Visibility Search for Pictures Using Multidimensional Data Structure

Yousuke Iwanaga, Yasuyoshi Kaneko, Shigeru Abe

Department of Electrical and Electronic Systems,
Faculty of Engineering, Saitama University
255 Shimo-Okubo Sakura-ku Saitama-shi 338-8570 JAPAN
{kaneko, abe}@ees.saitama-u.ac.jp

Abstract. Searching pictures from picture database in a geographic information system becomes important. A system which search pictures from map data (buildings and streets) and camera data (camera location and direction) by visibility search has been developed. If a user indicates a target point on a map, then pictures which include the object of the target point are searched and displayed in priority order. The k-d tree is used for the management of map data and camera data. Using photography range triangles and the range search, we have developed the visibility search algorithm.

1 Introduction

In recent years, we can take a large number of pictures using digital cameras and store them in a database, therefore it becomes important to search required pictures more quickly. Systems which search pictures with keywords are widely used. However, a system which search pictures from map data (buildings and streets) and camera data (camera location and direction) by visibility search is not appeared.

In our system, if a user indicates a target point on a map, then pictures which include the object of the target point are searched and displayed in priority order. It is easy to expand this system to an intelligent navigation system. The multidimensional data structure k-d tree is used for the management of map data and camera data. In addition to the nearest neighbor search and the range search, we have developed the visibility search algorithm. A prototype system shows the usefulness of the visibility search and good performance.

2 Data Management using the k-d tree

We use the k-d tree of multi- dimensional data structure. Two types of data structures, one for camera data and the other for map data, are used in the k-d trees. Fig. 1 illustrates coordinates and the data structures. Camera data are represented by triangles, that is, the vertexes indicate the camera locations and the vertical angles indicate the viewing angles. Map data such as buildings are represented by rectangles.

A k-d tree is shown in Fig. 2. The data in Fig. 2(a) are managed by the tree of Fig. 2(c). A bucket method is commonly used. In the bucket k-d tree, a leaf can contain more than one data. In Fig.2, the bucket capacity is 3. An internal node of the k-d tree can have only two children. In the k-d tree, area is split into sub areas recursively and each area corresponds to a node or a leaf.

Each node has a minimum enclosure rectangle that covers all regions of those of lower nodes. The sides of the enclosure rectangle are parallel to the axes. The enclosure rectangle plays an important role in handling data. Namely, a leaf has a minimum enclosure rectangle that covers all objects in the leaf such as camera data and map data. An internal node has an enclosure rectangle which covers their children's enclosure rectangles. In range search and nearest neighbors search, the enclosure rectangles are used to determine which node should visit to search the target object.

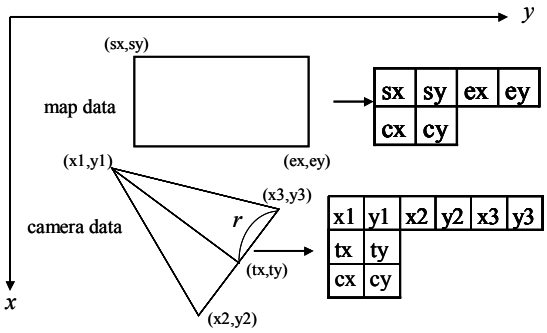
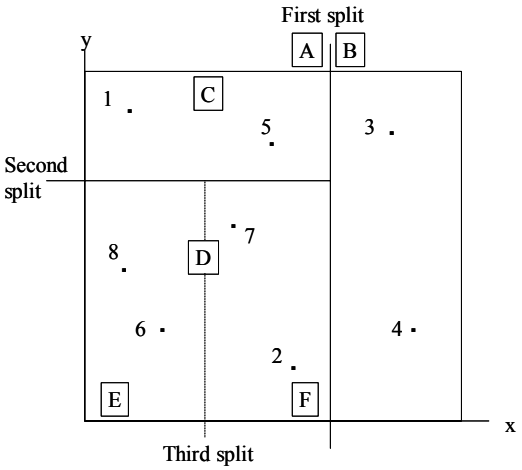
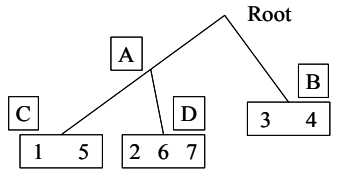


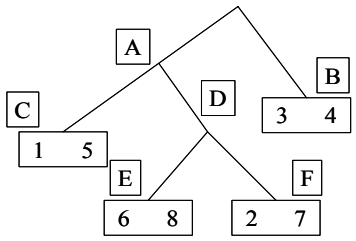
Fig.1 Data structures



(a) Data Location



(b) Before Insert Datum 8



(c) k-d tree of (a)

Fig.2 An example of a k-d tree

3 Visibility Search for Pictures

The picture data consist of image data and camera data. Image data are stored in the memory not in the k-d tree and the pointer to the image data is stored in the k-d tree. Many picture data are input to the visibility search system. Each picture's camera data that is a camera location, a camera direction and a viewing angle are input and the photography range triangle is stored in the k-d tree. The height of photography range triangle is the distance which we can

recognize the object in the picture and is determined properly in our prototype system. If a camera has an accurate GPS in future, it is possible to record the camera data when the picture is taken, and store them to the k-d tree automatically. Camera data and map data are stored in the two k-d trees respectively. The map data consists of building data and other obstacles of the visibility search.

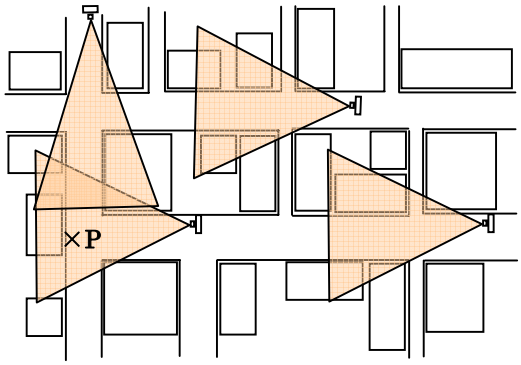


Fig. 3 Visibility Search for Pictures

We have developed two visibility search method. One is the point search and the other is the object search. In the case of the point search, if a user indicates the target point P(x, y) in the map on the screen as shown in Fig. 3, pictures which include P in their photography range are displayed on the CRT screen. In the object search, if a user indicates the target object in the map by clicking the inside of the object rectangle, pictures which include the object are displayed even if the whole object is not included.

Fig.4 shows the point search and object search. When the indicated point P is outside of buildings, the point search is executed, and when the indicated point Q is inside of a building, the building is recognized as the target object and the object search is executed.

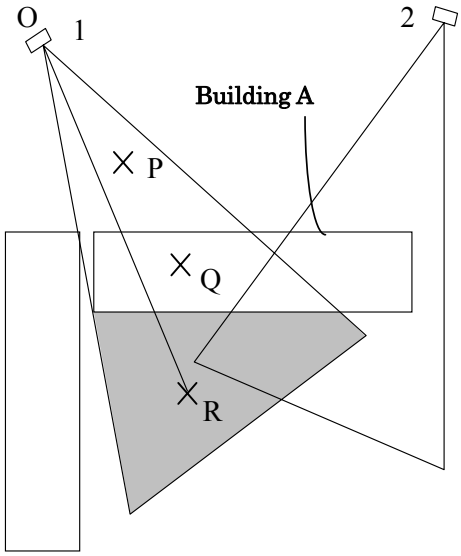


Fig. 4 Point Search and Object Search

3.1 Point Search

As shown in Fig.4, if the target point P is outside of buildings, the point search is executed and all the photography range triangles (later refer as triangle shortly) which include P inside are searched. As every triangle is managed by its minimum enclosure rectangle in the k-d tree, the rectangle which includes P is searched at first, then the triangle is examined whether P is inside of the triangle or not.

In the visibility search, we have to resolve the occlusion problem. In Fig. 4, the point R is inside of the camera triangle 1, however, the R is occluded by the building

A.To avoid the occlusion, we introduce the visibility check. A line segment between the camera location O and the target point R is drawn and the line segment OR is checked whether it intersects with other object or not.

3.2 Object Search

When the point Q is specified, the system searches all the pictures which include building A. In the object search, the target is rectangle A. Therefore, at first all the enclosure rectangles of the triangles which intersect with the target rectangle are searched. Then these triangles are checked precisely whether they intersect with the target rectangle A or not. For example, when the point Q is specified, not only the picture 1 but also picture 2 can be searched in Fig.4.

3.3 Priority to Display Pictures

The number of pictures is very large in our system. And many pictures may be searched and displayed at once. The priority to display pictures is important. In the case of point search, the picture which shows the target point (a) bigger and (b) in center has high priority. In the case of object search, the picture which includes whole the target building as large as possible has high priority. In navigation systems, the entrances of buildings are also important and we think the priority differs from applications.

3.4 Photography Range Adjustment

One method of deciding the height of the triangle is to exptend the bisector of camera viewing angle (vertical angle) until it reaches to a building (or any obstacles) and consider the length of the bisector as the height of the triangle. Also, the maximum length (visibility range) should be decided considering the size of objects in the picture.

3.5 Application to a Navigation System

A promising application of this system is a navigation system. If a present location and a target location are specified, the optimum route will be determined, and advisable pictures along the route can be informed to users in order.

4 Algorithms of Visibility Search

Algorithms of visibility search are described in Fig. 5.
A leaf node L is constructed with three elements:

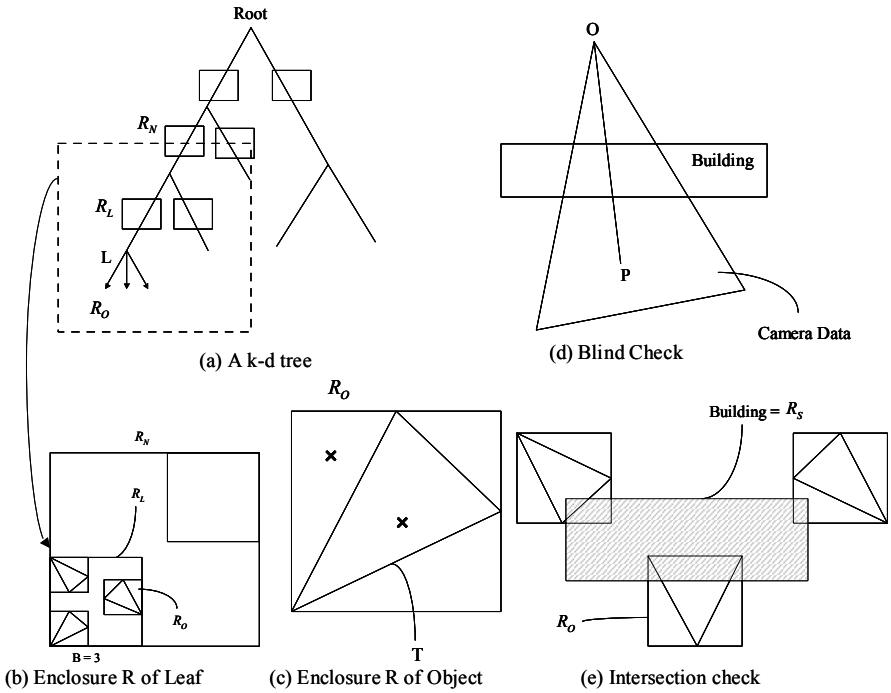


Fig.5 Tree and Enclosure Rectangles of Visibility Search

$$L = \{R_L, M, *Object[3]\}$$

where, R_L : an enclosure rectangle of leaf L,

M : number of object included in leaf L,

$*Object$: an array of pointers to object O.

Maximum number of M is the bucket capacity B.

An internal node N is a record with two elements:

$$N = \{R_N, *Child[2]\}$$

where, R_N : an enclosure rectangle of node N,

$*Child$: an array of pointers to children.

An object O is a record with three elements:

$$O = \{R_O, data(x_1, y_1).....(x_i, y_i), *image\}$$

where, R_O : an enclosure rectangle of object O.

$data$: camera data or map data.

$*image$: a pointer to the picture.

4.1 Algorithm VisibilitySearch

V1. A target point P (x , y) is determined.

V2. [PointSsearch or ObjectSearch] If the target point P exists inside of a building data, invoke ObjectSearch, else invole PointSesrch.

4.2 Algorithm PointSearch (N, P)

P1. From Root to Leaf visit Node and Leaf which R_N or R_L include the point P , and enumerate the leaves which R_L include the point P . (See Fig. 5(a))

P2. Examine each object O whether P is included in R_O . (Fig. 5(b))

P3. Examine further whether P exists in the triangle of the object O .(Fig. 5(c))

P3. Invoke BlindCheck.

4.3 Algorithm BlindCheck

B1. Draw a line segment OP between the camera location O and the target point P .

B2. If the line segment OP intersects with any objects O (usually bouildings), the camera data are deleted from the candidates to display, else the camera data are added to the candidates. (Fig. 5(d))

4.4 Algorithm ObjectSearch(N, R_S)

O1. Set Rs to the target building rectangle. Invoke RangeSearch(N, R_S) and the Objects which R_O intersect with Rs are obtained as the candidates.

O2. Examine each candidate whether its triangle (camera data) is intersect with the search rectangle Rs or not. (Fig. 5(e))

4.5 Algorithm RangeSearch(N, R)

R1. Set N as Root.

R2. If N is not a leaf, goto R4.

R3. (N is a leaf). Collect data in N which intersect with R . Return.

R4. For two children (i=1,2), if the child's R_N intersect with R , invoke RangeSearch($Child[i], R$).

5 Prototype System

We have developed a prototype system of visibility search, which can be used as a campus guide of Saitama University. Fig.6 shows an example of the execution result of our prototype system. Many rectangles are buildings of our campus. When a target

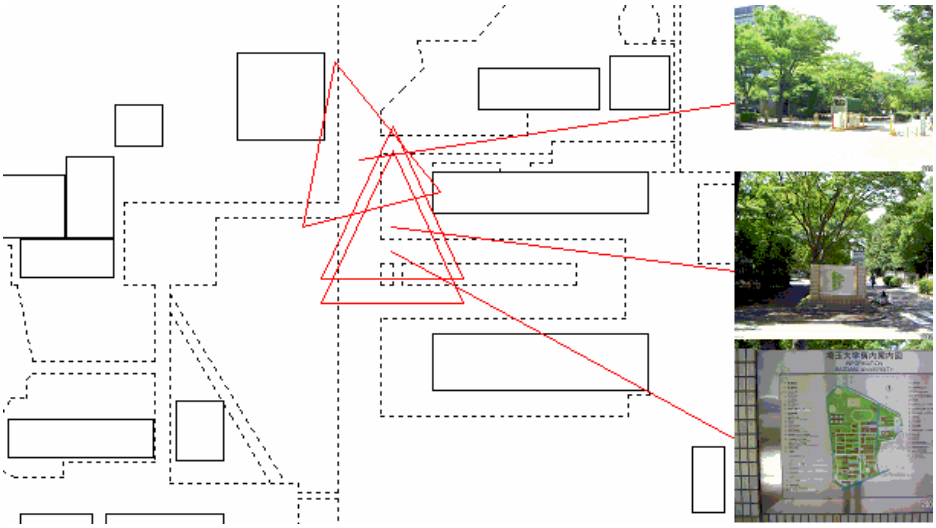


Fig. 6 Execution Result of Visibility Search

point is clicked with a mouse, pictures of camera data including the target point are displayed on the right side of the window and photography range triangles are also shown on the map.

6 Conclusion

We have developed a visibility search method for pictures using multi- dimensional data structure k-d tree. The k-d tree manages a large amount of picture data and its background map data hierarchically. Two search algorithms, the point search and the object search are presented. A prototype system shows the usefulness of the visibility search and good performance.

References

1. J. L. Bentley, "Multidimensional Binary Search Trees Used for Associative Searching," Communications of the ACM, Vol. 18, No. 9, pp. 509-517, 1975.
2. S. Abe, Y. Nakamura and K. Kamei, "Image data search method," Japan Patent 2981520, 1992.
3. Y. Nakamura, S. Abe, Y. Ohsawa, M. Sakauchi, "A Balanced Hierarchical Data Structure for Multidimensional Data with Highly Efficient Dynamic Characteristics", IEEE Trans. on Knowledge and Data Engineering, Vol.5, No.4, pp.682-693, 1993.