PROPOSAL OF AUTOMATIC TEXTURE MAPPING ONTO LARGE-SCALE 3D CITY MAP BY THI
時系列高さ画像を用いた大規模三次元住宅地図への自動テクスチャマッピング手法の提案

by

Jinge Wang
王　金戈

A Master Thesis
修士論文

Submitted to
the Graduate School of the University of Tokyo
on February 4, 2009
in Partial Fulfillment of the Requirements
for the Degree of Master of Information Science and Technology

Thesis Supervisor: Katsushi Ikeuchi　池内　克史
Professor of Computer Science

ABSTRACT

3D city maps, especially the color maps have become widely applied in recent year. However, most of shrinkwrapped 3D city maps are made by CG, which need huge labor hours and lack reality. Here, to reduce labor hours and improve the reality of building textures, we propose a automatic texture mapping method onto existing large-scale 3D building models with building textures acquired by on-vehicle camera. To achieve this goal, there are two main issues:

1. Recognize buildings from real image stably.

2. Determine corresponding relation between buildings recognized from images and building models.

In the first issue, we present a new expression of spatio-temporal volume, called "Temporal Height Image (THI)", and this can be considered as a advanced version of Epipolar Plane Image (EPI). Instead of constructing EPI by color transition, THI is constructed by height transition of buildings, thus it can ignore the effect of color noises inside buildings (i.e. windows), and recognize buildings stably.

Moreover, in the process of THI construction, we have to extract building roofs by edge extraction, whereas, electric wires interrupt. Thus, we propose a wire-delete filter based on median filter.

In the second issue, we design a Dynamic Programming matching, which utilizes the feature of THI and takes account of possible problematic cases in urban scene.

Then after approaching building facades as rectangles, texture mapping becomes the issue that to determine the four corresponding corners of rectangle. It can be well done by THI. We examine our method in the area around school and in Shinjuku.

論文要旨

近年，三次元住宅地図，特に色つきの三次元住宅地図は広く普及してきた．だが，殆どの市販三次元住宅地図はＣＧによって作られ，膨大な手間がかかるのみならず，リアリティに欠ける欠点がある．ここで，人の手間を減らし，リアリティを向上されるために，実画像を用いた大規模な三次元都市モデルへの自動テクスチャマッピング手法を提案する．これを実現するために，主に二つの課題を解決しなければならない：

１．頑健な実画像から建物の認識法

２．実画像上の建物と建物モデル間の対応付け

課題１を解決するために，新しい時空間画像の表現法として時系列高さ画像（ＴＨＩ）を提案する．これは従来のエピポーラ平面画像（ＥＰＩ）の改良版だと認識でき，ＥＰＩの欠点である内部の色のイズを除去することができる．よって，安定に建物を認識できる．

また，ＴＨＩを構築する際に，建物の屋根を認識する必要があるが，市街地でよく見かける電線は邪魔となる．よって，従来のメディアンフィルタをベースとして，新しい電線除去フィルタを提案する．

課題２において，ＤＰマッチングを導入した．特に，ＴＨＩの特性である建物高さの変化を用いた類似度判断関数を設計し，実画像上の建物と建物モデル間の対応付けを行う．

また，テクスチャマッピングを簡易化するために，建物の側面を長方形に近似した．よって，長方形の四つの対応コーナーを決めれば，テクスチャマッピングが可能となる．我々は学校付近のエリアと新宿エリアにおいて本手法の有効性を検証した．

Acknowledgements

First of all, I would like to express my sincere gratitude to Pro. Katsushi Ikeuchi. He gave me a lot of constructive opinions during meetings. His wealth of knowledge, precise mind, hard work and ... smile left me a very deep impression.

I wish to express my deepest gratitude to Project Research Associate Shintaro Ono. He is my direct senior associate. He gave me significant advices to my research, and corrected my poor Japanese carefully. He is a serious man, very different from me. I really want to learn his seriousness.

I also wish to thank all the members in Ikeuchi Laboratory. Without them, I could not lead a happy research life. It is a little part in my life, but it is special, it is the end of my 20-year study journey. Thanks a lot for helping me complete my thesis. Thanks a lot for make me finish my study journey with no regrets.

At last, this research is a coorperative project with many other companies and laboratories. This project gave me a chance to meet a great number of people from different backgrounds, that help me fit in with Japanese society quickly. Thanks all the members in this project a lot for your encouraging words.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In recent years, the rapid spread of in-car navigation systems, PCs and internet has lead to the use of 2D digital maps by a great many people. We anticipate the needs for maps that are closer to reality will grow as applications increase in the future. However, with there being a limit to the extent that those needs can be addressed by existing 2D maps, a supply of 3D digital maps that are intuitive and easy to follow are in demand.

3D digital maps, especially textured ones, will play a very important role. Some companies, for example Google[1], ZENLIN[2], GEO Technical Lab.[3] (Fig. 1.1(a)), Forum8[4] (Fig. 1.1(b)) and ASAHI[5], have made their debuts of textured 3D digital map products. Whereas, all those textures of building are made by CG and the process of texture mapping needs huge labor hours, moreover lacks of actuality.

Here, to improve the actuality of textures and reduce labor hours, an automatic texture mapping method onto existing colorness 3D building models by real building images is required. Especially, supposing the future use of in-car navigation system, 2D building images captured by on-vehicle cameras (ground-based images) are preferred because of high-resolution textures of building facades.

In the field of automatic texture mapping, image-based modeling technique[6,

7, 8, 9] is famous to construct 3D shape models from multiple 2D images of physical objects, and obtain textured models automatically. Therefore, this technique can be expected to achieve our goal. However, highly accurate camera calibration and a large amount of 2D images from multiple view angles has limited this method to be suit for only in-door small-scale objects.

Of course, Automated or semi-automated texture mapping onto 3D city models were well studied.

Frueh at al. [10] presented a automated texture mapping of 3D city models with oblique aerial images. They matched images and models using a optimisation process with line segments (building contours). However, this way needed inital manual calibration and a large amount of corresponding line segments. We have tried applying this method to match ground-based 2D images and building models, whereas, it did not work well unless accurate inital calibration, because on a single image we could only recognize much less necessary line segments than from aerial large-scale image. It means that this method is better adapted to large-scale, aerial images, but not ground-based images.

Lee at al. [12] tried automated texture mapping onto building walls by ground-based images captured from multi-views. They estimated the rotational and translational parameters of the ground level cameras by three 3D and 2D line segment correspondences. However, the length of the 2D line segment could be wrong due to incorrect edge detection or self-occlusions of buildings. This caused errors in calibration of ground view cameras. Furthermore, there was an assumption that they should know which building was being reflected to determine the corresponding 2D-3D line segments. It means that this method is applied only to a single building, but not a large-scale area.

So here, we present a robust and automated texture mapping onto large-scale existing 3D building models with ground-based 2D building image sequence captured by on-vehicle camera. Moreover, like Frueh‘s method[10, 11], we approach building facade as a rectangle, so that texture mapping will become easy while obtaining the information of corresponding corners of rectangle between images and

models.

There are two main issues to be addressed:

1. Recognize buildings from 2D ground-based images stably.

In our method, first we have to extract necessary textures of building facades from ground-base images. This work can be done by recognizing the field of buildings from images. Instead of using a single image, we generally use image sequence to recognize buildings which could reduce noises made from single image. For example, Epipolar Plane Image (EPI)[13] is a popular method for recognizing buildings[14] from image sequence and 3D-reconstruction[13]. After acquiring images by on-vehicle camera moving on a straight line with constant speed (Fig. 2.2(a)), we can obtain a Spatio-Temproal Volume (Fig. 2.2(b)). cutting off this volume, we call this cross section EPI. As shown in Fig. 2.2(b), the object on normal image can be expressed as a stright band on EPI, which becomes easy to be recognized by contour detection. Since the edges inside building (i.e. windows) may heavily affect the result, contours can not be recognzied correctly. Kawasaki at al.[14] addressed this problem by removing narrow edges and crossed edges, though it needed much processing time and was not stable.

It is considered that EPI has lost most of the information about the Spatio-Temporal Volume while being cut off, especially the dominating information – shapes of buildings. Because in normal urban scenes, shapes of buildings are much simpler than textures of buildings, using shapes to recognize buildings is easier than using textures. Here, to utilize the shapes of buildings, we propose a novel expression of Spatio-Temporal Volume, called "Temporal Height Image(THI)". Assigning a gray value in proportion to height to all the objects in the Spatio-Temporal Volume, we can obtain THI by looking at the volume from above. THI is considered to overcome shortages of EPI and recognize buildings as bands stably. Of course, after recognition of buildings, corners can be recognized easily.

2. Design a method to determine corresponding relation of buildings between

images and models.

In the Lee's research[12], it was assumed that while the pair of building on image and building model had been assigned, then corresponding line segments could be determined automatically. However, in a large-scale area, we need a efficient method to determine building pairs between images and models. Note that, buildings on images are converted to bands on THI by the same process, building models can be also converted to bands, by moving a virtual camera in 3D map. Matching these two kinds of bands, we can then obtain the corresponding relation.

Moreover, although THI can recognize buildings stabler than EPI, there are still some noises made from, for example, trees, electric lights and empty spaces between two approximate buildings. On the other hand, there is no noise in the THI made from building models. To match these two kinds of bands with or without noises, Dynamic Programming matching (DP matching)[15] is well used. For example, DP matching was used to match image sequence with a 2D digital map[14], to match two series of image sequences captured in urban area on different dates[21], and to match 3D building models acquired by range senser with 3D map[22, 23].

Here, we design a DP matching process, which exploits the feature of our proposed THI. For example, add a constraint of height transition between two approximate buildings and use aspect ratio of buildings which can be obtained from THI.

Through previous process, we can get building pairs between images and models. Then approaching the building facade as a rectangle, the four corners can be recognized easily from THI made from image sequence. Since the corresponding corners from models have been assigned, texture mapping can be well done.

In addition, we also need to consider the problem caused by electric wires. Since THI is created by the shape of buildings which can be decided by extracting highest edges, the existence of electric wires would cause cubersome. Thus we suggest a efficient wire-delete filter based on median filter to remove them.

(a) WalkeyeMap by GEO technical laboratory.

(b) UC-win/Road by Forum8.

Figure 1.1: Examples of 3D color maps.

Moreover, as a obiter application, using the result of corresponding buildings between images and models, we also estimate the vehicle location where the images were captured. It can be achieved by finding out corresponding image captured by virtual camera in 3D map, threfore the location of virtual camera can be looked upon as the real vehicle location.

1.2 Overview of Our Texture Mapping Process

According to the two main issues, as shown in Fig. 1.2, there are three steps to achieve texture mapping:

1. Construct THI to recognize buildings from real image sequence and 3D map.

There are two parts in this step:

(1). Recognize buildings by THI made from real image sequence.

(2). Recognize buildings by THI made from 3D map.

In the first part, after aquiring image sequence by a on-vehicle camera that move with constant velocity stright, we can Construct a THI. Here, buildings were expressed as bands, like the feature of EPI. Thus, it becomes easy to recognize buildings as bands by straight edges.

In the second part, we make a virtual camera move in the 3D map to acquire image sequence and Construct THI. Especially, to obtain the similar THI as the one from real image sequence, we have to design virtual camera parameters(orientation and direction) as near to real on-vehicle camera as possible.

2. Match bands recognzied from Image-THI (THI from image sequence) and Model-THI (THI from 3D map) by Dynamic Programming.

Generally, it is considered that bands recognzied from Model-THI mean buildings as there are only buildings on the map. On the other hand, it is not sure that bands recognized from Image-THI mean buildings as there are also some noises, for example electric lights, trees and empty spaces between two approximate buildings, which may be expressed as bands. To modify these noises, we utilize Dynamic Programming matching (DP matching) to find out appropriate building pairs between real image and 3D map. Of course, the information of corresponding vertical contours can be obtained at the same time.

3. Texture mapping onto building models by corresponding building corners.

Assuming building facade as a rectangle, it becomes easy to texture mapping, after determine the four corner pairs of rectangle between buildings on image and building models.

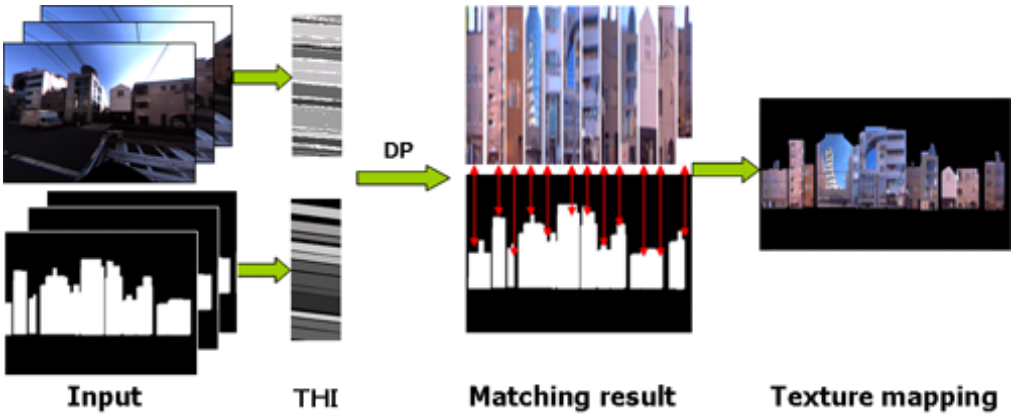


Figure 1.2: Overview of our proposed texture mapping process.

1.3 Thesis Outline

This paper is organized as follows:

Chapter 2 explains the proposal of Temporal Height Image (THI), and compares it with Epipolar Plane Image (EPI), at last, the process of THI formulation and bands recognition will be introduced. Chapter 3 discusses the way of matching recognized bands from Image-THI (THI made from image sequence)and Model-THI (THI made from building models)by DP matching, especially the design of DP matching.

Chapter 4 introduces the texture mapping method in detail, including recognition of building corners from real images by Image-THI.

At last, in chapter 5, we will show a obiter experiment of vehicle location estimation by image processing.

Chapter 2

Proposal of Temporal Height Image

To recognize objects from 2D images stably, instead of using a single image, we usually use image sequence (video), in which the object is reflected continuously. In this field, Epipolar Plane Image (EPI) is a famous method to express spatio-temporal volume and recognize object. so we will introduce the idea of EPI first. Then explain our proposal – Temporal Height Image (THI) and compare the features of EPI and THI. At last, I will illustrate how to construct THI from image sequence and 3D map.

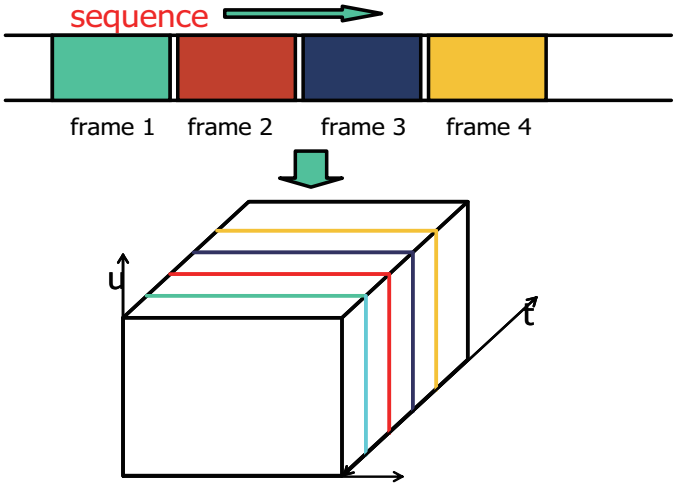


Figure 2.1: Image of spatio-temporal volume

2.1 Epipolar Plane Image (EPI)

Image sequence is a collection of images taken at certain sampling interval. A box that consists of these accumulating in time is a "spatio-temporal volume"(Fig. 2.1)[16, 17, 18, 19]. When the sampling interval is enough dense or when the motion of the camera or the photogenic objets is slow, the spatio-temporal volume forms images with strong correlations on the cross-sections. The motion of the camera or the photogenic objects is detected by analyzing the cross-sections.

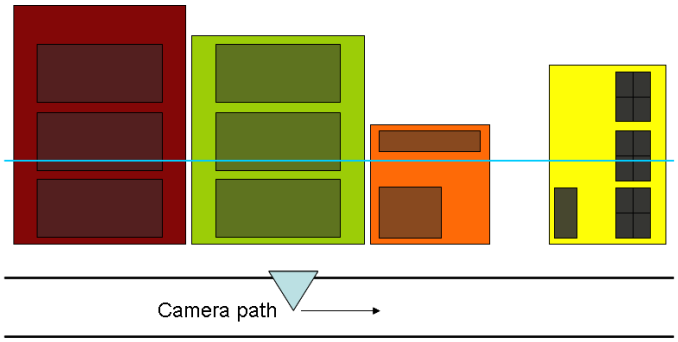
In this appendix, we suppose the situation where the camera moves in a uniform straight line to the direction parallel,to the optical axis taking stationary objects(Fig. 2.2(a)).

Let us consider the horizontal cross-sections of a spatio-temporal volume. This type of image is called an EPI (Epipolar Plane Image)[13](Fig. 2.2(b)). In an EPI, we can observe an interest point in space as a continuous trajectory. In our situation, a camera in a uniform straightly line motion, the trajectory of a stationary point in space forms becomes a line. In addition, a moving camera that is interrupted forms a stereopsis configuration with time differenc. Therefore, the following relation exists between the depth of the 3D point and the slope of the trajectory in EPI:

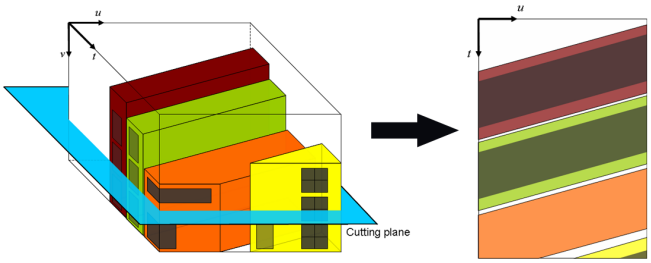
$$\frac{\triangle u}{\triangle t} = f\frac{V}{Z} \tag{2.1}$$

Here, f is the confocal lengh of camear, V is the velocity of the camera and Z is the depth of the interest 3D point. From the above equation, it is easily found that the further the 3D point, the steeper the slope becomes, and that the nearer the point, the gentler the slope becomes in the constant velocity. By this property, 3D reconstruction can be proposed[13].

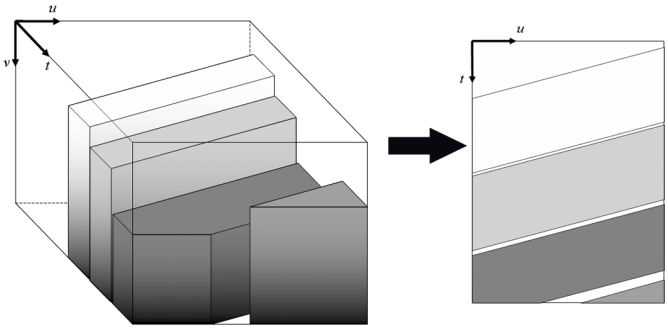
Kawasaki at al. aligned buildings on images to 3D city map by EPI[14], where they should recognize buildings as bands from EPI by edges. However, since the edges inside buildings (i.e. edges of windows), they could not recognize bands correctly, or they should first determine whether the edges extracted were inside or on the contours of buildings.



(a) Process of capturing pictures.



(b) Formulation of EPI.



(c) Formulation of THI.

Figure 2.2: Formulation of EPI and THI.

In addition, using EPI, first of all, we should determine a appropriate height to cut off the spatio-temporal volume, though there is no good way to decide. Fig. 2.3 shows some EPIs of different heights. There are obvious differences in the shape of EPIs, therefore, we can not recognize builidngs stably, or cogitate some method,

for example, make EPIs by cutting off from every height, then integrate these EPIs into one EPI by the average color. However, it needs huge labor hours and may cause some unpredictable error.

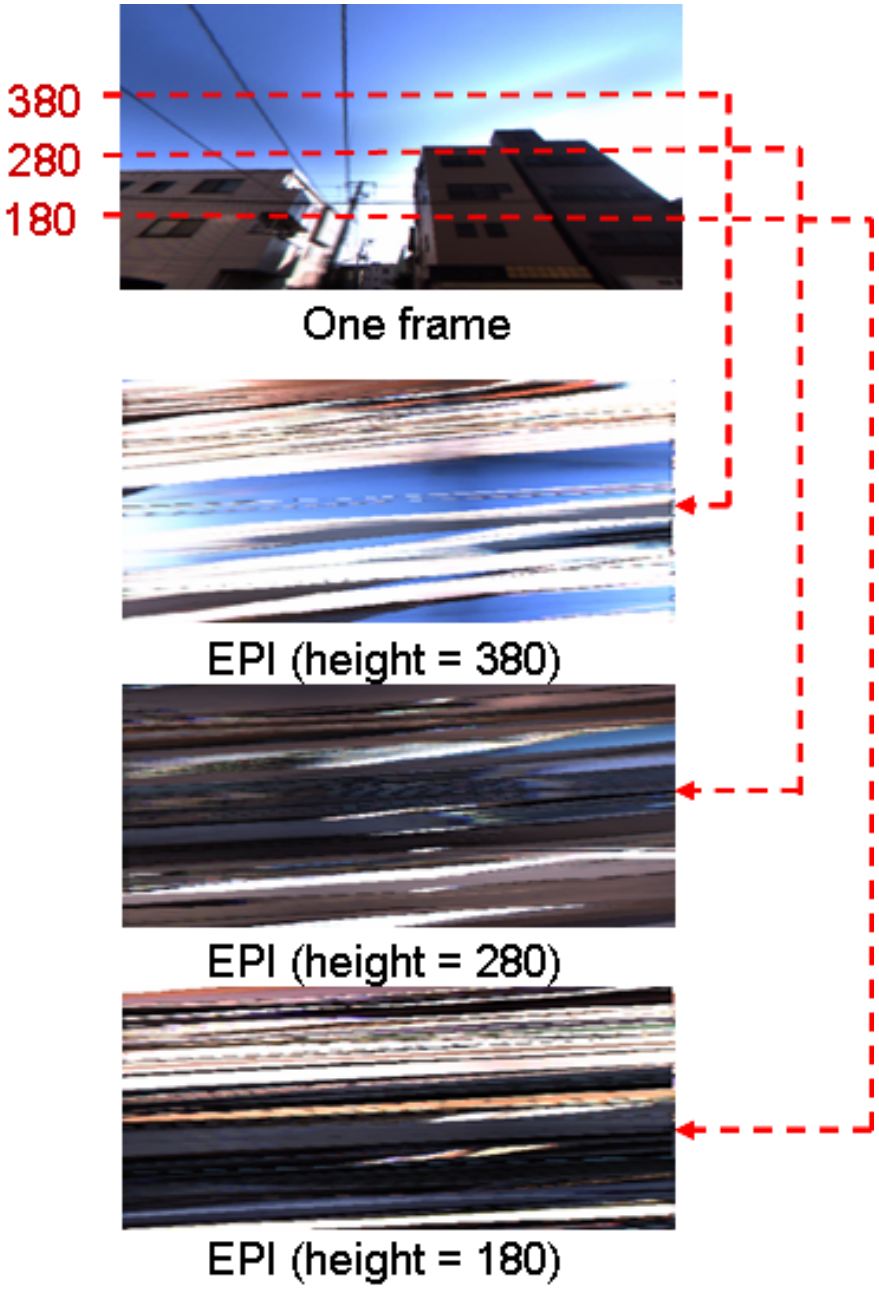


Figure 2.3: Different EPIs by being cut off from different heights.

2.2 Definition of Temporal Height Image (THI)

Herein we suggested a novel image(Temporal Height Image, THI) to express spatio-temproal volume. In EPI, because only a slice of spatio-temporal volume is used, most of the information about the volume is lost, especially the height of buildings can not be obtained from EPI. To get over these shortages, we take notice of the boundary of building and sky(normally building roof), because there is no affect of inside edges and we need not determine appropriate height.

We define THI as follows:

1. Obtain a spatio-temproal volume as the same process as EPI(Fig.2.2(a)).

2. For all the objects in the volume, assign a gray value in proportion to the height of the object. Through this process, in the volume, the higher the object, the brighter it becomes.

3. Look at this volume from above, we can see a overhead view called THI (Fig. 2.2(c)).

The key point is the edges of building roofs, thus there is a additional constraint that roofs must be reflected on images (means the sky must be reflected).

And because we mainly utilized the height of building, it is considered that, the larger the height change between two approximate buildings, the more effective THI becomes.

2.3 Comparison of EPI and THI

In this section, We will compare the features of EPI and THI. As shown in Fig. 2.4, the red points are used for EPI and the green broken lines are used for THI. In EPI, we can extract the red points by the intersection point of the cutting plane and the edges which are caused by blinking color change, meanwhile in THI, we can recognize the green broken lines as the highest edges on the image. So we can say EPI depends on color of buildings while THI depends on shape of buildings. We assemble the result of comparison as following:

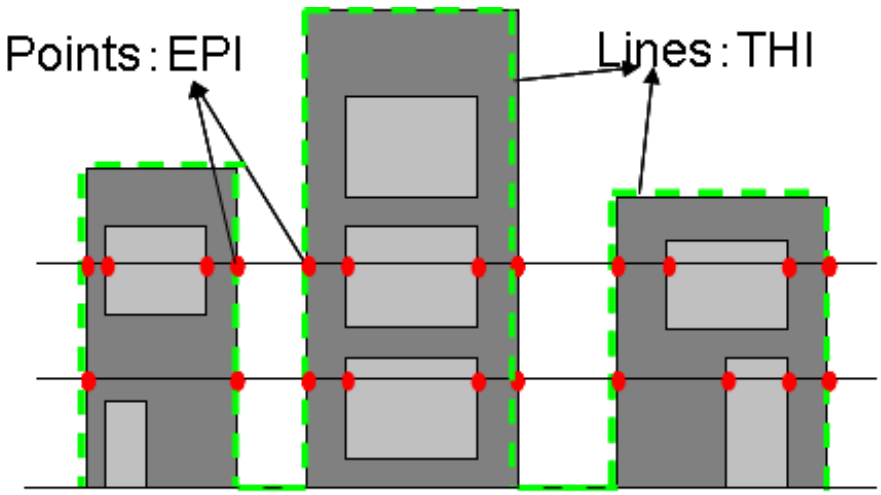


Figure 2.4: Comparison of EPI and THI.

Item	EPI	THI
Feature	Depend on color	Depend on shape
Information to be utilized	Vertical edges	Outside edges
Using cross section or not	Yes	No
Available information	Horizontal edges only	Horizontal edges and height

Table 2.1: Comparison of THI and EPI.

In EPI, vertical edges of building are being used and we have to find a adequate cross section to cut for making EPI, while in THI, only outside edges are being used and we do not need any else another treatment. Moreover, we can obtain horizontal edges only which means width of building by EPI while horizontal edges and height both can be obtained by THI.

Fig. 2.5 shows the comparison of EPI and THI by real urban scene. According to this reslut, it is obvious that there are more noise of edges in EPI than in THI, showing the advantage of THI.

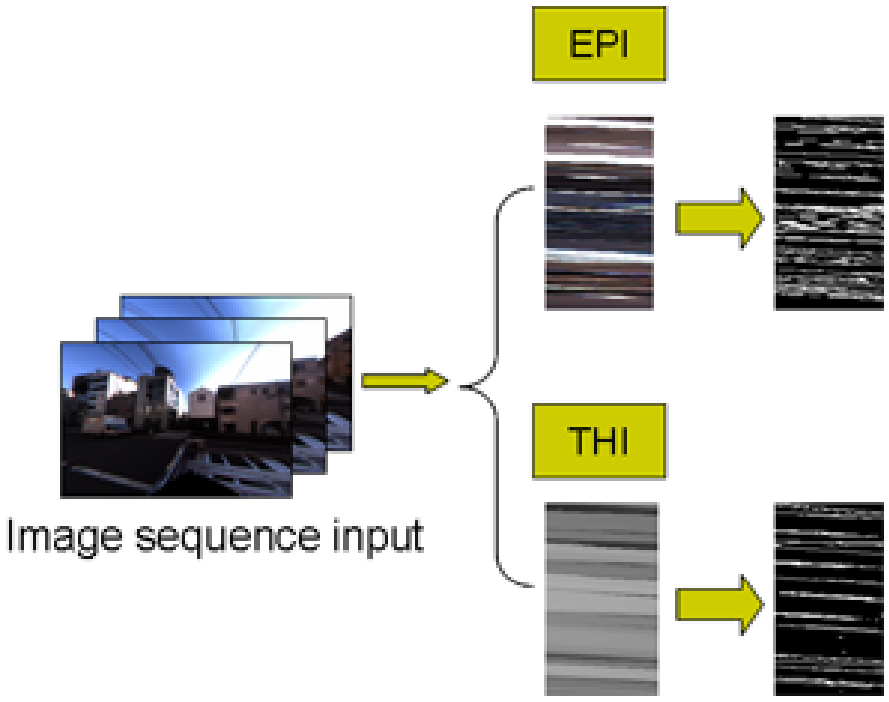


Figure 2.5: Comparison of EPI and THI by experiment.

2.4 Formulation of THI from Image Sequence

In this section, I will explain how to create a THI from urban image sequence caputred by on-vehicle camera in detail. The steps for making THI are shown as Fig. 2.6. For a single image, first, we remove electric wires to extract roof edges correctly, then, write out the ordinate values of roof edges in each abscissa pixel, thus a single image can be converted to a queue. Doing the sampe process for all the images and arranging converted queues temporally, we can obtain THI.

2.4.1 Removal of Electric Wires

We have to recognize building roof by extracting the highest edges on the image, but the electric wires usually seen in urban area become disturbed. so first, we should remove these electric wires, especially those above roof.

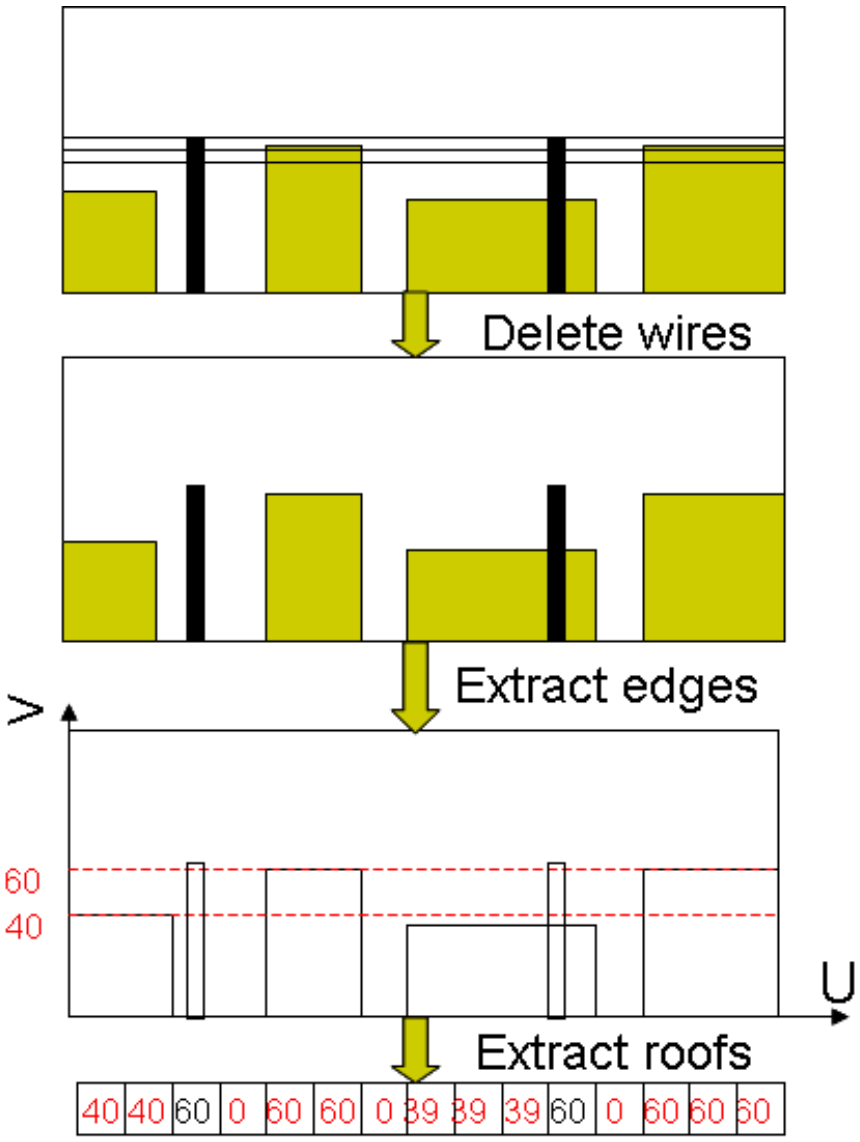


Figure 2.6: Process of creating THI from image sequence.

Median Filter

To remove noises from image, median filter is a very famous method, and our proposed method is based on it, so first I will explain the algorithm of median filter.

The median filter is a non-linear digital filtering technique, often used to remove

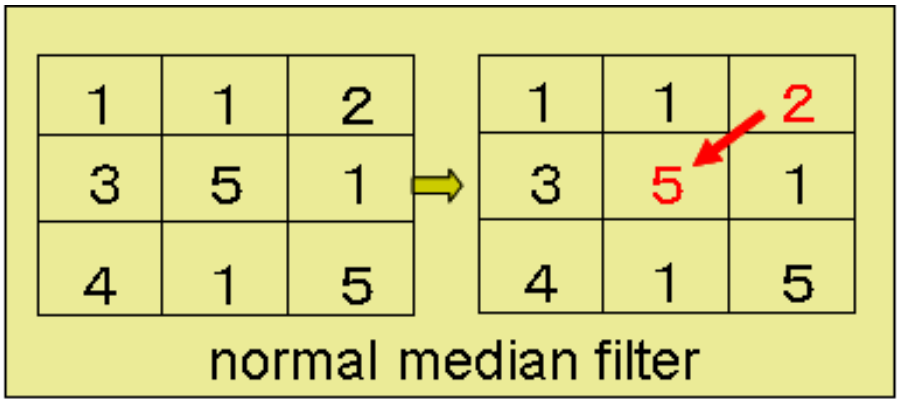


Figure 2.7: Principle of median filter (i.e. 3×3).

(a) Input.

(b) Output by 5×5 median filter.

Figure 2.8: Thrived example of median filter.

noise from images or other signals. The idea is to examine a sample of the input and decide if it is representative of the signal. As shown in Fig. 2.7, this is performed using a local window consisting of an odd number of samples. The values in the local window are sorted into numerical order, the median value, the sample in the center of the window, is selected as the output. The oldest sample is discarded, a new sample acquired, and the calculation repeats. Normally, the larger the local window, less the noises become, while unsharper the image becomes.

The median filter is particularly useful to reduce speckle noise and salt and pepper noise. For example, Fig. 2.8 shows a successful example to remove speckle noise(green noise) by median filter.

Its edge-preserving nature makes it useful in cases where edge blurring is undesirable. Fortunately, in this research, I have to extract the highest edge of building which conforms to the property of median filter.

On the other hand, it is not so useful to remove consecutive noises, since consecutive noises can occupy more spaces of local window so that even median value may be still noise. By this reason, electric wires can not be removed clearly by traditional median filter (Fig. 2.9).

Wire-Delete Filter

So inhere, we propose a new filter to delete electric wires which conforms to the property of electric wire in urban area. First, let us focus at the feature of electric wires in urban area. As shown in Fig. 2.9(a), there are two features:

1. most of the electric wires are set flatly.

2. the color of electric wire is darker than surround color (especially when circumambience of electric wires is sky)

To utilize there two features, I improve the normal median filter as following (Fig. 2.10):

1. Change the local window size from 2-dimension (i.e. 2×2) to vertical 1-dimension(i.e. 3×1)

(a) Input of urban image.

(b) Result by 9×9 median filter for the red field of input.

Figure 2.9: Fail to delete all the electric wires.

2. Instead of median value, we select a brighter value to modify the value in the center of the local window.

Actually, we have to determine some paramaters when using this wire-remove filter:

1. The size of local window $1 \times n$. Normally, the larger, the clearer wires can be removed, meanwhile image quality will fall down and processing speed will be down.

2. The turn of color value m to choose for substituting the central one. Normally, the brighter the m to choose, the better wires can be removed, meanwhile the whole image will become brighter.

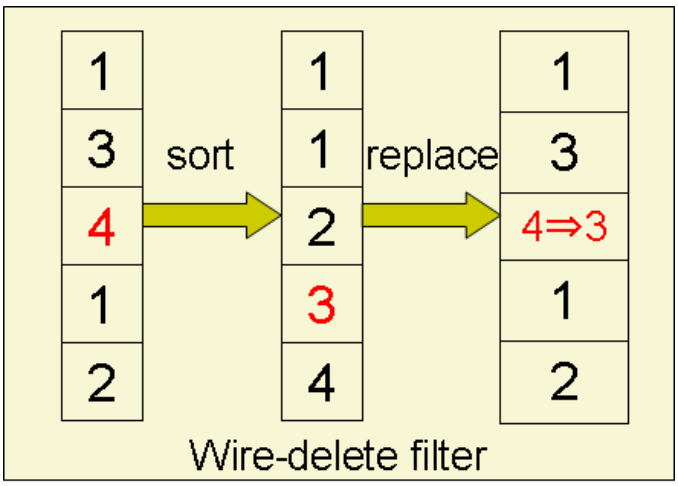


Figure 2.10: Principle of wire-remove filter.

So we have to decide a minimum of n and m, which can remove all the wires, at the same time, keep image quality.

Fig. 2.11 shows some examples processed by proposed wire-delete filter. We change the parameters of n and m from 3 to 7, then decide $n = 5, m = 4$ is the best for scene1, and $n = 5, m = 5$ is the best for scene 2. All the horizontal electric wires are removed clearly.

Using this wire-delete filter, we first remove the electric wires from image sequence when creating THI from real images if necessary.

2.4.2 Extraction of Building Height

In this part, we will explain how to extract the building roofs and create THI from wire-deleted image sequence.

As shown in Fig. 2.6, for a single image, after removing electric wires, we will extract edges, and decide the highest edges as roof edges. To express the ordinate of these roof edges as visible colors (0-255):

$$Brightness[i][x] = \frac{EdgeHeight[x] \cdot 255}{ImageHeight} \tag{2.2}$$

Here, $Brightness[i][x]$ means the brightness of abscissa x on the image i. $EdgeHeight[x]$ means the height of highest edge on the abscissa x. $ImageHeight$ means the height of image. Then process all the images by the same way(it means change i from 0 to the end of image sequence), we can obtain the THI by arranging $Brightness[i][x]$ temporally(Fig. 2.14(a)).

2.5 Formulation of THI from 3D House Map

To match buildings between image and house map, we have to create THI from house map as a reference object. First, because building heights are needed, instead of normal 2D house map, we prepared 3D house map including building height. Here, two kinds of 3D maps are employed:

1. ZENLIN map (Fig. 2.12(a))[2], it includes the information of story number, thus by assuming the height of one floor is constant, we can estimate the height of building, and the building can be expressed as a cube, even though it is not so accurate.

2. ASAHI map (Fig. 2.12(b))[5], it is made by aerial survey with very high accuracy, moreover even complex shapes can be expressed.

The process of making THI from building models is similar as the process from image sequence, just one step is added: Instead of real on-vehicle camera, we have to design a virtual camera in the 3D map to capture images. However, Note that in the next step of matching, we hope the same shape and color of bands from Image-THI(THI made from image sequence) and Model-THI(THI made from building models) if the two bands mean the same building essentially. So we have to design the virtual camera as similar as possible to the real on-vehicle camera. Fortunately, we use a omni-view camera (Ladybug2 Fig. 3.5, [20]), which is easy to adjust the camera parameters such as optic angle and capture direction. Therefore, it is not so difficult to calibrate the virtual camera to the real on-vehicle camera. Fig. 2.13(a) shows an example of picture captured by virtual camera on ZENLIN, and Fig. 2.13(b) shows an example of picture captured by virtual camera on ASAHI. It is

clear that ASAHI‘s map is nore accurate than ZENLIN‘s.

The remaining steps are just the same as the steps of making THI from image sequence. Fig. 2.15(a)) shows an example of THI made from building models.

2.6 Recognition of Bands from THI

The same as EPI, objects are expressed as bands in THI. In this part, we will explain the process of recognizing these bands. as explained in section 2.3, THI can remain only necessary edges (band contours), so that we can recognize these bands by edge division.

Here is a new problem, as shown in Fig. 2.14(a), the bands in the central field(red field) of THI can keep regular shapes as parallel lines (since all the buildings are on the same depth), and empty spaces between two approximate bands are large enough to be extracted by edges, however, in the surrounding field, the bands may overlap each other because the side faces of buildings are also reflected on the image, thus we can not extract contours exactly. To reduce the affect of side faces, we decide to use the central field only, even though this may lose some informations.

Fig. 2.14(b) shows the distended red field of Fig. 2.14(a), and Fig. 2.14(c) shows the reslut of adding edges onto Fig. 2.14(b). After adding edges onto THI, we can consider the area between two approximate edges as a band, Fig. 2.14(d) shows the result of recognized bands expressed by random colors.

If the band means one building, the brightness of band is in proportion to the height of building, and the width of band on temporal coordinate (ordinate) is in proportion to the width of building, when assuming all the buildings are on the same depth.

However, because there are some noises, for example, trees and electric poles, we can not estimate the brightness of band stably, here, we utilize the algorithm of $floodfill$ supplied by openCV to calculate the average brightness and estimate the width of band.

The $floodfill$, also called seed fill, is an algorithm that determines the area connected to a given node in a multi-dimensional array. It is used in the "bucket" fill tool of paint programs to determine which parts of a bitmap to fill with color. When applied on an image to fill a particular bounded area with color, it is also known as Boundary fill. The flood fill algorithm takes three parameters: a start node, a target color, and a replacement color. The algorithm looks for all nodes in the array which are connected to the start node by a path of the target color, and changes them to the replacement color. In this research, we define start nodes by a constant interval from the top of image to the bottom, and the abcissa is the center of image, then define the replacement color by random color, so we can obtain the result as Fig. 2.14(d), 2.15(d).

By these processes, we can recognize bands including informations of width and height of building, also the vertical contour‘s abscissa on the image which is useful when estimating vehicle location.

Next, for the THI made from map, because there are only buildings in the map, it is easier to recognize bands even not need to add edges, only by the algorithm $floodfill$ to extract necessary information: average brightness and the width.

(a) Input of scene 1.

(b) Result by wire-remove filter ($n = 5, m = 4$) of scene 1.

(c) Input of scene 2.

(d) Result by wire-remove filter ($n = 5, m = 5$) of scene 2.

Figure 2.11: Two examples of wire-remove filter

(a) ZENLIN

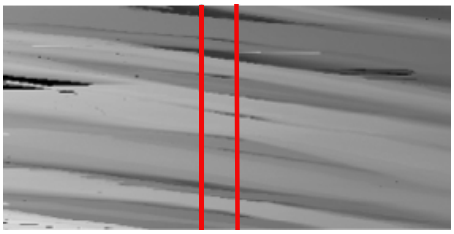
(b) ASAHI

Figure 2.12: Two kinds of 3D house maps.

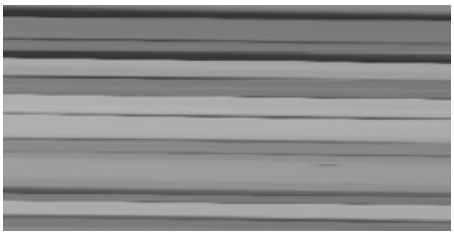
(a) One frame from ZENLIN

(b) One frame from ASAHI

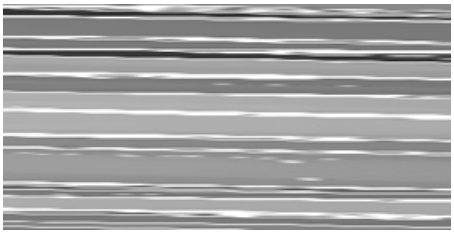
Figure 2.13: Examples of frame captured by virtual camera on 3D map.



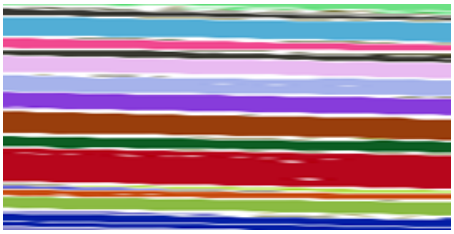
(a) An example of THI.



(b) Real field to be used.

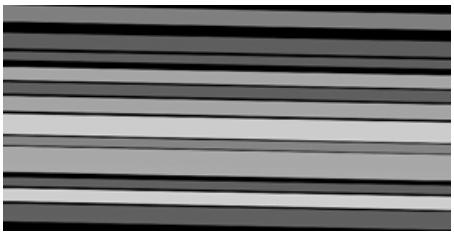


(c) THI with edges.

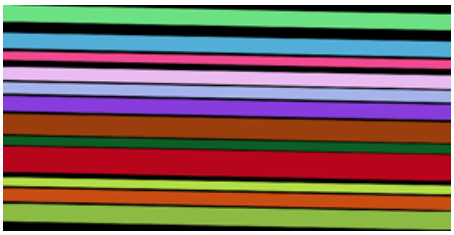


(d) Expressing recognized bands as random colors.

Figure 2.14: Recognition of bands from THI made from image sequence.



(a) THI from map.



(b) Expressing recognized bands as random colors.

Figure 2.15: Recognition of bands from THI made from maps.

Chapter 3

Mathcing Buildings from Image Sequence and Building Models by Dynamic Programming Matching

meant Using correspondence infromation of real image and building model, we can attach texture onto building model, and estimate vehicle location. In this chapter, we will explain the method to match buildings from real image and map model. In the previous chapter, we got bands from THI. Normally, bands from Model-THI (THI made from building model), can be considered that there are only buildings, and no noise, meanwhile bands from Image-THI (THI made from image sequence), some bands are made from noises (i.e. electric pole, tree etc.), and also, some adverse problem may happen (i.e. one building are divided to more than two bands because of unflat roof). To remove noises and adjust these adverse cases, we apply Dynamic Programming matching (DP matching).

3.1 Dynamic Programming Matching (DP Matching)

Pattern matching based on Dynamic Programming matching(DP matching) has been studied widely. In previous research, DP matching was used to match a video image with a 2D digital map[14], match two series of image sequences captured in urban area on different dates[21], and match building models acquired by range

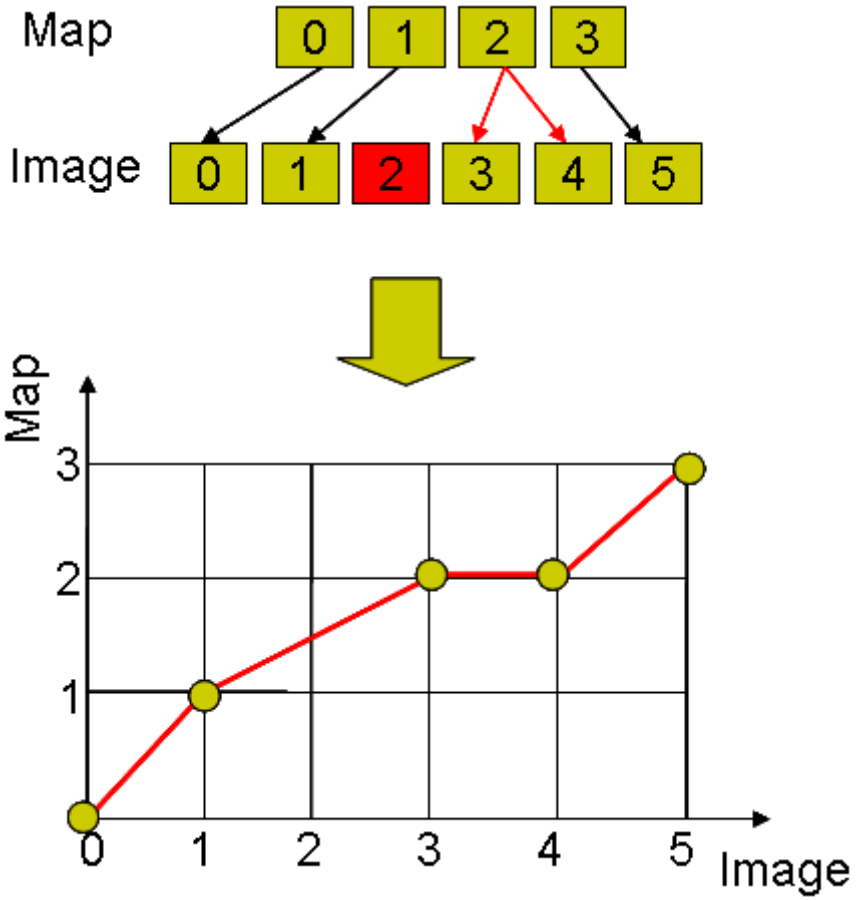
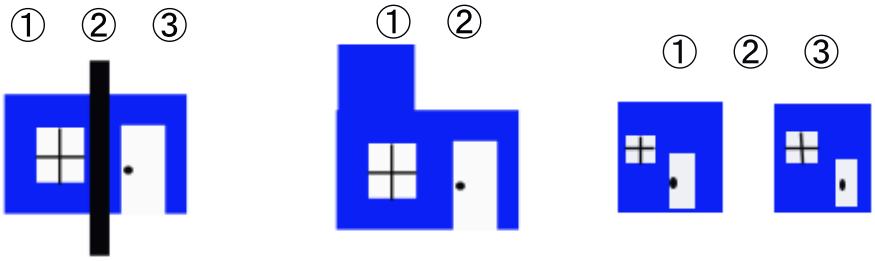


Figure 3.1: Dynamic Programming.

senser with 3D map[22, 23]. DP matching is suit to match queues including noises. Here we use DP matching to match bands from Image-THI and Model-THI.

DP matching is a method of solving problems exhibiting the properties of overlapping subproblems and optimal substructure that takes much less time than naive methods. Here, optimal substrcture means that optimal solutions of subproblems can be used to find the optimal solutions of the overall problem. For example, in our research, as shown in figure 3.1, from map model, we found 4 bands which means buildings, while from real image, we found 6 bands which include some noises (one electric pole, two bonds means the same building). The corresponding



(a) Case 1. (b) Case 2. (c) Case 3.

Figure 3.2: Problematic cases.

relation can be expressed as the graph below. DP matching is just an algorithm to find out the shortest path from (0, 0) to (5, 3).

To set up the DP matching, we have to determine the cost of passable paths between every two approximate intersection points, and decide some constraints, and the cost function:

1. Passable paths: possible corresponding relations between bands on Image-THI and bands on Model-THI.

2. Value of intersection points: key value of calculating path cost.

3. Constaints: for improving the accuracy.

4. Cost function: final function to calculate the shortest path (minimum cost) from start to end.

3.1.1 Passable Paths

The ideal experimental situation is that there is no tree, no electric pole, even no traffic light, additionally, all the building roofs are flat. Unfortunately, in fact, all these cases may exist in urban scene, we have to suppose all of the possible cases, and solve them by designing appropriate paths. But if setting all the paths, it needs huge processing time and may make DP matching unstable, so here, as shown in Fig. 3.2, we only picked up 3 main cases:

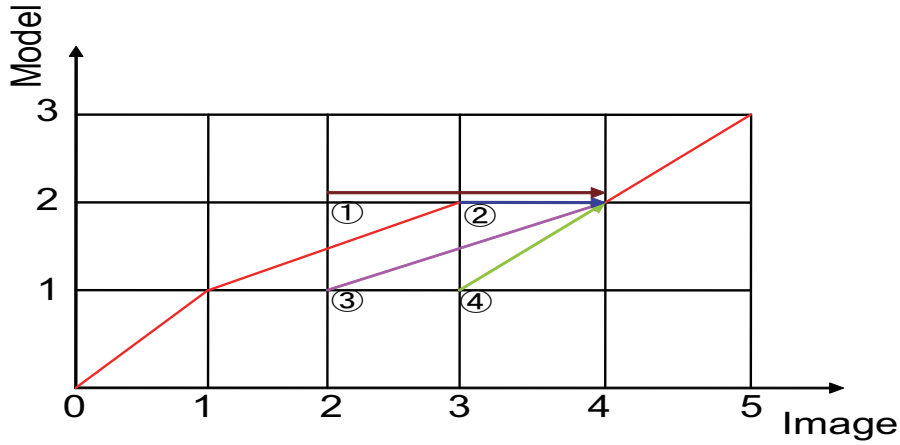


Figure 3.3: Design of passable paths.

Case 1. some noise (electric pole or traffic light) are recognized as a band on the real image, and the building are divided into two parts because the noise is in front of building and cuts off it.

Case 2. building roof is not so flat that when extracting highest edge, there many have some gaps which make one building recognized as two bands.

Case 3. some noise (electric pole or traffic light or aperture between buildings) is recognized as one band, and there is no building behind this noise.

Certainly, there are also some other cases, for example, some continuous trees, the flat is too complex that we can not divide the building to two bands simply and so on. According to our observation of experimental scenes, these cases are too rare. To save processing time, we ignored these rare cases.

According to possible cases pointed out above, We design the passable paths as Fig. 3.3. For example, let us determine the passable paths to point (4, 2), there are 4 paths can be decided.

Path 1. $(2, 2) \rightarrow (4, 2)$, corresponding to case 1, band 2 and band 4 on image correspond to the same band 2 on model, and band 3 means a noise, which means the noise (band 3) cut off the building to two parts (band 2 and band 4).

Path 2. $(3, 2) \rightarrow (4, 2)$, corresponding to case 2, band 3 and band 4 on image

correspond to the same band 2 on model, which means the building roof is not so flat that one building is divided to two parts.

Path 3. $(2,1) \rightarrow (4,2)$, corresponding to case 3, band 2 on image correspond to band 1 on model, and band 4 on image correspond to band 2 on model, the band 3 is considered as noise, which means some single noise is recognized as a band but fortunately, there is no building behind it.

Path 4. $(3,1) \rightarrow (4,2)$, this is a normal path which means band 3 on image correspond to band 1 on model and band 4 on image correspond to band 2 on model by turns. It means there is no noise around this building and the roof is flat enough.

3.1.2 Value of Intersection Point

To find out the shortest path from start to end, we have to determine the cost of each path, the value of intersection point can play this role.

I define the value of intersection point by difference of aspect ratio of buildings as:

$$V(i,j) = \left| \frac{H_I[i]}{W_I[i]} - \frac{H_M[j]}{W_M[j]} \right| \tag{3.1}$$

Here, V(i,j) is the value of point (i,j), $H_I[i]$ is the height of band i on image, $W_I[i]$ is the width of band i on image, $H_M[j]$ is the height of band j on model, $W_M[j]$ is the width of band j on model. So if band i on image and band j on model are pair, V(i,j) will become small, otherwise, V(i,j) will become large.

However, we should take notice that, as shown in Fig. 3.3, in path 1 and path 2, band i and band j ($j = i-2$ in path 1 and $j = i-1$ in path 2) are integrated. It means the value of intersection point should be changed in these case. We defined the integrated value $V_f(i,j,k)$as:

$$V_f(i,j,k) = \left| \frac{H_I[i] + H_I[j]}{2 \cdot (W_I[i] + W_I[j])} - \frac{H_M[k]}{W_M[k]} \right| \tag{3.2}$$

Here, i and j are the band indexes on image, in path 1, $j = i-2$, and in path 2, $j = i-1$. The integrated height of the two bands is the average of the two heights, and the integrated width of the two bands is the sum of the two widths.

After deciding the value of each intersection point, we can easily decide the cost of each path, by just plus $V(i,j)$, but because we designed several special paths, such as skip a noise and integrate two bands, we need determine the costs of these special actions. It will be discussed next.

3.1.3 Constraints

To find out the shortest path accurately and determine the costs of those special actions, we add three constraints.

1. The height transition of the two approximate buildings shoud be the same on image and model. For example, if left building is higher than right one on image, left building must be higher than right one on model. Write this constraint as equation:

$$D_h(i,j,m) = \begin{cases} 1 & (H_I[i] - H_I[j]) \cdot (H_M[m] - H_M[m-1]) \geq 0 \\ +\infty & (H_I[i] - H_I[j]) \cdot (H_M[m] - H_M[m-1]) < 0 \end{cases} \tag{3.3}$$

Here, i and j are band number on image, m is band number on model, if band i and j are continuous buildings, $j = i - 1$, otherwise, for example, if there is one noise between band i and j, $j = i - 2$. $D_h(i,j,m)$ is the value of this judgement function. $H_I[i]$ is the height of band i on image, $H_M[m]$ is the height of band m on model, so it means if close-by buildings on image and on model show the same height transition, $D_h(i,j,m)$ become 0, otherwise, it will become infinite.

2. The narrower the band recognized from Image-THI is, the higher probability it is noise. As explained above, noises, for example, electric poles or trees are much narrower than buildings, so that we can consider narrow bands as noise. Express this constraint as equation:

$$C_s(i) = P_s \cdot W_I[i] \tag{3.4}$$

Here, $C_s(i)$ is the probability that band i is noise, P_s is a scale constant number decided by scenes, and $W_I[i]$ is the width of band i on image.

3. The two approximate bands recognized from Image-THI may mean the same building if the heights of the two bands are close enough. As explained above, one building may be divided to two parts by noise in front of building. To compute the probability of this case, we designed the path as Fig. 3.3, and to improve the accuracy, we added this constraint shown by equation as:

$$C_f(i,j) = P_f \cdot \mid H_I[i] - H_I[j] \mid \tag{3.5}$$

Here, $C_f(i,j)$ is the probability that the two bands point to the same building, P_f is a scale constant number, which is decided by practical scene, and $H_I[i]$ means the height of band i on image.

3.1.4 Cost Function

According to the discussion above, we determine the cost function which includes the value of intersection points and three constraints. If we define the cost from start(0,0) to point (i, j) as $D(i,j)$, then

$$D(i,j) = \min \begin{cases} D(i-2,j) + V_f(i-2,i,j) \cdot C_f(i-2,i) \cdot C_s(i-1) \\ D(i-1,j) + V_f(i-1,i,j) \cdot C_f(i-1,i) \\ D(i-2,j-1) + V(i,j) \cdot C_s(i-1) \cdot D_h(i,i-2,j) \\ D(i-1,j-1) + V(i,j) \cdot D_h(i,i-1,j) \end{cases} \tag{3.6}$$

Here, each line means the cost of one passable path (Fig. 3.3):

The first line means the cost from point $(i-2,j)$ to (i,j) through path 1. In this path, there are processes of fusion and skiping noise.

The second line means the cost from point$(i-1,j)$ to (i,j) through path 2. In this path, there is process of fusion.

The third line means the cost from point $(i-2, j-1)$ to (i, j) through path 3. In this path, there are process of skiping noise and height judgement.

The forth line means the cost from point $(i-1, j-1)$ to (i, j) through path 4. There is no special action in this path.

Obviously, the cost of path 4 is the lowest one, but we can not always connect start point with end point only by path 4. other paths should be applied too. So we use this cost function to determine the lowest and practicalbe path.

3.2 Experiment of DP matching

To examine the effect of THI and DP matching, we matched buildings on image and on map in some real urban scenes. First, we will show our data-acqusition system, then we will show the DP matching results of three scenes.

3.2.1 Experimental Condition

We designed our data-acqusition system based on a TOYOTA van, as shown in Fig. 3.4, computers for data saving and image processing were placed into the van, and cameras (red ones) were set on the roof.

We have to keep the whole building being reflected all the time, which is difficult for normal camera. So here, we decide to utilize a omni-view camera – Ladybug, as shown in Fig. 3.5(a), it can take a picture with a 360-degree field vision, of course, the whole of building could be reflected easily.

However, this omni-view picture is not so easy to deal. Fortunately, omni-view camera is easy to determine the shoot direction, so we projected the omni-view picture onto a plane paralleling to the facade of building. Fig. 3.5(b) shows an example of projected image from Fig. 3.5(a). Buildings are reflected vertically, which will be useful in the process of texture mapping.

For the 3D map, we emploied ZENLIN (Fig. 2.12(a)) and ASAHI (Fig. 2.12(b)), two kinds of maps. In addition, to make DP matching simpler, one building model is preferred to be recognzied as only one band from THI, however, if the roof is

Figure 3.4: Our data-aquisition system.

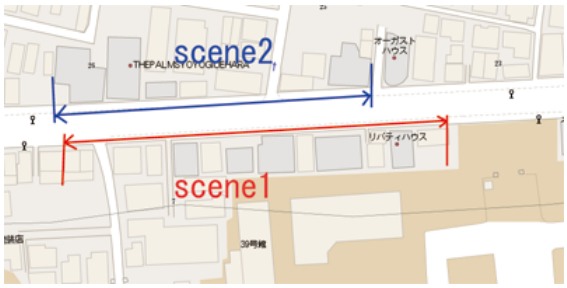
(a) Omni-view image. (b) Projected image.

Figure 3.5: Omni-view image and projected image.

not flat enough, the building may be recognized as more than two bands, so we uniformized the building roofs on the ASAHI map (Fig. 3.7(b)) by the average height.

3.2.2 Scene 1 (Around School).

As shown in Fig. 3.6(a), we first had a experiment around our school, scene 1 is the south area on the map. There are twelve buildings in scene 1 and Fig. 3.8(a,b)



(a) Location of scene 1 and 2 around school.

(b) Location of scene 3 in Shinjuku.

Figure 3.6: Experimental locations.

shows some examples of buildings in scene 1. In this scene, electric wires were not so noticeable that we do not need to remove them purposely. And most of buildings have complex texture but simple shape.

To compare the features of EPI and THI, we made both EPI(Fig. 3.8(c)) and THI(Fig. 3.8(d)), moreover, the edge results of EPI and THI (Fig. 3.8(e,f)). It was obvious that there were more edge noises in EPI than in THI. so we can say, THI is more effective than EPI to recognize buildings in the urban scenes like scene 1, where buidings have many windows, meanwhile the shapes are not so complex.

Fig. 2.15(b) shows bands recognized from Model-THI. Twelve bands were recognized, which conformed to the fact. However, in the result of bands recognized from Image-THI (Fig. 2.14(d)), fifteen bands were recognized, which means there may be three noises in the bands recognized from Image-THI.

Then we carried out DP matching, as shown in Fig. 3.9(a), the abscissa means fifteen bands recognized from Image-THI, ordinate means twelve bands recognized from Model-THI, and the graph shows the correspoinding relation of these two kinds of bands. As a result, the graph has skipped the band $2, 5, 12$, which means that these three bands are noises, for example, as shown in Fig. 3.9(b), the band 2 on Image-THI was actually made from the building behind the main street (the red one).

(a) ASAHI original map.

(b) The uniformizing result.

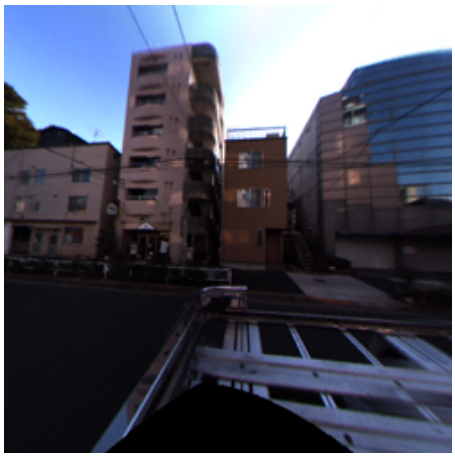
Figure 3.7: ASAHI original map and adjsuted map.

Both ZENLIN and ASAHI obtained the same result, and by our observation, all the buildings were well matched. So we can say that our proposed matching method works well in the scene with simple building shape and few noise, though very complex building texture.

3.2.3 Scene 2 (Around School).

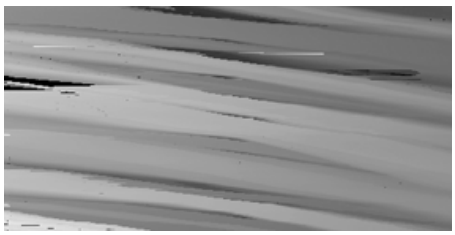
Scene 2 was just the opposite side of scene 1 (Fig. 3.6(a)). There were eight buildings, and with very noticeable electric wires and traffic light (Fig. 3.10(a,b)). So first we had to apply wire-delete filter to remove those eletric wires but there are too many wires to remove them clearly only by one time, as shown in Fig. **??**(c,d), they were the results of removing wires by 9×1 wire-delete filter only by once, some wires were still remained, so here we decided to carry out the wire-delete filter once more, the final results were Fig. 3.10(e,f), most of the horizontal electric wires

(a) Example 1 of scene 1.

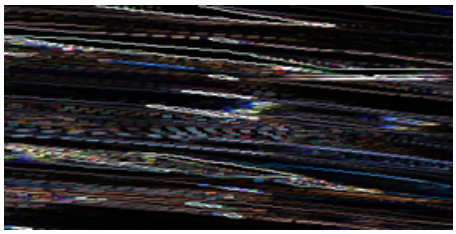


(b) Example 2 of scene 1.

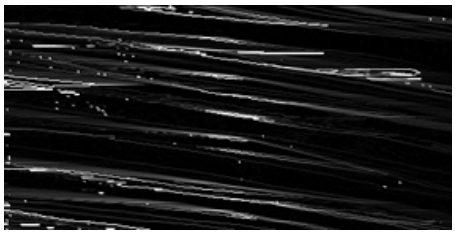
(c) EPI



(d) THI



(e) Edge image of EPI.

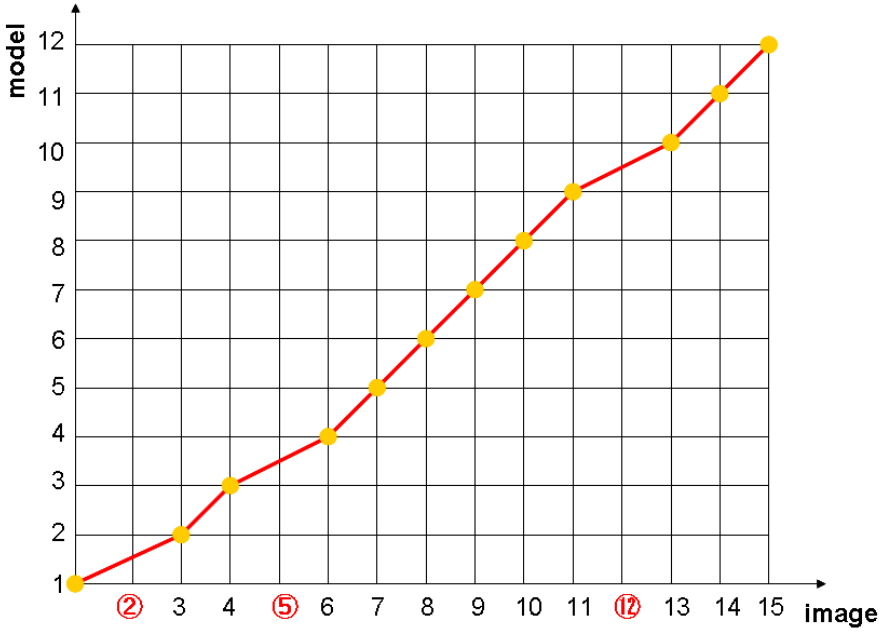


(f) Edge image of THI.

Figure 3.8: Experiment results of scene1.

were removed, though the color of buildings became unsharp. In addition, there were still some vertical electric wires on the final result (Fig. 3.10(e)), actually, we could delete easily it by designing the median filter as a horizontal filter (1×9), but because the vertical wires did not affect roof extraction, We skipped this step.

Since there were also many windows inside the buildings, the EPI result (Fig.



(a) The DP matching result of scene 1.

(b) An example of mismatching band.

Figure 3.9: Experiment results of scene 1.

3.11(a)) was too complex to recognize bands, of course the edge result of EPI was complex too (Fig. 3.11(b)). On the other hand, because there were still some traffic lights left, THI made from image sequence also remained some noise (Fig. 3.11(c,d)). And Fig. 3.11(e) showed the result of bands recognized from Image-THI and expressed by random colors, there were thirteen bands were recognized. In the meantime, Fig. 3.11(f) showed the result of bands recognizedd from Model-THI, there were eight bands recognized, that meant there were at least five mis-recognized bands on the Image-THI.

Fig. 3.12(a) showed the DP matching result. In this graph, red broken line meant the result using ZENLIN map, blue broken line meant the result using ASAHI, and the green line meant the true result by observation. In the result of using ZENLIN, band 3, 7 and 9 were mismatched, meanwhile, In the result of using ASAHI, band 3 and 9 were well modified, though band 7 was still mismatched. For band 3, it was a part of band 3 on model, but in the result of ZENLIN, it was recognized as a part of band 2 on model, for band 7, it was a part of band 5 on model but in the results of two maps, it was recognized as a noise, for band 9, as shown in Fig. 3.12(b), it was a part of band 7 on model, but in the result of ZENLIN, it was recognized as a noise. Accoding to this result, we could say the more accurate the 3D map was, the more correct the DP matching result became. As explained in section 3.1, in DP matching, the value of intersesction point was defined using aspect ratio of building, but if using ZENLIN, even though the two bands meant the same building, since the aspect ratio was not so accurate, the value of intersection point may be still large.

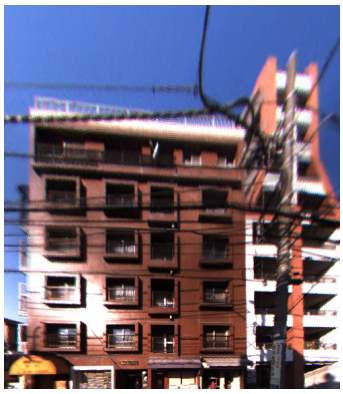
3.2.4 Scene 3 (Shinjuku).

At last, We challenged aligning tower buildings in Shinjuku (Fig. 3.6(b)). As shown in Fig. 3.13(a), there were almost tower buildings, with very simple shape but complex colors. This kind of scene was considered to be better suited for our proposed matching method. We chose a part in this scene, as shown in Fig.

3.13(b), where there were five tower buildings. And Fig. 3.14(a) was an example of omni-image input captured by Ladybug, and Fig. 3.14(b) was the projected image. According to this converted image, it was obvious that empty space between two neighbor buildings were large enough that buildings behind target buildings were reflected clearly too. Fortunately, there were no electric wires, so that we skipped the step of removing electric wires.

Fig. 3.15(a) showed the result of THI made from 3D map, and five bands were recognized (Fig. 3.15(b)). And Fig. 3.15(c) showed the result of THI made from image sequence, because the empty space between two approximate buildings was so large that buildings behind and the side faces were reflected clearly, to reduce these effect, we cut out the central field (red one) for recognizing bands. Fig. 3.15(d) showed the result of recognized bands from Image-THI, nine bands were recognized, much more than the true building number – five. Checking these recognized bands, most of the mis-recognized bands were the empty spaces between two approximate buildings or buildings behind. Here, since all the target buildings were much higher than empty space, We added a threshold of height to reduce the number of recognized bands which made from empty space, it meant that, we only selected those bands whose brightness were brighter than threshold. As a result, the number of recognized bands was reduced to six.

Fig. 3.16(a) was the result of DP matching in shinjuku. In this graph, abscissa meant bands recognized from Image-THI, six bands were recognized on Image-THI, and ordinate meant bands recognized from Model-THI, five bands (buildings) actually existed. As a result, the band 5 on Image-THI was made from the building behind target buildings (Fig. 3.16(b), drew round by red color).

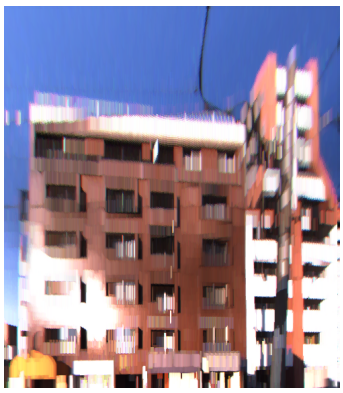


(a) Frame 1 from image sequence of scene 2.

(b) Wire-delete result of frame 1 by once.

(c) Frame 66 from image sequence of scene 2.

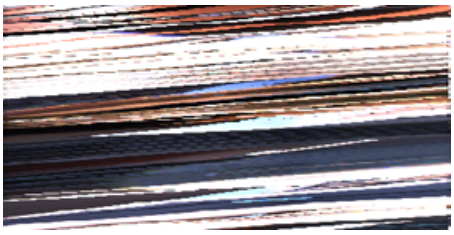
(d) Wire-delete result of frame 66 by once.



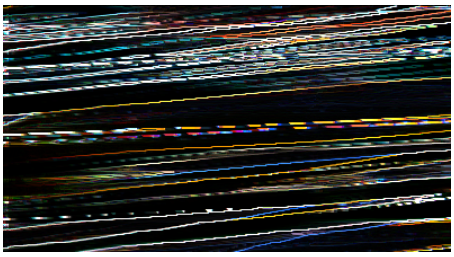
(e) Wire-delte result of frame 1 by twice.

(f) Wire-delete result of frame 66 by twice.

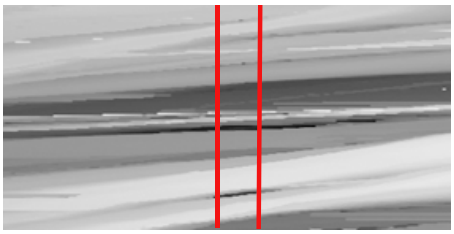
Figure 3.10: Input and the result of removing electric wire.



(a) EPI.



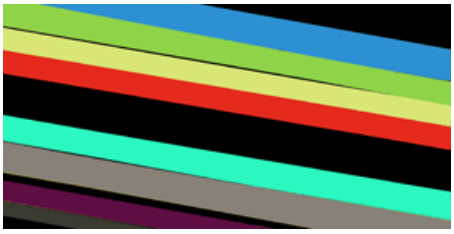
(b) Edge result of EPI.



(c) Image-THI.

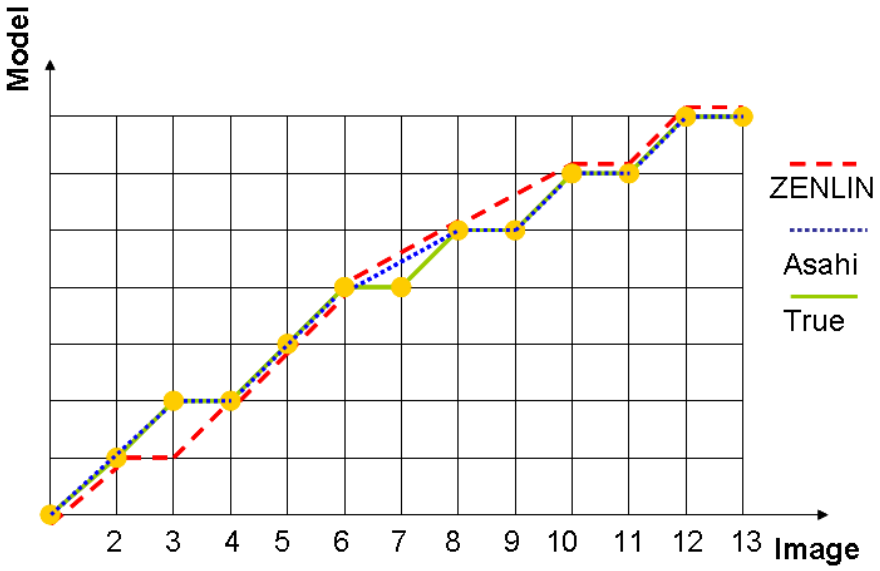
(d) Edge result of THI.

(e) Recoginzed bands from Image-THI.



(f) Recoginzed bands from Model-THI.

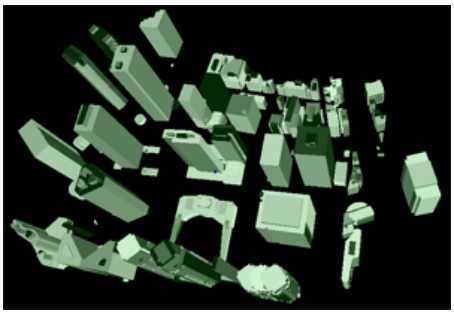
Figure 3.11: Experimental results of scene 2.



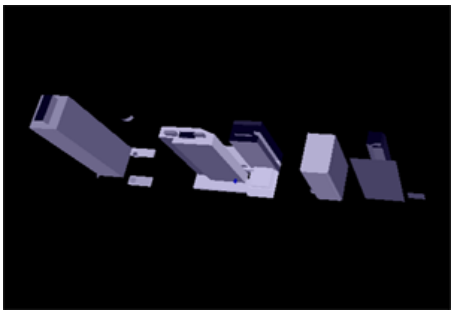
(a) The result of DP matching.

(b) Mismatching example of DP matching.

Figure 3.12: DP matching result of scene 2.

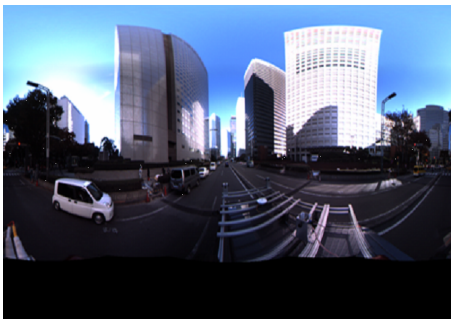


(a) The whole 3D map of shinjuku area.

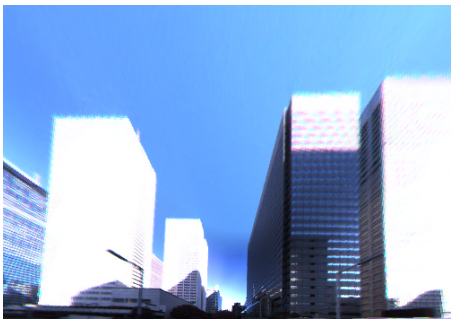


(b) Selected area of shinjuku.

Figure 3.13: 3D building model of shinjuku.

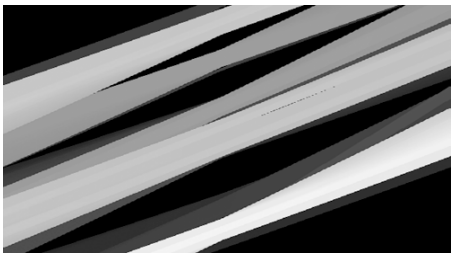


(a) An example of ladybug image captured in shinjuku.

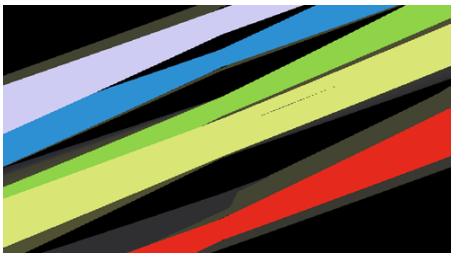


(b) A converted image from ladybug image of shinjuku.

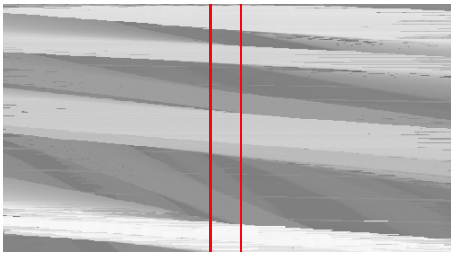
Figure 3.14: Ladybug and projected image of shinjuku.



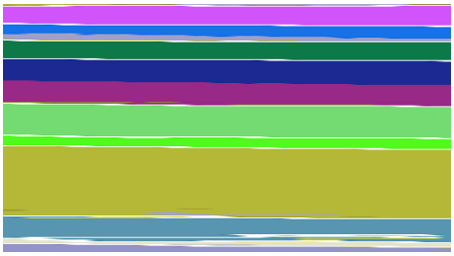
(a) THI made from 3D building model in shinjuku.



(b) Recognized bands from Model-THI.

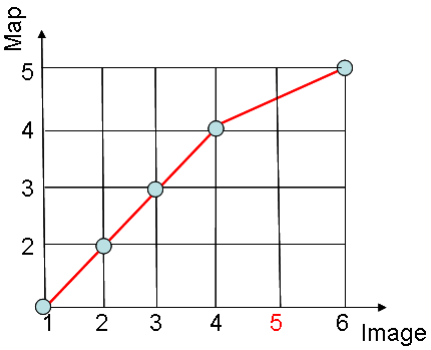


(c) THI made from image sequence in shinjuku.



(d) Recognized bands from Image-THI.

Figure 3.15: THIs of 3D building model and image sequence in shinjuku.



(a) DP matching result of shinjuku.

(b) The mismatching band.

Figure 3.16: DP matching result and mismatching example of shinjuku.

Chapter 4

Texture Mapping

In ITS field, 3D city map, especially a color one, plays very important role. Some companies (i.e. Google, Microsoft) have made their debuts of color 3D city map products, whereas, all those textures of building are made by CG. This process needs huge labor hours, and lacks of actuality. Here, we propose an automatic texture mapping method onto colorless 3D building model using real images captured by on-vehicle camera. To achieve this goal, there are three steps:

1. Recognize buildings on actual urban image stably.

2. Design a method of determining corresponding relation of building textures and building models.

3. Define the texture mapping method.

We have designed THI and DP matching to solve the first two issues. After DP matching, as shown in Fig. 4.1, we can know the corresponding bands between Image-THI and Model-THI. It just means corresponding indexs of bands, but not the correct position of each buildings on the single image and building models(red rectangle). In the building models, the contour (red rectangle) can be assigned in advance, so here, We have to propose a method to recognize the exact position of the contour (red rectangle) on images.

At the same time, using the corresponding relation of buildings on image and on model, we can also estimate the vehicle location where the images were captured. It can be achieved by finding out corresponding image captured by virtual camera

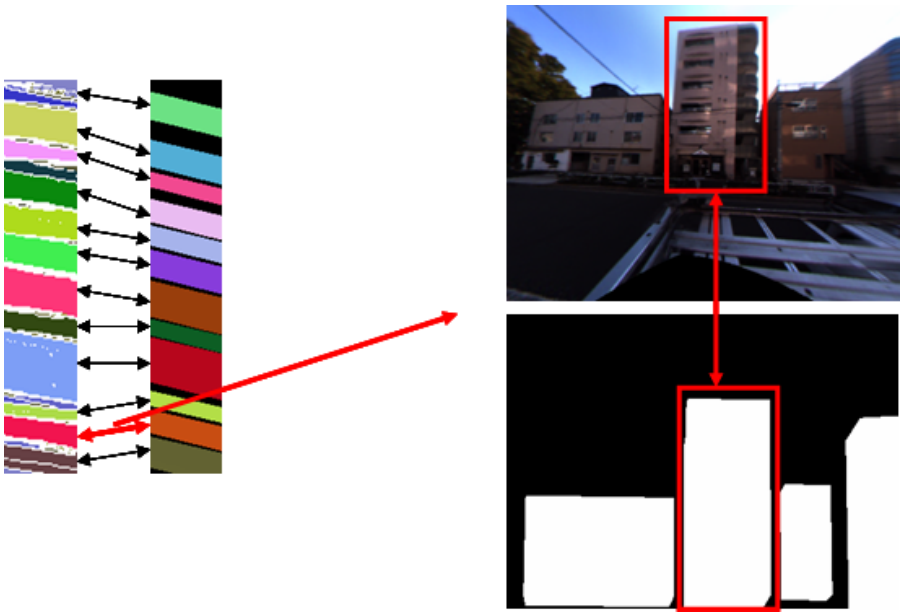


Figure 4.1: Corresponding relation obtained by DP matching.

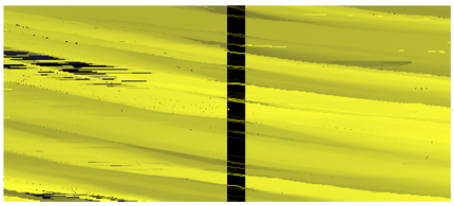
in 3D map, threfore the location of virtual camera can be looked upon as the real vehicle location. The process is very near to texture mapping, so we will explain it later.

4.1 Simplification of Texture Mapping Process

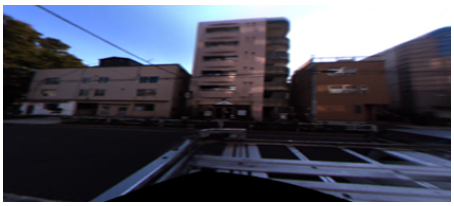
To make the process of texture mapping easily, we add three assumptions:

1. Approach the building facade as a rectangle. It means that the roofs with complex shape are ignored. By this assumption, texture mapping become the issue that to determine the four corresponding corners of the rectangle. Since the corners of building model has been assigned, we just have to extract the four corners of building facade on a appropriate single image.

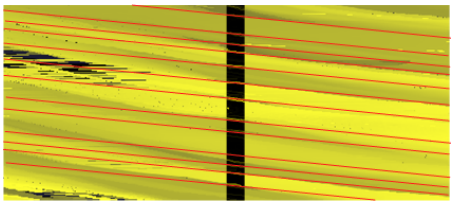
2. Buildings are reflected vertically on the real images, which can be achieved easily by adjust parameter of omni-view camera. By this assumption, corners on the same vertical contours could have the same abscissa, and just the same as the two vertical contours‘ abscissas.



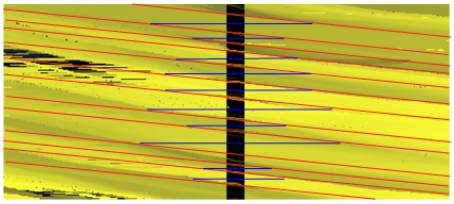
(a) An example of THI for recognizing corners on the roof.



(b) The frame correspoinding to the height of 191 on the THI.



(c) Dividing THI into several parallel bands.



(d) Estimation of building widths.

Figure 4.2: Process of estimating building widths.

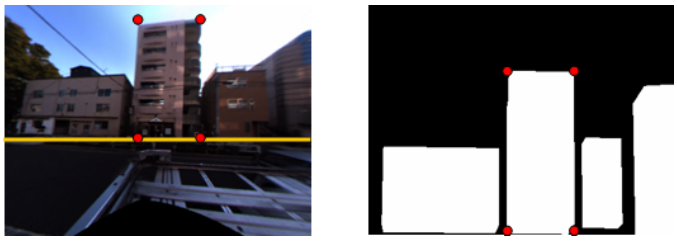
3. The height of boundary of buildings and ground should keep constant. It means the up and down of vehicle can be ignored. By this assumption, the ordinate values of corners on the ground can be assigned in advance.

After the three assumptions above, the issue of texture mapping can be converted into the issue of reconizing the heights of buildings and the abscissas of vertical contours.

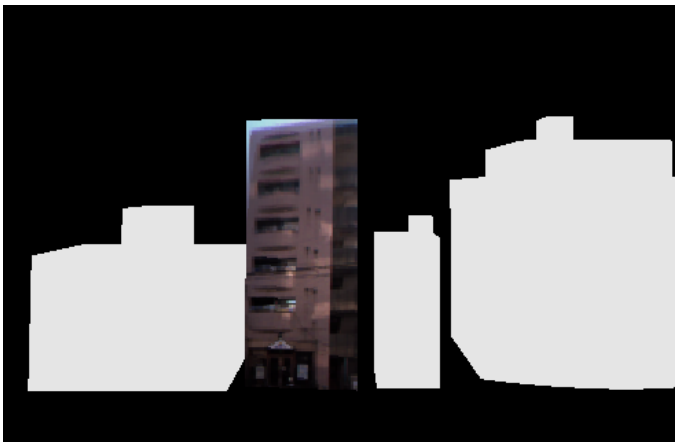
Since the heights of buildings can be easily decided from THI by the feature of THI, then we will mainly explain the step of recognizing the position of vertical contours by THI.

4.2 Recognition of Vertical Contours

As show in Fig. 4.2(a), actually, the vertical contours of buildings are expressed as band contours on THI. So by extracting band contours on THI, we can determine



(a) Corresponding corners between image and model.



(b) Txture mapping result by the four corresponding corners.

Figure 4.3: Process of texture mapping for one building.

the position of vertical contours.

We divide this process into two steps:

1. Extract band contours as parallel lines.

We extract the contours by edge extraction, but as shown in Fig. 4.2(a), the bands are overlapping each other in the surrounding field of THI, therefore, we can not extract peripheral contours exactly. To reduce the effect caused by the side faces of buildings, we just employ the edges in the central narrow field of THI (black field). However it is a dilemma that by cutting off most parts of THI,there is too little information left for edge detection. So we add a constraint:

Assuming all the buildings are on the same depth (means the distance between

vehicle and buildings is kept constant), all the bands can be considered to be parallel, and the band contours become a cluster of parallel lines. After extracting these contours as straight lines, and computing the average slope of these lines, we can modify these original lines to be parallel lines by the average slope(Fig. 4.2(c), red lines).

2. Estimate the Abscissas of Vertical Contours.

After extracting the band contours, the widths of buildings is just the horizontal distance between two approximate contours(Fig. 4.2(d), blue lines), though there are many choices to decide which frame should be used. To obtain the best texture of building facade, we have to determine an appropriate frame, on which building is just reflected on the central field. This can be achieved by choosing the frame, on which the center of width(Fig. 4.2(d), blue line) is just on the horizontal center of THI. Then, the abscissas of vertical contours can be recognized as the tips of the blue lines.

However, take notice that there may be some bands that do not mean buildings, we can check them by DP matching result. After obtain the abscissas of vertical contrours, we can decide the position of corners on the boundary of buildings and ground, since the ordinates of these two corners has been assigned.

Then, for the other two corners on the roof, the ordinates equal the height of building. As the feature of THI, the brightness is in proportion to the height of buildings, so average brightness of band means average height of the building in temporal axis. This way can recognize height stably.

By the two steps above, we can obtain the positions of the four coners from single image. As shown in Fig. 4.3, extract the field of the rectangle, and attach it onto the corresponding model (obtained by DP matching result), we can obtain the texture mapping result.

(a) Texture input 1.

(b) Texture input 2.

(c) Texture input 3.

(d) Texture input 4.

Figure 4.4: Some examples of texture input for scene 1.

4.3 The Results of Texture Mapping

We carried out texture mapping in two scenes: scene 1 (Fig. 3.6(a), around school) and scene 3 (Fig. 3.6(b), shinjuku).

4.3.1 Texture Mapping Result of Scene 1

We first extracted building corners from Image-THI (Fig. 3.8(d)), Fig. 4.4 showed some examples of frames as texture input, the field of marked rectangles were the extracted facades‘ textures. According to these texture inputs, we could say most of the buildings were well extracted, though in Fig. 4.4(b), the right contour was not well recognized because the two approximate buildings were too close to judge the contour on Image-THI exactly. For the building roof, as we applied average height, some unflat roofs were ignored.

Fig. 4.5 showed texture mapping result. Fig. 4.5(a) showed those extracted texture inputs, Fig. 4.5(b) showed the 3D building models for texture mapping, and Fig. 4.5(c) showed the final result of texture mapping.

4.3.2 Texture Mapping Result of Scene 3

Next, we carried out texture mapping in shinjuku (Fig. 3.6(b)). Fig. 4.6 showed some examples of extracted building textures (buildings surrounded with red rectangle). And Fig. 4.7 showed the texture mapping result of shinjuku. As shown in Fig. 4.7(a), building texture were well extracted, because the empty space between buildings were large enough and the roofs were all flat.

4.4 Estimation of Vehicle Location

Generally, we use GPS to get where we are, however, in some areas with tall buildings, the signal from satellite may not arrive. Here, we challanged estimating vehicle location only by image processing.

However, this is just a basic research, so we just try locating the image sequence (where did we take this pictures) captured for texture mapping. In fact, this process has became very easy since obtaining corresponding building edges between image and model by DP matching and similar as the process of texture mapping.

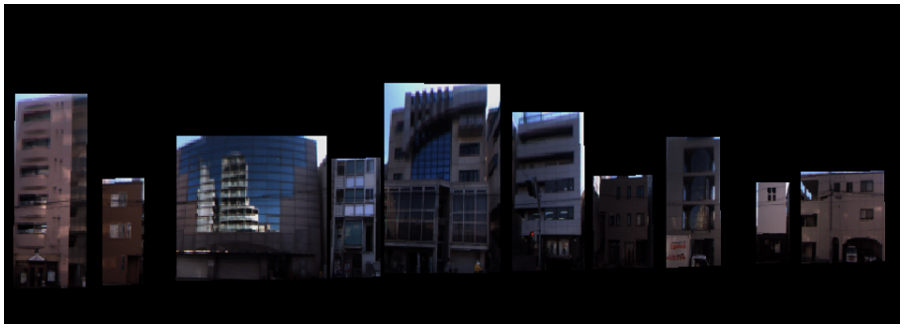
We first select frames from image sequence, as Fig. 4.8(a), in which vertical

contour of building is just on the central line. This process is similar as the process in texture mapping.

As shown in Fig. 4.8(b), the green line is defined as central line, and red lines are band contours which mean vertical contours of buildings in real image. Therefore, in the frame (blue line) through intersection point (red one) of central line and band contour, the vertical contour is just on the central line (Fig. 4.8(a)).

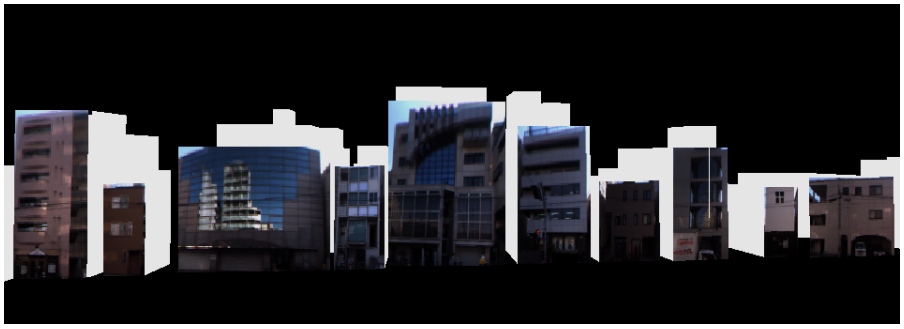
At this time, as we define on-vehicle camera captures buildings head-on, it can be considered that vehicle is just on the front face of the vertical contour. And as shown in Fig. 4.1, after DP matching, we can get corresponding relation of vertical contours between image and model, it means that we can find out the corresponding image of Fig. 4.8(a) in the image sequence captured by virtual camera in 3D map. Of course, the location where virtual camera took this picture can be decided. And it is just the same as the location of real camera in the real world. By this way, the location where we captured this picture can be known (Fig. 4.9). This is just the way to locate the frames in which vertical contour is on the central vertical line.

By the way, how about those pictures in which vertical contour can not be found on the central line? As we constraint the vehicle to keep constant velocity on a straight trajectory, the trajactory of pictures can be estimated lineraly.



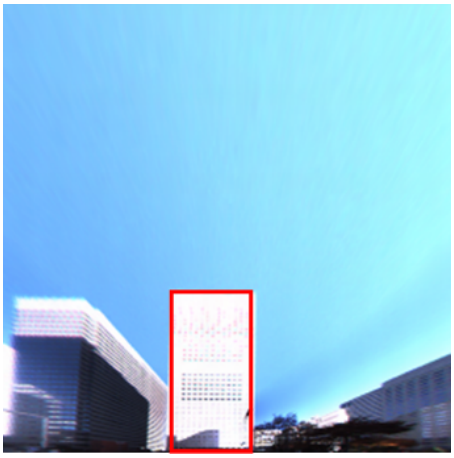
(a) Extracted textures.

(b) Building models for texture mapping.



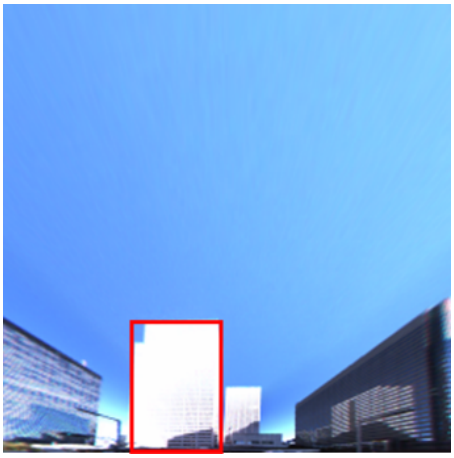
(c) Texture mapping result.

Figure 4.5: Texture mapping reuslt of scene 1.

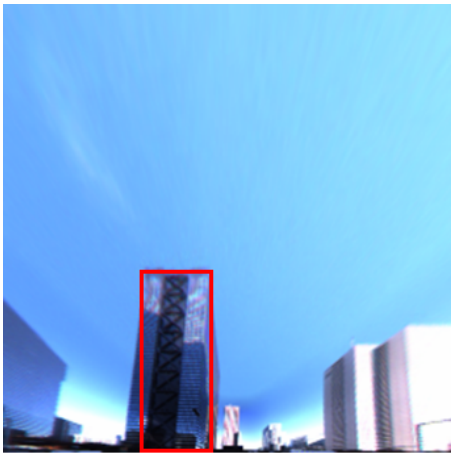


(a) Texture input 1.

(b) Texture input 2.

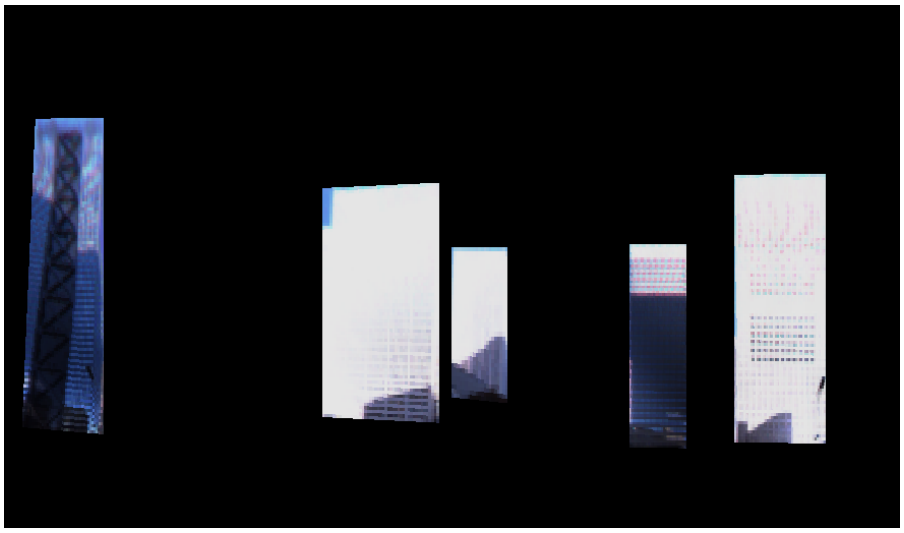


(c) Texture input 3.

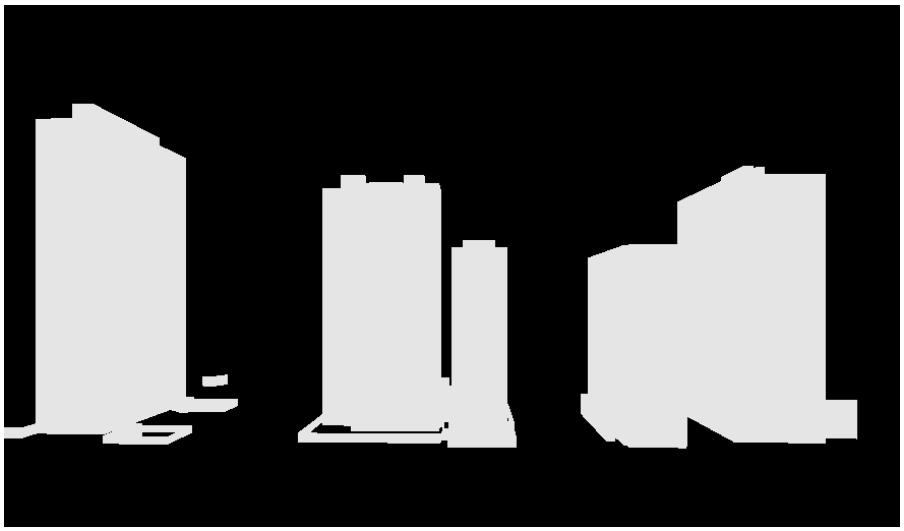


(d) Texture input 4.

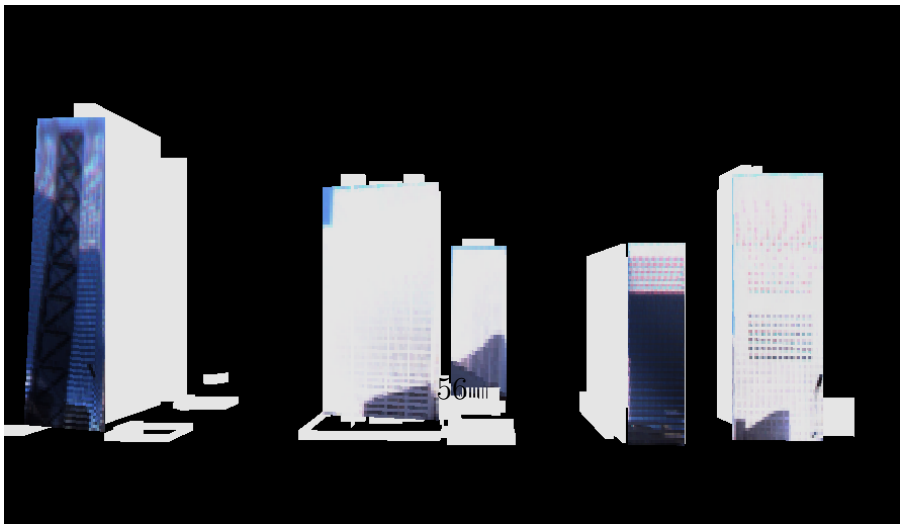
Figure 4.6: Some examples of texture input for scene 3.



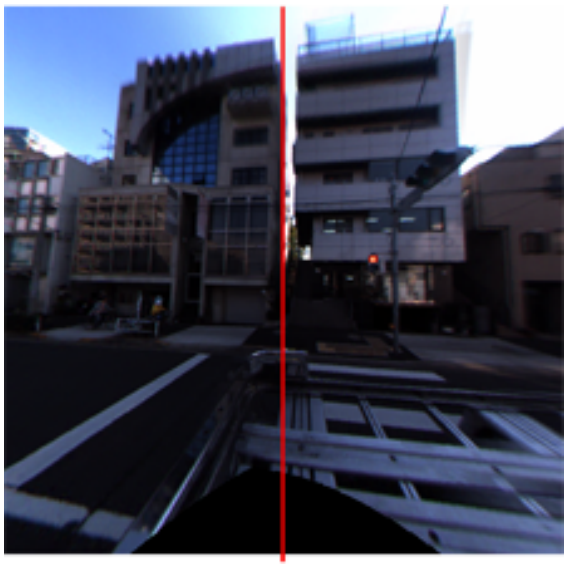
(a) Extracted textures.



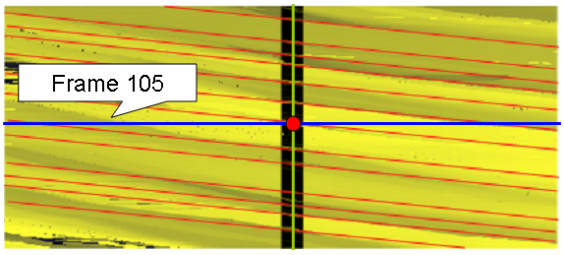
(b) Building models for texture mapping.



(c) Texture mapping result.



(a) Corresponding frame (No. 105).



(b) Searching one frame in which vertical contour is on the central line.

Figure 4.8: Process of searching frame in which wertical contour is on the central line.

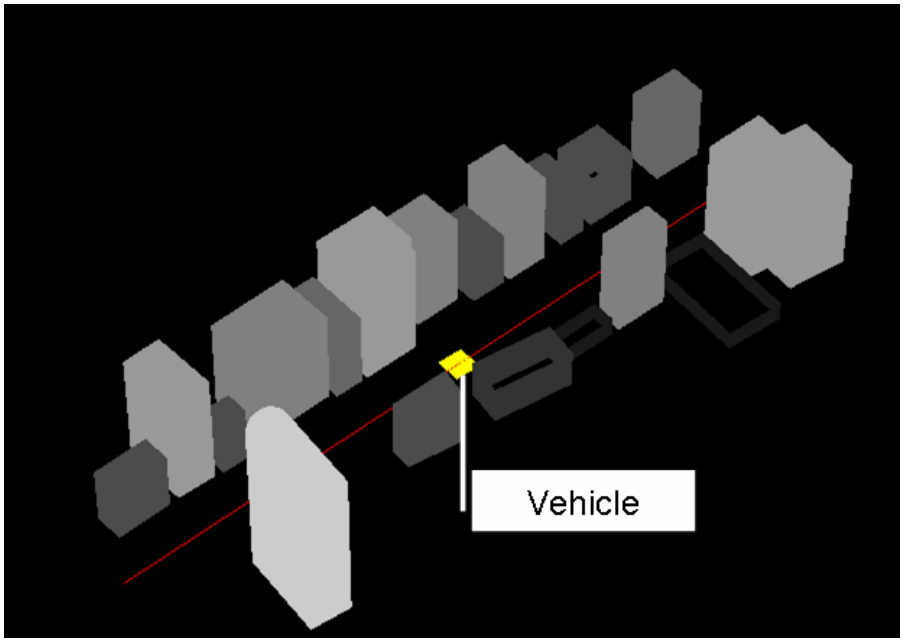


Figure 4.9: The result of location estimation for Frame 105.

Chapter 5

Conclusions

5.1 Main Achievements

In this paper, we proposed a automated texture mapping method onto 3D building models, it is hoped to reduce the labor hours of making textured 3D house map.

In this process, we also presented a novel expression of Spatio-Temporal Volume, called THI, it can be considered as an extended version of EPI, however, can overcome the shortages of EPI, for example, THI can remove the effect of edges caused by inside windows, and from THI both width and height of buidlings can be obtained, while on EPI, only width can be got. Using THI, we could recognize buildings stably.

Moreover, when creating THI, building height became necessary. We consider the highest edges as height, whereas, electric wires were annoying. To remove these electric wires, we suggested a wire-delete filter, based on median filter, but a vertical 1D one. It worked well in most scenes.

After recognizing buildings (bands on THI) from images and models, we matched these two kinds of bands by DP matching. It was designed to apply the feature of THI. In DP matching, we suggested several original passable paths to skip noises or integrate two approximate bands which mean the same buildings essentially, and added several constraints. Through several real scenes, our proposed DP matching can be considered as a robust matching method.

Deciding the corresponding buildings between images and models, texture mapping could be carried out. In this process, the key issue was to recognize four corners while we approached building facade as a rectangle. By assuming that buildings were reflected vertically on images and boundary of building and ground kept the same height, the four corners became easy to be recognized from THI. We showed two scenes of texture mapping. All the building front faces were well textured, although boundaries were a little coarse, since some building facades were too complex to approach as a rectangle simply.

At last, as a obiter application, using the result of corresponding buildings between images and models, we challenged estimating the vehicle location where to acquire the image. we succeeded in estimating locations on the straight trajectory.

5.2 Future Works

We are going to improve the accuracy of DP matching and texture mapping, then develop the automated texture mapping application to texture mapping commercially available 3D city maps. there are several issues to address in detail:

1. Currently, we just use buildings on one side of street for DP matching, we are going to design a DP matching using buildings on both sides. It is hoped to improve the accuracy of DP matching.

2. Now, we just consider the building facade as a rectangle simply, which is not enough for complex-shape buildings. Supposing the facade as a rectangle initally, then the process of changing the shape of rectangle to fit complex surface of building is required.

3. Now, only front faces of buildings are textured, we are going to texture both visible sides of buildings to improve the actuality.

References

[1] "Google Earth", http://earth.google.com/

[2] "ZENLIN", http://www.zenrin.co.jp/

[3] "WalkeyeMap", http://www.geogiken.co.jp/english/index.html

[4] "UC-win/Road", http://www.forum8.co.jp/english/

[5] "AERO ASAHI CORPORATION", http://www.aeroasahi.co.jp

[6] Wuhrer, S., Atanossov, R., Shu, C: "Fully Automatic Texture Mapping for Image-Based Modeling", NRC/ERB-1141. August 2006

[7] Y. Duan: "Topology Adaptive Deformable Models for Visual Computing", PhD thesis, State Univeristy of New York, 2003

[8] C. Hernandez-Esteban: " Stereo and Silhouette Fusion for 3D Object Modeling from Uncalibrated Images Under Circular Motion", PhD thesis, cole Nationale Suprieure des Tlcommunications, Paris, 2004

[9] Claudio Rocchini, Paolo Cignoni, Claudio Montani, and Roberto Scopigno: "Acquiring, stitching and blending diffuse appearance attributes on 3d models", The Visual Computer, 18(3) : 186–204, 2002

[10] Frueh, C., Sammon, R., Zakhor, A.: "Automated texture mapping of 3D city models with oblique aerial imagery", 3DPVT, 396–403, 6-9 Sept. 2004

[11] Brenner C., Haala N., and Fritsch D.: "Towards fully automated 3D city model generation", Workshop on Automatic Extraction of Man-Made Objects from Aerial and Space Images III, 2001

[12] Lee S. C., Jung S. K., and Nevatia R.: “Automatic pose estimation of complex 3D building models”, Workshop on Application of Computer Vision, 2002

[13] R. C. Bolles and H. H. Baker: “Epipolar-Plane Image Analysis: A technique for analyzing motion sequences”, Technical Note 459, AI Center, SRI International, 333 Ravenswood Ave. Menlo Park, CA94025, Feb. 1986

[14] H. Kawasaki, T. Yatabe, K. Ikeuchi, and M. Sakauchi, “Construction of a 3D City Map using EPI Analysis and DP Matching,” ACCV, vol. 2, pp. 1149–1155, 2000

[15] S. Uchida: “DP Matching : Fundamentals and Applications”, Technical report of IEICE. PRMU Vol.106, No.428(20061207) pp. 31-36 PRMU2006-166

[16] Y. Caspi and M. Irani: “A step Towards Sequence-to-Sequence Alignment”, CVPR, 13-15 June 2000, USA

[17] A. Rav-Acha , Y. Pritch , D. Lischinski and S. Peleg: “Dynamosaicing: Mosaicing of Dynamic Scenes”, PAMI, VOL. 29, NO. 10, Oct, 2007

[18] A. Rav-Acha, Y. Pritch, D. Lischinski and S. Peleg: “Spatio-Temporal Video Warping”, SIGGRAPH sketch, Aug. 2005

[19] M. Blank , L. Gorelick , E. Shechtman , M. Irani and R. Basri: “Actions as Space-Time Shapes”, ICCV, Oct. 2005, Beijing

[20] “LADYBUG ” http://www.viewplus.co.jp/products/ladybug2/index.html

[21] 佐藤准嗣，高橋友和，井手一郎，村瀬洋, “GPS 座標付き全方位映像群からの市街地映像マップの構築と町並変化の検出,” 電子情報通信学会論文誌, vol. J90-D, no. 4, pp. 1085–1095, Apr. 2007.

[22] トウ利洪，小野晋太郎，影沢政隆，池内克史, “距離センサを利用した住宅地図の 3 次元化とリファインメント,” 第 5 回 ITS シンポジウム, Dec. 2006.

[23] 渋久奈保, 佐藤准嗣, 高橋友和, 井手一郎, 村瀬 洋, 小島祥子, 高橋 新, “レーザレーダデータ間の対応付けによる自車位置情報の精度向上の検討,” 電子情

報通信学会総合大会講演論文集, D-12-50, vol. 情報・システム, pp. 182, Mar. 2006.