Automatic Modeling of a 3D City Map from Real-World Video

Hiroshi KAWASAKI(Student), Tomoyuki YATABE, Katsushi IKEUCHI, Masao SAKAUCHI

Institute of Industrial Science, University of Tokyo
7–22–1 Roppongi, Minato-ku, Tokyo, 106-8558, Japan
{h-kawa, yatabe, sakauchi}@sak.iis.u-tokyo.ac.jp, ki@iis.u-tokyo.ac.jp

Abstract

Mixed reality (MR) systems which integrate the virtual world and the real world have become a major topic in the research area of multimedia. As a practical application of these MR systems, we propose an efficient method for making a 3D map from real-world video data. The proposed method is an automatic organization method focusing on video objects to describe video data in an efficient way, i.e., by collating the real-world video data with map information using DP matching. To demonstrate the reliability of this method, we describe successful experiments that we performed using 3D information obtained from the real-world video data.

1 Introduction

With recent progress in technology, enabling the seamless integration of the virtual world and the real world, mixed reality systems have become an important and popular technology. In this mixed reality systems, real-world data and computational data must be related to each other ; much research has been done in this area. To integrate the real-world and the virtual world, photometric (image-based) or geometric matching is necessary. However, most research focuses on only one matching type, rather than addressing both. So we propose a new matching method between real-world video data and digital maps containing geometric data. To match the data, simple video data is not suitable. The amount of data is too large and the sequence of frames is almost identical; therefore, the data must be re-structured. We first explain the new data structure of the real-world video data for easy and efficient handling in our matching method. This data structure is specially designed for real-world video recorded by a camera which is mounted on a vehicle in a position that is perpendicular to the vehicle's direction of movement. To eliminate images of pedestrians, cars and the many other obstacles which usually exist at or near ground level, the camera is positioned at a slightly upward angle.

Our proposed matching method between the real-world and digital maps also contains a method for obtaining 3D information by using the optical flow which is acquired from video-data.

Finally, we show some applications of an example of this matching method and we also present acquired models of a 3D city map.

2 Structuring of real-world video data

Because video data is too large for reasonable handling on a computer and contains no typical structure to enable easy matching with a digital map, that data must be re-structured for the purpose of efficient management , along with spatiotemporal and spatial interpolation.

To satisfy both conditions, video object-based data structure [6] is preferable. In related research, video structure made from physical units such as "cut" or "scene" [8] is well known; however, we propose a more appropriate new data structure and structuring method based on video objects.

Extracting video objects from video data usually incurs some difficulties and is still a controversial matter among vision technologies. In this paper, we propose a specialized extraction method for real-world video which is specially recorded as mentioned above. This video object extracting method is based on EPI (epipolar plane image)[3]. Fig.1 is a usual EPI, but, in this situation, most buildings are the same distance from the vehicle; the EPI looks like Fig.2, and each belt is theoretically matched with one building. Furthermore, the EPI is based on the vehicle maintaining a constant velocity, but the vehicle cannot move at a fixed speed.

To solve the problem, we created the video object data by using the following process.

1. In each frame, use the vertical edge as boundary of the building
2. Assume the velocity of the vehicle by using the block matching method
3. Plot the edge data on the EPI and adjust the EPI plane according to the assumed velocity of the vehicle(Fig.3)

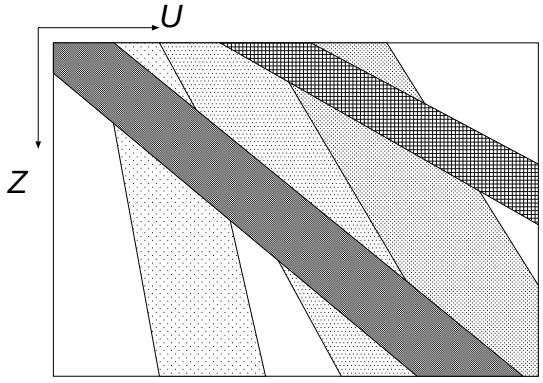

Figure 1: Tracking Image of singular point on EPI.

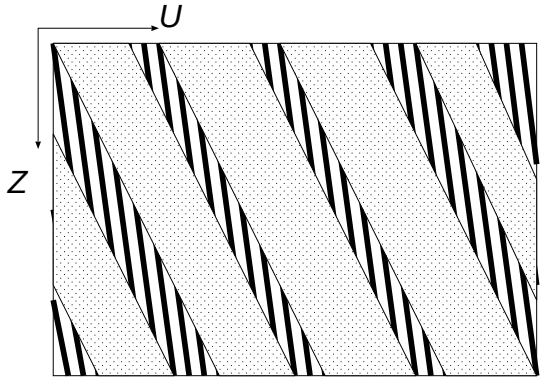

Figure 2: EPI made from the same depth of buildings.

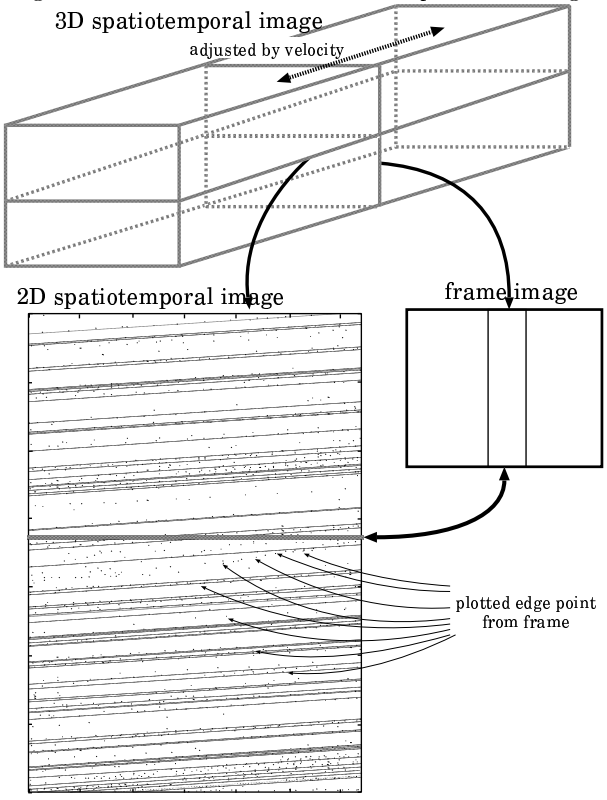

Figure 3: Spatio-temporal image consists of detected edge

4. Detect the approximate lines of the boundary edge on the EPI by using Hough transformation(Fig.3)
5. Remove the lines concentrated in the narrow space and leave only the ends of both sides
6. Restore the acquired lines in Step 4 on the EPI to each frame as a video object

Details of the process are explained in the following subsections.

2.1 Edge detection using perceptual organization

Before discussing edge detection techniques, we must discuss why we chose this technique to obtain the boundary of the building rather than other effective techniques like texture based segmentation. Our reason is mainly because real-world video footage of an urban city usually contains many linear edges which commonly coincide with the structure of buildings. In many cases, the detected edges exactly coincide with the boundaries of the buildings

The sequence of edge detection using perceptual organization [9]is as follows.(see Fig.4 as reference)

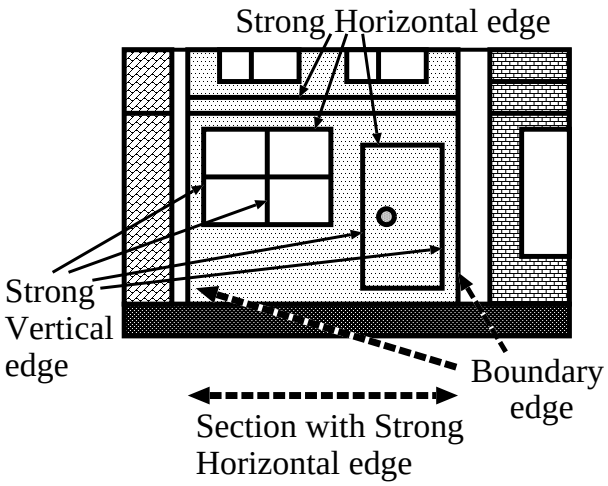

Figure 4: Structure of the building

1. Detect the vertical edge by using a Canny algorithm and if the edges are at an angle of $\frac{\pi}{16}$ or less on both sides, then group them onto one edge.
2. In the same way as Step 1, detect the horizontal edges.
3. The vertical edges detected by Step .1 have a number of errors, such as window frames, etc. To solve this problem, remove all the vertical edges in the sections where strong horizontal edges are detected.

2.2 Assumption of velocity

First, we get the motion vector by a simple block-matching method. However, because the simple block-matching method usually has a lot of noise, we apply the Gaussian filter shown on (1) to assume a reasonable velocity.

$$G[i] = (2\pi\sigma)^{\frac{1}{2}} \cdot \exp(-\frac{1}{2} \cdot \frac{(velocity[i] - ave)}{\sigma}) \qquad (1)$$

A sample of the retrieved data is shown in Fig.5. It is possible to see that the broken lines(retrieved by the simple block-matching method) have been improved to form a solid line(apply a Gaussian filter).

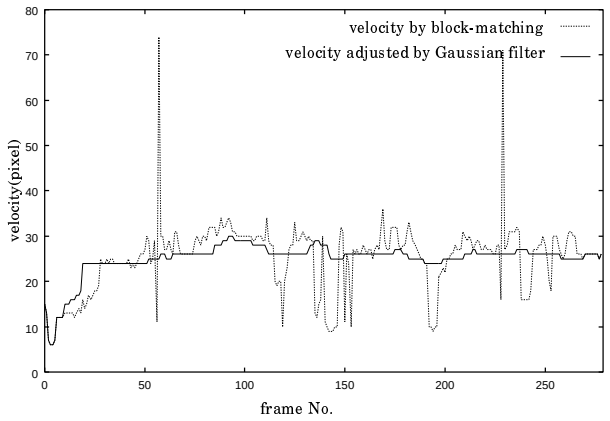

Figure 5: Result of assumed velocity

2.3 EPI of detected edge

The EPI made by plotting the edges in Sec.**2.1** with the adjustment of velocity in Sec.**2.2** is shown in Fig.6 (above). This figure shows that plotted edges form many lines with almost the same slope. These lines represent both the boundary of the buildings and noise created from obstacles like poles and trees. To make a video object, we must distinguish the boundary lines of the buildings from the other unnecessary lines.

In this paper we describe how we use the Hough transformation shown as (2) to get a straight line from the EPI.

$$\rho = x\cos\theta + y\sin\theta \tag{2}$$

Because the slope in the EPI is almost the same, the angle of the slope is restricted to a narrow range, and therefore the calculation cost is less. Then we remove the lines concentrated in the narrow space and leave only the ends of both sides. The results of these processes are shown in Fig.6 (below). Ultimately, we can obtain a building boundary known as the "panoramic boundary edge pattern". Fig.7 shows how the "panoramic boundary edge pattern" is obtained and the video object is made.

3 Matching between the real-world and digital maps

In the past, there have been some attempts to match the real-world and maps, for example, using an aerial photograph [10] or silhouette of a distant view of buildings [5] as real-world images. The former uses a stereo matching method with feature extraction, while the latter uses a DP matching method. Since they use different kind we uses of images for the real-world, their matching method is certainly different. We ourselves prefer to use the DP matching method for the following reasons:

- The re-structured data is suitable for pattern matching

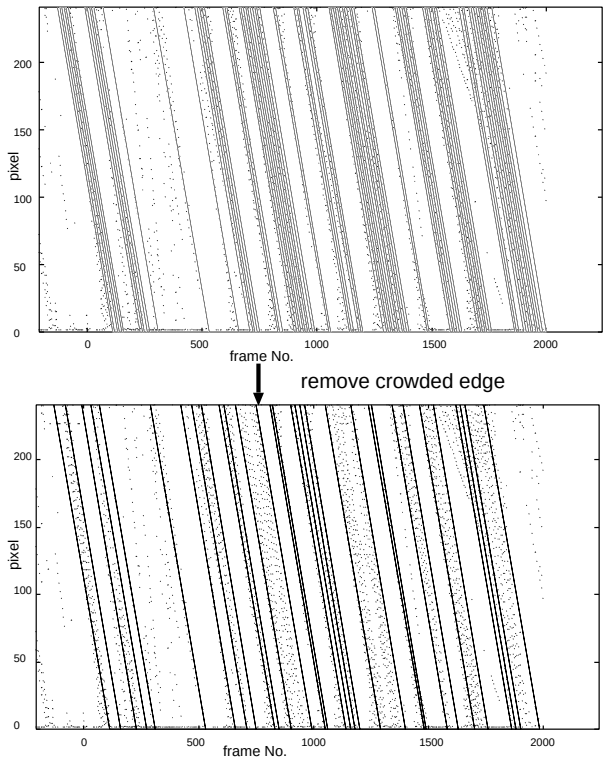

Figure 6: EPI made from edge data

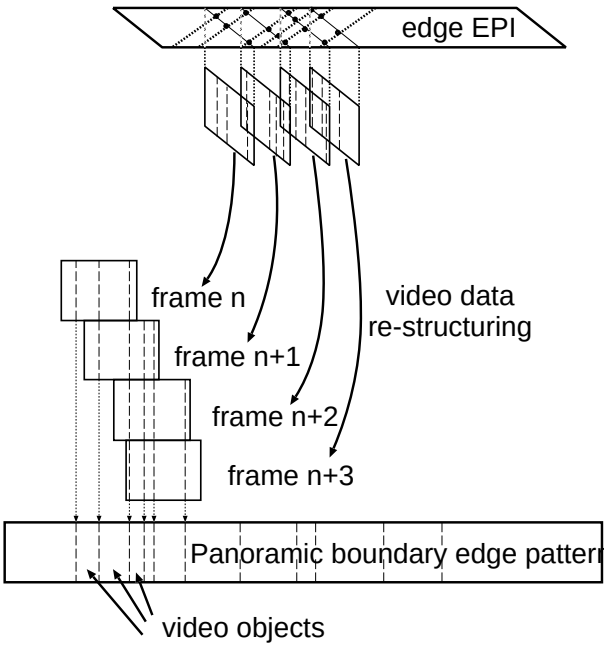

Figure 7: Structure of the "panoramic boundary edge pattern"

- This data structure is applicable only to the specialized video data mentioned above; this real-world video is easily captured; its good quality is stable.
- Making a pattern from a digital map for DP matching is easy

For matching, we use the "panoramic boundary edge pattern" acquired from the real-world video stated in Sec.**2.3** and the boundary pattern models obtained from the digital map. The latter method will be described in Sec.**3.2**. Scaling and positioning by DP matching allows us some errors in the modeling of the boundary pattern, but as the distance between the start and end point becomes greater, errors increase. Also, DP matching needs a pair of correspondent points at both ends of the sides, which usually contain some errors.

So far, we have implemented these matching methods using simple DP matching, but errors begin to be conspicuous when 6 or 7 buildings exist between the correspondent points and this method is not sufficient for practical use. In this paper, we describe how we use depth information in addition to a simple edge pattern and achieve high quality matching.

In this section, we first explain the reliable method of acquiring depth data from real-world video. Then, we make the boundary pattern models from the digital map(Sec.**3.2**). And in Sec.**3.3** we apply all the acquired information and data to DP matching. Finally we evaluate this method.

3.1 Acquiring depth from video

In the past, various research has been conducted regarding the acquisition of 3-dimensional (3D) information in a townscape. For example, a camera is fixed in the direction of movement and analyzing the cross section of the spatiotemporal image of the EPI [7] is suitable for the urban city in which most structure consists of planes. Moreover, analysis using the factorization method [12] with the restriction that urban city's mostly consist of planes, such as buildings, etc. is considerably effective.

On the other hand, in an actual city environment, it is difficult to acquire 3D information as right theory due to many obstacles, such as telegraph poles or trees (though we set the camera slightly upward to avoid taking pedestrians or cars) and buildings sometimes consist of complicated structures instead of plane surfaces. Actual buildings contain many complex texture and it often results in difficulties in extracting the feature points.

Overall, in this paper, we apply a technique called the "edge EPI" method together with the "panoramic boundary edge pattern" mentioned in Sec.**2.3** to acquire depth information for the matching.

To acquire depth information, we use the EPI plane made in Sec.**2.3**. On this EPI plane, boundary lines are already detected and the zone inserted into the adjacent boundary edges is considered one building or a gap between two buildings. So, acquisition of depth information is made by assuming the relationship of the adjacent zones by the motion vector analysis which represents the zone.

The actual process of acquiring depth is as follows(Fig.8).

1. All the frames are divided into vertical slits, and the motion vector of all those slits are assumed using a block-matching method.
2. Cluster the motion vectors of all the slits included in the same zone. Then select the maximum cluster in the zone and calculate the average of this maximum cluster and define this value as a representative value of this zone.
3. By using these representative values, assume the target zones depth information

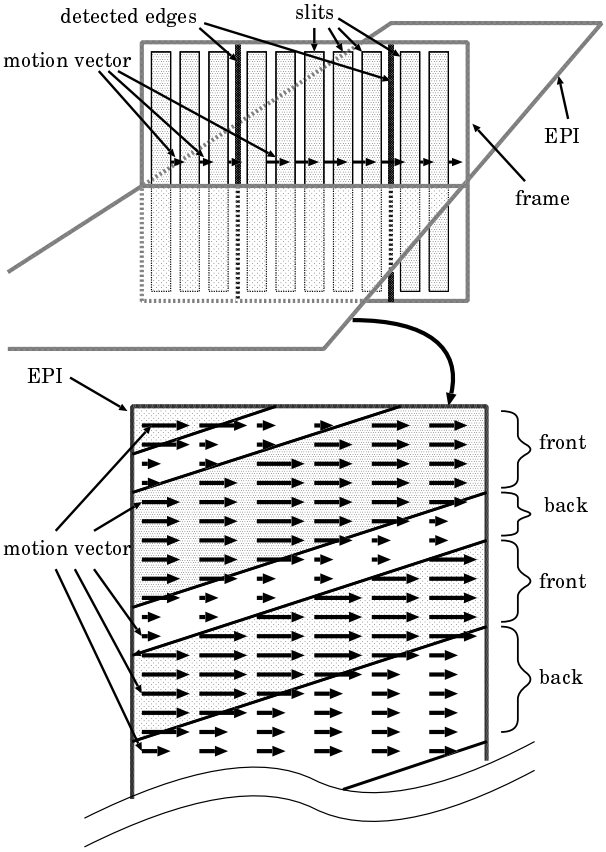

Figure 8: Acquire depth by motion vector

This method resembles a video game technique in which the front object moves faster and the background moves slowly. The depth information of the zone acquired by this technique is shown in Fig.9. The signs in this figure, all done manually, are the index of the building, and the capital letter J denotes the intersection.

Although this method still produces some noise, it can acquire depth information to a satisfactory degree. In particular, deep depths such as intersections can be detected with high stability by setting up a good threshold. So, using the depth information for matching below, mainly intersection data is used.

3.2 To make a pattern from a digital map

Since the target is a real-world video taken from a vehicle on the road, the model created from the digital map needs to correspond to it. By relating the geometric operation to the digital map, the pattern of the boundary of the building seen

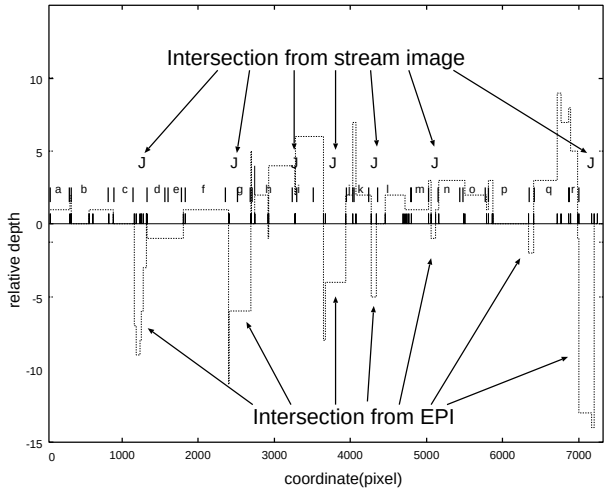

Figure 9: Estimated depth.

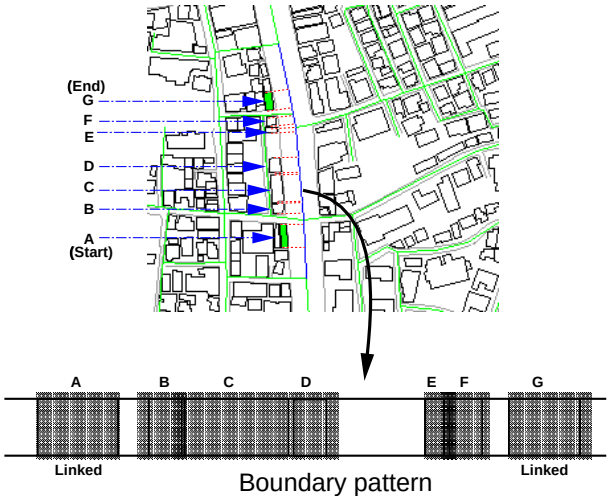

Figure 10: Automatic configuration of pattern.

from the road can be obtained. The procedure for making a boundary pattern of the buildings along the road is as follows.

1. Describe at least two buildings which face the route.
2. Determine the path which ties the two described buildings in Step 1.
3. Create a model of the boundary patterns of the building which exists between the described buildings.

In Step 1, both descriptions done manually or automatically by GPS are supposed in our system. Currently, only manual way is supported, but in the near future automatic method will be supported. And manual description can be done over network to lighten the work load of the user. Details are explained in Sec.**4.1**.

In Step 2, since the described information in Step 1 is only the location of the two buildings, the system must assume the route along which the vehicle travels. This assumption is done automatically as follows: Since the road data provided by the digital map is as short as $20 \sim 30m$, the system first searches the two roads touched by each building. Then, the shortest path to connect the two acquired roads is obtained.

In Step 3, the boundary pattern of the buildings is created by orthogonal projection from the route to the buildings (consisting of polygonal data) in the digital map.

Fig.10 shows the automatic modeling of the boundary pattern of the buildings obtained from the digital map using the two points described on the map.

3.3 DP matching

When the two buildings are described in real-world video, those two buildings correspond to the buildings in the digital map. With this described correspondence point, we can conduct DP matching between video data and the digital map.

In an actual situation, to get the corresponding point, we are going to use GPS ; we have already acquired a specially equipped car with video cameras, GPS and other sensors.

The results of matching with depth information are shown in Figs.11 and 12.

The matching results in the case of no depth information are shown in Fig.13.

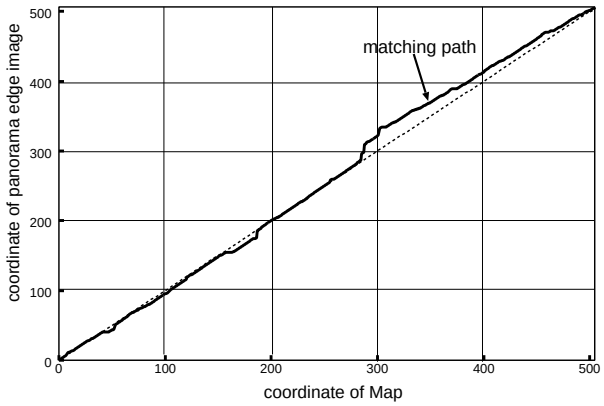

Figure 11: Path of DP matching.

3.4 Evaluation

To compare Fig. 12 with 13, we made two tables from the matching results. Table. 1 shows the individual error values of corresponding edges. Then we summarized the matching results to Table. 2 by calculating an average of all squared errors. Table. 1 and 2 indicates that the accuracy of matching is greatly improved by using the depth information obtained by the "edge EPI".

It is a well known fact that DP matching is greatly dependent on the matching weight, but in this implementation, several experiments show that weight does not affect the matching to any great extent. This can be attributed to the fact that we can obtain intersection from data of good quality.

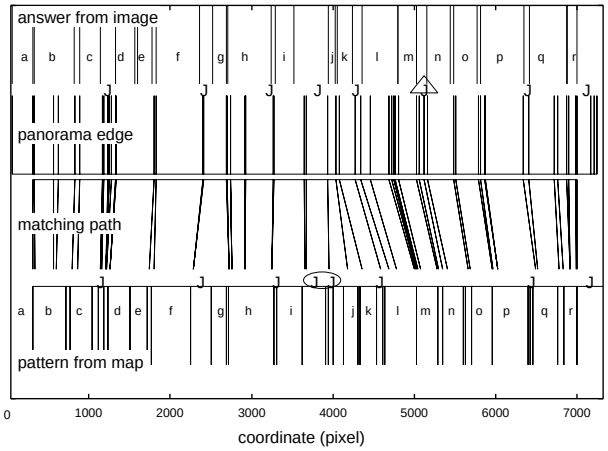

Figure 12: Result of DP matching.

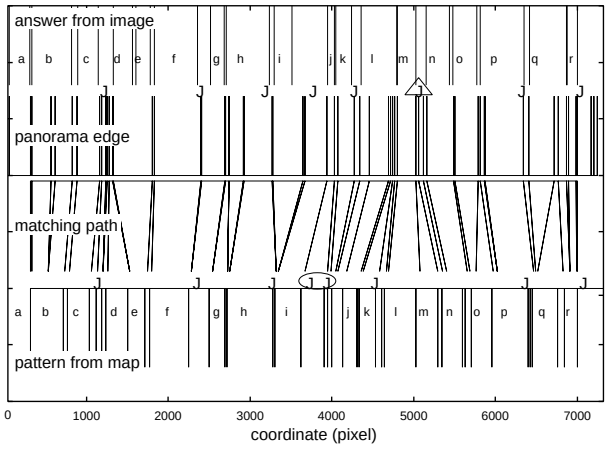

Figure 13: Result of DP matching without depth information.

Also, effectiveness of edge detection is an important factor for this method. We tried detecting the edge with various thresholds and ultimately obtained robust and reliable value irrespective of video data and maps.

In addition, if we were to be unable to detect all vertical edges by reason of obstacles or complicated building architectures, DP matching would very possibly solve these problems. See index **d**, **e** and **g** in Fig. 12. Although we were unable to detect these boundary edges, DP matching obtained the correct answers.

We also tried the same experiment in several streaming videos, and obtained quite similar satisfactory results..

4 Implementation of prototype systems

4.1 Real-time Query and Answer system

Fig.14,15,16 shows an example image of the system. This system has mainly two functions: one is to answer users' queries, the other to allow any users to describe the index to the building. Both functions can be done over the network,

Table 1: Errors of DPmatching

with depth info.			without depth info.		
edge No.	coord.	diff.	edge No.	coord.	diff.
1	100	0	1	100	0
2	111	0	2	111	0
3	522	44	3	522	0
4	578	56	4	578	78
5	800	67	5	800	0
6	944	0	6	944	222
7	1333	0	7	1333	0
8	1355	0	8	1355	0
9	1822	0	9	1822	0
10	2066	0	10	2066	-156
11	2111	0	11	2111	-22
12	2544	-56	12	2544	0
13	2544	-78	13	2544	-422
14	3100	-189	14	3100	-333
15	3177	-144	15	3177	-322
16	3211	-22	16	3211	-444
17	3366	0	17	3366	-511
18	3433	0	18	3433	-322
19	3811	0	19	3811	-222
20	4000	-56	20	4000	0
21	4100	0	21	4100	0
22	4366	0	22	4366	0
23	4388	0	23	4388	0
24	4622	156	24	4622	0
25	4688	0	25	4688	0
26	5077	22	26	5077	0
27	5133	0	27	5133	-22
28	5522	0	28	5522	67
29	5555	0	29	5555	0

(pixel)

Table 2: Summary of errors

depth information	average of error(square)
with	3567
without	37604

(pixel^2)

in particular the Internet, and can be processed in real-time. To realize this function, the systems are all written in Java language and work on any browser which can run Java with JMF [1].

We will show some typical usage of this system with short explanation.

- Indicate a building in the streaming video and make a description of it.(Fig.14)

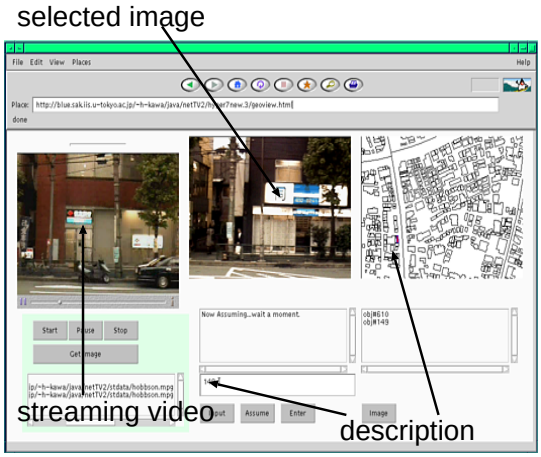

Figure 14: A snapshot of description

- Indicate a building in the streaming video; users can retrieve information of that building, i.e., name and location in the digital map.(Fig.15)

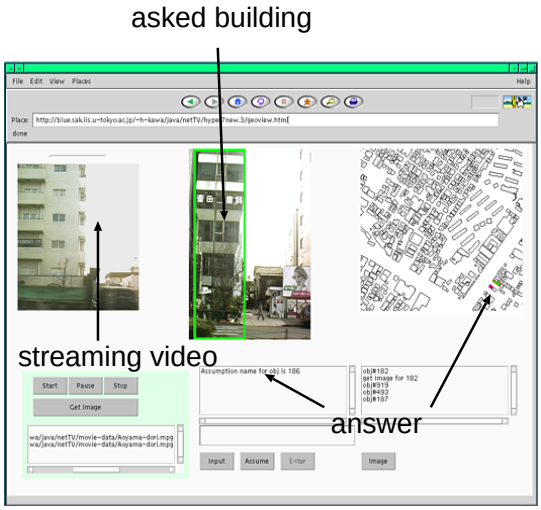

Figure 15: A snapshot of question and answer

- Indicate a building in the map; users can get an image of the building from the video data.(Fig.16)

4.2 Retrieval of individual building images

This system creates a texture database of buildings by cutting and dividing a texture image from the video data for each individual building.

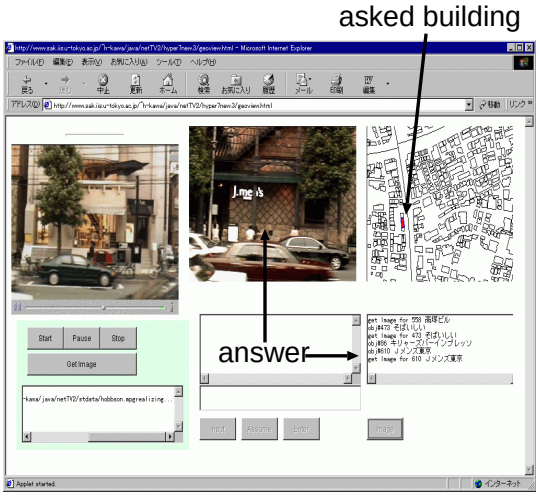

Figure 16: A snapshot of question and answer

All divisions are done automatically by projecting the corresponding edge pattern obtained by the DP matching to the panoramic image(Fig.17) made using a mosaicing method [11].

Figure 17: Panoramic image by video mosaicing.

An example of the retrieved texture data base is shown in Fig.18.

4.3 An automatic construction of 3D virtual map

Using the results of the two previous systems, we made a system which can generate a 3D virtual map automatically with a VRML form. This system works in two phases as follows:

1. make a geometric model using the digital map
2. put the texture data (retrieved in Sec.**4.2**) onto a geometric model

In both processes, sufficient accuracy of the matching between video data and the digital map is needed for practical use. A 3D map made by this system is shown in Fig.19.

5 Conclusions

We proposed an automatic 3D modeling method of a city map from real-world video using a digital map. To achieve this goal, we proposed and implemented an efficient structuring method of video data and a robust matching method

Figure 18: Texture database of buildings.

Figure 19: Virtual 3D map of VRML.

between the video data and the digital map data. The former method is realized by adopting the video objects for data structure and the latter uses depth information to raise the robustness.

To implement our method, we divided the proposed system into three sub systems. The first system is the query system used to make important and basic matching between real-world video and the digital map(Sec.**4.1**). The second is an automatic retrieval system which retrieves individual texture of the building from video data(Sec.**4.2**). The third makes a 3D map by integrating both the first and second systems' results into VRML(Sec.**4.3**).

In the future, we need to decrease errors caused by the many obstacles existent in the city, such as telegraph poles and trees. We are now trying to acquire more precise depth data and boundary edges of buildings from video. Moreover, examination of the further effective use of the 3D model still remains an important theme.

References

[1] Java(tm) media framework api home page. `http://java.sun.com/products/java-media/jmf/index.html`, Dec. 1998.

[2] Arisawa, H., and Tomii, T. Design of multimedia databese and a query language. In *Proceedings of MULTIMEDIA '96 IEEE* (Jun. 1996), pp. 462–467.

[3] Bolles, R., Baker, H., and Marimont, D. Epipolar plane image analysis:an approach to determining structure from motion. *Int.J.of Computer Vision 1* (1987), 7–55.

[4] Hampapur, A., Jain, R., and Weymouth, T. Digital video segmentation. In *Proceedings of Second Annual ACM Multimedia Conference* (Oct. 1994), pp. 357–364.

[5] Liu, P., Wu, W., Ikeuchi, K., and Sakauchi, M. Recognition of urban scene using silhouette of buildings and city map database. *Proc. the 3rd ACCV 2* (Jan. 1998), 209–216.

[6] Manske, K., Muhlhauser, M., and Vogl, S. Obvi:hierarchical 3d video-browsing. In *Proceedings of Second Annual ACM Multimedia Conference* (1998), pp. 369–374.

[7] Notomi, M., Ozawa, S., and Zen, H. Modeling of urban scene by motion analysis. *IEICE Trans. Information and Systems J81-D-II* (May 1998), 872–879.

[8] Oomoto, E., and Tanaka, K. OVID: Design and Implementation of a Video-Object Database System. *IEEE Transactions on Knowledge and Data Engineering 5*, 4 (Aug. 1993), 629–643.

[9] Sarkar, S., and Boyer, K. L. A computational structure for preattentive perceptual organization: Graphical enumeration and voting methods. *IEEE Trans.Systems, Man, and Cybernetics 24*, 2 (Feb. 1994), 246–267.

[10] Shi, Z. C., and Shibasaki, R. Automated building extraction from digital stereo imagery. *Automatic Extraction of Man-Made Objects from Aerial And Space Images* (May 1997), 119–128.

[11] Szeliski, R. Video mosaics for virtual envrionment. *IEEE Comput.* (1996), 22–30.

[12] Thomasi, C., and Kanade, T. Shape and motion from image stream under orthography: A factorization method. *Int.J.of Computer Vision 9* (1992), 137–189.