

國立交通大學

資訊科學與工程研究所

碩 士 論 文

地圖結合與互動全景街道系統

Map-based Interactive Street Panorama System



研 究 生：陳冠璋

指 導 教 授：張明峰 教授

中 華 民 國 九 十 七 年 七 月

地圖結合與互動全景街道系統
Map-based Interactive Street Panorama System

研究生：陳冠璋

Student：Kuan-Chang Chen

指導教授：張明峰

Advisor：Ming-Feng Chang

國立交通大學

資訊科學與工程研究所

碩士論文



A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年七月

地圖結合與互動全景街道系統

學生：陳冠璋

指導教授：張明峰教授

國立交通大學資訊工程學系 (研究所) 碩士班

摘要

近年來有非常大的興趣在網路應用及網路服務。網路服務所提供的網路應用的需求增加。網路服務提供者可以專心在內容的提供，其他需要的功能可以由其他專業的網路服務提供者提供。在許多的網路應用中，電子商務模式的服務有著龐大的商機。當在網路中有一個系統可以提服務和導遊，該系統可以提供一個逛街的能。而在本研究裡面，主要的問題是分成二個，一個是找最近點的問題，另一個是路徑規劃的問題。這二問題都是重要的議題在電腦圖學中，也是許多人所研究的。在本研究中，利用收集到的資料中觀察了資料的分佈特性去發展找最近點的演算法。這樣的資料分佈是本演算法中最重要的關鍵。資料分佈主要是以點的分佈可以由線段去表示。進而去討論如果找到最近的點所在的線段中，這樣的線段我們稱為一個 Group。利用這樣重新組織的方法，我們基於這樣的結果來發展我們的找最近點的演算法。而在路徑規劃中，我們也定義了自己的 A-star 函式進而用在自己的系統裡面。

Map-based Interactive Street Panorama System

Student: Kuan-Chang Chen

Advisor: Prof. Ming-Feng Chang

Department of Computer Science and Information Engineering

National Chiao Tung University

Abstract

Recently, web applications and web services have become prevalent on the Internet. New web applications and services can be fast developed and provisioned if the services are built on the top of existing services. In this thesis, we built a map-based interactive street panorama system that utilizes Google map service. First, sequences of pictures were taken along the streets; the pictures contain GPS information, the longitude and latitude. Pictures of a street are grouped together by a grouping algorithm presented in this thesis. The groups are stored as a kd-tree. To locate a target street view, we developed algorithms to locate on the map the closest picture of a user-chosen point and the shortest path between two user-chosen points. The closet point algorithm locates the top 3 closest groups in the kd-tree and use projection to locate the closest point. The shortest path algorithm is an A-star algorithm. Grouping the pictures of a street together increases the efficiency of the closest point algorithm and the shortest path algorithm.

誌謝

首先感謝我最敬重的導師 張明峰教授。在就讀碩士班的這兩年，在導師的督促及訓練獨立思考研究，讓我無論在作研究或者做人處事方面成長很多，謝謝老師的教誨。

特別要感謝善隆學長。在撰寫論文期間給予的意見以及在研究方法要注意的地方。而在這兩年學習過程，感謝在實驗室的好伙伴忠育、坤揚和君飛的支持和激勵下，提供我在研究過程中源源不斷奮鬥下去的動力；同時再一次感謝導師提供良好的研究環境與實驗儀器，使我可以在研究的過程中，沒有受到任何的阻礙與限制。

最後將此論文獻給我最親愛的家人。感謝您們在我求學期間全心全意的支持，讓我得以順利的完成學業。



List of Tables

Table 2- 1 An compare of data structures	9
Table 2-2 An compare of algorithms	12
Table 4- 1 The time complexity of nearest neighbor of kd-tree.....	28
Table 4- 2 The time complexity of kd-tree	29
Table 4- 3 The compare with kd-tree	31
Table 4- 4 The compare with kd-tree, m is the number of group	31
Table 4- 5 Compare algorithm of path finding	33
Table 4- 6 The relationship of path find algorithms.....	33

List of Figures

Figure 2-1 An example of graph.....	11
Figure 2-2 The shape of spanning tree.....	12
Figure 3-1 System Overview	13
Figure 3- 2 A position of the closest point	17
Figure 3- 3 A position of the closest point	17
Figure 3- 4 The flow of the algorithm of point query	18
Figure 3-5 An example of a 2d-tree	19
Figure 3- 6 An query of kd-tree	20
Figure 3-7 An position of the closest point.....	21
Figure 3-8 A position of the closest point in a group.....	22
Figure 3-9 The shortest path in a graph.	23
Figure 4-1 Two distribution of a kd-tree.....	28
Figure 4-2 A figure of the groups.....	29
Figure 4-4 An example of greedy BFS	34

Table of Contents

摘要.....	i
Abstract.....	ii
誌謝.....	iii
List of Tables.....	iv
List of Figures.....	iv
Table of Contents.....	v
Chapter 1 Introduction.....	1
1.1 Motivation.....	1
1.2 Research Questions.....	3
1.3 Related work.....	4
1.4 Overview.....	6
Chapter 2 Related Work.....	7
2.1 The closest point problem.....	7
2.2 The algorithm of the closet point problem.....	8
2.3 Path finding.....	9
2.4 The Algorithm of Path finding.....	10
Chapter 3 System Design and Implementation.....	13
3.1 System Overview.....	13
3.2 Point Query.....	14
3.3 Path Finding.....	23
3.4 Summary.....	26
Chapter 4 Analysis and Evaluation.....	27
4.1 Closet Point Finding Algorithm.....	27
4.2 Path Finding Algorithm.....	32

Chapter 5 Conclusions and Future work.....36
References.....37



Chapter 1 Introduction

1.1 Motivation

Recently, there have been wide interests in web applications and web services. The web service provider combines the other service. The web service provider concentrates on the content provided. The content is meaning the main service of the web service provider. For example, the main services of books shop are book selling and book information query. The web services provider whose web sites provide web services. They provide the services which are the books information, yellow book service, and the others on Internet. They also have a web site which is accessed by users who need information. For example, Google is a web service provider. Google provides information of the world on Internet.

The use of web technology on web service has increasingly been the object of study in recent year. The requirements of the interface of Open API have gathered great importance in social network and Web2.0. When the power of web service provider has progressed tremendously, the web service provider offers an interface which given users. Users can use the service on their own web site. The providers of Open API are listed as follows: Google provide the information of world, Amazon provides the books, DVD, and audio CD of the world and the others. The speed of service is the importance in the world, the web service is too. The importance thing of speed of web service is bandwidth. Over the past decade, the growth of bandwidth is consumed by Open API. The growth of Open APIs show that more evidence that web services drive network effects. Today, the growth of bandwidth supports the requirement of new web application. A web service provider creates a new service to

promote cooperation between the main idea and the other Open API of web service provider. Hence Open API provider is creating many kind of new web service as you can.

In many web applications, the service of electronic commerce base has huge commercial purposes. Try to imagine that we can do “shopping on Internet” which is not only meaning that you can buy something on Internet but also you can see the street at the same time. “Shopping on Internet” can implemented through Open API. The tour guide system greatly resembled “shopping on Internet” system. For example, National Palace Museum has a tour guide system which provides a web interface touring the work of art on Internet. When a system integrates electronic transactions and tour guide on Internet, the system will provide a shopping service on Internet. In this paper, we use the Google Map API to provide the Map data which is the bridge between users and our system. The name of system is interactive panorama street view map tour guide system which provides shopping function on Internet.

As was mentioned above, the shopping system is composed of several parts. First of all, we need take pictures of a shop and record longitude and latitude of a shop. Next we need to stitch a picture onto next picture and previous picture and cut off a panorama picture by shop. Then building a database is used for queries. This part is described in Chapter 3. Last of all, we provide a web interface which is provided query from user. When users shop on Internet, there have two operations which a system provided. One is a point query on a map which is provided by Google. When users click which records longitude and latitude on a map, the system will return a result of the click event which become the start point of shopping .The other operation is routing which is meaning that user selected several points on a map. Then a system will show a path for selected points and street view of request. The second operation is similar to path-finding which is a search problem of a graph.

The conventions query which needs the user type key word or a query term is a text-base query. The target of a conventions query is also a text-set. When the data is not organized, the time for each query is consuming. The previous step of query is building the index of data. The index of data is features of data. This is base on the type of data. The purpose of building index is that enhanced the speed of query.

In this system, the element of data set is images and longitude and latitude of images. As has been discussed, the feature of data is an importance thing to organize data. We need a reorganize method which enhanced query data. There are many solutions of stitch images, for example, Autostitch, photostich and the others. There are many databases e.g., SQL, MySQL and the others, solutions on Internet. The purpose of this study was the feature of data set and query algorithms. The query algorithms can solve the closest point problem and path-finding.

1.2 Research Questions



In detail, the primary research question that we address as follows: we present an algorithm of a point query which locate the closet image file of a given point and a path finding algorithm to plot the shortest path from a point to another. The point query algorithm is a method which finds the nearest point with query point. The distribution of points is an important feature of the point query algorithm. The data collection instruments uses with the digital camera and the GPS tracking device in this thesis. For each data consists of two part which the image of street and the GPS information of street at the same time. The features of data were observed is the data distribution which is similar to the road of city. Although the closest problem is solved in the several decades, we present the algorithm base on the feature of data [2, 3].

Path-finding is a commonly problem in the graph theory. The solution of path-finding is a graph search algorithm. Although there have many algorithms are used in the graph theory like DFS, BFS, and Greedy BFS, the A-star algorithm is a proper solution in this thesis. A-star is a commonly solution of path-finding in AI. It is used in a maze and Rubik's Cube. However, we selected A-star algorithm to solve path finding in this thesis. It is discussed in chapter 3 which the algorithm.

1.3 Related work

The closest point problem and path finding are commonly problem in computer geographical. The closest point problem and path-finding are a point search problem. The differentiation of the closest point problem and path-finding is the type of answer. The closest point will answer the point which is the nearest with the query point and path-finding will answer the best path from start point to end point. The following are terms of the closest point and path-finding.

First of all, the closest point problem, also referred to as the post-office problem, was due to Knuth [1]. It is stated as follows. There is a post-office which is closest to my current position. If you are in NCTU, then the answer may be is the post-office in 中正堂.

Various definitions of the closest point have been proposed over the course of decade of research. The closest point search, also known as nearest neighbor search, proximity search or similarity search, is an classic problem for finding closest point in metric space. The following is the definition of general metric space: In mathematics, a metric space M which consist a set S and a distance function d denotes $M=(S, d)$.

A distance function d satisfies:

Non negativity: for all s, t in S : $d(s, t) \geq 0$.

Symmetry: for all s, t in S : $d(s, t) = d(t, s)$.

Identity: $d(s, t) = 0 \Rightarrow s = t$

Triangle inequality: for all r, s, t in S : $d(r, t) \leq d(r, s) + d(s, t)$.

Euclidean space is a metric space. For example, in two-dimensional space \mathbb{R}^2 , for all points P, Q in \mathbb{R}^2 , the distance function d is Euclidean distance. Euclidean distance satisfies the definition of a metric space.

The problem is described as follows: given a fixed finite subset S in metric space, so that given a query object, one can effectively determine its closest object in metric space. The formal definition of the closest point problem is described in Chapter 2.

In this section, path finding problem is described as follows: Path finding in computer geographical is commonly approached as a graph search problem. If a problem is described as a graph model, then a problem is similar to solved Path-finding problem. Path-finding is used in a wide variety of games, it refer to AI finding a path around an obstacle. The purpose of the path-finding algorithm is finding a best path from A to B . A number of varying definitions have emerged. The mostly definition of best path is the less cost of the path. In a computer geographic, the less cost of the path has a name which is the shortest path.

In this thesis, path finding is used in the map service, it refer to finding a path form a point A to a point B on a map. There have several algorithms which are solved the path-finding problem is described as follows: DFS, BFS, greedy BFS, Dijkstra's algorithm and A-star algorithm in Chapter 2.

1.4 Overview

This dissertation is organized as follows. Chapter 2 presents related work. Chapter 3 presents the algorithms and system architecture. Chapter 4 discusses the results in the system design. Finally, Chapter 5 concludes this thesis and describes the future work.



Chapter 2 Related Work

In this research, we focus on the closest point and path finding problems. The closest point and path finding are important problems in computer geographical applications and have been studied by several researchers. In the following sections, we define the problems and describe the related research work.

2.1 The closest point problem

The closest point problem is a query problem which answers for the point of closest point of each query point. Traditional queries technology implies exact-match. For example, a point (2, 9) is in the set S or not. An answer of query will be yes or not. Effective index (B-tree, Hashing) and query processing have been developed. However, the purpose of exact-match does not suit with all applications. New applications have much complex needs. The closest point problem is an instance of new application. The closest point problem can also be a similarity query. The result of a similarity query is the point that has a minimal distance with the query point.

The data set of the closet point problem is the element of Non-Euclidean geometry space or Euclidean geometry space. The date set belongs to a metric space. The definition of a metric space has discussed as follows:

A metric space is a set S with a global distance function d that for given two point A , B in S , where $d(A, B)$ is the distance from a point A to a point B . $d(A, B)$ is a non-negative real number. A metric space must stratify following criteria:

Non negativity: $\forall s, t \in S: d(s, t) \geq 0$.

Symmetry: $\forall s, t \in S: d(s, t) = d(t, s)$.

Identity: $d(s, t) = 0 \Rightarrow s = t$.

Triangle inequality: $\forall r, s, t \in S: d(r, t) \leq d(r, s) + d(s, t)$.

The closest point problem is described as follow:

Problem 1 Give a set S of n points in \mathbb{R}^D , store it in a data structure such that for any query point q , $\forall p$ in \mathbb{R}^D , we can find its nearest neighbor.

i.e., a point p in S that is closer to a query point q ,

$$\text{The closest point } P = \min \{d(q, p), \forall p \in S\}$$

In general case, a function d is a distance function which is striking similarities between the two points. A unit of distance function depends on the kinds of application.

In this thesis, the data set referred is defined as the element collecting in a two-dimensional space. Before we introduce the algorithms of the closest point problem, the definition of the data is described as follow: In next section, we describe the algorithms commonly used to solve the closest point problem in two-dimension, that is, in plane. Each data element is represented as a set of point $\{p_1, p_2, p_3 \dots\}$, where for all $p_i = (x_i, y_i)$ and x_i, y_i are real numbers.

2.2 The algorithm of the closet point problem

There have been many algorithms developed to solve the closest point problem. They are described as follows. In first method, it is easy to implement. The main concept of Naïve method is that to measure the distance between all points and the query point. That is the simplest idea in this problem. Obviously, Naïve method is an

ineffective in solving the closest points. Quad tree [2] and kd-tree [3] are commonly data structure in spatial data. They have the same concept which partitions off space of two-dimensional. In this thesis, the space discussed is two-dimensional. Each search stage, Quad tree and kd-tree kept the part of search target. That has the same concept with binary search. Quad tree uses x-coordinate and y-coordinate to locate the query of point in each branch. Kd-tree uses x-coordinate and y-coordinate alternate to locate the query of point in each branch.

In Table 2-1, we discuss the differ part of kd-tree, quad tree and octree.

Table 2- 1 An compare of data structures

Name	p	s
Kd-tree	1	2
Quad tree	2	4
Octree	3	8

p is the times of sub-divided partition space and s is the number of region of a tree node. In Table 2-1, kd-tree and quad tree are a tree in a plane and octree is a tree in a cube.

2.3 Path finding

In graph theory, we can use a graph to represent the map. In a graph, a vertex represents a node and an edge represents a direct path of two nodes. Path finding is a graph search in computer geographical. A graph search is finding a path or not between two nodes in a graph. There have two definitions of the path which found.

Firstly, the path exists and the cost of path does not care. Next, the path which found is a shortest path which is the cost of the found path is minimal.

2.4 The Algorithm of Path finding

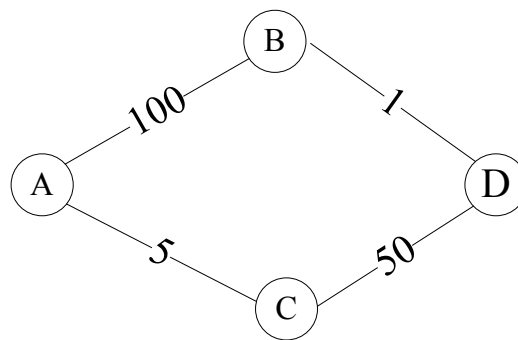
Many path finding algorithms have been presented. Firstly, Depth first search is similar to a preorder tree traversal, while breadth first search resembles a level order tree traversal. DFS and BFS are elementary graph operations.

The strategy of depth first search is to search “deeper” in the graph. When we start the search with a graph G and a begin vertex s . Then we choose a vertex unvisited. If the vertex unvisited is goal vertex, it is finished. If the vertex unvisited is not goal vertex, we do the same action recursively.

In DFS, if two vertices are connected, there is a path which is not always the best path in the graph. The strategy of breadth first search is to check the goal node which is neighbor of selected node. In breadth first search, we start the search with a graph G and a begin vertex. We select an unvisited neighbor node of a begin vertex. If the goal node is not in an unvisited neighbor node of a begin vertex, then we select a neighbor node which is begin node and we do search action recursively. The path is found by BFS is also not a best path. Because the strategy of algorithm is “Blindness”, there is no any information about the goal node.

Greedy BFS is BFS with a greedy method. A main greedy method is the minimal distance of a list of candidates with the goal node. A greedy method is used in “We select an unvisited neighbor node of a begin vertex.” In greedy BFS, we measure the distance between all unvisited neighbor nodes and goal. Then we selected the node which has a minimal distance with goal node be the next begin node. Although greedy BFS is better than BFS in some case, greedy BFS is not an optimal

algorithm. The algorithm is not the information about a start node.



Figure

Figure 2-1 An example of graph

In Figure 2-1, if node A is the start node and a node D is goal node, the path of greedy BFS is A→B→D. The best path is A→C→D because the cost of best path is 55. The cost of greedy BFS is 101. Hence greedy BFS is not optimal. Dijkstra's algorithm [4, 5] solves the single-source shortest-paths problems on a weighted directed graph. Dijkstra's algorithm is the one of greedy method. The strategy of Dijkstra's algorithm is selected next the minimum node from sub-tree spanned in algorithm. There has the information from the start node. Although Dijkstra's algorithm found the shortest path, the effect of selected path is not a good enough. The size of spanning tree is larger than A-star.

In computer science, A*[6] (pronounced "A star") is a best-first, graph search algorithm that finds the least-cost path from a given initial node to one goal node. A* is commonly used to find the solved path in AI of game. The solved states are steps of solved problem. It can represent a tree-base graph. If we can reduce the time of branch decide, we can find the least steps to the goal. As was mentioned above, the problem-solved steps can map a graph. Finding the problem –solved steps is referred as a shortest path finding. A* is also a greedy algorithm. The strategy followed by A* is to reduce the number of the neighbors of the node which choices. In a fixed gird, if we want find a shortest path between two girds. In Figure 2-2 shows that the order of

visited grids more like rippling in Dijkstra's algorithm. Simply stated, Dijkstra's algorithm tries many not good paths and the order of visited grids more like a river which flows to a goal in A* algorithm.

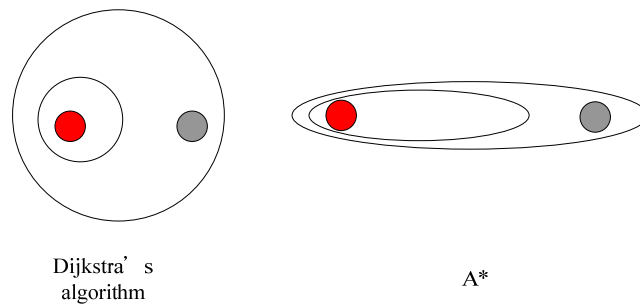


Figure 2-2 The shape of spanning tree

The reason of reduces t reduces the number of the neighbors of the node which choices in A* algorithm is that there are the information of start node and a goal node. It is described in Chapter 3.

The information of a start node and a goal node are importance factors in Path finding. Table 2-2 show that the factor of the information of the start node and goal node with several path finding algorithms.

Table 2-2 The compare of algorithms

Name	The information of start node	The information of goal node
DFS	No	No
BFS	No	No
Greedy BFS	No	Yes
Dijkstra's algorithm	Yes	No
A* algorithm	Yes	Yes

Chapter 3 System Design and Implementation

3.1 System Overview

In this research, we aim to create a street tour guide system. The system accepts two types of query from a user, which are the point query and the path query. When the user selects a point from a map and sends a point query, our system will return the street images that are closest to the selected point. When the user selects a set of points from a map and sends a path query, our system will generate the path connecting the selected points and then return the panorama street view of the path. Figure 3-1 shows the system architecture. In this section, we will describe the solutions of the closest point and path finding problems.

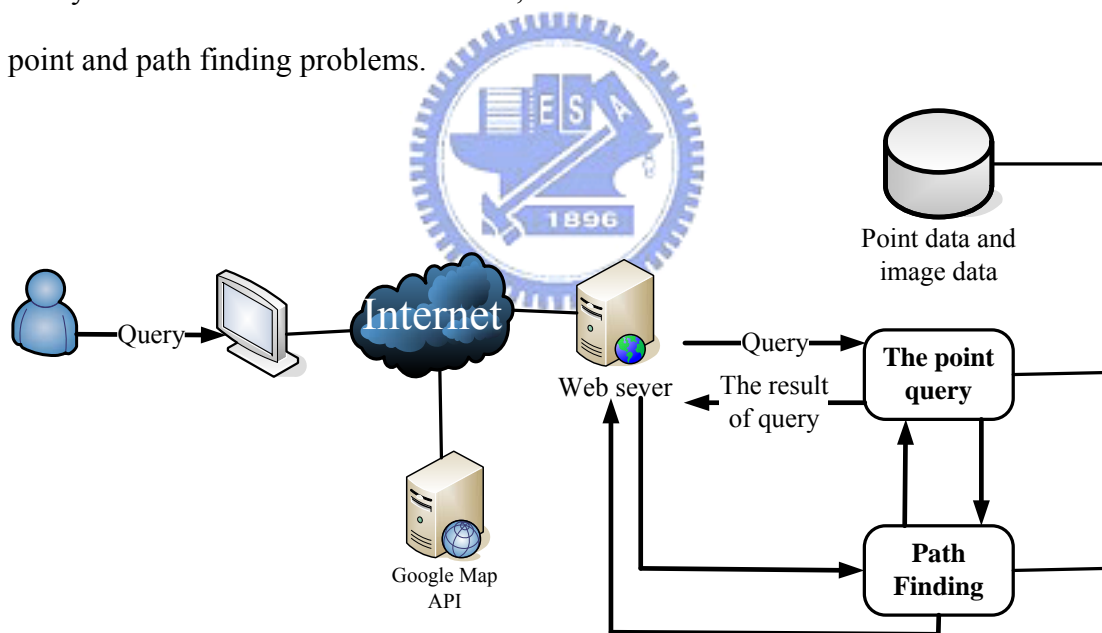


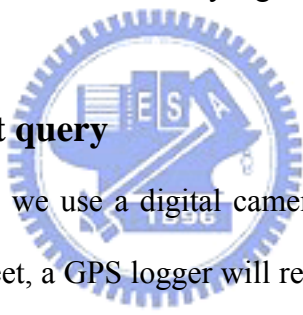
Figure 3-1 System Overview

3.2 Point Query

3.2.1 Overview

In this section, we present our method of point query which is a solution of the closest point problem. Although the closest point is a common problem in computer geographic, we observe the data distribution which helps to solve this problem in different ways. When user requests a point query, the result of query is the photo of street of the closest point. The speed of the naïve method of point query which is full search is slow. There must have a method which reorganizes the corresponding point of street. Because the distribution of the photo is similar to the street, hence we organize the point of photo of the same street by a group.

3.2.2 The method of point query



In collecting data stage, we use a digital camera and a GPS logger. When we take a photo of a shop of a street, a GPS logger will record the location information of the photo of the shops. When we decide a path for collecting data, the order of data which are the photo of a shop and the location of the shop is similar to a road of urban. In past solution of the closest point problem, the data are constructed to form a specific data structure for query.

The element of our collected data consists of two parts. One is the photos of shops; the other is the location information of shops. The data which collected we observe that serious points approximates for several lines. The feature shows that the lines divided the space to several polygons. Firstly the group classified method help us to classify points to several sets. The sets named *Group* in this thesis are described as follow.

A *Group* is a set of ordered points whose distribution approximates to a line

segment. These points are the GPS locations of shop photos. The group classified method is described as follow, the method of group classified is a method of corner detect. The point set is $\{P_1, \dots, P_n\}$. If the point P_i is a boundary of group, there have two situations. One is that P_i is P_1 or P_n ; the other is that P_i is a corner point. A corner point is the connection of two straight lines. The slope of two straight lines has difference. Two lines are not parallel. If we have two straight lines, we can decide a point P_i . But we need to decide a point P_i in a point set. If a point P_i is a corner point in a point set, the peak value of P_i has a difference between other points. The method of measure the peak value of P_i is measure an included angle of P_i and forward points or backward points. We pick up the forward points and backward points in fixed number of points. The corner point has a feature which shows that the points are near the corner and included angles of those points are similar to each other. If there is a point P which shows that an included angle of the point is suitable for the threshold of a corner point and the next point and previous point are not suitable for the threshold of a corner point, the point P is not the corner point. Hence the corner point P is one of the points which are suitable for the threshold of a corner point. The method of decide the corner point P is the minimal included angle of the points which are suitable for the threshold of a corner point.

The algorithm of the group classified method is described as following.

1. The fixed number is R .
2. PS is the point set. SPA is an array of slope of PS
3. for ($\forall P_i \in PS$)
4. {
5.
$$S_i = \frac{1}{5} \times \sum_{R=1}^5 \frac{X_{P_{i-R}} - X_{P_i}}{Y_{P_{i-R}} - Y_{P_i}} \quad S_i' = \frac{1}{5} \times \sum_{R=1}^5 \frac{X_{P_{i+R}} - X_{P_i}}{Y_{P_{i+R}} - Y_{P_i}}$$
- 6.
- 7.

8. $\angle S_i = \tan^{-1}(S_i), \angle S_i' = \tan^{-1}(S_i')$
9. $\angle A_p = |\angle S_i - \angle S_i'|$
10. }
11. Delete there has the point P_i which $\forall SPA[i] \leq \text{the threshold of corner}$ and $\forall SPA[i-1] > \text{the threshold of corner}$ and $\forall SPA[i+1] > \text{the threshold of corner}$
12. for($\forall SPA[i] \leq \text{the threshold of corner}$)
13. {
14. The number of a list of candidates is C
15. Find the midpoint of included angles from P_i to P_{i+c-1}
16. The midpoint P_k is the corner point
17. }

The result of the group classified method is the set of corner points. This is a method to simple classified a points set to several lines.

$$\{P_1, \dots, P_n\} \longrightarrow \{P_1, \dots, P_i\} \dots \{P_{i+1}, \dots, P_j\} \dots \{P_{k+1}, \dots, P_n\}$$

The method of the group classified method is the set of corner points is a simple method because the distribution of data does not need to consider the complex shape.

The method of lines construct to the groups is described as follows.

1. The center C_i of a group is a midpoint of line.
2. The radius R_i of a group = the half length of line.
3. A line segment formula measure with $\{P_{i+1}, \dots, P_j\}$
4. The neighbor list of groups refreshes.

The *groups* divided the space to several polygons. The field of a group node is described as following.

$\forall G_i \in \text{groups set}, \text{ where } i \in \mathbb{Z}^+$

groupID : the name of group

C_i : the midpoint of the line

R_i : the half length of the line

F_i : the line segment formula

L_i : the neighbor list of G_i

M_i : the members of G_i

In this section, we present a method to query the closest point. If the closest point is in one of groups, the method of decided the closest point is finding the closest point in the group. The minimal distance between a point and a line segment is the point line distance or one end of a line segment shows that in Figure 3-2.

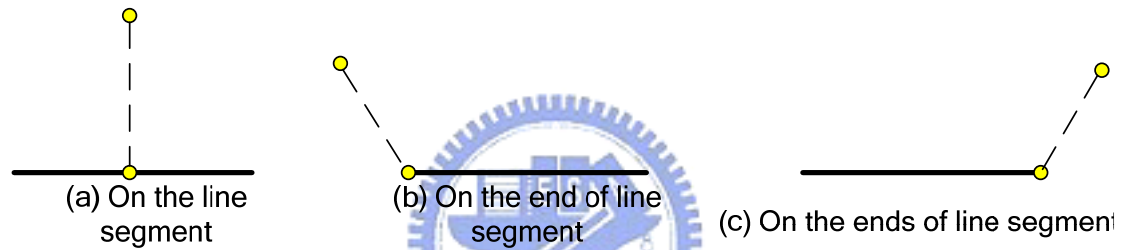


Figure 3- 2 A position of the closest point

First method of selected a group of the closest point measures the projection point for all groups. Obviously, First method is not an effective method. There have features of the group of the closest point. First idea shows that the closest group contains the closest point. Figure 3-3 (a) shows that the closet group does not contain the closest point.

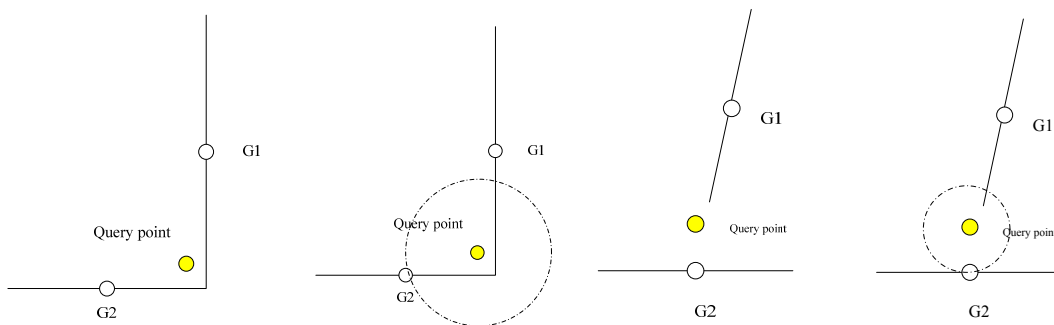


Figure 3- 3 A position of the closest point

Although the closest point is not in the closest group, the closest point is the range of

center which is query point and a radius which is the distance between G2 and query point in Figure 3-3 (b). Next idea shows that the closest point is in the closest group or the neighbors of closest group. Figure 3-3 (c) shows that the closest point is not in the closest group and the neighbors of closest group. The location of the closest point relates to the location of the closest group. The group of the closest point is in the range of center which is query point and a radius which is the distance between the center of the closest group and query point in Figure 3-3 (d). Hence the location of the closest point is in the K nearest neighbor of group. In this thesis, the number K selected 3. The distribution of groups is similar to the road of urban. The distribution of groups is random distribution. The groups divided the space to several polygons. Because we need to query K-nearest neighbor, the distribution of group suits the kd-tree to query. Hence we use groups to build a kd-tree. The kd-tree helps us to query K-nearest groups. The kd-tree is a good data structure for data query in spatial space. In the distribution of groups is random distribution, the nearest query of kd-tree is a highly effective method. Hence we select the kd-tree to enhance the speed of query K-nearest groups in this thesis.

The algorithm of point query

Figure 3-4 shows the structure of the algorithm which has two part described as following.

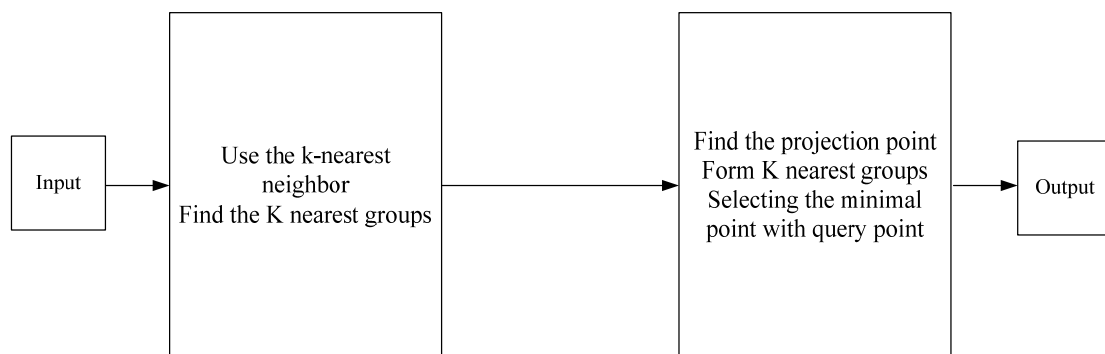
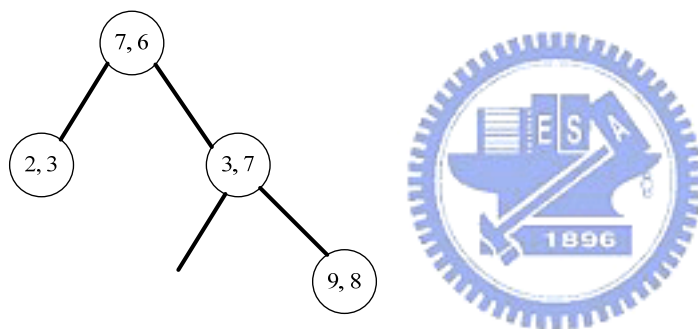
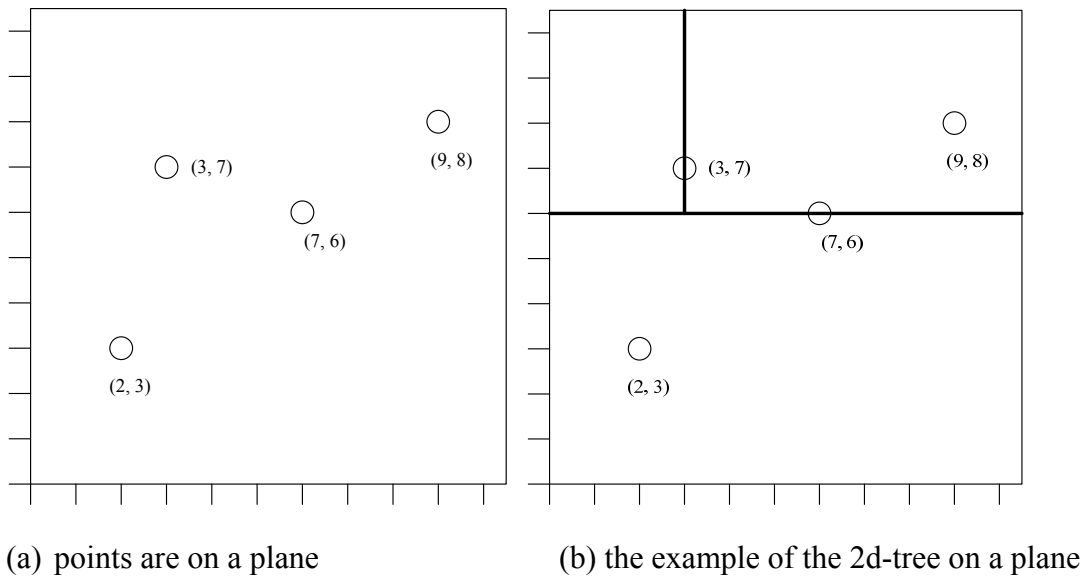


Figure 3- 4 The flow of the algorithm of point query

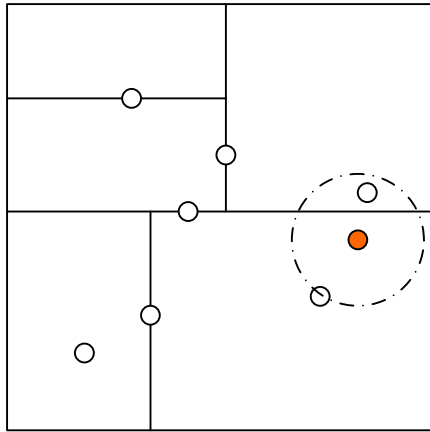
Before we describe the closest point query of kd-tree, kd-trees described as follows for example we have a point set $\{(2, 3), (3, 7), (7, 6), (9, 8)\}$.



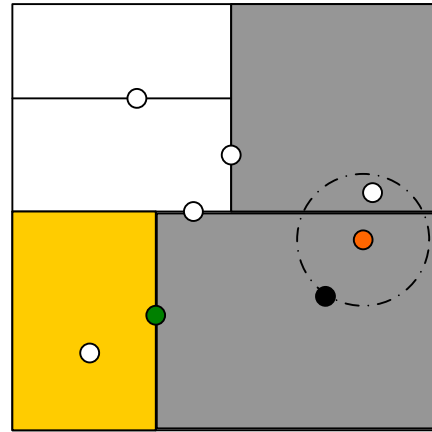
(c) the tree-view of the 2d-tree

Figure 3-5 An example of a 2d-tree

Figure 3-5 (c) shows that a kd-tree is a 2d-tree, which represents the nodes of the two dimensional space. The nearest neighbor query of kd-tree consists of two main parts. First part, each branch of query locates the position of query point. It is similar to the binary search in Figure 3-5 (c).



(a) the position of query point



(b) the approximate closest point and range check

Figure 3- 6 An query of kd-tree

Next part, check the range of circle has a closer node or not. Figure 3.6 (a) shows that the end of the branch and find the closest point so far. Figure 3.6 (b) shows that the distance of red dot and green dot which is the parent of black dot is far than the distance of red dot and black dot. Hence, the dots which are the child of green dot the range of yellow does not need to search and the range of gray color needs to search.

Second part of point query which is projection method is described as following.

The closest point of the group is the projection point of line or two end of line.

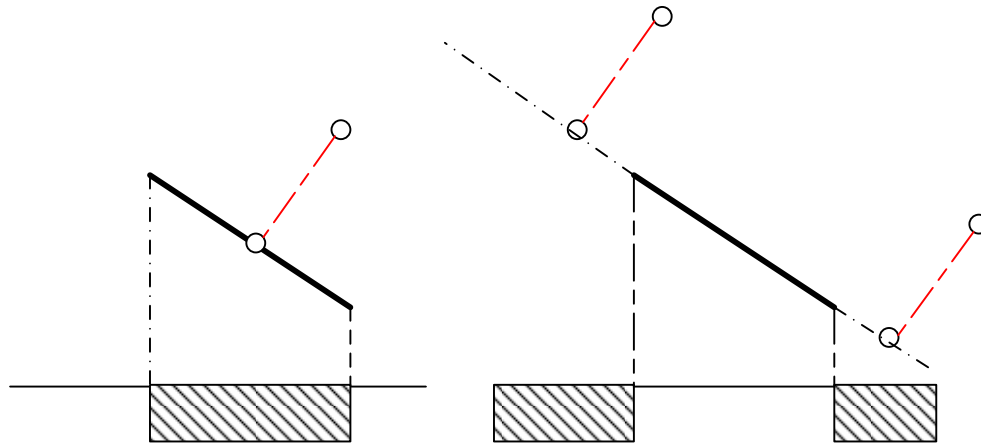
The formula of projection point

A point $Q(x_0, y_0)$ a line segment $L: ax + by + c = 0$

A projection point P

$$\left(x_0 - a \times \frac{(ax_0 + by_0 + c)}{a^2 + b^2}, y_0 - b \times \frac{(ax_0 + by_0 + c)}{a^2 + b^2} \right) \text{-----} (3.1)$$

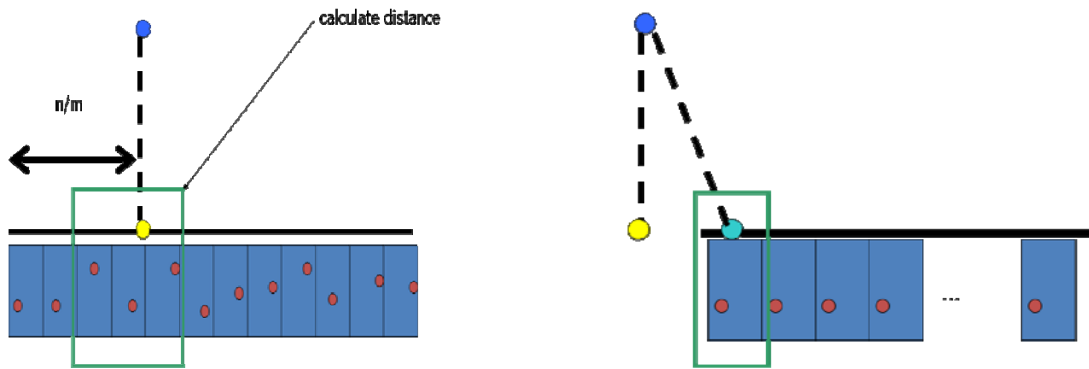
A projection point P is on a line but is not always on the line segment. Hence the position of projection point is on the line segment or not.



(a) a projection point is on the line segment (b) a projection point is not on the line segment

Figure 3-7 An position of the closest point

Figure 3-7 shows that the ends of line are boundary of projection point is on a line or not. Then we measure the closet point of the group, in case of projection point on the line segment described in Figure 3-7 (a). When we measure the projection point which can get the location of projection point on the line segment are the points of nearest projection point. Then we calculate the distance with three point of near projection point. Figure 3.7 (a) shows that use the rate of a length of line segment and a length of projection point and start point. The order of a projection point is the i -th point. In the end, we calculate the distance between $\{P_{i-1}, P_i, P_{i+1}\}$ and a query point. Then it selects a point of the minimal distance.



(a) the green rectangle contains the closest point (b) the green rectangle contains the closest point

Figure 3-8 A position of the closest point in a group

The other case which is projection point is not on the line segment. Obviously, the closest point of group is the end of line segment in Figure 3.7 (b). Then we calculate the distance with blue point which is the position of end point.

In the second part of point query, we need do projection method in K times. We measure the closest point in K groups with projection method in K times. Then we select the point which has the minimal distance with a query point from K candidates of the closest points. In this thesis, we select $K=3$ which provides correct location of the closest point in the data set presently.

3.3 Path Finding

In this chapter, a router operation of GIS system is a common function. Router operation is a search function, which mean there exist a path from start node to goal node or there does not exist a path. In the thesis, groups are referred as nodes of a graph and relation of two groups is referred as an edge of graph. Although there have many algorithms for path finding problem, A-star is an effective algorithm of them. A-star is described as following. A-star is a best-first algorithm for general search of optimal paths from node A to the other. Dijkstra's algorithm is a similar algorithm with A-star. The concept of A-star is the information of start node and goal node. In figure 3-9, there has a shortest path P , a node N is a node of the shortest path. The reason of a node N selected is that the cost form start node to node N is minimal in Dijkstra's algorithm and the cost form start node to node N and form node N to goal node is minimal in A-star algorithm.

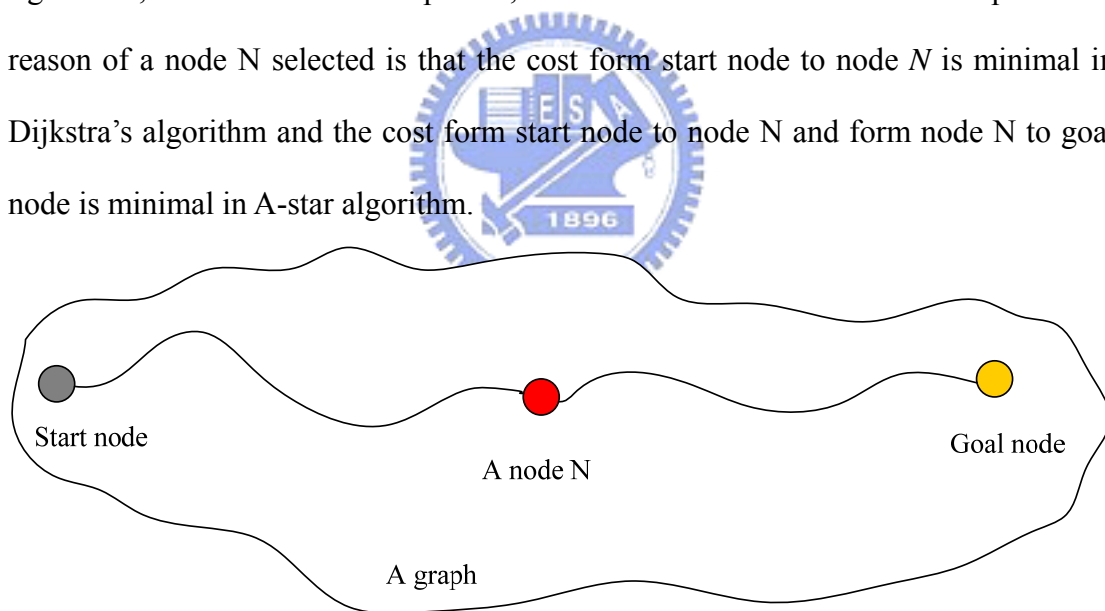


Figure 3-9 The shortest path in a graph.

A-star algorithm is a heuristic algorithm which ranks each node by an estimate of best-first choice through the node. The typical formula is described as follow:

$$f(n) = g(n) + h(n)$$

where n is a node of a graph

$g(n)$ is the minimal cost form start node to node n

$h(n)$ is an estimate cost form node n to goal node

The $g(n)$ value is similar as the length of previous path in Dijkstra's algorithm,

$$g(n) = g(n-1) + c(n, n-1),$$

$$\text{where } c(n, n-1) \doteq \left\{ \begin{array}{l} \infty, \text{ node } n \text{ and node } n-1 \text{ are not neighbors} \\ \text{the cost } C_1, \text{ node } n \text{ and node } n-1 \text{ are neighbors} \end{array} \right\}$$

$g(n-1)$ is g value of the parent node of node n .

$h(n)$ is a method of heuristic estimate for the remaining path in best-first search.

A-star is guaranteed to find the shortest path, as long as the heuristic estimate, $h(n)$, is admissible that is, it is never greater than the true remaining distance to the goal node.

The greedy method is only estimate the f value of neighbors of node n which selected node of minimal cost.

In this thesis, $g(n)$ and $h(n)$ are described as following

$$g(n) = g(n-1) + c(n, n-1),$$

$$\text{where } c(n, n-1) \doteq \left\{ \begin{array}{l} \infty, \text{ node } n \text{ and node } n-1 \text{ are not neighbors} \\ R_n + R_{n-1}, \text{ node } n \text{ and node } n-1 \text{ are neighbors} \\ \forall R_i \text{ is a radius of group } G_i \end{array} \right\}$$

$g(n-1)$ is g value of the parent node of node n .

Let lat_n and lng_n are latitude and longitude of node n

lat_{goal} and lng_{goal} are latitude and longitude of goal node

$$rad(d) = d \times \frac{\pi}{180}, \text{EARTH_RADIUS} = 6378.137$$

$$radlat_n = rad(lat_n), radlng_n = rad(lng_n), radlat_{goal} = rad(lat_{goal}), radlng_{goal} = rad(lng_{goal})$$

$$a = radlat_n - radlng_n$$

$$b = radlat_{goal} - radlng_{goal}$$

$$h(n) = 2 \times \text{Asin} \left(\sqrt{\sin^2\left(\frac{a}{2}\right) + \cos(radlat_n) \times \cos(radlat_{goal}) \times \sin^2\left(\frac{b}{2}\right)} \right) \times \text{EARTH_RADIUS}$$

$h(n)$ is a distance function between two points of the earth.

This $h(n)$ is satisfied that it is never greater than the true remaining distance to the goal node. The straight distance is the minimal distance between two points. Hence

$h(n)$ is satisfied this property.

A-star algorithm is described as following:


```

A-star(start node  $s$ , goal node  $g$  )
{
   $OPEN$  = priority queue
  Add  $s$  to  $OPEN$ 
   $CLOSED$  =  $\emptyset$ 
  while( $g \notin OPEN$ )
  {
     $current$  = Remove lowest rank item from  $OPEN$ 
    Add  $current$  to  $CLOSED$ 
    for( neighbors of  $current$ )
    {
       $cost_{current} = g_{current} + R_{current} + R_{neighbor}$ 
      if(neighbor  $\in OPEN$  and  $cost_{current} < g_{neighbor}$ )
      {
        Remove neighbor from  $OPEN$ 
      }
      if(neighbor  $\in CLOSED$  and  $cost_{current} < g_{neighbor}$ )
      {
        Remove neighbor from  $CLOSED$ 
      }
      if(neighbor  $\notin OPEN$  and neighbor  $\notin CLOSED$ )
      {
         $cost_{current} = g_{neighbor}$ 
        Add neighbor to  $OPEN$ 
        Set priority queue rank neighbor  $f_{neighbor} = g_{neighbor} + h_{neighbor}$ 
        Set neighbor's parent to current
      }
    }
  }
  reconstruct reverse path from goal to start by following parent pointers
}

```

In this thesis, we select a series points $\{P_1, P_2, \dots, P_n\}$ form user interface. Firstly, Point Query queries the group g_i of a point $P_i \forall i=1, \dots, n$. Then group list $\{g_1, \dots, g_n\}$ divided to $\{g_1, g_2\}, \dots, \{g_{n-1}, g_n\}$. Finally, let $\{g_{i-1}, g_i\}$ be the start node and goal node of A-star algorithm, then we combine the paths of A-star to a path form

a point P_1 to P_n .

3.4 Summary

In this chapter, we present the algorithm of point query and define the function of A-star in our system. We use the feature of data set to solve the point query algorithm in other way. The analysis of the point query algorithm presents in Chapter 4.



Chapter 4 Analysis and Evaluation

In this thesis, we present a closest point finding algorithm for point query and an A-star based path finding algorithm for path query. In this section, we will evaluate the time complexity of our proposed algorithms with other methods.

4.1 Closet Point Finding Algorithm

In the closest point finding algorithm, we organize the points into groups and represent the groups by kd-tree. The algorithm is an improved kd-tree algorithm for the virtual tour system. To evaluate the time complexity of the algorithm, we compare our algorithm with the traditional kd-tree algorithm.

In the closest point finding algorithm, we use the distribution of data to develop the method. We use the nearest neighbor query of kd-tree to enhance the speed of the closest groups. Before we discuss our closest point finding algorithm, we discuss the processing of the nearest neighbor query of kd-tree.

Recall the query processing of kd-tree. There have two parts in the query processing of kd-tree. First part is the search hierarchy of branch and bound which various tree-base data structure for metric space. First part of query processing traverses the tree to find the rectangle leaf containing the query point. The data point in the rectangle leaf is not always the nearest neighbor. The other part of query processing checks the parent of the rectangle of the query point and his sibling. In Table 4-1 show that the time complexity of the nearest neighbor of kd-tree.

Table 4-1 The time complexity of nearest neighbor of kd-tree

	The time complexity
Binary partition	$\log n$
Range Check	P is the number of range check

Generally the random distribution of data shows that the number of range check is a constant number. Hence the time complexity of nearest neighbor of kd-tree is $\log n$. The bad distribution of data which is similar to a circle or a rectangle shows that almost all nodes to be inspected in Figure 4-1. Hence the bad distribution is a cause of worst case of nearest neighbor query and the time complexity is n .

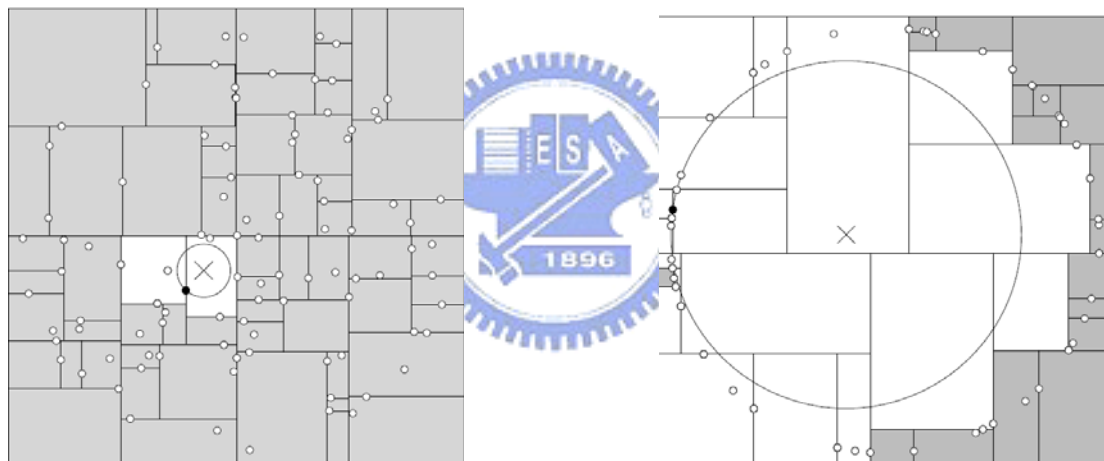


Figure 4-1 The two distribution of a kd-tree [7]

Although the distribution of data is not similar to a circle or a rectangle in this thesis, the distribution of groups divided the space to several polygons. Hence the number of range checks equals the number of the members of three or four group. The number of the member of three or four group is large than the number of range check in the random distribution of data. Table 4-2 described the cost of operation in three cases.

Table 4-2 the time complexity of kd-tree

	Binary partition	Range Check	Total cost
Average case	$\log n$	P	$\log n + P$
Worse case	$\log n$	n	n
In this thesis	$\log n$	$P(40\sim 120)$	$\log n + 120$

In the thesis, the number of point is n and the number of the member of a group is 10~30. Hence the number of groups is $\frac{n}{10} \sim \frac{n}{30}$. The process of our method has four parts. First part is group classify which classifies the point data as line segment. In data collected stage, we use GPS logger which records the points which we take the pictures of street. A track for each path traversing, a position of GPS device and weather influenced the result of locate processing of satellites. When we analysis the data of GPS logger, there have some error which produced by locate processing of satellites.



Figure 4- 2 A figure of the groups

Figure 4-2 show that the result of the group classified method. A red line segment is a group. The midpoint of line is the group index. We can find that groups divided the space to several polygons. The group indexes are evenly distributed in the space.

In second part, we use the result of group classified be the element of kd-tree.

The distribution of group indexes is similar to radon distribution. In this stage, we need to find the group which contains the closest point. The distribution of group indexes is suitable for used the nearest neighbor query of a kd-tree. As suggested above, the member of the closet group is not always the closest point. A query point is the center of a circle and a radius is the distance between a query point and the closet group. The closest point must locate in the circle. Hence we select a list of candidates of groups which contains the closest point. The method of select a list of candidates of groups is k nearest neighbor query of a kd-tree. The number of candidates of groups is three. This can be seen in the following in Figure 4-3.

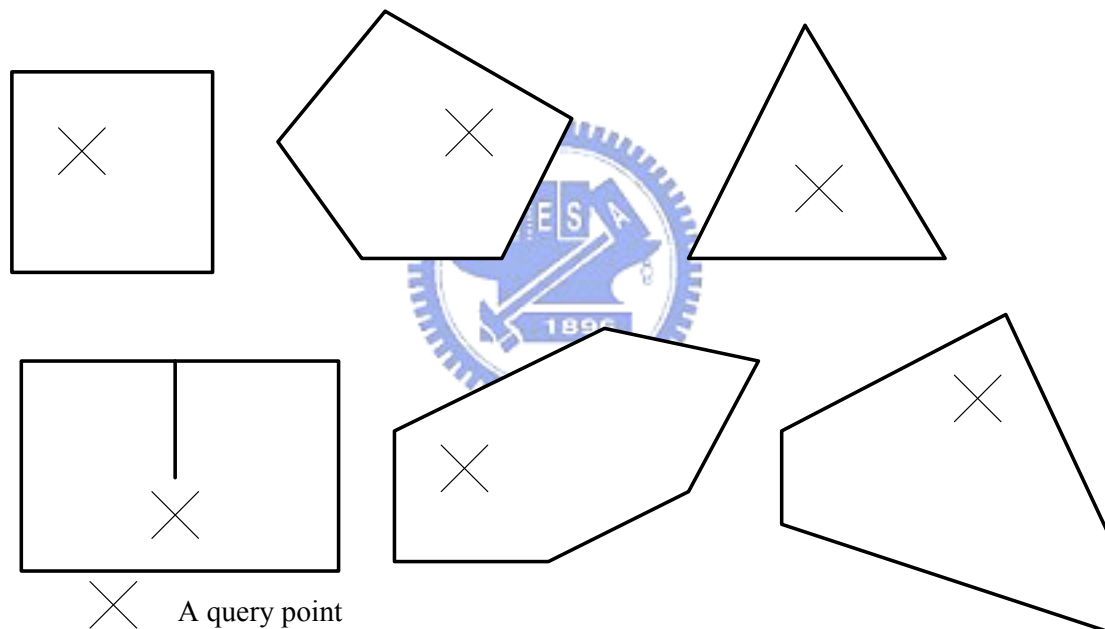


Figure 4-3 The typical shapes of urban roads

In Figure 4-3, the result reflected indicates that the position of the closest point is on one of the three nearest groups. In this stage, the number of candidates of groups is small for reduce the cost of projection point. Although there may be have the shapes of polygon is not satisfy the rule of candidates of groups, the probity of this situation is small.

In last step of query processing, the projection point method which decided the

closest point of a group. The steps of the projection point are described as follows. First step show that measure projection point. Then measure the location of the closest point on a group. Final step is calculated the minimal distance base on the location of the closest point in step 2. The cost of each projection method is five operations. There have do projection method in $5K$ times because the number of candidates of groups is K .

Table 4-3 The compare with kd-tree

	Binary partition	Range Check	Projection method	Total cost
Point query	$\log m$	H	$5K$	$\log n + H + 5K$
kd-tree	$\log n$	$P(40\sim 120)$	0	$\log n + P$

In Table 4-3, H is the number of range check of the closest group. The distribution of group indexes shows that H is similar to the average case of nearest neighbor query of a kd-tree. Hence H is a constant number. The cost of the projection method of the point query is $15(5*3)$ in this thesis. $5K$ is a fixed number.

Table 4-4 The compare with kd-tree, m is the number of group

	Binary partition	Range Check	Projection method	Total cost
Point query	$\log m$	H	$5K(15)$	$\log m + 5K$
kd-tree	$\log n$	n	0	n

In Table 4-4, the distribution of data is similar to a rectangle. The distribution is a bad distribution to the nearest neighbor query of kd-tree [7]. That is worst case of kd-tree

in Figure 4.1. The processing of point query will be group classified. The number of groups is m . Then K nearest neighbor query of groups kd-tree. Final step is projection method. We find that the cost of point query is $\log m + 5K$. The result shows that the distribution is similar to a set of polygon is suitable for use point query.

4.2 Path Finding Algorithm

The other part of this thesis is path finding. In this thesis we use A-star algorithm which is a search algorithm in AI. A-star which is a flexible algorithm is suitable for use to solve problem. The definition of the f function is described in Chapter 3. Although Dijkstra's algorithm is one of shortest path algorithm, the size of spanning tree is larger than A-star in fixed grid. In this thesis, path finding is a function which connects several points with a path. There have a shortest path between the node i and the node $i+1$. Hence we can find a path which connects from node 1 to node n . There $\{\text{node } 1, \dots, \text{node } n\}$ is the order of user selected. For each time path finding is $(U-1) * 2 * Q_p * P_r$, where U is the number of user-chosen point, Q_p is the time of the closest point and P_r is the time of A-star algorithm. The time of A-star is $L * B$, where L is the number of the level of goal node in spanning tree and B is the number of branch. Because the distribution of group is similar a net, A-star will select the best path for each time. There is faster than branch first search. The time of branch first search is L^B .

The reason of A-star working is the heuristic function in A-star algorithm. In this thesis, we use the distance of two points on the Earth to be the heuristic function of A-star. The heuristic function must has an importance property which is the estimation distance of a heuristic function does not larger than the real distance of two point. In Chapter 3, $h(n)$ estimates the distance of two points which is the shortest

distance between two points. If two points have a path, the cost of a path is larger than the direct distance between two points. If two points do not have a path; the cost of the path is infinity. The property of heuristic function is satisfied.

Table 4- 5 Compare algorithm of path finding

Name	The information of start node	The information of goal node
DFS	No	No
BFS	No	No
Greedy BFS	No	Yes
Dijkstra's algorithm	Yes	No
A* algorithm	Yes	Yes

Table 4-5 shows that A-star is a better algorithm in path finding. Because the algorithm consider the information of start node and goal node. Hence, A-star algorithm can do the best choice for each path spans out. Although the path finding algorithms create by different authors, there has the relationship between for each other in Table 4-6.

Table 4- 6 The relationship of path find algorithms

Name	$g(n)$	$h(n)$
BFS	0	0
greedy BFS	0	Define
Dijkstra's algorithm	Define	0
A* algorithm	Define	Define

Table 4-6 shows that A-star is a general algorithm of others. When $g(n) = 0$ and $h(n) = 0$, A-star is BFS. When $g(n) = 0$ and $h(n)$ defined, A-star is Greedy BFS. If $h(n) = 0$, then A-star is Dijkstra's algorithm. If the weight of edge = 1, A-star is greedy BFS.

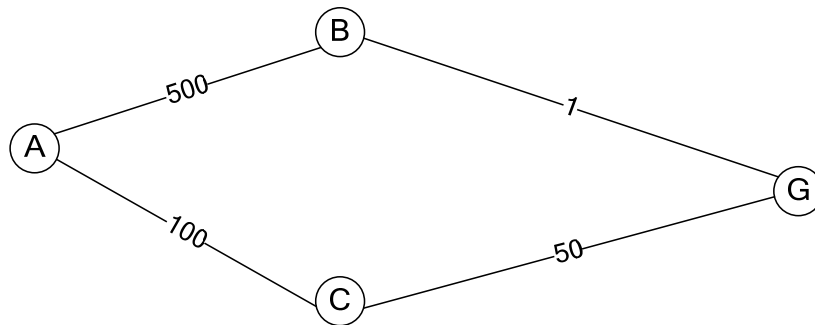


Figure 4-4 An example of greedy BFS

Figure 4-4 shows that the path of the greedy BFS is node A to node B to node G. the cost of the path is 501. The cost of the shortest path is 150. Hence the greedy BFS does not guarantee the path which finding by the greedy BFS is the shortest path. The path of the greedy BFS does not guarantee that the shortest path. The path of BFS also does not the shortest path.

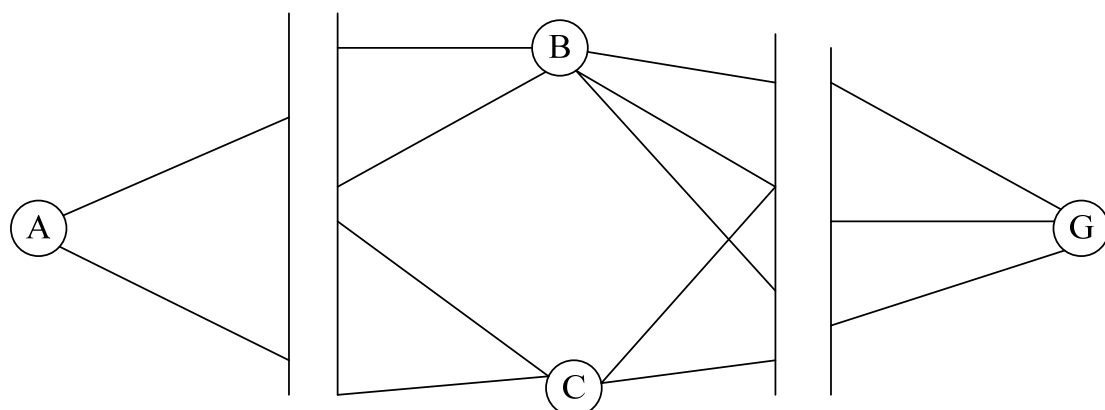


Figure 4- 5 An example of Dijkstra's and A-star

Figure 4-5 shows that if the cost of start node to node B and the cost of start node to node C equal. Dijkstra's algorithm selects node B or node C to span out.

A-star will estimate the heuristic function of node B and node C. A-star will select the minimal cost of node B and node C. Hence the size of spanning of A-star will be smaller than Dijkstra's algorithm. In the fixed grid, the cost of each node is one unit. The spanning tree of Dijkstra's algorithm is similar to the ripple of water and the spanning tree of A-star is similar to the flow from start node to the goal.

In this thesis, we present a method of point query and define the function of A-star. The feature of data provides the other view to design the query operation. The feature of data which groups indexes is suitable for the algorithm of A-star.



Chapter 5 Conclusions and Future work

In this thesis, we present an algorithm for the closest point query. First, a grouping algorithm groups the points of a street using corner detection. By grouping pictures of a street together, the efficiency of the point query algorithm is enhanced. Since a street map is a collection of polygons, the closest point must be in the top 3 closest groups. In addition, we use the A-star algorithm to find the shortest path between two user-chosen points. We define the g function, which is a grand total cost from start node to current node, and a heuristic function h , which is a straight line distance function.

For the closest point query, the time complexity is $\log m + H + 15$. The limitation of the closest point query is that the point data must be similar to the collection of line segments. The time to find a path is $(U-1) * 2 * Q_p * P_r$, where U is the number of user-chosen point, Q_p is the time to locate the closest point and P_r is the time of A-star algorithm. The time complexity of A-star is the level of spanning tree times the number of branches.

There are several functions that can enhance the efficiency of this system. Character recognition can be used to extract texts on the shop sign of street panorama. Based on the texts extracted, one can add on top of the shop signs hyperlinks and telephony URIs of the shops. Web mining can also be used to gather shop-related information from the Internet, and link the information to the shop signs of our system. We anticipate that more user interaction functions can be provided in this system.

References

- [1] Donald E. Knuth. The Art of Computer Programming. Volume 3. Second Edition
Reading, Massachusetts: Addison-Wesley, 1998.
- [2] Raphael Finkel and J.L. Bentley. Quad Trees: A Data Structure for Retrieval on
Composite Keys. Acta Informatica 4 (1), 1974.
- [3] Jon Louis Bentley, Multidimensional Divide and Conquer, Communications of the
ACM, 1980
- [4] E. W. Dijkstra: A note on two problems in connexion with graphs. In Numerische
Mathematik, 1, 1959
- [5] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001.
Section 24.3: Dijkstra's algorithm, pp.595–601
- [6] Amit J. Patel. Amit's Thoughts on Path-Finding and
A-Star. URL: <http://theory.stanford.edu/~amitp/GameProgramming/>, 2005
- [7] Andrew Moore, Efficient Memory-based Learning for Robot Control,; An
introductory tutorial on kd-trees, Computer Laboratory, University of
Cambridge, 1991.
- [8] Google map api, URL: <http://maps.google.com.tw>, 2008
- [9] Google Code, URL: <http://codes.google.com.tw>, 2008
- [10] Hanan Samet, Foundations of Multidimensional and Metric Data Structures,
Addison-Wesley, 1990.