PAPER

Single Camera Vehicle Localization Using Feature Scale Tracklets

David WONG[†a)], *Nonmember*, Daisuke DEGUCHI[††b)], *Member*, Ichiro IDE[†c)], *Senior Member*, *and* Hiroshi MURASE[†d)], *Fellow*

SUMMARY Advances in intelligent vehicle systems have led to modern automobiles being able to aid drivers with tasks such as lane following and automatic braking. Such automated driving tasks increasingly require reliable ego-localization. Although there is a large number of sensors that can be employed for this purpose, the use of a single camera still remains one of the most appealing, but also one of the most challenging. GPS localization in urban environments may not be reliable enough for automated driving systems, and various combinations of range sensors and inertial navigation systems are often too complex and expensive for a consumer setup. Therefore accurate localization with a single camera is a desirable goal. In this paper we propose a method for vehicle localization using images captured from a single vehicle-mounted camera and a pre-constructed database. Image feature points are extracted, but the calculation of camera poses is not required — instead we make use of the feature points' scale. For image feature-based localization methods, matching of many features against candidate database images is time consuming, and database sizes can become large. Therefore, here we propose a method that constructs a database with pre-matched features of known good scale stability. This limits the number of unused and incorrectly matched features, and allows recording of the database scales into "tracklets". These "Feature scale tracklets" are used for fast image match voting based on scale comparison with corresponding query image features. This process reduces the number of image-to-image matching iterations that need to be performed while improving the localization stability. We also present an analysis of the system performance using a dataset with high accuracy ground truth. We demonstrate robust vehicle positioning even in challenging lane change and real traffic situations.
key words: *ego-localization, monocular vision, feature scale*

1. Introduction

The problem of ego-localization relative to a known map is central to vehicle navigation systems. Reliable and accurate ego-localization is important not only for in-car navigation systems, but it is also a key component of many of the emerging vehicle technologies that automate driving tasks. Global Positioning Systems (GPS) are standard equipment on many modern automobiles, but while consumer models are effective when the vehicle has a clear view of the sky, localization performance is reduced in typical urban environments. Tall buildings, tunnels, trees and other structures common in city driving prevent a direct signal path to satellites and cause average errors of more than 5 m or complete failure to localize.

GPS inaccuracies can be somewhat overcome by augmenting with other sensors, such as an Inertial Measurement Unit (IMU), range sensors, and/or odometry. However, computer vision is already employed on some newer vehicles for use in lane detection, parking, and automated driving tasks. Therefore, cameras already in place in production vehicles could ideally be re-purposed for ego-localization.

Computer vision for localization is a very active area of research for automotive and robotics applications. There are many proposed methods, usually based around self-localization in an unknown environment or localization relative to a pre-constructed map. The availability of a pre-constructed map, or database, is appropriate for automotive localization. Visual methods for map-relative localization are usually one of two types; metric localization with map-relative updates [1], or a simpler system using direct matching of input images to pre-captured database images [2], [3]. While the latter approach offers potentially better scalability and reduced complexity for real-time operation, accuracy when lateral motion occurs — for example, when the vehicle changes lanes — can be problematic. Localization with a single sensor is a challenging task, so most visual systems use multiple cameras or include supporting sensors.

While feature-based methods offer many advantages in regards to occlusion and lane change situations, visual odometry-based methods [1] require calculation of camera poses and the essential matrix which is computationally intensive and usually requires iterative processes such as Random Sample Consensus (RANSAC) [4] and bundle adjustment. As many features are used in these processes, feature matching is a performance bottleneck. In addition, calculation of the essential matrix when the distance between the query and database images is small, can cause short baseline degeneracies [5]. Our previous work [6], [7] uses scale invariant features and compares the scale of corresponding query and database image features to determine a database image match. If two images have the same viewing direction, their corresponding feature points will have a similar scale when the capture positions were spatially close. As the distance between the images increases, the difference in corresponding feature scales also increase. Finding the minimum average feature scale difference allows feature-based image matching for localization without essential ma-

Manuscript received May 31, 2016.
Manuscript revised October 6, 2016.
[†]The authors are with the Graduate School of Information Science, Nagoya University, Nagoya-shi, 464-8601 Japan.
[††]The author is with the Information Strategy Office, Nagoya University, Nagoya-shi, 464-8601 Japan.
a) E-mail: davidw@murase.m.is.nagoya-u.ac.jp
b) E-mail: ddeguchi@nagoya-u.jp
c) E-mail: ide@is.nagoya-u.ac.jp
d) E-mail: murase@is.nagoya-u.ac.jp
DOI: 10.1587/transfun.E100.A.702

trix calculation.

In this paper, an assumption of forward vehicle motion allows us to continue with the comparison of query image and database image feature scales, and further utilize database information to improve image matching accuracy and performance. The proposed method consists of the following contributions:

1. We make use of the known forward motion of the vehicle. We can monitor the scales of matched features and only retain those that have stable scale increase consistent with forward motion, which reduces the number of features in the database while maintaining the best features for localization. Fewer features make for a smaller database size and faster feature matching in localization. By keeping track of the feature matches and scales in the database, we create "feature scale tracklets" for rapid scale comparisons.
2. A per-feature, look-ahead database match voting method in the localization phase for determining the closest database image to the current query image. This process uses the "feature scale tracklets" to allow more (fast) feature scale comparisons in the closest database image calculation, while reducing the number of (slow) image-to-image matching steps.

We use a dataset with accurate ground truth to evaluate the localization results of the proposed method. We confirm the stability and accuracy of the method even when lane changes occur. For evaluation, we compare the results to those obtained using an effective image descriptor for image matching [2].

This paper is organized as follows: In Sect. 2 we give a brief overview of related research. We describe our novel contributions in Sect. 3 and the overall localization process in Sect. 4. Experimental results are presented in Sect. 5, followed by a discussion in Sect. 6 and the paper is concluded in Sect. 7.

2. Related Work

While there are a large number of visual localization methods, we will briefly summarize some of those related to our approach. The majority of automotive and robotic localization systems extract repeatable feature points within images, such as the Scale Invariant Feature Transform (SIFT) [8] or the newer Speeded Up Robust Features (SURF) [9]. Recently there have been an increasing number of feature point detection and description techniques, each with various advantages in processing efficiency or robustness. Popular recent methods use a binary descriptor for fast detection and description. BRIEF [10], ORB [11], BRISK [12] and FREAK [13] are some modern binary descriptors, typically extracted at feature points detected by the FAST [14] detector, which incorporates machine learning into the corner detection process for determining good feature locations.

In robotics, Simultaneous Localization and Mapping (SLAM) [15] creates a map as self-localization proceeds. Many methods use feature-based camera pose estimation [16]–[18]. The feature points are matched between captured images and those stored in the map or database, and relative localization is performed using structure-from-motion techniques. The calculation of scene geometry and camera pose from feature points can also provide some high accuracy localization results in the automotive setting [1] using a pre-constructed feature map. Methods that calculate camera pose usually require many feature points within a RANSAC [4] framework to select an inlier feature set. This in turn can lead to scalability issues when the database covers a large distance as many features need to be stored per image frame. They can also be computationally intensive, often barely running in real-time.

Dynamic Time Warping (DTW) [19] is a process used to remove temporal differences between sequences. In the case of automotive visual localization, it can be applied to two image streams — a database image stream, and a query input image stream. By minimizing an image match cost, query and database streams can be spatially aligned, therefore allowing query images to be localized from known database capture locations. This process has been applied using various image similarity measures, including Euclidean distance of dimension reduced images [3] or intensity difference in down-sampled image streams [20]. These methods scale well to large databases and are computationally less complex than those that calculate camera pose from features.

In addition to Dynamic Time Warping, probabilistic approaches have been used to match database image locations to query images [2], [21], [22], employing a Bayesian network to determine the most likely current position based on image similarity. Incremental map construction has been included to make a SLAM-like system [22] which is very scalable to large maps, with a high recall rate but low precision. Badino et al. [2] proposed the use of a whole-image feature descriptor based on the SURF descriptor, named WISURF — Whole Image SURF [9], [23]. This descriptor can be used with either DTW or a Bayesian system, and real-time localization performance of high accuracy was demonstrated on large datasets when using two cameras. This method has a low complexity and database size while offering good localization performance. We used the WISURF descriptor to implement a baseline method for evaluation. Using WISURF descriptors results in one descriptor per image, allowing fast image comparisons over a large number of candidate database matches. However, as we show in Sect. 5, using many feature descriptors per image together with the proposed method provides more accurate localization and added robustness in challenging image matching situations such as occlusions and lane changes.

The DTW and probabilistic methods mentioned above scale to large sized maps and are less complex than feature-based approaches, which require calculation of scene geometry between views. However, they rely on whole image similarity, so tend to perform poorly when the query images and database images differ significantly in structure, for example, in the case of large occlusions and lane changes.

3. Our Contributions

3.1 Pre-Matched Database

Image databases for visual localization typically store descriptor representations for each image [2], [3], and feature-based methods will contain a set of many feature descriptors per database image [6], [24]. Since feature-based localization methods perform feature matching between database and localization images, a carefully constructed database of known inliers is important for better matching performance.

In the construction of the database, we know that the vehicle is constantly moving forward. We therefore know that the scale of features will increase as they are observed in consecutive database images. By making use of this and matching features between database frames, two things can be achieved in database construction:

1. Pruning of feature matches where feature scale does not increase significantly with forward motion. Since we compare the scale of matched features for localization of a query image, using only features which exhibit a linear scale change with changing capture distance reduces the number of features which make an incorrect contribution to location prediction.
2. As each feature in the database is matched to corresponding features in adjacent frames, matching of query features to database features can be performed just once instead of many times over several database images. The scales of the corresponding database features can also be quickly looked up, allowing a query feature's scale to be compared to a stream of matching database feature scales. We call the string of corresponding database feature scales a "feature scale tracklet".

The resulting database is a web of interconnected features, arranged into "feature scale tracklets", $T \in \mathcal{T}$ where $\mathcal{T}$ is the full set of database tracklets. A feature tracklet T consists of a list of M pre-matched database features from sequential frames:

$$T = \{g^{\lambda_1}, g^{\lambda_2}, ..., g^{\lambda_M}\}, \tag{1}$$

with each λ_m (with $m = 1, 2, ..., M$ as the index along the tracklet) being the database image index which refers to the database image containing the feature g.

3.2 Per-Feature Image Match Voting

Localization methods that use image similarity to predict location require the matching of a query image to a number of database image candidates, often using probabilistic methods [2], [21] or DTW [3], [7], [24]. To find the database image that matches the query image, typically a match cost is minimized and the query image is tested against many sequential database images. This is normally the most time consuming part of the localization process, particularly for feature-based methods, as feature matching is relatively slow.

In this paper, we introduce a novel method for rapid convergence on the correct database match without the need to calculate the match cost for all local database images. Our proposed method allows individual features to vote on the most likely database match, and convergence usually occurs within two or three image match tests.

In our per-feature look-ahead image matching method, features extracted from a query image are matched to the features from a candidate database image λ. The feature matching process results in N matched query image features, with each feature f_i $(i = 1, 2, ..., N)$ being mapped to a feature tracklet $T_t \in \mathcal{T}$ which contains the corresponding database feature g^λ, as follows:

$$f_i \mapsto T_t \quad \text{where} \quad \min_t \gamma(f_i, g^\lambda \in T_t). \tag{2}$$

Here the subscript t of T_t refers to the tracklet index within the database, and $\gamma(.)$ is the feature matching function which finds the corresponding database feature by searching for the minimum descriptor distance, which we present in more detail in Sect. 4.1.2. We can now identify the feature within the tracklet that is closest in scale to the current query image feature f_i, and therefore find the database image match predicted by the feature:

$$\lambda(f_i) = \arg\min_\lambda |s(f_i) - s(g^\lambda \in T_t)|, \tag{3}$$

where the function $s(.)$ returns the scale of a feature. Now all of the matched features can individually estimate and vote on which database image provides a match, with the most voted database image being selected as the image match as follows:

$$\lambda_{\text{match}} = \arg\max_k \sum_{i=1}^{N} v(k, \lambda(f_i)), \tag{4}$$

$$v(k, \lambda(f_i)) = \begin{cases} 1 & \text{if} \quad \lambda(f_i) = k \\ 0 & \text{otherwise} \end{cases}, \tag{5}$$

where λ_{match} is the database index of the image match.

An overview of the per-feature image match voting method is presented in Fig. 1, and the scale comparison process for an individual feature is illustrated in Fig. 2. By comparing query feature scale within the "feature scale tracklets", we replace the many expensive feature descriptor matching steps with a series of feature scale comparisons. Particularly if the candidate database image is actually quite far from the true match, there may be some variations in the voted match. In any case, the resulting database image is selected as the next candidate and the process is repeated. The process terminates when the current candidate database image results in a majority of votes for itself. When this happens, the database image is selected as a match.

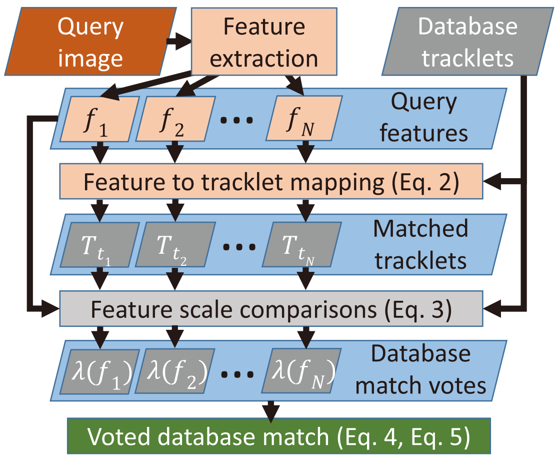

Query image
Feature extraction
Database tracklets
f_1 f_2 ••• f_N
Query features
Feature to tracklet mapping (Eq. 2)
T_{t_1} T_{t_2} ••• T_{t_N}
Matched tracklets
Feature scale comparisons (Eq. 3)
$\lambda(f_1)$ $\lambda(f_2)$ ••• $\lambda(f_N)$
Database match votes
Voted database match (Eq. 4, Eq. 5)

Fig. 1 An overview of the per-feature voting process for selecting the closest database match. Each feature extracted from the query image is matched to a database tracklet feature, and supplies a vote for a candidate database image. The most voted database image is selected as a match.

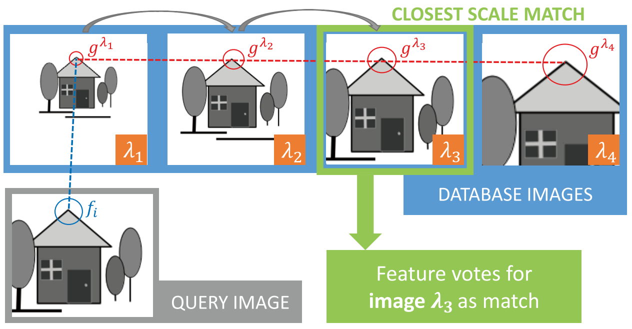


Fig. 2 Simplified diagram of the per-feature image match voting. Only one feature is shown here for clarity. The scale of the feature in the query image is scanned through the corresponding database feature's "feature scale tracklet", shown here in red. It can then vote on the predicted database image match, in this case image λ_3. The database image with the highest number of votes over all the features is then progressed to as the next candidate.

4. Localization Procedure

In this section, we describe how the proposed method achieves localization of input query image. The localization phase starts with feature extraction and matching to the features of the first candidate database image frame, and then comparison of feature scales within the database is used to converge on the closest database image match. An overview of the system process can be seen in Fig. 3. The process for the database construction phase is described below in Sect. 4.1. The database of the proposed image matching process described above in Sect. 3.2 depends on the use of appropriate feature points as explained in Sect. 4.1.1, and also accurate feature matching which is described in Sect. 4.1.2. This is followed by a more detailed explanation of the localization phase in Sect. 4.2.

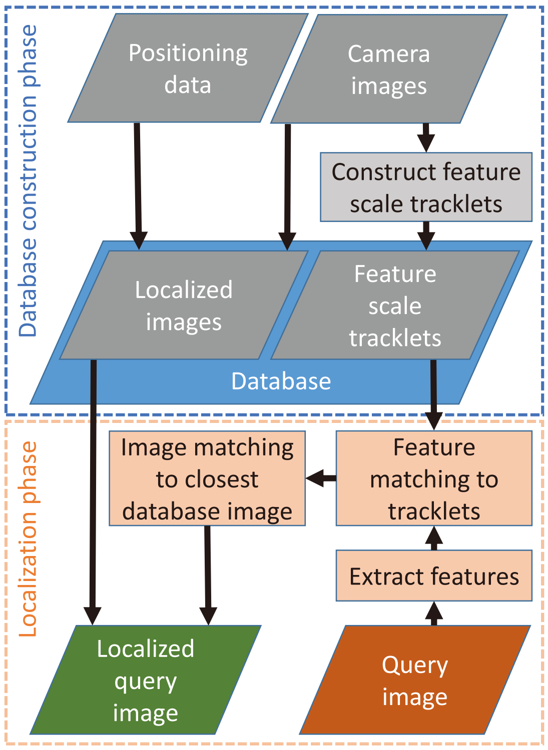


Fig. 3 Overview of the proposed system processes.

4.1 Database Capture and Construction Phase

Our system requires an image database with accurate capture locations, e.g. provided by a high quality GPS or mapping system. In this research, images were captured from a vehicle-mounted mapping system, incorporating high accuracy GPS, IMU, and odometry sensors for collecting localization data. All vehicle routes to be localized must be previously traversed by the database capture vehicle — while this appears a prohibitive step for large scale localization, datasets such as Google Street View have demonstrated that it is a possible task.

One consideration in the database capture process is the frame rate of image capture. The localization position of a query image comes directly from the closest database match, so a high frame rate is preferable to create a database with high spatial resolution. However, at very fast frame rates, image discrepancy between adjacent database images becomes very small and all image matching methods will struggle to distinguish between them. In addition, a small spacing between database frames leads to a very large database and slower localization. In this research, we found that an approximately 2 m database image spacing returned close to 1 m average localization error, with modest database size. More information on experimental setup and results is presented in Sect. 5.

4.1.1 Feature Point Extraction

Once the images have been captured, feature points are extracted from all of the frames. There is an increasing number of scale and illumination invariant feature detectors and descriptors, with potentially the two most frequently used methods being the original SIFT method [8] and its faster variant SURF [9]. The feature detector determines the feature locations and scales, whereas the descriptor describes the feature region such that it can be matched to corresponding features in other images. In our previous research [6], [7], we used the SURF detector and descriptor. However the work presented here uses the SIFT feature detector. We found that the feature scales extracted with SIFT keypoints were more consistent than those extracted by SURF. We use SIFT for feature detection and SURF for descriptors. SURF feature descriptors gave better matching performance at lower computational cost. In general, the proposed method is applicable to any feature extraction and description method with a scale property, so would work with more modern multi-scale feature detection methods such as FAST [14] and binary descriptors such as BRIEF [10], ORB [11], BRISK [12], and FREAK [13].

4.1.2 Feature Point Matching

The next step is to match extracted feature points between sequential database images. Since the proposed localization method does not calculate image geometry, typical inlier selection schemes for feature matching using RANSAC [4] cannot be used. However, in road environments, there are certain constraints that can be applied to improve the rate of inlier matches. We assume that the camera is forward facing, and that the primary camera motion is in the direction that the camera is pointing. Together with the assumption that camera height is known, we can restrict the search area for feature matches.

In addition to the scale change pruning outlined in Sect. 3.1, we apply a weighted match cost. This applies scale and feature response weights to help identify the best potential matches. The best feature match for feature f is calculated by finding the feature g in the next database image containing the pre-pruned set of features which minimizes the following equation:

$$\begin{aligned}\gamma(f,g) = w_s|s(f)-s(g)| + w_r|r(f)-r(g)| \\ +w_d(\mathrm{SSD}(f,g)),\end{aligned} \tag{6}$$

where function $s(.)$ returns the feature scale, function $r(.)$ the feature response, and $\mathrm{SSD}(f,g)$ is the standard sum of squared differences of the feature descriptors. The weights w_s, w_d, w_r are adjusted to give a strong inlier set while maintaining a high number of matched features. Finally the best matches are selected by dropping any matches which have a descriptor distance of more than twice the minimum descriptor distance within the set of matches. All un-matched features are discarded from the database.

4.2 Localization Phase

In the localization step, first a likely database image match is selected as a candidate based on the last match. In a real-world system, this could be initialized either with a normal consumer GPS or by using a visual search within a database to find a likely region to start in, similar to the method proposed by Cummins and Newman [22]. SIFT features are extracted from the query image, and then matched to the recorded features of the candidate database image. The same selective weighted feature matching given by Eq. (6) is used, with the exception of the restriction on increasing feature scale, since we do not know if the query image is before or beyond the first candidate database image.

The database image match is then predicted using the proposed per-feature image match voting using "feature scale tracklets" as described in Eqs. (2) to (5) of Sect. 3.2. Once the database match is selected, the localization information associated with it is applied directly to the query image. When we move on to the next query image, the next database image is used as the first candidate image. This method has merit in that it does not require a motion model or velocity estimates to proceed with localization. While we assume that the vehicle is moving forward in our method, it is not difficult to modify the method to accomodate backward motion. We also do not consider localization at junctions in this paper. However, we provide a discussion on possible implementation for junctions and backward motion in Sect. 6.1.2.

5. Experiment

To evaluate the localization performance of the proposed method, a dataset captured at a driving school was used. The location allowed various lane changes, traversal of intersections and maneuvers to be safely performed away from busy traffic. A sophisticated data capture system was employed to provide accurate ground truth data in this dataset. This process is presented in Sect. 5.1, and feature matching parameters discussed in Sect. 5.2. In Sect. 5.3, we describe the baseline methods used throughout these experiments, and present the results in Sect. 5.4. In addition, the proposed method was also tested in real traffic environments. For this data, accurate localization ground truth was not available, but the system was evaluated by looking at the image matching performance, presented in Sect. 5.5.

5.1 Vehicle Configuration

The database and query image streams were captured using a Mitsubishi Electric MMS-X320R Mobile Mapping System (MMS). This system incorporates three 5 megapixel cameras, three LASER scanners, GPS, and odometry hardware. Only one forward facing camera was used in these

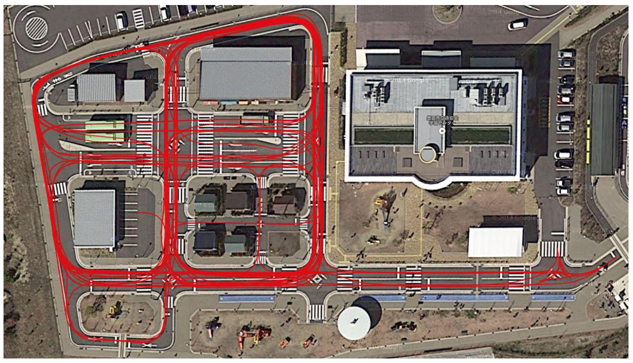


Fig. 4 The dataset location, showing the driving paths used. Satellite imagery: Google, ZENRIN.

experiments. The MMS provides a claimed localization error of less than 6 cm (RMS), and the system provided an estimated average error of below 1 cm in the experiments that were conducted. Image capture is at approximately 2 m intervals.

Several database sequences were captured around the driving school, always using the left-hand lane. Query image streams used a mixture of the left-hand lane and the right-hand lane where multiple lanes were available. Figure 4 shows the vehicle paths.

5.2 System Parametrization

In the database construction and localization feature matching, the weights w_s, w_d, w_r from Eq. (6) were selected by visually checking feature correspondences, and choosing values that minimized incorrect matches. The scale difference of correct matches varies depending on feature size, so a relatively small w_s value of approximately one tenth of w_d and w_r (which were approximately equal) was found to be effective. This configuration prioritizes the SSD of feature descriptors for determining the best feature match. The weights were normalized to sum to one, resulting in a w_s value of 0.0476, with both w_d and w_r values set to 0.476.

5.3 Baseline Methods

To evaluate the performance of the proposed system, we implemented three baseline methods.

1. **Feature Scale DTW.** This method is based on our previous work [7] and uses the average scale difference between matched features of individual images as a cost measure within a DTW framework. The image match cost can be considered as the averaged costs between matched features, as in Eq. (6) but using only the scale difference and setting w_d and w_r to zero. Like the proposed method, it uses matched feature scale comparisons; however, it does not use the pre-matching and tracklet construction techniques presented in this paper.
2. **Feature Match Cost DTW.** This method uses the same DTW framework as Feature Scale DTW, but instead of using average scale change as a cost measure for image matching, it uses the feature match cost from Eq. (6), averaged over all feature matches to a candidate database image. It includes scale difference, response difference, and descriptor distance. The results for Feature Match Cost DTW as shown here are using the weights w_s, w_d, w_r as optimized for feature matching (see Sect. 5.2). Again, this method omits the proposed pre-matching and tracklet construction techniques.
3. **WISURF-DTW.** We implemented a modified version of WISURF [2] for image matching. The WISURF method creates a single SURF descriptor for each image, and chooses a database image match based on descriptor distance. Instead of using a Bayesian filter [2], we performed image matching using WISURF image descriptor distance and DTW to remove the dependence on motion estimation for localization. The experimental results, when the same lane was traversed in both passes, gave similar results to those presented in [21] in downtown areas, which is impressive, considering only one camera was used in our testing.

5.4 Localization Performance

Databases were constructed for the proposed and baseline methods, and localization performed through sections of the dataset. The results focus on two areas; standard localization when the query images were captured in the same lane as the database images, and the lane change condition where the query images were captured in the right-hand lane. Some examples of the resulting image matching are shown in Fig. 5.

The localization process was performed using a standard desktop computer, with a 3.5 GHz Intel i7 processor and 8 GB of RAM. The OpenCV libraries were used for feature detection and descriptor extraction. Without any GPU implementation, multi-threading nor optimization of any kind, localization was performed at approximately 15 Hz. At 100 km/h, a 14 fps image capture rate is required to achieve the 2 m image spacing used in these experiments. Therefore 15 Hz localization would be possible at up to 100 km/h without increasing the image spacing beyond 2 m.

Localization accuracy for all methods was evaluated using the MMS localization data. For the query image stream, localization information provided the ground truth. One way to evaluate the effectiveness of the system is to regard image matching performance — how reliably the method can find the spatially closest database image given a query image. Figure 6(a) shows the image matching performance of the proposed method together with each of the baseline methods as described in Sect. 5.3, where database and localization streams were in the same lane. Figure 6(b) shows the image matching performance on a short sequence where the query images were captured in the right-hand

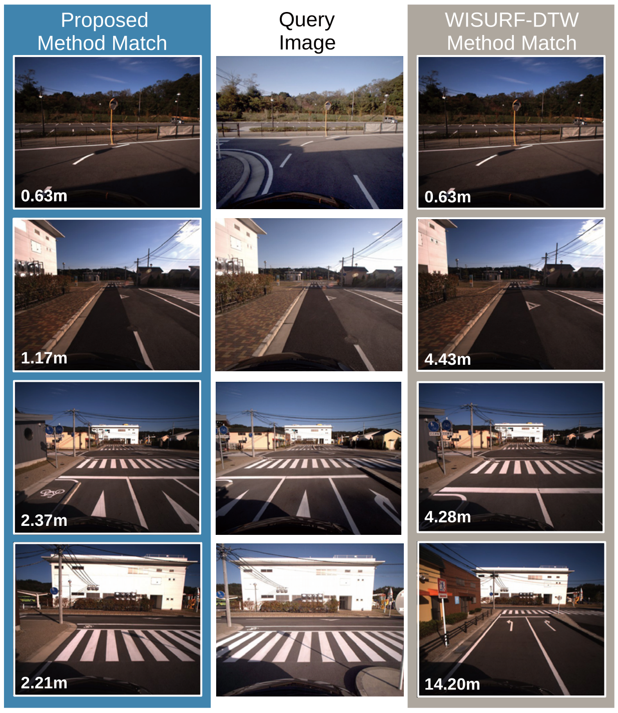
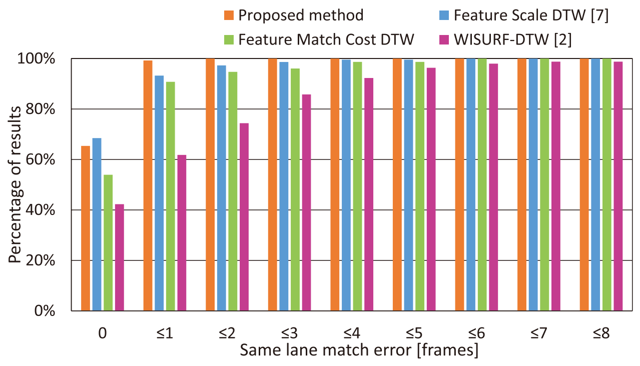
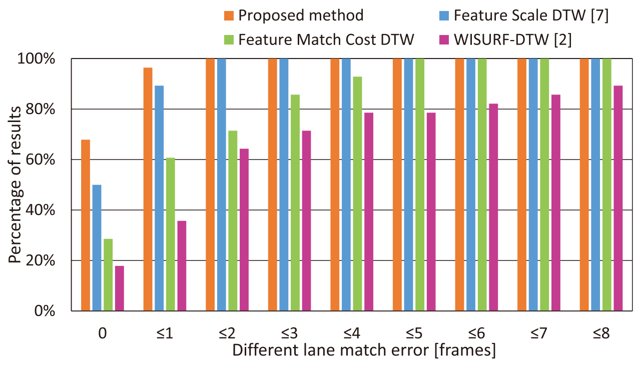


Fig. 5 Example image matching results. The top two rows show image matching in the same lane, whereas the lower two show matching where the query lane is different from the database lane. Current localization errors are shown in the bottom left-hand corner.



(a) Same lane



(b) Different lane

Fig. 6 Graph showing the image matching performance of the evaluated methods. These example sections are taken from areas where the query images are captured (a) in the same lane as the database images, and (b) in the lane to the right of the database image lane. For comparison, methods without tracklets that use the average feature scale change for image matching (Feature Scale DTW), average feature match cost (Feature Match Cost DTW) and the WISURF-DTW method are also shown. The *x*-axis shows the match error in terms of number of image frames between the selected database match and the actual closest database image. The *y*-axis shows the percentage of image match results that are within each error level.

lane.

A summary of the statistics of the tracklets constructed within the complete driving school dataset is presented in Table 1. The tracklet length is defined as the number of consecutive database images in which it appears, and the number of tracklets per frame is the number of tracklets that pass through a database image. Figure 7 shows the matched features of a sample set of tracklets tracked through four database frames.

Absolute localization errors are presented in Table 2 and Fig. 8. The proposed method achieved an average accuracy of 0.68 m in the same lane case and 3.17 m in the different lane case. It should be noted that when the query images were captured in the right-hand lane, even perfect image matching results would always result in at least a 2 m error. This corresponds to the approximate width of the lane, as both the proposed and baseline methods can only localize longitudinally in the direction of motion. This can be observed in Fig. 8, where the different lane curves start at an error level of approximately 2 m.

The proposed database construction method makes use of the forward motion of the vehicle for filtering strong features for localization. However, as can be seen in Fig. 4, the database includes many sharp corners. In these locations, the change in vehicle direction of motion causes some of the tracked features to disappear from the field of view. This results in the end of tracklets and the creation of new ones as the view changes. To determine the affect of corners

Table 1 Tracklet statistics.

	Max.	Min.	Avg.
Tracklet length [frames]	17	2	3.3
Number of tracklets per frame	789	78	255.4

on localization accuracy, we isolated one small section of the driving school database including two straight sections and two 90° corners, and compared the average localization error of corner areas and straight areas. The results showed an average localization error of 0.73 m in corner areas, and 0.68 m in straight sections with an overall error of 0.69 m for the complete tested area.

5.5 Real-World Traffic Performance

In addition to the driving school dataset, experiments were also performed on real-world data captured from a vehicle in traffic. This allowed the testing of the system performance

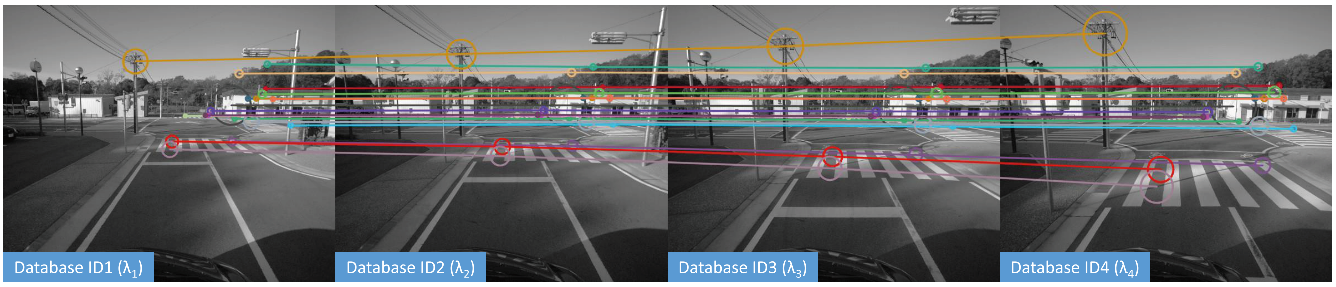
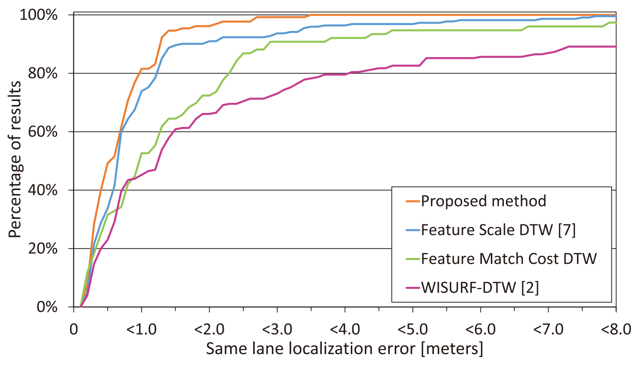

Database ID1 (λ_1)
Database ID2 (λ_2)
Database ID3 (λ_3)
Database ID4 (λ_4)

Fig. 7 Sample "feature scale tracklets", with each tracklet shown in a different color. The circles represent the feature positions and their diameters show the feature scale, illustrating the scale increase along the tracklet. Only some of the tracklets are shown for clarity.

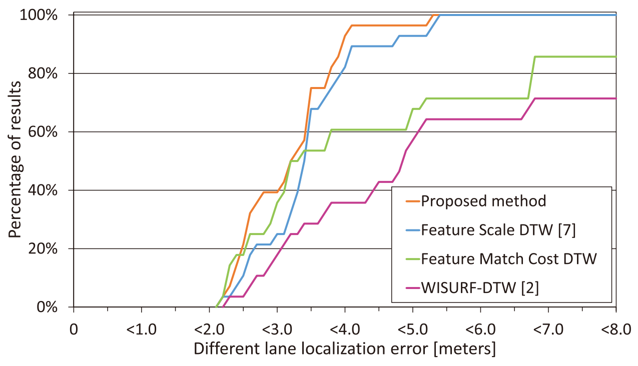
Table 2 Localization accuracy.

Method	Query lane	Avg. error (m)	Max. error (m)
Proposed	Same	0.68	4.61
	Different*	3.17	4.56
Feature Scale DTW [7]	Same	1.00	11.04
	Different*	3.43	5.25
Feature Match Cost DTW	Same	1.54	8.51
	Different*	4.57	10.47
WISURF-DTW [2]	Same	3.71	14.33
	Different*	7.14	24.10

*Includes approx. 2 m localization error from lateral lane offset



(a) Same lane



(b) Different lane

Fig. 8 Localization error of the proposed method using tracklets compared with matching images using average feature scale change as a cost measure (Feature Scale DTW), average feature match cost (Feature Match Cost DTW), and the WISURF-DTW method. The graph shows the percentage of localization results within each metric error level. Graph (a) shows the same lane localization results, and for the different lane results (b), the localization error starts at about 2.0 m which approximately corresponds to the width of the lane.

in situations where occlusions and temporal objects (such as pedestrians and other moving vehicles) are common.

In these experiments, a Point Grey Ladybug camera was mounted on the test vehicle and used to capture both database and query image streams in a variety of city driving environments. Only the front facing camera was used, with a capture rate of 15 frames per second (fps). The data capture covered approximately 1 km of road, with the database and query streams both being captured in the left-hand lane. For these tests, accurate localization information was not available, so the results are presented only with image matching performance, verified manually. Figure 9 shows example images from this dataset, where typical traffic and occlusions are observed. The resulting image matching results are presented in Fig. 10.

6. Discussion

6.1 Localization Results

The average localization error when database and query images came from the same lane was lower than in our previous work [7], even with the use of only one camera, showing that the per-feature database match voting system provides a more stable image matching platform. On the driving school dataset, the proposed "feature scale tracklet" method reduced the average same lane localization error by 32% and 44% over the Feature Scale DTW and Feature Match Cost DTW methods, respectively. The average localization error was reduced by 82% when compared to the baseline WISURF-DTW method.

The approaches of Feature Scale DTW and Feature Match Cost DTW differ only in the weights used for the

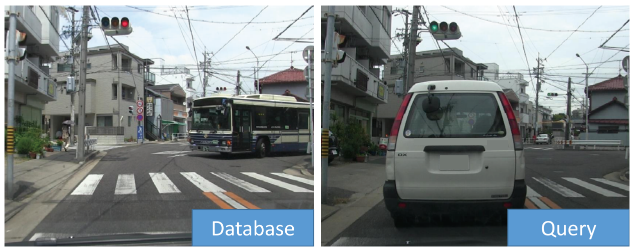


Fig. 9 Example images from the dataset captured in real traffic. The query image match for the database image has been largely occluded by a vehicle.

image match cost measure. While Feature Scale DTW uses only the scale component of Eq. (6) to determine an average image match cost over corresponding features, Feature Match Cost DTW uses Eq. (6) directly. Feature response and descriptor comparisons are included in the cost measure, with the same weights as used in feature matching. The superior image matching performance of Feature Scale DTW over Feature Match Cost DTW illustrates the strength of using feature scale for image matching in this environment.

The proposed method uses two cost measures; one optimized for finding feature matches (Eq. (6)), and one for image matching (Eq. (3)). The baseline Feature Match Cost method uses the feature match cost of Eq. (6) directly as an image match cost measure. To compare the image match cost measure of the proposed method and the baseline Feature Match Cost method, we also performed an experiment to optimize the weight values w_s, w_d, w_r of Eq. (6) independently of the feature matching procedure, in order to determine the best possible combination of all three weightings for an image match cost measure. It was found that the weight values that were optimal for individual feature matching in Eq. (6), as selected in Sect. 5.2, did not provide optimal image matching performance when using Eq. (6) averaged over all feature matches as an image match cost measure. The results showed that a scale difference weight w_s of 1.0 and 0.0 for w_d and w_r provided the best image matching. This is a system exactly equivalent to Feature Scale DTW, and also the fundamental image match cost used in the proposed method. The accuracy decrease when using descriptor distance is potentially because the descriptors of corresponding features from consecutive database frames are very similar. This results in ambiguity when choosing the image match based on descriptor distance, and therefore the occurrence of incorrect image matches. Feature scale difference appears to be a more discriminative measure for resolving forward camera motion. Note that the feature response difference is very small for matched features and does not contribute significantly to image matching. It is, however, effective in filtering incorrect matches in the feature matching process.

The superior image matching performance observed when using scale difference only in Feature Scale DTW and the proposed method confirms that feature scale change within a tracklet (Eq. (3)) is a good parameter for image matching, and also illustrates that comparing descriptors within the tracklet is unnecessary for the localization step. This is where the efficiency gains of the proposed method become apparent, as descriptor comparisons are computationally expensive.

The proposed method also offers other efficiency advantages. Most query images are localized within three image matching steps, compared to at least eight when using DTW and image match cost comparisons. We estimate that the image matching performance with our “feature scale tracklet” and per-feature voting system is on average at least four times faster than the DTW-based feature scale difference method (Feature Scale DTW) and feature match cost method (Feature Match Cost DTW) when using a similar number of features. While the image matching step is much faster when using the proposed tracklet system, the feature extraction process is the same for all feature-based methods and takes up a large proportion of the overall localization time. Therefore, the scale change cost and feature match cost methods used for comparison ran at around 7 Hz compared to 15 Hz for the proposed method. Note that Feature Scale DTW and Feature Match Cost DTW run at close to the same speed, as the image cost measure calculation differs only by two floating point number value comparisons per feature. However, WISURF-DTW runs much faster than all of the feature based methods and can run in real-time at all practical frame rates.

All three feature-based methods also showed significantly improved localization performance over WISURF-DTW, especially in the maximum observed error. While the use of the WISURF descriptor is a fast and efficient way to compare overall scene similarity, incorrect matches occur when consecutive database scenes either change very little in overall appearance, or when the query image scene is modified from the corresponding database scene — for example, in the case of a lane change or occlusion by another vehicle. Splitting each image into many descriptors adds robustness in these situations. Even consecutive database frames with similar overall appearance have individual feature points which change in scale, and are repeatable even when a lane change takes place or an occlusion obstructs some of the scene. The advantage of using many feature point descriptors instead of an overall image descriptor is further discussed in the following sections.

6.1.1 Different Lane Localization

Image matching is more challenging in the different lane case because the scene viewed by the camera changes significantly between lanes. This is illustrated by the baseline WISURF-DTW method results shown in Fig. 6(b) and Fig. 8. The WISURF-DTW method showed much higher localization error in the different lane case when compared to localizing in the same lane as the database, even when considering the 2 m lane offset. However the proposed method, after taking into consideration the lane offset, actually achieved similar localization performance (Fig. 8) and image matching performance (Fig. 6(a)) to the same lane case. This illustrates one strength of using features for localization, as opposed to whole image similarity.

6.1.2 Real-World Localization

The image matching performance in the real-world traffic environment is reduced for two reasons. Firstly, the camera captured images at a rate of 15 fps, resulting in much closer image separation and therefore smaller differentiation between frames. Secondly, occlusions and dynamic traffic scenes causing differences between the database and query images pose a challenge for image matching. Looking at individual matching failures, the baseline WISURF-DTW method was much more affected by both of these issues than the proposed method, which is evident in the image matching results (Fig. 10). However, what is harder to see, is that partially occluded scenes such as in Fig. 9 sometimes caused complete matching failure when using WISURF, whereas the proposed method's matching performance was mostly unaffected by such situations.

Database images captured from busy traffic environments will sometimes include tracklets with features extracted from temporal objects such as other vehicles and pedestrians. These tracklets do not have much effect on the localization process, as even if they are incorrectly matched to query features, their database index votes are outweighed by those of static features. In future work, vehicle and pedestrian detection systems could be used to remove the features from temporal objects. This would slightly reduce the database size and matching computation time. Another potential method to remove temporal object features would be to make multiple database capture passes and create tracklets from only consistent features over different capture sets. Information about matched features in the localization process could also be used to improve the database set of tracklets, which would also allow database evolution over time.

In this paper, we do not consider backward motion or junctions. While we assume forward motion, backward motion would usually be detected and correctly handled as long as the matched tracklets sufficiently cover database frames behind the current camera position. However, also testing the previous database image as a match candidate would potentially improve performance where backward motion is likely. Places where the road forks would also not be difficult to include. At such locations, several candidate images (one for each possible traversal route) would be selected as opposed to a single one. As localization progresses along each potential traversal route, the route with the lowest match cost would be selected as the correct one. A similar process has been used successfully with the WISURF method [21].

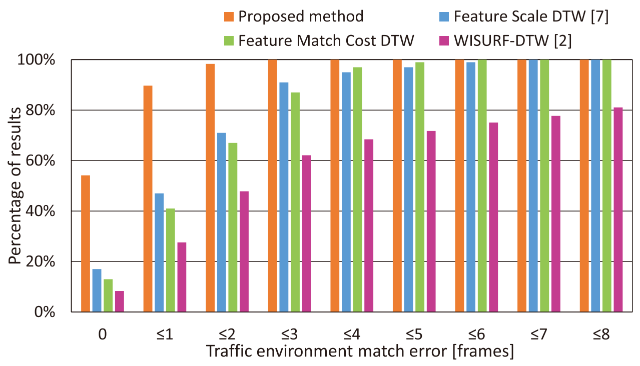


Fig. 10 Image matching results from the traffic dataset, presented in the same format as Fig. 6. This dataset includes other vehicles and occlusions typical of city driving.

6.2 Database Size and Lateral Localization

One consideration is the database size. The proposed method creates a database of up to about 100 times the size of the WISURF-DTW baseline method, depending on the number of features extracted per image. The descriptors take on average about 120 KB per meter. The localization accuracy would definitely be improved by creating a database with 1 m image spacing instead of the 2 m used in these experiments; however, this would also result in a database of twice the size. There are potentially many ways to reduce the database size, including using simpler descriptors, or monitoring and pruning descriptors which never get matched to query image features over time. Within the "feature scale tracklets", we could also maintain just one descriptor per tracklet rather than the full set of feature descriptors over several database images.

In urban environments where multi-lane roads are common, it is desirable to be able to accurately localize laterally as well as longitudinally without requiring database passes in all available lanes. This is an area for future research.

7. Conclusion

In this paper, we proposed a method of visual localization using the comparison of feature point scale across correspondences in a pre-constructed database. The experimental results showed that robust image matching can be achieved using this method. Our approach achieved average localization errors of 0.68 m when database and query image streams are in the same lane and 3.17 m in different lanes with different lane results including the lane offset error (approximately 2 m).

Experiments presented in this paper used approximately 2 m database image spacing. While closer spacing would provide more accurate localization, it would also increase the database size.

The performance of the proposed method was also verified in real-world traffic environments. Image matching was successful in dynamic road scenes with occlusions, illustrating the robustness of using the scale of feature points for localization.

Future works will include more testing with different datasets and occluded scenes, as well as different database image spacing. We plan to implement a lateral localization system to remove the offset error introduced when traveling in a different lane from the database stream. Currently, the proposed method uses a binary voting system for image match selection. We also plan to implement a probabilistic voting system to increase the contribution of strong feature correspondences in image matching.

Acknowledgment

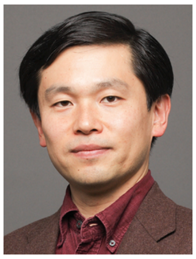
Parts of this research were supported by MEXT, Grant-in-Aid for Scientific Research.

References

[1] H. Lategahn, M. Schreiber, J. Ziegler, and C. Stiller, “Urban localization with camera and inertial measurement unit,” Proc. 2013 IEEE Intelligent Vehicles Symposium (IV2013), Gold Coast City, Australia, pp.719–724, June 2013.
[2] H. Badino, D.F. Huber, and T. Kanade, “Real-time topometric localization,” Proc. 2012 IEEE Int. Conf. on Robotics and Automation (ICRA2012), St. Paul, USA, pp.1635–1642, May 2012.
[3] H. Uchiyama, D. Deguchi, T. Takahashi, I. Ide, and H. Murase, “Ego-localization using streetscape image sequences from in-vehicle cameras,” Proc. 2009 IEEE Intelligent Vehicles Symposium (IV2009), Xi’an, China, pp.185–190, June 2009.
[4] M.A. Fischler and R.C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” Commun. ACM, vol.24, no.6, pp.381–395, June 1981.
[5] G. López-Nicolás, C. Sagüés, and J.J. Guerrero, “Parking with the essential matrix without short baseline degeneracies,” Proc. 2009 IEEE Int. Conf. Robotics and Automation (ICRA2009), Kobe, Japan, pp.1098–1103, May 2009.
[6] D. Wong, D. Deguchi, I. Ide, and H. Murase, “Single camera vehicle localization using SURF scale and dynamic time warping,” Proc. 2014 IEEE Intelligent Vehicles Symposium (IV2014), Detroit, USA, pp.681–686, June 2014.
[7] D. Wong, D. Deguchi, I. Ide, and H. Murase, “Vision-based vehicle localization using a visual street map with embedded SURF scale,” Computer Vision — ECCV 2014 Workshops Proc., Part I, Zurich, Switzerland, Lecture Notes on Computer Science, vol.8925, pp.167–179, Sept. 2014.
[8] D. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vision., vol.60, no.2, pp.91–110, Nov. 2004.
[9] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (SURF),” Comput. Vis. Image. Und., vol.110, no.3, pp.346–359, June 2008.
[10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary robust independent elementary features,” Proc. 11th European Conf. Computer Vision (ECCV2010), Part IV, Crete, Greece, Lecture Notes on Computer Science, vol.6314, pp.778–792, Sept. 2010.
[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” Proc. 2011 IEEE Int. Conf. Computer Vision (ICCV2011), Barcelona, Spain, pp.2564–2571, Nov. 2011.
[12] S. Leutenegger, M. Chli, and R.Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” Proc. 2011 IEEE Int. Conf. Computer Vision (ICCV2011), Barcelona, Spain, pp.2548–2555, Nov. 2011.
[13] A. Alahi, R. Ortiz, and P. Vandergheynst, “FREAK: Fast retina keypoint,” Proc. 2012 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR2012), Providence, USA, pp.510–517, June 2012.
[14] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” Proc. 9th European Conf. Computer Vision (ECCV2006), Part I, Graz, Austria, Lecture Notes on Computer Science, vol.3951, pp.440–443, May 2006.
[15] H.F. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: Part I,” IEEE Robot. Automat. Mag., vol.13, no.2, pp.99–110, June 2006.
[16] S. Se, D. Lowe, and J. Little, “Vision-based mobile robot localization and mapping using scale-invariant features,” Proc. 2001 IEEE Int. Conf. Robotics and Automation (ICRA2001), Seoul, Korea, pp.2051–2058, May 2001.
[17] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse, “MonoSLAM: Real-time single camera SLAM,” IEEE Trans. Pattern Anal. Machine Intell., vol.29, no.6, pp.1052–1067, June 2007.
[18] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, “Monocular vision based SLAM for mobile robots,” Proc. 18th IAPR Int. Conf. Pattern Recognition (ICPR2006), Hong Kong, pp.1027–1031, Aug. 2006.
[19] M. Muller, “Dynamic time warping,” in Information Retrieval for Music and Motion, pp.69–84, Springer, Berlin Heidelberg, Sept. 2007.
[20] M. Milford, “Visual route recognition with a handful of bits,” Proc. 2012 Robotics: Science and Systems Conf., Sydney, Australia, pp.297–304, July 2012.
[21] D. Xu, H. Badino, and D.F. Huber, “Topometric localization on a road network,” Proc. IEEE/RSJ 2014 Int. Conf. Intelligent Robots and Systems (IROS2014), Chicago, USA, pp.3448–3455, Sept. 2014.
[22] M. Cummins and P. Newman, “Appearance-only SLAM at large scale with FAB-MAP 2.0,” Int. J. Robotics Research, vol.30, no.9, pp.1–24, Aug. 2011.
[23] M. Agrawal, K. Konolige, and M.R. Blas, “CenSurE: Center surround extremas for realtime feature detection and matching,” Proc. 10th European Conf. Computer Vision (ECCV2008), Part IV, Marseille, France, Lecture Notes on Computer Science, vol.5305, pp.102–105, Oct. 2008.
[24] H. Kyutoku, T. Takahashi, Y. Mekada, I. Ide, and H. Murase, “On-road obstacle detection by comparing present and past in-vehicle camera images,” Proc. 12th IAPR Conf. Machine Vision Applications (MVA2011), pp.357–360, Nara, Japan, June 2011.

David Wong received his BE (Hons 1st) in Mechatronics Engineering from the University of Canterbury, New Zealand, in 2009. He is currently studying towards his PhD in Information Science at the Graduate School of Information Science of Nagoya University. His research focuses on image matching and processing for vehicle ego-localization.

Daisuke Deguchi received his BEng and MEng in Engineering and PhD in Information Science from Nagoya University, Japan, in 2001, 2003, and 2006, respectively. He is currently an Associate Professor in Information Strategy Office, Nagoya University, Japan. He is working on object detection, segmentation, and recognition from videos, and their applications to ITS technologies, such as detection and recognition of traffic signs.



Ichiro Ide received his BEng, MEng, and PhD from The University of Tokyo in 1994, 1996, and 2000, respectively. He became an Assistant Professor at the National Institute of Informatics, Japan in 2000. Since 2004, he has been an Associate Professor at Nagoya University. He was also a Visiting Associate Professor at National Institute of Informatics from 2004 to 2010, an Invited Professor at Institut de Recherche en Informartique et Systèmes Aléatoires (IRISA), France in 2005, 2006, and 2007, a Senior Visiting Researcher at ISLA, Instituut voor Informatica, Universiteit van Amsterdam from 2010 to 2011. His research interest ranges from the analysis and indexing to retargeting of multimedia contents, especially in large-scale broadcast video archives, mostly on news, cooking, and sports contents. He is a senior member of IEICE and IPS Japan, and a member of JSAI, IEEE, and ACM.

Hiroshi Murase received his BEng, MEng, and PhD degrees in Electrical Engineering from Nagoya University, Japan. In 1980 he joined the Nippon Telegraph and Telephone Corporation (NTT). From 1992 to 1993 he was a visiting research scientist at Columbia University, New York. From 2003 he is a professor of Nagoya University, Japan. He was awarded the IEICE Shinohara Award in 1986, the Telecom System Award in 1992, the IEEE CVPR (Conference on Computer Vision and Pattern Recognition) Best Paper Award in 1994, the IPS Japan Yamashita Award in 1995, the IEEE ICRA (International Conference on Robotics and Automation) Best Video Award in 1996, the Takayanagi Memorial Award in 2001, the IEICE Achievement Award in 2002, and the Ministry Award from the Ministry of Education, Culture, Sports, Science and Technology in 2003. Dr. Murase is a Fellow of IEEE and IEICE, and a member of IPS Japan.