$, $d = 3$		
 Noise 30%	$y = 7$, $d = 2$		
	$y = 15$, $d = 3$		

Figure 3.37: Segmentation of a picture of a swan using the quadtree decomposition.




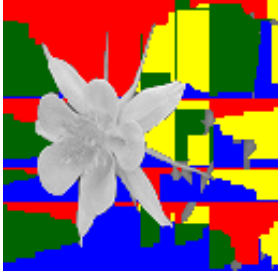

Input	Grid y , Depth d	Room-edge Region	Output
	$y = 3,$ $d = 2$		
	$y = 7,$ $d = 2$		

Figure 3.38: Segmentation of a picture of a flower using the quadtree decomposition.


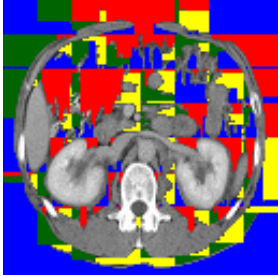
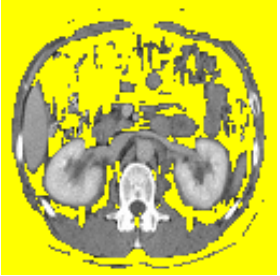
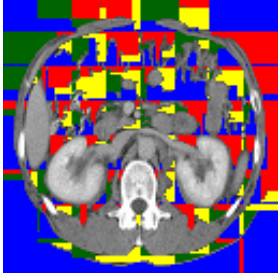
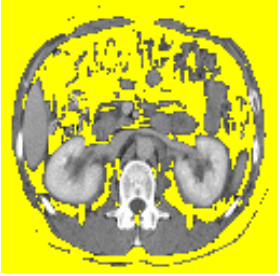
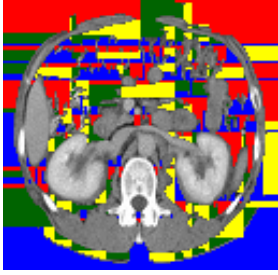

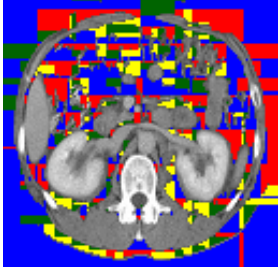
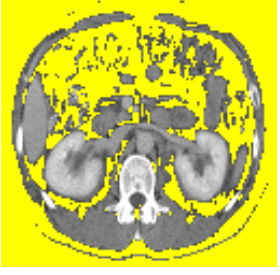
Input	Grid y , Depth d	Room-edge Region	Output
	$y = 7,$ $d = 3$		
	$y = 7,$ $d = 4$		
	$y = 15,$ $d = 3$		
	$y = 15,$ $d = 4$		

Figure 3.39: Segmentation of heterogeneous components using the quadtree decomposition.

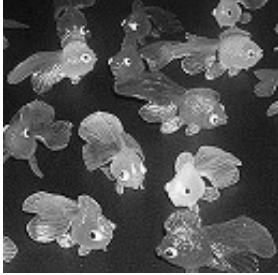
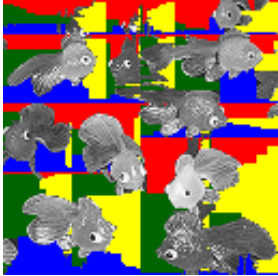
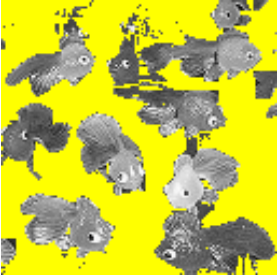
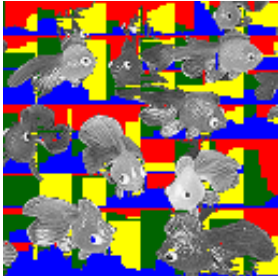
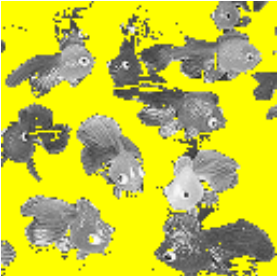
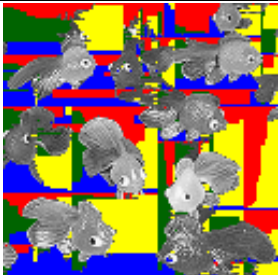

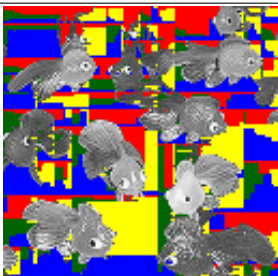
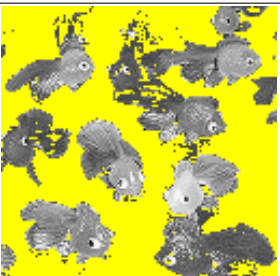
Input	Grid y , Depth d	Room-edge Region	Output
	$y = 7,$ $d = 2$		
	$y = 7,$ $d = 4$		
	$y = 15,$ $d = 3$		
	$y = 15,$ $d = 4$		

Figure 3.40: Segmentation of a picture of goldfish using the quadtree decomposition.



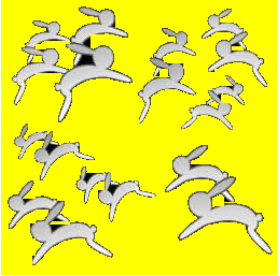
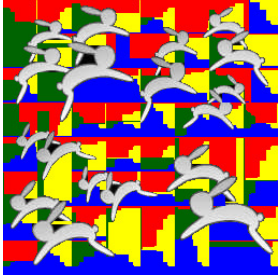
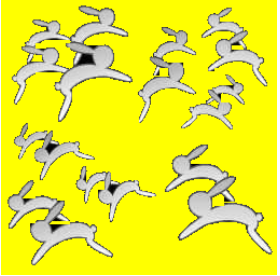

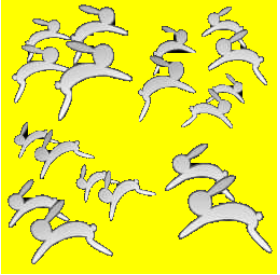


Input	Grid y , Depth d	Room-edge Region	Output
	$y = 7,$ $d = 3$		
	$y = 7,$ $d = 4$		
	$y = 15,$ $d = 3$		
	$y = 15,$ $d = 4$		

Figure 3.41: Segmentation of a picture of rabbits using the quadtree decomposition.


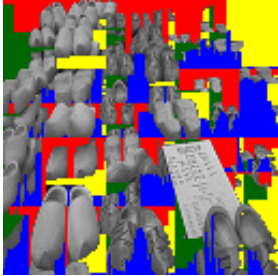
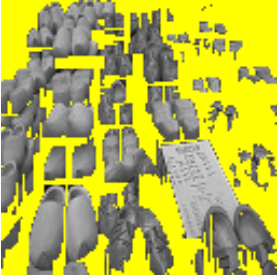
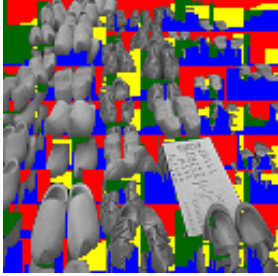
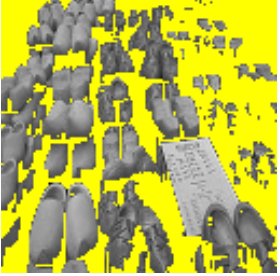
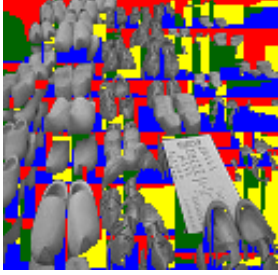
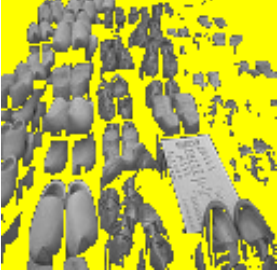
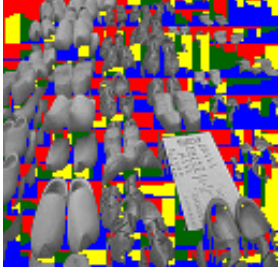
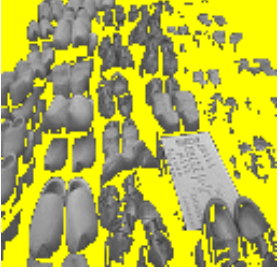
Input	Grid y , Depth d	Room-edge Region	Output
	$y = 7,$ $d = 2$		
	$y = 7,$ $d = 3$		
	$y = 15,$ $d = 3$		
	$y = 15,$ $d = 4$		

Figure 3.42: Segmentation of a picture of shoes using the quadtree decomposition.

3.4.5 Segmentation Result Comparison

Comparison with the proposed decomposition methods

We present the comparison of the segmentation of the proposed method that is the k -baseline, the optimal k -baseline location, and the d -depth quadtree decomposition.

Figure 3.43 shows the comparison of a picture of swan. It is clear from the comparison that the segmentation results of the three methods are similar. However, the segmentation of the shadow of the swan's neck shows that the decomposition using the k -baselines gives the better result than the optimal k -baselines.

The segmentation result using the optimal k -baseline location of the Figure 3.44 shows more reasonable location than the k -baseline method. However, the segmentation results obtained from the three methods are similar.

The comparison of the segmentation of the goldfish image can be found in Figure 3.45. Using $k = 3$ baselines, the result from the optimal k -baselines location gives better segmentation. Notice that the result from the k -baselines algorithm contains a few rectangular corners, however, those corner are removed if the baselines are optimally placed.

Figure 3.46 shows the comparison of the results of a picture of rabbits using the k -baselines, the optimal k -baselines location, and the d -quadtree decomposition. Comparing the result using the grid decomposition, it is obvious that the optimal k -baselines location is better than the k -baselines.

The segmentation of a picture in Figure 3.47 shows that the results of from the three methods are similar. Comparing the smoothness of the segmented region boundary, the optimal k -baseline gives the best result. The comparison of the segmentation result of a picture which contains homogeneous objects can be found in Figure 3.48.

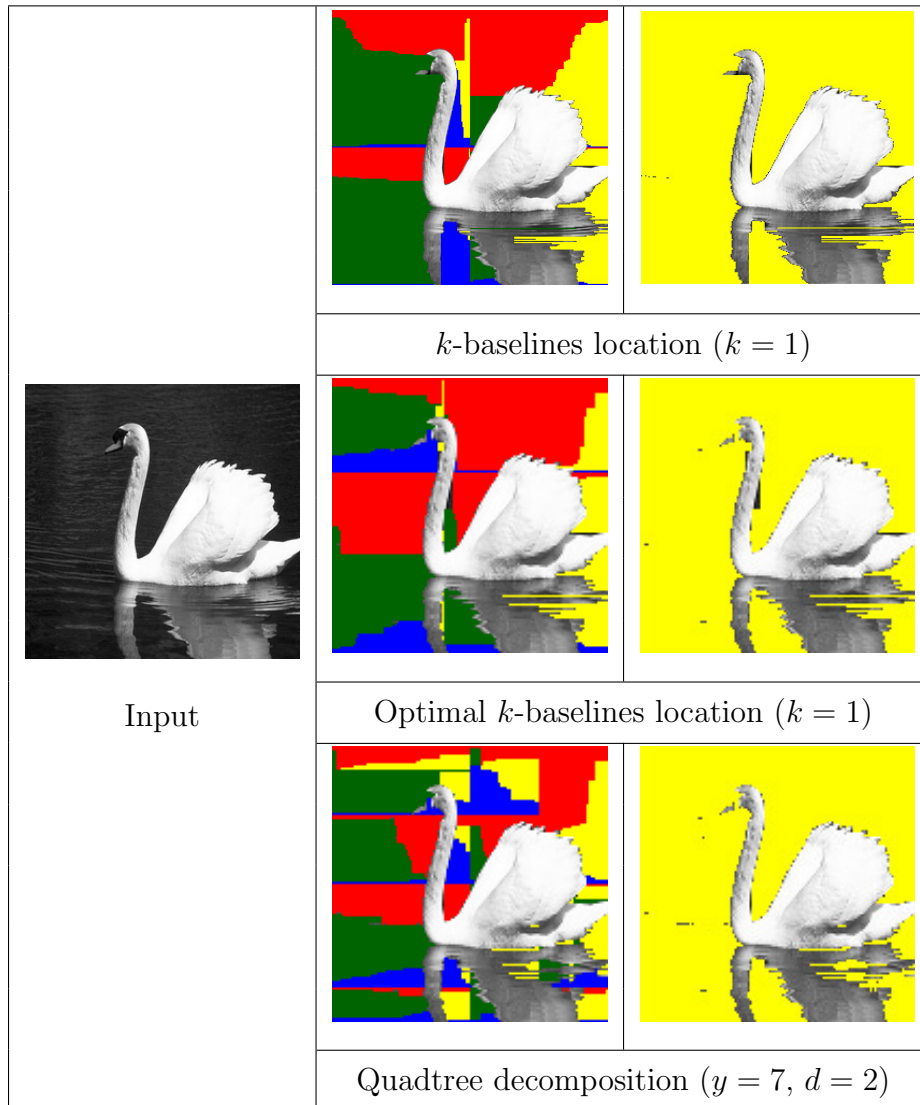


Figure 3.43: Comparison of the segmentation result of a picture of a swan using the k -baselines decomposition, the optimal k -baseline location, and the quadtree decomposition.

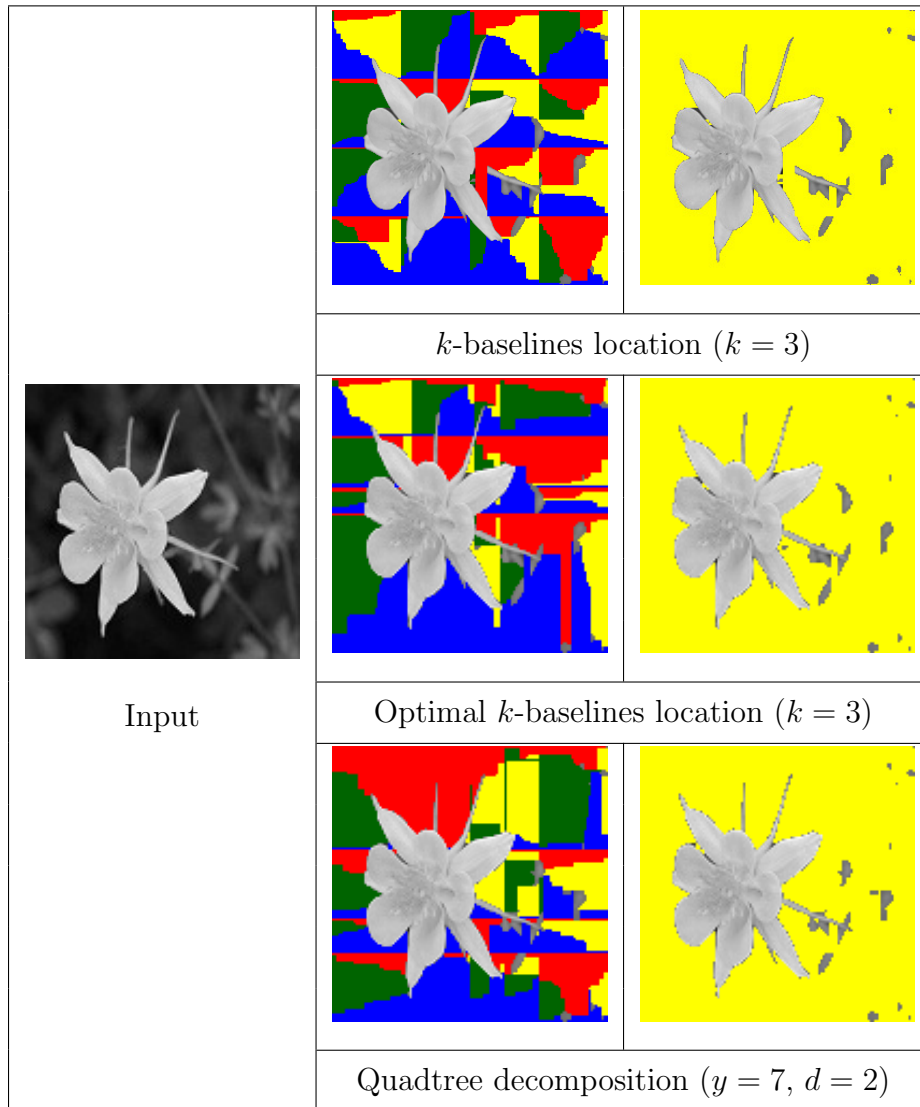


Figure 3.44: Comparison of the segmentation result of a picture of a swan using the k -baselines decomposition, the optimal k -baseline location, and the quadtree decomposition.

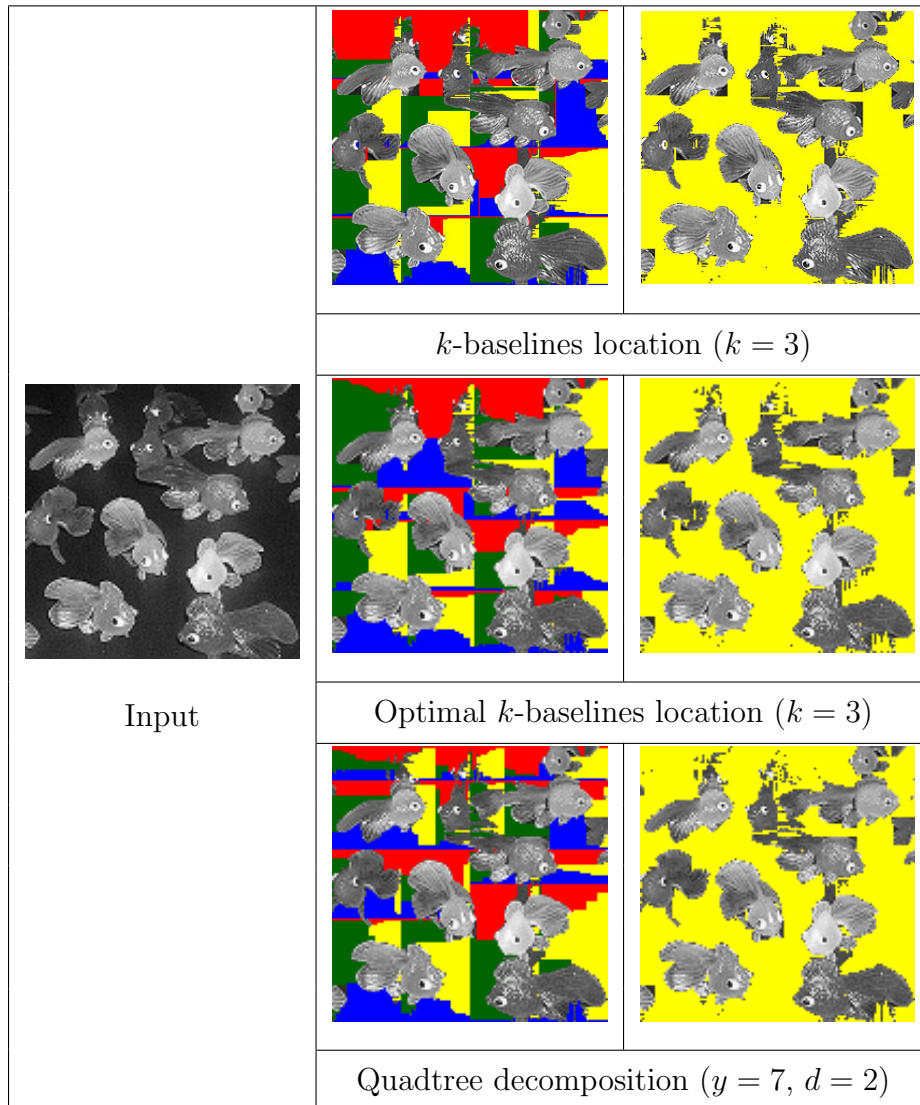


Figure 3.45: Comparison of the segmentation result of a picture of goldfish using k -baselines decomposition, optimal k -baseline location, and quadtree decomposition.

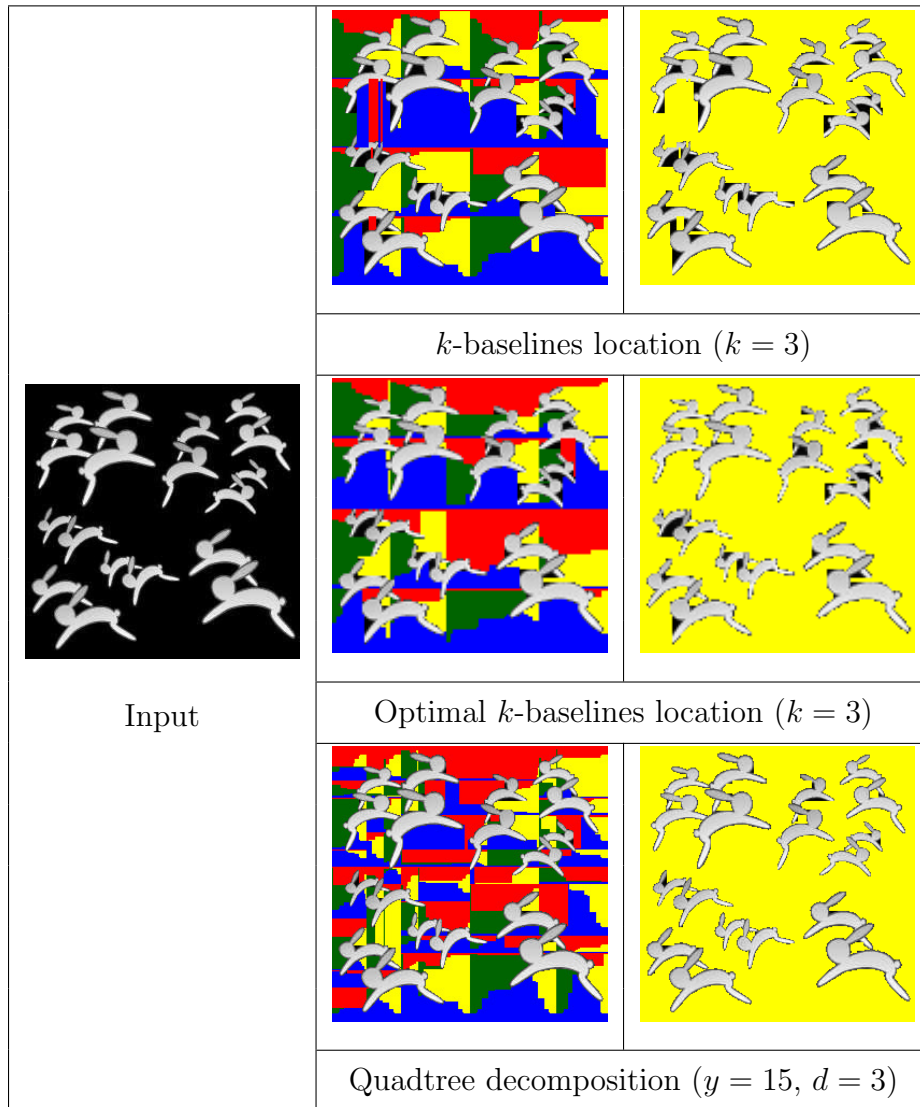


Figure 3.46: Comparison of the segmentation result of a picture of rabbits using k -baselines decomposition, optimal k -baseline location, and quadtree decomposition.

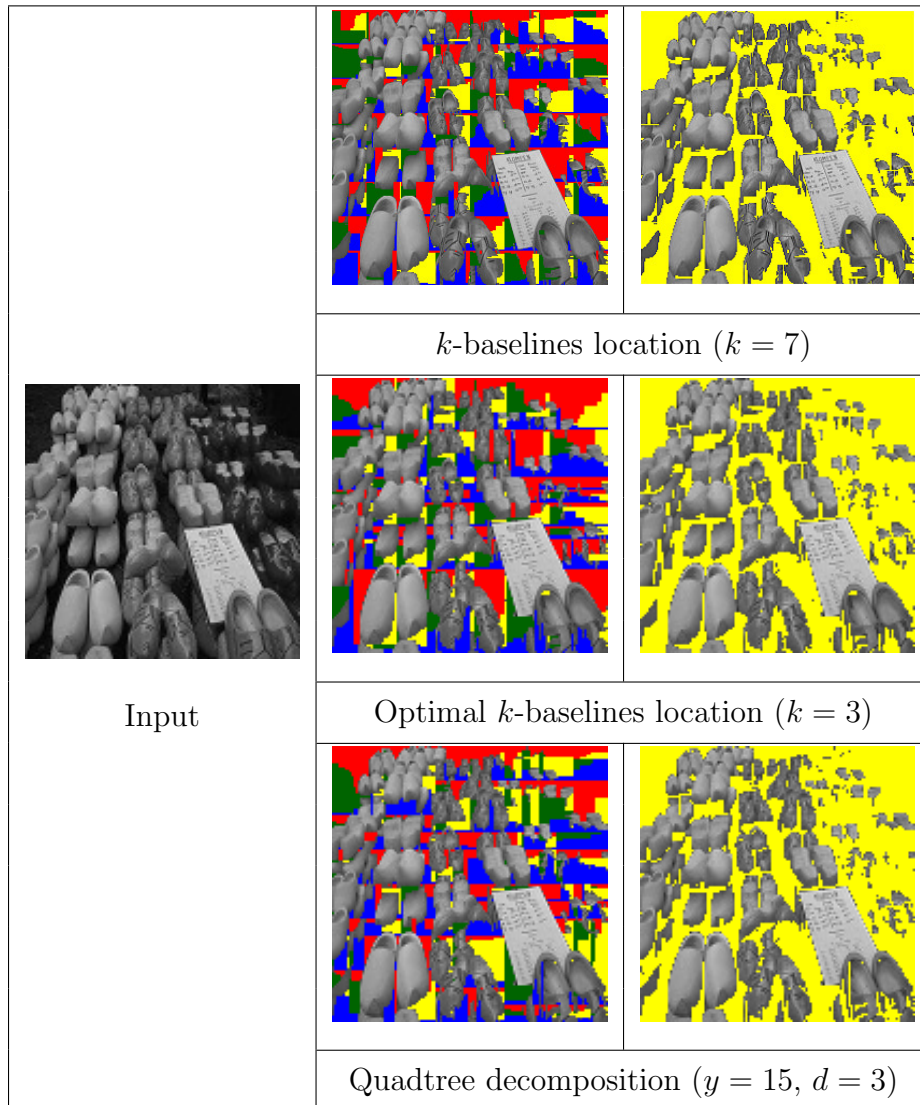


Figure 3.47: Comparison of the segmentation result of a picture of shoes using k -baselines decomposition, optimal k -baseline location, and quadtree decomposition.

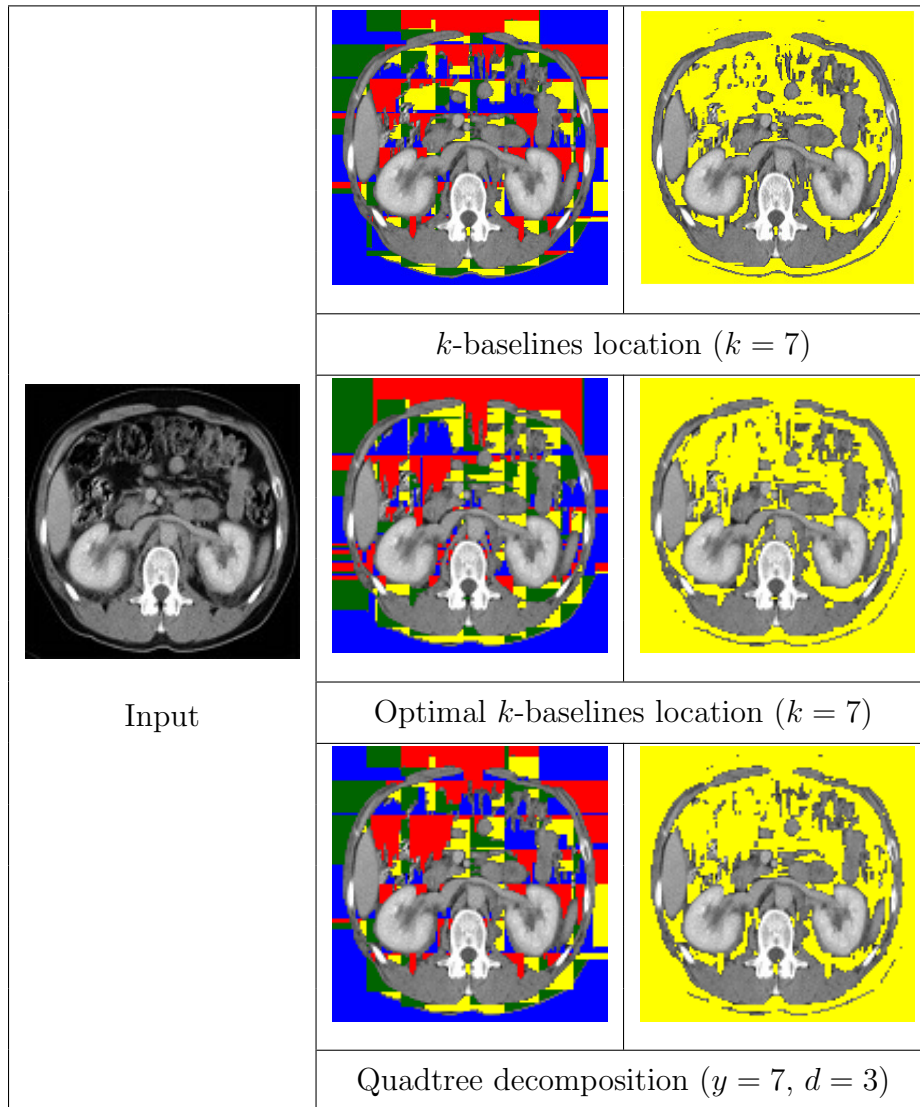


Figure 3.48: Comparison of the segmentation result of heterogeneous components using k -baselines decomposition, optimal k -baseline location, and quadtree decomposition.

Comparison with graph-cut algorithm

We compare the segmentation result using the MWRP room-edge region and the graph cut. The implementation of the image segmentation using graph cuts can be found in [87].

In the implementation in [87], two set of constraints which are a set of pixels in the object region and the one in the background region are specified by a user. Figure 3.49 shows the segmentation of a flower which a dark region in the middle of the flower is considered as an object region. From the segmentation result, it is clear that the two set of constraint have an effect on the output. Some dark pixels on the left picture which are not selected as the input constraint are removed. Comparing to the MWRP room-edge region method, the dark pixels are removed if the baselines pass through that area. On the other hands, if the dark pixels are selected as the input constraint, such region is not segmented by the graph-cut method. Also for the MWRP room-edge region, the dark pixel region is retained if the baselines do not across such region.

Figure 3.50 shows the segmentation result of the image which the background contains some bright region. Comparing the segmentation of the two methods, we notice that the MWRP gives less background region than the graph-cut method. Also the second constraint of the graph-cut method returns less background region than the first one.

The segmentation of pictures that contains many objects can be found in Figure 3.51. From the segmentation of the picture of rabbits, we notice that some narrow areas such as legs and ears are removed by the graph-cut method. Moreover, the background region in the overlapping area is not removed. On the other hands, the object regions, i.e., rabbits legs, are not removed; however, the overlapping region is remaining as the graph-cut method.

The segmentation region from the graph-cut method of the shoes picture is more connected than the one of the MWRP. On the other hands, the MWRP method is able to remove the entire dark region between each pair of shoes. Comparing to the graph-cut method, giving the constraint in this small region is not easy.

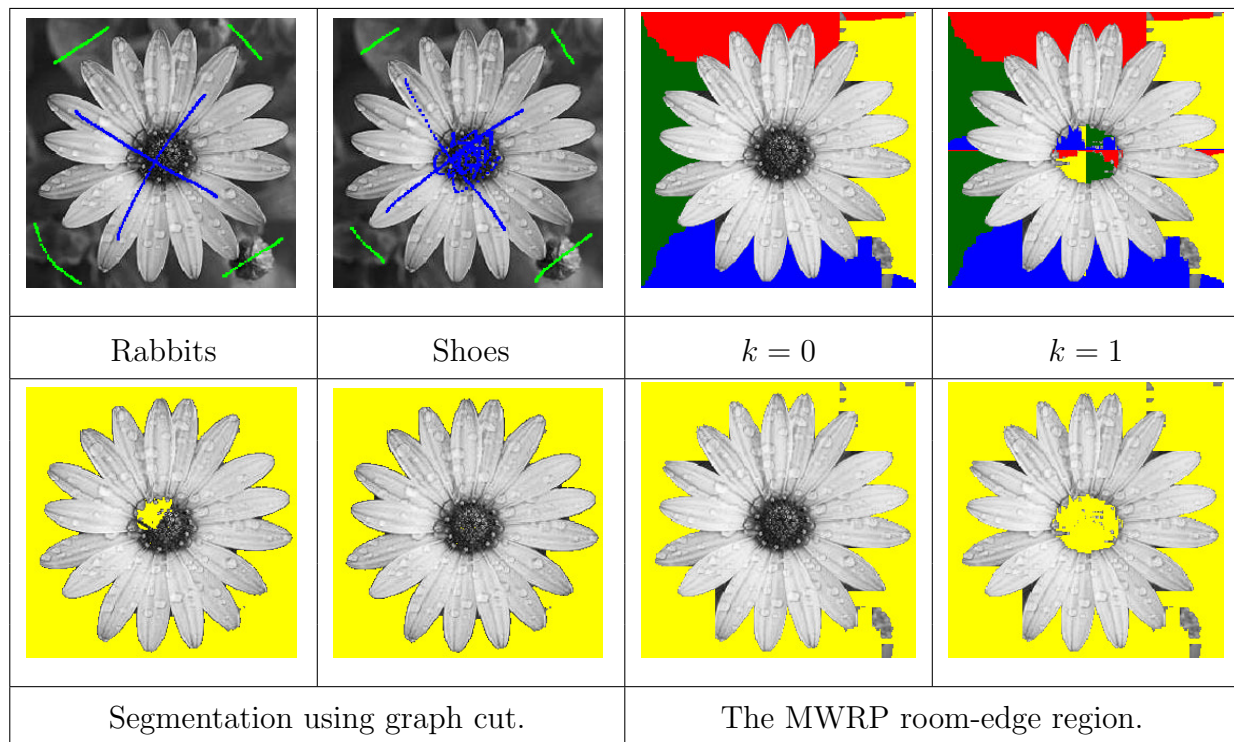


Figure 3.49: Comparison of the segmentation result of a picture of a flower using graph cut and the MWRP room-edge region. The upper images shows two set of pixels that are in the object region (blue) and the background region (green) of the graph-cut method and the optimal room-edge region of the MWRP method.

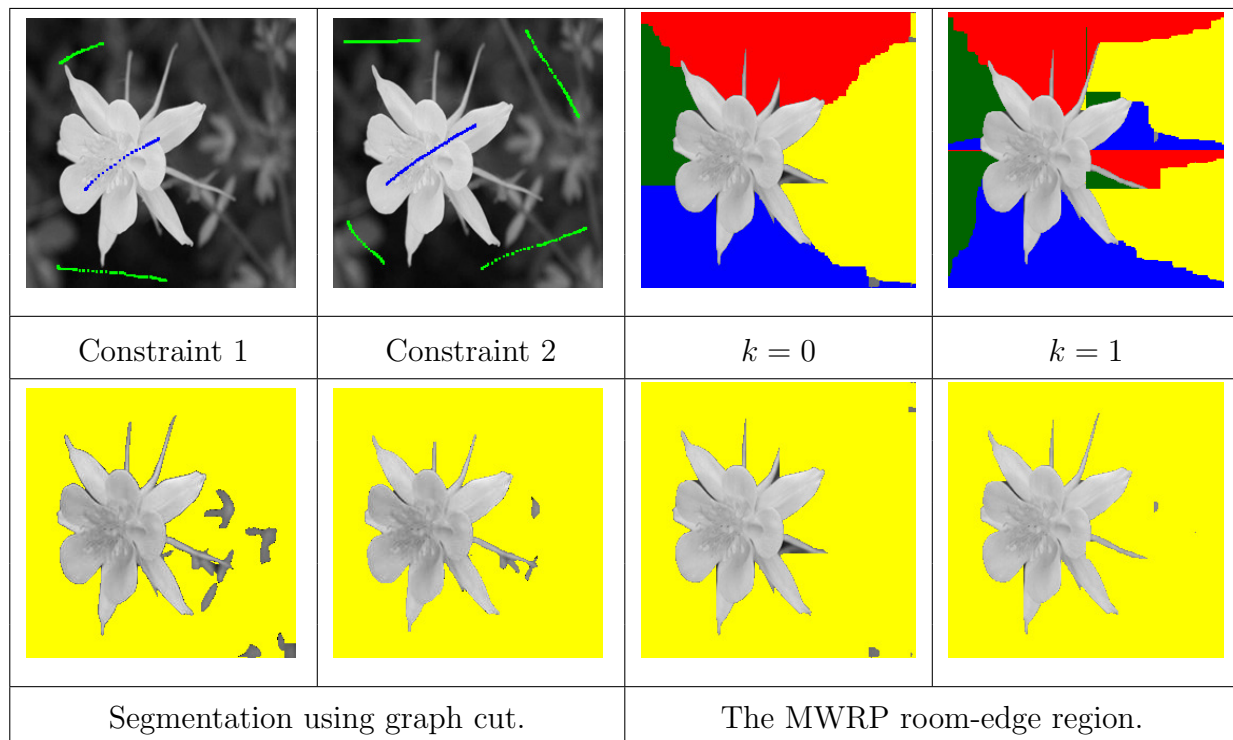


Figure 3.50: Comparison of the segmentation result of a picture of a flower with a complicated background using graph cut and the MWRP room-edge region. The upper images shows two set of pixels that are in the object region (blue) and the background region (green) of the graph-cut method and the optimal room-edge region of the MWRP method.

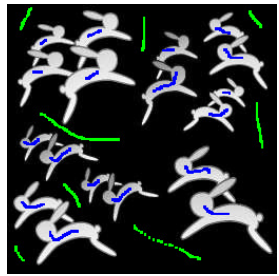
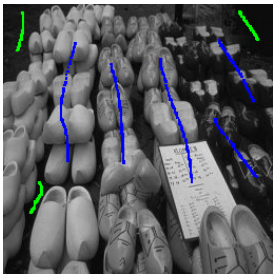
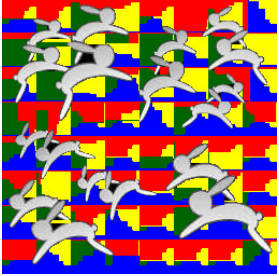
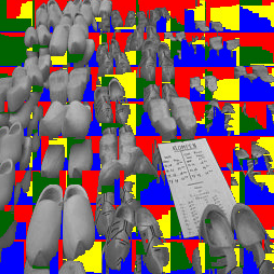
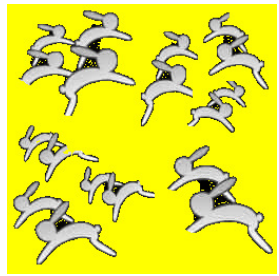

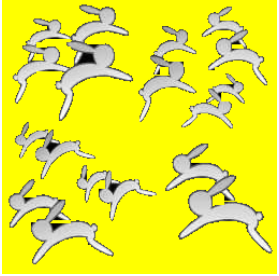
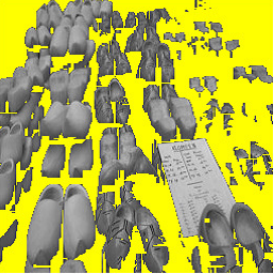
			
Constraint 1	Constraint 2	$k = 7$	$k = 7$
			
Segmentation using graph cut.		The MWRP room-edge region.	

Figure 3.51: Comparison of the segmentation result of a picture that contains many objects using graph cut and the MWRP room-edge region. The upper images shows two set of pixels that are in the object region (blue) and the background region (green) of the graph-cut method and the optimal room-edge region of the MWRP method.

3.5 Discussion and Conclusion

We consider a special case of the base-monotone region that is the room-edge region. The proposed algorithm can be applied to solve an image segmentation problem using the framework proposed by Asano *et al.*[10]. The room-edge region is applied for removing the background region and output the complement region. We experimentally demonstrated the efficiency of the proposed method. The results show that the room-edge region gives a good segmentation result of a picture which contains a single object.

The quality of the segmentation of a picture containing many objects can be improved by decomposing the pixel grids into small rectangular sub grid and independently solve the room-edge region. We proposed two decomposition methods: (1) A grid decomposition which a pixel grid is decomposed using k vertical and k horizontal baselines; (2) A quadtree decomposition which applies a depth-quadtree structure.

A complicated shape of object and an image which contains multiple objects and overlapping objects can be segmented nicely as shown in the experimental result. One problem when using the room-edge region is smaller pixels at the end of the regions of the segmentation result of a noise image. However, these pixels can be removed in a post-processing.

Given a positive integer k , the baselines are optimally placed in order to decompose the pixel grid. For the quadtree decomposition, a depth of the quadtree is given and the baseline segments are optimally placed. One problem of both methods is the number of baselines and the depth of the quadtree are given by a user. If the number is too large, there will be many small unconnected objects.

We also show the comparison of the segmentation result between the graph-cut method and our proposed MWRP room-edge region. From the result, it is easy to see that the segmentation of the graph-cut depends of the input constraint that is a set of pixels in the background region and the object region. To be applied in an automatic image segmentation, many approaches are presented in order to find a set of constraint pixels [83, 24].

Chapter 4

Distance-based Shape Descriptor

4.1 Introduction

Assume that image segmentation has been done and each object in the database is a simple polygon with m vertices. Our purpose is to find objects in the database whose shapes resemble to a query object Q . One method is to find a function $F(P)$ derived from shape boundary points of P which is called a shape signature. Shape signatures are usually normalized into being translation and scale invariant. Many shape signature exist which includes complex coordinate, centroid distance [26, 93], tangent angle, curvature [5], area and chord-length.

In order to compensate for orientation changes, shift matching is needed to find the best matching. However, the matching cost is too high for online retrieval and also the shape signatures are sensitive to noise, and slightly changes in the boundary, which can cause large errors. Therefore, it is undesirable to directly describe shape using a shape signature. Further processing is necessary to increase its robustness and reduce the matching load. Most of the signature matching is normalized to one dimension space. Some signature matching requires a matching in two dimensions space. For example, a shape signature can be simplified by quantizing the signature into a feature vector, which is rotationally invariant

[13, 14, 63, 70, 95, 97].

A distance distribution among points on the shape contour has been widely used as a two-dimension shape representation and also three-dimensional one [47]. Advantages of the distance-based shape signature are: a compact, easy to compute, and the similarity can be measured quickly. One problem of characterizing shapes based on distance between points is the existence of a homometric pair object. A simple example of a homometric pair object is shown in Figure 4.1 [85, 15]. From the figure, it is clear that the two different shapes of objects have exactly the same set of pairwise distances.

In this work, we focus on improving the accuracy of the distance-based shape descriptor. It is not only a compact shape representation but also easy to compute.

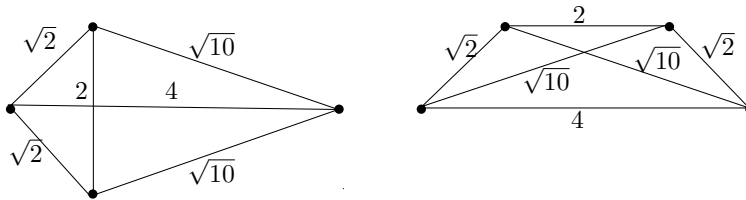


Figure 4.1: An example of a homometric pair object.

4.2 Summary of the result

In this chapter we propose two shape descriptors: (1) a Distance Interior Ratio (DIR); and (2) a Local Distance Interior Ratio (LDIR).

1. **Distance Interior Ration (DIR):** The DIR is a compact and easy-to-compute global-based descriptor computed from the shape contour using two distance features that are a relative distribution of pairwise distance and a distance interior.

We demonstrate the accuracy of the proposed descriptor and its efficiency to be integrated to an indexing data structure. The Bull's Eye Percentage (BEP) of the

proposed descriptor is 73.73% which is higher than the shape context and the inner distance shape context descriptors.

2. **Local Distance Interior Ration (LDIR):** We extend the idea of the DIR descriptor in order to compute the shape matching using the spatial data. The shape descriptor called an LDIR is attached to each point. In this way, the shape locality is preserved. Although the size of the LDIR descriptor is bigger than the DIR, it can be used as a correspondence finder for shape matching.

Computing the shape similarity measurement using the LDIR descriptor, the BEP is 78.27% which is higher than the DIR descriptor.

4.3 Distance Interior Ratio

Consider the Euclidean distance of the line segments from a point p to other points on the boundary of a triangle shape objects in Figure 4.2 (a) and also the one on a T shape object in Figure 4.2 (b). The Euclidean distance of the line segments of both shapes have the same distance distribution. This is because the Euclidean distance does not take into account whether or not the line segment crosses the boundary.

From the line segment in Figure 4.2, we can see that a line segment lies on the shape boundary in three pattern: (1) Entirely within the shape boundary; (2) Entirely outside the shape boundary; and (3) Lying across the shape boundary. From the three pattern, it is clear from the example that all the line segments in Figure 4.2 (a) are the first pattern. On the other hands, only 3 line segments in Figure 4.2 (b) are in the in the first pattern.

Inspired by the above idea, we can classify the line segments using the fraction of the length of the line segments lying in the polygon and the length of the line segment. Let consider the two shapes in Figure 4.2, the ratio of the line segments lying inside and outside the shape of all line segments in Figure 4.2 (a) equals 1, while it is strictly less than 1 for

some line segments in Figure 4.2 (b).

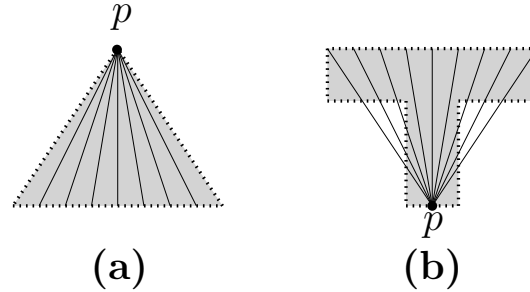


Figure 4.2: (a) Line segments from a point p to other points on the boundary of a triangle shape object. (b) Line segments from a point p to other points on the boundary of a T shape object.

Based on the proposed idea, we present a shape descriptor called a *Distance Interior Ratio* (DIR) which is the distribution of the fraction of the line segments lying in the polygon and the length of the line segments on the contour points. Two distance features are employed: (1) a Euclidean distance; and (2) the interior ratio which is the ratio of the line segments lying inside and outside the shape. The shape signature is derived from the two distance features computed from the set of points on the shape boundary. It is normalized so that the descriptor is invariant to transformation and scale changes using a statistical approach, i.e., a structural histogram. The similarity of the two shape descriptor is measured using L_1 norm.

4.3.1 Distance Features

We explain how to compute the two distance features especially the proposed interior ratio distance feature.

Given a set $P = \{p_1, \dots, p_m\}$ of m arbitrary points on the boundary of a simple polygon which is extracted from a binary image. Let $[pq]$ be the line segment connecting the two points $p, q \in P$.

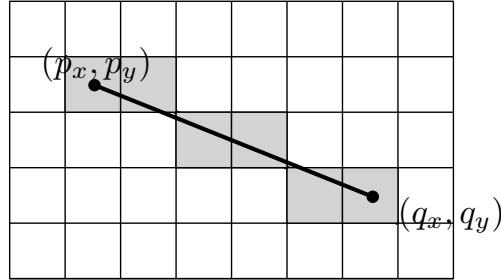


Figure 4.3: An example of an approximate digital line segment using the Bresenham's algorithm.

Let d_{pq} denote the Euclidean distance of $[pq]$. The Euclidean distance is computed as

$$d_{pq} = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}.$$

Let d_{pq}^{ir} be the interior ratio, i.e., the fraction of the line segment lying in the polygon and the length of the line segment. To calculate d_{pq}^{ir} , let $L(p, q) = \{l_1, \dots, l_k\}$ be a digital line segment from a point p and q where l_i for $1 \leq i \leq |L(p, q)|$ is a pixel on the $L(p, q)$. In this research, we use Bresenham's Algorithm [20] for approximating a digital line segment $[pq]$.

Given coordinate points (p_x, p_y) and (q_x, q_y) of a starting and ending points of \overline{pq} . Let (x', y') be a current pixel approximated by the algorithm. The Bresenham's algorithm keeps track of an error value when selecting an approximated position (x', y') . If the error value of a midpoint of a pixel at an integer position y' and the actual y is between -0.5 and 0.5 , then x' is increment by 1 and the error value is increased by the slope. Otherwise, y' is increased by 1 and the error value is decrement by 1. An example of an approximate line segment is shown in Figure 4.3

An example of a digital line segment $L(p, q)$ extracted from the shape contour can be found in Figure 4.4(b). As we only consider binary images, we assume that the intensity $f(l_i)$ of every pixel l_i is either 1 or 0. See Figure 4.4(b). Let d_{pq}^{in} be the distance of the fraction of $[pq]$ within the boundary of P . d_{pq}^{in} is the number of pixels l_i with $f(l_i) = 1$ for

$1 \leq i \leq |L(p, q)|$. For the DIR, we have

$$d_{pq}^{ir} = d_{pq}^{in} / |L(p, q)| \quad (4.1)$$

The shape information is transformed to a feature space F , where each line segment $[pq]$ is represented by the point (d_{pq}, d_{pq}^{ir}) . More precisely, the x -axis represents the Euclidean distances between the vertices of P and the y -axis corresponds to the interior ratio of P . The ranges of the x -axis and the y -axis are $[d_{min}, d_{max}]$ and $[d_{min}^{ir}, d_{max}^{ir}]$ respectively, where d_{min} is the minimum Euclidean distance between points in P and d_{max} , d_{min}^{ir} , and d_{max}^{ir} are defined analogously.

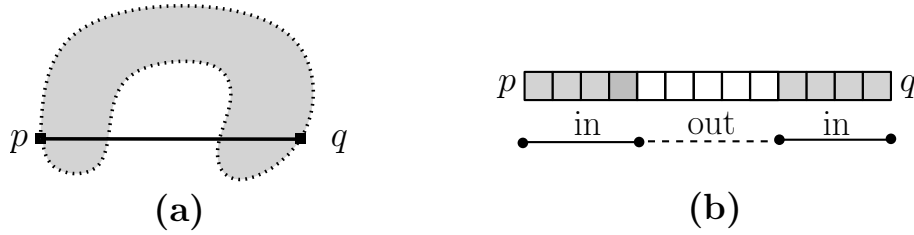


Figure 4.4: (a) A line segment $[pq]$. (b) The digital line segment extracted from point p to q . “in” represents the portion of a line within the shape boundary while “out” represents the outside portion.

4.3.2 Structural Histogram

In order to achieve a compact descriptor, the distance feature, i.e., the pairwise Euclidean distance and the interior ratio, is compressed using the statistical approach.

Given two positive integers r and c , the feature space F is equally divided into $r \times c$ blocks. The x -axis and the y -axis are divided into c and r blocks respectively. The interval of each block on the x -axis is $[(d_{max} - d_{min})/c]$ and the y -axis is $[(d_{max}^{ir} - d_{min}^{ir})/r]$. By dividing the feature space, we know that the interior-class line segment which is entirely or mostly within the shape is plotted in the upper block as depicted in Figure 4.5.

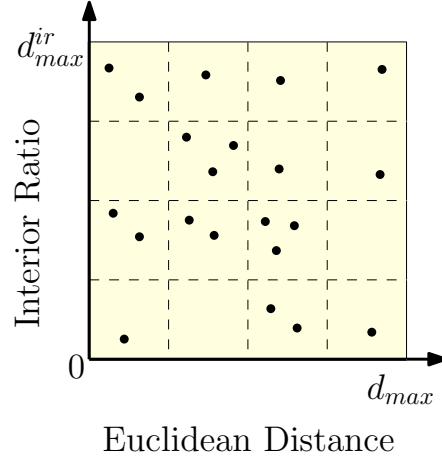


Figure 4.5: A feature space F which x -axis represents the Euclidean distances and y -axis represents the interior ratio. F is equally divided into 4×4 blocks.

Definition 4.3.1. *Multiplicity of the distance features, i.e., the Euclidean distance and the interior ratio, of each block is denoted by $h(u, v)$, where $0 \leq u \leq c - 1$ and $0 \leq v \leq r - 1$. We used $H(P)$ to denote the distribution of the classified distances of a set of the boundary points P which is a $c \times r$ histogram matrix. $H(P)$ can be written as:*

$$H(P) = \begin{bmatrix} h(0,0) & h(0,1) & \dots & h(0,c-1) \\ h(1,0) & h(1,1) & \dots & h(1,c-1) \\ \vdots & \vdots & \ddots & \vdots \\ h(r-1,0) & h(r-1,1) & \dots & h(r-1,c-1) \end{bmatrix}$$

Let (u, v) denote a block in \mathbb{F} where u and v are the index on the x -axis and the y -axis respectively. The interval of each block is denoted by $[x_u, x_{u+1}], [y_v, y_{v+1}]$. Let $h(u, v)$ be the number of points that fall into $[x_u, x_{u+1}], [y_v, y_{v+1}]$. To avoid the scaling problem, $h(u, v)$ is normalized by dividing the total number of points in the feature space. We have

$$h(u, v) = \frac{2}{m(m-1)} \sum_{i=0}^n \sum_{j=1}^{n-1} \hat{g}(d_{ij}, d_{ij}^{ir}, x_u, x_{u+1}, y_v, y_{v+1}),$$

where,

$$\hat{g}(d_{ij}, d_{ij}^{ir}, x_u, x_{u+1}, y_v, y_{v+1}) = \begin{cases} 1, & \text{if } d_{ij} \in [x_u, x_{u+1}] \text{ and } d_{ij}^{ir} \in [y_v, y_{v+1}], \\ 0, & \text{otherwise.} \end{cases}$$

The histograms of size $r = 32$ and $c = 64$ of the homometric pair objects in Figure 1.2 are shown in Figure 4.6. From the histogram, it is obvious that the DIR descriptors of the two shapes are totally different. For the triangle shape as shown in Figure 4.6 (a), the line segments are mostly in the lower and the upper part of the histogram. On the other hands, the T shape as in Figure 4.6 (b) is condensed in the middle of the histogram.

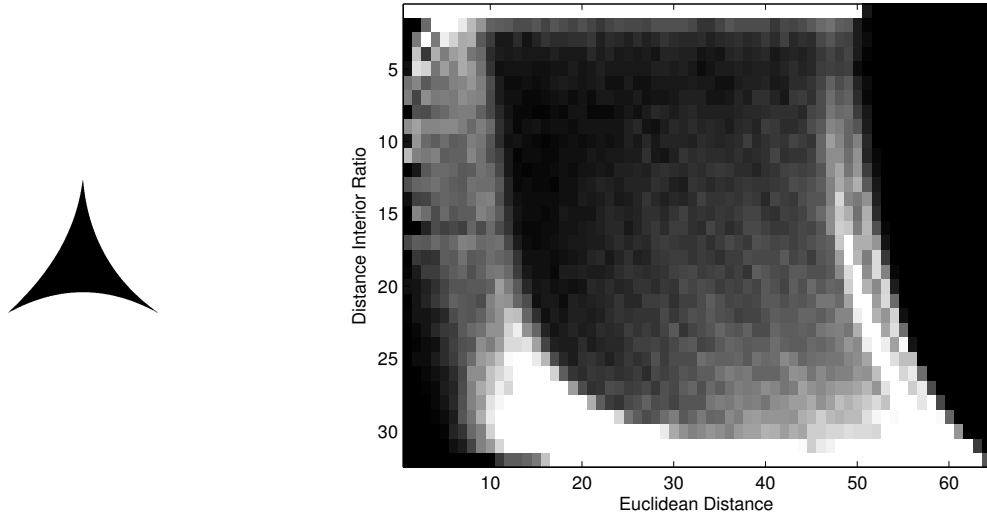
4.3.3 Similarity Measurement

After the DIR descriptor is computed for every object in the dataset, the similarity of the two descriptors is measured using a dissimilarity measurement. A number of techniques have been proposed. In this research, the dissimilarity is measured based on Minkowski L_P norms, in particular, the Probability Density Function (pdf) L_1 norm which is technique as presented by Osada *et al.*[70].

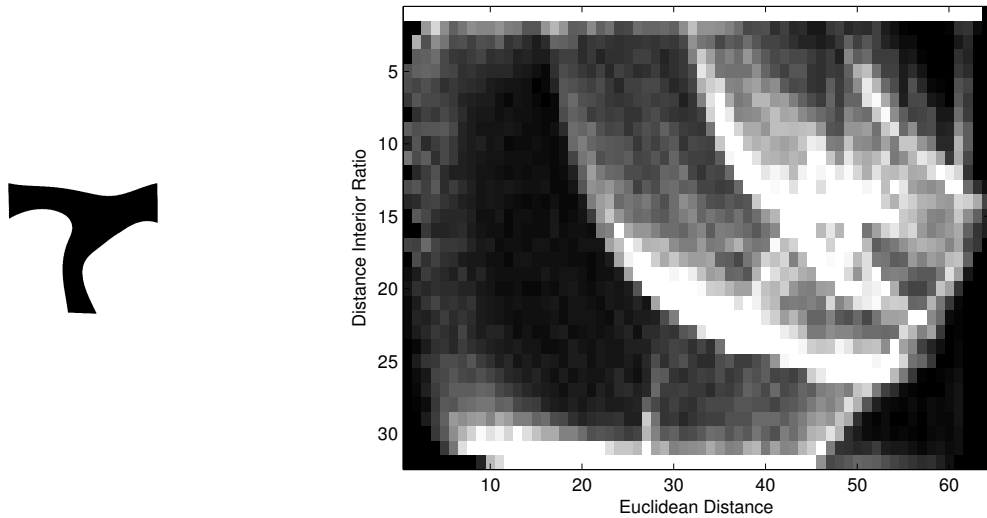
Given the two descriptors of shapes P and Q , let $dist_{DIR}(P, Q)$ be the distance between the descriptors. The dissimilarity measurement using L_1 norm is calculated as:

$$dist_{DIR}(P, Q) = \sum_{i=0}^{r-1} \sum_{j=0}^{c-1} |h_P(i, j) - h_Q(i, j)|$$

$dist_{DIR}(P, Q)$ is the total distance between the histograms $h_P(i, j)$ and $h_Q(i, j)$ where i and j are row and column position of bins respectively.



(a) DIR descriptor of a Triangle shape.



(b) DIR descriptor of a T shape.

Figure 4.6: The DIR of the homometric pair in Figure 1.2. The x -axis represent the Euclidean distance and the y -axis represent the DIR. The brightness of each pixel represents the number of points in the block. Bright pixel represents a high number of line segments.

4.3.4 Descriptor Properties

According to the histogram of the shape feature $H(P)$ of P , the convexity and the concavity properties can be derived from the descriptor. Let \hat{C} be the convexity value and let \hat{V} be the concavity value of an object.

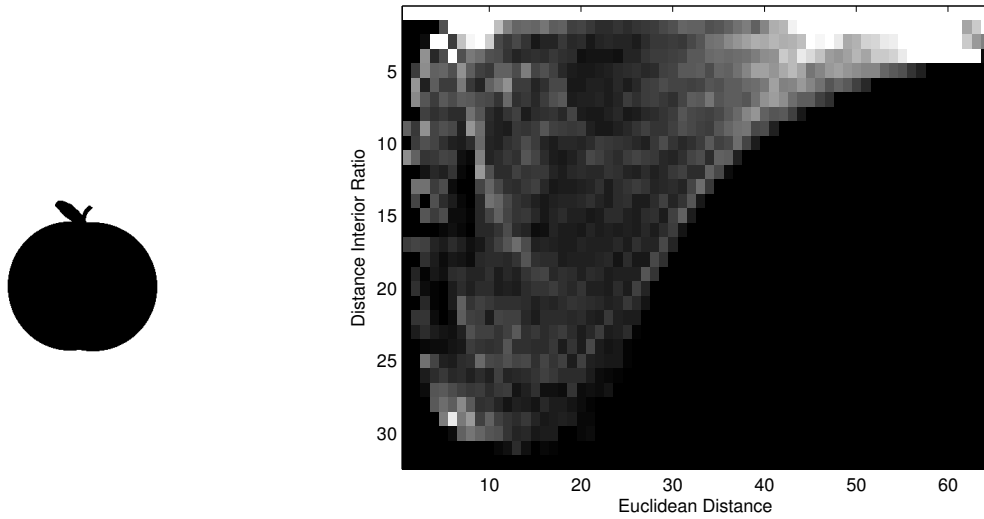
Definition 4.3.2. *For a given boundary points P of a shape, its convexity value is the probability that for any $p, q \in P$ picked uniformly at random, all points from the line segment $L(p, q)$ also belong to P .*

Given a digital line segment $L(p, q)$ from a pixel $p, q \in P$, if $L(p, q)$ lies within the shape contour then $d_{pq} = d_{pq}^{in}$. Thus, $d_{pq}^{ir} = 1$. From the distribution of the classified distance $H(P)$, the convexity value is the number of the line segments that map into blocks in $r - 1$ row of $H(P)$. The convexity value of the shape is $\hat{C} = \sum_{j=0}^{c-1} h(r - 1, j)$.

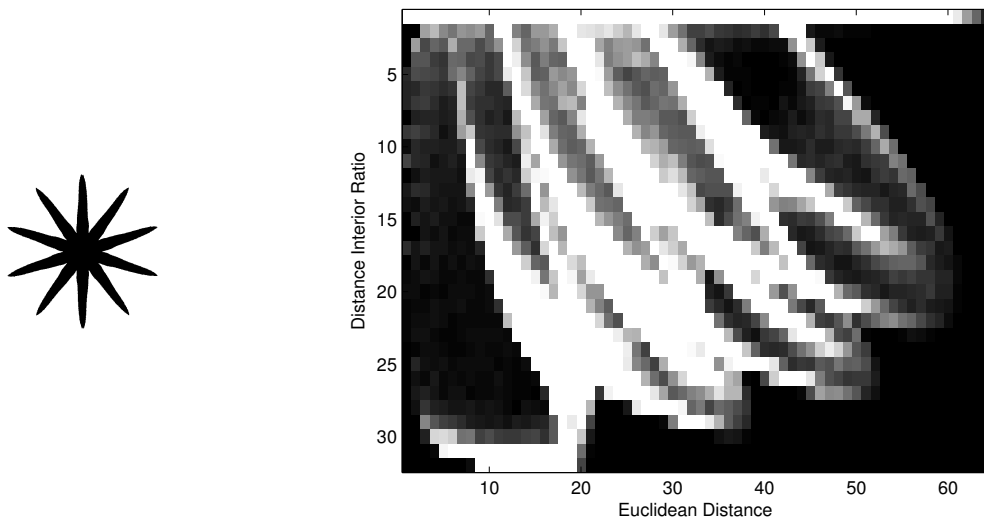
Definition 4.3.3. *For a given shape boundary points P , its concavity value is the probability that for any $p, q \in P$ picked uniformly at random, some points from the line segment $L(p, q)$ belong to P and some do not.*

On the contrary to the convexity value, d_{pq}^{in} of the digital line segment $L(p, q)$ that lies outside the shape contour is 0. Hence, $d_{pq}^{ir} = 0$. In this way, the concavity of the shape is the number of the line segments that are classified into the exterior and mixed. The concavity of the shape is $\hat{V} = \sum_{i=0}^{r-2} \sum_{j=0}^{c-1} h(i, j)$.

The DIR descriptors of the convex and the concave shapes are shown in Figure 4.7. From Figure 4.7 (a), it is obvious that the distribution of the line segments computed from the convex shape is in the upper part of the histogram. On the other hands, the distribution is in the middle of part of the histogram for the concave shape as depicted in Figure 4.7 (b).



(a) An example of the DIR descriptor of a convex shape.



(b) An example of the DIR descriptor of a concave shape.

Figure 4.7: The DIR descriptor of the convex and the concave shape. The x -axis represent the Euclidean distance and the y -axis represent the DIR. The brightness of each pixel represents the number of points in the block. Bright pixel represents a high number of line segments.

4.3.5 Computational Complexity

The DIR descriptor of a given set of m arbitrary points set P of the boundary of a simple polygon can be computed in $O(m^2)$ time. The time complexity for measuring the similarity of a pair of shapes using DIR descriptor is $O(rc)$, where rc is the descriptor size. Given a query object Q , the time complexity for retrieving the similar objects from the database using a naive method is $O(Orc)$ where O is the database size.

In this work, we apply the Locality Sensitive Hashing method proposed by Gionis *et al.*[42]. Using the LSH, we retrieve a set of elements and compare the descriptor for each element against the query. In theory, the candidate size depends on the table size L and the key size K , see Section 2.3.4. Let δO be the cardinality of the elements retrieved from LSH where δ is a compression ratio. Therefore, the retrieval time complexity is sublinear in O which is $O((\delta O)rc)$.

4.3.6 Experimental Result

The performance of DIR descriptor is demonstrated using the MPEG7 CE-Shape-1 [61] and Kimia's 99 [80] datasets. We conduct two experiments. The first one is to demonstrate the descriptor efficiency for an object recognition propose using Kimia's 99 [80] dataset. The other is to shows the efficiency when retrieving objects from a large size database using MPEG7 CE-Shape-1 [61] dataset. In the second experiment, we also show the efficiency of the DIR descriptor when integrating to an indexing data structure that is a Locality Sensitive Hashing (LSH).

Setting of the experiment

The boundary pixels of each object are extracted and then 1,000 points uniformly selected. For the distance interior ratio histogram matrix, we choose $c = 64$ and $r = 32$.

The experiments are conducted on a standard desktop machine with the Intel Core i7 3.4 GHz CPU and 16 GB of memory.

Recognition
























































We compare the recognition accuracy of DIR to the following established methods: Shape Context (SC) [14], Inner Distance Shape Description (IDSC) [63], D2 Distance Distribution (D2) [70], and Structural Feature Histogram Matrix (SFHM) [97]. The recognition is done on the Kimia’s 99 shape dataset consisting of 9 classes, and each containing 11 objects. For each query, we count the 10 closest matches that are in the same class as the query. The self-match is not included in the count.

Table 4.1: Comparison of the recognition result to other descriptors.

Method	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th
D2	95	88	82	77	76	70	65	55	58	35
SFHM [97]	96	96	91	89	84	84	75	77	56	55
SC [63]	97	91	88	85	84	77	75	66	56	37
DIR	97	98	90	91	81	74	79	70	61	49
IDSC+DP[63]	99	99	99	98	98	97	97	98	94	79

The comparison of the recognition rates is summarized in Table 4.1. The result shows that using the DIR descriptor 97 out of 99 of the most similar retrieved objects are from the same class. Comparing the overall result, our shape descriptor achieves a higher recognition rate than SC. On the other hand, the three most similar objects retrieved using the IDSC are always from the same class. Also, the recognition rate of the DIR is lower than the one of the IDSC descriptor. The comparison of the five closest objects using the DIR and the D2 descriptors can be found in Table 4.2.

Table 4.2: Comparison of the result between the descriptors D2 and DIR.

Input	Method	1 st	2 nd	3 rd	4 th	5 th
	D2					
	DIR					
	D2					
	DIR					
	D2					
	DIR					
	D2					
	DIR					
	D2					
	DIR					

Retrieval

The retrieval rate of our method is compared to the following methods: D2, SC, IDSC, and Contour Point Distribution Histogram (CPDH). We use the MPEG7 CE-Shape Part B [61] dataset which consists of 1,400 images from 70 classes and each class containing 20 shapes. Every image in the dataset is matched with every other image and among the top 40 most similar images from the same class are counted. The self-match is included in the count. So, at most 20 out of 40 candidates are correct hits. The retrieval accuracy is reported as a percentage of the maximum possible number of correct retrievals, which is 28,000 (1,400 shapes \times 20 correct retrieval per shape). This percentage score is called the *Bull's Eye Percentage* (BEP).

In this experiment, we aim to measure the efficiency of the proposed descriptor. For the SC and the IDSC descriptors, we use a shape context distance (D_{sc}) [14] as a similarity measurement. The detail of the computation can be found in equation 2.2 of Section 2.3.5.

The result in Table 4.3 shows that our proposed method achieves a retrieval rate of 73.72% which is higher than SC and IDSC achieved [63]: even if the size of DIR descriptor is much smaller.

Table 4.3: Comparison of the retrieval rate. For SC, IDSC, and CPDH, shape context distance (D_{sc}) [14] is used.

Descriptor	D2	SC [14]	IDSC [63]	CPDH [84]	DIR
BEP (%)	50.18%	64.59%	68.83%	71.89%	73.72%

Applying with Locality Sensitive Hash

To demonstrate the efficiency of the proposed descriptor when integrated to an indexing data structure in order to support a fast query, we retrieve objects via a Locality Sensitive

Hash (LSH). The DIR descriptor of each object from the MPEG7 CE-Shape [61] dataset is stored in the LSH.

The LSH framework being applied in this experiment is the one proposed by Gionis *et al.*[42] and an implementation can be found in [72]. Table 4.4 shows the comparison of the size of the elements retrieved from the LSH according to the table size L and the key size K and the BEP score. The table size L and the key size K are set as shown in the table; the bucket size, Bin , is \sqrt{O} . If the number of objects in one bucket exceeds the limit, an object is randomly removed from the bucket.

Given a query object Q , a set of objects that are in the same bucket as the shape signature of Q is retrieved from each table and stored in a set X . Let \mathbf{C} be a candidate set of objects such that $\mathbf{C} \subset X$ and \mathbf{C} does not contain any duplicate objects. Every object in \mathbf{C} is compared to Q using $dist_{DIR}$. The 40 most similar objects in \mathbf{C} are counted. The retrieval score is measured using the BEP score.

The retrieval result is shown in Table 4.4. From the result, we notice that the result is better than using the naive method. For example, the retrieval rate is 73.74% with setting $L = 60$ and $K = 64$. The retrieval rate become 73.80% when using $L = 80$ and $K = 64$. The average size of a candidate set is 573 which is very small as compared to the database size. The average retrieval time as shown in the table is much faster than the naive method which averagely takes 2.975 seconds.

4.4 Local Distance Interior Ratio

The DIR descriptor represents the global feature of the shape using two values that is the Euclidean distance and the interior ratio. Though the DIR descriptor is a rich descriptor, the locality information between a point and its boundary is ignored. The functional of the locality is for solving the correspondence problem, i.e., finding counterpart points on two shape contour [47]. For this reason, we extend the idea of the DIR descriptor in order to

Table 4.4: The retrieval result using different parameter L and K .

Table Size L	Key Size K	Candidate size	BEP (%)	Time (sec.)
2	64	45	23.4	0.1
4	64	90	32.99	0.177
8	64	191	51.81	0.46
10	64	244	56.15	0.53
20	56	433	68.72	0.91
30	64	517	71.36	1.15
40	56	662	72.90	1.41
60	64	784	73.74	1.56
80	64	868	73.80	1.77

improve the quality of the descriptor.

To preserve the locality of each point of the boundary, we present a shape descriptor called a *Local Distance Interior Ratio* (LDIR) which is derived from the DIR descriptor. The DIR descriptor is attached to a feature point in order to present the relation of the distance between a point and the others.

4.4.1 Distance Feature

Given a set $P = \{p_1, \dots, p_m\}$ of m arbitrary points on the boundary of a simple polygon. Let $P' = \{p'_1, \dots, p'_a\}$, where $P' \subset P$, be a set of a feature points uniformly selected from P , see Figure 4.8 (a).

Let $[p'p]$ be a line segment from $p' \in P'$ to a feature point $p \in P$ where $p' \neq p$. Two distance features, i.e., a Euclidean distance and a Distance Interior Ratio (DIR), are used

to describe the relation of a feature point and the shape boundary. Let $d_{p'p}$ denote the Euclidean distance of $[p'p]$. The distance interior of $[bp]$, which is the fraction of the line segment lying inside the polygon and the length of the line segment, is denoted by $d_{p'p}^{ir}$. The detail of the computation of $d_{p'p}$ and $d_{p'p}^{ir}$ can be found in Section 4.3.1.

Definition 4.4.1. *The description of the shape using the classified distances of the boundary points P is a list of DIR descriptors computed from the a feature points. We used $\mathcal{H}(P)$ to denote a list of DIR descriptors computed from a feature points. $\mathcal{H}(P)$ can be written as:*

$$\mathcal{H}(P) = \{H_P(1), \dots, H_P(a)\}$$

Definition 4.4.2. *Multiplicity of the the classified distances computed from a feature point p' of each block is denoted by $h'_p(u, v)$, where $0 \leq u \leq c - 1$ and $0 \leq v \leq r - 1$. We used $H_P(p')$ which is a $c \times r$ histogram matrix to denote the distribution of the classified distances between a feature point $p' \in P'$ and a set of the boundary points P . $H_P(p')$ can be written as:*

$$H_P(p') = \begin{bmatrix} h'_p(0, 0) & \dots & h'_p(0, c - 1) \\ h'_p(1, 0) & \dots & h'_p(1, c - 1) \\ \vdots & \ddots & \vdots \\ h'_p(r - 1, 0) & \dots & h'_p(r - 1, c - 1) \end{bmatrix}$$

The two distance features are computed from a feature point $p' \in P'$ to every point $p_i \in P$ where $1 \leq i \leq m$ as shown in Figure 4.8 (b). The shape information is transformed to a feature space \mathbb{F} , where each line segment $[p'p_i]$ is represented by the point $(d_{p'p_i}, d_{p'p_i}^{ir})$. More precisely, the x -axis represents the Euclidean distances and the y -axis corresponds the DIR.

The DIR descriptor of a feature point $p' \in P'$ is obtained by compressing the shape feature into a two-dimensional histogram. The shape feature is obtained by equally divide \mathbb{F} into $c \times r$ blocks. The interval of each block on the x -axis is $\lfloor (d_{max} - d_{min})/c \rfloor$ and the y -axis is $\lfloor (d_{max}^{ir} - d_{min}^{ir})/r \rfloor$.

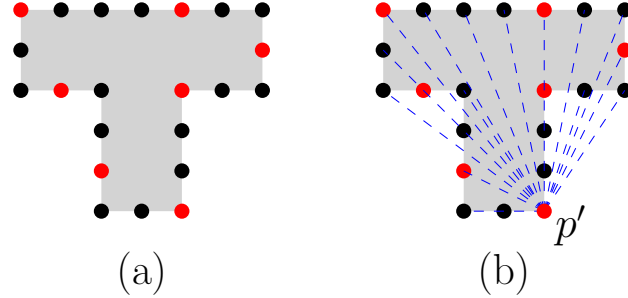


Figure 4.8: (a) A set of m arbitrary points on the boundary of P where the red points are the selected feature points. (b) A set of line segments from a point p' to every other points $p \in P$.

Let (u, v) denotes a block in \mathbb{F} where u and v are the index on the x -axis and the y -axis respectively. The interval of each block is denoted by $[[x_u, x_{u+1}], [y_v, y_{v+1}]]$. Let $h'_p(u, v)$ be the number of points in the feature space \mathbb{F} that fall into $[[x_u, x_{u+1}], [y_v, y_{v+1}]]$. To avoid scaling problem, $h'_p(u, v)$ is normalized by dividing the total number of points in the feature space. We have

$$h'_p(u, v) = \frac{2}{m(m-1)} \sum_{i=0}^m \sum_{j=1}^{m-1} \hat{g}(d_{ij}, d_{ij}^{ir}, x_u, x_{u+1}, y_v, y_{v+1}),$$

where,

$$\hat{g}(d_{ij}, d_{ij}^{ir}, x_u, x_{u+1}, y_v, y_{v+1}) = \begin{cases} 1, & \text{if } d_{ij} \in [x_u, x_{u+1}] \text{ and } d_{ij}^{ir} \in [y_v, y_{v+1}] \\ 0, & \text{otherwise.} \end{cases}$$

4.4.2 Similarity Measurement

Given the LDIR descriptors $\mathcal{H}(P)$ and $\mathcal{H}(Q)$ computed from shapes P and Q , the similarity of the two shapes is the summation of the distance of the DIR descriptor of each point.

Consider an LDIR descriptor $H_P(i) \in \mathcal{H}(P)$ of a feature point p'_i and $H_Q(j) \in \mathcal{H}(Q)$ of a feature point q'_j . The similarity of the two descriptions is measured using L_1 norm. Let

$dist_{DIR}(H_P(i), H_Q(j))$ be the distance between the two descriptors. It is computed as,

$$dist_{DIR}(H_P(i), H_Q(j)) = \sum_{k=0}^{r-1} \sum_{l=0}^{c-1} |h_{p_i}(k, l) - h_{q_j}(k, l)|$$

Let $C = \{c_1, \dots, c_a\}$ be a set of indices of the feature points in P which assigns to each feature point in Q if the distance $dist_{DIR}(H_P(c_k), H_Q(j))$ for $1 \leq k \leq a$ is minimum. It is,

$$c_j = i : \min_{1 \leq i \leq |\mathcal{H}(P)|} dist_{DIR}(H_P(i), H_Q(j))$$

Let $dist_{LDIR}(P, Q)$ denote the distance between $\mathcal{H}(P)$ and $\mathcal{H}(Q)$. Using the obtained set of indices C , we have

$$dist_{LDIR}(P, Q) = \sum_{j=1}^a (dist_{DIR}(H_P(c_j), H_Q(j))) \quad (4.2)$$

4.4.3 Computational Complexity

The LDIR descriptor of any given P can be computed in $O(ma)$ time, where m is the number of vertices on the shape contour P and a is the number of feature points P' . The time complexity for computing the distance between the two descriptors is $O(a^2rc)$, where rc is the descriptor size.

Given a query object Q , the time complexity for retrieving the similar objects from the database of size O using a naive method is $O(Oa^2rc)$.

4.4.4 Experimental Result

Setting of the experiment

The experiment is conducted using the MPEG7 CE-Shape-1 [61] and Kimia's 99 [80] shape datasets. The MPEG7 CE-Shape [61] dataset consists of 1400 images from 70 classes and 20 shapes for each class. The Kimia's 99 shape dataset consisting of 9 classes, and each containing 11 objects. The boundary pixels of each object are extracted and then 800 points

uniformly selected. 80 features points are selected from the 800 points. For the size of the DIR descriptor of each feature point, we choose $c = 8$ and $r = 6$. The experiments are conducted on a desktop machine with the Intel Core i7 3.4 GHz CPU and 16 GB of memory.

Recognition

We compare the recognition accuracy of Local Distance Interior Ration (LDIR) descriptor to the following established methods: Distance Interior Ration (DIR), Shape Context (SC) [14], and Inner Distance Shape Context (IDSC) [63]. The dataset being used in this experiment is the Kimia’s dataset [80].


































For each shape, we check whether the 10 closest matches are in the same class as the query. For LDIR, we find the correspondences using the method as explained in Section 4.4.2. The similarity of the two shapes is measured using $dist_{LDIR}$. For detail of calculation, see equation 4.2. To solve the correspondence, SC employs a Hungarian method [14] and IDSC employs a dynamic programming [63].

The recognition rates is summarized in Table 4.5. The recognition result of LDIR is better than DIR and SC. The two closest matched of LDIR are always from the same class. The overall result of LDIR is lower than IDSC. Table 4.6 shows the comparison of the five closest results using DIR and LDIR. From the result, LDIR is able to recognize an occlusion image as shown in the last input. For comparison, see Table 4.6.

Table 4.5: Comparison of recognition result.

Method	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th
SC [63]	97	91	88	85	84	77	75	66	56	37
DIR	97	98	90	91	81	74	79	70	61	49
LDIR	99	99	98	95	93	93	81	76	59	55
IDSC[63]	99	99	99	98	98	97	97	98	94	79

Table 4.6: Comparison of the five closest objects retrieved by using DIR descriptor and LDIR descriptor.

Input	Method	1	2	3	4	5
	DIR					
	LDIR					
	DIR					
	LDIR					
	DIR					
	LDIR					

Retrieval

We compare the retrieved result using a larger size of the dataset, i.e. the MPEG7 CE-Shape-1 [61]. The retrieval result is measured using Bull's Eye Percentage (BEP). Every image in the dataset is matched with every other image using the similarity measurement in Section 4.4.2 and among the top 40 most similar images which are from the same class are counted. So, at most 20 out of 40 candidates are correct hits. To compare the efficiency of the description, we use a shape context distance [13, 14] for SC, IDSC, and CPDH.

The BEP score is summarized in Table 4.7. Using only the descriptor, our proposed method achieves the retrieval rate of 78.27% which is more than other shape descriptor

except the distance set [47]. The average retrieval time of LDIR is 11.09 seconds.

Since the retrieval time is depended on the number of feature points, we conduct an experiment using different number of feature points. We select 100, 80, and 50 features points from the boundary points. The retrieval score and the time complexity is summarized in Table 4.8. From the table, it is clear that the retrieval time decreases as the size of the feature points smaller. A set of 80 feature points gives the highest score. The BEP is slightly dropped when using 50 feature points. However, comparing to the other descriptors as summarized in Table 4.6, it gives better result.

Table 4.7: Comparison of the retrieval experiment. The efficiency of the retrieval result is measured using Bull’s Eye Percentage (BEP).

Method	SC	IDSC	CPDH [84]	DIR [31]	LDIR	Distance Set
BEP (%)	64.92%	68.84%	71.89%	73.72%	78.27%	78.38%

Table 4.8: Comparison of the Bull’s Eye Percentage (BEP) and retrieval time using of 100, 80, and 50 feature points.

# Feature Points	100	80	50
BEP (%)	78.15%	78.27%	77.59%
Time (sec.)	25.86	11.09	4.21

Applying with Locality Sensitive Hash

We compare the result of LDIR and SC using MPEG7 CE-Shape-1 [61] dataset. Given a query, we want to a set of objects that best match from among the shape in the database. Each of the a query feature points from the query shape casts a vote for one of the objects in the dataset. We retrieved all objects that obtain a vote. The number of feature points

being applied in this experiment is $a = 80$. The parameters for LSH is as follow: the table size $L = 100$, the key size $K = 8$, the bucket size $Bin = \sqrt{Om}$.

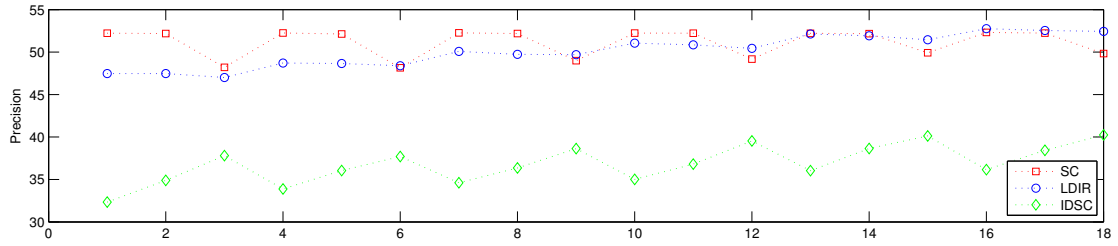
For the local descriptor, i.e., LDIR, SC, and IDSC, the similar objects are retrieved according to the vote. Given a descriptor of a query object Q , each of the descriptor of a feature points q' casts a vote for one of the 1,400 objects. For all the feature points, the votes of each object are sorted and the 40 best match that receives the most votes are retrieved. In this experiment, we determine a query feature's vote by finding its nearest neighbor from LSH. The parameter of the LSH is as follow: The table size $L = \{50, 60, 70, 80, 90, 100\}$ and the key size $k = \{8, 16, 32\}$.

The retrieval efficiency is measured using precision and recall. Precision is defined as the ratio of the number of retrieved relevant shapes to the total number of retrieved shapes. Recall is defined as the ratio of the number of retrieved relevant shapes to the number of relevant shapes in the whole dataset. To compare the overall result, F-measure is applied in order to measure the test's accuracy which is defined as an average of the precision and the recall value.

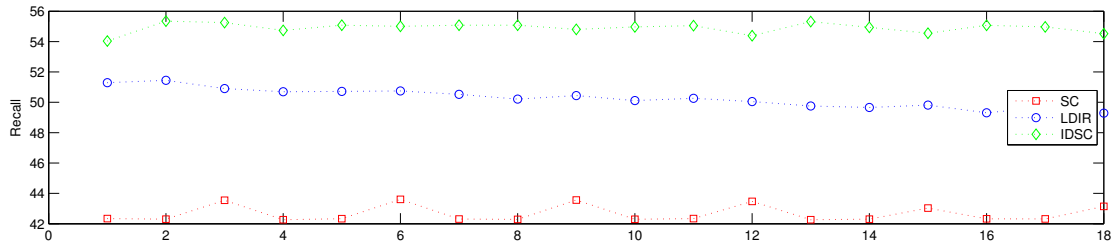
The comparison of the experimental result of LDIR, SC, and SC is shown in Figure 4.9. Using $L = 100$ and $k = 8$, the LDIR achieves slightly higher precision that is 52.76%, SC achieves 52.32%, and IDSC achieves 36.16%. Comparing the performance using recall value, LDIR acquires 49.31% while SC and IDSC obtains 42.33% and 55.08% respectively. The comparison of the result of the F1-value among the three methods is shown in Figure 4.9 (c). The LDIR descriptor yields a better F1-measure than the other methods in every setting of the LSH's parameters.

4.5 Discussion and Conclusion

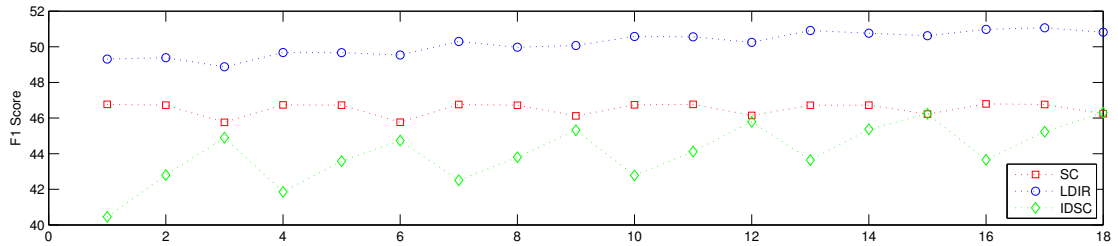
In this chapter we target the accuracy when retrieving a set of similar object using the shape descriptor as a similarity measurement. We are interested in using the distance-based



(a) Precision



(b) Recall



(c) F1 measure

Figure 4.9: Comparison of the result of LDIR and SC when retrieving via LSH. The x -axis represents the parameter of the LSH. For example, $x = 1$ refers to $L = 50$ and $k = 8$, $x = 2$ refers to $L = 50$ and $k = 16$, $x = 3$ refers to $L = 50$ and $k = 32$, and so on.

descriptor since it is compact and can be integrated to an indexing data structure easily. One problem of pairwise distribution is the existence of the homometric pair objects which affects the retrieval accuracy.

To improve the quality of the retrieved result, we present two shape descriptors: the DIR and the LDIR descriptors. The idea of the descriptors is to describe the shape using the relative distribution of the Euclidean distance and the Distance Interior Ratio (DIR) computed from the points on the shape contour. The DIR represents the relative distance distribution among points on the shape contour. On the other hands, the LDIR represents the relative distance distribution between a feature point and the other points on the shape contour. In other words, the DIR represents the global shape information while the LDIR preserves the shape locality using the DIR descriptor with respect to a feature point.

We demonstrate the efficiency of the proposed descriptor using experiments and compare with the state-of-art distance-based descriptors that are the shape context (SC) and the inner distance shape context (IDSC). The DIR and LDIR perform better than the other descriptor when retrieving via an indexing data structure which we apply the locality sensitive hashing (LSH). However, the recognition rate is slightly worse than the IDSC. One reason is that the IDSC apply dynamic programming which considers the order of the points.

Chapter 5

Framework for Shape-based Image Retrieval

5.1 Introduction

Given the two set of boundary points P and Q , where $m = |P|$ and $\hat{t} = |Q|$, the similarity of the two shape is measured using the modified Hausdorff distance $H_{mod}(P, Q)$ as explained in equation 2.1. In order to compute the optimal Hausdorff distance, Q needs to be transformation using a transformation $T = T[p, p', q, q']$ which is obtained by aligning a line segment pp' to qq' . To obtain the optimal Hausdorff distance, we need to find $T = T[p_i, p_l, q_j, q_k]$ for $1 \leq i < l \leq m$ and $1 \leq j < k \leq t$ which maps Q to the most similar of P . Therefore, the time complexity for computing the Hausdorff distance is $O(m^2\hat{t}^2mt)$ where $O(m^2\hat{t}^2)$ is for solving the optimal transformation.

The problem of employing the Hausdorff distance in the retrieval system is not only the time complexity for one comparison but also the retrieval time for one query is linear in the database size. A query has to be compared to each shape in the database. Unfortunately, the search space of shapes based on the shape distance is not easy to organize to support

fast query.

In this chapter, we target two problems when applying a shape distance as a similarity measurement in an image retrieval system:

1. The time complexity for computing shape distance between two shapes.
2. The retrieval time complexity when applying shape matching in the system.

For each problem, we propose a framework for improving the computational time complexity.

5.2 Summary of the result

We propose two methods for improving the time complexity when computing the Hausdorff distance and retrieving similar shapes from the database. The idea of the two methods is to ignore elements which are very different from the query. High accuracy similarity measurement is applied to the rest of the elements for ranking the retrieved result. The summary of the result of each method is as follow:

1. The first method is for improving the time complexity when computing the Hausdorff distance between the two objects. The idea of “correspondence” point [14, 86, 47] is used for estimating transformation that maps two boundaries. By selecting \mathcal{K} correspondence points, the time complexity for computing the similarity of the two shape boundary P and Q where $m = |P| = |Q|$ and $\mathcal{K} \ll m$ is $O(\mathcal{K}^2 m^2)$. We demonstrate the performance of the proposed method using the experiments. The result shows that the proposed method gives better result than the Hausdorff distance.
2. The second method is for reducing the processing time of one query of a shape-based image retrieval framework. The main idea is to avoid processing unneeded data such as the object which is totally different from the query. A set of potentially similar shapes called “candidates” is retrieved via an efficient search structure based on a similarity

of the DIR shape descriptor. The retrieved result is ranked according to the Hausdorff distance which is applied to each candidate set member. By ignoring unneeded similarity measurement computation of the database elements, the time complexity for retrieving objects is sub-linear in the database size.

5.3 Computing shape distance using correspondence

Let P and Q denotes a set of arbitrary points on the boundary of the two simple polygons, where $m = |P| = |Q|$ as shown in Figure 5.1(a). To compute the similarity of the two polygons using the Hausdorff distance, we need to find a transformation T such that $H_{mod}(P, T(Q))$ is minimized. Let $T = T[p, p', q, q']$ be a transformation that mapped an edge $(p, p') \in P$ and $(q, q') \in Q$. For illustration, see Figure 5.1 (c).

In order to answer how “similar” they are, we need to find a transformation which gives the closest Hausdorff distance between the two polygon. In this way, we need to compute the Hausdorff distance using all possible transformations. There are $O(m^2)$ edges of $(p, p') \in P$. For each edge (p, p') , it maps to all $O(m^2)$ edges $(q, q') \in Q$ as shown in Figure 5.1 (d). To compute the optimal Hausdorff distance, we need to compare P and Q using at most $O(m^4)$ transformations. Therefore, the optimal Hausdorff distance can be computed in $O(m^6)$ using a naive method. By using the parametric search method, the time complexity for comparing the two shapes is $O((m)^4 \log^2 m^2)$ [29]. However, the parametric search method is not practical.

It is easy for human to align a shape Q as shown in Figure 5.1(a) to the most similar part of P . Examples of pair of points of P and Q that can be mapped to each other are presented in Figure 5.1(b). We call this pair of points as correspondence.

To compute the shape distance, if the correspondence points on the two shapes are known then one shape can be overlaid to the most similar part of the other without spending computation sources for unneeded pair of points such as the one illustrated in Figure 5.1(d).

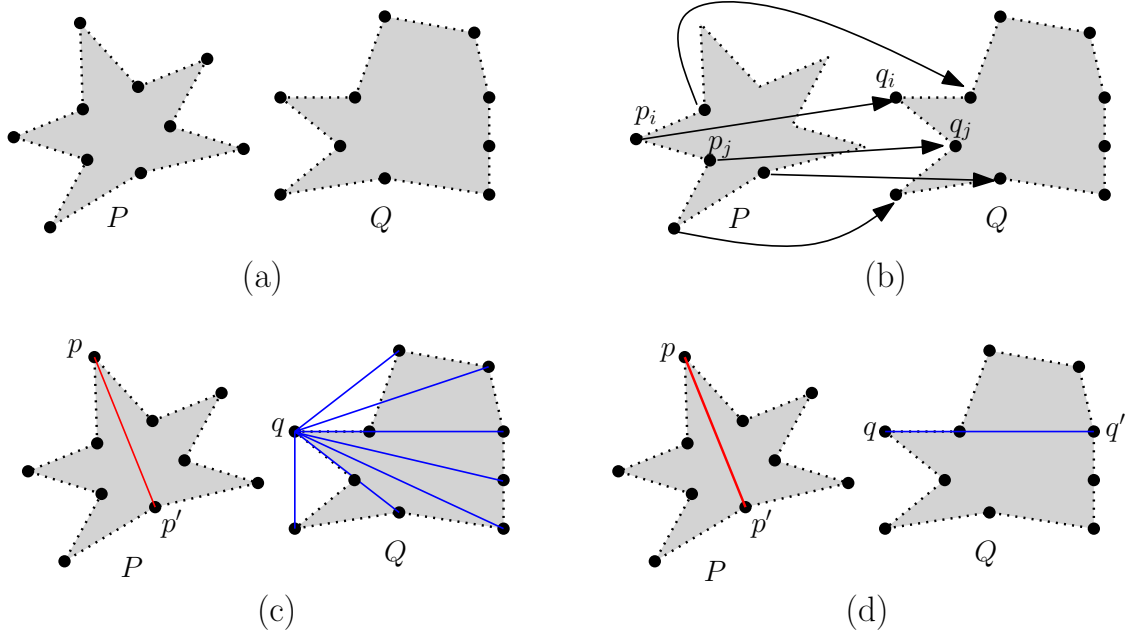


Figure 5.1: (a) Arbitrary points on the boundary of two polygons, (b) The correspondence points of the two shapes. (c) A pair of alignment edges $(p, p') \in P$ and $(q, q') \in Q$. (d) The alignment that map (p, p') to every edges (q, q') .

Therefore, the time complexity for computing the shape distance of the two shapes is reduced.

5.3.1 Solving Correspondence

We apply the Local Distance Interior Ratio (LDIR) descriptor as a spatial data which is attached to each feature point. The spatial data is used for solving the correspondence of the two shapes. The detail of the descriptor computation can be found in Section 4.4.

Given a set of arbitrary points on the shape contour of a polygon P , a set of a feature points $P' = \{p'_1, \dots, p'_a\}$ is uniformly selected from P . Similarly, a set of arbitrary points on the shape contour of a query polygon Q is given and a set of a feature points Q' is defined analogously. The LDIR descriptor of both P and Q is computed for each feature point which are $\mathcal{H}(P) = \{H_P(p'_1), \dots, H_P(p'_a)\}$ and $\mathcal{H}(Q) = \{H_Q(q'_1), \dots, H_Q(q'_a)\}$ respectively.

To establish the corresponding points between P' and Q' , we need to find a one-to-one matching in which the total cost is minimized. A point $p' \in P'$ is said to be a correspondence point of $q' \in Q'$ if $H_P(p') \approx H_Q(q')$. A cost C_{ij} between the LDIR descriptor of a feature point p'_i and q'_j is $dist_{DIR}(H'_P(i), H'_Q(j))$ which is measured using L_1 norm and can be computed as follow:

$$C_{ij} = dist_{DIR}(H'_P(i), H'_Q(j)) = \sum_{a=0}^{r-1} \sum_{b=0}^{c-1} |h_{p'_i}(a, b) - h_{q'_j}(a, b)|$$

where r and c are the descriptor size. The cost is computed for every pair of feature point q'_j, p'_i for $1 \leq j \leq a$ and $1 \leq i \leq a$.

Definition 5.3.1. *Given a cost matrix C between Q' and P' , a pair of matching point of (p', q') where $p' \in P'$ and $q' \in Q'$ is solved using Hungarian method. For every point $q' \in Q'$, let A be a list of pairs of one-to-one matching point between Q' and P' which is,*

$$A = \{(p'_1, q'_1), \dots, (p'_a, q'_a)\}$$

To solve a one-to-one matching of the feature points of Q' and P' , we apply Hungarian method which is an instance of the weighted bipartite matching problem and can be solved in $O(a^3)$ time [57] where a is the number of the feature points of P' and Q' . The cost C_{ij} is defined as the weight between the feature points of $p'_i \in P'$ and $q'_j \in Q'$. The Hungarian method finds a permutation π that minimize the total cost of matching, that is:

$$\tilde{C}_g(\pi) = \sum_j^a C_{p'_{\pi(j)}, q'_j}$$

Therefore, the list of one-to-one matching feature points of P' and Q' is $A = \{(p'_{\pi(1)}, q'_1), \dots, (p'_{\pi(a)}, q'_a)\}$ in which $\pi(j)$ is a permutation.

Definition 5.3.2. *A point p'_i is called a a correspondence point of q'_j if $(p'_i, q'_j) \in A$ and the cost C_{ij} is minimum for $1 \leq i \leq a$. We define D to be a set of correspondence point points between Q' and P' which is defined as follow,*

$$D = \{(p'_{\pi(j)}, q'_j) \mid (p'_{\pi(j)}, q'_j) \in A \wedge cost(C_{\pi(j)j})\}$$

where,

$$\text{cost}(C_{\pi(j)j}) = \begin{cases} 1, & \text{if } C_{\pi(j)j} == \min_{1 \leq i \leq m} C_{ij}, \\ 0, & \text{otherwise.} \end{cases}$$

We check the cost $C_{\pi(j)j}$ for each pair of matching point $(p'_{\pi(j)}, q'_j) \in A$ whether or not it is minimum among C_{ij} for $1 \leq i \leq a$. If $C_{\pi(j)j}$ of $(p'_{\pi(j)}, q'_j)$ is minimum, $(p'_{\pi(j)}, q'_j)$ is added to the correspondence point list D . After obtaining the correspondence point list D , the cost of matching of the two shape P and Q using the LDIR descriptor is the average cost of matching of the correspondence points $(p'_{\pi(j)}, q'_j) \in D$ for $1 \leq j \leq |D|$, that is

$$\text{dist}(P, Q) = \frac{1}{|D|} \sum_i^{|D|} C(d_i) \quad (5.1)$$

where, $d_i = (p'_{\pi(i)}, q'_i)$ and $d_i \in D$.

Computational Complexity

Given a feature points P' and Q' of the boundary point sets P and Q respectively. The LDIR descriptors of P and Q are $\mathcal{H}(P) = \{H_P(1), \dots, H_P(a)\}$ and $\mathcal{H}(Q) = \{H_Q(1), \dots, H_Q(a)\}$ respectively, where the size of $H_P(\cdot)$ is $r \times c$. The cost matrix C is computed in $O(a^2rc)$. A set of one-to-one matching point A between P' and Q' is solved using Hungarian method which can be computed in $O(a^3)$ time. The correspondence points of P' and Q' can be solved in $O(a)$ times using the one-to-one matching set A . The time complexity for finding the correspondence between P' and Q' is $O(a^3)$.

5.3.2 Computing shape distance

After solving a set of correspondence points D between a set of feature points Q' and P' , the shape distance is computed using the transformation estimated from D . In this research, we apply a modified-Hausdorff distance as a similarity measurement. The detail of the computation can be found in Section 2.3.1.

Definition 5.3.3. Let $T = T[p'_i, p'_j, q'_i, q'_j]$ denotes the similarity map (translation, rotation, and scaling) that aligns $p'_i p'_j$ with $q'_i q'_j$ in which (p'_i, q'_i) and (p'_j, q'_j) are pairs of correspondence points in D .

Definition 5.3.4. Let $T(Q)$ be the transformed boundary points of Q where $T = T[p'_i, p'_j, q'_i, q'_j]$.

For each edge $p'_i p'_j$ from D , consider an alignment of $p'_i p'_j$ with a edge $q'_i q'_j$ from Q in which q'_i is a corresponding point of p'_i . q'_j and p'_j is defined analogously. The boundary point Q is transformed according to the transformation vector $T = T[p'_i, p'_j, q'_i, q'_j]$. An example of the transformation between P and Q using $p'_i p'_j$ and $q'_i q'_j$ is shown in Figure 5.2 (b).

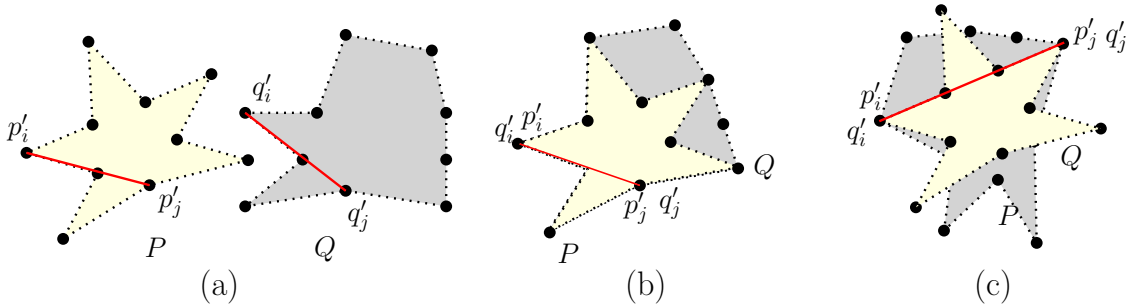


Figure 5.2: (a) Sets of m arbitrary points on the boundary of a polygon P and Q . A set of big dots indicates a set of a feature points. (b) Alignment of Q on P using a transformation $T = T[p'_i, p'_j, q'_i, q'_j]$. (c) Another example of the alignment.

Definition 5.3.5. Let $dist_{shape}(P, Q)$ be the shape distance between the polygons P and Q . Among all transformations T , $dist_{shape}(P, Q)$ is the modified-Hausdorff distance which is minimum.

Let $H_{mod}(P', T(Q'))$ be a modified-Hausdorff distance between P' and the aligned boundary points $T(Q')$. Among all transformations T , the shape distance between the two polygons P' and Q' is the modified-Hausdorff distance which is minimum. It is,

$$dist_{shape}(P', Q') = \min_{1 \leq i \leq |D|} \min_{1 \leq j \leq |D|} H_{mod}(P', T(Q'))$$

where $T = T[p'_i, p'_j, q'_i, q'_j]$ for $1 \leq i \leq |D|$ and $1 \leq j \leq |D|$.

Computational Complexity

By aligning the boundary points P and Q using the transformation $T = T[p'_i, p'_j, q'_i, q'_j]$ where p'_i is a corresponding point of q'_i and p'_j is the one of q'_j , the $dist_{shape}(P, Q)$ can be computed in $O(|D|^2 a^2)$.

It is easy to see that the time complexity for computing the shape distance depends on the cardinality of the correspondence points set D . By selecting the corresponding points which have the \mathcal{K} smallest cost, the time complexity for computing the shape distance becomes $O(\mathcal{K}^2 a^2)$ time.

5.3.3 Experimental Result

Setting of the experiment

In this experiment, we focus on the efficiency of the proposed algorithm for computing a shape-based image retrieval using a modified-Hausdorff distance as a similarity measurement.

This experiment is conducted using the Kimia's 99 [80] shape datasets which consists of 9 classes, and each containing 11 objects. The boundary pixels of each object are extracted and then 800 points uniformly selected. 80 features points are selected from the 800 points. For the size of the LDIR descriptor of each feature point, we choose $c = 8$ and $r = 6$. The experiments are conducted on a desktop machine with the Intel Core i7 3.4 GHz CPU and 16 GB of memory.

For each query shape, we compute the correspondence points to every other shape in the data set. For each pair of shapes, $\mathcal{K} = 5$ correspondence pair of points are selected and the shape distance is computed using the modified-Hausdorff distance. For each query, we count the 10 closest shapes that are in the same class as the query. The self-match is not included in the count.

Recognition

The recognition result is shown in Table 5.1. Due to the time complexity when computing the modified-Hausdorff distance, we apply an approximation scheme that 16 points on each shape boundary points are randomly selected and the transformation is computed based on the selected points. The result of the approximate modified-Hausdorff distance is shown in “Approx.Mod.Haus.” row. The “SC” represents the result using the shape context and the Hungarian method for solving the correspondence. The row “LDIR+Hung.” shows the result when using the LDIR descriptor and the Hungarian method for solving a one-to-one correspondence. The detail of the computation can be found in Equation 5.1. We also compare the result with the IDSC shape descriptor in which the descriptor is attached to 80 feature points and the dynamic programming is applied to solve the correspondence. The “LDIR+” shows the result when using the correspondence points for computing the shape distance that is $dist_{shape}(P, Q)$.

Table 5.1: Comparison of recognition result of min Hungarian method of LDIR.

















































Method	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th
Mod.Haus.	99	96	94	95	92	90	85	81	70	60
SC [63]	97	91	88	85	84	77	75	66	56	37
LDIR+Hung.	98	92	95	94	89	82	81	69	61	58
IDSC80+DP	98	95	95	95	91	90	85	78	66	59
LDIR+	99	98	95	97	94	94	84	77	67	56

From the result, it is easy to see that our proposed method performs as good as the approximate modified-Hausdorff distance. Using the corresponding points, the closest shape distance to the query is always from the same class. The five closest shapes using the approximate modified-Hausdorff distance, LDIR+, and SC are shown in Table 5.2. From

the five closest objects of the last query, we notice that the result from LDIR+ and SC are similar, excepts that the LDIR+ gives better relevant ranking of the retrieved result.

The average computation time for computing the approximate modified-Hausdorff distance of the two sets of feature points using the correspondence takes 0.391 seconds while the naive method takes 1.506 seconds.

Table 5.2: Comparison of the five closest objects retrieved objects.

Input	Method	1	2	3	4	5
	Mod.Haus.					
	LDIR+					
	SC					
	Mod.Haus					
	LDIR+					
	SC					
	Mod.Haus					
	LDIR+					
	SC					

5.4 Shape-based image retrieval framework

Given a set of boundary points Q of a query polygon where $|Q| = m$, we want to find objects in the database whose shapes resemble to Q using a geometric pattern matching metric such as a Hausdorff distance.

There are O objects in the database. To retrieve the k most similar objects in the database in a naive shape-based image retrieval, the retrieval time is linear in the database size and polynomial time for each comparison. Using the naive Hausdorff distance, the time complexity for each comparison is $O(Om^8)$. However, in information retrieval, query time should be independent of O because it is often quite huge.

We demonstrate that by not expending computational resources on unneeded accuracy of similarity, both the speed and the accuracy of retrieval can be increased. The proposed framework composes of three steps. The outline of each step is as follow:

1. **Preprocessing:** A shape feature of each shape in the database is stored in a data structure in order to support an efficient search. A feature vector is computed from a global-based shape signature and compressed using a statistical-based approach.
2. **Screening:** A global-base shape information is used in a screening stage in order to remove unneeded elements for further processing. For a given query, a shape signature of the boundary points is computed. A set of candidate shapes whose shape signature are similar to the query is retrieved via a search structure.
3. **Refining:** A refining process is used to rank the shapes from the candidate set in order to improve the relevance of the retrieved result. The Hausdorff distance and the corresponding point points are applied for measuring the similarity between the query and each candidate.

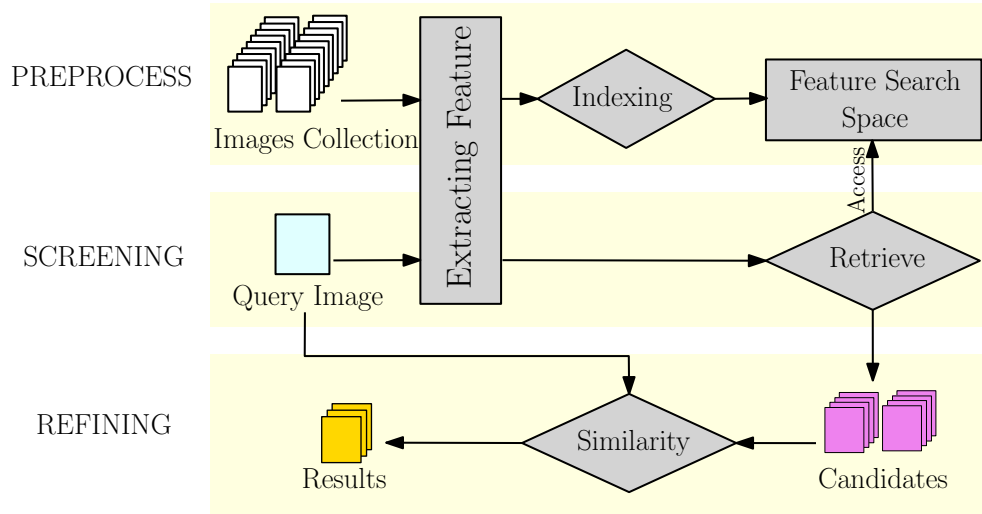


Figure 5.3: Overview of the 3-stages process: (1) Preprocessing, (2) Screening, and (3) Refining.

5.4.1 Preprocessing Stage

To support a fast query, we prepare a data structure of the shape feature of each element in the database. The outline of the preprocessing stage is illustrated in Figure 5.4. There are two data to be processed: (1) A search structure of the global-based shape signature $M(P)$ which is used in the screening stage; and (2) A set of spatial data shape descriptor $\mathcal{H}(P)$ is used for solving the correspondence in the refining stage. The Locality Sensitive Hashing (LSH) is applied as a search structure for storing the shape signature $M(P)$.

Global feature descriptor and indexing

Assume that the image segmentation has been done and a polygon P is represented as a set of n vertices uniformly selected from the shape contour. Given a set of m boundary points P , set of a feature points P' is uniformly selected. The DIR descriptor is used as a shape signature. On the other hand, the LDIR descriptor is used as the spatial data of the a feature points.

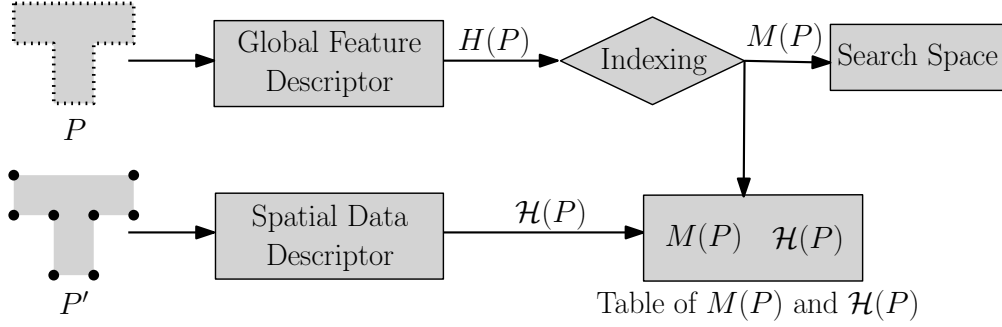


Figure 5.4: Outline of the preprocessing stage.

The shape boundary of P is described using the DIR descriptor which is a relative distribution of the Euclidean distance and the distance interior ratio. A multiplicity vector of the boundary points P is denoted by $H(P)$ which is a $c \times r$ histogram matrix of $h(u, v)$, where $0 \leq u \leq c - 1$ and $0 \leq v \leq r - 1$. The detail of the computation can be found in Section 4.3. $H(P)$ is written as:

$$H(P) = \begin{bmatrix} h(0,0) & h(0,1) & \dots & h(0,c-1) \\ h(1,0) & h(1,1) & \dots & h(1,c-1) \\ \vdots & \vdots & \ddots & \vdots \\ h(r-1,0) & h(r-1,1) & \dots & h(r-1,c-1) \end{bmatrix}$$

Definition 5.4.1. Shape signature of the boundary points is denoted by $M(P)$ which is derived by concatenating each row of the DIR descriptor $H(P)$. It can be written as:

$$M(P) = \{h(0,0), h(0,1), \dots, h(0,c-1), h(1,0), \dots, h(r-1,0), h(r-1,1), \dots, h(r-1,c-1)\}$$

The Local Sensitive Hashing (LSH) is used as a search structure in this work. A hash function $g_l(\cdot)$ for $1 \leq l \leq L$ is applied to the signature $M(P)$ in order to assign the index of the first-level hash table. For a hash table l , another hash function $h_{k,l}(\cdot)$ for $1 \leq k \leq K$ is applied for an index assignment of the second-level hash table. By using the two-level hash table, the shape signatures which are similar will be assigned to the same index for both hash tables.

The shape signature $M(P_i)$ and the index assignment is computed for every object P_i in the database where $1 \leq i \leq O$ and O is the database size.

Spatial data of a feature point

The spatial data which is used for solving the corresponding points on the two shapes is attached to each feature point. The LDIR descriptor which is a set of the DIR descriptor is applied in this work. For each feature point $p' \in P'$, a DIR descriptor describes the relative distribution of the Euclidean distance and the distance interior ratio between p' and every other boundary points.

The spatial data of P' is denoted by $\mathcal{H}(P) = \{H(1), \dots, H(a)\}$ where $H(i)$ is a DIR descriptor computed from a feature point $p'_i \in P'$ for $1 \leq i \leq a$. The spatial data of a feature points is stored in a table associated with the signature $M(P)$.

5.4.2 Screening Stage

The advantage of the global shape feature is that the shape can be represented using a vector feature which can be stored in a search space and also the similarity between the two vectors can be computed efficiently. According to these advantages, we utilize the global shape feature for removing irrelevant element in this stage. The overview of this stage is illustrated in Figure 5.5.

Retrieving from the search structure

Given a query which is a set of boundary points Q where $|Q| = m$, we uniformly select a set of feature a points Q' from Q . The DIR descriptor $H(Q)$, the shape signature $M(Q)$, and the LDIR descriptor $\mathcal{H}(Q)$ are computed by using the method as in the preprocessing stage which the size of the descriptor and the signature are set to be the same as the elements in the search structure. See Section 5.4.1.

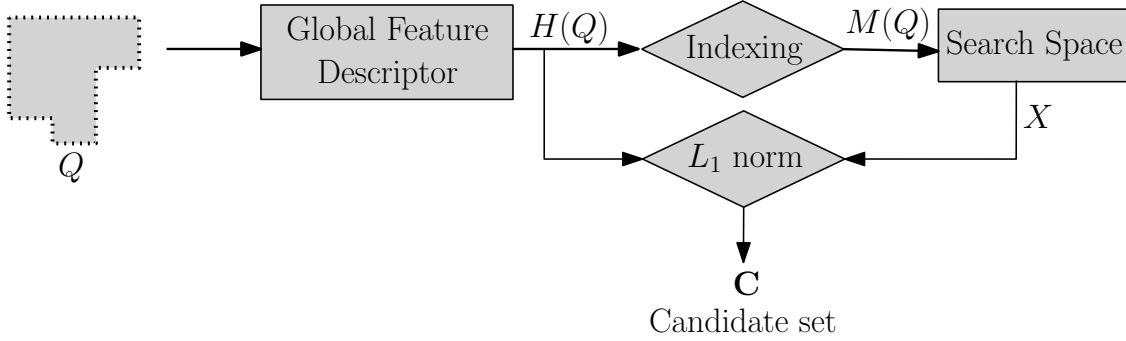


Figure 5.5: Outline of the screening stage.

Definition 5.4.2. Let X be a set of element retrieved from the search structure. $X = \{x_1, \dots, x_b\}$ where $|X| = b$ and $\mathcal{H}(x_i)$ for $1 \leq i \leq b$ denotes the classified distance shape descriptor of each element.

Similar to the preprocessing stage, the first-level hash function $g_l(\cdot)$ for $1 \leq l \leq L$ and the second-level function $h_{k,l}(\cdot)$ for $1 \leq k \leq K$ are applied to the signature $M(Q)$. The elements that are in the same bucket as $M(Q)$ is retrieved as stored in a set X . We assume that there are no duplicate elements in X .

Selecting candidate

We roughly select a set of candidate elements according to the similarity of the global shape feature. The similarity of the descriptors between each elements in X and $H(Q)$ is measured by using $dist_{DIR}$ which is L_1 norm. The k elements in X whose DIR descriptor are closest to Q are retrieved as a candidate set.

Definition 5.4.3. Given a shape descriptor $H(Q)$ of a query object Q and a shape signature $H(x_i)$ of an object $x_i \in X$ for $1 \leq i \leq b$, x_i is a candidate of Q if $H(x_i) \approx H(Q)$.

Definition 5.4.4. Let \mathbf{C} be a set of k candidates, where k is given. $\mathbf{C} = \{c_1, \dots, c_k\}$ is a set of candidate elements taken from the k^{th} smallest L_1 norm from X .

To select the k objects, the $dist_{DIR}$ is computed between the DIR descriptor of the query and each element $x_i \in X$ where $1 \leq i \leq b$. The $dist_{DIR}$ between $H(Q)$ and $H(x_i)$ is computed as follow,

$$dist_{DIR}(H(Q), H(x_i)) = \sum_{l=0}^{r-1} \sum_{k=0}^{c-1} |h_{x_i}(l, k) - h_Q(l, k)| \quad (5.2)$$

After computing and sorting $dist_{DIR}(H(Q), H(x_i))$ for $1 \leq i \leq b$ in ascending order, the candidate elements whose $dist_{DIR}(H(Q), H(x_i))$ is less than the $(k+1)^{th}$ are selected as the candidates \mathbf{C} .

5.4.3 Refining Stage

As previously shown in the experimental results in Section 5.1, the retrieved result of the global-based shape descriptor such as the DIR descriptor is not as good as using the Hausdorff distance. In this framework, we apply the modified-Hausdorff distance for ranking the retrieved result. We avoid a linear scan of the database by computing the Hausdorff distance between each candidate element in \mathbf{C} and the query Q' . The outline of the refining stage is illustrated in Figure 5.6.

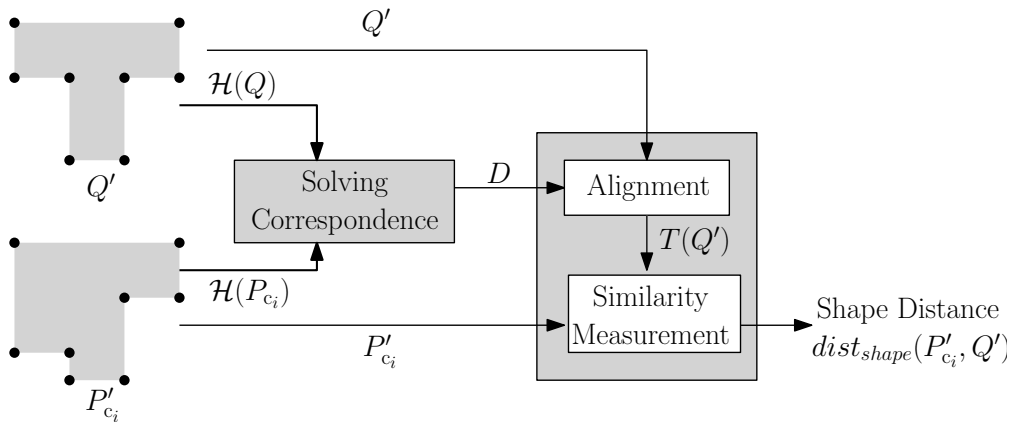


Figure 5.6: Outline of the refining stage.

Solving correspondence

Given a set of feature points Q' of the query and a candidate set \mathbf{C} , we compute a cost matrix C in order to obtain a set of matching points A which is solved using Hungarian method. The cost C_{uv} of a point $p'_u \in P'_{c_i}$ of a candidate c_i for $1 \leq u \leq a$ and a point $q'_v \in Q'$ for $1 \leq v \leq a$ of the query is the L_1 norm between the spatial data $H(p'_u) \in \mathcal{H}(P'_{c_i})$ and $H(q'_v) \in \mathcal{H}(Q')$ that is $dist_{DIR}(\mathcal{H}(Q), \mathcal{H}(c_i))$. For detail of computation, see equation 5.2.

Given the cost matrix C , the Hungarian method returns a set A which is a list of pairs of matching points between the feature points Q' and P'_{c_i} . For each matching pair $(p'_{\pi(j)}, q'_j) \in A$, if the cost $C_{\pi(j)j}$ is minimum all C_{uj} for $1 \leq u \leq a$, then $(p'_{\pi(j)}, q'_j)$ is selected as a corresponding point and stored in a set D .

Alignment and similarity measurement

Each pair of the corresponding point in D is used for estimating the Euclidean motion that best aligns Q' and $c_i \in \mathbf{C}$ for $1 \leq i \leq k$. Let $T = [p'_i, p'_j, q'_i, q'_j]$ be a transformation obtained from the alignment of an edge $p'_i p'_j$ with an edge $q'_i q'_j$ which $(p'_i, q'_i) \in D$ and $(p'_j, q'_j) \in D$.

The transformation T is applied to Q' before computing the Hausdorff distance between the candidate element P'_{c_i} and $T(Q')$. The shape distance denoted by $dist_{shape}(P'_{c_i}, Q')$ is the modified-Hausdorff distance which is minimum among all transformation T .

$$dist_{shape}(P'_{c_i}, Q') = \min_{1 \leq i \leq |D|} \min_{1 \leq j \leq |D|} H_{mod}(P'_{c_i}, T(Q'))$$

where $T = [p'_i, p'_j, q'_i, q'_j]$ and $H_{mod}(P'_{c_i}, T(Q')) = \max(d(P'_{c_i}, T(Q')), d(T(Q'), P'_{c_i}))$. For the detail of $H(P'_{c_i}, T(Q'))$ calculation, see Equation 2.1.

The shape distance $dist_{shape}(P'_{c_i}, Q')$ is computed for all $1 \leq i \leq |\mathbf{C}|$ and sorted in ascending order. The final result is return in order of the sorted shape distance.

5.4.4 Analysis and Time Complexity

Preprocessing

Given a set of boundary points P where $|P| = m$ and a set of feature a points P' , the two shape descriptors are computed. For each shape P in the database, the DIR descriptor can be computed in $O(m^2)$ time and the spatial data which is the LDIR descriptor can be computed in $O(ma)$ time.

Screening

Given a shape signature of a query object, the elements set X is retrieved via the LSH. The cardinality of the set X of element retrieved via the LSH is $\mathcal{Q} = \delta O$, where O is the total number of objects in the data set. The parameter δ is called a compression ratio.

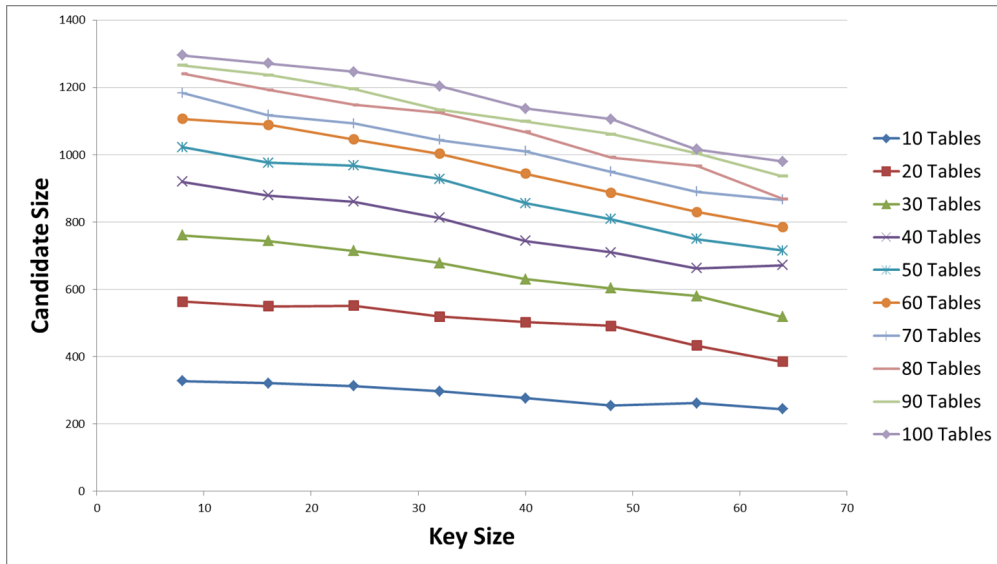


Figure 5.7: The comparison of the accuracy of the MPEG7 dataset retrieved from LSH with table size $L = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and the key size $K = \{8, 16, 24, 32, 40, 48, 56, 64\}$.

In the proposed framework, δ depends on parameter of the Locality Sensitive Hash-

ing (LSH) that is the number of tables L and the key size K . The comparison of the candidate size using table size $L = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and the key size $K = \{8, 16, 24, 32, 40, 48, 56, 64\}$ is shown in Figure 5.7. Fixing a table size, the candidate size is decreased as the key size increased. On the other hands, the bigger the number of table L causes the larger the candidate size. The comparison of the accuracy of the retrieved result is shown in Figure 5.8. The accuracy is measure using precision which is a ratio between the accurate element retrieved and the total number of retrieved elements in Figure 5.7. Using $L = 80$ and $K = 64$ the precision is 73.8% which is the highest.

The L_1 norm of the DIR descriptor is computed for each element and the query and the k^{th} closest element are selected as a set of candidate \mathbf{C} . The time complexity for selecting \mathbf{C} is $O(rcQ)$ which r and c is the DIR descriptor size and Q is the cardinality of the element set X .

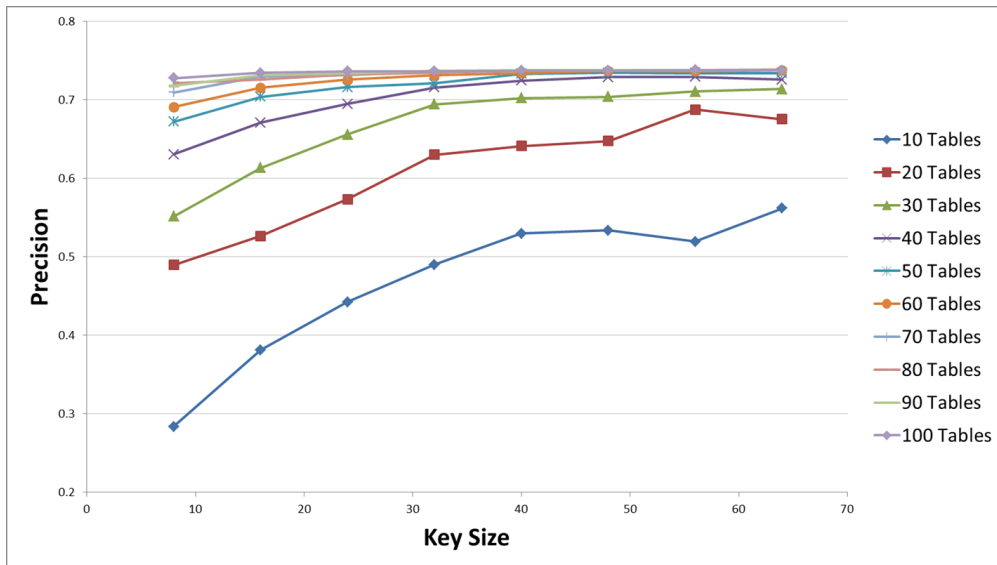


Figure 5.8: The comparison of the precision of the MPEG7 dataset retrieved from LSH with table size $L = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ and the key size $K = \{8, 16, 24, 32, 40, 48, 56, 64\}$.

Refining

The time complexity for computing the corresponding point is $O(a^3)$ where a is the size of the feature points. Using the corresponding points, the similarity of two planar point sets P' and Q' using the Hausdorff distance can be computed in $O(|D|^2a^4)$ time where D is the set of corresponding pairs of points.

The total time complexity for computing $dist_{shape}$ of the query against each candidate in the candidate set \mathbf{C} is $O(|D|^2a^4|\mathbf{C}|)$. For simplifying descriptions, we abbreviate the time complexity by \mathcal{T} .

Theorem 5.4.5. *The time complexity for one query using the three-stage framework is $O(rc\mathcal{Q} + \mathcal{T})$, assuming that the shape signature is described in the statistical information of a constant size. The preprocessing time is $O(O(m^2 + am))$ time, where m is the number of boundary points, a is the number of feature points uniformly taken from each object in the database, and O is the database size.*

5.4.5 Experimental Result

The performance of proposed framework is evaluated using the MPEG7 CE-Shape-1 [61] and Kimia's 99 [80] datasets. For each query shape, we compute the corresponding points to every other shape in the data set.

Setting of the experiment

The boundary pixels of each object are extracted and then 800 points uniformly selected. 80 features points are selected from the 800 points. The size of the DIR descriptor of each shape boundary, we choose $c = 64$ and $r = 32$. For the size of the LDIR descriptor of each feature point, we choose $c = 8$ and $r = 6$.

The Locality Sensitive Hashing (LSH) is applied in the screening process. According to the experimental result in Section 4.3.6, the table size L is set to 80 and the key size K

is 64. The bucket size is \sqrt{O} , where O is the dataset size. For each query, the screening stage returns $4 * o$ candidates where o is the number of object is each class. For the MPEG7 dataset, the candidate size is $4 * 20$ that is 80. The Kimia data set can be defined analogously.

For each pair of shapes, 5 corresponding pair of points are selected and the shape distance is computed using the modified-Hausdorff distance. For each query, we count the 10 closest shapes that are in the same class as the query. The self-match is not included in the count.

The experiments are conducted on a desktop machine with the Intel Core i7 3.4 GHz CPU and 16 GB of memory.

Result

Since the LSH is applied for retrieving a set of candidates objects, we use the MPEG dataset which consists of 1,400 objects in this experiment. The BEP score is used to measure the efficiency of the proposed framework.

Using the same setting as in Section 4.3.6, the BEP score of the candidate objects retrieved from the LSH is 73.80%. By applying the proposed LDIR descriptor and the Hungarian method for solving correspondence and the mean of the optimal matching cost is used as the similarity measure, such method achieves 74.25% BEP score. However, when applying the modified-Hausdorff distance in the last stage, the BEP score is decreased to 69.19%.

The comparison of the five closest objects retrieved from the screening process and the refining process are shown in Table 5.3. The comparison of the result when the refining process could not recognize the objects from the same class as the input query are shown. From the first example, we can notice that the objects recognized by the refining process have similar shapes. However, they are from the different class. For the third example, though the shapes of the objects being recognized by the refining process have very similar shape, they are from the different classes.

Table 5.4 shows the examples of the result when the refining process is able to rerank

the final objects correctly. From the last example clearly shows that the DIR descriptor is not able to uniquely describe the shapes, however, the refining process is able to remove all irrelevant objects from the final result. Similarly, the third query shows the example of a complicated shape of object.

5.5 Discussion and Conclusion

In this chapter, we target the time complexity problem when applying shape as a similarity measurement. Comparing the similarity of the two shapes, we present an idea which combines the shape descriptor and the spatial data for reducing the time when computing the Hausdorff distance. A method for solving a set of corresponding points whose spatial data on the two shapes are similar is proposed. The correspondence between the two shapes boundary are used for estimating the transformation which maps one shape to the most similar part of the other. In this way, unneeded points can be ignored for finding the best transformation.

A framework for the shape-based image retrieval using the modified-Hausdorff distance as a similarity measurement is proposed. The idea is to avoid computing the modified-Hausdorff distance to every shape in the database. The proposed framework composes of three stages: (1) Preprocessing; (2) Screening; and (3) Refining. A search structure of the global-based shape descriptor is precomputed. Given a query, a set of candidates is retrieved and selected via the search structure. In the screening stage, the vector features are used for deciding the similarity of the shape. Therefore, it can be computed quickly and only possible shapes are selected as candidates for further processing. The modified-Hausdorff distance is used for refining the retrieved result. In the refining stage, only candidate objects are being taken into consideration, therefore, the total retrieval time is independent to the database size.

In experiment using the Kimia dataset, we compare the result using the modified-

Hausdorff distance $d_H(P, Q)$ and our proposed method which is $dist_{shape}(P, Q)$. In theory, the modified-Hausdorff distance gives an optimal value for matching the two shapes that is $d_H(P, Q) \leq dist_{shape}(P, Q)$. However, due to the computational time when computing the optimal value, we apply the approximate scheme. We notice from the experimental result that computing the optimal modified-Hausdorff distance takes time and also the approximation of the optimal transformation is not easy to compute. Comparing the result using the correspondence for estimating the alignment transformation and the approximate modified-Hausdorff distance, our method gives better result.

When applying a larger size of the dataset (MPEG7) we found that that BEP of the proposed framework is less than the naive method which is contradict to the result using the linear scan method. This is because the objects in the MPEG7 dataset are classified by human which the similar shape of objects. For illustration, see the comparison of the result in Table 5.4.

Table 5.3: Example of the five closest objects which are correctly retrieved from the screening process.















































































































Input	Method	1	2	3	4	5
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					

Table 5.4: Example of the comparison on the five closest objects that the retrieved is correctly reranked using the refining process.

Input	Method	1	2	3	4	5
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					
	Screening					
	Refining					

Chapter 6

Conclusion

The goal of this thesis is to improve the accuracy of the retrieved result efficiently in a Content-Based Image Retrieval (CBIR) using shape as a main feature. Given a query image, the retrieval system must be able to retrieve the relevant image promptly. The process of the image retrieval system includes browsing and identifying the similarity of images in the database using the shape feature. In order to browse for similar images from a large database promptly, shape features of images in the database are extracted and stored in an efficient search structure. Similarly, to retrieve relevant results, shape similarity among images is identified using an accurate approach such as pattern matching.

Shape of objects in each image in the database is located and extracted using image segmentation, however, the present approach requires constraints such as precise location of object, a set of sample pixels in an interested area, etc. which is not suitable for the huge size database. Moreover, the time complexity for identifying the similarity measurement using the pattern matching such as Hausdorff distance is linear in the database size.

We present three main results in order to cope with the retrieval time complexity while retaining the accuracy of the pattern matching.

Room-edge Maximum Weight Region: We present semi-automatic image segmen-

tation algorithm which applies an optimization algorithm. A framework for transforming the segmentation algorithm to the optimization algorithm is presented. Comparing to the similar method, i.e., graph-cut, our proposed method gives the optimal segmentation. Moreover, the shape of the object can be controlled using the room-edge region. To improve the quality of the segmentation region, a pixel grid is decomposed into small subrectangles and the optimal region is separately solved. Given a positive integer, the two decomposition algorithm automatically finds optimal location of the baselines.

Distance Interior Ratio (DIR) shape descriptor: After extracting shape of an object in an image, shape feature is represented using distance distribution. The advantage of the distance-based descriptor is that it can be compressed into a feature vector. In this way, the similarity between the two shape descriptor is measured in a vector space. The shape descriptor can also be stored in the search structure efficiently. We present a novel distance-based shape descriptor in order to cope with the homometric pair problem when representing the shape using the pairwise distance distribution. The experimental result shows that the accuracy of the proposed descriptor is higher than the state-of-art distance-based descriptors.

We present a framework for improving the time complexity for computing the Hausdorff distance of each pair of shapes. Instead of computing the shape distance using the transformation estimation computed from every possible alignment of the line segments, only the alignments which are very likely mapped the two shapes to the most similar part of each other are selected. Therefore, the time complexity for computing the shape distance is improved. We extend the idea of the proposed shape descriptor for solving correspondence points on the two shape boundary. The experimental result shows that the proposed method achieved higher accuracy than the approximate modified-Hausdorff distance.

Framework for shape-based image retrieval: A problem of applying the Hausdorff distance directly as a similarity measurement is retrieval time that is linear in the database size. Since the search space using such metric is not easy to organize, a query needs comparing against each object in the database. To avoid a linear computation of each shape in the

database, three-stage image retrieval framework is proposed. Every image in the database is preprocessed using the previous two proposed methods. To keep away from the linear time complexity in the database size, only images which are likely to be similar to the query are selected for computing the shape distance. A set of candidate objects is retrieved via the precomputed search structure using the DIR shape descriptor. Therefore, the time complexity depends on the cardinality of the candidate set which is much smaller than the database size.

6.1 Future Work

One limitation of the proposed image segmentation algorithm is the number of baselines which is given by a user. To accomplish fully-automatic image segmentation, it is necessary for the algorithm to be able to decide the optimum number of baselines for decomposing the image such that the segmentation result is optimal. The difficulty of the image segmentation is to measure the quality of the result. Interclass variance of the intensity level between the two segmented regions is applied. Unfortunately, such method does not give good result.

The sampling method of points on the shape contour for computing the shape descriptor affects the accuracy of the similarity measurement using the Hausdorff distance. We found from our preliminary experimental result that the uniform sampling achieves higher result than the non-uniform method. In the future, it is interesting to analyze which sampling method is best suitable for the DIR descriptor. There are many sampling method such as logarithmic sampling, smoothness sampling, etc.

As shown in the experimental result, the Hausdorff distance is able to recognize the similar shape of objects. Due to the limitation of the classification method of the dataset, the similar shapes are classified differently. In the future, we would like to apply different dataset which has larger size of the shapes. So that the efficiency of applying the Locality Sensitive Hashing and the proposed framework can be measured nicely.

Bibliography

- [1] S. Abbasi, F. Mokhtarian, and J. Kittler. Curvature scale space image in shape similarity retrieval. *Multimedia Systems*, 7(6):467–476, 1999.
- [2] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.
- [3] Airlines.Net. Airlines.net homepage, 2005.
- [4] T. Akutsu, H. Tamaki, and T. Tokuyama. Distribution of distances and triangles in a point set and algorithms for computing the largest common point sets. In *Proceedings of the thirteenth annual symposium on Computational geometry*, pages 314–323, 1997.
- [5] N. Alajlan, M.S. Kamel, and G.H. Freeman. Geometry-based image retrieval in binary image databases. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):1003–1013, 2008.
- [6] H. Alt and G. Aichholzer, O. and Rote. Matching shapes with a reference point, 1994.
- [7] H. Alt, O. Aichholzer, and G. Rote. Matching shapes with a reference point. In *Proceedings of the tenth annual symposium on Computational geometry*, pages 85–92, 1994.
- [8] H. Alt and L. Scharf. Shape matching by random sampling. In *WALCOM: Algorithms and Computation*, volume 5431, pages 381–393. 2009.

-
- [9] E. Aptoula and S. Lefevre. Morphological description of color images for content-based image retrieval. *Image Processing, IEEE Transactions on*, 18(11):2505–2517, 2009.
- [10] T. Asano, D.Z. Chen, N. Katoh, and T. Tokuyama. Efficient algorithms for optimization-based image segmentation. *International Journal of Computational Geometry and Applications*, 11(2):145–166, 2001.
- [11] M. M. Babu, S. K. Mohan, and Wing F. L. Shape measures for content based image retrieval: A comparison. *Information Processing and Management*, 33(3):319–337, 1997.
- [12] Ronen Basri, Luiz Costa, Davi Geiger, and David Jacobs. Determining the similarity of deformable shapes. *Vision Research*, 38(1516):2365 – 2385, 1998.
- [13] S. Belongie and J. Malik. Matching with shape contexts. In *Content-based Access of Image and Video Libraries*, pages 20–26, 2000.
- [14] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.
- [15] M. Boutin and G. Kemper. On reconstructing n-point configurations from the distribution of distances or areas. *Advances in Applied Mathematics*, 32(4):709 – 735, 2004.
- [16] Y. Boykov and G. Funka-Lea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.
- [17] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

-
- [18] Y. Boykov and O. Veksler. Graph Cuts in Vision and Graphics: Theories and Applications. In *Handbook of Mathematical Models in Computer Vision*. Springer US, 2006.
- [19] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.
- [20] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.
- [21] Chen C. C., H. D. Wactlar, J. Z. Wang, and K. Kiernan. Digital imagery for significant cultural and historical materials. *International Journal on Digital Libraries*, 5:275–286, 2005.
- [22] J Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.
- [23] S. Carlsson. Order structure, correspondence, and shape based categories. In *Shape, Contour and Grouping in Computer Vision*, volume 1681, pages 58–71. 1999.
- [24] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3241–3248, 2010.
- [25] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Visual Information and Information Systems*, volume 1614, pages 509–517. 1999.
- [26] C.C. Chang, S.M. Hwang, and D.J. Buehrer. A shape recognition scheme based on relative distances of feature points from the centroid. *Pattern Recognition*, 24(11):1053 – 1063, 1991.

-
- [27] X. Chen, S. An, W. Liu, and W. Li. Object detection by admissible region search. In *Advances in Artificial Intelligence*, volume 7106, pages 521–530. 2011.
- [28] H.D. Cheng, X.H. Jiang, Y. Sun, and Jingli Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12):2259 – 2281, 2001.
- [29] L.P. Chew, M.T. Goodrich, D.P. , K. Kedem, J.M. Kleinberg, and D. Karvets. Geometric pattern matching under euclidean motion, 1997.
- [30] Cheng-Chieh Chiang, Yi-Ping Hung, Hsuan Yang, and Greg C. Lee. Region-based image retrieval using color-size features of watershed regions. *Journal of Visual Communication and Image Representation*, 20(3):167 – 177, 2009.
- [31] J. Chun, N. Kaothanthong, and T. Tokuyama. Classified-distance based shape descriptor for application to image retrieval. In *The 15th International Conference on Computer Analysis of Images and Patterns*, 2013.
- [32] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):1–60, 2008.
- [33] M. de Berg, M. van Kreveld, M.and Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000.
- [34] T. Deselaers, D. Keysers, and H. Ney. Features for image retrieval: an experimental comparison. *Information Retrieval*, 11(2):77–107, 2008.
- [35] M.P. Dubuisson and A.K. Jain. A modified hausdorff distance for object matching. In *Pattern Recognition*, volume 1, pages 566–568 vol.1, 1994.
- [36] Alon Efrat, Alon Itai, and Matthew J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31:128, 2001.

-
- [37] J. Fan, G. Zeng, M. Body, and M.-S. Hacid. Seeded region growing: an extensive and comparative study. *Pattern Recognition Letters*, 26(8):1139 – 1156, 2005.
- [38] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. *International Journal of Computer Vision*, 87(3):284–303, 2010.
- [39] R.A. Finkel and J.L. Bentley. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974.
- [40] H. Freeman. On the encoding of arbitrary geometric configurations. *Electronic Computers, IRE Transactions on*, EC-10(2):260–268, 1961.
- [41] A. Frome and J. Malik. Object recognition using locality-sensitive hashing of shape contexts, 2006.
- [42] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *The 25th International Conference on Very Large Data Bases*, pages 518–529, 1999.
- [43] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 518–529, 1999.
- [44] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Prentice Hall, 3 edition, 2007.
- [45] K. Grauman and T. Darrell. Fast contour matching using approximate earth mover’s distance. In *Computer Vision and Pattern Recognition*, volume 1, pages 220–227, 2004.
- [46] K. Grauman and T. Darrell. Fast contour matching using approximate earth mover’s distance. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages 220–227, 2004.

-
- [47] C. Grigorescu and N. Petkov. Distance sets for shape filters and shape recognition. *Image Processing*, 12:1274–1286, 2003.
- [48] Robert M. Haralick and Linda G. Shapiro. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29(1):100 – 132, 1985.
- [49] E. Hjelm and B. K. Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236 – 274, 2001.
- [50] D. P. Huttenlocher, K. Kedem, and J. M. Kleinberg. On dynamic voronoi diagrams and the minimum hausdorff distance for point sets under euclidean motion in the plane. In *Proceedings of the eighth annual symposium on Computational geometry*, pages 110–119, 1992.
- [51] D.P. Huttenlocher and W.J. Rucklidge. A multi-resolution technique for comparing images using the hausdorff distance. In *Computer Vision and Pattern Recognition*, pages 705–706, 1993.
- [52] P. Indyk and N. Thaper. Fast Image Retrieval via Embeddings. In *3rd International Workshop on Statistical and Computational Theories of Vision*, 2003.
- [53] C.Y. Ip, D. Lapadat, L. Sieger, and W.C. Regli. Using shape distributions to compare solid models. In *The seventh ACM symposium on Solid modeling and applications*, pages 273–280, 2002.
- [54] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal Of Computer Vision*, 1(4):321–331, 1988.
- [55] A. Khokher. Image Retrieval: A State of the Art Approach for CBIR. *International Journal of Engineering Science and Technology*, 3(8):6698–6704, 2011.

- [56] V. Kolmogorov and R. Zabini. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004.
- [57] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.
- [58] Li-Jia L., R. Socher, and L. Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Computer Vision and Pattern Recognition*, pages 2036–2043, 2009.
- [59] Lucchese L and Mitra S. K. Color image segmentation: a state-of-the-art survey. In *Proceedings of the of the Indian National Science Academy*, volume 67, pages 207–211, 2001.
- [60] Y. Lamdan and H.J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Computer Vision., Second International Conference on*, pages 238–249, 1988.
- [61] L.J. Latecki, R. Lakamper, and T. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Computer Vision and Pattern Recognition*, volume 1, pages 424–429 vol.1, 2000.
- [62] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transaction on Multimedia Computation and Communication Application*, 2(1):1–19, 2006.
- [63] H. Ling and D.W. Jacobs. Shape classification using the inner-distance. *Pattern Analysis and Machine Intelligence*, 29(2):286–299, 2007.
- [64] H. Liu, W. Liu, and L. J. Latecki. In *Computer Vision and Pattern Recognition*, pages 97–104. IEEE, 2010.

- [65] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [66] R. Mehwish, I. Muhammad, S. Muhammad, and Mudassar R. Content based image retrieval: Survey. *World Applied Sciences Journal*, 19(3):404–412, 2012.
- [67] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.
- [68] Y. Mingqiang, K.K. Idiyo, and R. Joseph. A Survey of Shape Feature Extraction Techniques. *Pattern Recognition*, pages 43–90, 2008.
- [69] H. Mller, N. Michoux, D. Bandon, and A. Geissbuhler. A review of content-based image retrieval systems in medical applications – clinical benefits and future directions, 2004.
- [70] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM Transactions on Graphics*, 21:807–832, 2002.
- [71] Inner Distance Shape Context package.
http://www.dabi.temple.edu/~hbling/code_data.htm.
- [72] LSH package.
<http://ttic.uchicago.edu/~gregory/download.html>.
- [73] Shape Context package.
http://www.eecs.berkeley.edu/research/projects/cs/vision/shape/sc_digits.html.
- [74] T. Pavlidis. *Algorithms for Graphics and Image Processing*. Computer Science Press, 1982.
- [75] B. Peng and O. Veksler. Parameter selection for graph cut based image segmentation. In *British Machine Vision Conference*, pages 1–10.

- [76] P. Petrou, M. and Bosdogianni. *Image Processing - The Fundamentals*. Wiley, second edition, 1999.
- [77] S. A. Raut, M. Raghuwanshi, R. Dharaskar, and A. Raut. Image segmentation - a state-of-art survey for prediction. In *Proceedings of the 2009 International Conference on Advanced Computer Control*, pages 420–424, 2009.
- [78] T. V. Robertson. Ex traction and classification of objects in multispectral images. *Machine Processing of Remotely Sensed Data*, pages 27–34, 1973.
- [79] P.L. Rosin and C.L. Mumford. A symmetric convexity measure. *Computer Vision Image Understanding*, pages 101–111, 2006.
- [80] T.B. Sebastian, P.N. Klein, and B.B. Kimia. Recognition of shapes by editing their shock graphs. *Pattern Analysis and Machine Intelligence*, 26(5):550–571, 2004.
- [81] N. Senthilkumaran and R. Rajesh. Edge detection techniques for image segmentation - a survey of soft computing approaches. *International Journal Of Recent Trends in Engineering*, 1(2):250–254, 2009.
- [82] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing)*. The MIT Press, 2006.
- [83] Eitan Sharon, Meirav Galun, Dahlia Sharon, Ronen Basri, and Achi Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):810–813, 2006.
- [84] X. Shu and X.J. Wu. A novel contour descriptor for 2d shape matching and its application to image retrieval. *Image Vision Comput.*, 29(4):286–294, 2011.

-
- [85] S.S. Skiena, W.D. Smith, and P. Lemke. Reconstructing sets from interpoint distances (extended abstract). In *The sixth annual symposium on Computational geometry*, pages 332–339, 1990.
- [86] B. J. Super. Retrieval from shape databases using chance probability functions and fixed correspondence 1. *Pattern Recognition and Artificial Intelligence*, 20(8):1117–1138, 2006.
- [87] Image Segmentation using Graph Cut.
<https://code.google.com/p/graphcuts/>.
- [88] Peter J. Van Otterloo. *A contour-oriented approach to shape analysis*. Prentice Hall International (UK) Ltd., 1991.
- [89] R. C. Veltkamp and L. J. Latecki. Properties and performance of shape similarity measures. In *Data Science and Classification*, pages 47–56. Springer Berlin Heidelberg, 2006.
- [90] R.C. Veltkamp. Shape matching: similarity measures and algorithms. In *Shape Modeling and Applications*, pages 188–197, 2001.
- [91] Remco C. Veltkamp and Michiel Hagedoorn. Principles of visual information retrieval. chapter State of the art in shape matching, pages 87–119. 2001.
- [92] Remco C. Veltkamp and Mirela Tanase. Content-based image retrieval systems: A survey. Technical report, Department of Computing Science, Utrecht University,, 2000.
- [93] S. Yang. Symbol recognition via statistical integration of pixel-level constraint histograms: A new descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:278–281, 2005.

-
- [94] D. Zhang and G. Lu. A comparative study of fourier descriptors for shape representation and retrieval. In *The 5th Asian Conference on Computer Vision*, pages 646–651, 2002.
- [95] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, 2004.
- [96] H. Zhang, J. E. Fritts, and S. A. Goldman. Image segmentation evaluation: A survey of unsupervised methods. *Computer Vision and Image Understanding*, 110(2):260 – 280, 2008.
- [97] J. Zhang and L. Wenyin. A pixel-level statistical structural descriptor for shape measure and recognition. In *The 10th International Conference on Document Analysis and Recognition*, pages 386–390, 2009.
- [98] J. Zunic and P.L. Rosin. A convexity measurement for polygons. *Pattern Analysis and Machine Intelligence*, 26:173–182, 2002.

List of Publications

Journal

- [1] J. Chun, T. Horiyama, T. Ito, N. Kaothanthong, H. Ono, Y. Otachi, T. Tokuyama, R. Uehara, and T. Uno. Base-object location problems for base-monotone regions. *Theoretical Computer Science*, 2013.
- [2] J. Chun, N. Kaothanthong, R. Kasai, M. Korman, M. Nöllenburg, and T. Tokuyama. Algorithms for computing the maximum weight region decomposable into elementary shapes. *Computer Vision and Image Understanding*, 116(7):803–814, 2012.
- [3] N. Kaothanthong, J. Chun, and T. Tokuyama. Classified-distance based shape descriptor for image retrieval. *invited, submitted*.
- [4] C.-T. Nguyen, N. Kaothanthong, T. Tokuyama, and X.-H. Phan. A feature-word-topic model for image annotation and retrieval. *ACM Transactions on the Web*, 7(3):12:1–12:24, 2013.

Refereed International Conference

- [1] L. Chiu, N. Kaothanthong, T. Theeramunkong, and C. Nattee. A corpus-based approach for thai romanization. In *Proceedings of the Seventh International Symposium*

- on Natural Language Processing*, 2007.
- [2] J. Chun, T. Horiyama, T. Ito, N. Kaothanthong, H. Ono, Y. Otachi, T. Tokuyama, R. Uehara, and T. Uno. Base location problems for base-monotone regions. In *Proceedings of the 7th International Workshop on Algorithm and Computation*, pages 53–64, 2013.
- [3] J. Chun, N. Kaothanthong, Y. Ota, and T. Tokuyama. Distance-based shape invariants for image retrieval. In *The 4th Annual Meeting of the Asian Association for Algorithms and Computation*, 2011.
- [4] J. Chun, N. Kaothanthong, Y. Ota, and T. Tokuyama. Image retrieval system using distance-based shape recognition. In *Proceedings of the 14th Korea-Japan Joint Workshop on Algorithms and Computation*, 2011.
- [5] J. Chun, N. Kaothanthong, H. Takahashi, and T. Tokuyama. How to cut a complicated figure by using scissors? computing the maximum weight region consisting of base monotone regions. In *The 5th Annual Meeting of the Asian Association for Algorithms and Computation*, 2012.
- [6] J. Chun, N. Kaothanthong, H. Takahashi, and T. Tokuyama. Optimal grid decomposition for maximum weight region computation with application to image segmentation. In *Computational Geometry: Young Researchers Forum*, 2012.
- [7] J. Chun, N. Kaothanthong, and T. Tokuyama. Classified-distance based shape descriptor for application to image retrieval. In *Computer Analysis of Images and Patterns*, volume 8048, pages 1–8. 2013.
- [8] J. Chun, N. Kaothanthong, and T. Tokuyama. Correspondence finder using classified distance distribution for efficient shae retrieval. In *Proceedings of the 16th Korea-Japan Joint Workshop on Algorithms and Computation*, 2013.

-
- [9] J. Chun, N. Kaothanthong, and T. Tokuyama. Shape description using classified distances. In *Proceedings of the 6th Annual Meeting of the Asian Association for Algorithms and Computation*, 2013.
- [10] C.-T. Nguyen, N. Kaothanthong, X. H. Phan, and T. Tokuyama. A feature-word-topic model for image annotation. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1481–1484, 2010.

Workshops and Unrefereed Conference

- [1] J. Chun, T. Horiyama, T. Ito, N. Kaothanthong, H. Ono, Y. Otachi, T. Tokuyama, R. Uehara, and T. Uno. Algorithms for computing optimal image segmentation using quadtree decomposition. In *Thailand-Japan Joint Conference on Computational Geometry and Graphs*, 2012.
- [2] J. Chun, N. Kaothanthong, Y. Ota, and T. Tokuyama. Image retrieval using shape recognition. In *The 10th Forum on Information Technology (FIT)*, 2011.
- [3] J. Chun, N. Kaothanthong, and T. Tokuyama. Image recognition and retrieval by using distance information. In *LA Symposium*, 2011.
- [4] J. Chun, N. Kaothanthong, and T. Tokuyama. Image segmentation using maximum weight region. In *Technical Committee on Theoretical Foundations of Computing (COMP)*, 2011.
- [5] J.Chun, N. Kaothanthong, H. Takahashi, and T.Tokuyama. An image segmentation using maximum weight region with shape constraint. In *The First ETH-JAPAN Workshop on Science and Computing*, 2012.
- [6] N. Kaothanthong. Study on a shape-based image retrieval using distance information. In *The 4th Thailand-Japan International Academic Conference*, 2011.

- [7] C.-T. Nguyen, N. Kaothanthong, X. H. Phan, and T. Tokuyama. A feature-word-topic model for image annotation. In *The 3rd Student Organizing International Mini-Conference on Information Electronics Systems*, 2010.