Vision-Based Localization in Urban Environments

Michael McHenry*[a], Yang Cheng [a], Larry Matthies [a]
[a] Jet Propulsion Laboratory – California Institute of Technology
4800 Oak Grove Drive
Pasadena CA 91009

ABSTRACT

As part of DARPA's MARS2020 program, the Jet Propulsion Laboratory has developed a vision-based system for localization in urban environments that requires neither GPS nor active sensors. System hardware consists of a pair of small FireWire cameras and a standard Pentium-based computer. The inputs to the software system consist of: 1) a crude grid-based map describing the positions of buildings, 2) an initial estimate of robot location and 3) the video streams produced by the stereo pair. At each step during the traverse the system: captures new image data, finds image features hypothesized to lie on the outside of a building, computes the range to those features, determines an estimate of the robot's motion since the previous step and combines that data with the map to update a probabilistic representation of the robot's location. This probabilistic representation allows the system to simultaneously represent multiple possible locations. For our testing, we have derived the a priori map manually using non-orthorectified overhead imagery, although this process could be automated. The software system consists of three primary components. The first is a stereo-based visual odometry system that calculates the 6-degree of freedom camera motion between sequential frames. The second component uses a set of heuristics to identify straight-line segments that are likely to be part of a building exterior. Ranging to these straight-line features is computed using binocular or wide-baseline stereo. The resulting features and the associated range measurements are fed to the third software component, a particle-filter based localization system. This system uses the map and the most recent results from the first two to update the estimate of the robot's location. This report summarizes the design of both the hardware and software and describes the results of applying the system to the global localization of a camera system over an approximately half-kilometer traverse across JPL's Pasadena campus.*

Keywords: robotics, localization, vision, unmanned-ground vehicles, urban

1. INTRODUCTION

This paper describes a vision-based system that performs localization in outdoor urban environments by detecting building features and relating those features to an a priori map derived from an aerial image. Our approach is motivated by the desire to enable a human operator to intuitively direct an autonomous unmanned vehicle (AUV) to a specific goal location without the use of GPS. Implicit in this problem is the presumption that the operator has sufficient knowledge of the environment to identify a desirable goal location. This goal location would, we expect, be selected because it placed the robot's sensors and/or actuators in some particular relationship to features of interest in the environment.

The approach described here allows the operator to specify the goal position as a point within an overhead image. Given the rapidly expanding capability of commercial and military satellites as well as the availability of unmanned aerial vehicles, we believe assuming the availability of such imagery is reasonable particularly considering the benefits. Perhaps the greatest benefit is the ease of developing a simple and intuitive operator interface. In fact, the operator could also suggest a particular path or impose constraints on the path by which the robot navigates to the goal. This would allow the operator to make use of any knowledge he might have regarding desirable paths as well as various tradeoffs, for example between transit speed and stealth. Given that autonomous path following has already been demonstrated [1], the fundamental problem becomes that of localizing the vehicle with respect to the provided overhead image.

* Michael.C.McHenry@jpl.nasa.gov: Telephone (818) 354-2445 http://robotics.jpl.nasa.gov.

Our localization approach is currently purely vision based; no IMU or wheel encoder data is collected. The sensor system consists of a stereo camera pair containing two Firewire (1394) b/w cameras with image resolution of 640 by 480. The cameras are separated by a distance of 15 cm. The field of view (FOV) of each camera is 75^o vertical and 90^o horizontal. The results described in this paper were computed offline from a data set collected with the sensor system mounted on a four-wheeled pull cart at a height of approximately one meter. Stereo imagery was collected at 7.5 Hz while the cart was manually steered at a moderate walking pace across the JPL campus for a distance of more than 500 meters. The cart remained on road surfaces during the entire traverse.

The software processing consists of three major components.

- Visual Odometry
- Urban Feature Detection and Ranging
- Particle Filter-based Localization

Visual Odometry automatically tracks point features in the imagery and computes the relative camera motion between sequential frames. The urban feature detection and ranging component uses the same imagery to identify and measure the 3D position of linear features that are likely to lie on the exterior wall of a building. The output of each of these components is fed to the third component, an implementation of particle-filter based localization developed by Sebastian Thrun et al and freely available as part of the CARMEN robot software distribution [2].

2. VISUAL ODOMETRY

Visual Odometry is used to provide incremental motion estimates. The visual odometry or ego-motion estimation algorithm was originally developed by Matthies [3]. Following this work, some minor variations and modifications were suggested for improving its robustness and accuracy [4][5]. In a nutshell, the algorithm selects point features, uses multi-resolution area correlation to match them in stereo, tracks them through the image sequence, and uses the tracking results to estimate the six degree of freedom camera motion between consecutive stereo pairs. In preparation for fielding the algorithm on the MER rovers currently operating on Mars, we evaluated its performance using earth-based rover testbeds and found that it can achieve 2% accuracy over distances of 30 meters [6].

The basic steps of visual odometry are as follows. First, features that can be easily matched between stereo pair images and tracked between image steps are selected using the Forstner interest operator. A minimum distance between features constraint is enforced to ensure features are selected evenly across the image scene. Next, stereo matching is used to determine the 3D positions of each point feature. The same process is then repeated for the next stereo pair. Correspondences between the features in consecutive stereo pairs are determined by performing area-based correlation in a 2D window. Figure 1 shows an example of the features detected and the motion of those features from the previous frame.

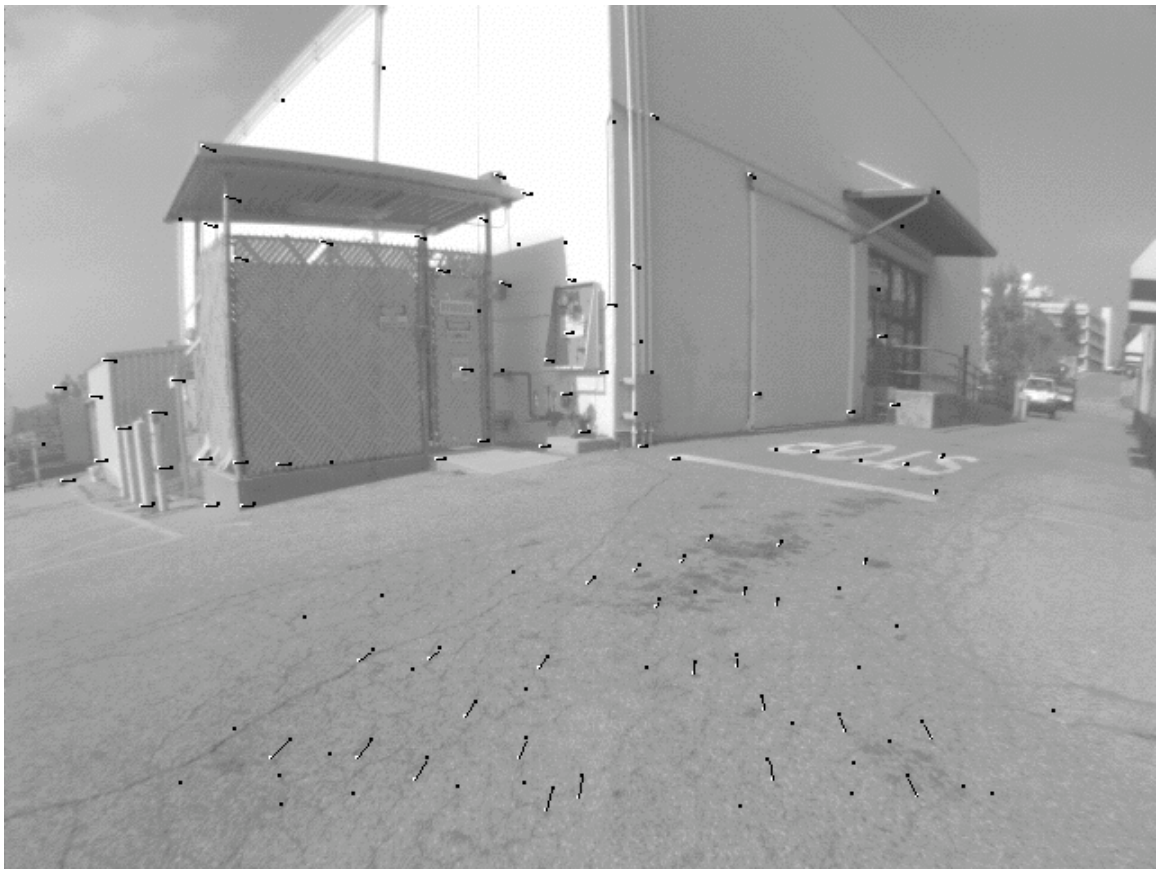

Figure 1: Example of features used by Visual Odometry. The vectors show the displacement of features between sequential images. In this image, one can see there is both forward camera motion (as evidenced by those features on the ground) as well as a strong rotational component (as evidenced by those features lying on the buildings).

A maximum likelihood estimator takes into account the 3D position differences and an associated error model to estimate the relative camera motion between frames. Let Q_{pj} and Q_{cj} be the observed features positions before and after a robot motion. Then we have

$$Q_{cj} = RQ_{pj} + T + e_j$$

where R and T are the rotation and translation of the robot and e_j is the error in the observed position of the j-th feature. The 3 axis rotations and translation T are directly determined by minimizing the summation $\sum r_j^T W_j r_j$, where $r_j = Q_{cj} - RQ_{pj} - T$ and W_j is the inverse covariance matrix of e_j which is a 3x3 matrix determined by the feature location relative to the camera positions and to the quality of the correlation. The minimization of this nonlinear problem is solved using an iterative linearization technique[6]. Two desirable properties of maximum-likelihood estimation make the algorithm powerful. First, it estimates the 3 axis rotations directly so that it eliminates the error caused by rotation matrix estimation such as by the least-squares estimation. Secondly, it incorporates error models in the estimation, which greatly improves the accuracy.

The errors associated with position estimates derived solely from visual odometry, or any incremental approach (commonly referred to as "dead reckoning"), will grow without bound. Figure 2 shows the complete path reconstructed from visual odometry. In examining the results, we identified two brief intervals in which the computed incremental heading measurements were substantially poorer than during the rest of the traverse. In the first such interval, the rotation rate exceeded a software threshold defining the maximum frame-to-frame pixel displacement. In the second interval, a bus passing directly in front of the cameras caused the system to mistake the motion of the bus with camera

motion. Planned improvements, which include the addition of a compass and the incorporation of JPL's vision-based moving object detection capability described in [7] should address both of these issues.

Figure 2: The result of Visual Odometry overlaid on the overhead imagery. X marks the actual traverse end point.

3. URBAN FEATURE DETECTION

The second major component of our localization system is urban feature detection. Robustly recognizing buildings and determining their boundaries in 3 dimensional space is challenging for many reasons:

- Many building walls lack texture which makes stereo ranging difficult.
- The diversity of building types and construction (e.g. width, height, color, texture, materials and geometric complexity) makes it hard to automatically extract relevant features.
- Even when a building feature has been recognized, computing the 3D position of that feature may not be feasible using simple binocular stereo. For example, as horizontal lines become coincident with the epipolar line, the mathematics of binocular stereo (in a stereo pair with horizontally mounted cameras) become singular.

Rather then try to create an explicit 3D model of the buildings we adopted a more modest goal, namely detecting and computing the 3D coordinates of vertical and horizontal lines which are likely to be part of or close to a building exterior. The basic processing sequence is as follows:

- Edge Detection
- Straight Line Detection
- Gradient Filter
- Roofline Detection
- 3-D Reconstruction
- Heuristic Pruning

3.1 Edge Detection

Edges are first extracted using a Canny edge detector [8]. This results in a set of linked lists of edge pixels and two gradient (horizontal and vertical) images.

3.2 Straight Line Detection

Edges which are well approximated by straight-line segments are extracted via a recursive method which repeatedly divides the link lists containing edge pixels. A straight line is formed from two ends of each linked list and the maximum distance (*dm*) between that line and any contained edge pixel is found. If *dm* is less than a threshold, the process stops and the line segment is returned. Otherwise the linked list is divided at the pixel that determined *dm* and the process continues on the two resulting linked lists (Fig. 3). The process stops only when every edge has been decomposed into straight-line segments. The length of each segment is computed and short segments are discarded.

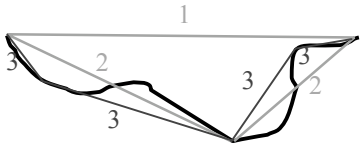

Figure 3. A recursive method is used to extract straight-line segments from a linked list of edge pixels. The numbers indicate the level of recursion.

3.3 Gradient Filter

Any line segments of length greater than some manually determined threshold (currently 30 pixels) are presumed to be man-made and are automatically passed on to the next processing stage. For shorter line segments we use a heuristic that the intensity gradient along man-made features will be similar across the length of the segment. Thus we apply the following test at each pixel within the segment:

$$\frac{gx_i gx_{i-1} + gy_i gy_{i-1}}{sqrt(gx_i^2 + gy_i^2)sqrt(gx_{i-1}^2 + gy_{i-1}^2)} > tg$$

where *gx* and *gy* are the intensity gradients and *tg* is a threshold, which is close to but less than 1. At those edge pixels where the test fails, the line segment is sub-divided and processing continues on the remaining edge pixels. A more accurate line fit based on all the constituent pixels is then computed using the least squares method. Fig 4.c shows an example result at this stage in the computation.

3.4 Roofline Detection

Detecting rooflines in the ground imagery is important for two reasons. First these rooflines correspond directly to the building extents visible from the aerial image and hence our map. Secondly, identifying the roofline provides valuable context that aids in the identification of other prospective building features. Our simple heuristic for identifying rooflines is that any line that is more than 30 pixels long and has no other prospective roofline anywhere above it is a potential roofline (pending the 3D analysis described below). All features that are entirely above the set of prospective rooflines or below the horizon line (as determined by the visual odometry derived camera pose) are discarded.

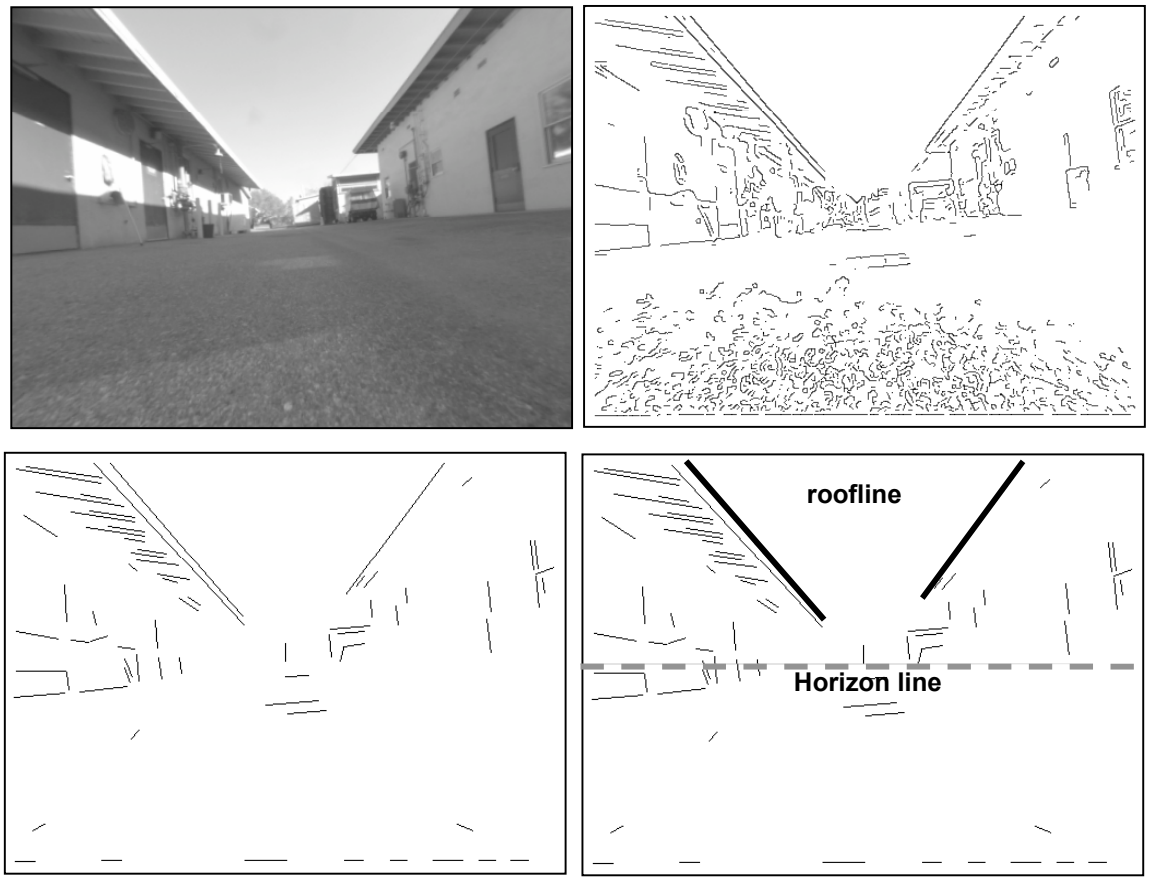

Figure 4: a) Original image b) Results of Edge Detection c) Extracted straight line Segments d) Detected rooflines and horizon line

3. 5 3-D Reconstruction

This step computes the 3D position of the detected linear segments. The required triangulation can be performed by either binocular stereo (using the knowledge of the relative positions between the left and right cameras determined through camera calibration) or by wide-baseline stereo (using the relative camera position between two successive frames as determined by visual odometry). Although the relative position between the left and right cameras in a calibrated stereo pair is far more accurate than the camera positions derived from visual odometry, the accuracy with which each approach can determine 3D feature position depends on three factors:

- The baseline length. The longer the baseline is, the more accurate the position estimation is.
- The matching error. When the image gradient direction is close to perpendicular to the epipolar line, a large matching error frequently results.
- The position of the feature with respect to the cameras.

Matching error will be greatest when there is little intensity gradient along the epipolar line (the intersection of the plane determined by the camera centers and the target feature with the second image). Our stereo pair has the cameras

mounted horizontally (i.e. at the same height), thus the epipolar lines are approximately horizontal. The result is that matching errors for binocular stereo will be most pronounced when viewing horizontal lines in the image.

Care must be taken when using wide-baseline stereo since the effective baseline length is a function of both where in the field of view the feature appears and also the relative position of the camera in the two frames. If the camera is moving in straight line along the optical axis, the effective baseline to features close to the center of expansion (in the center of the image) approaches zero.

Our algorithm chooses between binocular stereo and wide-baseline stereo by first determining the angle between the intensity gradient and the (binocular stereo) epipolar line. If this angle is below a predetermined threshold we use binocular stereo to compute the feature's 3D position. Otherwise, we evaluate the suitability of wide-baseline stereo by computing the angle between the epipole (the line through the two camera centers) and the ray determined by the camera position in the first frame and the feature in question. If this angle exceeds a threshold, meaning the feature is sufficiently far from the center of expansion, the feature's 3D position is computed using wide-baseline stereo.

The mathematics for triangulation is the same for both the binocular and wide-baseline stereo cases. In both cases, the linear features are selected from the left camera of the stereo pair. If binocular stereo is being used, the other image is the right image from the same stereo frame. Whereas if wide-baseline stereo is being used, the other image is the left image from the following frame. In either case, the 3D position of each endpoint can then be calculated as follows:

$$P_i = C_{left} + r_i \frac{(r'_1 \times r'_2) \cdot (C_{other} - C_{left})}{(r_1 \times r_2) \cdot r_i} \qquad i = 1,2$$

Where P_1 and P_2 denote the 3D positions of the two line-segment endpoints, C_{left} and C_{other} are the camera locations when the left and the other image were acquired, and r and r' denote rays from the camera centers to the end points on the first and second image respectively**.**

3.6 Heuristic Pruning

The last step of urban feature detection filters the remaining linear segments using the computed 3D data. In particular,

- All lines that are not approximately vertical or horizontal are discarded.
- Horizontal lines above 2 meters are selected.
- Vertical lines that extend above 2 meters are selected.
- Sets of parallel lines are selected.

Figure 5 shows some representative examples of the features detected during the traverse through the JPL campus.

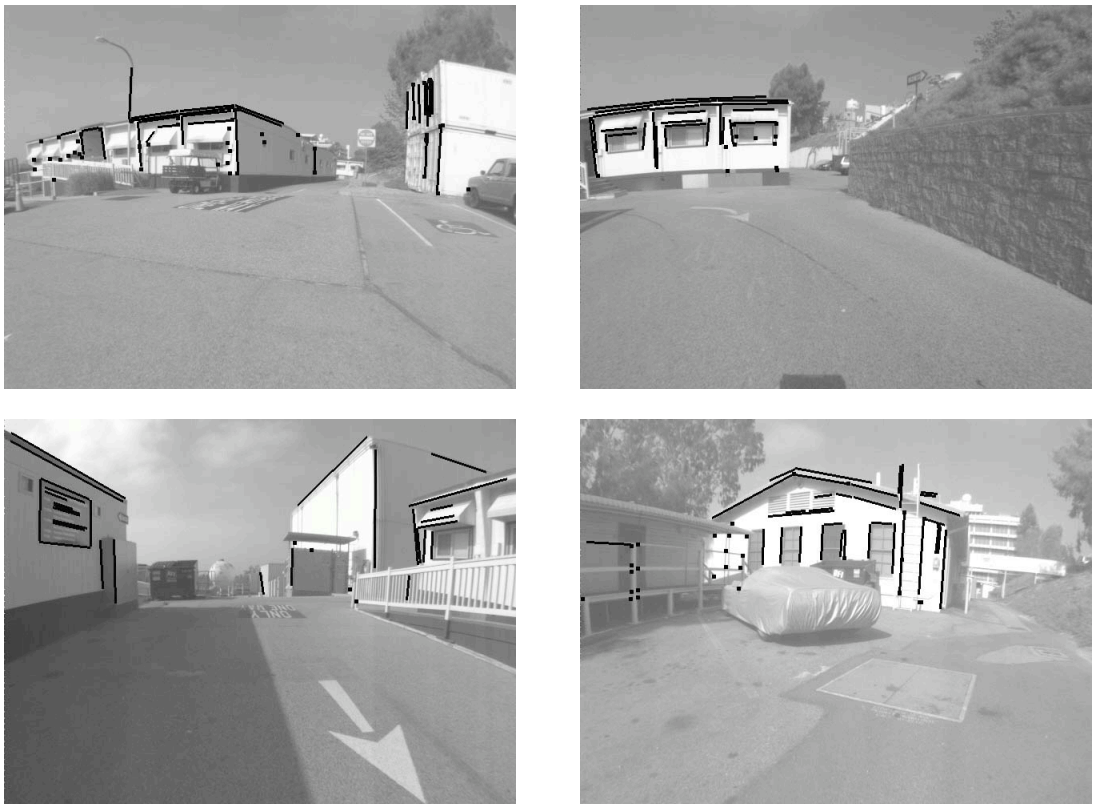

Figure 5: Images collected during the traverse annotated with the building features detected.

4. PARTICLE-FILTER BASED LOCALIZATION

This section describes how the results of visual odometry and building feature detection are integrated with particle-filter based localization software publicly available as part of the CARMEN robot software library. Typically the a priori map used by this software would be constructed automatically from data previously collected as the robot traversed through an environment making measurements with the same LIDAR range sensor(s) used for online localization (for traverses through the same environment).

The problem addressed in the work described here is more challenging in several ways. First, the vision-based ranging used here does not match the accuracy of time of flight LIDAR ranging. And while most LIDARs used in autonomous ground vehicles sense only in a single plane, they typically have a horizontal field of view that is 2-3 times that of our camera optics. Secondly, in our approach the a priori map comes from a fundamentally different sensing modality than that used for the ranging. Establishing the relation between the linear segments as detected from ground imagery and a map constructed from aerial imagery is considerably more challenging than relating measurements taken from the ground with a map constructed using the same sensor and from approximately the same perspective. The key advantage of our approach is that it allows a ground robot to navigate to a particular goal point in an environment in which neither it nor any other ground based sensor system has previously traversed and it does so without relying on GPS.

It is important to note that the a priori map used was created with minimal effort and is unlikely to be very accurate. The map is derived directly from the aerial image by manually outlining the buildings; a process which no doubt introduced significant errors. In addition, the aerial image was not corrected for lens distortion nor for perspective effects. It is also the case that building outlines as evident from an aerial image can be a poor indication of the true location of external walls because of roof overhangs. Furthermore, often the only available aerial imagery will be from a time far enough in the past that new construction (or deconstruction) will have occurred in the intervening time period.

The theory of particle-filter based robot localization is well described elsewhere [9] and will not be covered in detail here. The key notion is that a particle filter produces an approximate probability density function (describing the robot's position as well as heading) using Monte Carlo techniques. While particle filters and Kalman filters both share rigorous mathematical underpinnings, one key advantage of particle filter based approaches is their ability to represent arbitrary (rather than just Gaussian) distribution functions. For the experiment described here, we used publicly available software provided as part of the Carnegie Mellon Robot Navigation Toolkit (CARMEN) software package developed by a team led by Prof. Sebastian Thrun. The software is available from http:www.cs.cmu.edu/~carmen.

The creation of the a priori map from the aerial image was facilitated by a utility included in the CARMEN software distribution that converts gray scale images directly into an occupancy map. Each pixel intensity is translated to a floating point value representing the probability that the corresponding map cell is occupied. In our case, we simply loaded the aerial image into a paint program, outlined the buildings and colored between the lines, filling those areas within the building outlines as black and those in exterior areas as white. The resulting image was then converted to a CARMEN map using the provided utility. Figure 6 shows the manually created map.

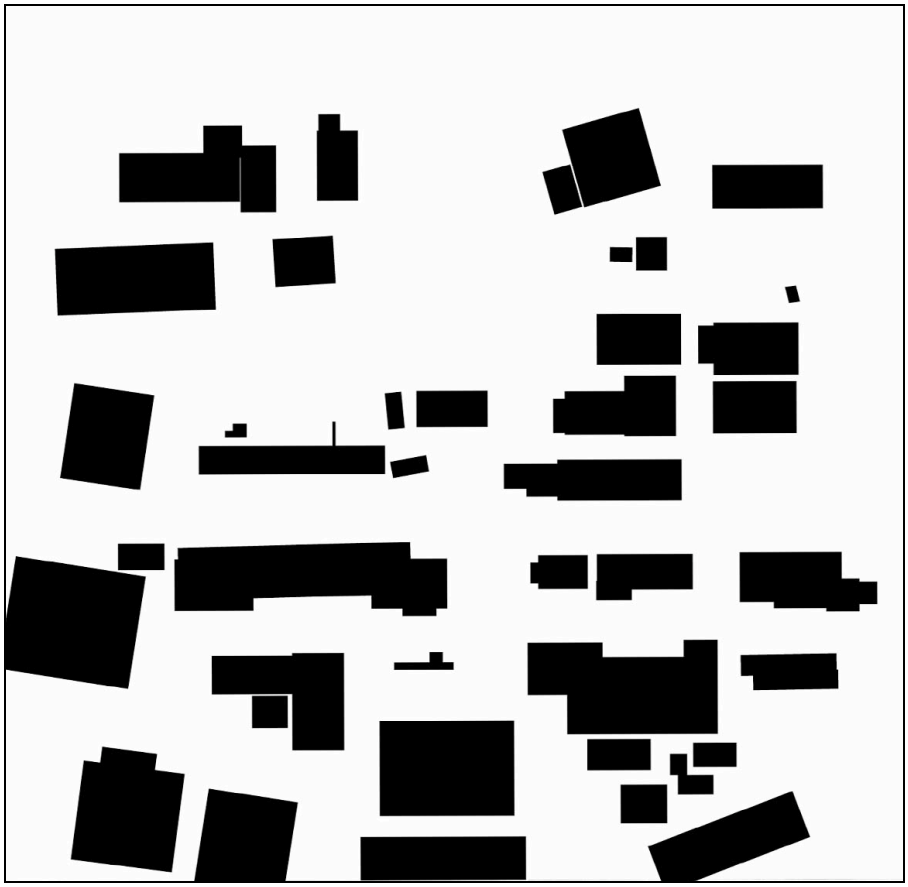

Figure 6: Map manually created from overhead image.

In addition to the map, the localization software requires two basic types of input: position estimates (i.e. odometry) and range measurements. The format of each was clearly defined so substituting the system's vision derived data was simply a matter of converting to CARMEN's data formats. CARMEN was designed for planar robots so our conversion of Visual Odometry derived position estimates simply discarded the height, roll and pitch portions. One of the key attributes of particle filter based localization is the ability to incorporate knowledge of measurement uncertainty in a rigorous manner. Thus in addition to providing odometry data and range measurements we also needed to provide a description of the uncertainty associated with each. For odometry this requires only specifying the standard deviations

associated with the incremental translational and rotation results obtained from visual odometry. Since we did not have ground truth data available, we arrived at these parameters manually through a trial-and-error process.

The translation of range measurements was more involved because the planar model implicit in the localization software does not precisely fit our 3D vision-based measurements. A single axis LIDAR will, in general, return only a single range measurement for each particular bearing, while our vision-based system frequently finds multiple features at different heights along a particular bearing. The simple strategy we adopted was to generate simulated single axis LIDAR data by projecting all the 3D features onto the horizontal plane and sampling the field of view at a regular interval (1 degree). The distance to the first feature found is returned as the range measurement. This is a reasonable strategy since our conversion of the overhead image produces a building outline that has maximum extent (i.e. if the bottom floor extends over a wider area than higher floors, the map will describe the outline of the bottom floor). This strategy also matches well with the probabilistic sensor model on which the localization algorithm is based and is illustrated in Figure 7. The sensor model can be divided into four areas. In the case of LIDAR data, the wide area of low probability starting at d=0 reflects the possibility of unmapped obstacles interrupting the path to the building wall. In our case, this initial area represents the potential for false positive feature detections (or new construction not represented in the map). The Gaussian shaped hump at the center represents measurement noise around the true distance to the obstacle. The following area of very low probability represents the possibility of ranging to something that is far beyond/within the building boundary. The peak at the far right occurs at a special value used to signal that no range measurement was returned. In our case this corresponds to no building feature being detected along that bearing.

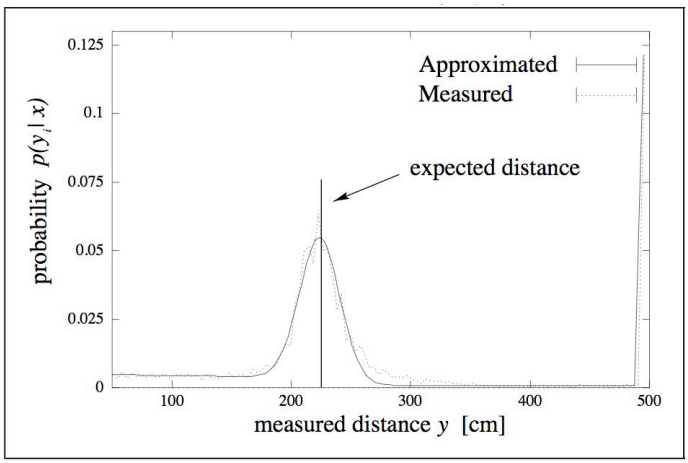

Figure 7: The sensor model used by the localization software. From Thrun et al [9], used with permission.

Note that there were a significant number of false positives in the urban feature detection and the computed range values were certainly noisier than those obtained from a time-of-flight LIDAR. In addition, there were long sections of the traverse in which no features were detected because trees and automobiles lining the street obscured the buildings. Yet despite these challenges Figure 8 shows the system was able to localize the sensor system across the entire traverse. The fact that our experiment proved successful despite these difficulties is a testament to the robustness of the particle filter-based localization algorithm and its implementation in CARMEN as well as the excellent incremental position estimates obtained from our visual odometry implementation.

Figure 8: Results of complete system overlaid on overhead image. The X marks the true end point of the traverse.

5. FUTURE WORK

The experiment described here represents a proof of concept. Integrating all the software components into a complete system capable of online localization has not yet been accomplished. We hope to complete this integration and test the system more thoroughly in a range of urban environments.

Besides the system integration necessary to field these components in an online implementation, there are many significant improvements that can be implemented. One very easy enhancement that would offer considerable performance improvement is the addition of a compass. By providing an absolute measurement and thereby bounding the overall heading uncertainty the task of the particle filter portion of the system will become considerably easier. Adding side-looking stereo pairs would provide additional features particularly when the robot is moving down urban streets.

It is also desirable to enhance the breadth of features used beyond the current limited set. There is considerably more information in the overhead image that could be automatically incorporated to a map and recognized in the video stream.

For example, road boundaries, surface characteristics (e.g. asphalt versus concrete) and even road markings such as cross walks are often visible in overhead imagery and should be readily observable from a ground vehicle. In some environments, individual trees and other vegetation might also be useful to include in the map and feature detection.

It will also be useful to extract planar wall sections more directly. The relationship of planar surfaces in pinhole camera imagery can be represented as a homography transformation. The planar sections extracted could be used as features within the visual odometry algorithm to improve accuracy and robustness. In addition, we could use the detection of vertical walls to measure pitch and roll thus (in combination with a compass) bounding the systems overall orientation error. In addition, large planar features should be trackable for long distances allowing the application of multiple baseline methods which will improve building surface reconstruction and ranging.

Currently there is no feedback between the localization component and the urban feature detection. Such feedback would enable a degree of active sensing/perception. For example, at those locations where the positional certainty is very high, the urban feature detection could focus its computation on those areas of the imagery most likely to contain key features such as roof edges or the corners of buildings. And since we are presuming the availability of the a priori map, the robot could incorporate its knowledge of where the most informative features are likely to be observed in its path planning.

The probabilistic sensor model incorporated in the localization software was developed for use with single axis LADARs not for vision-based feature detection. Hand tuning of the sensor model parameters were sufficient for the limited experiment described here, but empirical characterization of detection and ranging errors should enable more robust and accurate localization.

We would also like to simplify the process of map generation. Obviously the process of converting an aerial image into map form could be automated or at least semi-automated. Ultimately, we would like to explore the system's ability to localization using an even less accurate map, for example a map which a person might sketch by hand based on recollection or a single ground based view.

ACKNOWLEDGMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was supported by the Mobile Autonomous Robot Software Robotic Vision 2020 Program of the Defense Advanced Projects Agency (DARPA), Information Processing Technology Office under contract NAS7-03001 and task order NMO715515.

REFERENCES

1. R. Hogg, A. Rankin, M. McHenry, D. Helmick, C. Bergh, S. Roumeliotis, L. Matthies Sensors and Algorithms for Small Robot Leader/Follower Behavior *In proceedings of the SPIE 15th AeroSense Symposium* Orlando, Florida, April 2001
2. Carnegie Mellon Robot Navigation Toolkit (CARMEN) http://www.cs.cmu.edu/~carmen
3. L. Matthies. *Dynamic Stereo Vision*, PhD thesis, Carnegie Mellon University, October 1989.
4. C. F. Olson, L.H. Matthies, M. Shoppers, and M. Maimone, Robust stereo ego-motion for long distance navigation, In *proceedings of the IEEE Conference in Computer Vision and Pattern Recognition*, Vol. 2. 2000.
5. C. F. Olson, L.H. Matthies, M. Shoppers, and M. Maimone, Stereo ego-motion Improvements for robust rover navigation, In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1099-1104, 2001.
6. C.F. Olson, L. Matthies, and M. Schoppers, Rover Navigation using Stereo Ego-motion. *Robotics and Autonomous Systems*, 43(4), June 2003
7. A. Talukder, and L. Matthies, Real-time Detection of Moving Objects from Moving Vehicles using Stereo and Optical Flow, *IEEE Conference on Intelligent Robots and Systems (IROS) 2004*, Sendai, Japan, Sept. 2004.
8. J. Canny, A Computational Approach to Edge Detection, *IEEE PAMI*, 8 (6) 1986.
9. S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo Localization for Mobile Robots. *Artificial Intelligence*, 128(1-2), 2001