Position Interpolation using Feature Point Scale for Decimeter Visual Localization

David Wong, Daisuke Deguchi, Ichiro Ide, Hiroshi Murase
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8601 Japan
{davidw, deguchi, ide, murase}@murase.m.is.nagoya-u.ac.jp

Abstract

Vehicle ego-localization is a critical task not only for in-car navigation systems, but also for emerging intelligent and autonomous vehicle technologies. Visual localization methods that determine current location by performing image matching against a pre-constructed database have an accuracy limited by the spatial distance between database images. In this paper we propose a method that uses the scale of feature points to interpolate the position of the query image between two database images. We show how this simple contribution offers an appreciable improvement in localization accuracy with an extremely minimal increase in processing time, especially when used in conjunction with image matching methods that already monitor feature scale. Our experiments showed an increase of up to 33% in average localization accuracy when compared to a method without any interpolation.

1. Introduction

Ego-localization is a central part of vehicle and robotic navigation systems. For automotive use, standard localization systems utilize a Global Navigation Satellite System (GNSS). Production vehicles typically incorporate a single frequency Global Positioning System (GPS) receiver for an in-car navigation system. While suitable for approximate positioning, these systems are not capable of the decimeter-level localization accuracy required for emerging intelligent vehicle systems and automated driving tasks. Particularly in city environments, the "urban canyon" situation, together with tunnels and road structures, can cause large localization inaccuracies. That is, signal shadowing, where GPS satellites are not in a direct line of sight from the receiver can lead to localization failure or poor position triangulation performance. The multi-path effect, where the GPS signals are reflected off buildings, can lead to apparent rapid motion or jumping of the receiver's output. These issues can cause average errors of 15 m or more in urban environments [16]. Therefore, more accurate localization can be achieved by using dual frequency GNSS systems, RTK (Real Time Kinematic) GPS systems where a reference station provides real-time corrections, or Inertial Navigation Systems (INS) which include high precision GPS together with an Inertial Measurement Unit (IMU). While these systems can provide highly accurate localization, they still may fail in heavily built up areas and are also too expensive and complex for consumer vehicles. They also may fail to recognize common road situations such as parallel, multi-level roads where raised highways run above secondary roads. Altitude error of GNSS systems is significantly higher than horizontal error, so it can be difficult for GNSS systems to determine which of the parallel stacked roads is being traversed. Alternatively, localization methods using laser scanners such as the popular Velodyne [17] have been proposed. However, laser scanners are still typically too expensive and difficult to integrate into standard vehicles for production automobile localization systems.

There are a number of methods that use computer vision for map-relative localization. Visual localization methods typically use a pre-built database of image descriptors or features captured by a vehicle-mounted camera. Localization can either be performed using extracted feature points [11], [12], [23], or by matching query images to database images using whole image similarity [2], [6], [22], [24]. The correct database image match can be determined either using Dynamic Time Warping (DTW) with an image similarity measure [11], [22], [23] or a Bayes filter [2], [6], [24]. Methods using feature points have the advantage of robustness to occlusions and lane changes, but also come with increased complexity. Most techniques employing feature points calculate the camera pose relative to the matched database frame [10], [12]. This requires calculation of the essential matrix, usually together with iterative processes such as Random Sample Consensus (RANSAC) [8] and bundle adjustment. In addition, the feature matching process of comparing many feature de-

scriptors is a computationally intensive process. In order to overcome these performance bottlenecks, feature-based localization methods which avoid calculation of image geometry [2], [23] use either descriptor distance [2] or feature scale [23] to determine the closest database image directly.

For visual localization methods which calculate the closest database image to the a query image, accuracy is limited by the spacing of the database images. There is a trade-off between reasonable database size and localization performance. A database with spatially close images will provide better localization accuracy but the localization process may be slower and the database size can become unwieldy.

In this paper we introduce a method for improving the localization accuracy of image-matching based systems, specifically those that already make use of feature points. The proposed method uses the known scale of database feature points, and the scale of matched query image points, to interpolate the query image position between the two closest database image frames. The observation that this technique makes use of is the fact that feature point scale increases approximately linearly with the inverse of distance to the camera. Change in feature scales is continuous, so can be used directly for position interpolation between database frames. We demonstrate the improvement in localization accuracy when using a recent image matching method for localization with a precise ground truth. The proposed method is simple so adds nearly no overhead to localization computation time.

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

2. Related Work

The method proposed by this paper is an extension of visual ego-localization systems that determine the current vehicle localization by matching an image captured by an in-vehicle camera to a pre-constructed database of known capture locations. While there are many ego-localization methods using a variety of sensors or combinations of sensors, we limit our discussion to vision-only methods.

2.1. Visual Localization

The problem of visual ego-localization for automotive applications is closely related to the Simultaneous Localization and Mapping (SLAM) [7] field in robotics. In SLAM, visual methods typically use feature points extracted from images to calculate camera poses and build a map of the environment being explored, with modern methods often using a pose graph approach [4], [9]. A challenging problem in SLAM is scaling to large environments, which is necessary in automotive applications. For vehicle localization on a road network, a reasonable assumption is that a pre-constructed map is available for all possible roadways. Large on-line databases such as Google Street View have demonstrated the feasibility of creating such databases. Where a pre-constructed map or database is available, the localization problem resembles the loop closure, or place recognition component of visual SLAM [4], [6] where the current location must be found within the existing map. In vehicle ego-localization, this has been achieved using dense feature maps and pose estimation from matched features [12]. While excellent localization results were achieved, database sizes were quoted as approximately 5 GB for 7 km of road. Simpler approaches match whole images, using simplified image representations such as Euclidean distance of dimension reduced images [22] or a descriptor based on a Whole Image SURF feature, WISURF [2]. These methods create manageable databases of manageable size for large- scale mapping, and can run in real time. The image matching process can be performed using Dynamic Time Warping (DTW) [15], [19], [22], or using a Bayesian filter [2], [6], [24].

2.2. Feature-based Methods

The above image matching methods for localization use a whole image descriptor for comparing the query and database images. Feature-based image matching approaches can overcome some of the limitations of using whole image similarity, providing stability in situations where occlusions obscure the camera view, or when the scene appearance changes, for example in lane changes. Techniques for image matching using features include monitoring the epipole position of features matched between query and database images [11]. When two images were captured from similar locations, the epipole position of matched features moves towards the outside of the image, and this can be used as a similarity measure within a DTW method. This system does however still require calculation of the essential matrix. Alternatively, the scale change between matched features can be compared and used as a similarity measure [23]. Feature points that have a size or scale property, such as the Scale Invariant Feature Transform (SIFT) [14], will vary in size depending on the distance from which they are captured by the camera. As the query image capture location becomes close to the corresponding database image, the scale of matched features will also become closer. The scale difference can be used as a cost measure in DTW [23].

2.3. Feature Detection and Description

Feature point detection and description in vehicle localization methods is most commonly based on the Scale In-

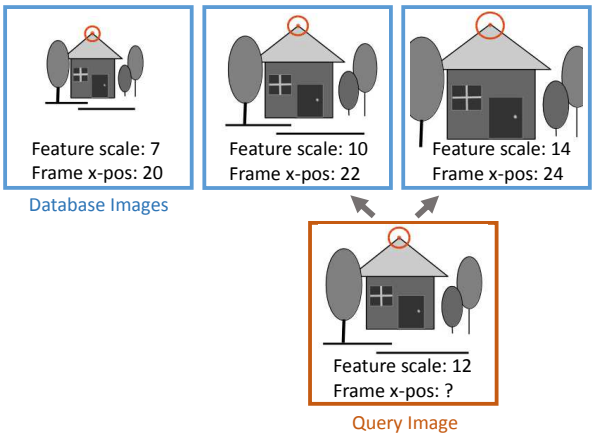

Figure 1. A simplified depiction of how a query image feature's scale can fall between corresponding feature scales in two database images. The basic concept in this paper is to interpolate the position between the database frames by using the relative difference in scale of corresponding features in the query image and each database image.

variant Feature Transform (SIFT) [14] or related methods. Speeded Up Robust Features (SURF) [3] offers some performance advantages in the detection and description of features. There have been an increasing number of feature point detection and description methods, each with various advantages in processing efficiency or robustness. Popular recent methods use a binary descriptor for fast detection and description. BRIEF [5], ORB [21], BRISK [13] and FREAK [1] are some modern binary descriptor methods, typically used with a FAST [20] type detection method, which incorporates machine learning into the corner detection process for determining good feature locations.

3. Concept of the Proposed Method

In this paper, we describe a method that extends localization methods that use image matching to a database. This process makes use of feature scale, so is most easily applicable to methods which already use scale-invariant features [11], [23] (and also a variant of the method presented by Badino *et al*. [2] which uses features instead of WISURF, as introduced in the same paper). In particular it is applicable to systems using feature scale [23]. The size, or scale, of extracted feature points is related to the distance between the capture location and the real-world position of the feature. As the camera moves towards the observed feature, the observed scale increases approximately linearly with the inverse of distance to the camera. This property is used in image matching methods, but here we make use of the continuous nature of feature scale change with distance to predict query image location without being bounded by the accuracy limitations of discrete image matching.

We present two methods: in Section 3.1 a very simple interpolation method for sub-database resolution accuracy applicable to all feature-based image matching systems, and in Section 3.2 a more sophisticated method which integrates with a feature scale based image matching method [23] to model scale change over database features.

3.1. Position Interpolation using Feature Scale

In most image matching methods for localization, when a query image is matched to a database image, the localization information from the selected database image is used directly. However, it is very unlikely that the actual location of the query image capture will correspond exactly to the selected database image. Even if the image match is correct, the true location will be somewhere between the selected database image and the next closest database image. This means that localization errors of half of the database spacing will always be potentially present, even with an ideal system that can perform perfect image matching.

The proposed method predicts the location of the query image by calculating the relative scale difference between features in the query image and two database images. Essentially, the position is linearly interpolated between the two known database image locations using feature scale. This basic concept is shown in Fig. 1. The predicted position is averaged over all features with correspondences in all three images.

3.2. Scale Modeling

The position interpolation using feature scale is extended by modeling feature scale changes within the database. This method is easiest to apply to an image matching method that pre-matches database features as it requires information about a database feature's appearance in preceding and succeeding database frames.

If consecutive database images have their features matched in the database construction process, the relative scales of the features is known and corresponding database features (together with their scale) can be arranged into the order of the database images they originated from. In the database construction phase, the vehicle is constantly moving forward so feature scales will also increase approximately linearly. Position interpolation using feature scale performs a linear fitting between two database points. Feature scale is not always completely stable, and the additional information provided by the corresponding features from adjacent database frames can be also useful for recognizing outliers. We model the error in scale change with forward motion as Gaussian, and perform a linear regression of database frame positions using feature scale. This allows more reliable position interpolation, and rejection of features of unstable scale from the position estimation.

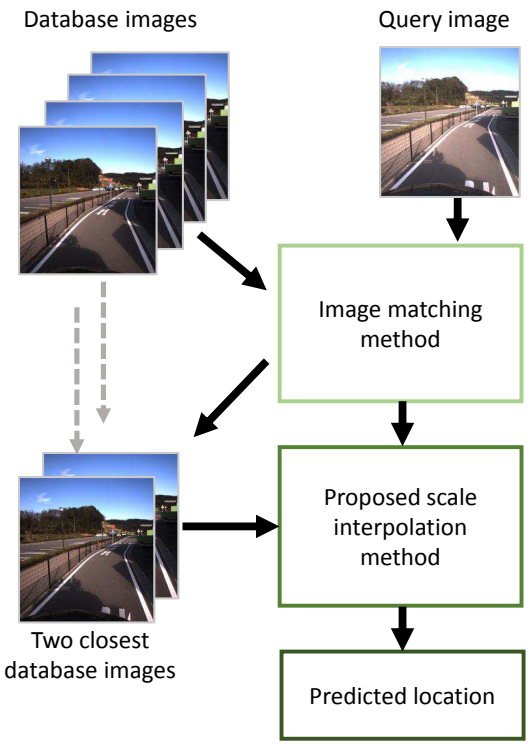

Figure 2. Overview of the processes in the proposed system.

4. Localization Method

In this section the proposed method is described in more detail. In Section 4.1 we explain the basic position interpolation method using feature scale, and the extension employing linear regression of database feature scales is described in Section 4.2.

4.1. Basic Position Interpolation

The basic interpolation method can be used in conjunction with any image matching method for localization. After the closest database image has been selected by the image matching method of choice, the adjacent database image is also selected such that the query image is between the two. Feature points with a scale property (SIFT, SURF, MSER, and so on) are detected, together with descriptors for the feature points (any of SIFT, SURF, BRISK, FREAK, and so on), if this has not already been performed by the image matching method. The query image features are then matched to each of the two database images using descriptor distance. Again, this step may have already been performed by the image matching method, so can be repurposed here. Only matches that are consistent in both database images are considered. The result is a set of N matched features, with the query image features denoted q_i, where $i = 1, 2, ..., N$. The database image features are denoted a_i and b_i for the frames behind and in front of the query image respectively. The position of the database images are given as $\boldsymbol{x}_a$ behind the query image, and $\boldsymbol{x}_b$ in front of the query image, with $\boldsymbol{x}$ containing the x and y co-ordinates on the horizontal plane. We can predict the position of the query image $\boldsymbol{x}_q$ by averaging the results of interpolation using feature scale over all the features as follows:

$$\boldsymbol{x}_q = \frac{1}{N}\sum_{i=1}^{N}\frac{(s(q_i) - s(a_i))(\boldsymbol{x}_b - \boldsymbol{x}_a)}{s(b_i) - s(a_i)} + \boldsymbol{x}_a. \quad (1)$$

Here the scale of a feature f is given by $s(f)$.

In this research we used a simple averaging of the locations predicted by the features. The result of the averaging process may be affected by features that have unstable scale change with capture distance. However, this problem can be avoided by employing a pruning method as proposed in [23]. For automotive localization, we know that the camera height is fixed, and vehicle motion is primarily in the forward direction. This allows the search for feature matches to be constrained in position and scale, helping to maintain an inlier feature set. When used with a set of un-pruned features, rather than averaging for predicted position, an outlier rejection method such as RANSAC [8] would improve results. Even over many features, the calculations in Eq. (1) are computationally inexpensive so an iterative process such as RANSAC would not be expensive.

An overview of the proposed method and how it relates to the image matching process is shown in Fig. 2.

4.2. Linear Regression using Feature Scale

By finding correspondences between features of consecutive database images, we can determine the ordered scale and positions of a particular feature of interest. For the basic interpolation method we used the feature scales of the closest database image and it's adjacent frame only; where feature scales do not vary constantly with capture distance, an inaccurate position prediction can result. If we model the scale error as Gaussian, we can perform a a multivariate linear regression for position given feature scale over all instances of the feature in the database. The interpolation then takes place on this regression line rather than between two points. This process includes the information from all corresponding feature points in the database rather than just the two adjacent database images, so is more robust to scale errors. If X is the $M \times 2$ matrix of capture coordinates $\boldsymbol{x}$ (in the horizontal plane) of M consecutively matched database features, and S is the $M \times 2$ design matrix containing the corresponding feature scale row vectors $\boldsymbol{s} = \begin{bmatrix}1 & s\end{bmatrix}$, then the coefficients for the linear regression Θ can be calculated by ordinary least squares as follows:

$$\Theta = (S^{\mathrm{T}}S)^{-1}S^{\mathrm{T}}X. \quad (2)$$

Additionally the sample variance, σ^2 of the set of feature positions with respect to scale can be estimated as:

$$\sigma^2 = \frac{\sum_{j=1}^{M} ||\boldsymbol{s}_j\Theta - \boldsymbol{x}_j||^2}{M}. \quad (3)$$

The sample variance to mean ratio varies depending on feature position. However, in order to classify the feature as a suitable contributor to the localization prediction, we can approximate a variance-to-mean ratio, VMR, using the position mean at the center of the regression line:

$$\mathrm{VMR} = \frac{\sigma^2}{\mu_m}, \quad \text{where} \quad \mu_m = \frac{||\boldsymbol{s}_1\Theta + \boldsymbol{s}_M\Theta||}{2}. \quad (4)$$

Feature sets with a VMR below a threshold are used for the interpolation process, and others are discarded. Following pruning, the predicted position of the query image can be found by averaging the results of individual feature set predictions on their regression lines:

$$\boldsymbol{x}_q = \frac{1}{N}\sum_{i=1}^{N} \begin{bmatrix} 1 & s(q_i) \end{bmatrix} \Theta_i. \quad (5)$$

The inter-matching of features between consecutive database frames and calculation of least squares coefficients can be performed offline so localization speed is not affected. The inclusion of linear regression coefficients for each feature set in the database has little effect on the database size compared to the feature descriptors.

5. Experimental Results

To evaluate the localization performance of the proposed method, a dataset captured at a driving school was used. The location allowed traversal of intersections and maneuvers to be safely performed away from traffic. A sophisticated data capture system was employed to provide accurate ground truth data in this dataset, which is described in Section 5.1. The method was tested with a database relative localization system using feature point scale to determine image matches [23], which allowed testing of both the basic interpolation method and the scale linear regression method. This process is described in Section 5.2 and results using the proposed method are presented in Section 5.3.

5.1. Mobile Mapping System

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 experiments. The localization hardware of the MMS was used to construct an image database and also to obtain a highly accurate ground truth for the query image set for evaluation.

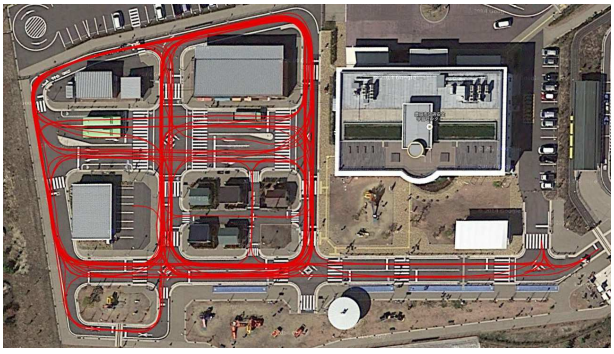

Figure 3. The dataset location, showing the driving paths used. Satellite imagery: Google, ZENRIN.

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. The MMS system captures images at approximately 2 m intervals. Fig. 3 shows the vehicle paths around the driving school. The query images and database images were captured on the same day at different times, giving some degree of lighting variations.

5.2. Image Matching Method

We used two different image matching methods to illustrate the performance of our method. Firstly, the WISURF whole image descriptor [2] method was used within a DTW framework to create a database and localize input frames. This method is included here to show a baseline for localization performance using image matching.

Next we implemented a feature scale [23] method which compares query image feature scales to corresponding database features within a DTW framework to find the closest database image. For the image matching localization process, an important consideration is the selection of feature detector type and descriptor type. After testing with a variety of modern feature extraction and detection techniques we found that the method providing the most stable feature scale was the original SIFT detector [14], but matching performance and speed was best when using the FREAK feature descriptor [1] together with a FLANN based matcher [18]. The results of localization using different feature detection and description types are discussed in Section 6. Although the native capture resolution of the MMS camera was 2,400 x 2,000 pixels, we found that localization performance was maintained after down-scaling to 600 x 500 pixels which improved computation speed. These experiments used a standard desktop computer with a 3.5 GHz Intel i7 processor. No GPU, multi-threading nor optimization was used to increase performance, and the OpenCV C++ library was used to implement feature detectors and descriptor extractors.

Table 1. Localization results of proposed and comparative methods. SIFT features use FREAK descriptors.

Method	Avg. error (m)	Variance	Max. error (m)
WISURF	3.71	2.83	4.61
SIFT scale only	0.66	0.15	3.46
SIFT scale + basic interpolation	0.51	0.15	2.96
SIFT scale + linear regression	0.45	0.17	2.84

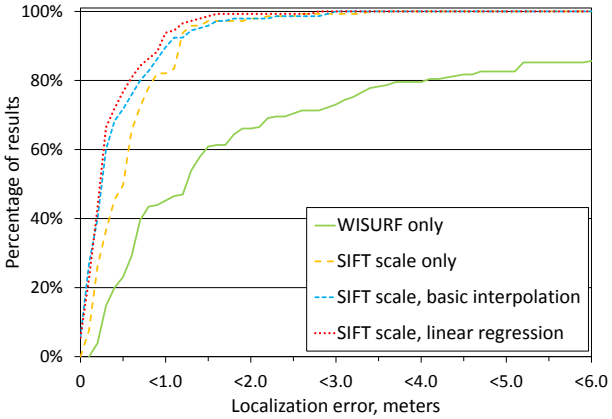

Figure 4. Localization error of the proposed method compared with standard image matching methods.

5.3. Localization Results

Localization of query images was performed using both image matching methods on the driving school dataset. Both the proposed basic interpolation method and the scale regression interpolation method were tested as an extension to the feature scale image matching method. Localization performance of all tested methods was evaluated by comparing localization results to the ground truth data provided by the MMS. A summary of the average localization errors of the various tested methods is presented in Table 1. The localization error rates of the different methods are shown in Fig. 4. The proposed method achieved an average localization error of 0.51 m when using basic interpolation, representing a 24 % improvement over the feature scale image matching method alone. When using the interpolation method with regression, the improvement over the feature scale image matching method increased to 33 % with an average localization error of 0.45 m. All methods using feature scale were significantly better than the baseline WISURF localization method in these experiments.

6. Discussion

While we tested many feature types, the original SIFT detector appeared to result in features with the most stable scale, providing robust image matching and position interpolation with feature scale. The results of testing of feature detectors and descriptors are shown in Table 2. Some other detection and description techniques, such as ORB, provided nearly as good results with much faster computation speed. A good compromise was provided by using the SIFT detector with binary FREAK descriptors. The fast binary descriptors allowed rapid detection and matching while reliable feature scale was provided by the SIFT detector. In these experiments, there was not a great deal of time spent on fine tuning individual feature extraction and detection parameters; in future work, further tuning of all methods will be tested. In addition, some of the theoretically fast feature detection and description methods were surprisingly slower than SIFT when used in the image matching system. Further investigation into the implementation details of these methods is required for speed optimization.

The image matching performance using SIFT scale with DTW and FREAK descriptors was very effective. Example image matching results can be seen in Fig. 5. The image matching method alone had an average localization error of 0.66 m, which shows close to perfect image matching since the database image spacing was approximately 2.0 m.

Interpolation of the query image position using feature scale was effective in all tested combinations of feature detector and descriptor types. In particular the SIFT detector appeared to produce features with enough scale resolution for interpolation to be effective. We only tested the regression technique with the SIFT detector and FREAK descriptor, but improvements with other feature types is likely as well.

While the database size varied significantly with the number of extracted feature descriptors, when SIFT features and descriptors were used it was around 90 MB per kilometer. The use of FREAK descriptors produced a database of half the size, at about 40 MB per kilometer. The MSER feature detector resulted in fewer matched features so created the smallest databases of as little as 7 MB per kilometer with FREAK descriptors. However, the smaller number of features also affected localization performance as can be seen in Table 2.

The dataset used in experiments is limited by the fact

Table 2. Results of feature descriptor and descriptor testing. Minimum values are shown in **bold.**

Extractor / descriptor	Avg. error, no interpolation (m)	Variance, no interpolation	Avg. error, basic interpolation (m)	Variance, basic interpolation	Avg. time per query frame (ms)	Avg. database size per frame (KB)
SIFT / SIFT	0.66	**0.15**	0.52	**0.15**	102	182
SIFT / SURF	0.67	0.24	0.52	0.26	69	303
SIFT / BRISK	**0.65**	0.22	0.55	0.23	188	98
SIFT / FREAK	**0.65**	0.25	**0.51**	0.16	65	91
SURF / SURF	0.66	**0.15**	0.57	0.19	153	582
SURF / BRISK	**0.65**	0.20	0.59	0.21	180	171
SURF / FREAK	0.69	0.26	0.65	0.29	101	150
BRISK / BRISK	1.74	3.59	1.73	3.54	605	55
ORB / ORB	0.98	0.68	0.86	0.61	**24**	102
MSER / ORB	0.70	0.45	0.62	0.45	88	**13**
MSER / FREAK	1.06	3.17	0.88	2.82	50	20

Figure 5. Example image matching results of query images (left) using the WISURF image descriptor (center) and SIFT feature scale (right). Current localization errors are shown in the bottom left-hand corner.

that a single camera type and mounting system was used for both query and database images. While it would be desirable to use a variety of configurations to test robustness, this was not possible with the hardware used. Nevertheless, this method uses feature scale only and not pixel position, so we predict that it will provide some robustness to uncalibrated cameras of varying types even when mounted at different positions.

7. Conclusion

In this paper we presented a method for using feature point scale to interpolate the position of a query image between the two closest database images. An extension of this method that models the scale of corresponding database features with a linear ordinary least squares regression was also presented. Both techniques were demonstrated to improve the localization accuracy of a feature-based image matching method which determines the position of a query image by finding the most similar database frame. The localization accuracy improved by an average of 24 % with a basic position interpolation using feature scale, and 33 % with the addition of linear regression, resulting in an average localization error of 0.45 m.

An experiment using different types of feature detectors and descriptors was also performed, in order to investigate the best combination for image matching and interpolation using feature scale. The original SIFT detector gave the best localization results, while the FREAK binary descriptor provided a faster alternative for feature description and matching. Even with the slower SIFT implementation and no speed optimization, real-time operation of around 15 Hz was achievable on standard hardware.

Future work will include further experiments with different feature and descriptor implementations, as well as testing of feature repeatability and scale robustness in changing lighting conditions. We also plan to test the method on larger databases in traffic environments with a wider variety of camera hardware and mounting positions.

Acknowledgments

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

References

[1] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast retina keypoint. In *Proc. 2012 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR2012)*, pages 510–517, 2012.

[2] H. Badino, D. F. Huber, and T. Kanade. Real-time topometric localization. In *Proc. 2012 IEEE Int. Conf. on Robotics and Automation (ICRA2012)*, pages 1635–1642, 2012.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[4] T. Botterill, S. Mills, and R. D. Green. Bag-of-words-driven, single-camera simultaneous localization and mapping. *J. Field Robotics*, 28(2):204–226, 2011.

[5] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary robust independent elementary features. In *Proc. 11th European Conf. on Computer Vision (ECCV2010), Part IV*, volume 6314 of *Lecture Notes on Computer Science*, pages 778–792, 2010.

[6] M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *Int. J. Robotics Research*, 30(9):1–24, 2010.

[7] H. F. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics and Automation Mag.*, 13(2):99–110, 2006.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[9] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Mag.*, 2(4):31–43, 2010.

[10] H. Kume, A. Suppé, and T. Kanade. Vehicle localization along a previously driven route using image database. In *Proc. 13th IAPR Conf. on Machine Vision Applications (MVA2013)*, pages 177–180, 2013.

[11] H. Kyutoku, T. Takahashi, Y. Mekada, I. Ide, and H. Murase. On-road obstacle detection by comparing present and past in-vehicle camera images. In *Proc. 12th IAPR Conf. on Machine Vision Applications (MVA2011)*, pages 357–360, 2011.

[12] H. Lategahn and C. Stiller. Vision-only localization. *IEEE Trans. Intelligent Transportation Systems*, 15(3):1246–1257, 2014.

[13] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *Proc. 2011 IEEE Int. Conf. on Computer Vision (ICCV2011)*, pages 2548–2555, 2011.

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Computer Vision*, 60(2):91–110, 2004.

[15] M. Milford. Visual route recognition with a handful of bits. In *Proc. 2012 Robotics: Science and Systems Conf.*, pages 297–304, 2012.

[16] M. Modsching, R. Kramer, and K. ten Hagen. Field trial on GPS accuracy in a medium size city: The influence of built-up. In *Proc. 3rd Workshop on Positioning, Navigation and Communication (WPNC2006)*, pages 209–218, 2006.

[17] F. Moosmann and C. Stiller. Velodyne SLAM. In *Proc. 2011 IEEE Intelligent Vehicles Symposium (IV2011)*, pages 393–398, 2011.

[18] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proc. 4th Int. Conf. on Computer Vision Theory and Applications (VISAPP2009)*, volume 1, pages 331–340, 2009.

[19] M. Muller. Dynamic time warping. In *Information Retrieval for Music and Motion*, pages 69–84. Springer, Berlin Heidelberg, 2007.

[20] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Proc. 9th European Conf. on Computer Vision (ECCV2006), Part I*, volume 3951 of *Lecture Notes on Computer Science*, pages 440–443, 2006.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proc. 2011 IEEE Int. Conf. on Computer Vision (ICCV2011)*, pages 2564–2571, 2011.

[22] H. Uchiyama, D. Deguchi, T. Takahashi, I. Ide, and H. Murase. Ego-localization using streetscape image sequences from in-vehicle cameras. In *Proc. 2009 IEEE Intelligent Vehicles Symposium (IV2009)*, pages 185–190, 2009.

[23] D. Wong, D. Deguchi, I. Ide, and H. Murase. Vision-based vehicle localization using a visual street map with embedded SURF scale. In *Computer Vision —ECCV 2014 Workshops Proc., Part I*, volume 8925 of *Lecture Notes on Computer Science*, pages 167–179, 2014.

[24] D. Xu, H. Badino, and D. F. Huber. Topometric localization on a road network. In *Proc. IEEE/RSJ 2014 Int. Conf. on Intelligent Robots and Systems (IROS2014)*, pages 3448–3455, 2014.