Accurate waypoint navigation using non-differential GPS

Michael H. Bruch, G.A. Gilbreath, J.W. Muelhauser, J.Q. Lum
Space and Naval Warfare Systems Center
53406 Woodward Rd., San Diego, CA 92152
www.spawar.navy.mil/robots/
bruch@spawar.navy.mil

ABSTRACT

Land-based waypoint navigation usually requires accurate position information to effectively function in either natural or man-made terrain. Most systems solve this problem by using differential GPS and/or high-quality, expensive inertial navigation systems. In an effort to make waypoint navigation available to smaller tactical platforms, a tightly packaged, portable and inexpensive waypoint navigation system was developed. This system was implemented on the Man Portable Robotic System (MPRS) Urban Robot (URBOT)[1]. The package uses inexpensive sensors and a combination of standard Kalman Filter and waypoint following techniques along with some novel approaches to compensate for the deficiencies of the GPS and gyroscope sensors. The algorithms run on a low-cost embedded processor. A control unit was also developed that allows the operator to specify path waypoints on ortho-rectified aerial photographs.

1. Background

The goal of this project was to develop a robust waypoint navigation capability for a small mobile robot that did not rely on the availability of differential GPS. Here waypoint navigation is defined as the process of automatically following a predetermined path defined by a set of geodetic coordinates. The requirement for using non-differential

GPS stemmed from the idea that in a tactical situation, the operator is not going to have the time or resources to set up a local differential base station at each new location. Unlike large vehicles, the limited space and power on a small robot precludes the traditional practice of using a highly accurate inertial navigation system (INS).

All of the subsequent discussions are a direct result of testing performed on the MPRS URBOT outfitted with a NovAtel OEM4 dual-frequency GPS receiver, a Systron Donner QRS11 quartz gyro, a Microstrain 3DM electromagnetic compass, and Hall-effect sensors for the odometry.

2. GPS Deficiencies

The errors in an uncorrected GPS signal come in many forms and arise from a variety of different sources. In this paper these errors have been divided into two broad categories: 1) high frequency noise and 2) long-term drift. The first category pertains to the errors that manifest themselves as high frequency noise or spikes. These errors are easily identifiable on a 2-D plot of the GPS track recorded from a moving platform (Figure 1). Although, no attempt has been made to formulate an explicit definition of what constitutes noise, a general example would be single-epoch jumps in the GPS position. An epoch is one GPS cycle (milliseconds). The difficulty arises from the fact that in some instances the position can jump several meters and then either jump back on the next epoch or maintain that new position for a few seconds or indefinitely. If the new position is maintained for more than approximately 30 seconds, then it is no longer considered noise but lies in the gray area between the two categories.

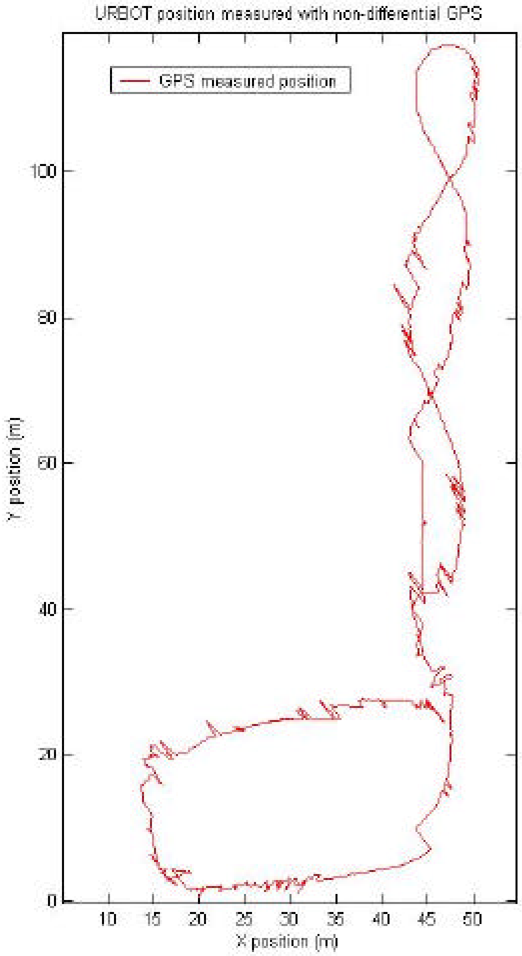


Figure 1. Non-differential GPS track of the URBOT

Experience has shown that the two main causes of GPS noise are satellites coming in and out of the view of the GPS receiver and multi-path effects. The magnitude of these errors varies from a few feet to hundreds of feet.

The second category of GPS error is classified as drift. These errors are much more difficult to see on a track plot, since they change over a period of hours rather than seconds like the noise errors. It is difficult to determine the exact cause of these types of errors, but they are typically attributed to atmospheric effects in the ionosphere and troposphere and satellite geometry. The magnitude of these errors can vary from no error at all to thirty feet or more.

3. GPS noise remedy

In order to perform reasonable waypoint navigation, a robot needs to have a relatively noise-free estimate of its current state. Obviously, a non-differential GPS solution alone is not capable of providing that estimate.

The most common solution for solving the problem of GPS noise (and the solution used here) is to augment the GPS with other sensors and employ a Kalman Filter to optimally combine all of those sensor inputs. An inertial sensor is an ideal companion for the GPS in a navigation package, as the two sensors have complementary errors (i.e., inertial sensors generally have very little noise but drift without limit, whereas, GPS is quite noisy but has finite drift). The benefits of the Kalman Filter are depicted in Figure 2, which is the same GPS plot shown in Figure 1 but with the Kalman Filter state estimate overlaid on it. Note that almost all the spikes in the GPS noise have been smoothed out. The Kalman Filter does an excellent job of compensating for the noise in the GPS position, but is of no help with the long-term drift error in the GPS position.

3.1. Kalman Filter

This paper is not intended to be a comprehensive guide to Kalman Filtering. Readers unfamiliar with the Kalman Filter and it's applications in mobile robots are encouraged to read "*A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles* [2]."

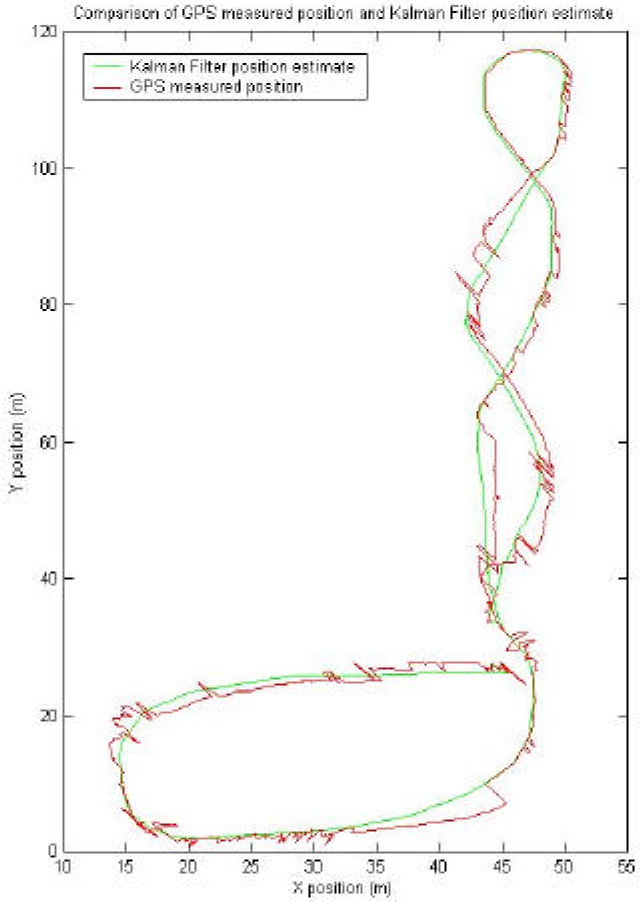


Figure 2. Kalman Filter vs. GPS track

We used an Extended Kalman Filter with nine inputs (sensor measurements) and seven outputs (the vehicle states) as shown in Tables 1 and 2. The Kalman Filter is formulated using the body frame system[2], which makes the following assumptions:

1. the vehicle translates only along the body y axis (see Figure 3)
2. the vehicle rotates only around the body z axis

A low-dynamics assumption has also been made so that no acceleration states are required.

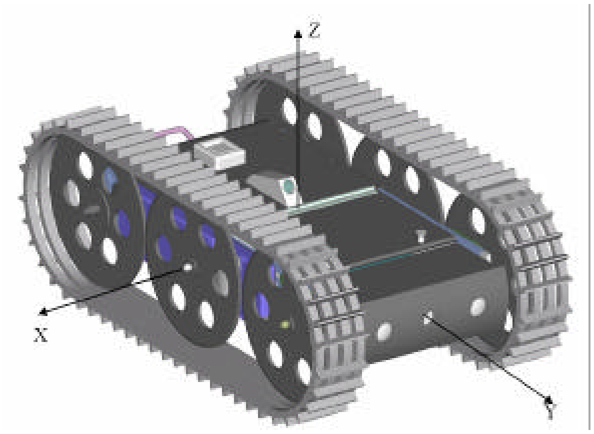
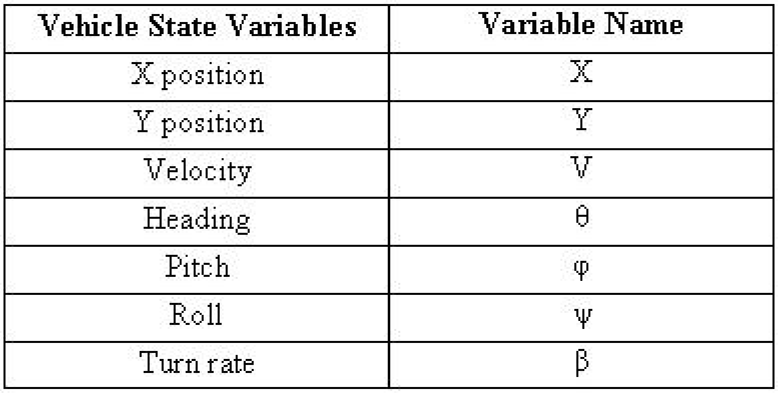


Figure3. Coordinate system assigned to the URBOT

Measurement	Sensor
X position	GPS
Y position	GPS
Heading	GPS
Velocity	GPS
Heading	Compass
Pitch	Compass
Roll	Compass
Velocity	Encoders
Turn rate	Gyro

Table 1. Kalman Filter measurements and associated sensors



Vehicle State Variables	Variable Name
X position	X
Y position	Y
Velocity	V
Heading	θ
Pitch	φ
Roll	ψ
Turn rate	β

Table 2. Kalman Filter states

Using this system the state transition matrix, F, is given as

$$F = \begin{bmatrix} 1 & 0 & -\sin\psi\cos\theta dt & 0 & 0 & 0 & 0 \\ 0 & 1 & \cos\psi\cos\theta dt & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \sin\theta dt \\ 0 & 0 & 0 & 0 & 1 & 0 & -\tan\theta\cos\phi dt \\ 0 & 0 & 0 & 0 & 0 & 1 & (\cos\phi dt / \cos\theta) \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where the vehicle state, x_{k+1}, at time t_{k+1} is given by

$$x_{k+1} = F_k x_k \tag{2}$$

Equation 2 is essentially the dead reckoning equations written in matrix form. For example the X position at the next time interval will be the current X position plus the velocity, V, multiplied by ($-\sin\psi\cos\theta dt$), where dt is the time interval. Equation 2 is one of the five equations that make up the basic Extended Kalman Filter used here (see pseudo code in the appendix).

With equation 2 the Kalman Filter predicts the next vehicle state based on the current state. It then uses the sensor measurements and their associated variances to correct the prediction. In addition, if GPS measurements are not available, the Kalman Filter will automatically degrade to using just the gyro, compass, and odometry in a dead reckoning mode to continue to provide a full state estimate.

To address the gray area between GPS noise and drift errors mentioned above, a preprocessing technique has been employed that dynamically adjusts the measurement covariance matrix, R, in the Kalman Filter. In layman’s terms, the R matrix defines the believability of each sensor measurement. A preprocessing step increases the position variance in the R matrix when the GPS location jumps over two meters. If the GPS stays in the new location, the position variance elements in the R matrix are gradually decayed to their original value. In this way the Kalman Filter slowly gains confidence in the position measured by the GPS after it experiences a large jump. During this period of low confidence, the Kalman Filter is relying heavily on the gyro and odometry sensors. Because the gyro is subject to drift and the odometry sensors suffer from track slip the Kalman Filter cannot provide accurate position estimates for long distances without a GPS fix.

The track slip errors can have a very large adverse effect on the accuracy of the Kalman Filter when operating without GPS. For instance, if the odometry sensors have been tuned to provide accurate speeds on pavement, they will most likely have large errors when operating on gravel or sandy terrain. For that reason, a technique known as state vector augmentation[3] has been employed to mitigate those errors. State vector augmentation is essentially the process of adding artificial states to the Kalman Filter to estimate those values. In this case the additional state is the track slip error. With this technique the Kalman Filter continuously calculates the track slip error (actually the odometry velocity error) by using the Kalman Filter velocity estimate (which in turn is heavily influenced by the GPS velocity measurement). In this way, when GPS is lost the odometry sensors should provide a relatively accurate estimate of the vehicle’s speed as long as the terrain surface doesn’t change significantly after loss of the GPS signal.

Figure 4 shows a track plot where the GPS was artificially turned off part way through the run. Notice that after almost 120 meters of travel without the aid of GPS the Kalman Filter position estimate is still within approximately 5 meters of the GPS position. The errors that do occur are mostly due to inaccuracies in the vehicle heading estimate which is measured indirectly with a quartz gyro. In this example it proved to be advantageous to also turn off the electronic compass when the GPS is lost. That is in part due to the fact that the loop at the bottom of the figure is actually encircling a metal shed. Because of this and general inaccuracies of the compass it has become standard practice to only use the compass when the GPS heading is also available or when the gyro measurement is not available.

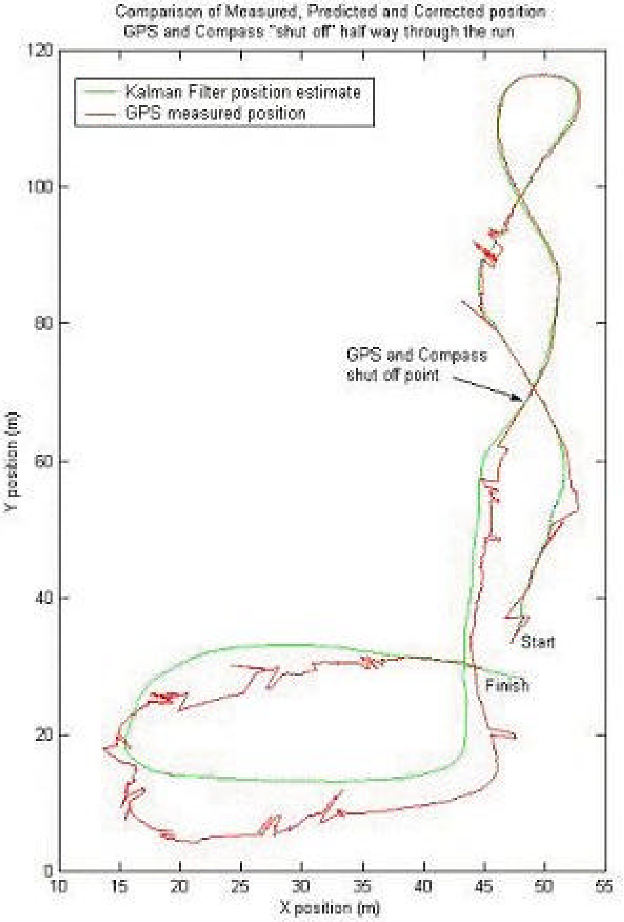


Figure 4. Example of Kalman Filter operating without GPS

The accuracy of the Kalman Filter position estimate would almost certainly be improved if a more accurate gyro were used such as a fiber optic gyro (FOG). Figure 5 shows the same track plot but without the track slip compensation. This demonstrates how useful the concept of state vector augmentation can be. To further improve the accuracy work is in progress to add information on the vehicle pitch to the track slip compensation calculation. It is reasonable to assume that the track slip will be a function of the slope the vehicle is negotiating. However, more data needs to be collected to determine what that function is. Additional state vectors could be included to compensate for other sensor errors such as gyro bias and drift.

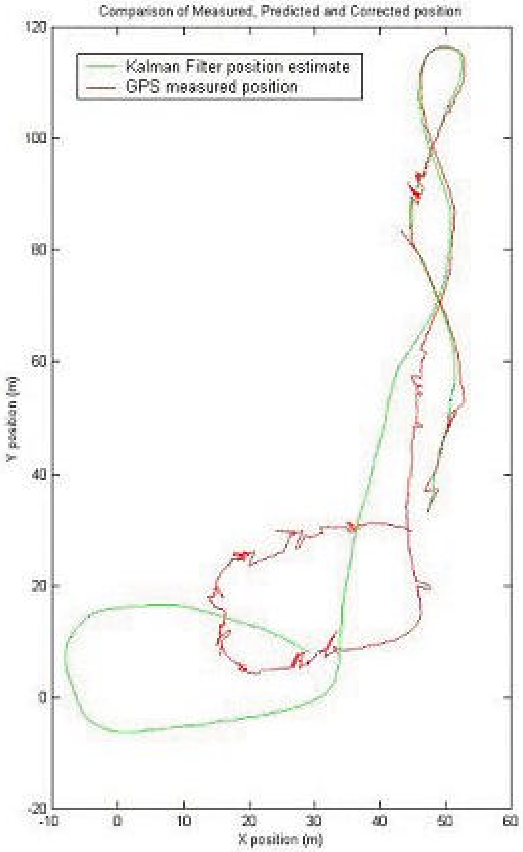


Figure 5. Same plot as Figure 3 but without track slip compensation

3.2. Position Update Message

To address the second category of GPS errors (drift) a position correction or update message was developed. The basic concept is to use landmarks in the terrain that can be correlated to a known position. In this case, the terrain recognition was done via the human supervisor using the real-time video image from the robot and an aerial photo of the area of interest. For example, the operator locates a unique object on the photo that is in the general area of the robot. Then the operator drives the robot to that location and, using the OCU (see section 5), selects the location on the photo that is now known to be the correct location. That geodetic coordinate is then sent to the robot which uses that coordinate to calculate a correction for the GPS position measurement. That correction is then applied to all subsequent measurements much like a differential correction. This method has proven to be very practical and useful.

4. Path Following

The Kalman Filter process described above provides only an estimate of the robot's current state (position, heading, velocity, etc.). A separate process is needed to perform the waypoint following task. That process includes receiving and parsing the path message sent from the OCU, determining the robot's current position on the path, calculating the current desired heading and velocity, and executing the PID controller to obtain that heading and velocity.

There are many different path-following techniques described in the literature. The one employed here is most commonly known as *follow-the-carrot/goal*[4]. Generally this technique uses a proportional controller fed by the heading error, whereas we have implemented a full PID controller. The heading error is the difference between the heading to the goal point and the robot's current heading. The goal point is defined as a point on the path that is some fixed distance (the look-ahead distance) ahead of the robot. Figure 6 shows a simulation of the path following algorithm developed in Matlab. Part of the path message sent from the OCU to the robot is a point-tolerance value that defines the distance from the waypoint that the robot must obtain before it considers that point to have been reached. This causes the robot to cut corners. If the look-ahead distance is such that the trajectory of the robot will miss a waypoint then the goal point is pulled back along the path until the trajectory runs through the point tolerance circle. In fact, the goal point calculation becomes quite tedious after one takes into consideration all of the possible path types, look-ahead distances and point tolerance distances.

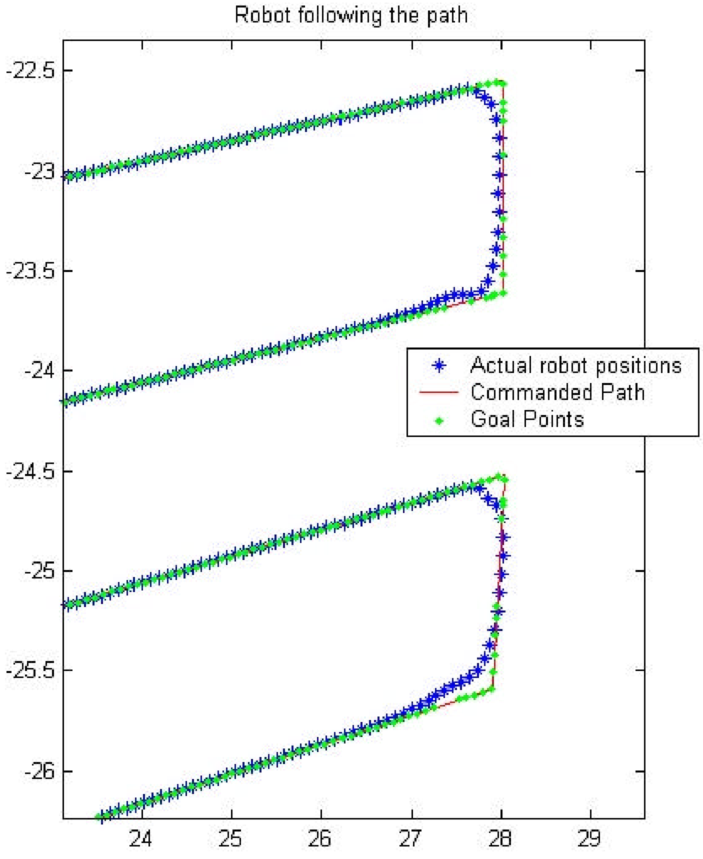
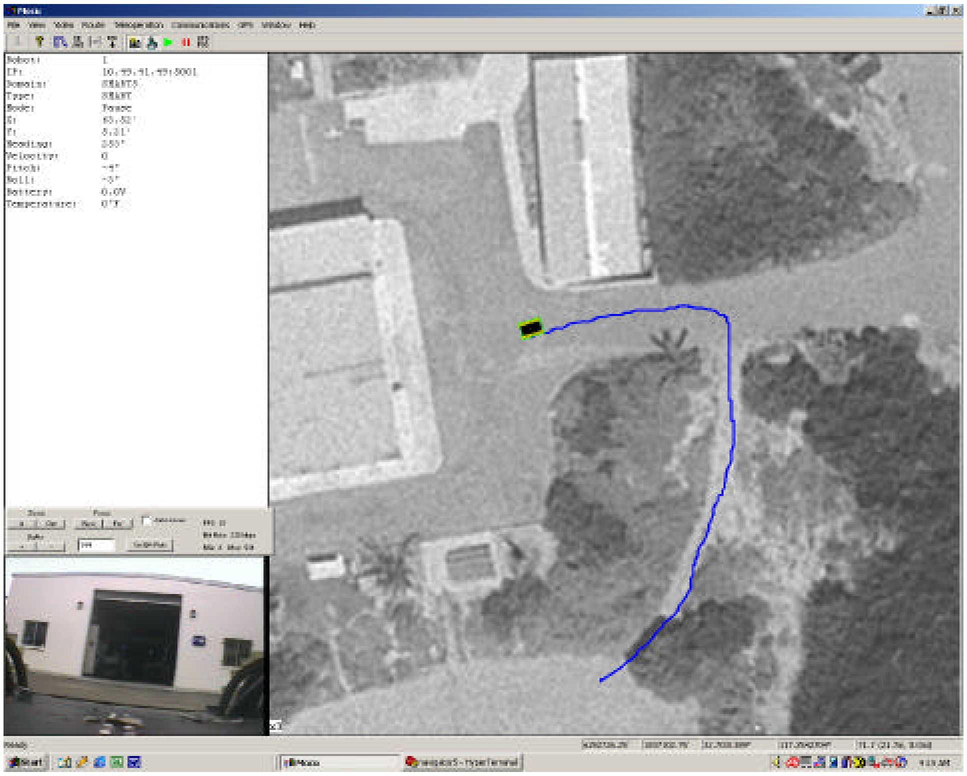


Figure 6. Simulation of path following algorithm

5. Operator Control Unit

The original wearable OCU[5] for the URBOT only supported pure teleoperation and was fairly limited in its capabilities. To support waypoint navigation, a new OCU was needed that would meet the following minimum requirements: (1) Display an aerial photograph of the area of interest, (2) enable the user to define and download waypoints to the robot, (3) allow the user to correct the location of the robot, (4) display video from the robot, (5) allow the robot to be teleoperated with a joystick, and (6) allow the operator to control multiple robots from a single control unit. The resulting control software was dubbed the Multi-robot Operator Control Unit, or MOCU (see Figure 7).



Robot: 1
IP: 10.99.41.99:5001
Domain: SMARTS
Type: SMART
Mode: Pause
X: 65.52'
Y: 5.21'
Heading: 253°
Velocity: 0
Pitch: -5°
Roll: -3°
Battery: 0.0V
Temperature: 0°F

Figure 7. Screen shot of a path created with MOCU.

The aerial photograph is displayed in the large pane on the right. This image can actually consist of several separate images – MOCU automatically tiles the images together using information from the associated GEOTIFF files. The waypoints (or paths) are drawn directly on this ortho-rectified image, which allows the user to specify the path with a precision up to the resolution of the image data. The text window in the upper left pane displays the current robot status. The real-time video from the robot is displayed in the lower left corner. If desired, the video and map windows can be swapped if a larger video image is desired (for example, when the vehicle is being teleoperated). The current geodetic coordinate of the mouse cursor on the map can be read in a variety of formats in the status bar at the bottom of the screen.

6. Summary

This project has demonstrated the ability to provide an effective waypoint navigation capability to small low-cost systems such as the MPRS URBOT. The entire navigation package easily fits into a small shoebox-size enclosure and could easily be ported to other robotic vehicles.

It was also demonstrated that by separating the GPS error sources into two categories it is possible to attain accurate navigation results without the aid of differential GPS corrections. The test results show that this system is able to navigate almost as accurately as a vehicle using a real-time kinematic (RTK) GPS solution as long as there are an adequate number of landmarks available for referencing. The ability to use a non-differential GPS receiver extends the possible application of waypoint navigation into the tactical realm.

It has also been shown that these navigation techniques do not require a great deal of computational power. The Kalman Filter and path following algorithms run on an embedded 66MHz PowerPC running a non-real-time operating system (POSIX based pKernel).

This project was funded by the Unmanned Ground Vehicle/Systems Joint Program Office (UGV/S JPO).

Appendix A

Kalman Filter equations:

```
Xhatminus = PHI*Xhatplus;     % next predicted state by dead reckoning

Pminus = PHI*Pplus*PHI' + Q;  % predicted covariance

K = Pminus*H'/(H*Pminus*H'+R);               % Kalman gain

Pplus = Pminus - K*H*Pminus;                 % corrected covariance

Xhatplus = Xhatminus + K*(Z- H*Xhatminus);   % corrected state
                                               vector

X = Xhatplus;              % set X to the new corrected state estimate
```

References

1. Bruch, M.H., Laird, R.T., and H.R. Everett, "Challenges for Deploying Man-Portable Robots into Hostile Environments", SPIE Proc. 4195: Mobile Robots XV, Boston, MA, November 5-8, 2000.

2. A. Kelly, "A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles", Techinical Report, CMU-RI-TR-94-19, Robotics Institute, Carnegie Mellon University, May, 1994.

3. M. H. Grewal, L. R. Weill, A. P. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*, John Wiley & Sons, New York, 2001.

4. M. H. Hebert, C. Thorpe, A. Stentz, A. Kelly, *Intelligent Unmanned Ground Vehicles, Autonoums Navigation Research at Carnegie Mellon*, Kluwer Academic Publishers, Massachusetts, 1997.

5. Laird, R.T., Bruch, M.H., "Issues in Vehicle Teleoperation for Tunnel and Sewer Reconnaissance," IEEE Conference on Robotics and Automation, San Francisco, CA, April 2000.