Visual Localization in Fused Image and Laser Range Data

Nicholas Carlevaris-Bianco, Anush Mohan
Dept. Electrical Eng. & Computer Science
University of Michigan
Ann Arbor, Michigan 48109
Email: {carlevar,anushm}@umich.edu

James R. McBride
Research and Innovation Center
Ford Motor Company
Dearborn, Michigan 48124
Email: jmcbride@ford.com

Ryan M. Eustice
Dept. Naval Architecture & Marine Eng.
University of Michigan
Ann Arbor, Michigan 48109
Email: eustice@umich.edu

***Abstract*— This paper reports on a method for tracking a camera system within an *a priori* known map constructed from co-registered 3D light detection and ranging (LIDAR) and omnidirectional image data. Our method pre-processes the raw 3D LIDAR and camera data to produce a sparse map that can scale to city-size environments. From the original LIDAR and camera data we extract visual features and identify those that are most robust to varying viewpoint. This allows us to include only the visual features that are most useful for localization in the map. Additionally, we quantize the visual features using a vocabulary tree to further reduce the map's file size. We then use vision-based localization to track the vehicle's motion through the map. We present results on urban data collected with Ford Motor Company's autonomous vehicle testbed. In our experiments the map is built using urban data from winter 2009, and localization is performed using data collected in fall 2010 and winter 2011. This demonstrates our algorithm's robustness to temporal changes in the environment.**

I. Introduction

Vehicles capable of autonomous navigation in urban environments typically rely on expensive perception and navigation sensors and significant amounts of computing power. Consider, for example, the vehicles used in the DARPA Urban Challenge [1]. Most of these vehicles used expensive inertial navigation sensors, 3D light detection and ranging (LIDAR) scanners capable of measuring a million ranges per second and multi-view camera systems. The rich data provided by these sensors allowed the vehicles to operate autonomously in their environment. Unfortunately, such a sensor suite costs hundreds of thousands of US dollars. Clearly, this prohibits widespread use of this technology.

In this paper we demonstrate that it is possible to collect the vision and LIDAR data once for an area using a fully instrumented vehicle, and afterward localize to the data using a low cost camera system. This allows us to extend some of the capabilities of advanced autonomous vehicles to much lower cost instrumented camera systems.

Our method is divided into two parts: *(i)* a pre-processing stage in which the map is constructed from the raw data provided by the full sensor suite and *(ii)* an online localization stage where the instrumented camera system localizes itself within the *a priori* map (Fig. 1).

At a minimum, the low cost system would consist of a single monocular camera. However, the addition of an orientation sensor and global positioning system (GPS) can provide a bounded estimate of the camera's orientation and position, which can be used to bootstrap the visual tracking and to reinitialize the system after regions where tracking fails. (Though, as we will discuss in Section VII, recent progress in computer vision may allow for (re)initialization using purely visual methods.) Instrumented camera systems, which combine a camera, orientation sensor, and GPS are becoming ubiquitous. For example, such integrated systems are typically found in modern smart phones.

With this work we seek to address several challenges. First, because the map is only collected once and then later used for localization, the algorithm must be robust to

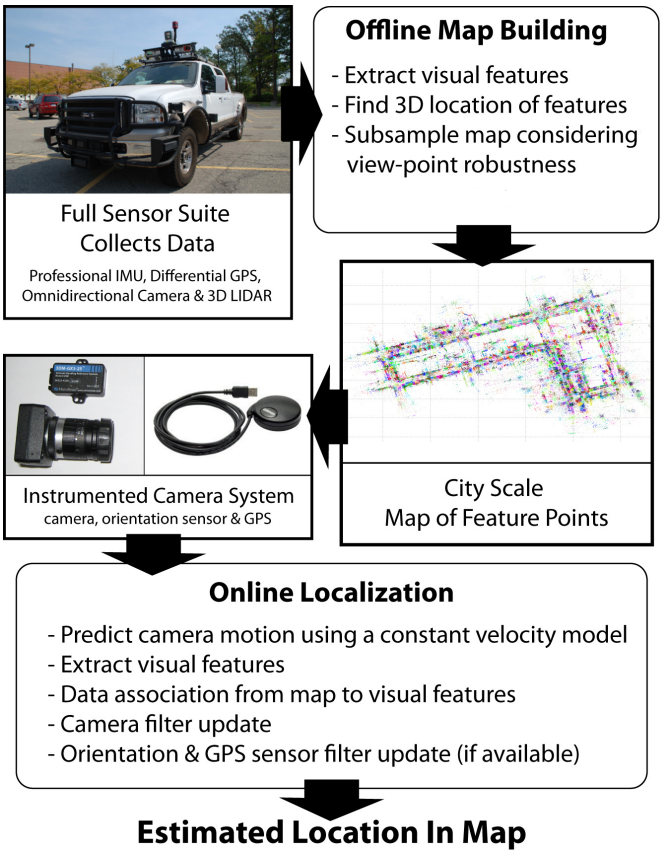


Fig. 1. Algorithm overview. Data collected using a fully instrumented vehicle is used to build a fused modality map of the environment. Then a lesser instrumented camera system, for example a camera with an attitude sensor and GPS, localizes itself within the map (possibly years later).

dynamic changes in the environment. Second, because the camera trajectory may not be the same between mapping and localization, we need to be robust to view point change. To account for this, we specifically build the map to include the most viewpoint-robust visual features. Finally, we seek to produce a map that is scalable to very large areas. This requires that our map representation be sufficiently compact so that the map can be easily stored.

In our experiments, we consider a system with four monocular cameras, GPS, and an attitude sensor, which we use to localize an automobile traveling through an urban environment. We present results in which the map is built using data from fall 2009 while localizing to the map with data collected in fall 2010 and winter 2011. This demonstrates the ability of our system to provide a location estimate that is significantly more accurate than that of a consumer-grade GPS while dealing with large changes in a dynamic environment.

In Section II we discuss how our algorithm relates to existing work in the fields of vision-based simultaneous localization and mapping (SLAM), place recognition, and visual localization. In Section III we provide an overview of our system. Sections IV and V explain in detail how the map is generated and our proposed method for localization. Experimental results are presented in Section VI. Finally, we discuss the results and future work in Section VII.

II. Related Work

We attempt to solve the problem of camera localization within an *a priori* known map. Previous work in visual localization can be roughly split into two categories: tracking methods and localization-by-recognition methods.

Tracking methods, which are common in the SLAM and robotics communities, use a strong pose prior over a small baseline to perform visual localization of sequential images. Notable examples that address the full SLAM problem include the works by Davison et al. [2], Eade and Drummond [3] and Klein and Murray [4]. Davison uses an extended Kalman filter (EKF) based solution while Eade extends FAST-SLAM [5]. Klein performs mapping based on keyframe bundle adjustment while separately tracking the camera motion by localizing to the current map using a position prior from a constant velocity motion model.

The localization portion of these methods is similar to our proposed tracking method. When a strong pose prior is available, we actively search for known map features in the current image based on their expected location in the image. Similar to [4], we then find the estimated camera pose by minimizing the re-projection error between the image and map features.

However, the scale of the maps that these techniques deal with is fairly small. While they focus on localizing a camera within environments such as closed rooms, we focus on localizing a camera in an outdoor city-scale environment. Furthermore, these methods rely on high-frame-rate short-baseline imagery in order to use patch-based features to characterize points in the image. This is sufficient when the localization and map building are performed simultaneously by the same camera; however, in our application the map building data and the localization data will be collected at different times with a different trajectory and possibly a different camera. We expect that patch based methods will not provide a sufficient level of robustness to the variance cause by the dynamic environment and changes in view point. Therefore, we have chosen to use robust descriptors such as scale invariant feature transform (SIFT) [6], [7] instead of image patches.

Localization-by-recognition methods, [8], [9], [10], seek to localize the camera by recognizing the current view with respect to a set of known locations. This is often performed with little or no prior information on the pose of the camera. These methods have been addressed in the past by the computer vision community. It is common practice to model the locations as a collection of visual vocabulary (quantized visual feature descriptors). This "bag of words" model [11] is then used to determine which locations have a similar distribution of features compared to the current image. Hypothesis locations are geometrically verified, which also serves to localize the camera with respect to that location.

Recently, two closely related works, [10], [12], attempt to localize a camera in a map created by structure-from-motion. These methods are similar to our approach as they seek to localize the camera with respect to an *a priori* known map of 3D feature points, yet differ from our proposed method in map representations and localization methods.

Irschara et al. [10] propose a method that represents the map as a set of spatially sampled 'synthetic views' created by projecting the 3D feature points into evenly spaced synthetic cameras. They then use image-to-image recognition techniques to compute the camera's pose with respect to a synthetic view—thus localizing the camera. This differs from our proposed method as we perform direct matching between the image and the map's 3D feature points. Additionally, unlike our proposed method, Irschara et al. do not make use of a pose prior and instead search over all synthetic views at each new image. This makes their algorithm capable of localizing a stream of images that are not necessarily sequential. However, for the specific task of localizing a moving camera we consider it important to include prior pose information.

Arth et al. [12] also seek to localize images in a map created using structure-from-motion. Their method divides the map into 'potentially visible sets'; discrete sets of feature points which, based on occlusion, are all visible from a given location. In indoor environments this often means that a room will be considered a potentially visible set. They then require information from a GPS sensor, radio frequency (RF) beacon, or user input to determine in which potentially visible set the camera currently is. The feature points within the given potentially visible set are then matched against the current image to localize the camera. In this sense, the potentially visible sets are very similar to our proposed spatial clusters in that they quickly allow one to reduce the quantity of visual features considered for matching, based on a pose prior.

III. System Overview

The first step of our proposed method, Alg. 1, is a pre-processing stage in which the map is constructed from the raw data provided by the full sensor suite. During map creation we extract visual features from the environment, currently SIFT [6], and then track these features as the fully instrumented robot moves through the environment. We cluster features spatially, over their X, Y, Z position and ψ, the azimuth angle from which the feature was observed. We then consider the number of frames a feature has been successfully tracked over to select only the most "trackable" features of each cluster to include in the map. This allows us to control map size and ensure good spatial coverage of the environment while maintaining the utility for visual localization. Additionally, we hierarchically cluster the visual features extracted from the map images to produce a vocabulary tree that quantitizes the visual features in the map. This drastically reduces the memory required to store map features and provides a rapid method for determining putative feature correspondences. Map generation is discussed in detail in Section IV.

Given the map, we then seek to localize the instrumented camera as it moves through the environment using a (linear) Kalman filter (KF) (our plant and observation models are linear as formulated). We use measurements from GPS and orientation sensors to initialize the KF and supplement localization in feature-poor regions of the environment. However, for accurate localization we rely on the camera to provide motion constraints. To do so we must first associate the SIFT features extracted from a given image with the features in the map. We use the centroid of the spatial clusters to quickly determine which clusters could contain candidate features based on the current estimate of the camera pose. We then project these candidate map features into the image and search for visual correspondence. We use the uncertainty in the camera pose to geometrically constrain the location of possible image-to-map correspondences. Then given a set of putative correspondences we use random sample consensus (RANSAC) [13] to identify geometrically consistent inliers. Finally, we use non-linear optimization to determine a pose constraint that is used to provide a linear update on state to the filter. Localization is discussed in detail in Section V.

IV. Map Generation

The final map consists of M spatial clusters of feature points. Each cluster has a mean position, $\bar{\mathbf{X}} = [\bar{X}, \bar{Y}, \bar{Z}]^\top$, and a mean view azimuth, $\bar{\psi}$. The view azimuth is the azimuth angle of the vector that points from the center of the cluster to the center of the camera from which the features in the cluster were observed. Within each cluster, C_i for $i = 1 \dots M$, there are N_i feature points. Each feature point, F_j^i for $j = 1 \dots N_i$, has an associated 3D location, $\mathbf{X}_j = [X, Y, Z]^\top$, view azimuth, ψ_j, and feature vocabulary id, v_j. The feature vocabulary id, v_j, is the id of the leaf node in the vocabulary tree to which the SIFT descriptor for the feature corresponds. The steps used to produce the map are as follows.

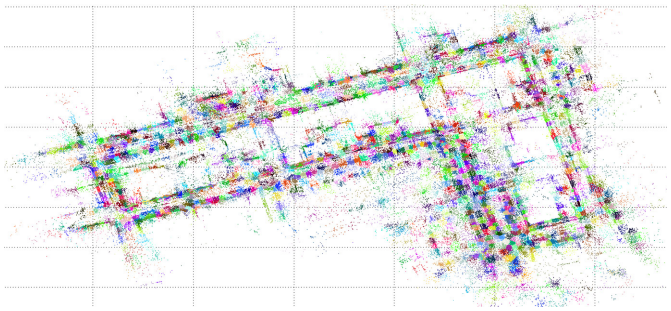
Algorithm 1 System Overview

Pre-processing

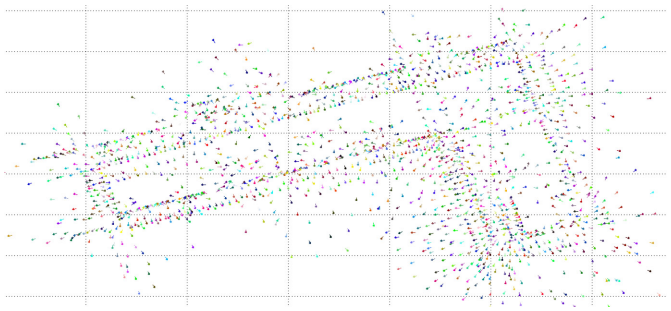
1: Extract visual SIFT features
2: Track features through environment
3: Quantize features with vocabulary tree
4: Cluster features spatially
5: Sub-sample map based on 'trackability'

Online Localization

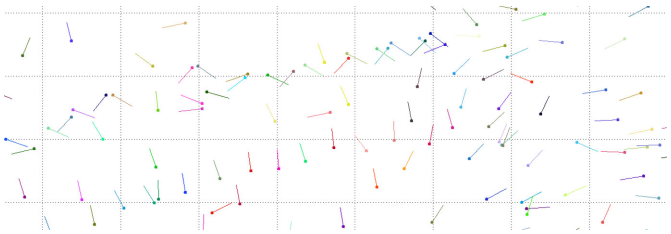
1: Predict motion
2: Extract and quantize visual SIFT features
3: Cull clusters based on centroid
4: Identify putative image-to-map correspondences
5: Determine geometrically consistent inliers (RANSAC)
6: Nonlinear optimization to produce pose estimate
7: Update filter



(a) Map features.



(b) Map clusters.



(c) Map clusters zoomed.

Fig. 2. A sample map built from a 1.36 km trajectory in an urban environment is shown in (a). Each point is a different feature, the colors are set by the cluster to which a feature belongs. A reduced map with only cluster centroids and mean view azimuth vectors is shown in (b) with a zoomed view in (c).

A. Feature Tracking

As the fully instrumented robot moves through the environment it extracts SIFT features from each image it collects. The features from sequential images are compared to produce a set of putative correspondences. From these putative matches a set of inlier correspondences is determined by fitting a fundamental matrix using RANSAC. The inliers that have been tracked between two images are considered as possible map features.

As we continue processing features from sequential images we count the number of frames through which each feature was tracked. A higher count indicates that the feature was robust to view point change and, therefore, a good choice for inclusion in the map. By normalizing this count by the maximum in the whole map, we produce a view robustness score that varies between 0 and 1. In order to produce a single descriptor for the feature we average the descriptors from each image in which the feature was detected and then find the vocab tree leaf that corresponds to this averaged feature. This method, of tracking features over multiple frames and averaging to determine the most robust features, was proposed by Kawewong et al. in [14] and has been shown to produce good results for place recognition in dynamic environments.

Additionally, we must determine the location of the feature points in space with respect to a local coordinate frame. We assume the pose of the robot to be known with very low uncertainty during the map construction phase. In our experiments this comes from the fact that the robot is instrumented with a differential GPS and an extremely accurate inertial measurement unit (IMU). If we can project a range measurement from the laser scanner onto the feature in any of the images where it was observed, we use the laser scanner measurement as the location of the feature. However, if no 3D information from the laser scanner is available, the locations of the points can be triangulated from the camera views using the accurate vehicle pose. Because triangulation is inherently more noisy than the laser scanner, we rely on two heuristics to reduce noisy triangulations. First, as the fully instrumented robot has five cameras during the data collection phase, we only attempt to triangulate points in the cameras that are not aligned with the direction of the motion of the vehicle. Second, we require that a feature be seen in three or more sequential frames so that we can compute a least squares solution for triangulation.

B. Spatial Clustering

Given all the extracted features, we then seek to spatially cluster the features. Clustering allows us to preserve good spatial coverage when sub-sampling the map. The spatial clustering also helps during localization by allowing us to quickly reject a large number of clusters based on their centroid location and mean view angle. We aim to cluster the features based on their location, $[X, Y, Z]^\top$, and view angle, ψ. However, clustering over the view angle, ψ, is problematic as it is a circular quantity. Therefore, in practice, clustering is performed using K-Means over a 5-dimensional space, $[X, Y, Z, X_c, Y_c]^\top$, where, X_c and Y_c are the position of the camera that observed the feature. We then can calculate ψ after clustering based on X, Y, X_c and Y_c.

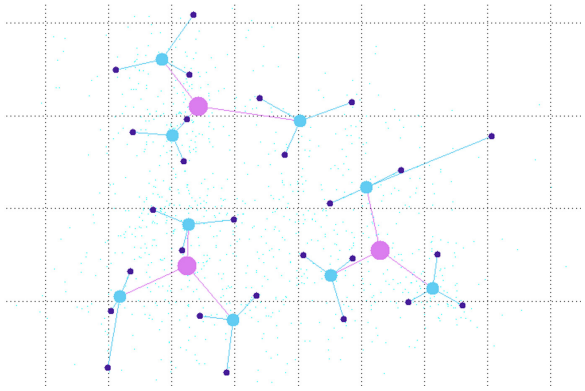


Fig. 3. Sample vocabulary tree, with $k = 3$ and $L = 3$ (only 2 of 128 dimensions shown).

The number of clusters is a function of map size and environment. We found that in an urban environment it was acceptable to automatically adjust the number of clusters so that the average number of features per cluster was $\backsim$ 200, where 200 is approximately the average number of features tracked between images during map building. Fig. 2(a) shows an example map collected over a 1.36 km trajectory in an urban environment. Each point is a different feature, the colors are set by the cluster to which a feature belongs. Fig. 2(b) contains a reduced map with only cluster centroids and mean view azimuth vectors shown (to reduce clutter). One can see that nearby features will be separated into distinct clusters if the features were originally viewed from different locations.

C. Vocabulary Tree Generation

A vocabulary tree [15] provides a method to quantize the map's visual features in order to reduce the amount of data that must be stored for each feature point. Additionally, the tree provides a fast method to determine potential feature correspondences because a new feature can be quantized using a small number of comparisons.

The vocabulary tree is produced by hierarchically clustering the visual features in the 128-dimensional SIFT feature space. Clustering is performed at each level using K-Means to produce k clusters, where k is referred to as the branch factor. Repeating this process for a set number of levels, L, produces k^L leaf nodes. Traversing the tree in order to quantize a feature descriptor requires only kL comparisons.

Fig. 3 illustrates two dimensions of a vocabulary tree with $k = 3$ and $L = 3$. In practice, vocabulary trees may have hundreds of thousands to millions of leaf nodes depending on the application and number of features used in training. The results in this paper were produced with approximately 5 million training features yielding a relatively coarse vocabulary tree with $k = 8$ and $L = 5$ with $8^5 = 32,768$ leaf nodes.

TABLE I

NUMBER OF FEATURES AND MAP FILE SIZE FOR SAMPLE TRAJECTORY

	Number of Features	File Size
Raw Data	—	110 GB
Map after Feature Extraction	5,809,691	4.7 GB
Map after Tracking and Vocab ID	366,687	18 MB
Map after Sub-sampling	174,441	12 MB

D. Map Sub-sampling

Even for short trajectories the size of the raw image and LIDAR data collected by the fully instrumented robot will be on the order of hundreds of gigabytes. By representing the map as a sparse collection of 3D points described by their associated visual vocabulary we can discard the majority of the image and LIDAR data producing maps on the order of hundreds of megabytes in size. However, depending on the memory limitations of the instrumented camera system, and on the extent of the map, one may wish to further reduce the file size of the map. By considering the view robustness score (the normalized number of frames the feature was tracked over), we can rank the features within a cluster. We can then sub-sample the map by selecting the "best" features from each cluster until a predetermined memory limit. Note that this is performed on a per-cluster basis. Given a memory limit and the number of clusters in the map we calculate a maximum number of features per clusters. Clusters with many features are sub-sampled down to this limit while clusters with too few features are not sub-sampled. This allows us to maintain the good spatial coverage of the map during sub-sampling.

Table I shows file sizes for the map used in our experiments. The raw data collected by the map building robot is over 100 GB for the 1.36 km urban trajectory. By extracting visual features and discarding the raw images and laser scans we reduce the data size to 4.7 GB. By tracking features over multiple frames and describing them using a vocabulary tree the map can be reduced to 18 MB—with further size reduction possible through map sub-sampling.

V. LOCALIZATION FILTER

The localization filter is used to track the state of the instrumented camera system through the map. Our state vector, $\mathbf{x} = [\mathbf{r}, \boldsymbol{\theta}, \mathbf{v}, \boldsymbol{\omega}]^\top$, contains the 3D position $\mathbf{r} = [x, y, z]^\top$, Euler orientation $\boldsymbol{\theta} = [r, p, h]^\top$, linear velocity $\mathbf{v} = [\dot{x}, \dot{y}, \dot{z}]^\top$ and Euler rates $\boldsymbol{\omega} = [\dot{r}, \dot{p}, \dot{h}]^\top$, all with respect to a fixed world coordinate frame.

Similar to Davison et al. [2], we use a constant velocity, constant angular velocity motion model. This model assumes that the camera is driven by unknown Gaussian distributed accelerations with a constant velocity over a single time step. This unknown acceleration accounts for the unknown dynamics of the system as well as the unknown control input to the instrumented camera system.

A. Camera Constraints

The camera observation model is the conventional projective camera model that takes a homogeneous point $\mathbf{X} = [X, Y, Z, 1]^\top$ in 3D and projects it onto the image plane at a location $\mathbf{p} = [u, v, w]^\top$ as shown in (1). The projection matrix $\mathbf{P} = \mathrm{K}[\mathrm{R} \mid \mathbf{t}]$ is composed of the intrinsic camera matrix K, and the extrinsic camera parameters R and $\mathbf{t}$, that capture the camera's rotation and translation.

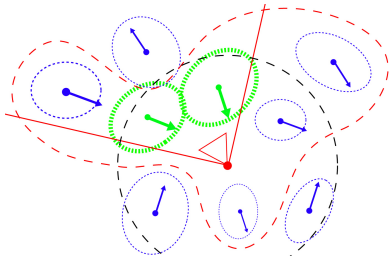


Fig. 4. Estimating potential clusters based on centroid and mean view azimuth. First, we eliminate clusters outside of the perceptual radius (dotted black line). Second, we remove clusters with a centroid that does not lie in a 90° cone in front of the camera (solid red lines). Finally, clusters with a mean view azimuth that does not align with the camera's view azimuth within $\pm 45^\circ$ are removed (dashed red lines). This leaves only the pose-viable candidate clusters (bold green) to consider when performing data association.

$$\mathbf{p} = \mathbf{P}\mathbf{X} \tag{1}$$

1) Visual Data Association: In order to correct our predicted state using these camera measurements, we need to perform data association to establish a correspondence between SIFT features extracted from the image and features in the map. A large map may have hundreds of thousands of point features. In order to avoid unnecessary computational costs, we first wish to quickly eliminate highly unlikely feature points. To do so we consider only the clusters' centroids and mean view azimuths. First, we exclude clusters whose centroid is beyond the "perceptual radius" of the current pose estimate. For our experiments the perceptual radius was set to 100 m—the maximum range of the laser scanner used to build the map. Second, we eliminate clusters with a centroid that does not lie in a 90° cone in front of the camera. Finally, we remove clusters with a mean view azimuth that does not align with the camera's view azimuth within $\pm 45^\circ$. This process is illustrated in Fig. 4. The centroids of the remaining, much reduced, subset of clusters are projected into the camera frame. If they fall within a bound around the image then we consider the cluster to be a candidate for matching.

We then project all of the feature points within the candidate clusters onto the image plane and search for matching image features within a neighborhood determined by the first-order covariance of the camera pose:

$$\Sigma_{zz} = J^{\mathbf{P}}_{\mathbf{x}} \Sigma_{\mathbf{xx}} (J^{\mathbf{P}}_{\mathbf{x}})^\top + J^{\mathbf{P}}_{\mathbf{X}} \Sigma_{\mathbf{XX}} (J^{\mathbf{P}}_{\mathbf{X}})^\top, \tag{2}$$

where $\mathbf{z} = [u, v]^\top$ are the pixel coordinates of the feature and $J^{\mathbf{P}}_{\mathbf{x}}$ and $J^{\mathbf{P}}_{\mathbf{X}}$ are the Jacobians of the camera projection function, (1), with respect to the camera pose, $\mathbf{x}$, and the feature's 3D coordinates, $\mathbf{X}$, respectively. Fig. 5 depicts the data association procedure, with projected map points shown as stars and image features shown as dots. The ellipse around

Fig. 5. Geometrically constrained correspondence search. Projected map points are shown as stars and image features are shown as dots. The ellipse around each map feature represents a 99% confidence bound on where the feature should lie in the image.

each map feature represents the 99% confidence bound on where the feature should lie based on the uncertainty associated with projecting that 3D map point into the image. A match between a map feature and an image feature is established when an image feature from the same vocabulary tree leaf as the map feature is found within the search ellipse. Finally, we use RANSAC to select only those associations that are geometrically consistent.

2) Camera Observation Update: As a result of the data association step, we obtain a correspondence between the image features, $\mathbf{z} = [u, v]^\top$, and the map features $\mathbf{X} = [X, Y, Z]^\top$. The projected pixel locations of the map features $\hat{\mathbf{z}} = [\hat{u}, \hat{v}]^\top$ are obtained from the model described in (1). We then try to find the state vector, $\hat{\mathbf{x}} = [\hat{x}, \hat{y}, \hat{z}, \hat{r}, \hat{p}, \hat{h}]^\top$, that minimizes the re-projection error between image features and map points, as shown in (3),

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \phi(\begin{bmatrix} u \\ v \end{bmatrix}, \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}). \tag{3}$$

In our experiments the cost function ϕ is simply squared error; however, a Huber [16] cost function could be used to reduce the influence of outliers. We solve for the optimal pose using the Levenberg-Marquardt optimization algorithm. The first-order approximation of the covariance of the estimated state is given by

$$\Sigma_{\hat{\mathbf{x}}\hat{\mathbf{x}}} = ({J_{\mathbf{x}}^{\phi}}^\top \Sigma_{\mathbf{pp}}^{-1} J_{\mathbf{x}}^{\phi})^{-1} \tag{4}$$

where $J_{\mathbf{x}}^{\phi}$ is the Jacobian of the cost function with respect to state, and $\Sigma_{\mathbf{pp}}$ is the covariance of the pixel coordinates of the extracted features, commonly assumed to be isotropic with unit variance.

VI. Experimental Results

In order to evaluate our algorithm in a real-world scenario we present results using the Ford Campus Vision and LIDAR Data Set [17]. This data set was collected with Ford's autonomous ground vehicle testbed (Fig. 1) outfitted with an omni-directional camera, professional IMU and differential GPS. To build the map we use the full sensor suite with data collected in the winter 2009. To test localization we then use data collected with the same vehicle driving a similar trajectory in fall 2010 and winter 2011. For localization we use four of the cameras from the omnidirectional system, excluding the forward looking camera. Additionally, we consider attitude measurements from a consumer-grade orientation sensor and position from a consumer-grade GPS.

A. Localization Results

The trajectories produced by our proposed algorithm are shown in Fig. 6(a) and 6(b). At any given point in the trajectory the size and color of the marker are proportional to the spatial uncertainty in the state estimate. Spatial uncertainty is defined in terms of the determinate of the x and y position covariance as

$$\sigma = (\det \Sigma_{xy})^{1/4}, \tag{5}$$

which has units of length. Sample imagery corresponding to the numbered locations in the trajectories is shown in Fig. 6(c) through 6(h). These images illustrate some of the challenges presented by the data set including; low saliency regions, 6(c), changing structure and appearance, 6(d) and 6(f), and poor lighting and exposure, 6(g) and 6(h). Even under the best conditions, 6(e), visual matching must still contend with lighting changes.

We see that our proposed method allows for low-uncertainty localization for both the fall 2010 and winter 2011 data sets. As one would expect, the snow present in the winter 2011 trajectory proves slightly more difficult, especially in regions with less visually interesting features, such as the leg around region 1.

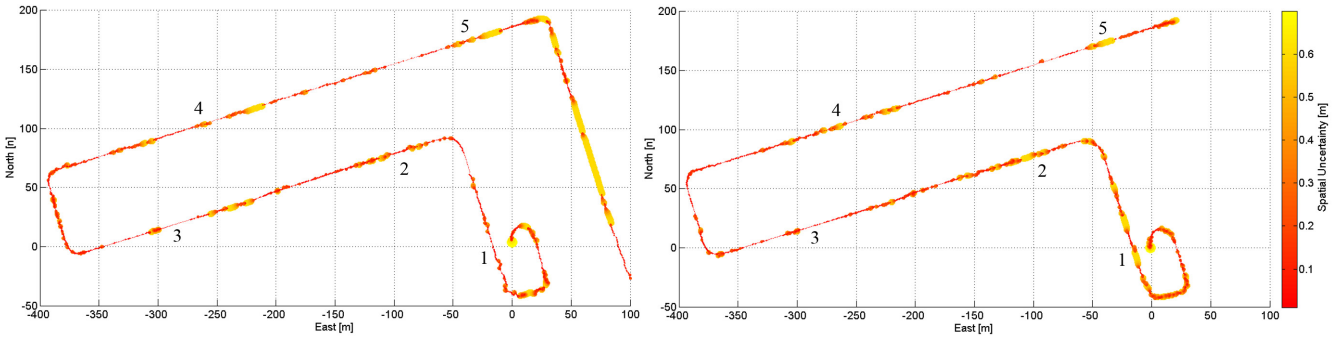
B. Map Sub-sampling Results

Additionally, we consider the effect of map sub-sampling on the localization utility of a map. Fig. 7 compares the moving average of spatial uncertainty for varying map sizes using the proposed sub-sampling method outlined in Sec. IV-D. As one would expect, reducing the number of features in the map results in a higher average uncertainty. However, even after removing approximately half of the map's features, visual tracking still provides a substantial reduction in spatial uncertainty.

During development we also considered sub-sampling based on the visual uniqueness of features within a cluster. In order to identify features that may be visually aliased in a local area, we computed a simple local saliency score, s, for each feature with respect to its spatial cluster. Using the vocabulary id for the visual features we consider, for each feature in a spatial cluster, the ratio between the number of features from the same vocab in that spatial cluster $n_{same\ vocab}$, and the total number of features in the spatial cluster, n_{total}. The local saliency score, s, is then defined as

$$s = 1 - \frac{n_{same\ vocab}}{n_{total}} \tag{6}$$

where s varies between 0 and 1 with visually repetitive features receiving low scores.



(a) Fall 2010 trajectory.

(b) Winter 2011 trajectory.

(c) Sample images 1.

(d) Sample images 2.

(e) Sample images 3.

(f) Sample images 4a.

(g) Sample images 4b.

(h) Sample images 5.

Fig. 6. Experimental results. (a) and (b) show the trajectory produced by our algorithm using data from 2010 and 2011. At any given point in the trajectories the size and color of the marker are proportional to the spatial uncertainty in the xy state estimate. Sample imagery from the map and both localization sets corresponding to the numbered locations in the trajectory are shown in (c)–(h). These images illustrate some of the challenges presented by the data set including; low saliency regions, (c), changing structure and appearance, (d) and (f), and poor lighting and exposure, (g) and (h). Even under the best conditions, (e), visual matching must still contend with lighting changes.

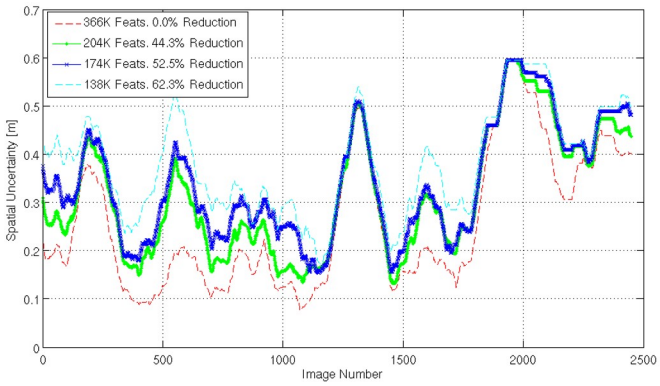


Fig. 7. Effect of map sub-sampling: a moving average of the spatial uncertainty is plotted for varying map sizes.

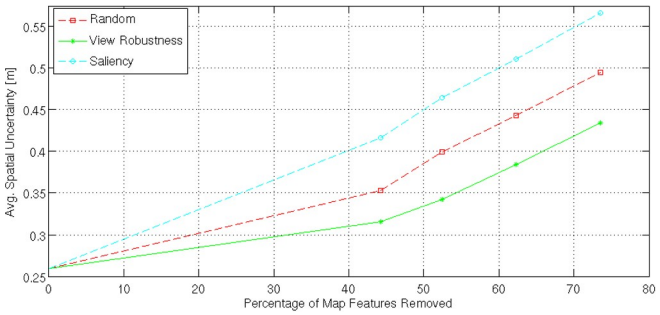


Fig. 8. Average spatial uncertainty is plotted for varying map sizes using the proposed view robustness sub-sampling method (IV-D), saliency based, and naïve random sub-sampling.

Comparing our proposed method, Sec. IV-D, local saliency and naïve random sub-sampling, Fig. 8, we see that for a given map size our proposed view-robustness method provides a substantial reduction in localization uncertainty.

Additionally, we note that local saliency sub-sampling systemically performs worse than random sub-sampling. Though unexpected, this result provided two interesting insights into the proposed algorithm. First, we note that through cluster-based feature culling (Fig. 4) and geometrically constrained correspondence search (Fig. 5) we greatly reduce the effect of visually aliased features in the environment. This alone, however, does not explain why saliency sub-sampling should perform worse than random. In fact we found that multiple instances of the same feature can be added to the map. This happens during map construction when a feature

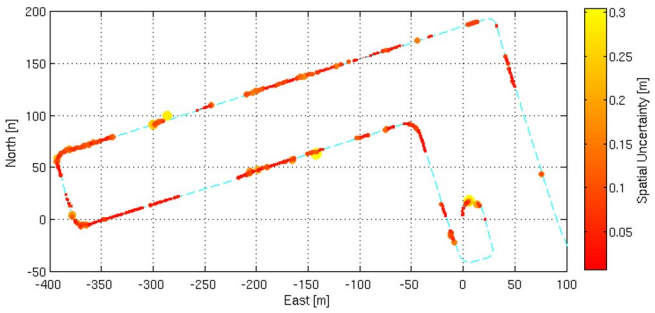


Fig. 9. Vocabulary-tree-based location recognition was used in place of GPS to bootstrap the localization filter over the fall 2010 trajectory. Again, the spatial uncertainty is proportional to size and color of the marker. The GPS measured trajectory is plotted in cyan for reference.

is tracked for several frames, is missed during one frame, then is detected again and tracked in subsequent frames. Without a way to re-establish tracking, multiple instances of the same feature are inserted into the map. These duplicate features will be considered visually aliased and removed during saliency sub-sampling even though they are trackable over a wide change in view point. It is because of this that saliency sub-sampling harms the ability to localize to the map.

VII. Discussion and Future Work

Recent works, including [4] and [18], have advocated for methods in which visual tracking is supplemented by localization-by-recognition methods. When initializing tracking or when tracking is lost, these methods provide a vision-only method to (re)initialize the visual tracking filter. We are currently working toward a visual method for providing a global estimate of the camera position to supplement or replace our current use of GPS. As a preliminary result we have implemented vocabulary-tree-based recognition as described by [15], using the map's spatial clusters as our location model. The 2010 trajectory, processed with this method, is shown in Fig. 9. When the robot encounters visually salient regions of the environment the location recognition algorithm is able to bootstrap the visual filter. However, when tracking is lost in visually uninteresting regions, tracking results are unavailable until the location recognition algorithm can reinitialize the filter.

Currently, our framework does not have a mechanism to update the *a priori* map based upon feedback from the localizing camera system. We feel that if used over long periods of time (such as in life-long learning) that it would be imperative for the map to be improved using feedback from the localizing camera—developing even more temporally stable maps through repeated use.

Finally, we are also interested in applying a similar structure to multi-robot SLAM where different robots have varying sensing capabilities.

VIII. Conclusions

This paper presented an algorithm for localizing an instrumented camera system within an *a priori* known map constructed from co-registered 3D LIDAR and omnidirectional image data. Our method intelligently sub-samples the rich 3D LIDAR and image data to produce a compact map of visual features that are both robust to varying view point and that are visually salient. We demonstrated the use of vision-based localization to track an auxiliary camera's motion through the map. and the ability of the algorithm to localize in a map built with real-world data collected over multiple years.

Acknowledgments

This work was supported in part by the National Science Foundation under NSF Award IIS-0746455 and in part by the Ford Motor Company via the Ford-UofM Alliance Award #N009933. We would also like to thank Gaurav Pandey for his help in the data collection process.

References

[1] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, 1st ed. Springer Publishing Company, Incorporated, 2009.

[2] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, pp. 1052–1067, June 2007.

[3] E. Eade and T. Drummond, "Scalable monocular SLAM," in *IEEE Conf. on Comp. Vis. Pat. Rec.*, vol. 1. IEEE, 2006, pp. 469–476.

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE Int. Symp. on Mixed and Aug. Reality*, Nara, Japan, November 2007, pp. 1–10.

[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: a factored solution to the simultaneous localization and mapping problem," in *18th Nat. Conf. on A.I.* Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 593–598.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, pp. 91–110, November 2004.

[7] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Int. J. Rob. Res.*, vol. 21, pp. 735–758, 2002.

[8] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Rob. Res.*, vol. 27, no. 6, pp. 647–665, 2008.

[9] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *IEEE Conf. on Comp. Vis. Pat. Rec.* IEEE, 2007, pp. 1–7.

[10] A. Irschara, C. Zach, J. Frahm, and H. Bischof, "From structure-from-motion point clouds to fast location recognition," in *IEEE Conf. on Comp. Vis. Pat. Rec.* IEEE, 2009, pp. 2599–2606.

[11] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," *IEEE Int. Conf. on Comp. Vis.*, vol. 2, pp. 1470–1477 vol.2, Apr. 2003.

[12] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg, "Wide area localization on mobile phones," in *IEEE Int. Symp. on Mixed and Aug. Reality*. IEEE, 2009, pp. 73–82.

[13] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography," *Comms. of the ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[14] A. Kawewong, N. Tongprasit, S. Tangruamsub, and O. Hasegawa, "Online and incremental appearance-based SLAM in highly dynamic environments," *Int. J. Rob. Res.*, vol. 30, pp. 33–55, January 2011.

[15] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," *IEEE Conf. on Comp. Vis. Pat. Rec.*, vol. 2, pp. 2161–2168, 2006.

[16] Z. Zhang, "Parameter estimation techniques: a tutorial with application to conic fitting," *Img. and Vis. Comp.*, vol. 15, no. 1, pp. 59–76, 1997.

[17] G. Pandey, J. McBride, and R. Eustice, "Ford campus vision and lidar data set," in *Int. J. Rob. Res.*, 2011, in Press.

[18] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *Int. J. Rob. Res.*, vol. 29, no. 8, pp. 941–957, 2010.