Intelligent Vehicles Symposium (IV 2011)
Baden-Baden, Germany, June 5-9, 2011

H. Badino, D. Huber and T. Kanade, "Visual Topometric Localization," Intelligent Vehicles Symposium (IV 2011), Baden-Baden, Germany, 2011.

Visual Topometric Localization

H. Badino, D. Huber and T. Kanade
Robotics Institute, Carnegie Mellon University
Pittsburgh, PA 15213, USA

***Abstract*— One of the fundamental requirements of an autonomous vehicle is the ability to determine its location on a map. Frequently, solutions to this localization problem rely on GPS information or use expensive three dimensional (3D) sensors. In this paper, we describe a method for long-term vehicle localization based on visual features alone. Our approach utilizes a combination of topological and metric mapping, which we call topometric localization, to encode the coarse topology of the route as well as detailed metric information required for accurate localization. A topometric map is created by driving the route once and recording a database of visual features. The vehicle then localizes by matching features to this database at runtime. Since individual feature matches are unreliable, we employ a discrete Bayes filter to estimate the most likely vehicle position using evidence from a sequence of images along the route. We illustrate the approach using an 8.8 km route through an urban and suburban environment. The method achieves an average localization error of 2.7 m over this route, with isolated worst case errors on the order of 10 m.**

I. INTRODUCTION

One of the fundamental requirements of an autonomous vehicle is the ability to determine its location on a map. Frequently, solutions to this localization problem rely on GPS information or use expensive three dimensional (3D) sensors. While GPS seems to offer a simple, low-cost solution, the GPS signal is unavailable in many situations. For example, in downtown areas, tall buildings can block the GPS signal, rendering GPS-based vehicle navigation systems useless. Similar situations can arise in suburban environments, where trees can block the GPS signal, or in military situations, where the GPS signal can be jammed. Thus, for practical autonomy in many environments, a vehicle must be able to localize in GPS denied situations.

Some researchers have had success in localization using 3D sensors. Most visibly, the Google Car [1] and some entries in the DARPA Urban Challenge competition [2], [3] use 3D sensors to help localize the vehicle. Such sensors are effective for this purpose, but the scanning laser range finders used in these demonstrations can cost tens of thousands of dollars and may not be sufficiently durable due to rapidly moving internal parts. To be practical for autonomous vehicles, we need a more cost effective sensor that can operate for long periods of time.

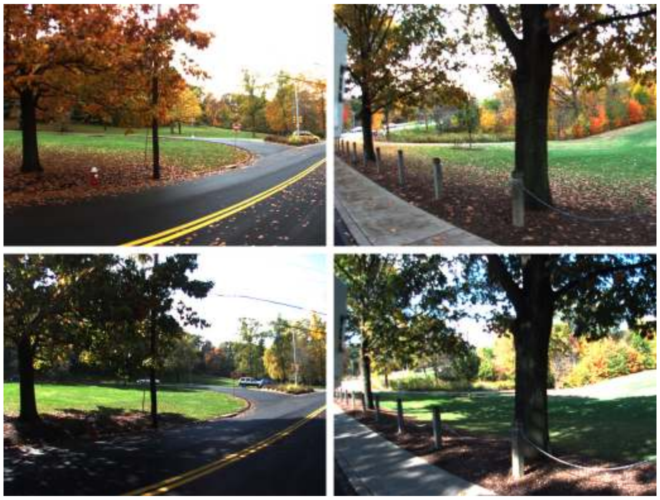


Fig. 1. Example of long-term localization on a natural varying environment. The figure shows left and right views for query (top) and matched (bottom) images.

In this paper, we use vision-based sensing for long-term localization (Fig. 1). Our approach uses a single, low-resolution video camera to capture the appearance of a route, and then localizes the vehicle with respect to this original route. Storing and maintaining a database of route appearance may seem like a daunting task, but already we have a proof of concept in the Google Streetview project [4], which has mapped most of the roads in the major cities in the U.S. and throughout the world. It is easy to envision storing a visual database like the Streetview maps for the purpose of visual localization.

The literature on visual localization focuses on two main types of approaches: *metric* and *topological* localization. Metric localization is achieved by computing the coordinates of the location of the observer (e.g., latitude and longitude). The coordinates of the vehicle pose are usually obtained by triangulation (e.g., structure from motion [5]) or alignment (e.g., occupancy grids [6]). With topological localization [7], the position of the observer is found from a finite set (typically small) of possible locations. Such methods provide coarse localization information such as “I’m in the kitchen.” Topological maps are usually represented by graphs, where nodes indicate possible locations (e.g., the kitchen) and edges are connections between locations. The weight of an edge indicates the similarity or proximity between locations (e.g., the kitchen is close to the dining room). Metric approaches provide accurate localization results, but tend to fail and to drift over time as the vehicle moves large distances in its environment. Due to its finite state space, topological approaches provide a robust localization but only rough position estimates.

In this paper, we propose a fusion of the metric and topological approaches in order to achieve accurate metric results while maintaining the robustness of topological matching. We call this hybrid approach *topometric localization*. To accomplish the topometric localization, we use a fine-grained topological map, where each node has an associated coordinate of its real metric location (Fig. 2c). Finding the node of the current location also means finding the metric coordinate of the observer. Our focus in this paper is on long-term, robust localization in large outdoor environments, but the method could be applied equally to indoor localization tasks. Our topometric localization algorithm involves two main stages:

Creation of the map (Section III). A vehicle equipped with cameras and GPS first travels the route to be recognized. GPS and inertial sensor units are used to create a graph of the environment. The graph is metric in the sense that the nodes contain the exact location of the vehicle. From the acquired images, visual local features are extracted. Each feature is stored in a database with a reference to the node corresponding to its real location.

Localization (Section IV). At runtime, the vehicle drives over the routes included in the a priori map. Video imagery is processed online to obtain features. A Bayes filter estimates the probability density function of the position of the observer as the vehicle drives and features are matched with those in the database.

II. RELATED LITERATURE

The literature on visual localization is huge. We will give here only a small sample of the most related approaches.

Most visual localization method rely on the extraction of local features from the images to build a visual database of the environment. Valgren and Lilienthal [7] evaluated the use of SIFT and SURF features for long-term topological localization. The authors used high-resolution panoramic images acquired over long periods of time to capture the natural seasonal changes in an outdoor environment. A comparison of results using variants of the SURF and SIFT features was performed. The authors concluded that SURF performs better than SIFT for the purpose of localization in outdoor environments. Ascani et al. [8] came to a similar conclusion, finding also that SIFT performs better in indoor environments. SIFT features were also used indoors using a topological approach by Andreasson et al. [9] and a metric approach by Se et al. [10]. Murillo et al. [11] proposes a two step approach. First, SIFT features are used to obtain the topological location of the observer. Second, a refinement of the location is obtained by computing the trifocal tensor between the best and second best database image matches. Silveira et al. [5] proposed a metric localization approach without explicit visual local feature detection. Although all these methods show promising localization results, the problem was approached as pure topological or metric, and the size of the maps varied from small (a few dozen images) to relatively small (less than 2500 images.)

The fusion of topological and metric localization has also been approached before mainly in the simultaneous localization and mapping (SLAM) domain and using 3D sensors. Tomatis et al. [12], Kouzoubov and D. Austin [13], Bosse et al. [14], and Blanco et al. [6] used hybrid approaches to connect local metric submaps using highlevel topological maps. In all cases, the fusion aims at segmenting metric maps represented by topological nodes in order to organize and identify submap relations and loop closures. Instead of relying on 3D sensor to built metric maps, our proposed topometric approach integrates metric data directly into the topological nodes by creating a data base of the route, as adresssed in the next section.

III. CREATION OF THE MAP DATABASE

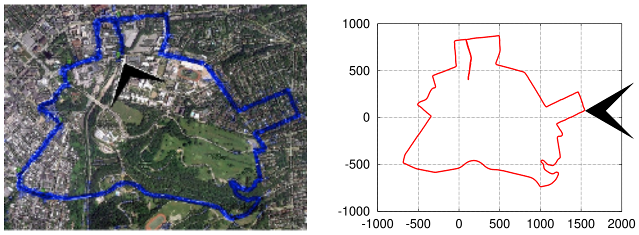
In this section, we address generation of the the route map and the feature database.

A. Route Map

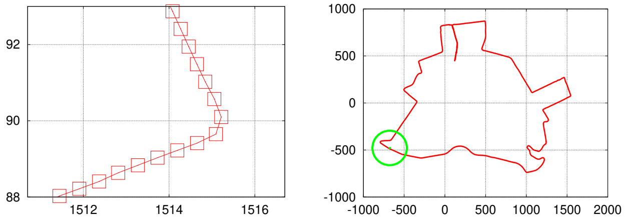
The route map is created as a directed graph. The vertices of the graph represent locations; the edges represent transitions between locations (Fig. 2b). The map is created with a constant Euclidean distance between nodes (Fig. 2c), which we call the sampling factor ρ. The advantage of having nodes separated at constant distance is that it is simple to predict the vehicle position within the graph by just using its estimated displacement. If the vehicle moves with speed s between two consecutive image acquisitions, its displacement on the graph is $s\Delta t/\rho$ nodes, where Δt is the cycle time (i.e., time elapsed between the system updates).

B. Visual Information Database

The visual information database is a list of features extracted from the environment. The features are stored together with their corresponding ground truth location – obtained with GPS – on the map. We use the term *feature position* to denote the position of the vehicle when the feature was observed. If the same feature is found in a future run,



(a) *Route of the vehicle.* (b) *Created map data.*



(c) *Corner marked in 2b* (d) *Ground truth data of test run.*

Fig. 2. Created map and ground truth data for evaluation. The units are shown in meters. The arrow in 2a shows the start and end position. Figure 2c amplifies the corner of the route marked in 2b. The circled area in 2d shows the location of the construction site during the test run.

the likelihood that the vehicle is located on the position referenced by the database feature is increased.

A number of feature descriptors could be used for localization [15]. Among these, SURF features have shown to be robust in outdoor environments [7] [8]. In this paper, we use the upright SURF (U-SURF) feature [16], which is invariant to scale and to rotations of the vertical axis. A descriptor vector $\boldsymbol{d} \in \mathbb{R}^{64}$ is built from each detected feature as specified in [17]. The descriptor contains gradient image information of the neighborhood surrounding the feature. The descriptors are used to find matches between two sets of features: the set of features of the database having a common graph location (l_i) and the set of features extracted from the current view at runtime. The matching is performed with the nearest neighbor method as in [18].

The visual information database consists of the set $\mathcal{D} = \{\boldsymbol{r}_i, \boldsymbol{r}_2, ..., \boldsymbol{r}_n\}$ with components $\boldsymbol{r}_i = \{\boldsymbol{d}_i, l_i, s_i\}$, where $\boldsymbol{d}_i$ is the U-SURF feature descriptor vector, l_i is the location of the feature in the map, and s_i is the velocity of the vehicle when the feature was observed.

IV. LOCALIZATION

In this section, we present a Bayesian approach to the localization problem. The localization is performed using a discrete Bayes filter, which tracks the probability density function of the vehicle position as it advances along the route and new measurements are acquired.

A. Discrete Bayes Filter

The state X_t defines the position of the vehicle in the map at time t (i.e., the graph vertex x_k where the vehicle is located). The probability of the vehicle located at a particular position x_k in the graph is specified as $p(X_t = x_k)$ or simply $p(x_k)$. The vertices of the graph define the universe of possible values that X_t might assume, i.e., $x_k, k = 1, 2, 3, ..., N$. A *discrete* Bayes filter [19] makes a partitioning of the state space, assigning an individual probability $p_{k,t}$ to each vertex x_k of the graph. This is equivalent to a histogram, where x_k represents a bin and $p_{k,t}$ its value. Thus, the probability $p_{k,t}$ specifies the belief that the vehicle is located at position x_k of the graph at time t. The discrete probability distribution is then expressed as the set $p(x_k) = \{p_{k,t}\}$. The Bayes filter keeps track of the probabilities as the vehicle moves and new measurements are acquired.

There are two actions that modify the probability density function: vehicle motion and measurement. The incorporation of the motion into the probability density function is called *prediction* and is specified as $\bar{p}_t = p(x_k | z_{1:t-1}, u_{1:t})$. The input $u_{1:t}$ specifies all motion controls since initialization (e.g., velocity of the vehicle); the term $z_{1:t-1}$ includes all the measurements up to previous time $t-1$.

The second action that modifies the state of the system is the measurement. The incorporation of a measurement z_t obtained at time t leads to the following posterior distribution: $p_t = p(x_k | z_{1:t}, u_{1:t})$. This is known as an *update* of the system and it incorporates both the controls and measurements up to the current time.

The discrete Bayes filter specifies the required operations to perform these two steps recursively while considering the discrete nature of the state space. The algorithm is as follows.

```
Input: $\{p_{k,t-1}\}, u_t, z_t$
Output: $\{p_{k,t}\}$
foreach $k$ do
    # Predict
    $\bar{p}_{k,t} = \sum_{i=1}^{N} p(X_t = x_k | u_t, X_{t-1} = x_i)\; p_{i,t-1}$;
    # Update
    $p_{k,t} = \eta \; p(z_t | X_t = x_k) \; \bar{p}_{k,t}$
end
```

The probability $p(X_t = x_k | u_t, X_{t-1} = x_i)$ of the prediction step is called *state transition probability*. It specifies how the state evolves over time as the vehicle moves in the environment.

The update step requires the *measurement probability* $p(z_t | X_t = x_k)$, which specifies how measurement probabilities are generated from the system state x_k. The scalar η ensures that the resulting probability density function integrates to one.

The discrete Bayes filter requires the definition of the state transition probability and the measurement probability. It also requires the initial probability density function $p_0 = \{p_{k,0}\}$. The following sections are dedicated to derive these functions for our localization problem using SURF features.

B. State Transition Probability

We will assume that the vehicle moves with velocity s_t at time t and that the velocity s_t is a zero-mean noisy measurement with variance σ_s^2. The translation of the vehicle between times $t-1$ and time t is then $d_t = s_t \Delta t$ with a variance of $\sigma_d^2 = \Delta t^2 \sigma^2$. Based on these considerations, the control input u_t of the system is only the velocity s_t, and the state transition probability is defined with a Gaussian probability density function (Fig. 3a):

$$p(x_k | u_k, x_{k-1}) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{(x_k - \mu)^2}{2\sigma^2}\right) \tag{1}$$

where $\mu = -s_t \Delta t / \rho$ and $\sigma^2 = (\rho \Delta t \, \sigma_s / \rho)^2$.

C. Measurement Probability

The measurements are given by the location of the SURF features matched between the current view and those stored in the database. SURF matches provide evidence of the vehicle at a specific location. The challenge here is to find the probability density function of the measurements given the state.

The set of SURF features extracted from the current view is matched with those in the set $\mathcal{D} = \{\boldsymbol{r}_i, \boldsymbol{r}_2, ..., \boldsymbol{r}_n\}$. Each component $\boldsymbol{r}_i = \{\boldsymbol{d}_i, l_i, s_i\}$ corresponds to a SURF feature with an associated location l_i and the velocity s_i of the vehicle while acquiring the feature. The measurement required for our Bayesian method is the location l_i of the matched feature.

We compose the probability of the measurement by a combination of two density functions. The first probability density function models the expected distribution of the feature location along the route. We assume that the measurement l_i of a matched feature is a noisy observation of the true feature location. Furthermore, the accuracy of the measurement decreases with increasing vehicle speed due to factors such as synchronization, timing, and registration inaccuracies. We model this effect with a uniform distribution over an interval proportional to the vehicle velocity (Fig. 3b). Specifically, we model the probability of measuring a match at location $z_t^i = l_i$ as

$$p_{match}(z_t^i|x_k) = \begin{cases} \eta_1 & \text{if } \ (z_t - x_k) \leq \dfrac{(s_i\,\Delta t)}{\rho} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where η_1 is a normalization factor to make probability density function integrate to one.

The second density function accounts for the effect of incorrect matches. We model this as a uniform distribution over the entire measurement space (Fig. 3c):

$$p_{rand}(z_t|x_k) = \eta_2 \tag{3}$$

where η_2 is a normalization factor. Incorrect matches arise most commonly from false positives, which occur mainly because of noise and model inaccuracies. Incorrect matches can also occur when the feature matches the correct object but at a different (and therefore incorrect) location along the route. This can happen, for example, when the same vehicle is parked at a different location along the route.

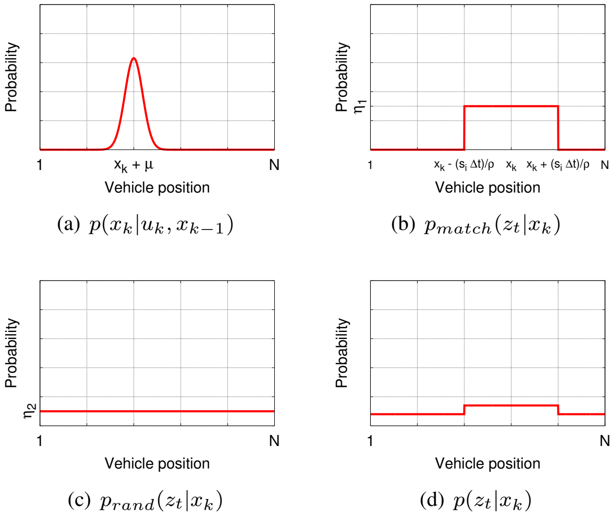
The combined distribution $p(z_t|x_k)$ is a weighted average of both density functions:

$$p(z_t|x_k) = \begin{bmatrix} z_{match} \\ z_{rand} \end{bmatrix}^T \begin{bmatrix} p_{match}(z_t|x_k) \\ p_{rand}(z_t|x_k) \end{bmatrix} \tag{4}$$

with weighting factors $z_{match} + z_{rand} = 1$ (Fig. 3d).

Equation 4 defines the probability density function for one single match. If, for a given time t, there are K matches, then the final probability function is obtained (assuming the independence of the measurements) by the product of the individual probabilities, i.e.,

$$p(z_t|x_k) = \prod_{i=1}^{K} p(z_t^i|x_k). \tag{5}$$



(a) $p(x_k|u_k, x_{k-1})$

(b) $p_{match}(z_t|x_k)$

(c) $p_{rand}(z_t|x_k)$

(d) $p(z_t|x_k)$

Fig. 3. State transition and measurement probability density functions.

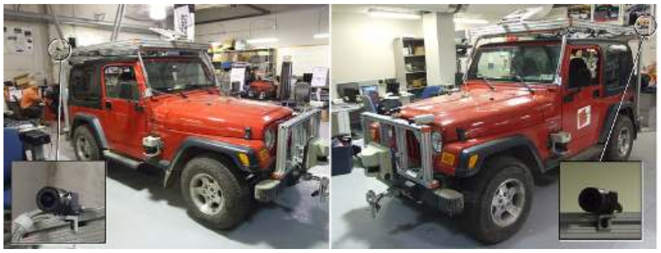


Fig. 4. Evaluation vehicle.

D. Initialization

The discrete Bayes filter also requires the initial probability density function $p_0 = \{p_{k,0}\}$ of the vehicle position. If the initial position is unknown, a uniform density function can be used. If there is belief that the vehicle is located at specific locations x_k along the route, the initial probability for those locations $p_{k,0}$ should be set higher than the for the rest of the state space.

E. MAP Estimation

The estimated location of the vehicle at every time step will be that with the largest probability within the set of possible locations, i.e., the maximum a posteriori (MAP) estimate. From the update equation of the Bayes filter algorithm, the MAP estimate is

$$X_t = \arg\max_k (p_{k,t}). \tag{6}$$

V. EXPERIMENTAL RESULTS

We conducted our experiments using a test vehicle with mounted video cameras operating on a 8.8 km test route under varying environmental conditions.

A. Vehicle

Fig. 4 shows our evaluation vehicle sensor suite. Two cameras were mounted on the roof of the vehicle, oriented approximately 45 degrees to the left and right of directly forward and configured to acquire 256×192 pixel images. The vehicle is also equipped with a GPS sensor for map generation and acquisition of ground truth data. Note that GPS measurements are not used in our localization algorithm. Vehicle velocity information is obtained from inertial sensors.

B. Evaluation Route

For the purpose of analysis and evaluation, we selected a complex, 8.8 km route that contains a variety of environments, ranging from urban to residential to parklike settings (Fig. 2a). The route includes man-made and natural structures: buildings, traffic signs, foliage, open spaces, and multiple slopes, as well as moving objects such as vehicles, pedestrians, cyclists, and motorcyclists. The trajectory makes a loop, meaning that the vehicle must face all orientations during the trajectory. This is important in order to test the robustness of the method to illumination artifacts, such as specularities made by direct sunlight exposure.

C. Storage and Processing Requirements

A full traverse requires storing and processing between 10 and 14 thousand images. The generation of a SURF database from such a large amount of data makes it intractable to access and match features in real time. Therefore, for the generation of the database, we process and store SURF features only for images that were obtained with at least 2 m distance from each other. Observe that this is independent of the value of the sampling factor ρ. This optimization reduces the database to a practical size of approximately 1 GB.

At runtime, and in order to further reduce the computational requirements, we don't evaluate every SURF feature in the database but only those in a close neighborhood of the current best position estimate of the vehicle. Such a procedure is known as *tracking*, as opposed to *global localization*. Tracking the current position of the vehicle has the advantage that the estimation can be made robust, since the probability of false positives decreases with the reduction of the search space. On the other hand, the reduced search space to consider is centered on our best position estimate. If the position of the vehicle is not correctly estimated at any point, the search space may not contain the real location of the vehicle, and the estimation will not only fail, but may diverge from the true solution. With all the above simplifications, we are able to process the left and right video data at 2 frames per second using a standard laptop.

D. Data Sets

The database used for the experimental results was obtained by processing a sequence obtained on October 19, 2010 at 11:14 AM, which was a sunny day. The data set contains significant shadowing and artifacts from the sun appearing in the images. A maximum of 200 SURF features per image were extracted. The SURF feature descriptors $\boldsymbol{d}_i$, together with their location l_i within the map and the velocity of the vehicle s_i while acquiring the image, are stored in a database $\mathcal{D}$. The created map is shown in Fig. 2. The evaluation data set was acquired nine days later on a cloudy day at approximately the same time (11:57 AM). Due to the cloudiness, the light in this sequence was more uniformly distributed in the environment and was significantly different from the first data set (Figs 6 and 7).

E. Localization Results

In this section, we demonstrate the efficiency of our proposed Bayesian approach. The initial probability density function $p_0 = \{p_k, 0\}$ is initialized with a point mass distribution on the ground truth position of the vehicle at the start of the route. The location of the vehicle at each time step is obtained from the MAP of the estimated probability density function. The parameters used for the discrete Bayes filter are $\sigma_s = 1$ m/s, $z_{match} = 0.01$, $z_{rand} = 0.99$, and $\rho = 0.5$ m. For the tracking, we consider only those measurements at a maximum distance of ± 40 nodes (i.e., ± 20 m) from the current estimated position of the vehicle.

The localization results are shown in Fig. 5. The plots corresponds to the localization results using the left and right cameras. Each camera was considered independently, with each one having its own Bayes filter. The high correlation between both curves is an indication of the consistency of the localization algorithm.

The average localization error was 2.68 m for the left camera and 2.69 m for the right camera with a standard deviation of 1.36 m in both cases. This is a excellent result considering that: 1) visual information was only stored with a minimum distance of 2 m between images; 2) the best possible location estimate is constrained by the value of ρ, which is 0.5 m; and 3) the ground truth was obtained with GPS, which actually provides fairly noisy vehicle position measurements.

Some error peaks in the plot of Fig. 5 are caused by inaccuracies of the ground truth data. The image results for the first peak marked in the plot is shown in Fig. 6a. As it can be seen from the figure, there is a small error in the localization of the vehicle, but the error is far from being 9 m, as the plot indicates. The second peak in the plot is also incorrect. The visual result of that peak is shown in Fig. 6b, from where it can be seen that the localization is accurate. On the other hand, the fourth peak corresponding to the plot of the left camera does actually correspond to a wrong estimation of the vehicle position, as it can be seen in Fig. 6d.

The third peak in the curve was produced because the vehicle traveled different paths. A short street of our predefined route was closed because of a construction site in the south-west part of the route (see Figs 2d and 6c). We have increased the tracking area in this part of the sequence to 100 m in order to allow the Bayesian method to find the actual location once the vehicle rejoined its standard route. Observe that this is a typical example of what is called *kidnapped robot problem*.

The largest error in the estimation occurred for the left camera around image numbers 9600 and 11100. Extreme occlusion cases were the cause of these errors (Fig. 7d). In both cases, the divergence was compensated by the Bayes filter of the right camera.

VI. SUMMARY AND FUTURE WORK

In this paper, we have developed a method to localize an autonomous vehicle using only visual inputs. The method represents visual features using a topometric map and then localizes against these features using a discrete Bayes filter. Our experiments show that the method is capable of reliably localizing a vehicle over long routes in real time. The method succeeds despite challenging differences between the reference database and the runtime conditions, including significant differences in the illumination, occlusions, ground cover due to seasonal change, and sky appearance (Fig. 7).

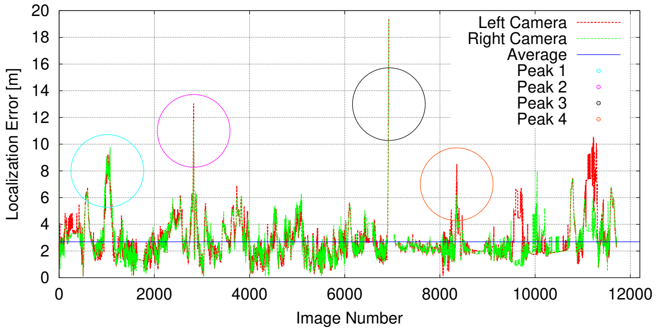
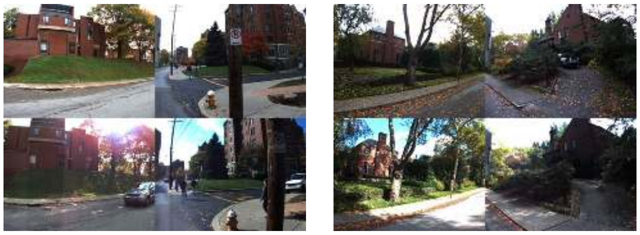
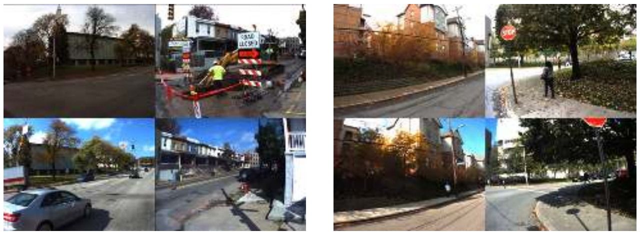


Fig. 5. Localization results for data set acquired on October 28, 2010.



(a) *Peak Number 1* (b) *Peak Number 2*



(c) *Base of Peak Number 3* (d) *Peak Number 4*

Fig. 6. Localization results for the peaks marked in Fig. 5. The top images (in each subfigure) show the views acquired by left and right camera of the current position. The bottom images show the views obtained from the database for the estimated vehicle location.

Although these initial results are promising, we are continuing the development along several dimensions. First, we are investigating the effects of long-term environmental changes on the ability to localize. Second, we are considering short term effects that can disrupt localization, such as temporary occlusions from moving vehicles. Finally, we are analyzing to expand the topology of the route from the single ring structure evaluated here to the more general case of directed graph with cycles. This involves the development of new methods for the automated generation of topometric maps.

VII. ACKNOWLEDGMENTS

This work was supported by the Agency for Defense Development, Jochiwongil 462, Yuseong, Daijeon, Korea.

REFERENCES

[1] S. Thrun, "http://googleblog.blogspot.com/2010/10/what-were-driving-at.html," 2010.

[2] S. Thrun et al., "Stanley: The robot that won the DARPA Grand Challenge," *J. Robotics Systems*, vol. 23, pp. 661–692, 2006.

[3] C. Urmson et al., "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*, vol. 25, no. 1, 2008.

[4] http://maps.google.com/help/maps/streetview/

[5] G. Silveira, E. Malis, and P. Rives, "An efficient direct approach to visual SLAM," *IEEE Transactions on Robotics*, 2008.

[6] J. Blanco, J. Fernández-Madrigal, and J. González, "Toward a unified Bayesian approach to hybrid metric-topological SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 259–270, 2008.

[7] C. Valgren and A. J. Lilienthal, "SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments," *Submitted to Elsevier*, 2010.

[8] A. Ascani, E. Frontoni, and A. Mancini, "Feature group matching for appearance-based localization," in *IROS*, 2008.

[9] H. Andreasson, A. Treptow, and T. Duckett, "Localization for mobile robots using panoramic vision, local features and particle filter," in *ICRA*, 2005.

[10] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Transactions on Robotics*, 2005.

[11] A. C. Murillo, J. J. Guerrero, and C. Sagüés, "SURF features for efficient robot localization with omnidirectional images," in *ICRA*, 2007.

[12] N. Tomatis, I. Nourbakhsh, and R. Siegwart, "Hybrid simultaneous localization and map building: a natural integration of topological and metric," *Robotics and Autonomous Systems*, vol. 3, no. 14, 2003.

[13] H. Kouzoubov and D. Austin, "Hybrid topological/metric approach to SLAM," in *ICRA*, vol. 1, 2004, pp. 872–877.

[14] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An Atlas framework for scalable mapping," in *ICRA*, vol. 2, 2003.

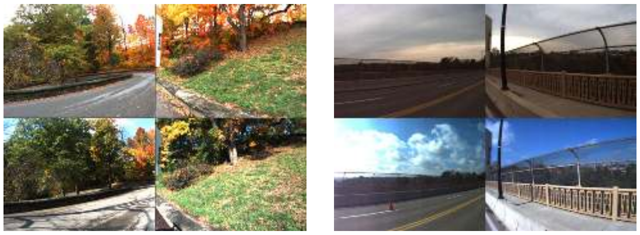
[15] A. Cumani and A. Guiducci, "Comparison of feature detectors for rover navigation," *Mathematical Modelling and Applied Computing*, vol. 1, no. 1, 2010.

[16] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *ECCV*, 2006.

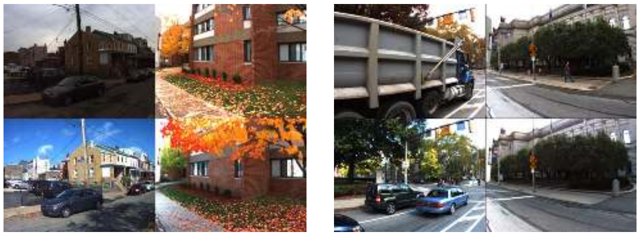
[17] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *eccv*, vol. 5305, 2008, pp. 102–115.

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Robotics Research*, vol. 2, 2004.

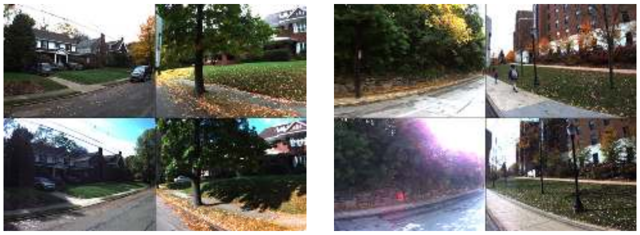
[19] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.



(a) *Seasonal change* (b) *Difference in the sky*



(c) *Illumination change* (d) *Occlusions*



(e) *Shadows* (f) *Sunlight exposure*

Fig. 7. Localization results on challenging situations. The top images (in each subfigure) show the views acquired by left and right camera of the current position. The bottom images show the views obtained from the database for the estimated vehicle location.