Integrating an Electronic Compass for Position Tracking on a Wheeled Tricycle Mobile Robot

Praneel Chand[1]

[1]*Waikato Institute of Technology, Rotokauri Campus, Hamilton, New Zealand*

E-mail: praneel.chand@wintec.ac.nz

Abstract

Dead-reckoning via encoders on wheeled-mobile robots is a simple but inaccurate method to estimate position. The major drawback of encoders is wheel slippage errors that accumulate over time. This problem is often addressed by using additional sensors such as compass, gyroscope, or GPS. This paper details the integration and effectiveness of a relatively low-cost solution using an electronic compass to reduce positioning error on a wheeled tricycle mobile robot. A customised Visual Studio program has been developed to adjust the settings of the electronic compass and integrate it with the Visual Studio based robot control system. The electronic compass heading data is fused with the encoder odometry heading data in three different ways: simple fusion, linear weighted fusion, and Kalman filter fusion. Simple fusion and linear weighted fusion rely on parameters determined from angular acceleration and angular velocity, respectively. The Kalman filter uses variance data for the encoders and electronic compass to determine an optimal heading. Experiments have been conducted in an indoor corridor environment to evaluate and compare the various fusion methods. Position error is successfully reduced and is sufficient to locate the robot within the corridor.

Keywords:

Mobile robots; Position estimation; Sensor fusion; Electronic compass; Indoor navigation;

1. Introduction

Point-to-point or waypoint navigation in mobile robots commonly relies on a robot's position. It involves utilising the current position of the robot and moving to a target location based on a navigation strategy or algorithm. The navigation strategy could be based purely on reactive control (Chand 2015), (Pandey 2017), (Baklouti et al. 2017), (Boujelben et al. 2017) or incorporate deliberative control (global path planner based) (Chand and Carnegie 2011), (Patle et al. 2019), (Zhang 2019). Global path planners are well-suited for navigation in known environments.

However, in dynamic environments where obstacle positions are unknown or can change, reactive control is incorporated to handle uncertainties. The reactive control often includes some form of obstacle avoidance for moving around obstacles to reach the target location.

A key component of wheeled mobile robot navigation in the real world is position estimation. While moving from an initial position to a target location, a robot needs to know where it is to adjust actuator commands for movement. One of the simplest and cheapest ways to estimate position is through dead-reckoning via encoders (Lee-Johnson et al. 2007). Encoders are quick at estimating position and can function satisfactorily over short distances (up to approximately 10 metres). However, wheel slippage errors accumulate over time and eventually dead-reckoning becomes unreliable for position estimation on its own. Other sensors such as compasses, inertial measurement units (IMUs), global positioning system (GPS), beacons, or cameras need to be incorporated for improving position estimation.

Sensor fusion is the process of combining data from multiple sensors into meaningful information. In robot navigation, algorithms are used to combine data from complementary and/or redundant sensors for position estimation. A variety of sensors and a range of algorithms such as Kalman filtering, rule-based techniques, Bayesian theory, fuzzy logic, or neural networks can be used (Kam et al. 1997), (Fung et al. 2017), (Mohamed et al. 2018).

Chen and Zhang (2017) presented a novel method for indoor mobile robot navigation that fuses odometry and electronic compass data. Two calibration methods are utilised to calibrate odometry and compass errors, respectively. A complex adaptive neuro-fuzzy inference system (ANFIS) implemented in MATLAB is used to calibrate the compass. Following calibration, a fusion algorithm based on an adaptive extended Kalman filter (AEKF) combines the odometry and compass data. The fusion algorithm is evaluated via experiments which compare odometry only navigation with odometry and compass navigation. Error is reduced by a factor of approximately ten.

An omnidirectional mobile robot for intelligent manufacturing that fuses data from four wheel encoders and a Kinect visual sensor is described in (Qian et al. 2017). The extended Kalman filter (EKF) algorithm is used recursively to integrate the data from the wheel encoders and the Kinect sensor. The frequency of position estimation is based on the frame rate and processing of the visual sensor (30 Hz). A complex process is used to compare colour and depth (RGB-D) images for measuring three-dimensional position and posture increments. Indoor production line rectangular path experiments show that the average deviation of distance is 207 mm.

Alatise and Hancke (2017) also used vision data to supplement mobile robot pose estimation. Data from a six degrees of freedom (6-DoF) inertial sensor which consists of a 3-axis accelerometer and a 3-axis gyroscope is fused with monocular vision-based object detection algorithm data using the EKF algorithm. Speeded-up robust feature (SURF) and random sample consensus (RANSAC) algorithms recognise objects from images. The IMU-based pose estimation itself includes fusion of the accelerometer and gyroscope data using the Kalman filter. This in turn reduces drifts and errors. The experimental setup comprised a small four-wheel drive (4WD) mobile robot with an Arduino 101 microcontroller. Camera images and IMU data are sent to a PC via USB cable. Consequently, image processing and pose estimation is performed offline using MATLAB. This is a drawback since the robot remains tethered to a PC. The RMS error of pose estimation is within 0.14 m.

Khatib et al. (2020) presented a low-cost method for wheeled robot navigation in indoor and outdoor environments. Probabilistic approaches are used to combine sensor data from low-cost visual/inertial sensors. A Microsoft Kinect stream depth sensor extracts landmarks for indoor navigation. This information is fused with wheel odometry data via the EKF algorithm. A reduced inertial sensor system and GPS are employed for outdoor navigation. The absolute error for trajectory tracking is within 0.3 m accuracy based on experiments in indoor and outdoor environments.

A pair of tricycle robots has previously been designed and constructed for cooperative tasks (Carnegie et al. 2005). The robots have been constructed at a relatively low-cost of approximately US$2000. These robots have a variety of sensors such as wheel encoders, infrared rangefinders, and GPS. They also employ a hybrid navigation system with reactive and deliberative control for traversing an environment. Experiments in an outdoor rectangular sports field environment showed that the robots can use GPS data to form a closed loop. However, GPS is not suitable for indoor navigation. Hence, this paper investigates the relatively low-cost option of adding electronic compass modules to the robots for indoor navigation. Low-cost in this research implies not relying on cameras and their associated expensive complex software (such as MATLAB for image processing), memory use, and processing requirements.

2. Method

2.1. Overview of Robot Hardware

One of the tricycle robots is shown in Figure 1. The front two wheels drive the robot via an electric motor and a mechanical differential. A single steering wheel at the rear controls rotational movement. There are shaft encoders

on the left and right wheels for position estimation. A GPS sensor is mounted for outdoor navigation. The robot is powered by a pair of 12 V deep cycle batteries and an onboard computer runs the robot's control system. The control system is explained further in section 2.4.

Figure 1. Tricycle robot hardware.

2.2. Electronic Compass Selection

The robot's onboard computer is a small form factor desktop PC that can support multiple devices via USB connection. USB port to standard DB9 serial port converters are readily available for interfacing RS-232 based devices to the computer. Three-axis electronic compass modules are widely available and preferred since the robot may have to navigate uneven or inclined terrain. Some commercially available electronic compass modules have been reviewed for implementation on the robots. These included the KVH C100 Compass Engine (KVH 2019), Devantech CMPS14 compass module (Robot-Gear 2009), and Honeywell HMR3300 compass module (Digi-Key 2021).

The KVH C100 compass engine is a standalone sensor that is suitable for industrial applications. It has ±0.5º RMS accuracy when level, can be interfaced via RS-232 serial port, but has a high cost of approximately US$800. On the other hand, the Devantech CMPS14 module is much cheaper (approximately US$45) with less functionality and is better suited to hobby use. The Honeywell HMR3300 module has features such as tilt compensation, onboard filtering, various operational commands, 0.5º repeatability, RS-232 interfacing, and costs approximately US$620. Hence, the HMR3300 compass module has been selected for implementation on the robots.

2.3. Compass Interface Design and Integration

To allow connection to the robot's computer through a USB to RS-232 converter, the serial UART interfacing protocol has been used (Cawley 2006). The designed interface board incorporates a MAX232 serial transceiver to convert the logic level serial data to higher voltage for transmission over the serial line. A schematic diagram of the board is shown in Figure 2. It includes an onboard regulated 5 V power supply that gets input from the robot's 12 V batteries.

Figure 2. Schematic diagram of compass interface board.

The compass module and associated electronics is housed securely within a small blue ABS plastic enclosure as shown in Figure 3(a). The enclosure is empirically mounted on a non-metallic (wooden) pole at a height of 0.6 m above the other electronics and motors to minimise electromagnetic interference (Figure 3(b)).

Figure 3. Compass module. (a) Enclosure; (b) Placement on robot

The software to test functionality of the compass module has been developed using Visual C++. This ensures compatibility with the existing robotic control system that is also programmed using Visual C++. A screenshot of the program developed for calibration and data logging is shown in Figure 4. The first section on the top left enables compass data acquisition and logging to a text file. Next to that section, the user can poll the compass module for its current configuration. Adjacent to this, there are controls for resetting the compass and exiting the program. The second and third rows of the program window permit adjustments such as compass calibration, declination angle setting, baud rate setting, system and magnetic filter setting, and z-axis offset. Further details of the operational commands to communicate with the compass module are available in (Digi-Key 2021).

Figure 4. Compass module demo and setup software.

2.4. Robot Control System Overview

The hybrid reactive deliberative control system is illustrated in Figure 5. It consists of several modules for sensing (information extraction and sensor fusion), modelling (localisation, environment map), planning (path planner) and actuation (reactive control, low level motion control/execution). The hierarchy of the modules also gives an indication of the temporal decomposition of control ranging from real-time to on-demand. The navigation system (Chand and Carnegie 2011) relies on the current position to determine wheel speeds for navigating to a target location. It is based on a combination of the Vector Field Histogram (Ulrich and Borenstein 1998) and Dynamic Window (Fox et al. 1997) methods. The path planner for deliberative control is based on a modified version of the A* path planning algorithm for grid maps (Pearl 1984), (Chand and Carnegie 2011).

Figure 5. Overview of robot control system.

The control system is implemented as a Visual C++ graphical user interface (GUI) program with buttons and tabs for various features (Figure 6). The left side of the GUI has the buttons for switching between manual and

autonomous control, as well as the option for data logging. On the right side, there are several tabs, including the important hardware and control tabs. In the hardware tab, the status of the various sensors and communication ports are displayed. The control tab enables position to be set or reset, and also facilitates the selection of sensor fusion algorithms for position estimation.

Figure 6. Robot control system program. (a) Main controls and hardware tab; (b) Control tab

2.5. Position Estimation

This paper assumes that the mobile robot will travel in planar motion encountered in an indoor building or flat outdoor terrain. In environments with uneven or elevated terrain, GPS data could also be utilised for position estimation and a modified approach similar to Khatib et al. (2020) could be used. The tricycle drive robot has encoders on the front left and front right wheels. Thus, the position (*x, y*) is estimated with respect to the mid-point between the front wheels (Figure 7). This can be easily translated to the centre of the robot via an offset distance y_{off} in the robot's reference frame. The estimation process of the robot's posture using the odometers is described by equations (1)-(5) below:

$$d\theta_e(k) = \frac{1}{w}(ds_l(k) - ds_r(k)) \quad (1)$$

$$ds(k) = \frac{1}{2}(ds_l(k) + ds_r(k)) \quad (2)$$

$$\theta_e(k) = \theta_e(k-1) + d\theta_e(k) \quad (3)$$

$$x(k) = x(k-1) + ds(k)sin\left(\frac{\theta_e(k) + \theta_e(k-1)}{2}\right) \quad (4)$$

$$y(k) = y(k-1) + ds(k)cos\left(\frac{\theta_e(k) + \theta_e(k-1)}{2}\right) \quad (5)$$

Where: $x(k)$, $y(k)$: robot position estimated from encoders at the kth instant

$\theta_e(k)$: robot heading estimated from encoders at the kth instant

w : distance between the two drive wheels

$d\theta_e(k)$, $ds(k)$: variations of the robot's heading angle and displacement between the kth and $(k-1)$th sample instant

$ds_l(k)$, $ds_r(k)$: displacement of the left and right wheels between the kth and $(k-1)$th sample instant

Figure 7. Reference point for position estimation of the tricycle robot.

Heading values from the compass module are denoted as θ_c. The in-built filter on the compass module is used to reduce heading noise. Three methods for combining compass heading θ_c and encoder heading θ_e into a fused heading θ_f have been implemented: simple fusion (threshold-based), linear weighted fusion, and Kalman filter (Welch and Bishop 2001) fusion. The fused heading θ_f replaces encoder heading θ_e in equations (4) and (5), and the robot's position (x_f, y_f) is calculated as:

$$x_f(k) = x_f(k-1) + ds(k)sin\left(\frac{\theta_f(k) + \theta_f(k-1)}{2}\right) \quad (6)$$

$$y_f(k) = y_f(k-1) + ds(k)cos\left(\frac{\theta_f(k) + \theta_f(k-1)}{2}\right) \quad (7)$$

2.5.1. *Simple Fusion*

Simple fusion uses a threshold based on angular acceleration (8). If the absolute angular acceleration α is below a threshold α_t then the fused heading θ_f is determined by the compass module θ_c. Otherwise, the fused heading is determined based on encoder readings θ_e. The rationale for this is that the in-built compass filter introduces delays in estimating heading while the robot is turning.

$$\theta_f = \begin{cases} \theta_c, & |\alpha| < \alpha_t \\ \theta_e, & |\alpha| \geq \alpha_t \end{cases} \quad (8)$$

2.5.2. *Linear Weighted Fusion*

Linear weighted fusion uses weightings to combine encoder heading θ_e and compass heading θ_c into the fused heading θ_f (9). The encoder heading weighting is denoted as k_e and compass heading weighting is denoted as k_c. The weightings vary as a function of the robot's angular velocity ω and are unit interval values (10),(11). This function could be a linear or non-linear function. Like simple fusion, the rationale for the variable weights is based on the motion of the robot. Sharp and sudden turns cause delays in compass heading change.

$$\theta_f = k_e.\theta_e + k_c.\theta_c \quad (9)$$

$$k_e = f(\omega) \quad (10)$$

$$k_c = 1 - k_e \quad (11)$$

2.5.3. *Kalman Filter Fusion*

The Kalman filter (Welch and Bishop 2001) is an optimal filter that employs a set of equations to implement a predictor-corrector type estimator. It achieves optimality by minimising the estimated error covariance. The ongoing Kalman filter cycle is illustrated in Figure 8. First, the time update part of the cycle predicts the state and error covariances for the next state. Following this, the measurement update part of the cycle applies corrections to the prediction by computing the Kalman gain, updating the estimate with a measurement, and updating the error covariance.

Figure 8. Kalman filter cycle.

The process model for heading is governed by the difference equation presented in (3). Essentially, the next heading estimate is the current estimate plus the change in heading determined from the left and right wheel displacements (encoders). A process variance Q is assumed for error covariance prediction. Hence, the time update equations for the next state and error covariance are denoted by (12) and (13), respectively.

$$\hat{\theta}_k^- = \hat{\theta}_{k-1} + ds(k) \quad (12)$$

$$P_k^- = P_{k-1} + Q \quad (13)$$

In the measurement update phase, the Kalman gain is computed (14) and the compass heading z_k is employed to update the estimate (15). Hence, $\hat{\theta}_k$ is the equivalent fused heading θ_f. The error covariance is also updated using the Kalman gain (16).

$$K_k = P_k^- (P_k^- + R)^{-1} \quad (14)$$

$$\hat{\theta}_k = \hat{\theta}_k^- + K_k(z_k - \hat{\theta}_k^-) \quad (15)$$

$$P_k = (1 - K_k)P_k^- \quad (16)$$

3. Results and Discussion

3.1. Experiment Configurations

An indoor corridor environment has been used to evaluate the various position estimation methods. In the experiments, the robot moves around the long section of the corridor similar to a patrol robot. It is remotely controlled (manually) via a joystick attached to a laptop computer. Markers on the corridor floor provide waypoints for navigation and ground truth data. Figure 9 illustrates the corridor environment with the waypoints marked as magenta circles. A single data set with the different fusion methods applied in post-processing is utilised to achieve the most directly comparable results by eliminating manual control variations.

Figure 9. Indoor corridor environment with waypoints.

The threshold value for simple fusion has been selected based on the dynamic characteristics of the robot. Multiple tests of the robot in straight and circular paths within the dimensions of the corridor were used to estimate typical values of angular acceleration. Based on the collected data, the threshold for simple fusion is empirically set to 0.37 rad/s^2.

Three types of encoder heading weighting functions have been selected for linear weighted fusion. These weighting functions are: linear (17), non-linear piecewise (18), and fuzzy rule-based (Table 1). The membership functions for the input and output of the fuzzy approach are shown in Figure 10. Each of the three encoder weighting functions is illustrated in Figure 11. The main purpose of the functions is to give preference to the compass heading when the robot has low angular velocity. Weights and threshold values for each function have been empirically selected based on manual operation of the robot in the corridor and labs.

$$k_e = 0.5|\omega| \tag{17}$$

$$k_e = \begin{cases} 1, & |\omega| \geq 1.4 \\ \dfrac{0.7073|\omega|^3 - 13.99|\omega|^2 + 35.71|\omega| - 8.482}{16}, & |\omega| > 0.5 \; and \; |\omega| < 1.4 \\ \dfrac{3.958e^{0.2608|\omega|} + 0.002262e^{12.98|\omega|}}{16}, & |\omega| \leq 0.5 \end{cases} \tag{18}$$

Table 1. Fuzzy weighting function rules.

Figure 10. Membership functions of the fuzzy weighting function. (a) Input; (b) Output

Figure 11. Three encoder heading weighting functions for linear weighted fusion. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

Kalman filter fusion requires values for process variance Q and measurement variance R. The encoder provides good accuracy over short distances (<= 5 m). From straight line tests with a maximum speed of 0.3 m/s to reduce wheel slippage, the standard deviation was determined to be approximately 1.5º. Thus, the process variance Q is approximately 2.25º. On the other hand, the compass provides reliable readings over long distances. Its standard deviation is approximately 3º. Hence, the measurement variance R is approximately 9º.

3.2. Position Estimation Results

3.2.1. Encoder Only

Relying solely on the encoders for position estimation produces large errors as shown in Figure 12. The estimated position drifts beyond the corridor walls and is clearly insufficient for robot navigation.

Figure 12. Position estimation using encoders only.

3.2.2. Simple Fusion

Figure 13 shows the position estimation results when simple fusion is used. The estimated position remains within the boundaries of the corridor. However, uncertainties in the encoders and compass cause deviations from the waypoints. Figure 14 illustrates the difference between the compass measurement and fused heading. As expected, variations occur when the angular acceleration is large enough to cause the fused heading to be equal to the encoder heading.

Figure 13. Position estimation using simple fusion.

Figure 14. Difference between compass measurement and simple fusion heading.

3.2.3. Linear Weighted Fusion

The results for linear weighted fusion are shown in Figure 15 and Figure 16. Similar to simple fusion, the position estimates remain within the boundaries of the corridor. The trajectory of the robot's estimated travel is similar for all three types of encoder heading weighting functions. However, the difference between compass measurement and fused heading is smaller when the linear encoder heading weighting function is used. This doesn't necessarily indicate superior results since the in-built compass filter introduces delays in measuring heading while the robot

is turning. The other two encoder weighting functions (non-linear piecewise and fuzzy) can provide increased bias to the encoder heading while the robot is turning.

Figure 15. Position estimation using linear weighted fusion with encoder heading weighting functions. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

Figure 16. Difference between compass measurement and linear weighted fusion heading. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

3.2.4. *Kalman Filter Fusion*

Using the Kalman filter for encoder and compass heading fusion produces the estimated position shown in Figure 17. This result is similar to the other methods and the estimated position remains within the boundaries of the corridor. The resulting variation between the compass heading and fused heading is shown in Figure 18. The difference between compass measurement and fused heading is larger than all three types of linear weighted fusion. This indicates that a higher weighting is provided to the encoder heading at low angular speeds as well as at higher angular speeds. This is likely due to the process variance (encoder variance) being smaller than the measurement variance (compass variance) over short distances.

Figure 17. Position estimation using the Kalman filter

Figure 18. Difference between compass measurement and Kalman filter fusion heading

3.2.5. *Overall Comparison*

For an overall comparison, the estimated positions produced by each fusion method have been plotted on a single graph and magnified to view the similarity at various sections of the corridor. Figure 19 highlights the estimated position in the upper section of the corridor. The estimated position in the middle section of the corridor is compared in Figure 20. Figure 21 illustrates the estimated positions towards the end of path. All three figures show the red and magenta lines corresponding to simple fusion and Kalman filter fusion tracking closest to the waypoints. Table 2 details the maximum error, mean error, and standard deviation from all waypoints. The lowest errors have been achieved for simple fusion and the Kalman filter.

Table 2. Error analysis from all waypoints

F

Figure 19. Upper section of corridor comparison

Figure 20. Middle section of corridor comparison

Figure 21. End of path comparison. (a) Second last waypoint; (b) Last waypoint

3.2.6. *Discussion*

The position estimation results are sufficient for indoor corridor navigation in applications such as patrol/surveillance robots. The system can be improved by using sensors such as beacons to apply further corrections to positions over long periods of navigation. For example, there could be a beacon located at one end of the corridor that updates position data and resets the overall error after the robot loops around the corridor. Due to the robot being a custom-made in-house device, there are deviations in straight line travel due to steering wheel sensor and actuator noise. Occasionally, there are delays in communicating commands from the laptop base station to the robot. This can also produce fluctuations in the robot's path.

Simple fusion and linear weighted fusion rely on empirical measurements of angular acceleration and angular velocity to adjust the preference for using the encoder heading for position estimation. On the other hand, the Kalman filter relies on empirical measurements of the encoder heading estimate variance and the compass measurement variance. While the results are adequate for indoor corridor navigation using the conventional single model Kalman filter, an interacting multiple model (IMM) estimator could be applied to filter the entire trajectory rather than just heading. This would require the robot to employ constant turn and constant velocity motion. In the current implementation and in a dynamic environment this may not always be feasible.

Compared with other position estimation methods reviewed in section 1, the developed system produces slightly greater errors for indoor navigation without the use of a vision sensor. Alatise and Hancke (2017) used vision data to supplement mobile robot pose estimation and their RMS error for pose estimation is within 0.14 m. The low-cost method for wheeled robot navigation developed by Khatib et al. (2020) produces an absolute error for trajectory tracking within 0.3 m. Qian et al. (2017) also use a visual sensor to achieve an average deviation of 0.207 m from rectangular indoor production line paths.

The implementation and results presented in this paper do not assume the presence of magnetic objects or obstacles. However, there may have been magnetic objects present in the building. Any effects have been

minimised by calibrating the compass before experimentation. The various parameters and thresholds employed in sensor fusion have been tuned for the robot used in the experiments. While the robot has a fixed operating speed range, changing the operating speed could influence thresholds and parameters. This could require re-calibration or re-turning via traversing standard paths.

4. Conclusion

Position estimation is an important part of point-to-point navigation for mobile robots. This paper has presented a relatively low-cost solution for using an electronic compass to reduce positioning error on a wheeled tricycle mobile robot. The low-cost solution does not rely on cameras and associated expensive complex software such as MATLAB for image processing. Cost is also reduced by developing an in-house interface board and customised Visual Studio software for the electronic compass.

The electronic compass heading data has been fused with the encoder odometry heading data in three different ways: simple fusion, linear weighted fusion, and Kalman filter fusion. Simple fusion relies on a single threshold parameter determined from angular acceleration. For linear fusion, three encoder heading weighting functions based on angular velocity have been evaluated: linear, non-linear piecewise, and fuzzy. The primary purpose of the encoder heading weighting function is to give higher preference to the electronic compass heading when the robot is not turning. Kalman filter fusion uses variance data for the encoders and electronic compass to determine an optimal heading. Experiments have been conducted in an indoor corridor environment to evaluate and compare the various fusion methods. Position error has been successfully reduced and is sufficient to locate the robot within the corridor.

Future improvements to the system could include the addition of exteroceptive sensors such as beacons to apply corrections over long periods of navigation. One or two beacons at the extreme boundaries of the corridor could be used to update position information and the reset the overall position error when the robot loops around the corridor. Incorporating a single axis gyroscope could provide a direct measurement of angular velocity to improve performance. A more expensive compass or a custom filter could also be used to improve performance by reducing lag. Additional future work can include the investigation of performance in static and dynamic obstacle environments.

Acknowledgements

The author would like to thank the University of Waikato and Victoria University of Wellington for laboratory and equipment access to conduct the research.

Competing Interests

The author declares there are no competing interests.

Funding

This research was supported by University of Waikato internal research funding.

References

ALATISE, M. B. and HANCKE, G. P. 2017. Pose Estimation of a Mobile Robot Based on Fusion of IMU Data and Vision Data Using an Extended Kalman Filter. *Sensors,* 17**,** 2164.

BAKLOUTI, E., AMOR, N. B., and JALLOULI, M. 2017. Reactive control architecture for mobile robot autonomous navigation. *Robotics and Autonomous Systems,* 89**,** 9-14.

BOUJELBEN, M., REKIK, C., and DERBEL, N. 2017. A reactive approach for mobile robot navigation in static and dynamic environment using fuzzy logic control. *International Journal of Modelling, Identification and Control,* 27**,** 293-302.

CARNEGIE, D., PAYNE, A., and CHAND, P. 2005. The Design of a Pair of Identical Mobile Robots to Investigate Cooperative Behaviours. *In:* V. KORDIC, A. L. A. M. M. (ed.) *Cutting Edge Robotics.* Croatia: IntechOpen.

CAWLEY, C. 2006. *The Enhancement of a Multi-Terrain Mechatron for Autonomous Outdoor Applications.* Master of Science MSc Thesis, University of Waikato.

CHAND, P. Fuzzy reactive control for wheeled mobile robots. 2015 6th International Conference on Automation, Robotics and Applications (ICARA), 17-19 Feb. 2015 2015 New Zealand. IEEE, 167-172.

CHAND, P., and CARNEGIE, D. 2011. Development of a navigation system for heterogeneous mobile robots. *International Journal of Intelligent Systems Technologies and Applications,* 10**,** 250-278.

CHEN, W., and ZHANG, T. 2017. An indoor mobile robot navigation technique using odometry and electronic compass. *International Journal of Advanced Robotic Systems,* 14**,** 1729881417711643.

DIGI-KEY. 2021. *HMR3300* [Online]. New Zealand: Digi-Key. Available: https://www.digikey.co.nz/en/products/detail/honeywell-aerospace/HMR3300/396917 [Accessed 20 September 2021].

FOX, D., BURGARD, W., and THRUN, S. 1997. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine,* 4**,** 23-33.

FUNG, M. L., CHEN, M. Z. Q., and CHEN, Y. H. Sensor fusion: A review of methods and applications. 2017 29th Chinese Control And Decision Conference (CCDC), 28-30 May 2017 2017. 3853-3860.

KAM, M., XIAOXUN, Z., and KALATA, P. 1997. Sensor fusion for mobile robot navigation. *Proceedings of the IEEE,* 85**,** 108-119.

KHATIB, E. I. A., JARADAT, M. A. K., and ABDEL-HAFEZ, M. F. 2020. Low-Cost Reduced Navigation System for Mobile Robot in Indoor/Outdoor Environments. *IEEE Access,* 8**,** 25014-25026.

KVH. 2019. *C100 Compass Engine - Innovative Stand-alone Heading Sensor* [Online]. USA. Available: https://kpp-public.s3.amazonaws.com/DATASHEET+-+C100+Compass+Engine [Accessed 20 September 2021].

LEE-JOHNSON, C. P., CHAND, P., and CARNEGIE, D. A. Applications of an Adaptive Hierarchical Mobile Robot Navigation System. Australasian Conference on Robotics and Automation, 10-12 December 2007 Brisbane, Australia. ARAA.

MOHAMED, A., REN, J., EL-GINDY, M., LANG, H., and OUDA, A. 2018. Literature survey for autonomous vehicles: sensor fusion, computer vision, system identification and fault tolerance. *International Journal of Automation and Control,* 12**,** 555-581.

PANDEY, A. 2017. Mobile robot navigation and obstacle avoidance techniques: A review. *International Robotics & Automation Journal,* 2**,** 96-105.

PATLE, B. K., BABU L, G., PANDEY, A., PARHI, D. R. K., and JAGADEESH, A. 2019. A review: On path planning strategies for navigation of mobile robot. *Defence Technology,* 15**,** 582-606.

PEARL, J. 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving,* Reading, MA, Addison-Wesley.

QIAN, J., ZI, B., WANG, D., MA, Y., and ZHANG, D. 2017. The Design and Development of an Omni-Directional Mobile Robot Oriented to an Intelligent Manufacturing System. *Sensors,* 17**,** 2073.

ROBOT-GEAR. 2009. *Devantech CMPS14 - Tilt Compensated Compass Module* [Online]. Australia: Robot Gear. Available: https://www.robotgear.com.au/Product.aspx/Details/6782-Devantech-CMPS14-Tilt-Compensated-Compass-Module [Accessed 20 September 2021].

ULRICH, I., and BORENSTEIN, J. VFH+: reliable obstacle avoidance for fast mobile robots. Proceedings of the 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146), 20-20 May 1998 1998. IEEE, 1572-1577 vol.2.

WELCH, G., and BISHOP, G. 2001. *SIGGRAPH Course 8 - An Introduction to the Kalman Filter,* California, USA, ACM.

ZHANG, J. A 2019. Hybrid Reactive Navigation Strategy for a Non-holonomic Mobile Robot in Cluttered Environments. 2019 Chinese Control Conference (CCC), 27-30 July 2019 China. IEEE, 3839-3844.

Table 1. Fuzzy weighting function rules.

Rule	**Input $\|\omega\|$**	**Output k_e**	**Rule Weight**
1	Low	Low	1
2	High	High	1

Table 2. Error analysis from all waypoints

Fusion Method	Simple fusion	Linear fusion with linear encoder weighting function	Linear fusion with non-linear piecewise encoder weighting function	Linear fusion with fuzzy encoder weighting function	Kalman filter
Maximum error (m)	0.443	0.479	0.473	0.473	0.452
Mean error (m)	0.299	0.327	0.313	0.313	0.302
Standard Deviation (m)	0.145	0.117	0.133	0.132	0.143

Figure Captions

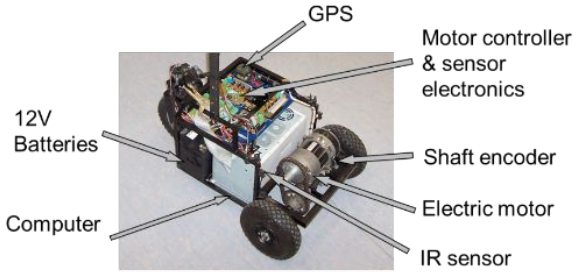


Figure 1. Tricycle robot hardware.

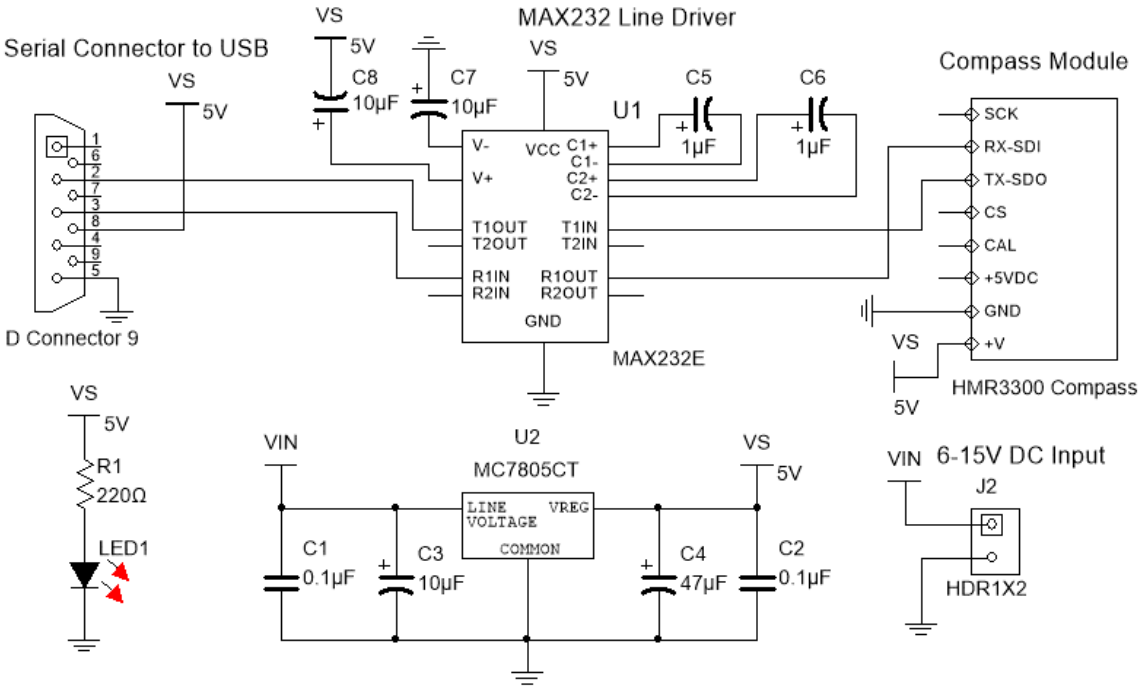
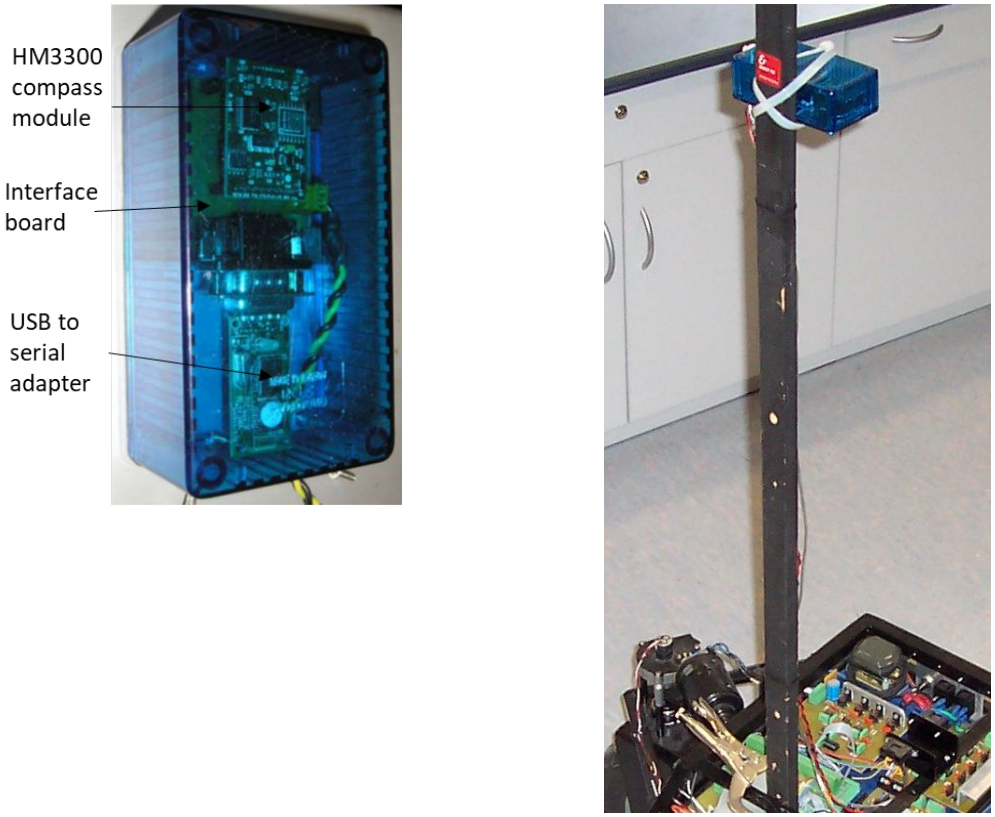


Figure 2. Schematic diagram of compass interface board.



(a) (b)

Figure 3. Compass module. (a) Enclosure; (b) Placement on robot

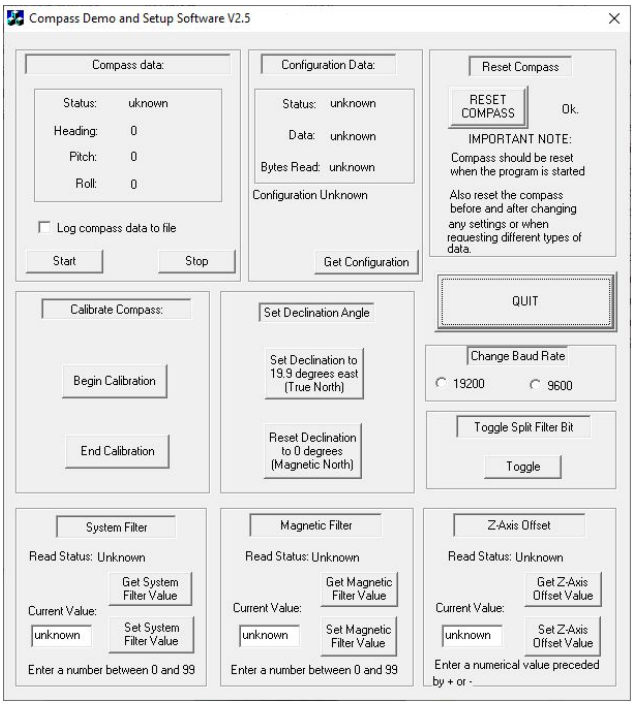


Figure 4. Compass module demo and setup software.

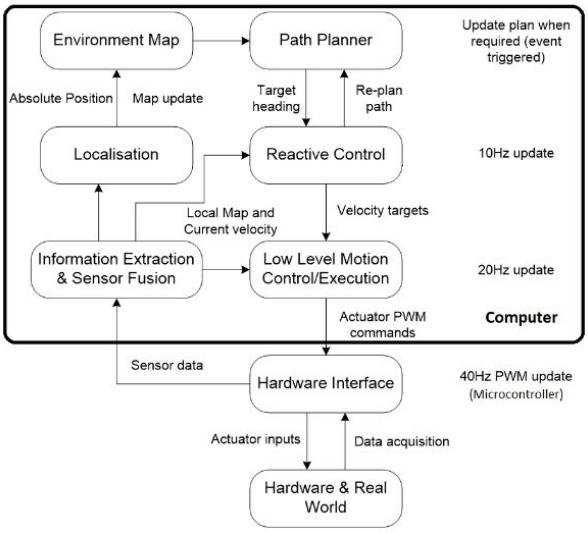
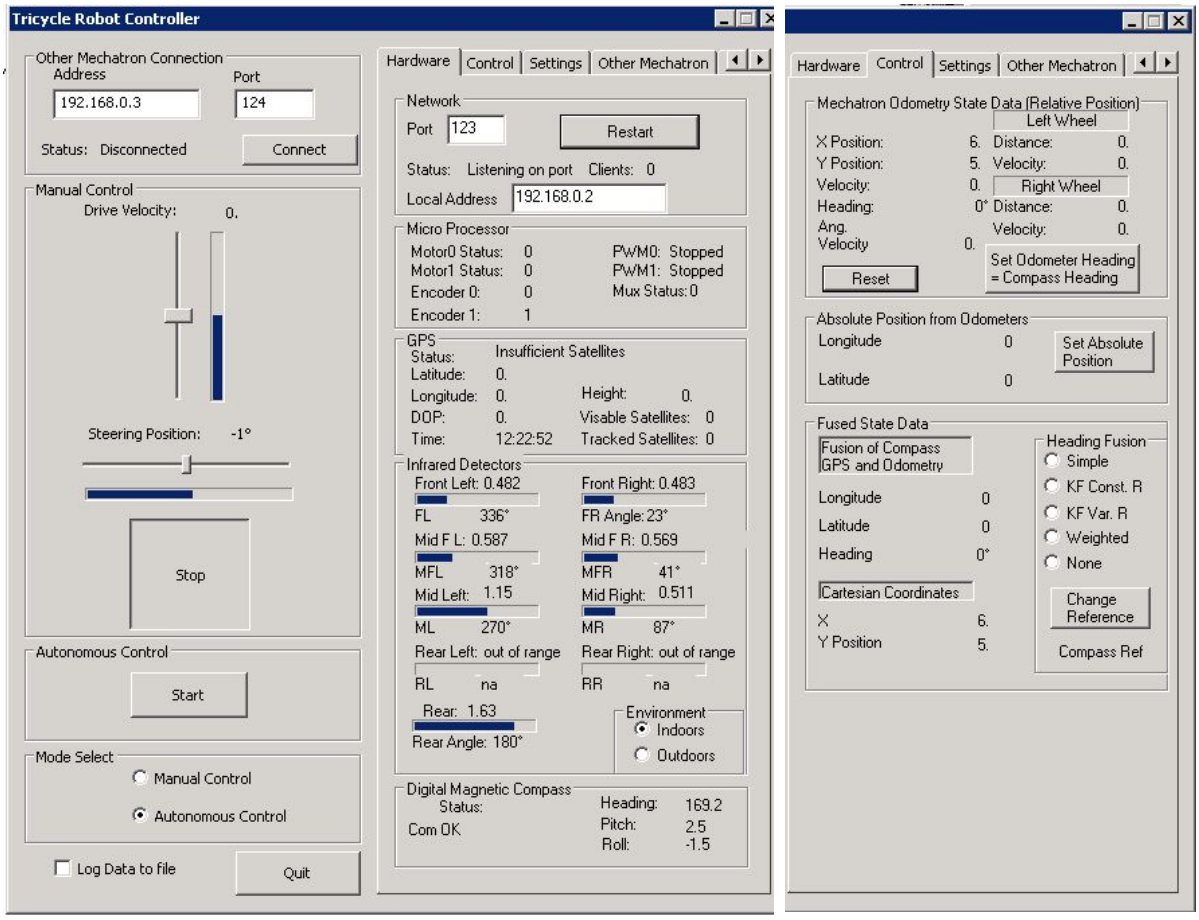


Figure 5. Overview of robot control system.



(a) (b)

Figure 6. Robot control system program. (a) Main controls and hardware tab; (b) Control tab

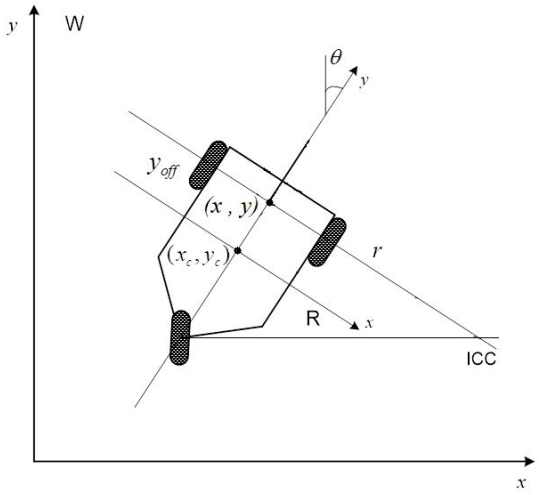


Figure 7. Reference point for position estimation of the tricycle robot.

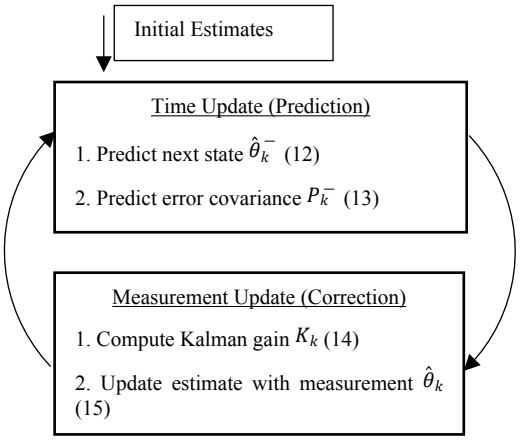


Figure 8. Kalman filter cycle.

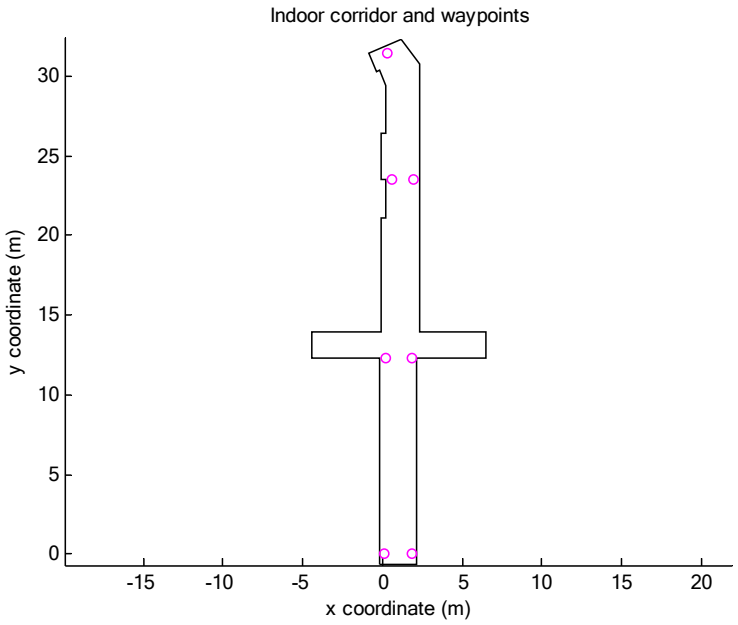
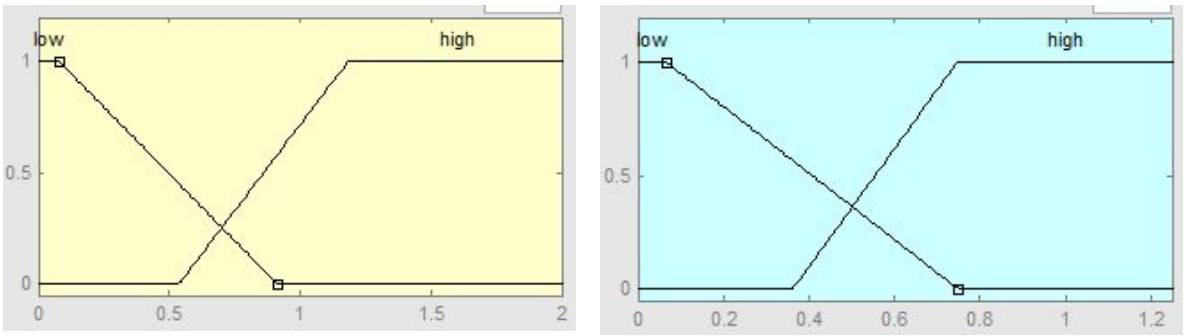
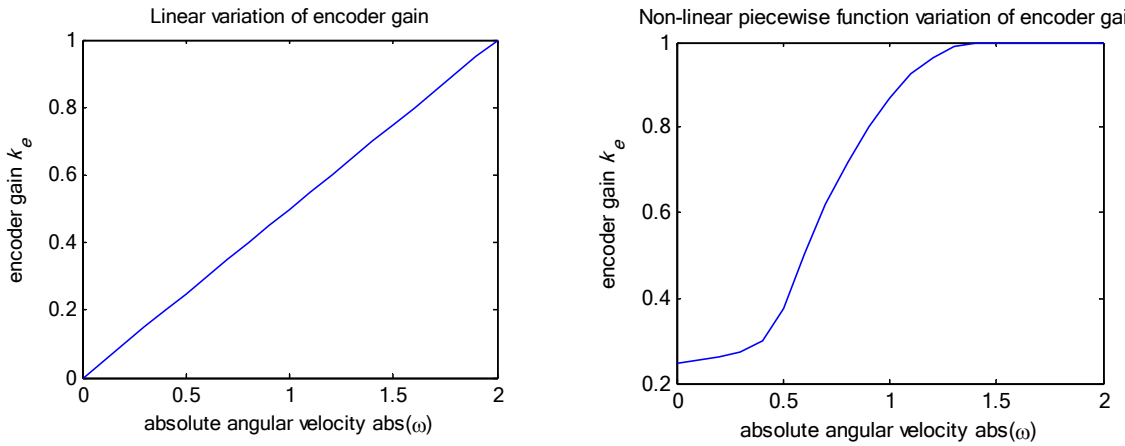


Figure 9. Indoor corridor environment with waypoints.

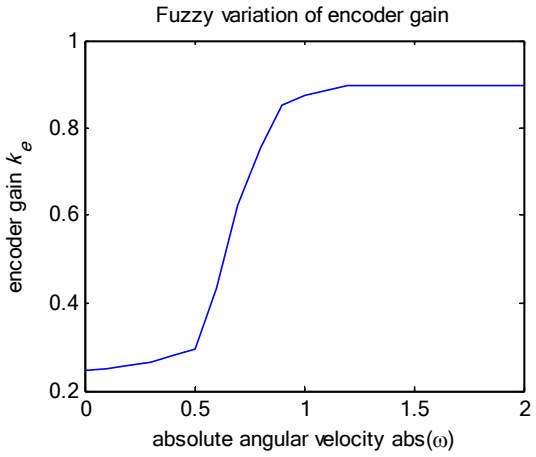


(a) (b)

Figure 10. Membership functions of the fuzzy weighting function. (a) Input; (b) Output



(a) (b)



(c)

Figure 11. Three encoder heading weighting functions for linear weighted fusion. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

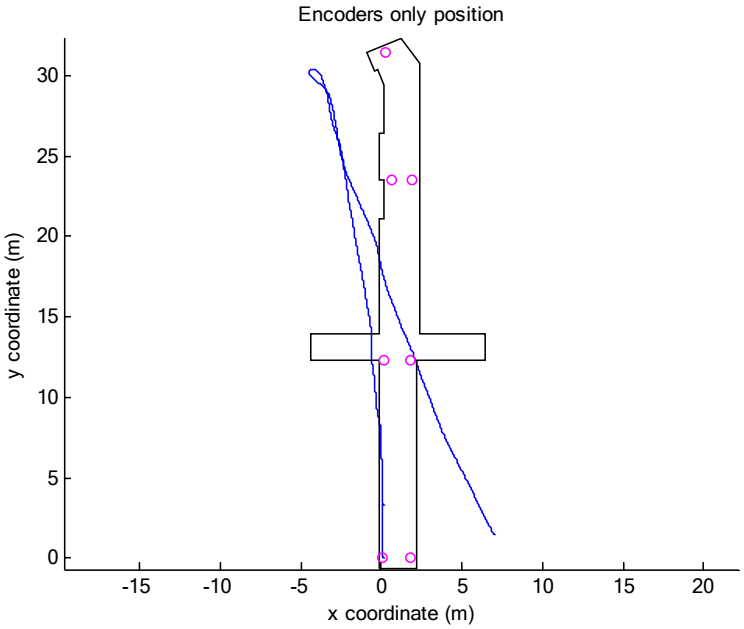


Figure 12. Position estimation using encoders only.

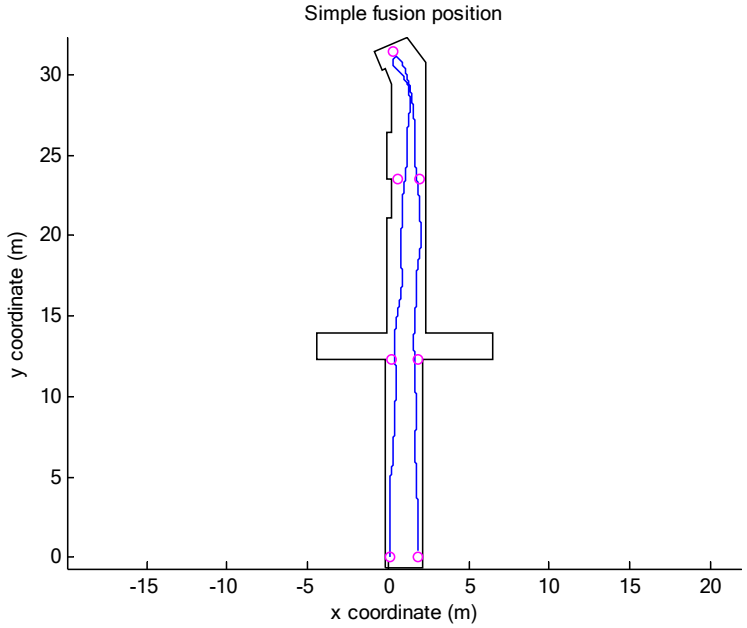


Figure 13. Position estimation using simple fusion.

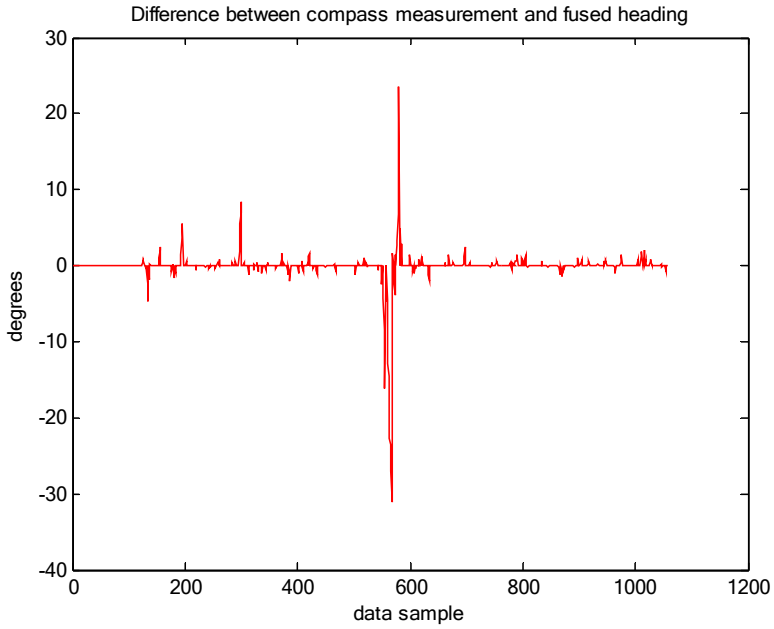


Figure 14. Difference between compass measurement and simple fusion heading.

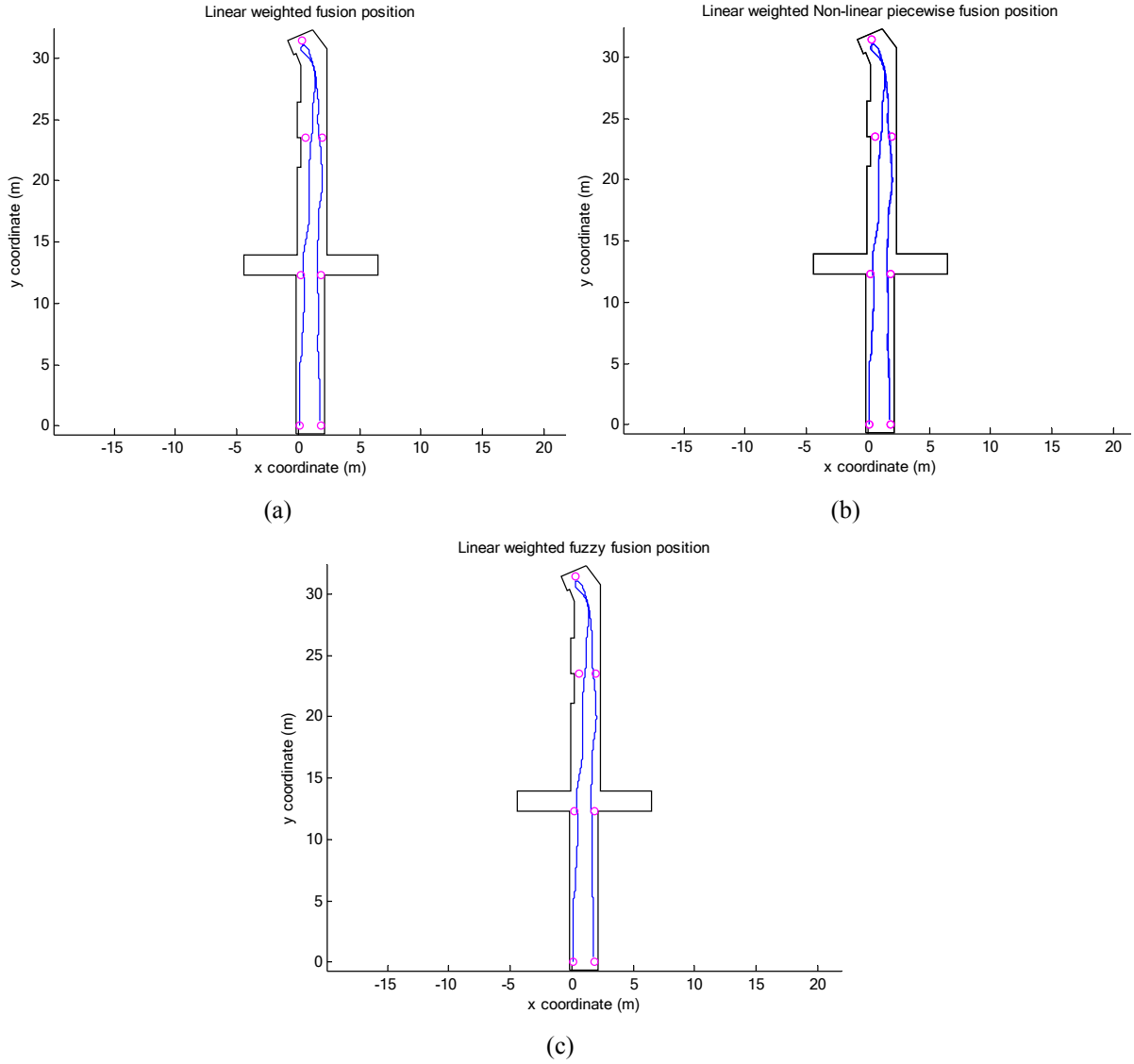


Figure 15. Position estimation using linear weighted fusion with encoder heading weighting functions. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

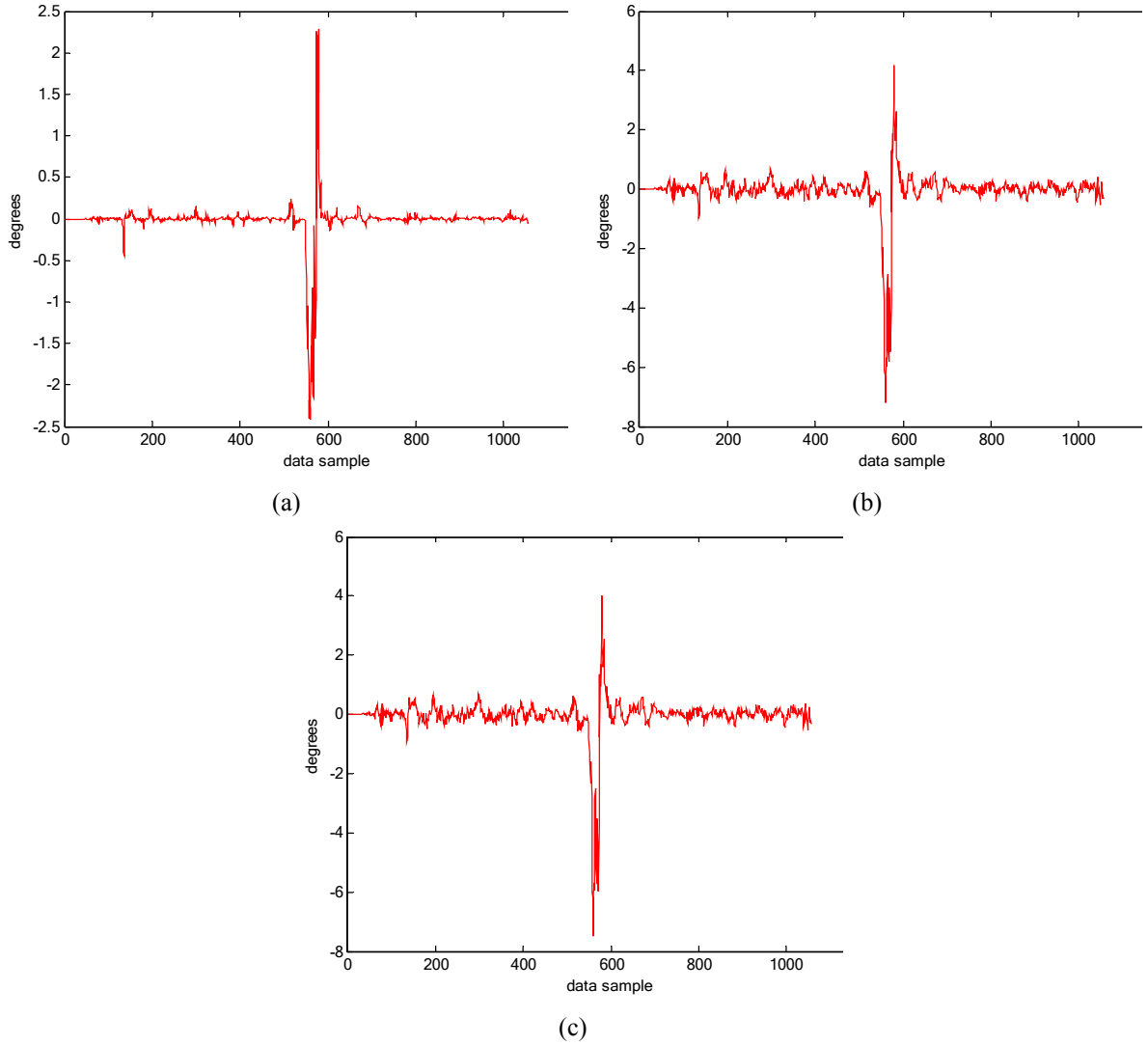


Figure 16. Difference between compass measurement and linear weighted fusion heading. (a) Linear; (b) Non-linear piecewise; (c) Fuzzy

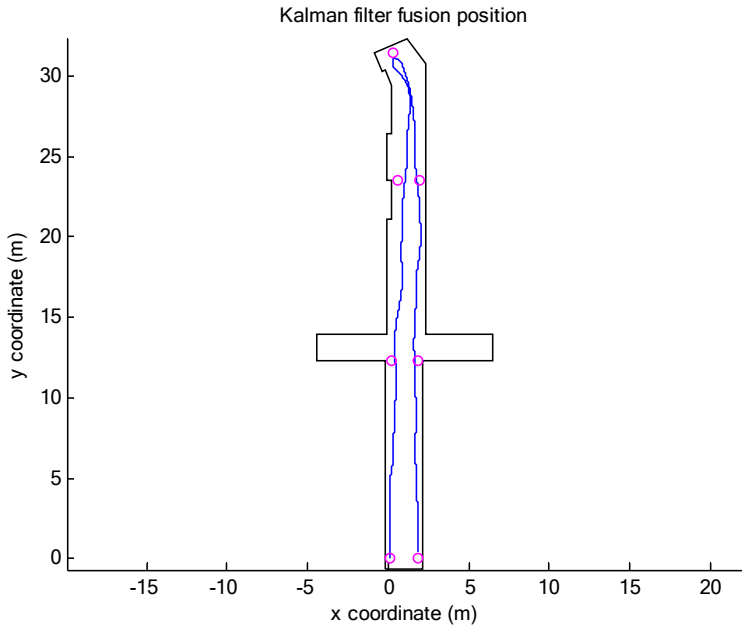


Figure 17. Position estimation using the Kalman filter

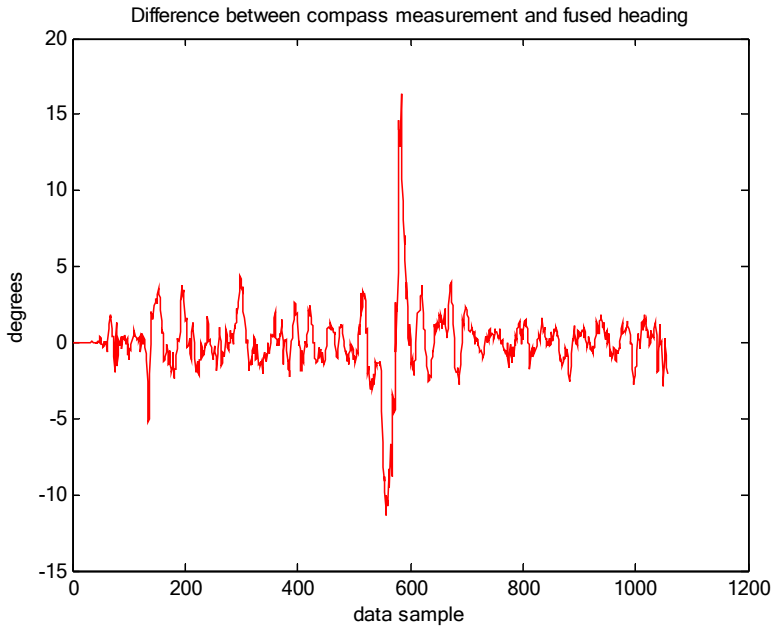


Figure 18. Difference between compass measurement and Kalman filter fusion heading

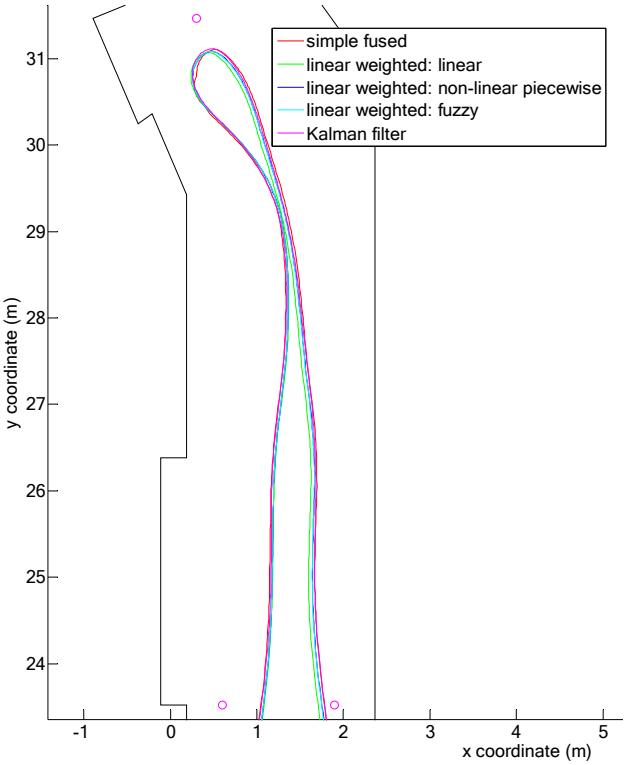


Figure 19. Upper section of corridor comparison

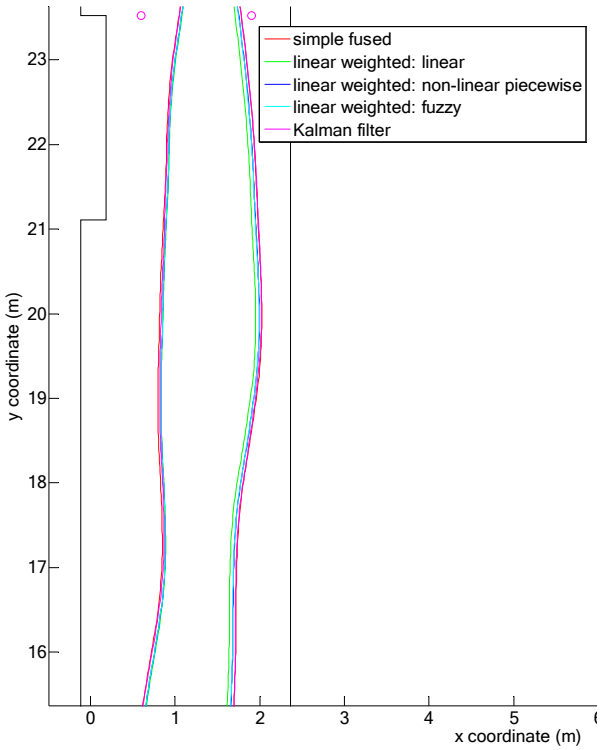
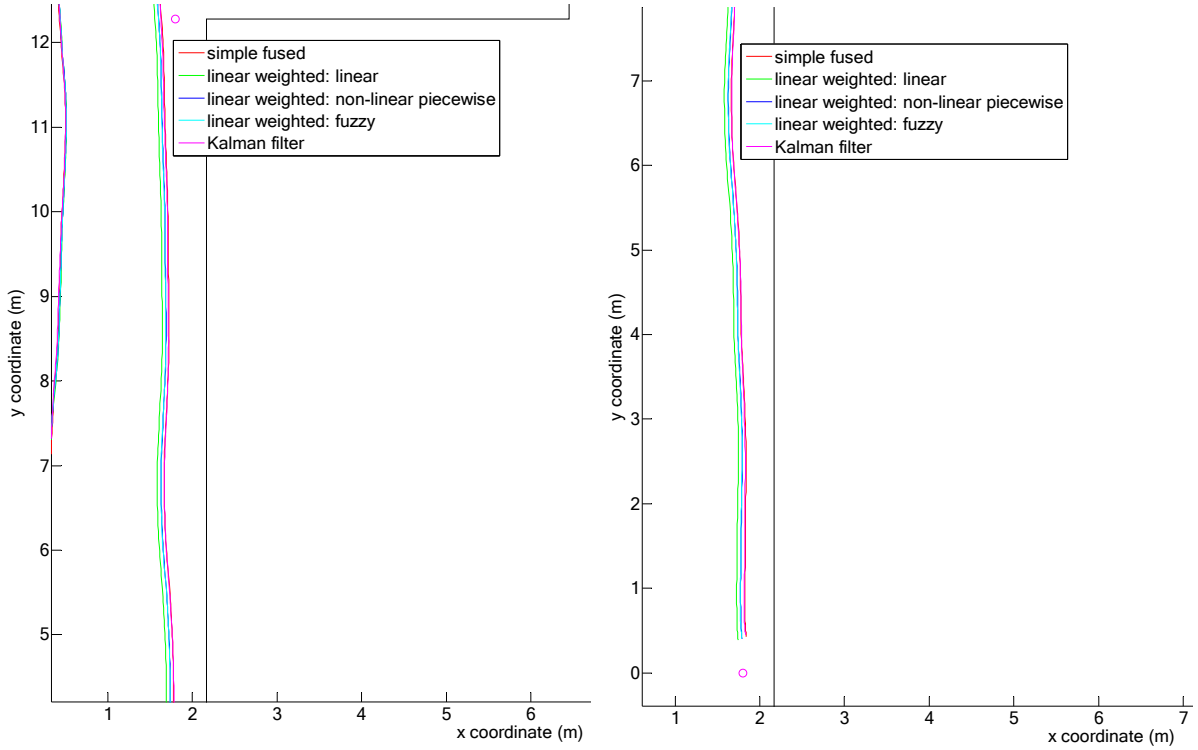


Figure 20. Middle section of corridor comparison



(a) (b)

Figure 21. End of path comparison. (a) Second last waypoint; (b) Last waypoint