On-Vehicle Videos Localization using Geometric and Spatio-temporal Information

Kazuma Fukumoto, Hiroshi Kawasaki
Kagoshima University
1–21–40 Korimoto, Kagoshima City, Japan

Shintaro Ono
The University of Tokyo
4–6–1 Komaba, Meguro-ku, Tokyo, Japan

Hiroshi Koyasu
Saitama University
255 Shimo-Ohkubo, Sakura-ku, Saitama City, Japan

Katsushi Ikeuchi
The University of Tokyo
4–6–1 Komaba, Meguro-ku, Tokyo, Japan

ABSTRACT

Recently, a number of researches are conducted to construct the actual city into computers for the purpose of web services, ITS, disaster analysis, landscape simulations and so on. Further, with the spread of on-vehicle video cameras, it becomes common to share the on-vehicle video on website. If locations of the videos are available, the data can be efficiently used for virtual city construction. In this paper, we propose a method to realize localization of anonymous on-vehicle videos uploaded on the web by using video matching technique with Temporal Height Image (THI), Affine SIFT and Bag of Feature (BoF). THI retains information of relative building heights from temporal image sequences and the Affine SIFT realizes a robust matching for variance of both camera speed and driving lane. Finally, BoF representation allows us to realize a stable matching with less computational cost. We conducted several experiments using real image sequences of the actual city to show the successful results of the proposed method.

Keywords: *Scene localization, Temporal Height Image, Bag of Feature, Affine SIFT*

1 INTRODUCTION

Recently, many researches are conducted to construct the actual city into computers, such as Google Earth, Virtual World or Google Street View. Currently, those researches are mainly used for web services, however, they have a great potential for other practical purposes, such as ITS, disaster analysis, landscape simulations and so on. One drawback of those researches is that, since they basically assume special probe cars for data acquisition, frequent updates of database are limited only in urban areas.

On the other hand, installation of on-vehicle cameras are significantly increasing for the purpose of recording car accidents and/or intelligent driving support systems. With the spread of on-vehicle cameras for general cars, some drivers start to upload their on-vehicle videos to the web to share their driving experience and the number of those videos are surely increasing. In other words, every car in the city could be a probe car, if we can efficiently use its data. However, there is one critical issue to use those database for analysis or ITS applications, *i.e.*, there is no guarantee that all the data have geographical information like GPS, and there is no standard method to determine where the video was captured.

In this research, we propose a method to identify locations of on-vehicle video data captured by anonymous users by extending a video search technique. The method consists of two parts, the first is a learning part using video data which already have geographical information and the second is a video search part to find the video which includes the same scene of our target video from the database.

At the learning part, we first make a special data structure, called Temporal Height Image (THI) [17], which efficiently represents a spatio-temporal information of our target, and then, extract a feature from the data which is invariant to speed and directional change of the car. At the video search part, we use the Bag of Feature (BoF) [6] which is known to realize a robust and efficient search from huge database.

In addition, we propose a method to drastically increase both the precision and recall rates by using omni-directional images obtained by probe cars. Although the proposed matching is robust for the camera speed variance and the driving lane change, its quality is degraded in the case where the car moving direction is opposite. The combination of common on-vehicle cameras and omni-directional cameras is a solution to realize the matching between such opposite direction videos.

The rest of this paper is structured as follows. In Section 2, related works including vision-based localization in robotics and image matching of outdoor scene in computer vision are described in detail. The overview algorithm of the proposed method is described in Section 3. Then the proposed feature extraction method and the video retrieval method are described in Section 4. The experimental results, including the effectiveness of our THI correction and localization results by using videos taken from various on-vehicle cameras, are shown in Section 5. Finally, we conclude the paper in Section 6.

2 RELATED WORKS

Google Street view [1] provides images gathered by people, cars and bicyclists equipped with omni-directional cameras and GPS. Google Earth [2] and Bing Maps for Enterprise [3] provide 3D information of the city reconstructed by images taken from the sky. As these methods require special equipments for data, therefore areas whose data are updated frequently are limited only in urban areas.

Another research of on-vehicle camera is self-localization proposed by Murase [12]. In this method, a range data map, which provides the location information and range data of a target area, is constructed beforehand and estimation of the location of vehicle is done by DP matching between the observed range data sequence and the range data map. Range data observation is done by a laser radar. However, the laser radar is not a common device, and thus, it is difficult to use such a special device for general cars.

In the field of Robotics, many methods called vision-based localization which estimates robot positions by using images have been proposed. Commins and Newman proposed a method to localize the robot by the BoF frame work [7]. The method can deal with a large scale environment, however, it does not consider the variations of the appearance caused by the change of seasons or time of day. To deal with such variations, Miura and Yamamoto proposed a method which matches regions representing building, trees and so on [10]. The region extraction method is based on support vector machine (SVM), so the method is robust for such appearance variations. However, the matching by an image pair is generally unstable. Therefore they apply Markov localization frame work to deal with this matching ambiguity, but not using temporal image sequences.

The method proposed by Ono *et al.* reconstructs 3D city models by images taken by multiple probe cars [14]. In the method, epipolar plane image (EPI) and the THI; both are kinds of temporal image sequence; are used to estimate the location of a probe car. Continuous DP matching is conducted to evaluate matching scores, therefore, the method can estimate the location robustly despite the variation of car speed. However, the computation time of the DP matching rapidly increases as the size of database increases. Moreover, the variations of ap-

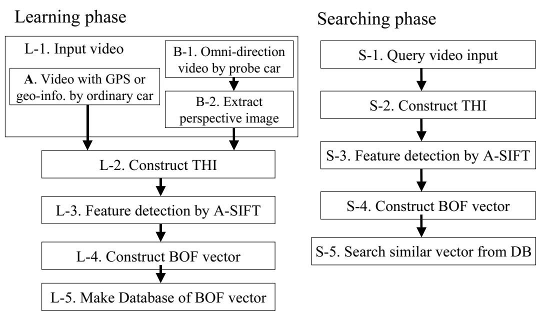

Learning phase
L-1. Input video
A. Video with GPS or geo-info. by ordinary car
B-1. Omni-direction video by probe car
B-2. Extract perspective image
L-2. Construct THI
L-3. Feature detection by A-SIFT
L-4. Construct BOF vector
L-5. Make Database of BOF vector
Searching phase
S-1. Query video input
S-2. Construct THI
S-3. Feature detection by A-SIFT
S-4. Construct BOF vector
S-5. Search similar vector from DB

Figure 1: Algorithm overview

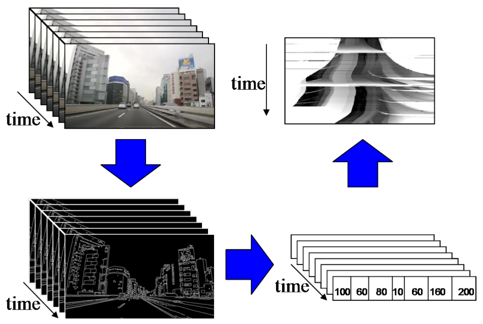


Figure 2: Construction step of THI.

pearance caused by weather and seasons degrade the estimation results, which are acquired by texture information on EPI.

Other researches which estimate the camera position in an urban scene have been proposed. Baatz *et al.* [5] proposed a localization method for smart phones. Their method deals with images taken by different cameras. In this method, Vanishing Point (VP) is extracted from the obtained image then walls of building in the image are projected onto the same plane to reduce variations of appearance caused by the view points. After the conversion, similar images are searched by using BoF algorithm. Murase *et al.* [13] proposed a similar method. In their method, an obtained image is matched with images taken from the sky, which contain large road regions, so they convert the obtained image to a bird-eye view image by detecting road regions in the image. Sivic *et al.* [15] proposed a method for the same purpose. In their method SIFT features are extracted from the obtained image, then they search for the similar image by the method which is an extension of the document search to the image search. However, these methods are based on single image matching, and not by using temporal information.

3 OVERVIEW OF THE TECHNIQUE

Our purpose is to find the actual place of the scene which was captured by on-vehicle video camera without using any special sensors such as GPS. The method consists of two phases, the first one is a learning phase and the other is a searching phase as shown in Fig. 1. The learning phase consists of five steps. In step L-1, video data which have geographic information are used as an input. The data include the videos which are captured by a probe car. Since an omni-directional camera is common for a probe car, *i.e.*, Google car for Street-view, we also assume that we have a certain amount of omni-directional videos for learning phase. Then, we construct THI (step L-2) and extract features from THIs using A-SIFT [11](step L-3). Since the number of features is too large, we apply BoF which is known to be a typical solution for such a case to make a compact representation of data (step L-5).

At searching phase, we apply the same step for the learning phase to construct BoF vector (step S-4). Once we get the vector, we search the similar vector from the database to determine where the video was taken.

4 VIDEO LOCALIZATION BASED ON VIDEO SEARCH

4.1 Spatio-Temporal Data Structure

First, we construct the spatio-temporal data to extract the feature which efficiently represents the scene captured by on-vehicle video camera. For the purpose, we adopt THI for a data structure. In the following, we describe what THI is, how to construct it, and an extended method to improve the quality of THI.

4.1.1 Construction of THI

Temporal Height Image (THI) is a spatio-temporal image representing the height of buildings. Each pixel value represents height of buildings of given video sequence. The coordination of THI is same as other temporal image representation such as epipolar plane image (EPI); y indicates frame number of the video and x is same as x-coordinate of the input video. Since, a pixel intensity along a row of THI represents height information of the corresponding frame by a gray scale value, THI represents the change of the height information of the buildings. Fig. 2 shows how to make a THI. First, a silhouette of the buildings is detected by image processing, and then, the heights of the buildings are estimated from the silhouette. Then, the height information are converted to gray scale value and a single row is constructed for each frame. Finally, all the rows from consecutive frames are accumulated to create THI. At the image retrieval using THI,we found that if the same building shows up only a few frames, it is difficult to construct the THI. In experiment therefore, we only to use videos which sufficiently satisfied the shutter speed and frame rate.

4.1.2 Construction of THI from Omni-directional Image

Although the proposed method is robust for the camera speed and the driving lane, it cannot deal with the opposite direction of the car motion, because appearances of both THIs are totally different. Omni-directional camera which can observe $360°$ view surroundings can be a solution for the problem. Since the omni-directional image can be easily converted into a perspective image viewed from the arbitrary camera position to any view direction, two THIs for both car directions, one views forward and the other views opposite, can be constructed. By using those two THIs, one of them always matches to either direction of the car motion.

4.1.3 Correction of THI

For construction of THI, there are mainly two issues remaining. The first one is a wide variety of obstacles observed from the street, such as closer electric cable to further clouds in the sky. The second problem is a small vibration of the car.

In terms of the first issue, we use the method proposed by Wang [17] to remove the electric cables in the sky. The method is based on a median filter using a special shape of kernel, such as 1*n, which has long and thin shape. Although the noise removal process using the median filter efficiently works on cables, it cannot remove large objects like clouds. For solution, Graph-Cut(GC) algorithm [18][8] which can efficiently segment the scene into a small number of elements, such as sky, wall of buildings, trees and roads can be used. One problem of the GC is that it is not easy to set an appropriate parameter for general case, *e.g.*, if we set a strong regularization term, small but important objects are all wiped out. On the other hand, if we set a small regularization term, small noises remain in the scene. Therefore, it is difficult to realize both removing clouds and keeping thin, but important objects in the scene, such as street lights; note that such an object like street lights makes a T-junction in THI and is an important feature for similarity retrieval step. In our method, we set a small regularization term and apply the opening of morphological filter for solution. With the method, some parts of a thin object are removed by GC, but are recovered in the next step.

Fig. 3 shows the cloud removing result. In the original captured image (a), the complicated textures of clouds are observed in a sky and simple edge detection cannot remove those clouds as shown in (b). However, after applying our method, the sky regions are segmented correctly whereas street light is also kept in the scene as shown in Fig. 3 (c). Fig. 4 shows THIs with and without the cloud removing method, In Fig. 4(a) THI has many noises, however, those are all removed in Fig. 4(b).

In terms of the second problem, *i.e.*, vibration of the car, typical solution is to install a shock absorber between the camera and the car. However, since we assume that the video is captured

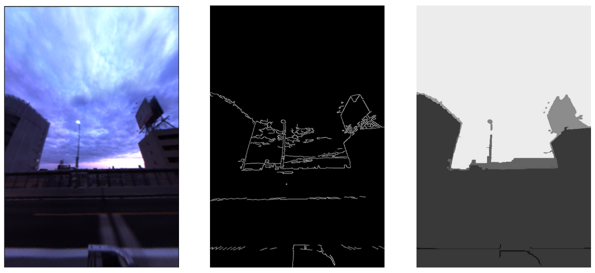

(a) Input image (b) Edge image (c) Graph cuts

Figure 3: Example image with cloud in the sky and removing result.

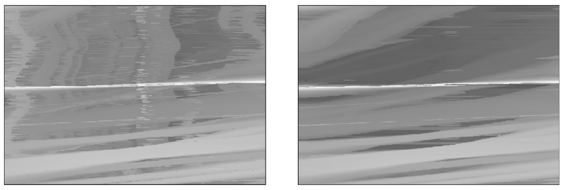

(a) Before (b) After

Figure 4: Effect of removing clouds to make THI.

by anonymous people and cannot expect to use such absorber for installation, we apply an image processing technique for solution. Because our target video is captured by an on-vehicle camera which is usually toward the moving direction, the video naturally includes a focus of expansion (FOE) in the scene. Therefore, we use FOE to detect a small vibration of the car. Since FOE is a point where optical flows are come out, the point can be extracted as the intersection point of flow vectors in the scene. Note that it is known that the optical flow is robustly extracted from videos, and thus, FOE can be stably detected. By using the FOE, we can adjust the hight of the building to refine the THI.

To estimate FOE consistent to temporal direction, we used a particle filter for FOE tracking in the paper. Although estimation of FOE is almost correct, small vibration still occurs because of miss-matched optical flows or wrong detection by surrounding cars. Those errors are efficiently modified by the temporal filter, especially the particle filter. In the method, each particle represents a position of FOE. If the line of an optical flow passes through a particle, the likelihood of the particle increases. By maximizing the likelihood, temporally consistent FOE tracking is realized.

4.2 Feature Extraction by Affine SIFT

Since THI uses only the height information of the buildings, THIs are not affected by the texture of the buildings. As a result, THIs are robust for appearance variation caused by the weather or time of day. However, THIs have some variation even if cameras take same scene. The variation is caused by mainly two reason; one is variation of the camera speed and the other is variation of the driving lane. Due to this, another feature extraction method which is robust for such variation is needed.

In THI, the camera speed appears as the incline of the building's border. Thus it can handle by the 2D homographic transformation. The driving lane appears as pixel brightness since the appearance of the building height changes with the distance between the camera and the building. Thus it can handle by the illumination change. Therefore, we use Affine SIFT [11], which is a feature descriptor that can handle both the illumination change and the affine transformation.

Fig. 5 shows the results of extracted Affine SIFT features from THIs. In the figures, three THIs and a real scene corresponding to the THI are shown. Points on each THI represent extracted positions of the Affine SIFT features. Since an original number of points is too large for subsquent processes, we conduct clustering by k-means algorithm for aggregation. Points in same cluster are drawn by the same color in the figures. We can see that the most feature points extracted by Affine SIFT appear in the borders between buildings (the areas where intensity difference is large) and near thin structures such as street lamps in the scene (near the areas where thin lines exist). The intersection of the boundaries of those two areas is called T-junction.

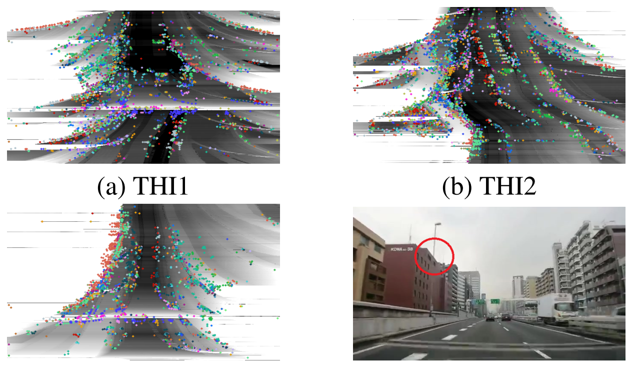

(a) THI1 (b) THI2 (c) THI3 (d) Sample frame of the scene (c)

Figure 5: Example of extracted features using Affine SIFT on THI.

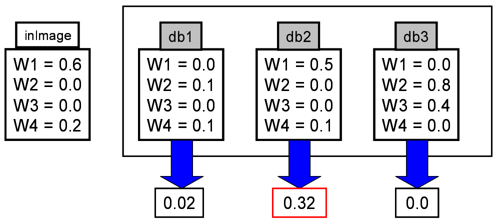


Figure 6: Image retrieval by using BoF

4.3 Similarity Scene Detection using BoF

For image retrieval, we use Bag of Feature (BoF) [6], which is a well-known image retrieval method for web images. In the BoF, each image is represented by the histogram of typical features in the image called visual words, and then, the similarity search of the image is done by the histogram matching between the images in the database and the input image. We apply the BoF to search a similar THI, then the location of input video is determined by the search result.

For implementation, BoF needs clustering of the feature vectors. We use hierarchical k-means algorithm for clustering. Then, we use the center from each cluster to decide delegate feature, called visual words. Since the number of clusters is equal to the number of visual words, the number is an important parameter of BoF's recognition. Although the large number of visual words is preferable for robust recognition, it requires a large memory and computational cost. For solution, we randomly sample the features. If the ratio of the sampled feature is too low, the recognition results may be degraded. In our method, we set the sampling ratio to be 20% by evaluating the real data; details are described in Section 5.4.

After generating visual words, a normalized histogram of the visual words is calculated for each THI. We apply approximate nearest neighbor (ANN) [9] for calculation. For matching, we use *tf-idf* weighting method [16]. The weight of a visual word v_i in an image j, is denoted as w_{ij}, is calculated as follows:

$$w_{ij} = \frac{tf_{ij}}{\sum_i tf_{ij}} \log \frac{N}{df_{ij}}, \tag{1}$$

where tf_{ij} represents the histogram value of v_i in the image j, df_j represents the number of images including features belonging to v_i, and N represents total number of images. The w_{ij} is the bin value of the histogram.

For retrieval when a query video sequence is given, the video is converted into THI, Affine SIFT features are extracted from the THI, and the histogram of the visual words weights is calculated. Then similarity score is calculated by the inner product of this histogram pair. By calculating all the similarity between the database, the video sequence which has the highest similarity score is selected as the corresponding scene.

5 EXPERIMENTS AND EVALUATIONS

5.1 Experiment

We conducted experiments to confirm the effectiveness of the method. We searched the video with keyword "Tokyo car mounted video" from YouTube [4] and downloaded several car-mounted videos as the input. We also captured the video using the probe car as shown in

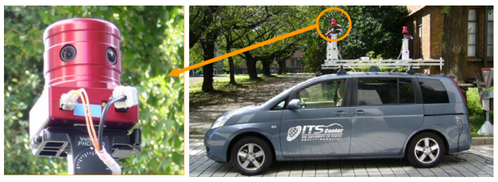


Figure 7: Our probe car with an omni-directional camera.

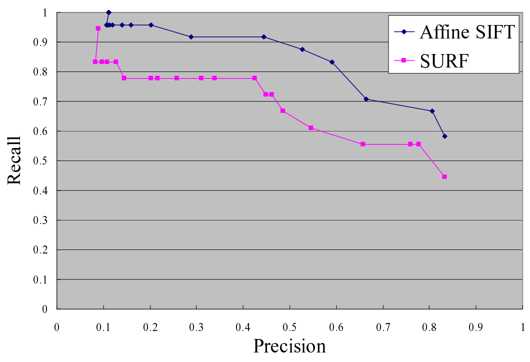
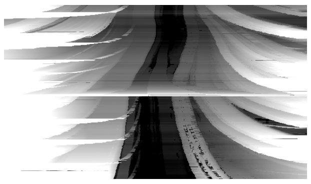
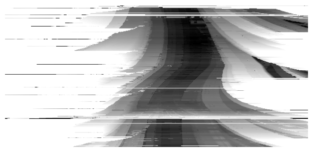


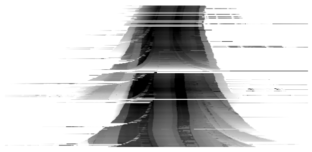
Figure 8: Recall and precision rate using Affine SIFT and SURF.



(a) Input THI

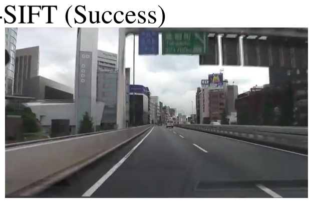


(c) Search result from DB using A-SIFT (Success)

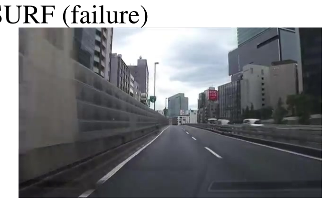


(e) Search result from DB using SURF (failure)

(b) Sample frame of the scene (a)



(f) Sample frame of the scene (e)



(d) Sample frame of the scene (c)

Figure 9: Example of input and search results using Affine SIFT and SURF.

Fig. 7 (right). The car was equipped with the omni-directional camera as shown in Fig. 7 (left) and could capture 360 degree environment with 30 fps.

5.2 Comparison between Affine-SIFT and SURF

As explained in the Sec. 4.2, THIs are always distorted by affine transformation at each video. Affine SIFT for feature detection is our solution. To confirm the effectiveness using Affine SIFT, we compare the result with SURF which cannot cope with affine transformation.

Fig. 8 shows the recall and precision rate of both methods. We can clearly see that the results using Affine SIFT are always better than thats of SURF. The left column of Fig. 9 left column shows an example of an input THI for query (a), result of Affine SIFT (c) and result of SURF (e). With the figure, although the input THI and the searched THI with Affine SIFT are not globally similar, their local features look similar and this is the reason why correct THI was selected with our method. On the other hand, wrong THI was selected with SURF approach where appearance of THI does not look like the input. The right column of Fig. 9 right column shows the example frames of each scene. We can confirm that the same scene of input (b) is extracted by using our Affine SIFT in (d) even if the camera and the position of the car are totally different.

5.3 Effect with the Correction of THI

To confirm the effectiveness of our THI correction using FOE explained in Sec. 4.1.3, we compared the result with and without THI correction. Fig. 10 shows the recall and precision rate

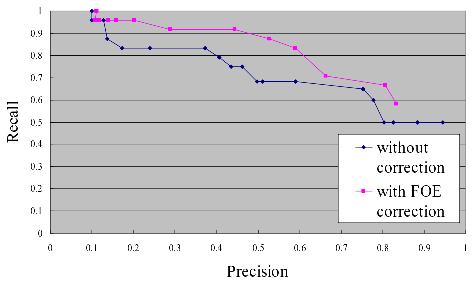


Figure 10: Effect of THI refinement using extracted FOE.

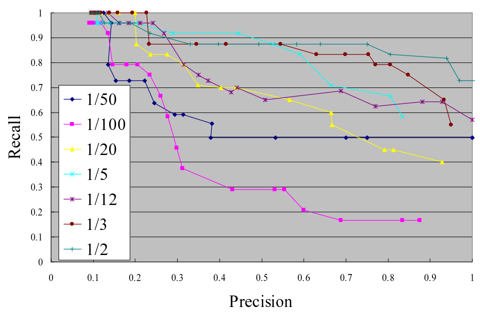


Figure 11: Effect on random sampling.

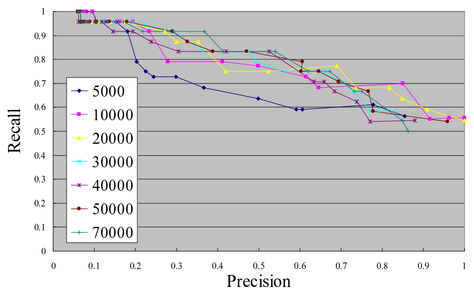


Figure 12: Effect on the number of visual words.

of both methods. We can clearly see that the results with THI correction improve the results.

5.4 Effect of the Number of Features on BoF

We use BoF for recognition part in the paper. Since BoF is based on the idea that a large number of simple features has a good potential for recognition, the number of features is an important factor to define the ability of the method. Since clustering technique, which has basically high computational cost, is required in the process of BoF, it is not easy to increase the number of feature points simply. Therefore, we tested the method with respect to the number of the features on BoF with the following two approaches. (1): We randomly sample the feature points from every THI to reduce the number of feature points. (2): We reduce the number of visual words, but do not change the number of the feature points. The purpose of the second evaluation is to check the effect when a bin size of histogram is enlarged; large bin size can drastically reduce the computational cost for clustering.

Fig. 11 shows the effect of (1). We can clearly see that drastic reduction severely degrade the results, however, if there is a certain amount of data, quality of results does not change. Fig. 12 shows the effect of (2). We can see that the number of visual words does not affect the quality of the result much. From those two results, it suggests that it is possible to reduce the number of both feature points and visual words by some threshold to make the low computational cost keeping the quality of the result.

5.5 Effect of using Omni-Directional Camera for the Probe Car

The video captured by a probe car is expected to improve the results. In the paper, we propose a method to use an omni-directional camera for the probe car to further improve the results. To confirm the effectiveness of our method, we compared the results of 2 databases constructed by the omni-directional video; one using only one converted perspective video (DB1) and the other using both the perspective and the opposite (DB2). Three query videos were downloaded from YouTube. All videos captured the same scene from the same route, but their directions are

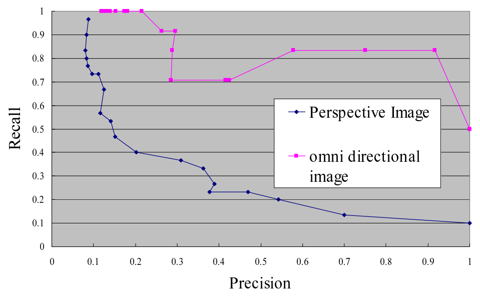


Figure 13: Effect on using omni directional camera.

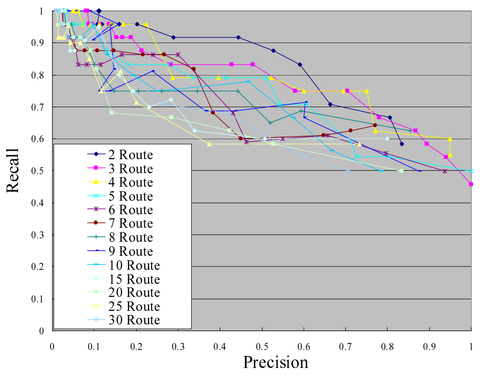
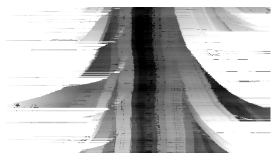
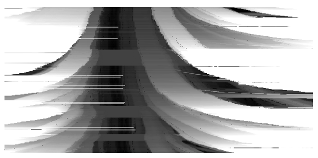


Figure 14: Effect on changing the database size.



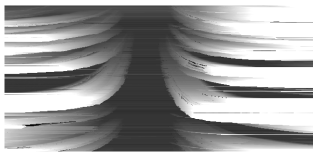
(a) Input THI created by YouTube video



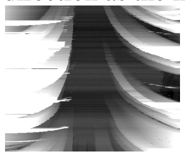
(b) Search result of THI from Omni-directional camera (Same direction as the input)

(c) Sample frame of the scene (a)

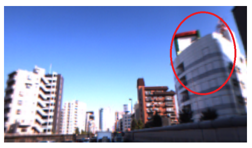
(d) Sample frame of the scene (b)



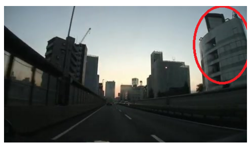
(e) THI of Omni-directional camera with the same frame of (b), but opposite direction



(f) Search result THI corresponding to (e)



(g) Sample frame of the scene (e)



(h) Sample frame of the scene (f)

Figure 15: Since direction of the car of (a) and (f) is opposite, THI's appearances are totally different and cannot be searched directly. However, using omni-directional camera, those two THIs are successfully connected. As the same building is depicted by red circles in all frames, we can confirm that all the images are capturing the same scene.

different.

Fig. 13 shows the recall and precision rate of the results. In Fig. 13, "Perspective Image" denotes the results using the DB1, and "omni-directional image" denotes those using the DB2. Since the shapes of THI created from the videos capturing the same scene along the same road but from opposite directions are totally different, it is almost impossible to find correct match between them. However, it is drastically improved by adding the video captured by omni-directional camera as shown Fig. 13. Fig. 15 shows example THIs and scenes of this experiment. Fig. 15(a) shows a query THI, and Fig. 15(c) is an example frame of the sequence constructing (a). Fig. 15(b) is the matched THI with (a) in DB2, and Fig. 15(d) is an example frame of (b). Fig. 15(e) is the THI of same scene as (b) but using opposite directional video, and Fig. 15(g) is an example frame of (e). Fig. 15(f) is matched THI with (a) in DB1, and Fig. 15(h) is example frame of (f).

The right column of Fig. 15 shows the example frames extracted from each THI. We can confirm that the same scene of input (c) was successfully extracted from the omni-directional camera (d) and ordinary camera (h) which was captured from the car moving toward the opposite direction.

5.6 Large Database Test for Scalability Evaluation

Finally, we carried out the experiment to verify the scalability of our method by changing the size of database. We used ten video sequences downloaded from YouTube. Fig. 14 shows the recall and precision rate of each experiment changing a number of routes from two to ten. We can see that even if we increase the database size, recall and precision rate keep almost the same accuracy with similar tendency, and thus, we can expect that our method properly works with larger databases.

6 CONCLUSION

In the paper, we proposed a method to realize the matching of videos which capture the same scene from car-mounted cameras without using any sensors, such as GPS. In order to achieve the robust matching with enough scalability, our method uses THI and Affine SIFT for feature detection and BoF for search. In the experiments, we confirmed effectiveness of our method by using real data that were downloaded from YouTube and captured by the probe car. In the future, we plan to use larger database downloaded from Internet, to connect them together to construct a large and dense image database over the world.

References

[1] Google Street View : http://maps.google.com/help/maps/streetview/.

[2] GoogleEarth : http://www.google.com/earth/index.html.

[3] Bing Maps for Enterprise : http://www.microsoft.com/maps/.

[4] YouTube : http://www.youtube.com/.

[5] Georges Baatz, Kevin Köser, David Chen, Radek Grzeszczuk, and Marc Pollefeys. Leveraging 3d city models for rotation invariant place-of-interest recognition. Vol. 96, , 2012.

[6] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cedric Bray. Visual categorization with bags of keypoints. In In Workshop on Statistical Learning in Computer Vision, ECCV, pp. 1–22, 2004.

[7] M. Cummins and P. Newman. Invited Applications Paper FAB-MAP: Appearance-Based Place Recognition and Mapping using a Learned Visual Vocabulary Model. In 27th Intl Conf. on Machine Learning (ICML2010), 2010.

[8] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. Int. J. Comput. Vision, Vol. 59, No. 2, pp. 167–181, September 2004.

[9] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In Proceedings of the thirtieth annual ACM symposium on Theory of computing, STOC '98, pp. 604–613, New York, NY, USA, 1998. ACM.

[10] Jun Miura and Koshiro Yamamoto. Robust view matching-based markov localization in outdoor environments. In IROS, pp. 2970–2976. IEEE, 2008.

[11] Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. SIAM J. Img. Sci., Vol. 2, No. 2, pp. 438–469, April 2009.

[12] Hiroshi Murase. Recognition of vehicle camera video. 4th Spoken Document Processing WS, Vol. 6, , 2010.

[13] Noda, Takahashi, Deguchi, IDE, Murase, Kojima, and Naitou. Ego-localization by sequantial matching of road-surface in aerial image and in-vehicle camera images(generic object recognition image feature quantity). PRMU, Pattern Recognition Media Understanding, Vol. 109, No. 306, pp. 177–182, nov 2009.

[14] Shintaro Ono, Ryota Matsuhisa, Katsushi Ikeuchi, and Hiroshi Kawasaki. Vehicle localization using space-time matching of on-vehicle video. In 9th ITS symposium 2010, 2010.

[15] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the International Conference on Computer Vision, Vol. 2, pp. 1470–1477, October 2003.

[16] Karen Spärck Jones. A statistical interpretation of term specificity and its application in retrieval. 1972.

[17] Jinge Wang, Shintaro Ono, and Katsushi Ikeuchi. Matching on-vehicle image and building model by temporal height image and proposal of texture mapping. Technical Report of IEICE. IE, Image engineering, Vol. 108, No. 425, pp. 103–108, 2009-01-28.

[18] Jianxiong Xiao and Long Quan. Multiple view semantic segmentation for street view images. In ICCV, pp. 686–693. IEEE, 2009.