View-Based Localization in Outdoor Environments Based on Support Vector Learning

Proc. 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3083-3088, Edmonton, Aug. 2005.

Hideo Morita† Michael Hild‡ Jun Miura† Yoshiaki Shirai†
†*Department of Mechanical Engineering, Osaka University*
‡ *Department of Engineering Informatics, Osaka Electro-Communication University*
{morita,jun,shirai}@cv.mech.eng.osaka-u.ac.jp, hild@hilab.osakac.ac.jp

***Abstract*— This paper describes a view-based localization method using support vector machines in outdoor environments. We have been developing a two-phase vision-based navigation method. In the training phase, the robot acquires image sequences along the desired route and automatically learns the route visually. In the subsequent autonomous navigation phase, the robot moves by localizing itself based on the comparison between input images and the learned route representation. Our previous localization method uses an object recognition method which is robust to changes of weather and the seasons; however it has many parameters and threshold values to be manually adjusted. This paper, therefore, applies a support vector machine (SVM) algorithm to this object recognition problem. SVM is also applied to discriminating locations based on the recognition results. This two–stage SVM-based localization approach exhibits a satisfactory performance for real outdoor image data without any manual adjustment of parameters and threshold values.**

***Index Terms*— Outdoor mobile robot, Vision-based localization, Support vector machines.**

I. Introduction

Navigation in outdoor environments by robot vehicles has become an important problem to be solved. One of the key technologies is the *localization* of vehicles. Many approaches have been proposed so far. Distinctions among these approaches can be made with respect to whether an environment map is used or not (i.e. map–free approaches), or whether vehicle positions with respect to a scene coordinate frame are sensed and utilized (i.e. scene coordinate system–based approaches), or whether globally stationed non–vision–based sensors such as GPS are mainly used (e.g., [14]) or not (i.e. non–vision–based approaches). In this paper we take the stance that because GPS–based approaches are known to be unreliable in some situations and (metric) map–based approaches often require considerable efforts for creating and maintaining the maps, vision–based techniques are necessary.

Our approach is entirely vision–based. We assume that the robot vehicle is equipped with color video cameras which during a training run acquire image sequences along the desired route, automatically learn the route visually and store this learned representation of the route for subsequent autonomous driving. This approach is *map–free* and *coordinate system–free*. We have developed a navigation method using an object recognition method which is robust to changes of weather and seasons [8]. Similar two-phase approaches have been proposed for indoor [10], [9] and outdoor [13] environments, although they do not consider color changes of objects according to changes of weather and/or seasons and, thus, may not be robust enough. The most difficult part of this approach is finding the most appropriate internal representation (including feature selection) and an appropriate learning algorithm which is capable of generating this internal representation. Another vision-based approach is to fully rely on local visual features such as road boundaries [6], [3], but such features are not always available in real outdoor environments.

To date, several vision–based learning and representation methods have been proposed, but neither of them is completely automatic in the sense of not requiring the manual setting of threshold values and parameters. In our previous method [8] mentioned above, for example, each object model is defined by a set of possible ranges of image features such as color, edge density, and edge segment length, and such ranges were manually adjusted. However, such parameter adjustment represents a potential drawback for navigation systems, as this praxis limits the generality and applicability of systems.

Recently support vector machine (SVM) [15] has been successfully applied to several object recognition problems such as 3D object recognition [12], face recognition [4], [5], [11], and pattern matching-based tracking [1]. These methods exhibit better discrimination performances than previous methods such as PCA-based ones. Since our view–based localization problem can be interpreted as finding the most similar image to the current input image from a set of images learned during the training phase, the problem is a suitable application of the SVM-based recognition paradigm. Yamano et al. [16] used SVM in localization based on RFID signals, but this approach requires an environment setting and is difficult to apply to outdoor environments.

In this paper we describes a *vision–based navigation* system which is able to learn scenery views along a route automatically. It does not require manual setting of thresholds at all. After a feature extraction phase, feature vectors are learned with a *support vector machine* algorithm. During the navigation/localization phase, features are extracted in the same way and classified by the trained SVM, producing estimates of vehicle location along the route. This method is implemented as a two–stage process in which one SVM is employed for general scene feature learning and classification, while another SVM is used for learning and classifying scene locations based on the feature classification results from the first SVM.

In the following sections we first describe our system configuration in more detail, then discuss the support vector

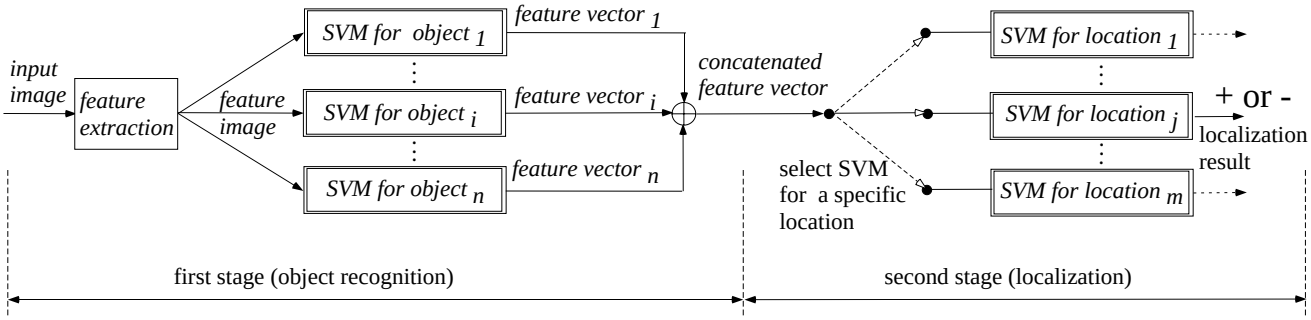

Fig. 1. Two-stage localization using SVMs.

learning and classification issues and present the results of experiments conducted in an outdoor campus environment.

II. Overview of the Method

This section gives an overview of our localization method. We first briefly explain the essence of support vector machines and then outline our two-stage approach to view-based localization.

A. Support Vector Machine

Support vector machine (SVM) [15] is a binary classification method that finds the optimal separating hyperplane based on the concept of margin maximization. For the case where the training data are linearly separable, there are in general several hyperplanes that can separate the two classes. SVM selects the hyperplane that maximizes the distance to the nearest samples in both classes.

Let $(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_N, t_N)$, $\boldsymbol{x}_i \in R^m$, $t_i \in \{-1, +1\}$ be the training vectors separated by a hyperplane $\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{h} = 0$. If the training data are linearly separable, there exist parameters $\boldsymbol{w}$ and $\boldsymbol{b}$ that satisfy:

$$t_i(\boldsymbol{w}^T\boldsymbol{x}_i - h) \geq 1, \quad (i = 1, \ldots, N). \tag{1}$$

For such parameters, the two classes are separated by two hyperplanes, $\boldsymbol{w}^T\boldsymbol{x} - h = 1$ and $\boldsymbol{w}^T\boldsymbol{x} - h = -1$, and no data exist between the hyperplanes. Since the distance between the hyperplanes is $\frac{2}{||\boldsymbol{w}||}$, the optimal parameters are determined by minimizing the objective function:

$$L(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{w}||^2 \tag{2}$$

under the constraint represented by eq. (1). A solution to this problem is as follows: First we introduce Lagrange multipliers α_i (≥ 1), $i = 1, \ldots, N$ and define the following Lagrange functional:

$$L(\boldsymbol{w}, h, \boldsymbol{\alpha}) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{N} \alpha_i\{t_i(\boldsymbol{w}^T\boldsymbol{x}_i - h) - 1\}. \tag{3}$$

At the saddle point, where $\partial L/\partial \boldsymbol{w} = 0$ and $\partial L/\partial h = 0$, the following relations hold:

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i t_i \boldsymbol{x}_i, \tag{4}$$

$$0 = \sum_{i=1}^{N} \alpha_i t_i. \tag{5}$$

Then we obtain the dual problem that is to be maximized:

$$L_D(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i\alpha_j t_i t_j \boldsymbol{x}_i^T\boldsymbol{x}_j \tag{6}$$

under the constraints:

$$\sum_{i=1}^{N} \alpha_i t_i = 0, \tag{7}$$

$$\alpha_i \geq 0, \quad i = 1, \ldots, N. \tag{8}$$

The training data $\boldsymbol{x}_i$ with non-zero α_i are on either one of the hyperplanes $\boldsymbol{w}^T\boldsymbol{x} - h = 1$ or $\boldsymbol{w}^T\boldsymbol{x} - h = -1$; such data are called *support vectors* because they are the only data that determine the parameters.

SVMs can be applied to non-linear separating surfaces using kernel tricks [15]. We use SVM^{light} [7] as the actual SVM software.

B. Two–Stage Localization

Fig. 1 shows the process of our SVM-based localization. The process is divided into two stages. At the first stage, objects in the image are recognized. Image features such as color and edge density are extracted from an input image and a feature image, and at each pixel a vector of such feature values is generated. This feature image is then sent to a set of SVMs, each of which is trained to recognize objects of a specific class. The output of an SVM is an image representing the location of the detected objects in the image. The output vectors are concatenated to produce the final recognition result. The change of object views due to changes of weather and seasons is handled at this stage, by training SVMs with object images taken under various conditions.

Given this recognition result, vehicle localization is carried out at the second stage. We train a set of SVMs, each of which can discriminate one given location from the others. The discrimination is based on the recognition results (i.e., the concatenated vectors) from the first stage, not on raw images, so that the localization becomes robust in outdoor environments.

To see if the robot is at a specific location, the input image is tested with the SVM trained for the location. When the robot follows the learned route, for example, the robot switches the SVMs for localization one after another. In this case, only one SVM is used at a time.

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

Fig. 2. Example images.

TABLE I

OBJECTS, REGION NAMES, AND IMAGE FEATURES USED.

object	region name	image features used			
		(r, g, b)	$f_{density}$	$f_{distrib}$	f_{hough}
trees with leaves	*tree region*	$\surd$	$\surd$		
trees without leaves	*tree region*	$\surd$	$\surd$	$\surd$	$\surd$
sky, building side walls	*uniform region*	$\surd$	$\surd$		
building windows and boundaries	*building region*	$\surd$	$\surd$	$\surd$	$\surd$

It may be possible to take the single-stage approach, that is to train SVM for localization directly from a set of images taken under various conditions. This approach, however, may require lots of training data, that is, images which display a sufficiently wide variation of weather and seasonal characteristics at *every* location. We, therefore, divide the problem into two stages: object recognition and localization using recognition results. What are required in this case are a sufficiently wide variation of object views for the first stage and a large number of locations for the second stage; this thus reduces the total size of the training data set.

III. SUPPORT VECTOR LEARNING FOR OBJECT RECOGNITION

A. Objects to be Recognized

We are interested in navigation in urban environments such as our campus, where buildings, trees, cars, bicycles, and other small objects exist. Fig. 2 shows some example images. Since views of cars and bicycles differ from time to time, we use buildings, trees, and the sky, which are relatively large and exist in the upper half of input images, as objects to be used for localization. We recognize the following four kinds of objects:

- Trees with leaves. Seasonal color changes of leaves are allowed.
- Trees without leaves. Only branches are observed.
- Sky and side walls of buildings that are observed as uniform regions in the images.
- Building windows and boundaries that are observed as strong straight line segments in the images.

B. Features Used for Object Recognition

We divide the upper half (304×128 pixels) of the image into a set of small windows (of 16×16 pixels), examine colors and edges within each window, and classify the windows into one of the above mentioned objects. Table I shows the relationship between the objects to be recognized by SVMs, the region names (labels), and the image features used. For trees, we prepare two object classes, but we assign the same label (*tree*) to the respective regions. The features are computed as follows:

1) (r, g, b) *:* This is the normalized color. Each component is calculated by, for example, $r = R/(R + G + B)$. The normalized color of each pixel is averaged over a small window to produce the color feature. The range of each component is $[0 : 1]$.

2) $f_{density}$ *:* This is the edge point density calculated by dividing the number of edges in a window by the area of the window. An edge is the pixel whose gradient magnitude calculated by Sobel operator exceeds 3. This threshold was determined by estimating the noise level of the image capturing system with the camera being pointed at a white paper target.

3) $f_{distrib}$ *:* This feature measures the degree of distribution of edge directions and is useful for recognizing trees without leaves, because branches produce edges in various directions. Since the edge direction value is cyclic, we employ *circular statistics* [2] for calculating the variance of the distribution. Consider the case where there are n edges in a window, and their direction and magnitude are given by $\phi_1, \ldots, \phi_n$ and $a_1, \ldots, a_n$, respectively. We first convert each direction value into a point on the unit circle and calculate the averaged position $(\bar{x}, \bar{y})$, weighted by the edge magnitudes, as follows:

$$(\bar{x}, \bar{y}) = \frac{1}{\sum_{i=1}^{N} a_i} \left(\sum_{i=1}^{n} a_i \cos 2\phi_i, \sum_{i=1}^{n} a_i \sin 2\phi_i \right). \quad (9)$$

Here we consider the value of $d = (\bar{x}^2 + \bar{y}^2)^{1/2}$; d becomes smaller when edge directions are more diverse, and its value range is $[0 : 1]$. So we use $S = 1 - d$ as the directional variance; that is, $f_{distrib} = S$.

4) f_{hough} *:* This is the maximum value of voting in the hough space for the edge points in a window. This value becomes larger where strong line segment such as building windows and boundaries exist. We divide the maximum value by its empirically-obtained largest possible value (currently, 300) for normalization.

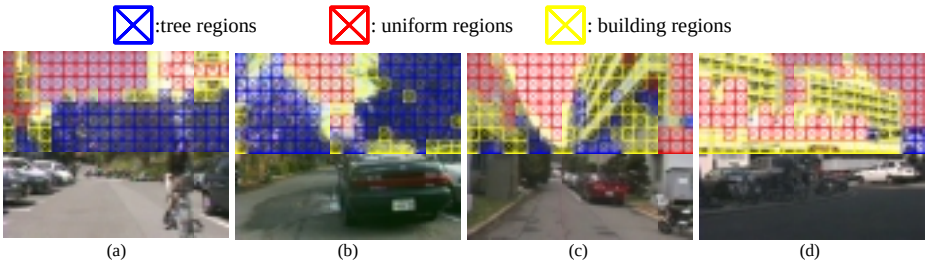

Fig. 3. Recognition results.

The sextuplet of the above feature values are obtained for each window. An input image is thus converted into a 19×8 array of the sextuplet. This array, called a *feature image*, is the input to the SVMs for object recognition (see Fig. 1); each SVM uses four or six components of the sextuplet for recognition (see Table I).

C. Training SVMs for Object Recognition

We use one SVM for each object class. In order to collect training data, we examined image data captured on our campus in various annual seasons and under various weather conditions, and manually selected, for each object class, about 300 windows for which only the single object class was present per window. These windows were then converted into the sextuplet of feature values. We finally have four sets of sextuplets for four object classes.

In order to train one SVM per class, we use all sextuplets (or quadruplets in the case of "tree with leaves" and "uniform regions") of the corresponding set as positive samples, and randomly select the same number of negative samples from all windows not containing positive samples. We use the SVM with RBF kernel ($K(\boldsymbol{x}_1, \boldsymbol{x}_2) = exp(-\gamma||\boldsymbol{x}_1 - \boldsymbol{x}_2||)$, $\gamma = 50$) for object recognition.

Each SVM receives a sextuplet of feature values and returns value 1 (if the output is positive) or zero (otherwise). Since the size of feature images is 19×8, the SVM produces a 152-dimensional 0-1 vector, called a *feature vector* (see Fig. 1).

D. Recognition Results

Fig. 3 shows recognition results for the example images shown in Fig. 2. Each block with $\times$ mark indicates the recognition result (tree, uniform, or building) for a window. We compared these results with the ones obtained by our previous method [8] and found that the results of both methods are comparable. Our new approach, however, has the big advantage of not using any parameters and threshold values to be adjusted.

IV. Support Vector Learning for Localization

The second stage performs localization by SVMs using the object recognition results of the first stage (see Fig. 1). The first stage outputs three feature vectors (152-dimensional 0-1 vectors) because we have three kinds of labels, *tree*, *uniform*, and *building* regions (see Table I). We concatenate the vectors into one 456-dimensional 0-1 vector and use it as the input to the SVMs for localization.

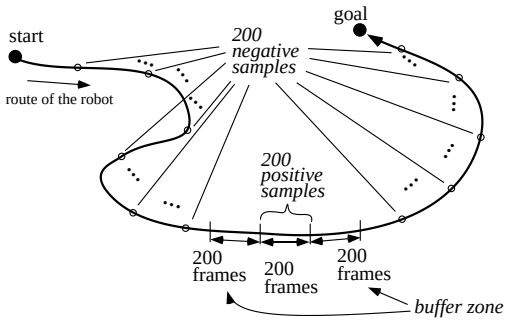

Fig. 4. Making training data for localization SVM.

A. Generating Training Data for SVM learning

We prepare one SVM for each specific location, set along the robot's route. Each SVM is trained by declaring the data taken near the location as positive samples and the data at other locations as negative ones. The detailed process of generating training data is as follows.

Image data are captured at the rate of $30\,[frames/s]$ while the robot is moving at a speed of about $0.9\,[m/s]$. For a given location, we select 200 consecutive images taken around the location (this roughly corresponds to the movement of $6\,[m]$), and then generate 200 concatenated feature vectors as positive samples. We set *buffer zones* before and after the positive samples, and pick up 200 images from the remaining frames in regular intervals to produce the 200 negative samples (see Fig. 4). These positive and negative samples are used to train the SVM for the location. For other locations, we perform the sample selection and learning in the same manner. We use the linear SVM for localization.

B. Localization by SVM

An SVM outputs positive values if the input is judged as a positive sample. To see if the robot is at a given location, we give the concatenated feature vector, generated from the current input image, to the SVM for that location and see if its output is positive. From our experiences, however, we know that SVMs for localization have very noisy outputs and produce frequent erroneous positive values. We, therefore, average the output values (which are signed distances from the separating hyperplane) within a certain interval of frames (currently, 21 frames centered at the current frame) and use the averaged value, which we call the *score*, for localization.

C. Criteria for Evaluating Localization Performance

There are two cases of localization using SVMs. If the robot knows its rough location because it has been moving along a trained route, it only needs to test the current input with the SVM for the predicted position (or a few SVMs for the locations near the predicted position) and to confirm the score is positive. On the other hand, if the robot has no knowledge of its location, it has to test the current input with *all* SVMs to see which one outputs a positive score. If multiple SVMs output positive scores, the highest score may indicate the most probable location.

Considering the above two cases, we use the following two statistical measures of the performance for the SVM-based localization method:

1) *Recognition ratio*: the ratio of numbers of locations that are correctly recognized by the SVMs in charge of the locations versus the total number of locations. This applies to the first case.
2) *Highest-score ratio*: the ratio of the number of locations at which the positive *and* the highest scores are obtained by the SVMs in charge for the locations versus the total number of locations. This applies to the second case.

D. Localization Experiments

Fig. 5 shows the route of about $350\,[m]$ length used for our experiments. The robot moved from Start to Goal at the speed of about $0.9\,[m/s]$ and captured images at the frame rate of $30\,[frames/s]$ The number of images for one run is about 12,000.

We obtained two image sets, one for training and another one for testing. The training image set was obtained on Nov. 13, 2004 at 4pm (sunny), by pushing the robot vehicle along the path by hand. The test image set was obtained on Dec. 20, 2004 at 2pm (cloudy), by controlling the robot with the joystick interface to the steering control system. The differences between the image sets are summarized as follows:

1) Many trees had already dropped all their leaves in the test set, while they had not in the training set.
2) The views of objects are different due to different weather.
3) The robot orientation is fairly aligned to that of the route for the training set, because we pushed the robot by hand, whereas the robot orientation exhibits some variation in the case of the test set, because we controlled the robot with the joystick.

We selected 50 locations on the route and trained SVMs for them using 200 positive samples and 200 negative ones as explained above. We tested about 12,000 test images against the 50 SVMs. For each frame, the score was calculated as the average of the SVM outputs for 21 consecutive images centered at a given frame. Fig. 6 shows a result of localization. We calculated the scores for all frames for the SVM trained at location A. Since the robot passed location A twice, the graph exhibits two distinctive positive-scored regions. This result shows that the robot can correctly recognize location A.

We next describe some statistical results. As a preliminary experiment, we first verified the SVMs by testing them against the training image set and found that both criteria recorded 100%; this shows the potential ability of SVM-based localization. Concerning the result for the test image set, the recognition ratio was 88% and the highest-score ratio was 78%.

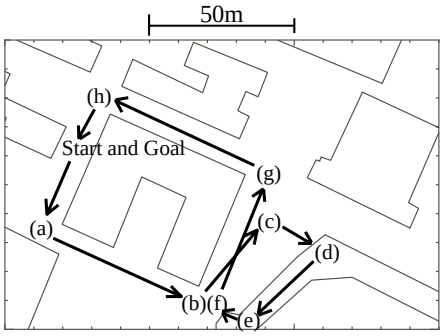

Fig. 5. The route used for experiment.

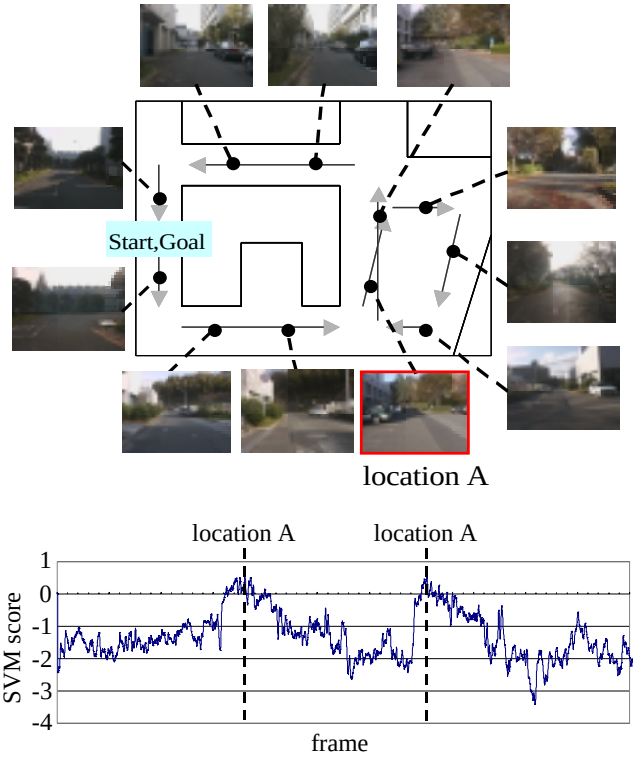

Fig. 6. Result of a localization experiment in our campus.

We then compared this result with the one obtained by our previously published localization method [8] for the same route. The previous method measures how well the regions of each object are matched between the learned and the input image, and decides the success of matching using a threshold for the measured value. The recognition ratio was 95%, but the highest score ratio was only about 57%. This comparison implies that the previous method uses a relatively low threshold for a high recognition ratio at the cost of a lower highest-score ratio. The proposed SVM-based method, however, exhibits at least a comparable performance without using any parameters and threshold

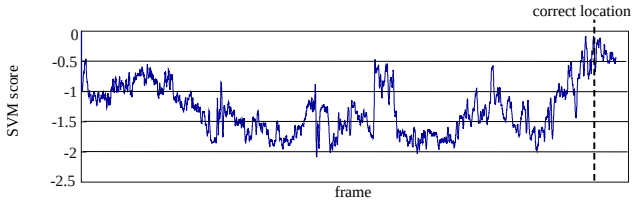

Fig. 7. Localization result when the correct position does not get a positive score but the highest one.

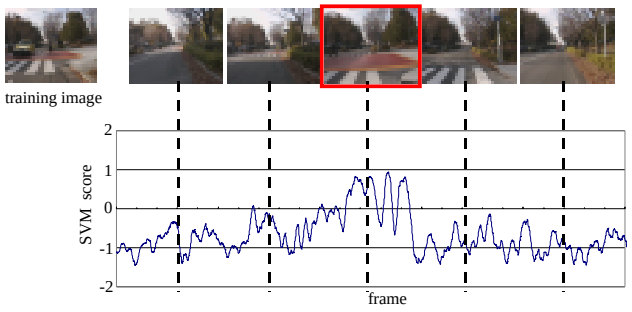

Fig. 8. Localization result when views are similar.

values. This is the most important point of the new method.

We then analyzed those locations where we did not get a positive score. Fig. 7 shows such a case. The score for the correct position is not positive but is very close to the positive domain and reaches the highest score there. This result indicates that it may be possible to base the localization decision on the "highest score", rather than the absolute positive score. In the case where the robot moves on a learned route, such a consideration seems to be helpful for guessing the robot's probable locations. The use of the non-positive highest score for localization is to be further developed in the future.

We also performed the experiment on another route of about $1\,[km]$ length. In this experiment, both the training and the test image sets were taken on Jan. 13, 2005 (sunny) around 3pm. The route includes several long paths with little scene changes along them. Fig. 8 shows the result of localization for such a path. The SVM used was learned for the location indicated by the training image shown on the top left of the figure. The graph shows the changing SVM score along the path. The images for five selected positions are indicated. Although these images are quite similar to each other (note that only the upper part of the image is used for localization), the correct location is identified. For 50 selected locations, the recognition ratio was 100% and the highest-score ratio was 92%. Such a high performance is obtained probably because the training and the test image set were taken at almost the same time, although the robot orientation was not exactly the same.

V. Conclusions and Discussion

This paper describes a novel localization method in outdoor environments based on support vector learning. The method employs a two–stage process in which one SVM is employed for general scene feature learning and classification, while another SVM is used for learning and classifying scene locations based on the feature classification results from the first SVM. The method has been tested in real outdoor scenes and exhibits a performance comparable with our previous method, but without using any parameters and threshold values to be adjusted; the previous method had to adjust lots of parameters and thresholds.

The previous localization method was successfully combined with a navigation method including robot control and local collision avoidance capabilities to enable the robot to move autonomously along learned routes. We are currently working on using the new method in the place of the previous localization method for on-line navigation experiments.

References

[1] S. Avidan. Support vector tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 8, pp. 1064–1072, 2004.

[2] E. Batschelet. *Circular Statistics in Biology*. Academic Press Inc., London, 1981.

[3] J.D. Crisman and C.E. Thorpe. Scarf: A color vision system that tracks roads and intersections. *IEEE Trans. on Robotics and Automat.*, Vol. 9, No. 1, pp. 49–58, 1993.

[4] G. Guo, S.Z. Li, and K. Chan. Face recognition by support vector machines. In *Proceedings of the 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 195–201, 2000.

[5] B. Heisele, P. Ho, and T. Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Proceedings of the 8th Int. Conf. on Computer Vision*, pp. 688–694, 2001.

[6] H. Ishiguro, K. Nishikawa, and H. Mori. Mobile robot navigation by visual sign patterns existing in outdoor environments. In *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 636–641, 1992.

[7] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*. The MIT Press, 1999.

[8] H. Katsura, J. Miura, M. Hild, and Y. Shirai. A view-based outdoor navigation using object recognition robust to changes of weather and seasons. In *Proceedings of 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2974–2979, 2003.

[9] K. Kidono, J. Miura, and Y. Shirai. Autonomous visual navigation of a mobile robot using a human-guided experience. *Robotics and Autonomous Systems*, Vol. 40, No. 2-3, pp. 121–130, 2002.

[10] Y. Matsumoto, M. Inaba, and H. Inoue. Visual navigation using view-sequenced route representation. In *Proceedings of 1996 IEEE Int. Conf. on Robotics and Automation*, pp. 83–88, 1996.

[11] T. Okabe and Y. Sato. Support vector machines for object recognition under varying illumination conditions. In *Proceedings of 2004 Asian Conf. on Computer Vision*, pp. 724–729, 2004.

[12] M. Pontil and A. Verri. Support vector machines for 3d objec recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 6, pp. 637–646, 1998.

[13] Y. Takeuchi and M. Hebert. Evaluation of image-based landmark recognition techniques. Technical Report CMU-CS-95-116, The Robotics Institute, Carnegie Mellon University, July 1998.

[14] R. Thrapp, C. Westbrook, and D. Subramanian. Robust localization algorithms for an autonomous campus tour guide. In *Proceedings of 2001 IEEE Int. Conf. on Robotics and Automation*, pp. 2065–2071, 2001.

[15] V.N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.

[16] K. Yamano, K. Tanaka, M. Hirayama, E. Kondo, Y. Kimuro, and M. Matsumoto. Self-localization of mobile robots with rfid system by using support vector machine. In *Proceedings of 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3756–3761, 2004.