Sensors and Materials, Vol. 32, No. 2 (2020) 611–623
MYU Tokyo

S & M 2126

Image-retrieval Method Using Gradient Dilation Images for Cloud-based Positioning System with 3D Wireframe Map

Junji Takahashi,[1]* Kawabe Masato,[1] Seiya Ito,[2] Naoshi Kaneko,[2]
Wataro Takahashi,[2] Toshiki Sakamoto,[3] Akihiro Shibata,[3] and Yong Yu[1]

[1]Department of Mechanical Engineering, Graduate School of Science and Engineering, Kagoshima University, 1-21-40, Korimoto, Kagoshima, Kagoshima 890-0065, Japan
[2]Department of Integrated Information Technology, Graduate School of Science and Engineering, Aoyama Gakuin University, 5-10-1 Fuchinobe, Chuo, Sagamihara, Kanagawa 252-5258, Japan
[3]Department of Architecture & Architectural Engineering, Graduate School of Science and Engineering, Kagoshima University, 1-21-40, Korimoto, Kagoshima, Kagoshima 890-0065, Japan

(Received November 1, 2019; accepted December 9, 2019)

Keywords: indoor localization, 3D wireframe map, cloud computing, gradient dilation image, parallel computing

We propose a novel cloud-based precise positioning system that uses visual sensing data. Any mobile module with a vision sensor and wireless communication can be a client and can receive benefit from this system. When the client module takes a picture of an environment and uploads it to the server, it receives the shooting position with 6 degrees of freedom (DoFs) with an accuracy on the order of centimeters within a couple of seconds. The server maintains a map of the environment and localizes the uploaded picture in the map. The contributions of this paper are threefold. First, we develop a new visual localization method using a 3D wireframe map. The method proceeds in three steps: (i) the generation of an arbitrary perspective 2D image composed of line segments from a 3D wireframe map, (ii) the gradient dilation of a line segment image for effective image retrieval, (iii) pixelwise-AND-based image-similarity evaluation by parallel computing. Second, we build 3D CAD models of an actual building from a 2D design drawing and with manual measurements. Third, we experimentally evaluate our method using virtual sensing data.

1. Introduction

The Internet of Things (IoT) is the network of physical objects embedded with sensors, actuators, software, and connectivity that enables these objects to be connected and to exchange data. The IoT is a highly versatile technology that is expected to pervade many situations and environments, such as a home and a public space, by connecting resources and demands to supply efficient services.(1) Moreover, the location information of resources and demands, which indicates the position information of objects related to the service, is very important in supplying a physical service.(2) However, the positioning technology suited for the IoT context

*Corresponding author: e-mail: takahashi@mech.kagoshima-u.ac.jp
https://doi.org/10.18494/SAM.2020.2678

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

has not developed sufficiently. The requirements for IoT positioning are precision necessary and sufficient for applications, response speed suited to applications, long-term stability, low initial cost, low maintenance cost, low running cost, and the universality of coping with various sensors. There are no technologies or studies to meet the above requirements at the same time.

The problem of positioning a mobile object has been studied in the robotics for a long time. When a robot needs to localize itself without knowledge of its environment, it must create a map of the environment at the same time. Dissanayake *et al.* formulated this problem as simultaneous localization and mapping (SLAM) and Thrun *et al.* organized solutions for SLAM.(3,4) Then, many SLAM-based localization solutions using a camera and a light-detection-and-ranging sensor (LiDAR) have been proposed and developed, and have recently become a key technology of self-driving vehicles.(5,6) However, the large computational load and weight of sensors could be a bottleneck to applying sensors to small objects in IoT applications.

An intelligent environment approach, in which an environment is customized by embedding with physical equipment, has also long been in development. In automated factories and large warehouses, automated guided vehicles (AGVs) had been introduced early.(7) First, physical guides made from metal were adopted, and they subsequently become a wire and magnetic and visual types, so as to reduce the cost of resetting. As a visual marker, since QR codes are easy to install and are able to provide precise positioning, many researchers are studying for various applications in human living areas as well as at production sites.(8–10) Regardless of the easy installation, the maintenance cost cannot be cut, making it difficult for the QR code approach to contribute to the IoT scenario. The method using Wi-Fi signals surrounding a mobile object does not require the maintenance of devices physically.(11) However, the mapping process is needed on a periodic basis, but the accuracy is low. The maintenance and running costs and low accuracy are fundamental concerns in the intelligent environment approach for pervading human living areas.

For realizing a positioning system with minimal maintenance and running costs as well as low computational burden on mobile objects, we have proposed a cloud-based positioning infrastructure system named Universal Map (UMap).(12) The merits of the UMap are that it does not require any physical infrastructure in the environment, and the requirements of sensor and computational resources on the client are minimal. The main research issues are how to prepare a map on the server and how to localize a query data from a client. These problems have been studied in the field of computer vision. Visual localization in large-scale environments is often dealt with as an image retrieval problem. In urban environments, a geotagged image database (DB) is prepared beforehand, and the query image location is matched with the geotag information of the most similar image retrieved from the DB.(13–15) However, the main demerit of this method is that it predicts only an approximate location of the query, not an accurate 6-degree-of-freedom (DoF) position. Another approach is to predict the 6-DoF camera pose with respect to a pre-constructed 3D map. The map usually consists of a 3D point cloud built via the Structure-from-Motion (SfM) method(16,17) or using data measured with a red-green-blue-depth (RGBD) sensor.(18) The query pose is predicted by feature matching and solving a Perspective-n-Point (PnP) problem. The demerit is that the map built at a time will become unusable after a certain degree of environmental change has occurred.

In contrast to the above 3D map, the UMap is composed of 3D wireframes and surfaces transformed from an architectural 3D CAD model. Once the 3D wireframe map is constructed, it can be useful permanently unless the building suffers from damage. The query image is converted to a line segment image and the camera position is predicted by retrieving the most similar line segment image in the DB generated from the 3D model. In this paper, we propose a new method of using gradient dilation images for efficient retrieval. The blurred lines make the retrieval process robust to pixel gaps in the image caused by camera position gaps. Because of this effect, the prediction accuracy is expected to improve and the grid interval between DB images can be wide so that a smaller DB can be organized.

Our contributions are threefold. First, we develop a generator for arbitrary perspective 2D line segment images to organize an image DB. Second, we develop an algorithm of gradient dilation transform to make a blurred line segment image. Third, we develop a pixelwise-AND-based similarity-evaluation algorithm working on a graphic board for parallel computing. All methods are validated by detailed experiments.

2. Materials and Methods

2.1 Overview of cloud-based positioning system

The system overview of the cloud-based positioning system named UMap is drawn in Fig. 1. The UMap consists of three subsystems: a central server that maintains a 3D wireframe map; clients who access the server to obtain their own positions; agents who detect and report environmental changes to the server. The client usually uploads the newest sensing data, then the server localizes the sensing data in the 3D wireframe map. Finally, the localization result is downloaded to the client.

The UMap has been developed multidirectionally. Various types of data, such as a standard camera image,(12) an omnidirectional camera image,(19) and 3D line segment data from an RGBD sensor(20) and a LiDAR sensor,(21) are confirmed to be query data for the UMap. To improve performance, the restructuring of the DB(22) and an investigation of the allowance

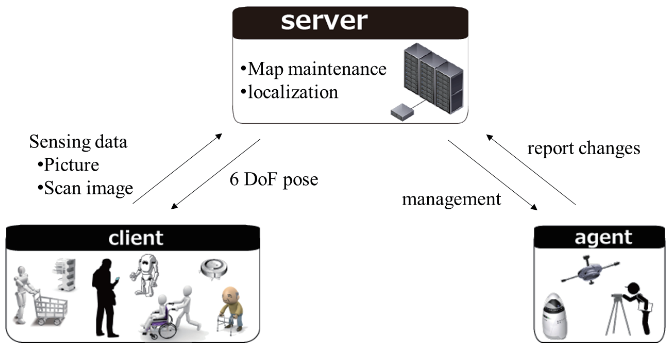


Fig. 1. (Color online) Overview of the cloud-based positioning system (UMap).

error between a map and an actual environment[23] have been performed. Moreover, the agent part has been developed where the structure edge from the 3D wireframe model and color edges detected from the borders of a poster are integrated into the hybrid map to cope with environmental changes.[24]

In this paper, we deal with the server-client part of the UMap. The workflow of the proposed method is drawn in Fig. 2. A sensing data, which is an image taken by client's camera, is sent to the server. The server receives the image and detects line segments. Then, the line segment image is used as a query for the retrieval process. The image DB is created by a DB image generator (DBIG) beforehand. The DBIG reads a 3D CAD model and generates an arbitral projection image. The details of each process are described in each subsection.

2.2 Sensing and inquiry processes on the client side

A problem of posing a rigid body in 3D space generally has 6 DoFs. The UMap basically can cope with a 6-DoF positioning problem. On the other hand, the geometrical condition of constraint depending on the application reduces the DoF. In this study, we assumed a client module set on a cart, as shown in Fig. 3(a). In this case, the UMap deals with a 3-DoF posing problem: predicting values for the x-axis, y-axis, and θ angle (horizontal angle) when the z-axis, vertical angle, and roll angle are constant.

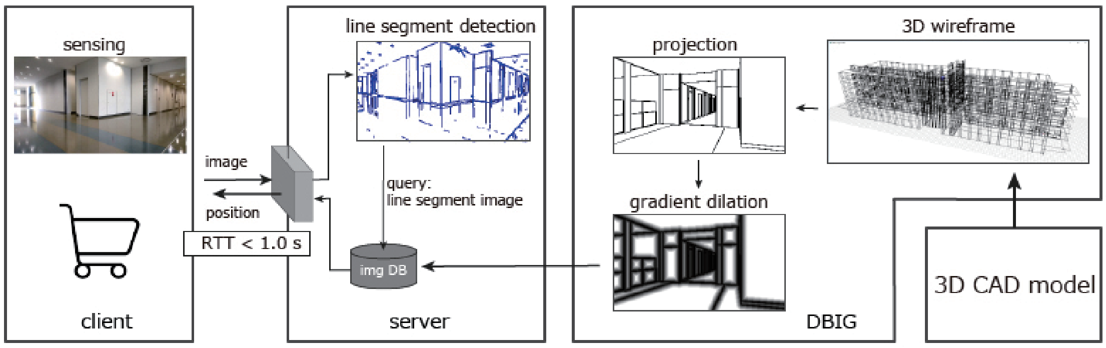


Fig. 2. (Color online) Workflow of the proposed method.

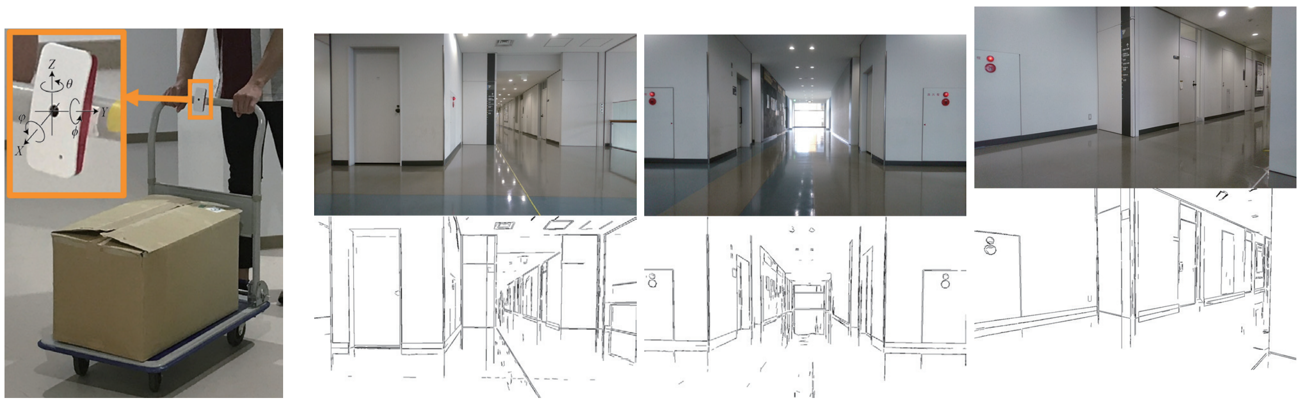


Fig. 3. (Color online) (a) Example of cart and embedded client module. (b) Sample images taken by a camera and line segment images.

2.3 Pre-processing to the sensing image for retrieval process on the server side

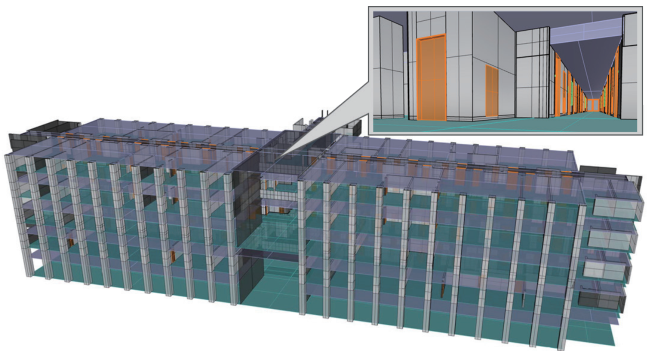
The server executes a line segment detection process to the uploaded camera image immediately. Any line detection algorithms can be used for this process, such as a Hough transform, a canny method, and a line segment detector. On the basis of our pilot experiment, we found that the line segment detector was best suited for our system. Examples of a sensing image and their line segment images are shown in Fig. 3(b). Each line segment image is resized to the DB image before the retrieval process.

2.4 3D CAD models for prior map

We built 3D CAD models of an actual building on the basis of their 2D design drawings and manual measurements. During the construction of CAD models, we noted that there are unignorable differences between the 2D design drawing and the actual building. For example, we noted that a door in the actual building does not exist in the drawing and the end of the wall is short compared with the drawing. To correct these differences, the manual measurement process was used. In Fig. 4, the 3D CAD models constructed are shown.

2.5 DB of 2D images with correct position information

For query image localization, the important features of the 3D CAD models are structural boundaries of the building between wall and wall, ceiling and wall, floor and wall, door and wall, and window and wall. These boundaries are projected and drawn as line segments in an image when the viewpoint and view direction with several camera projection parameters are given. The viewpoint and view direction represent the 6-DoF position in the 3D-map coordinate system. Therefore, the problem of query image localization is converted to the problem of searching for the reasonable viewpoint and view direction for projecting the line segment image, which is similar to a query image. Moreover, the problem of searching a view-


(a)

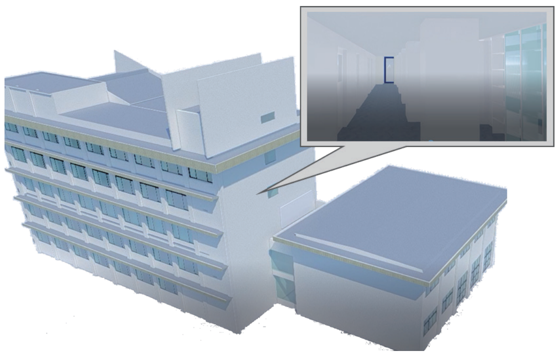

(b)

Fig. 4. (Color online) (a) 3D CAD model of O-building in Aoyama Gakuin University (AGU) (total floor area: approx. 11000). (b) 3D CAD model of 2nd building of Dept. of Architecture (total floor area: approx. 3000).

point direction is the same as retrieving the best similar image in an image DB consisting of many drawn images with various viewpoint directions.

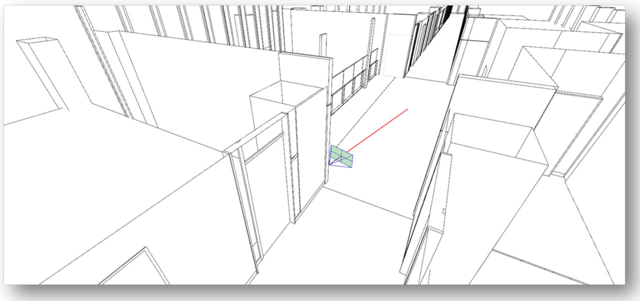
We developed a DBIG for the efficient drawing of line segment images with specified camera parameters. The DBIG is an application executed on Windows and based on the Open GL library. Table 1 shows the camera parameters for drawing an image. When these parameters are given, the DBIG can draw an image as if a camera has taken a picture in the 3D wireframe map. Moreover, the DBIG accepts various ranges of axes and grid intervals to automatically generate images for the DB. The required parameters for this process are described in Table 2. For example, in the case of $0 \leq x, y \leq 10$, $0 \leq \theta \leq 360$, $dx = dy = 0.1$, $z = 1.2$, $d\theta = 10$, and $\phi = \psi = 0$, 360000 images are generated. A DBIG scene that displays camera positions for the DB and a sample picture are shown in Fig. 5.

Table 1
Parameters for drawing an image using DBIG.

	Parameter	Unit	Description
Camera position and posture	x, y, z	m	Viewpoint in coordinate system of 3D wireframe map
	θ	degree	Horizontal view direction (yaw)
	ϕ	degree	Vertical view direction (pitch)
	ψ	degree	Roll direction
Camera specification	α	degree	Horizontal angle of view
	W_{img}	pixels	Width of the image
	H_{img}	pixels	Height of the image

Table 2
Parameters for generating a DB using DBIG.

Parameter	Unit	Description
$x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}$	m	Range of each axis
dx, dy, dz	m	Grid interval in each axis
$\theta_{min}, \theta_{max}, \phi_{min}, \phi_{max}$	degree	Range of each rotational axis
$d\theta, d\phi$	degree	Angle interval in each axis


(a)

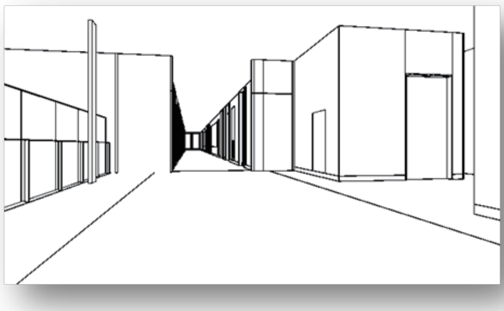

(b)

Fig. 5. (Color online) (a) DBIG shot of bird's-eye-view mode where a virtual camera is drawn in the center. (b) Picture taken by the virtual camera in the DBIG.

2.6 Gradient dilation transform for efficient similarity evaluation

The image retrieval process is conducted as a similarity evaluation. In our previous method,(12) the similarity between two line segment images was equal to the total number of surviving pixels, after pixelwise logical conjunction. However, this method is too sensitive to the position gap. When the shooting position of the query image moves slightly (for example 1.0 cm), the line segments in the image move more than 1 pixel. Even if the camera movement is small, in the images, the line segments that overlap each other do not overlap after the movement. This phenomenon possibly makes the DB image with the shooting position geometrically closest to the query image becomes lower in similarity than the other DB images.

To solve the above problem of hypersensitivity to viewpoint shifts, the blur process is used. We first applied the distance transform for the blur process and confirmed that the distance-transformed images are robust to viewpoint shifts.(25) However, the distance transform approach has difficulties in limiting the dilation width and in designing an arbitrary gradient. Therefore, we developed a new blur process named gradient dilation transform. We assume that a line segment image drawn in gray scale, where the color intensity of a pixel on line segments is 1 and that of a background pixel is 0. Let ${p_{i,j}}^{DB}$ denote the color intensity of a target pixel (i, j) and q the distance from the nearest pixel on line segments. Let q_w denote the width of dilation and p_{limit} the lower limit of color intensity. Then, the color intensity of each pixel in the transformed image is given as

$$p_{i,j}^{DB} = \begin{cases} 1 - \dfrac{1 - p_{limit}}{q_w} q & (q \le q_w) \\ 0 & (q > q_w). \end{cases} \tag{1}$$

Examples of gradient dilation images are shown in Fig. 6.

2.7 Pixelwise-AND-based similarity evaluation

The similarity evaluation between a query image and a DB image is conducted by pixelwise AND calculation. Let ${p_{i,j}}^{query}$, ${p_{i,j}}^{DBk}$, and ${p_{i,j}}^{AND}$ denote the color intensities of a target pixel (i, j) in the query, k-th DB, and resulting AND images, respectively; then, the resulting pixels are given as

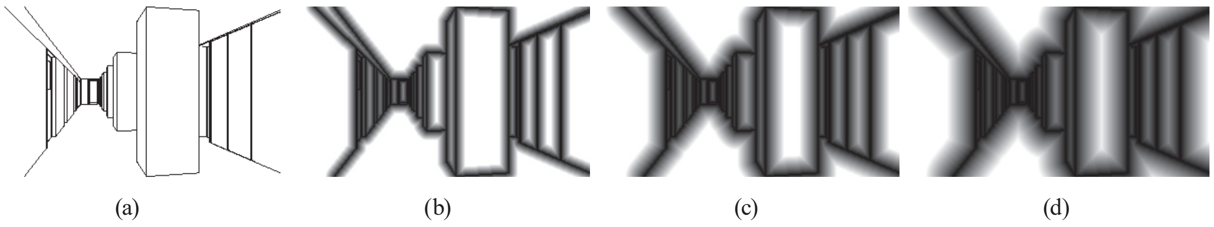

(a) (b) (c) (d)

Fig. 6. Examples of gradient dilation images: (a) $q_w = 0$, (b) $q_w = 10$, (c) $q_w = 20$, and (d) $q_w = 30$.

$$p_{i,j}^{AND} = p_{i,j}^{query} \cdot p_{i,j}^{DBk} . \tag{2}$$

Figure 7(a) shows an overlay image of a query image, a *k*-th DB image, and a resulting AND image. The pixels in the overlay image can be classified into 6 classes, A, B, C, D, E, and F, as shown in Fig. 7(b). Let num(*X*) represent the total number of member pixels in class *X* and int(*X*) the sum of the color intensities of member pixels in class *X*. Then, we define the similarity s_k between the query image and the *k*-th DB image as

$$s_k = \frac{\mathrm{int(B)} + \mathrm{int(C)}}{\mathrm{num(A)} + \mathrm{num(B)} + \mathrm{num(C)} + \mathrm{num(D)}} . \tag{3}$$

The image in the DB that has the maximum similarity is regarded as the best-matched image and its position is adopted in the prediction result of the positioning system.

3. Experimental Evaluation

3.1 Experimental environment and conditions

We conducted 2 types of experiment. In experiment I, we investigated the maximum performance of our method assuming sufficient computational resources. The validity of the similarity index was evaluated in this experiment. In experiment II, we investigated the practical performance. Every experiment was conducted in the environment of the 5th floor of the O-building in Aoyama Gakuin University (AGU) [Fig. 4(a)]. We used a smartphone device (Lenovo Phab 2 Pro) as the client module in both experiments. The camera parameters are $\alpha = 74.6$ degree, $W_{img} = 320$ pixel, and $H_{img} = 180$ pixel.

The performance of the positioning method was basically measured by the error distance between the predicted position and the groundtruth. Since our method can predict the view direction as well as the view position, we extended the error distance to the error norm. Let $\boldsymbol{q}$ and $\boldsymbol{b}_k$ denote the coordinate value vectors of the query and *k*-th DB images, respectively, then

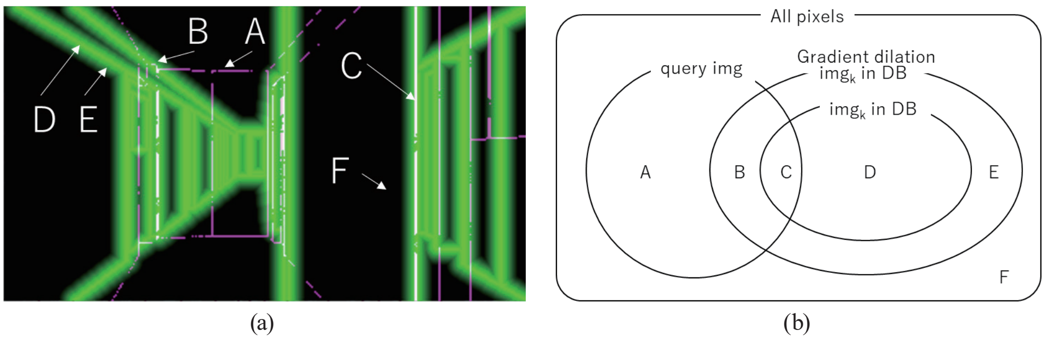


Fig. 7. (Color online) (a) Example of overlay image with pixel class indication where the line segments of the query, *k*-th DB, and resulting AND images are drawn in magenta, green, and white, respectively. (b) Venn diagram of pixel classes.

the error norm e_k is defined as

$$e_k = |\boldsymbol{b}_k - \boldsymbol{q}| \\ = \sqrt{(x_k - x_q)^2 + (y_k - y_q)^2 + (z_k - z_q)^2 + (\theta_k - \theta_q)^2 + (\phi_k - \phi_q)^2 + (\psi_k - \psi_q)^2}\ , \tag{4}$$

where the unit of x, y, and z is meter, and the unit of θ, ϕ, and ψ is radian. Note that we presume that an error of 0.1 rad (≈ 5.7 degree) equals an error of 0.1 m.

(1) Experiment I: In order to deeply investigate the relationship among the parameters of the DB, similarity index, and error norm, we limited the target area to 4 m^2 and conducted experiments with scrupulous configurations. We prepared 200 types of DB in total with various grid intervals, angle intervals, and gradient dilation widths. We also prepared 10 query images randomly. These query images were virtual sensing images. Then, the position of each query image was predicted by our proposed positioning method. The positions of virtual cameras for the virtual sensing image and several arrangements of virtual cameras for DB images are shown in Fig. 8.

(2) Experiment II: We used a standard PC (Core i7 3.7GHz, 32 GB RAM) with a graphic board (GTX 1080 Ti 11 GB VRAM) as a server machine covering the target area. To investigate the environmental characteristics, two areas were selected as experimental fields. Area A was a long corridor with an area of 70 m^2 that included highly symmetric and repetitive elements. Area B was a T-junction of the corridor that appeared asymmetric. With the size of a DB image assumed to be 320 × 180, the maximum number of DB images that can be loaded on the graphic board at one time was 100000. This limitation was due to the VRAM size of 11 GB and our implementation method. We prepared 6 types of DB for each area. The details are described in Table 3. For this experiment, 250 query images were prepared randomly for each area. The location of each area and virtual camera positions for virtual query images are shown in Fig. 9.

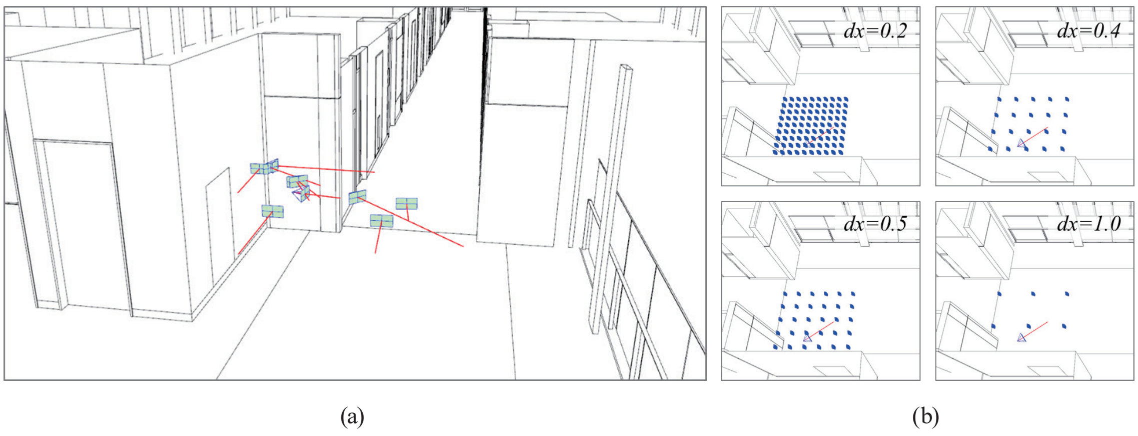


Fig. 8. (Color online) (a) Positions of virtual cameras for virtual sensing image and (b) arrangements of virtual cameras for DB images.

Table 3
Specifications of DBs (dz, $d\phi$, $d\psi$ = constant) for experiment II.

	Range	Grid interval $dx = dy$	Angle interval $d\theta$ (°)	GD width q_w pixels	Number of images
Area A	$-6 < x < 37.5$		2	0, 10	98100
	$0.2 < y < 1.8$	0.4	5	0, 10	39240
	Approx. 70 m^2		8	0, 10	24525
Area B	$-6 < x < 37.5$		2	0, 10	45540
	$0.2 < y < 1.8$	0.4	5	0, 10	18216
	36 m^2		8	0, 10	11385

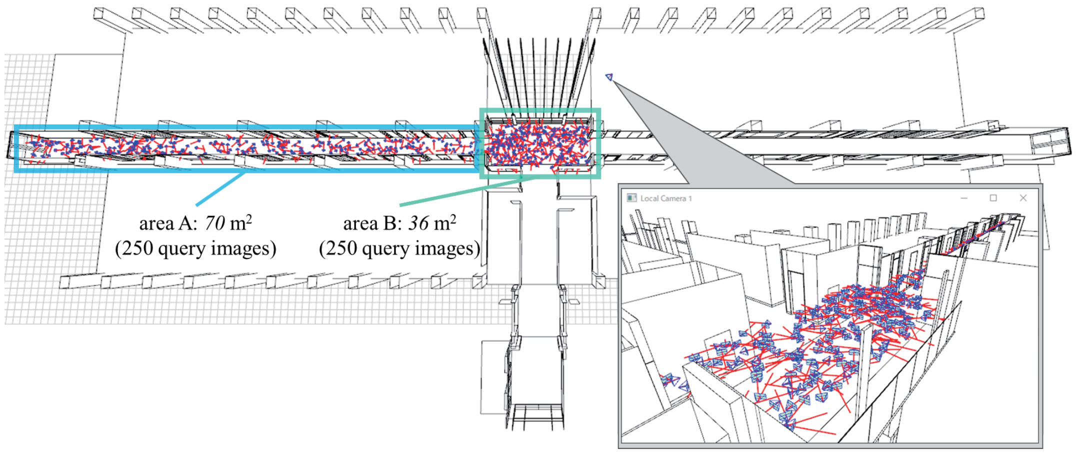


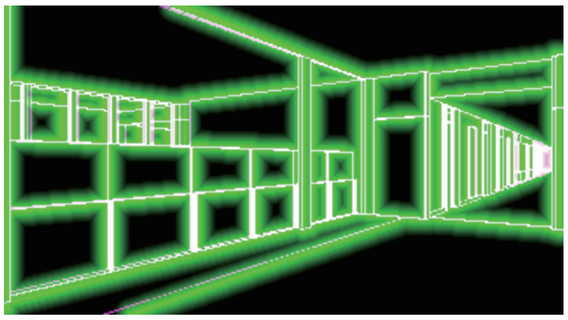
Fig. 9. (Color online) 5th floor of O-building in AGU including virtual cameras.

3.2 Results and discussion

(1) Experiment I: Figure 10 shows (a) a typical overlay image and (b) a graph of the similarity and the error norm versus all DB images when the position of the query image is predicted. It is confirmed that the predicted DB image, k = 595, with the maximum similarity is very close to the correct answer, k = 585, the error of which is minimum. Although the predicted number is not the optimum, the similarity curve appears to be smooth and unimodal, and it is expected that the method can predict a value close to the optimum. In other cases, the similarity graph tends to be unimodal. This finding supports the validity of the similarity index defined as Eq. (2).

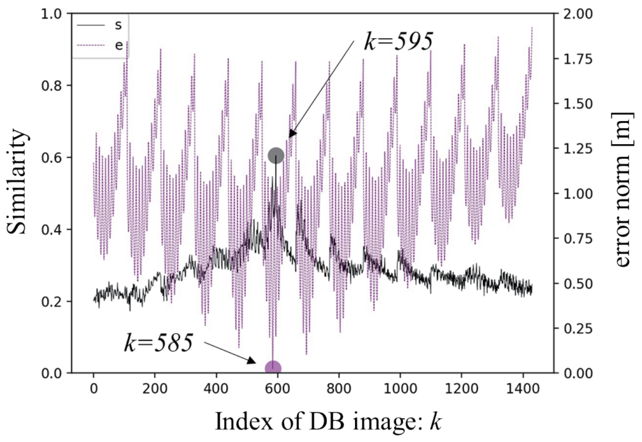
Table 4 shows the aggregate results chosen from all prediction experiments with every prepared query and DB. In the most accurate case, where $dx = dy = 0.1$, $d\theta = 1.0$, and q_w = 10 pixels, the average error norm is 0.075 m.

(2) Experiment II: Figures 11(a) and 11(b) show graphs of the cumulative frequency of prediction results. In the case of area A, the prediction accuracy is expectedly not very high. Because of the symmetric appearance and many repetitive elements, the similarity of an incorrect position DB image becomes incidentally higher than that of an adapting correct DB image. Although this problem is difficult to solve using only visual information, the consistency



$\boldsymbol{q} = (38.325, -0.430, 19.1, 36.1, 0.0)$
$\boldsymbol{b}_{595} = (38.300, -0.400, 19.1, 35.0, 0.0)$
$e_{595} = 0.204$ m

(a)

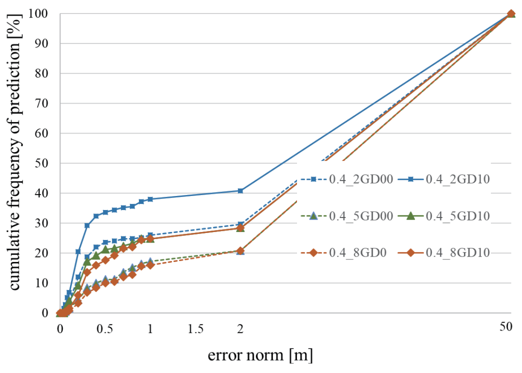


(b)

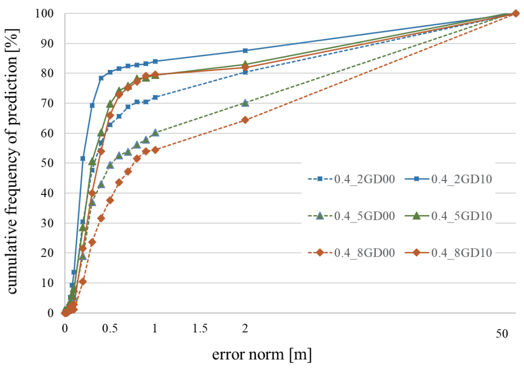
Fig. 10. (Color online) (a) Obtained overlay image and (b) graph of similarity and error versus all DB images.

Table 4
Average error norm (m) of each condition.

		$dx = dy$ (m)											
		0.1			0.2			0.4			0.5		
θ_w (pix)		0	10	20	0	10	20	0	10	20	0	10	20
	1	0.087	**0.075**	**0.075**	0.120	0.113	0.113	0.152	0.152	0.153	0.325	0.226	0.295
	3	0.189	0.155	0.160	0.201	0.161	0.327	0.310	0.191	0.267	0.480	0.222	0.471
$d\theta$ (°)	5	0.190	0.172	0.172	0.261	0.208	0.205	0.478	0.246	0.342	0.686	0.267	0.504
	7	0.341	0.314	0.327	0.346	0.347	0.365	0.459	0.452	0.399	0.550	0.451	0.383
	9	0.223	0.225	0.214	0.319	0.257	0.548	0.331	0.324	0.531	0.547	0.302	0.473



(a)



(b)

Fig. 11. (Color online) Prediction results in areas (a) A and (b) B.

of movement or another sensing modality, such as Wi-Fi signals, will lead to a solution. In the case of area B, it is confirmed that the parameter setting of the DB has a considerable effect on prediction accuracy. To improve the accuracy, the grid and angle intervals should be small, and the gradient dilation width should be set to 10 rather than 0. In the best case, where $dx = 0.4$, $d\theta = 2$, and $q_w = 10$, the rate of images predicted with error under 0.5 m is around 80%. One of the reasons why some images are matched with the wrong DB image is the miss-shooting of query images. Since the virtual sensing images are generated randomly, some images are not suited to positioning. For example, if the image is taken very near the front of a wall, the image tends to become all white without any line segments. Actually, the query image data set includes some all white or comparable images. In the practical case, multiple sensing can alleviate this problem.

The round-trip time from the time when the client uploads an image to the time when the client receives the predicted position information was below 1.0 s in all experiments. We confirmed that the system can be used in practical applications, owing to the use of a graphic board.

4. Conclusions

We proposed a cloud-based positioning system using a 3D wireframe model as a map. To localize the query image taken by the client in the 3D map, we developed a 2D image generator. This generator reads the 3D wireframe model and outputs arbitrary viewpoint images consisting of line segments efficiently to organize an image DB. The positioning problem is converted to an image-retrieval problem, that is, finding the most similar image to the query image from the DB. To enhance the image similarity evaluation process, we developed a new image blur method named gradient dilation transform, which is suited to blurred line segments with detailed tuning. We also developed a method of evaluating the similarity between two line segment images on the basis of pixelwise AND. This process can be implemented on a graphical processing unit with calculation by parallel computing.

We conducted two types of experiment and confirmed that the smallest average error is 0.075 m in an ideal setting. In the case of a T-junction of the corridor that appeared asymmetric, 80% of the query images are successfully predicted at an error less than 0.50 m; the round trip time is below 1.0 s in all experiments.

One of the topics for future study is evaluation with real sensing query. We confirmed, in a pilot experiment, that real sensing data could be predicted correctly like virtual sensing query. We will conduct the experiment as soon as the real sensing dataset is ready.

Acknowledgments

This work was supported in part by JSPS KAKENHI JP 16K21339 and SECOM Science and Technology Foundation.

References

1 A. A. Fuqaha, M.Guizani, M. Mohammadi, M. Alendhari, and M.Ayyash: IEEE Commun. Surv. Tutorials **17** (2015) 2347. https://doi.org/10.1109/COMST.2015.2444095
2 L. Chen, S. Thombre, K. Järvinen, E. S. Lohan, A. Alén-Savikko, H. Leppäkoski, M. Z. H. Bhuiyan, S. Bu-Pasha, G. N. Ferrara, S. Honkala, J. Lindqvist, L. Ruotsalainen, P. Korpisaari, and H. Kuusniemi: IEEE Access **5** (2017) 8956. https://doi.org/10.1109/ACCESS.2017.2695525
3 M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba: IEEE Trans. Rob. Autom. **17** (2001) 229. https://doi.org/10.1109/70.938381

4 S. Thrun, W. Burgard, and D. Fox: Probabilistic Robotics (MIT Press, Cambridge, 2005) p. 309.
5 C. Cadena, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, L. Reid, L. Carlone, and J. Leonard: IEEE Trans. Rob. **32** (2016) 1309. http://hdl.handle.net/1721.1/107697
6 G. Bresson, Z. Alsayed, L. Yu, and S. Glaser: IEEE Trans. Intell. Veh. **2** (2017) 194. https://doi.org/10.1109/TIV.2017.2749181
7 T. L. Anh and M. B. M. De Koster: Eur. J. Oper. Res. **171** (2006) 1. https://doi.org/10.1016/j.ejor.2005.01.036
8 H. Zhang, C. Zhang, W. Yang, and C. Y. Chen: Proc. 2015 IEEE Int. Conf. Robotics and Biomimetics (IEEE ROBIO, 2015). https://doi.org/10.1109/ROBIO.2015.7419715
9 M. Nowicki, M. Rostkowska, and P. Skrzypczyński: Proc. 2016 IEEE Algorithms, Architectures, Arrangements, and Applications (IEEE SPA, 2016). https://doi.org/10.1109/SPA.2016.7763605
10 P. Nazemzadeh, D. Fontanelli, D. Macii, and L. Palopoli: IEEE/ASME Trans. Mechatron. **22** (2017) 2588. https://doi.org/10.1109/TMECH.2017.2762598
11 C. Chen, Y. Chen, Y. Han, H. Lai, and K. J. R. Liu: IEEE IoT J. **4** (2017) 111. https://doi.org/10.1109/JIOT.2016.2628701
12 J. Takahashi: Proc. 3rd Int. Workshop on Smart Sensing System (2018).
13 R. Arandjelović and A. Zisserman: Proc. Asian Conf. Computer Vision (Springer ACCV, 2014) 188. https://doi.org/10.1007/978-3-319-16817-3_13
14 A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla: Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (IEEE CVPR, 2015). https://doi.org/10.1109/CVPR.2015.7298790
15 T. Sattler, B. Leibe, and L. Kobbelt: Proc. European Conf. Computer Vision (ECCV, 2012) 752. https://doi.org/10.1007/978-3-642-33718-5_54
16 S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski: Commun. ACM **54** (2011) 105. https://doi.org/10.1145/2001269.2001293
17 S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt: Proc. European Conf. Computer Vision (ECCV, 2014) 268. https://doi.org/10.1007/978-3-319-10605-2_18
18 H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii: Proc IEEE Int. Conf. Computer Vision and Pattern Recognition (IEEE CVPR, 2018). https://arxiv.org/abs/1803.10368
19 Y. Kusuno, J. Takahashi, and Y. Yong: Proc. IEEE/SICE Int. Sympo. System Integration (IEEE SII, 2019). https://doi.org/10.1109/SII.2019.8700393
20 N. Kaneko, J. Takahashi, and T. Yoshida: Proc. LBRP Poster IEEE IROS (2016).
21 R. Yamashita, J. Takahashi, and Y. Yong: Proc. SICE Symp. System Integration (2018) (in Japanese).
22 A. Akada, J. Takahashi, and Y. Yong: Proc. IEEE Int. Conf. Pervasive Computing and Communications Workshop (IEEE PerCom Workshops, 2019). https://doi.org/10.1109/PERCOMW.2019.8730768
23 K. Shimmura, J. Takahashi, and Y. Yong: Proc. SICE Symp. System Integration (2018) (in Japanese).
24 T. Kaneko, J. Takahashi, S. Ito, and Y. Tobe: SICE J. Control Meas. Syst. Integr. **12** (SICE JCMSI, 2019) 149. https://doi.org/10.9746/jcmsi.12.149
25 S. Ito, N. Kaneko, J. Takahashi, and K. Sumi: Proc. 2nd IEEE Int. Conf. Imaging, Vision & Pattern Recognition (IEEE icIVPR, 2018). https://doi.org/10.1109/ICIEV.2018.8641025