ABSTRACT

Fuzzy logic and Neural Network-aided Extended Kalman Filter for Mobile Robot Localization

Zhuo Wei
University of Guelph, 2011

Advisor:
Professor Simon X. Yang

In this thesis, an algorithm that improves the performance of the extended Kalman filter (EKF) on the mobile robot localization issue is proposed, which is aided by the cooperation of neural network and fuzzy logic. An EKF is used to fuse the information acquired from both the robot optical encoders and the external sensors in order to estimate the current robot position and orientation. Then the error covariance of the EKF is tracked by the covariance matching technique. When the output of the matching technique does not meet the desired condition, a fuzzy logic is employed to adjust the error covariance matrix to modify it back to the desired value range. Since the fuzzy logic is lack of the capability of learning, a neural network is presented in the algorithm to train the EKF. The simulation results demonstrate that, with the comparison to the odometry and the standard EKF method under the same error divergence condition, the proposed extended Kalman filter effectively improves the accuracy of the localization of the mobile robot system and effectively prevents the filter divergence.
Dedication

To my dear parents, for their endless love to me.
Acknowledgements

I would like to express my sincere gratitude and indebtedness to my supervisor, Dr. Simon X. Yang for continuous support, guidance, and inspiration throughout my graduate studies and thesis research.

I wish to express my immense gratitude to my advisory committee member Dr. Dutta and the examination committee chair Dr. Gregori for their constructive suggestions for this thesis.

I would like to acknowledge all members in ARIS lab for creating a friendly and enjoyable working environment throughout my study.

There are many other people whom I must thank for their help over the past two years. Thank you for having been invaluable to me.
Contents

List of Figures vi
List of Tables ix
List of Symbols x
List of Abbreviations xii

1 Introduction 1
  1.1 Statement of the Problem 2
  1.2 Objectives of This Thesis 3
  1.3 Contributions of This Thesis 4
  1.4 Organization of This Thesis 4

2 Background 6
  2.1 Mobile Robots and Robot Modelling 6
    2.1.1 Single Wheel Driven 6
    2.1.2 Synchro-drive 7
    2.1.3 Ackerman Steering 7
    2.1.4 Differential Drive 8
  2.2 Sensors for the Mobile Robot 10
    2.2.1 Internal Sensors 10
    2.2.2 External Sensors 11
  2.3 Multi-sensor Fusion Methods 13
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.1</td>
<td>Bayesian Network</td>
<td>13</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Dempster-Shafer Theory</td>
<td>13</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Kalman Filters</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>Artificial Intelligence</td>
<td>15</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Fuzzy Logic</td>
<td>15</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Neural Networks</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>Literature Survey</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>Localization Methods</td>
<td>19</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Relative Methods</td>
<td>19</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Absolute Methods</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td>Artificial Intelligence-aided Extended Kalman Filters</td>
<td>24</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Extended Kalman Filters</td>
<td>24</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Fuzzy Logic-aided Extended Kalman Filters</td>
<td>25</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Neural Network-aided Extended Kalman Filters</td>
<td>28</td>
</tr>
<tr>
<td>3.3</td>
<td>Concluding Remarks</td>
<td>29</td>
</tr>
<tr>
<td>4</td>
<td><strong>Fuzzy Logic-aided Extended Kalman Filter</strong></td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Robot Modelling and Sensors</td>
<td>30</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Robot Modelling</td>
<td>31</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Robot Sensors</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>The Proposed Method</td>
<td>37</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Extended Kalman Filter</td>
<td>37</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Fuzzy Logic-aided EKF</td>
<td>45</td>
</tr>
<tr>
<td>4.3</td>
<td>Simulation Results</td>
<td>49</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Typical EKF</td>
<td>50</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Fuzzy Logic-aided EKF</td>
<td>55</td>
</tr>
<tr>
<td>4.4</td>
<td>Concluding Remarks</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td><strong>Neural Network-aided Extended Kalman Filter</strong></td>
<td>62</td>
</tr>
<tr>
<td>5.1</td>
<td>The Proposed Method</td>
<td>62</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Model of the single wheel mobile robot. ............................... 7
2.2 Model of the synchro drive. ........................................... 8
2.3 Model of the Ackerman steering. .................................... 8
2.4 Model of a differential driving robot. ............................... 9
2.5 A fuzzy logic architecture. ............................................. 16
2.6 A feed-forward neural network architecture. ....................... 17

3.1 The diagram of the position estimation system. ...................... 20
3.2 Odometry estimation via home positioning. ......................... 22
3.3 Absolute localization method via RFID system. ..................... 23
3.4 Block diagram of the localization system. .......................... 23
3.5 Membership function of a process state. ............................ 26

4.1 The kinematic model of a mobile robot. ............................. 31
4.2 The relative localization estimation of the mobile robot. .......... 33
4.3 Difference between dead reckoning and the robot actual position. 33
4.4 Sensor locations on the experimental robot and its diagram. ...... 36
4.5 Ultrasonic sensor measurement. ....................................... 37
4.6 The ongoing Kalman filter cycle. .................................... 39
4.7 The EKF schemes. .................................................... 42
4.8 The EKF performance at various $R$ values. ......................... 43
4.9 The flow diagram of the proposed EKF tuned by fuzzy logic. .... 46
4.10 The employed fuzzy architecture. .................................. 46
4.11 Input membership function. .......................................... 47
6.7 The localization estimation of the robot of sine wave experiment. . . 84
6.8 The robot follows the rectangular trajectory. . . . . . . . . . . . . . . . 85
6.9 The velocities of the robot to follow a rectangular trajectory. . . . . 86
6.10 Measurements of robot sensors. . . . . . . . . . . . . . . . . . . . . . . . 87
6.11 The robot position on the rectangular path. . . . . . . . . . . . . . . . . . . . . 88
6.12 The localization of the robot on rectangular trajectory experiment. . 89
List of Tables

4.1 Rule table for the scaling factors. 49
6.1 Employed components of the experimental robot. 76
6.2 Comparison between the actual position and the estimated position. 79
List of Symbols

\( C_r \) Actual residual vector of the process
\( d \) Distance between the two wheels
\( d_l, d_r \) Travelled distance of left and right wheels
\( E \) Encoder difference of the optical encoder
\( e_i \) Error of the \( i \)-th neuron
\( K \) Gain of the Kalman filter
\( P \) Error covariance of the state \( \hat{X} \)
\( P^- \) Priori state of the error covariance at current time stamp
\( P^j \) A plane in the environment
\( P^{\phi}_j \) Angle between the normal line to \( j \)-th plane and the \( X \)-direction
\( P^r_j \) Normal distance of \( j \)-th plane from the origin \( O \)
\( P^\nu_j \) Reflecting factor of \( j \)-th plane
\( Q \) Process noise covariance
\( R \) Measurement noise covariance
\( r \) Error of the measurement
\( t_i \) Sample data of the \( i \)-th neuron
\( v \) Linear velocity of the robot
\( v_L, v_R \) Velocities of the left and right wheels
\( w \) Error of the modelling process
\( w_i \) Weight of the \( i \)-th neuron
\( \hat{X} \) State of the robot movement process
\( \hat{X}^- \) Priori state of the process at current time stamp
\( x, y \) \( X \) and \( Y \) coordinates of a robot in Cartesian workspace
$x_i', y_i'$  \quad X$ and $Y$ coordinates of $i$-th sensor in Cartesian workspace  

$y_i$  \quad Computed output of the $i$-th neuron  

$Z$  \quad Sensor readings from the employed sensors during the process  

$\alpha$  \quad Degree of mismatch  

$\varepsilon$  \quad Predicted residual vector of the Kalman filter  

$\hat{\varepsilon}_r(t)$  \quad Theoretical value of the innovation process  

$\zeta(t)$  \quad Average of the diagonal elements of $\hat{\varepsilon}(t)$  

$\eta$  \quad Learning rate of the learning process  

$\eta_R, \eta_Q$  \quad Scaling factors of the error covariance $R$ and $Q$ respectively  

$\theta$  \quad Orientation angle of the robot  

$\theta_i'$  \quad Orientation of $i$-th sensor  

$\xi(t)$  \quad Average of the diagonal elements of $\varepsilon(t)$  

$\sigma_l, \sigma_r$  \quad The variance of encoder errors of left and right wheel  

$\omega$  \quad Angular velocity of the robot
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>BP</td>
<td>Back Propagation</td>
</tr>
<tr>
<td>DR</td>
<td>Dead Reckoning</td>
</tr>
<tr>
<td>EKF</td>
<td>Extended Kalman Filter</td>
</tr>
<tr>
<td>FL</td>
<td>Fuzzy logic</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial Measurement Unit</td>
</tr>
<tr>
<td>KF</td>
<td>Kalman Filter</td>
</tr>
<tr>
<td>LMS</td>
<td>Least Mean Square</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
<tr>
<td>NNEKF</td>
<td>Neural Network-aided Extended Kalman Filter</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RFID</td>
<td>Employing Radio-frequency Identification</td>
</tr>
<tr>
<td>TSFS</td>
<td>Takagi-Sugeno Fuzzy Structure</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

During the last few decades, many researchers have dedicated their efforts to construct revolutionary robots and to provide them with some kind of artificial intelligence to perform risky or tedious tasks historically assigned to the human being (Castellanos and Tardós, 1999). At present day, robots have widely played important roles in many applications such as deep ocean exploration, manufacturing and delivery in factories, and entertainment. Among them, the mobile robot is undoubtedly one of the most state-of-the-art research directions. With the capability of moving around in their working environment but not being fixed to one location, the mobile robot has flexibility and adaption for the working space to fulfil much more complicated tasks. The main categories of the mobile robots can initially be divided into: working within the known indoor environment and oriented to the unknown outdoor environment.

No matter what kind of the environment the mobile robots are working in, one of the major problems to operate the driving robots is navigation. Stated most simply, the problem of navigation can be summarized into answering the following three questions: “where am I”, “where am I going”, and “how should I get there” (John and Durrant-Whyte, 1991). This thesis is aimed at answering the first question which is also called localization problem, based on the previous knowledge of the environment and current measurements from the robot. For many localization scenarios, what robots need to know is their position and orientation at all times. For example, a cleaning robot needs to make sure it covers the whole target area without getting
lost. Thus it should be able to navigate the working place and know its position and orientation regarding its starting point.

The mobile robot navigation is in general an incremental process that can be summarized in four main steps (de Almeida, 1998):

- Environment perception and modelling: Any motion requires a representation of the local environment at least, and often a more global knowledge.

- Localization: The robot needs to know where it is with respect to its environment and goal.

- Motion decision and planning: the robot has to decide how, where and which way to go.

- Motion execution: the command corresponding to the motion decisions are executed by the control process.

This thesis will focus on using the multi-sensor fusion to provide reliable and accurate localization information to the robot for decision making and motion control tasks.

1.1 Statement of the Problem

For the mobile robot moving in an indoor environment, localization is always an issue of concern. Odometry method is the traditional method, which uses the measurement of wheels rotation and/or steering orientation from the internal sensors to estimate the position of the robot. However, during a long period of time of working on a long trajectory, especially with some wind path, it will inevitably encounter the problem of error accumulation. The localization estimation will further diverge from the desired value as the proceeds, even worse, the divergence of the estimation would be boundless. Thus the mobile robot is usually equipped with several exteroceptive sensors to interact with the surrounding environment. The multi-sensor system eliminates the limitation where only internal sensors are used to provide the information for the
robot position. To fuse the measurements from external sensors and internal sensors is currently one of the best ways to deduce error accumulation.

Multi-sensor fusion is the combination of sensory sources to generate better information in a certain sense than these sources were used alone. The term "better" in this case means more accurate and more reliable. Fusion methods require well predefined models of both the robot and errors. Without these preliminary requirements, the performance of the fusion may be unpredicted and unstable. Artificial intelligence (AI) is a very feasible way to optimize the sensor fusion mechanism. There are several methods of the AI, like neural network (NN), fuzzy logic (FL) and genetic algorithm.

1.2 Objectives of This Thesis

The objectives of this thesis are to develop an effective method to fuse the sensory data, in addition, to apply the AI algorithms to optimize the performance of the selected fusion method and prevent it from divergence.

There are usually several sensors mounted on the mobile robot to interact with the environment. Sensor fusion algorithms work on combining data from sensors in order to obtain augmented results. In mobile robots, sensor fusion algorithms are used at different levels: on one hand, to fuse information obtained from different sensors, and on the other hand, to integrate current sensor observations into previously available knowledge of the navigation area (Castellanos and Tardós, 1999). There are three kinds of sensor fusion methods being widely employed in robots: Kalman filter, Bayesian networks, and Dempster-Shafer. The effective and suitable sensor fusion method would be able to prevent error accumulation and reduce the error of localization. In this thesis, an extended Kalman filter (EKF) is the way selected to do the sensor fusion. The Kalman filter is recursive in that each update of the state is computed from the previous estimate and the new input data of both internal encoders and exteroceptive sensors (Haykin, 2001).

In addition to present sensor fusion, the way to guarantee the good performance of the EKF method is also required. Actually the accuracy of the EKF is largely
dependent on the predefined model, especially the model of error covariance which
is not as reliable because the error is not fixed. Thus an error track mechanism to
track the change of the error is needed. In addition, an adjustment algorithm to
simultaneously adapt the error model to any error changes is presented in this thesis.
A FL is employed beforehand to discover target values for training the NN. An well
trained NN will effectively aid the EKF to control the error covariance according to
the error updates.

Experiments on a real mobile robot will be conducted to validate the effectiveness
and accuracy of proposed algorithms.

1.3 Contributions of This Thesis

The main contribution of this thesis is to develop a more reliable EKF sensor fusion
algorithm to improve the accuracy of the mobile robots localization. The way of the
NN-aided EKF on the mobile robot localization problem is first proposed. With the
NN-aided EKF, the predefined error covariance of the EKF would be able to respond
to the errors change, and the provided localization value is more reliable than the
normal EKF.

In addition, a fuzzy logic method designed to train the NN is also presented.
Based on the fuzzy logic controller, the desired values of the NN training is explored.
The output of the fuzzy controller will be used to train the NN.

With the aid of the NN and fuzzy logic, the EKF is capable of estimating the
position and orientation of the robot. By experimenting on a real robot, it can
demonstrate the effectiveness of the proposed EKF algorithm.

1.4 Organization of This Thesis

The organization of this thesis is as follows.

Chapter 1 gives a brief introduction to the thesis. The problems to be solved, the
objectives and the contributions of this thesis are summarized in this chapter.
Chapter 2 provides the background related to the methods. The basic knowledge about the mobile robot modelling is addressed first. Then the classifications of sensors, normal multi-sensor fusion methods, and the fundamentals of artificial intelligence, including fuzzy logic and neural network, are also discussed in this chapter.

Chapter 3 presents the literature review of the studied topics. The introduction and conclusion about two main kinds of localization method, relative and absolute method, are presented at the beginning of the chapter. The employment of the EKF in the mobile robot localization is also addressed. The NN-aid and FL-aided extended Kalman filters regarding the localization problem are discussed respectively later.

Chapter 4 presents the proposed fuzzy controller for the EKF error covariance tuning. A fuzzy controller for the mobile robot error control is designed and analyzed. The proposed controller take the diagonal elements of the matrix of degree of mismatch as input to adjust the errors covariance corresponding to the dynamic error change.

In Chapter 5, a NN is employed to be trained for the working environment. The NN takes the output of FL as the desired value to train itself. The well trained NN module can then be used to control the EKF to prevent it from divergence. The performance of the NN-aided EKF is illustrated by simulation studies.

Chapter 6 addresses the experiments of the proposed algorithm on a real robot, including desired trajectory following and localization.

Conclusions and future work are presented in Chapter 7.
Chapter 2

Background

In this chapter, the common kinematic models of mobile robots are briefly introduced. In addition, the commonly used sensors are classified into two categories. The widely used sensor fusion methods for the mobile robot, and the fundamentals of the artificial intelligence–neural network and fuzzy logic are also presented.

2.1 Mobile Robots and Robot Modelling

Mobile robots have the capability of moving around in the environment. There are several kinematic models of commonly used robots.

2.1.1 Single Wheel Driven

Having a single wheel that is both driven and steered is the simplest conceptual design for a mobile robot (Braunl, 2006). The single wheel driven model is also called tricycle driven. This kind of design requires at least two passive wheels to keep the balance of the robot. The dead-reckoning equations of the tricycle wheel driven robot are similar to that of the Ackerman-steered robot discussed later. A drawback associated with the single wheel driven model is the robot centre of gravity tends to move away from the driven wheel when traversing up an incline, causing a loss of traction (Everett, 1995). A single wheel driven robot is illustrated in Figure 2.1.
2.1.2 Synchro-drive

In the synchronous drive, three or more wheels are coupling to run toward the same direction at the same speed. When executing a turn, all the wheels steer in union regarding their respective axes. This drive and steering “synchronization” results in improved odometry accuracy through reduce slippage, since all wheels generate equal and parallel force vectors at all times (Johann et al., 1996). A synchro-drive robot is almost a holonomic vehicle, in the sense that it can drive in any desired direction (Braunl, 2006). The model of the synchro drive is illustrated in Figure 2.2. The linear displacement of each wheel in the moving direction is given as follows

\[ D = \frac{2\pi N}{C_e} R_e, \]  

(2.1)

where \( D \) is the vehicle displacement along path, \( N \) is the measured counts of drive motor shaft encoder, \( C_e \) is the encoder counts per complete wheel revolution, and \( R_e \) is the effective wheel radius.

One drawback of this approach is seen in the decreased lateral stability that results when one wheel is turned in under the vehicle (Everett, 1995).

2.1.3 Ackerman Steering

Ackerman steering is known as the standard automobile system, which is composed of two linked driven rear wheels and two coupled steerable front wheels. Ackerman steering is much easier in moving toward any direction, because its driven and steering control are operated on the independent rear and front wheels. The driving library
contains two independent velocity-position controllers, one for the rear driving wheels and one for the front steering wheels (Braunl, 2006). Ackerman steering provides a fairly accurate odometry solution while supporting the traction and ground clearance needs of all-terrain operation. Ackerman steering thus is the choice for outdoor autonomous vehicles (Johann et al., 1996). Figure 2.3 illustrates the model of Ackerman steering.

2.1.4 Differential Drive

The structure of a differential drive design consists of a passive caster used to keep the balance of the robot, and two driven motors mounted on the left and right wheels of the robot, driving each wheel separately. Differential drive is mechanically simpler
than single wheel drive, however, the coordination of two driven wheels makes moving control for differential drive more complex than for single wheel drive. Figure 2.4 demonstrates the driving actions of a differential drive robot.

Figure 2.4: Model of a differential driving robot. (a) Robot drives straight forward or backward at both motors with the same speed; (b) Robot drives in a curve with one motor running faster than another; (c) Robot turns on the spot with both motors running at the same speed in opposite directions. (Modified from Braunl (2006))

In this design, incremental encoders mounted onto the motors count wheel revolutions. The robot can compute its current position via dead reckoning at each sample instance. The incremental linear displacement of the robot centre point, denoted $\Delta S$, is calculated by

$$\Delta S = \frac{1}{2}(v_l + v_r)\Delta t,$$  \hspace{1cm} (2.2)

where $v_l$ and $v_r$ are the linear velocities of the left and right wheels, respectively. The incremental change of the robot orientation is

$$\Delta \theta = \frac{(v_r - v_l)\Delta t}{d},$$  \hspace{1cm} (2.3)

where $d$ is the distance between two driven wheels. The new relative orientation $\theta$ of
the robot can be computed from

\[ \theta(t + \Delta t) = \theta(t) + \Delta \theta, \quad (2.4) \]

and then the relative position of the centre point is

\[ x(t + \Delta t) = x_t + \Delta S \cos \theta, \quad (2.5) \]
\[ y(t + \Delta t) = y_t + \Delta S \sin \theta, \quad (2.6) \]

where \( x_t \) and \( y_t \) are the relative position of the robot centre-point position at an instance.

Differential drive is the most widely used model in the indoor environment. The robot we experimented on, Dr. Robot Sentinel, is based on the design of the differential drive with two steerable driven wheels and a passive caster.

### 2.2 Sensors for the Mobile Robot

A large variety of the robotic applications take place in large and unstructured areas with the challenge of uncertainty. Starting from the goal of following a path or reaching the destination precisely, a robot should be able to interact with the surrounding environment by sensing. There are a vast number of different sensors used on solving robotics localization problems, applying different measurement techniques, and using different interfaces to a controller (Teslic et al., 2009). In an outdoor setting this could be the satellite-based GPS. In an indoor setting, a global sensor network with infrared, sonar, laser, or radio beacons could be employed.

For the indoor mobile robot systems, the mounted sensors can be classified into two categories: internal and external.

#### 2.2.1 Internal Sensors

Internal sensors are the sensors monitoring the robot internal state (e.g. motor speed, wheel axes, battery status).
Encoders

Encoders are the type of sensor mounted on the motor shaft to measure the velocity or the rotational displacement of the robot (Tseng et al., 2010). Magnetic and optical are widely used encoders. They count the number of passed segments from a certain starting point to measure the speed, although they employ different technologies to complete the count. Encoders are standard sensors for odometry to estimate the position and orientation of the robot. In addition, encoders are equipped as the fundamental feedback sensors for motor control (Yang et al., 2010).

Accelerometers

Accelerometer is a simpler orientation sensor to measure the robot orientation. In some cases, the orientation measured from accelerometer is needed to compensate for any small momentary angular error. Accelerometer is designed to measure a single or two axes at once. However, there are several crucial drawbacks. It lacks the capability of properly handling the jitter. Moreover, it is quite sensitive to positional noise (Becker and Burgard, 2010).

2.2.2 External Sensors

External sensors are the sensors monitoring the robot running environment. Besides the internal sensors, almost all mobile robots are equipped with various external sensor types for measuring distances to the nearest object around the robot for navigation purposes (Fujii et al., 2010).

Sonar Sensors

Sonar sensors interact with the environment via emitting a short acoustic signal at an ultrasonic frequency and measuring the time-of-flight till the echo is received back to the sensor (Hwang et al., 2010). When several sensors are working together, crosstalk may be the main problem that interferes with the accuracy of sonar measurements.
Sonar sensors can be seen in the vast number of indoor robotics systems, due to its low-cost and ease of use.

**Infrared Distance Sensors**

Infrared sensors typically consist of a pulsed infrared LED together with a detection array. The measure of distance to the object is according to the angle changes of the reflected beams (Wu and Wei, 2009). The transferred infrared wave can only respond to the IR detection boards. The main limitation of an infrared sensor is that its measurement range is significantly shorter than any other sensors.

**Laser Scanners**

Laser sensors have gradually replaced sonar sensors in many robots. A highly precise local 2D map, or even 3D map, is returned by the laser sensor at the viewpoint of the robot. However, the mobile robot has to compromise mobility to the large and heavy employed lase sensor (Hu et al., 2009).

**Overhead Cameras**

Digital camera is the most complicated and advanced sensor used in robotics. A high frame rate is required to update sensor data from the moving robot as fast as possible, and also the image resolution must be sufficient to detect the desired object from a certain distance. Thus there is always a trade-off between high frame rate and high resolution (Yoon et al., 2010). It is an inevitable large expense on a camera with high resolution and transfer rate.

The mobile robot used in this thesis is equipped with five external range sensors for simulation studies. The WiRobot in the ARIS Lab is equipped with three ultrasonic sensors, seven infrared sensors, one CCD camera and several internal sensors (voltage sensor, current sensor, etc.).
2.3 Multi-sensor Fusion Methods

A mobile robot always equips with several internal and external sensors to accomplish different tasks. Mobile robots have to coordinate different sensors via sensor fusion to discover the given environment. Sensor fusion is a method of extracting sensory data from a different source to integrate into single signals (Sasiadek, 2002). The widely applied sensor fusion algorithms are the 3 types: Bayesian network, Kalman filter and Dempster-Shafer theory.

2.3.1 Bayesian Network

Occupancy grid is a stochastic tessellated representation of spatial information about the robot operating environment (Zhou and Sakane, 2002). The environment is explicitly represented by lots of small square spatial cells, which store the probabilistic estimation of the occupancy. The occupancy grid is incrementally updating via the Bayesian network using readings from several sensors. The Bayesian network allows us to represent casual and contextual relations between sensing data or evidence and beliefs about the global situation in a natural manner (Singhal and Brown, 1997). The experimental results show that the occupancy grid approach is robust under sensor uncertainty and errors, and allows explicit handling of uncertainty (Stanislav et al., 2010). The main drawback of the Bayesian network is that a well predefined model of employed sensors are needed otherwise Bayesian network cannot work as expected.

2.3.2 Dempster-Shafer Theory

Dempster-Shafer theory is a generalization of Bayesian theory of subjective probability, which provides a mathematical theory of evidence (Cao et al., 2009). Dempster-Shafer theory is a common used sensor fusion method. It arrives at a degree of belief based on independent items of evidence from different sources and provides a belief function to combine these evidences to form a set of aggregating beliefs. In applying the Dempster-Shafer theory, sensory data is a source of evidence about a set of
propositions of interest (Yi et al., 2000). Its one disadvantage is computationally intractable because a belief function must distribute belief to the power set of all the features in the world (Murphy, 1998).

2.3.3 Kalman Filters

The Kalman filter (KF) is a form of optimal estimation characterized by recursive evaluation, an internal model of the dynamics of the system being estimated, and a dynamic weighting of incoming evidence with ongoing expectation that produces estimates of the state of the observed system (Guo et al., 2009; Wu and Sun, 2010; Park et al., 2010). The purpose of the KF is to utilize the noisy measurements that are observed over times, and to produce estimations that tend to be closer to the true values of the measurements and the state.

The importance of the usage of the KF is that all the plant dynamics and noise processes are exactly known and the noise processes are zero mean white noise (Park et al., 2009). However, it is difficult to get a priori reliable definition about both the state-space of the robot and the measurement error covariance, because of the model inaccuracy. It is well known that a poor estimated error covariance may cause a serious divergence. There are two kinds of divergence: Apparent divergence and True divergence. In the apparent divergence, the actual estimate error covariance remains bounded, but it approaches a larger bound than the predicted error covariance. In true divergence, the actual estimation covariance eventually becomes infinite (Panzieri et al., 2008).

In recent decades, artificial intelligence approaches are proposed to assist the Kalman filter. The AI methods aim at helping the Kalman filter to prevent from divergence and improve its accuracy. Fuzzy logic and neural networks are the AI algorithms being employed for this purpose. Both of them are taken to control the error covariance of the Kalman filter.
2.4 Artificial Intelligence

Artificial intelligence is one of the common methods employed in tuning the Kalman filter. In this chapter, a brief introduction about two common artificial intelligence algorithms, fuzzy logic and neural networks, is presented.

2.4.1 Fuzzy Logic

Fuzzy logic is a form of many-valued logic derived from a fuzzy set theory to deal with classes that are approximated rather than fixed and exact, which cannot be handled by the tradition logic controller (Zadeh, 1965). A fuzzy set is a class of variables with a continuum of membership, which is classified by a grade ranging between 0 and 1 from a membership function. In simple words, fuzzy logic is a super set of conventional logic that is designed to deal with the partial true or partial false situation and put the information into the partial membership set rather than a crisp set (Amouri-Jmaiel et al., 2009).

Fuzzy set theory defines fuzzy operators based on IF-THEN rules, which is usually expressed in the form: IF variable is property THEN action. A Fuzzy control system is consisted of several parts which is illustrated in the Figure 2.5. There are two typically used fuzzy controller systems which are designed by Mamdani (Mamdani and Assilian, 1975) and Sugeno (Takagi and Sugeno, 1985) respectively. The steps of fuzzy controller system design is basically accomplished as following:

- Determine the fuzzification method. Select membership functions for both inputs and outputs.
- Determine the fuzzy inference system. Document the rule set via a set of IF-THEN expressions.
- Determine the defuzzification method.

Fuzzy systems are suitable for the modelling of vague knowledge. A fuzzy controller is mainly for systems without models or with inconvenient nonlinear structures, which are difficult to control with a classical controller (Tan, 2009).
2.4.2 Neural Networks

Neural networks, which are also called artificial neural networks, are the algorithms often used on solving classification problems or decision making problems that cannot be handled by a simple or straightforward algorithmic methods. A neural network is an adaptive system that adjusts its structure according to the external or internal information that flows through the network (Parhi and Singh, 2009). Actually, terms “neural networks” are non-linear statistical data modelling tools.

A neural network is constructed from a number of layers which is arranged by interconnected neurons. The outputs of one layer are the inputs of the following layer. The first layer of neurons is called the “input layer”, and the last layer of neurons is called the “output layer”. In addition, all neuron layers between the input layer and output layer can be called “hidden layers” (Braunl, 2006). Neurons are the processing elements that are linked with its certain neighbour via weighted connections. Each individual neuron has a number of inputs, a processing node and a single output. Processing in a neural network takes place in parallel in all neurons (Wang, 2004).

A model of the feed-forward neural networks architecture, which is all connected from one way direction, is illustrated in Figure 2.6, where $x$ is the input vector, $y$ is the output vector of the neural network and $v, w$ are the weights between the layers.

A neural network has the capability of learning. In its learning process, it training itself by employing sample cases mapping from input to output, and then being able
to generalize this mapping to cases not seen previously (Judd, 1990). The learning is accomplished by adjusting weights between the layers to obtain the closely desired outputs. There are several different learning techniques used in neural networks, which include supervised and unsupervised techniques, depending on whether desired values to a training case existed or not, as well as on-line or off-line learning, depending on whether the network evolves during the execution or not.

In this thesis, a feed-forward network architecture, back propagation (BP), is taken to simulate the control of error covariance of both input and measurement error covariance of the Kalman filter. BP can be used to identify a certain situation from the network input and produce a corresponding output signal. The connection of the BP neural network among layers is one way direction, so that the learning process is composed of the forward propagation of the input and back propagation of the error. The network starts with random weights and is tested by a number of test cases called testing set. The outputs of the network are compared with the
known testing set. Having done this for a number of iterations, the NN hopefully has learnt the complete training set and should be able to produce the expected outputs corresponding to known or unknown inputs. The well trained NN is able to identify a certain situation from the network input and produce corresponding output signals.
Chapter 3

Literature Survey

The localization algorithms are employed to obtain the robot localization in the global coordinate system. The goal of the robot localization is concerned with the problem of producing precise and reliable robot positions during the operation.

3.1 Localization Methods

For accomplishing the navigation tasks, the robots need to know where they are, and need to make a plan for how to reach a goal destination. These two problems are not isolated from each other, but rather closely linked. If a robot does not know its exact position during the process of a planned trajectory, it will encounter problems in reaching the destination (Braunl, 2006). The solutions of the robot localization can be roughly divided into two different kinds of categories: relative and absolute.

3.1.1 Relative Methods

Relative method is based on the internal dynamic variables of the vehicle, like velocity and orientation, provided from internal sensors. Optical incremental encoders which are the typical internal sensors, are mounted on the axis of the driving wheels or the steering axis of the vehicle, to measure wheel rotation or/and steering orientation.

At each sampling instant, the position of the robot is estimated on the basis of
the encoder increments along the sampling interval. Cox presented an relative algorithm to estimate the position of Robot Blanche (Cox, 1991). In his paper, \((x, y, \theta)\) is employed to denote the position of the robot with respect to the global frame. Meanwhile, a map which represents the operating environment is also brought into account. The algorithm compares the output position from odometry with desired location on the referred map to see if they are matched. This method effectively corrects the odometry error via an additional controller. However, the main drawback of this method is that the representation map is not easy to be precisely created and also the error from the encoder still has not been controlled in the position estimation. Figure 3.1 illustrates the block diagram of this method.

Methods focused on calibrating the encoder error are presented to aim on improving the relative localization accuracy. Johann classified the errors into two categories: systematic error and nonsystematic error. Systematic errors are vehicle specific and do not usually change during run, which are caused by imperfection in the design and mechanical implementation of a mobile robot. Nonsystematic errors are due to the interaction of the robot with unpredictable features of the operating environment (Borenstein and Feng, 1996). In the paper (Martinelli et al., 2007), the presented relative method has a high accuracy because of the correction of errors. A bench mark test is applied to discover the odometric systematic errors, which are caused by the
wheel diameter ratio, wheelbase uncertainty and scaling. Then as a result of running calibration experiment, the correction of odometry error is accomplished. The robot performance on tracking the reference trajectory, $4m \times 4m$ closed squared path, is improved because of the new odometry. However, the calibrated robot is working well only in a short period of time and also on a path with less winding. In addition, the nonsystematic error is ignored.

The methods that are able to deal with both nonsystematic and systematic error are also presented. One algorithm utilizes the Jacobian matrix to determine the discrete pose measurements of the vehicle instead of velocity ones (Batlle et al., 2010). This approach is able to deal with systematic and nonsystematic odometry errors. The experiment of proposed odometry calibration is operated on a circular trajectory, which is the challenging path to run. This method is suitable for almost all the models of indoor mobile robots with the invariant Jacobian. Another novel algorithm is focused on a practical aspect using home positioning. “Home positioning” is the term referred to a position where the robot starts from to track a arbitrary trajectory and then return back. The main scenario of this algorithm is: starting the robot from its home position to follow an arbitrary trajectory, compare the estimated end position with home position after it comes back to its home, and then repeat these steps to update the systematic parameters till it converges to the expected range (Yun et al., 2008). Figure. 3.2 presents the estimation process.

Both of these methods calibrate the robot odometry estimation and improve the relative localization well under the circular path. However, the main drawback is that they are both working off-line. That means the robot has to move several times at the same trajectory to upgrade its odometry performance prior to the real operation. Also, the robot is unable to handle any abrupt odometry error in time during the operating.

### 3.1.2 Absolute Methods

Absolute method is performed via an appropriate of sensors measuring the features of the environment in which the robot is working. Absolute positioning methods usually
Figure 3.2: Odometry estimation via home positioning. (Modified from Yun et al. (2008))

rely on: (a) navigation beacons, (b) active or passive landmarks, (c) map matching, and (d) satellite-based navigation signals (Borenstein and Feng, 1996).

An absolute localization scheme is proposed for the indoor mobile robot by employing radio-frequency identification (RFID) systems (Han et al., 2007). The mobile robot which carries an RFID reader at the bottom of its body reads the RFID tags attached on the floor. Figure 3.3 presents the model of it. The robot calculates its position and orientation based on the absolute position information contained in these RFID tags. However, this system is too complex and expensive to be widely used for any robot applications. In addition, the RFID tags should be set in the operating environment beforehand and maintained after each usage.

Active beacon is another popular absolute localization method. In the paper (Kim et al., 2006), it illustrates a method using the active beacon and radio frequency to measure the distance. The proposed active beacon system consists of two components: a radio frequency (RF) receiver and an ultrasonic transmitter. During the navigation, the running robot can communicate with any certain beacon by sending the beacon code via RF. When a beacon confirms the receiving code with its own ID, it emits ultrasonic wave to the robot to measure the distance between them. In addition, an algorithm based on a system consisting of a micro electro mechanical
Figure 3.3: Absolute localization method via RFID system. (From Han et al. (2007))

Figure 3.4: Block diagram of the localization system. (Modified from Kim et al. (2006))

systems based digital in-plane 3-axis inertial measurement unit (IMU), and an active beacon system is presented (Lee et al., 2009). The in-plane 3-axis IMU is composed of a x/y-accelerometer which measures the travelling distance, and z-axis gyroscope which measures the angle of the robot. The active beacon system is constituted of the transmitters, sonar sensors and a tag, which is used for absolute position estimation. The employment of two ultrasonic sensors of active beacon system provides the rotation angle calculation of the robot. However, the main drawback of the active beacon based algorithms is the requirement of costly installations and maintenance.

In addition, with any one of these localization methods, the work environment needs either to be prepared or be known and mapped with great precision. The
absolute measures is largely dependent on the features of the environment. A little change to the environment may give rise to erroneous interpretation of the measurements taken by the localization algorithm. Thus in some certain short term, the absolute method probably cannot perform at all.

3.2 Artificial Intelligence-aided Extended Kalman Filters

As mentioned above, the relative localization and absolute localization method have their own uneasy fixed disadvantages. It is common to fuse these two methods to compensate the problem of each other. Thus the external sensory readings and internal sensory readings should both be taken into account. The Kalman filter is one of the most important methods implemented to do the sensor fusion.

3.2.1 Extended Kalman Filters

In 1960, Kalman invented a recursive solution to deal with the discrete-data linear filtering problem (Kalman, 1960). In the paper, the proposed filter, which is called the “Kalman filter”, works in a recursive way to predict the current state of the process which is based on the employment of the current measurement and the priori state of process. The Kalman filter is very powerful in minimizing the state estimation error.

State estimation is performed by two steps which is composed of a set of equations. In the first step, the prediction step, also called time update step, a priori estimation of the state is done. In the second step, the correction step, measurements of the environment features are taken into account to compute the posteriori state which turns to be the current state estimation. While the normal Kalman filter can only be used on handling the linear problems, the extended Kalman filter is aimed at dealing with the non-linear models. The EKF additionally linearizes the model in the prediction step via calculating the partial derivatives of the state variables.

In the EKF, noises of process and the sensory resources are assumed as zero
mean Gaussian distributions with independent error covariance. However, the well
defined models of the error covariance are seldom achieved. If the estimated process
and measurement noises are not precisely modelled, the Kalman filter would diverge
or at best converge to a large bound (Sasiadek and Hartana, 2002). In this thesis,
artificial intelligence is employed to aid the extended Kalman filter to model the
system, especially the error covariance.

There are two ways to solve the divergence problems due to the modelling inac-
curacy: (1) to add the undefined state, but it increases complexity to the filter and
the added variables can never be sure as the suspected unstable states; and (2) to
adjust noises of the system, it guarantees that the Kalman filter is driven by zero
mean white noise, and prevents the filter from disregarding new measurement.

In this thesis, in order to tune the noise in the filter, the artificial intelligence
algorithms are employed. Fuzzy logic and neural network are the methods introduced
to accomplish the task of weighting the EKF and prevent the Kalman filter from
divergence.

### 3.2.2 Fuzzy Logic-aided Extended Kalman Filters

Fuzzy logic has been widely applied to the improvement of the EKF. The works
which use Takagi-Sugeno fuzzy structure (TSFS) based fuzzy model to represent
the nonlinear process and measurement models of the vehicle are presented. Either
kinematic model or the measurement model of a nonholonomic mobile robot can be
simulated by a set of pseudo linear models (Carrasco and Aldo, 2004; Watanabe et
al., 2008). A set of linear equations are used to represent all the kinematic models
and observation models of the robot. Each linear model is attached to a KF instead
of a EKF. Then, the TSFS is used to create the desired nonlinear model via multiple
linear equations integration. The process which is composed of a set of linear models
is used to ensure the error covariance matrix of KF rather than the nonlinear ones
of the EKF. By utilizing this fuzzy design to solve the robot localization issue, the
robot local state estimations are combined to form the global state estimation of
the system (Simon, 2003). For each time instance $t$, each position estimation of the
Figure 3.5: Membership function of a process state. (a) Membership function of $x$ position; (b) Member function of $y$ position. (Modified from Carrasco & Aldo (2004))

The mobile robot from the linear Kalman filter is fuzzified into the TSFS via the same membership functions to build the fuzzy model of the robot. The output of the whole mechanism is the final estimation at the current time. In addition, Chandima proposed a scheme that uses TSFS pseudo linear models to formulate the model of the whole system, including both the observation model and the process model (Pathiranage et al., 2010).

However, the main drawback of the process simulation method is that the linearization of the observation model and the kinematic model cannot be carried out together. Moreover, the linearization of the model via a set of linear equations will add the computational complexity to the system, because each of the linear equations has to compute their own KF process.

Gaussian distributions are not always the most fit noise model for the mobile robot. Several articles come from the innovation that possibility distributions are used to substitute Gaussian distributions for the filter (Matia et al., 2004; Tsalatsanis et al., 2007; Matia et al., 2006). Normally, noise of the model and measurement should be
white Gaussian distribution, however, propagation of Gaussian distributions through non-linear equations may cause the error accumulation. Some characteristics of this approach are that uncertainty does not need to be symmetric, and that a wide region of possible values for the expectations is allowed (Matia et al., 2006). Then the EKF uses the updated noise distributions from the fuzzy controllers to fuse the observation acquired from the sensors to provide a more accurate estimate of the robot position (Tsalatsanis et al., 2007). The distribution replacement method deduces the dependency of a 100% percent accurate model, but also produces less errors in propagating than the ones in Gaussian distribution. However, the shortcoming of possibility distribution is that usage of it is limited to the landmark measurement. Only the noise from the landmark is appropriate to be simulated as the possibility distribution rather than all the other types of sensors.

Normally, the noise is defined priori to the filtering process and will be constant during the process. For a mobile robot, the process noise and the measurement noise are environmentally dependent which are hard to be predicted. Therefore, it is inappropriate to set unchanged noise variance to compute the KF. Several papers have proposed an adaptive way to tune the filter. The pre-requisite of this method is to figure the residuals or the innovation of the filter which indicates the discrepancy between the actual measurement and the one of the filter. The hypothesis of the tuning method is that deviation of the innovation vector from zero by more than a certain value suggests reduction in performance of the filter that can be corrected with a scale factor applied to the covariance matrix (Wang, 2004). The fuzzy inference system uses the innovation as input and scale factor as the output. For a finer gradation of the scale factor, fuzzy logic is adopted to use rules to map from inputs to outputs (Reina et al., 2007). The Kalman filter can be adjusted in different ways due to different parameters of the filter: tuning the error propagation of the process (Reina et al., 2007), online adjusting the measurement matrix $R$ (Chatterjee, 2009), fitting the process covariance $Q$ (Ip et al, 2010; Jetto et al, 1999), adapting both $R$ and $Q$ together via the same scaling factor (Sasiadek et al., 2000). However, none of them are going to design a structure that separately adjusts process error covariance.
and observation error covariance $R$ at the same time.

In this thesis, these two parts will be tuned simultaneously to correct the error of the model, aim at improving the performance of the Kalman filter and prevent the divergence of it.

### 3.2.3 Neural Network-aided Extended Kalman Filters

To define appropriate statistical noise to both the process model and the observation model is a key problem associated with the Kalman filter. Fuzzy logic is a method to accomplish the error covariance tuning to reflect the actual static noise of the robot; however, there are some shortcomings. The computation of fuzzification according to the fuzzy rules and the defuzzification process need to be repeated at each sampling instance, resulting in an obvious time delay. In addition, fuzzy logic lacks learning capability. The robot cannot remember the operated environment no matter how many times it has worked. Thus neural network with learning ability is another good choice for the adjustable Kalman filter.

Neural network is not the first time to be used on aiding the EKF. However, algorithms concerned with the robot localization issue are limited. The neural network can be performed from several aspects to aid the EKF. In the paper (Kramer et al., 2007; Zou et al., 2005), the presented algorithm combining NN and KF employ NN to calibrate the sensor measurement first, and then take the calibrated measurement into the Kalman filter to improve the performance. Another way NN is employment to approximate the uncertainty of the model. Since the EKF is based on the white noise assumption, if there are coloured noise or systematic bias error in the system, the EKF inevitably diverges (Choi et al., 2007). In the method presented by Choi (Choi et al., 2007; Choi and Lee, 2010) and Kang (Kang et al., 2010; Wang et al., 2009), a neural network, adaptive to the changes of environmental information flowing through the network during the process, can be combined with an EKF to compensate for some of the disadvantages of an EKF, which represents the state uncertainty by its approximate mean and variance, and has biased systematic errors even after appropriate compensation in real situations (Choi and Lee, 2010). A neural
network is employed to predict the next sampling time GPS output and a new Kalman filter based data fusion method is proposed to do the navigation data fusion with GPS/DR system (Wang et al., 2009). Neural network based extended Kalman filter can capture the undefined dynamics after the training, and adapt to the changed conditions intelligently. However, a drawback of all the above algorithms is that the NN is not operated on the EKF directly. These algorithms take NN to simulate the error or calibrate the sensor. The employed external sensor is limited by the GPS, which can provide the measurement of the robot position in a large working area, but in most cases, especially the indoor environment, GPS is not available for the light and flexible mobile robot.

In this thesis, a back propagation (BP) NN is presented to tune the EKF. The presented NN is able to learn the working environment via the sensory readings from the ultrasonic sensors or the infrared sensors. After training, the NN has the capability of capturing the unpredicted errors and minimizes the affects from the errors.

### 3.3 Concluding Remarks

From the literature review, the new trend of the robot localization is to fuse the absolute and relative methods to compensate the shortcomings of each other. The Kalman filter is the most popular method to do the fusion. However, the main disadvantage of it is that the process models and observation models are not easy to predict or define. Thus artificial intelligence is taken to assist the Kalman filter to add the flexibility of the Kalman filter of handling unpredicted errors to prevent the filter divergence. With the comparison to other AI algorithms, such as fuzzy logic, neural network has its own advantage of learning ability. It can effectively reduce the redundancy of repeating the adaptive steps as fuzzy logic does and also keep the accuracy of localization.
Chapter 4

Fuzzy Logic-aided Extended Kalman Filter

A precise localization system of the robot with respect to the global coordinate is the fundamental requirement of the robot navigation. The relocation of the mobile robot offers the estimation of the robot location and orientation at each sampling step along with its trajectory by using the current sensor measurements and the priori knowledge of the operating environment. Either the typical method, dead reckoning or the absolute methods have their own drawbacks, so that they cannot accurately reflect the robot actual position. The current trend is to fuse the absolute method and the dead reckoning (Pathiranage et al., 2010; Teslic et al., 2010; Chen et al., 2009; Tran et al., 2008; Santana et al., 2008). In this chapter, the artificial intelligence aided extended Kalman filter is presented, which is employed to tune the Kalman filter.

4.1 Robot Modelling and Sensors

The localization of the robot in the working environment depends on the employed robot model and sensors.
4.1.1 Robot Modelling

The localization of the mobile robot in the two-dimensional space is the knowledge of the global coordinates $X$ and $Y$ of the central point $(x, y)$ of the robot, and of the angle $\theta$ which is between the heading direction of the robot and the $X$-coordinate. $(x, y)$ is called location in this thesis. The used nonholonomic mobile robot is restricted to three degrees of freedom which describe the unique status of a mobile robot in the workspace. The kinematic model of the robot is illustrated in Figure 4.1.

![Figure 4.1: The kinematic model of a mobile robot.](image)

The position of the mobile robot in the global frame is noted as $X = [x \ y \ \theta]^T$. The kinematic constraint of the robot can be expressed as

$$-\dot{x} \sin \theta + \dot{y} \cos \theta = 0.$$  (4.1)

The used mobile robot is a differential driven model and its simplified kinematic model is defined as

$$
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} =
\begin{bmatrix}
\cos \theta & 0 \\
\sin \theta & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix},
$$  (4.2)

where $v$ and $\omega$ are the linear and angular velocity of the robot, respectively. Velocities $v$ and $\omega$ are obtained from the encoders mounted on two independent driving wheels of the robot as

$$v = \frac{v_L + v_R}{2}, \quad \omega = \frac{v_R - v_L}{d},$$  (4.3)
where $d$ is the distance between the two wheels, and $v_L$ and $v_R$ are the velocities of the left and right wheels, respectively.

According to the dead reckoning, estimation of the current position is based upon the previously determined position and the known or estimated velocities over elapsed time (Kim et al., 2011). The measured linear and angular velocities $v$ and $\omega$ are assumed to be constant over the sampling time $\Delta t$. The increment of the linear translation distance $\Delta S$ and the increment of the angle $\Delta \theta$ of the robot within the sampling time $\Delta t$ are

$$
\Delta S = v \Delta t, \quad (4.4)
$$

$$
\Delta \theta = \omega \Delta t, \quad (4.5)
$$

respectively. According to the odometry method, the current position of the robot $X(t + \Delta t)$ is based on the incremental linear and angular distance

$$
x(t + \Delta t) = x(t) + \Delta S \cos \theta, \quad (4.6)
$$

$$
y(t + \Delta t) = y(t) + \Delta S \sin \theta, \quad (4.7)
$$

$$
\theta(t + \Delta t) = \theta(t) + \Delta \theta. \quad (4.8)
$$

The estimation of the robot localization according to the odometry is shown in Figure 4.2.

Since new positions are calculated from previous positions, the error of the process is accumulated. The error of the position grows with the time. The inertial error of the dead reckoning will accumulate over the time and there is no bound of the final error. Thus for the long and wind path, the performance of the robot would be acted on one way, and the behaviour of the odometry estimation would be on the other way, which is shown in Figure 4.3. In the figure, there is an obvious difference between the actual position of the robot and the dead reckoning estimation. However, in a short period of time, the rolling conditions are assumed to be ideal, so that little errors are existed in the odometry output. For short travelled distance, the dead reckoning is assumed to be precise.
Figure 4.2: The relative localization estimation of the mobile robot.

Figure 4.3: Difference between dead reckoning and the robot actual position.
4.1.2 Robot Sensors

The equipped sensors of the robot are always composed of internal sensors and external sensors. The internal sensors are employed to measure internal variables produced by the robot itself, like velocities or the torque. The external sensors interact with the operating area to measure the features of the environment. In my approach, the sensory readings from these two types of sensors are fused to estimate the robot position. The experimental robot in this study, Dr. Robot Sentinel, is composed of two integrated 800 count per cycle optical encoders placed on the driving wheels (internal sensors), on-board DUR5200 ultrasonic range sensors and GP2Y0A21YK sharp infrared distance measuring sensors (external sensors).

Optical encoders, which are the widely used internal sensors, are mounted on the driving wheels of the robot to measure their rotations. Its drawback is that the measuring errors grow, like the drift, the bias or the slippage, are inevitable. These errors will accumulate over time as an integration error. However, for the short-distance travel, it is reasonable to assume that only a small error occurs. With this assumption, the error can be eliminated at each localization sampling instant (Xu and Collins, 2009).

The errors of the encoder are assumed to be zero mean white noise which is uncorrelated with the previous or next unit of travel. This leads to a reasonable assumption that the variance of each wheel at a unit of travel is proportional to the travelled distance

\[ \sigma_l^2 = k_l^2|d_l|, \quad \sigma_r^2 = k_r^2|d_r|, \]  

where \(d_l\) and \(d_r\) are the distances travelled by the left and right wheels, respectively; and \(k_l^2\) and \(k_r^2\) are constants with unit \(m^{1/2}\) (Chong and Kleeman, 1997).

The measurement of the environment from the sonar sensor is related to the sensor position. It is necessary to transmit the coordinate of the sensor from the local frame to the global frame. As shown in Figure 4.4, \((x'_i, y'_i, \theta'_i)\) denotes the position of \(i\)-th sensor with respect to the local coordinate \((X', 0', Y')\) fixed on the robot. The position of it at the sampling time \(t\) referred to the global frame \((X, 0, Y)\) is defined
as
\[
x_i(t) = x(t) + x_i'(t) \cos(\theta(t)) - y_i'(t) \sin(\theta(t)) \\
y_i(t) = y(t) + x_i'(t) \sin(\theta(t)) + y_i'(t) \cos(\theta(t)) \\
\theta_i(t) = \theta(t) + \theta_i'
\] (4.10)

The angular difference between each two sensors is approximately 36°, so that all the faced 180° can be fully covered.

Since the surrounding environment of the robot is known, the walls and the obstacles in the indoor environment can be represented by a proper set of planes orthogonal to the plane \(XY\) with respect to the global coordinate system. As shown in Figure 3.5, the plane \(P_j, j = 1, 2, \ldots, n_p\), where \(n_p\) is the number of planes existed in the indoor environment, is represented by geometric parameters \(P^r_j, P^n_j, P^v_j\) (Sariel and Erdogan, 2008). The variable \(P^r_j\) is the normal distance of the \(j\)-th plane from the origin \(O\); \(P^n_j\) is the angle between the normal line of the \(j\)-th plane and the \(X\) coordinate; and \(P^v_j\) is a binary variable, which defines the face of the \(j\)-th plane reflecting the signal beam (Wei and Yang, 2011).

Although ultrasonic sensors embedded on the robot provide sensory data individually, the measurement functions of them are the same. The measurement function \(d_i\) from the \(i\)-th sonar to the \(j\)-th plane \(P^j\) is given as
\[
d_i = P^v_j (P^r_j - x_i \cos P^n_j - y_i \sin P^n_j), \quad i = 1, 2, 3, \ldots, n,
\] (4.13)
where \(x_i\) and \(y_i\) are the location of the \(i\)-th sensor with respect to the global coordinate, and \(n\) is the number of the sensors employed (Sharma and Ghose, 2009). In the project, the number of the applied sensors is \(n = 5\). The measurement of \(d_i\) is shown in Figure 4.5.

Since the travelling of the ultrasonic signal is a cone shape, the angle \(P^n_j\) should belong to a range
\[
P^n_j \in [\theta_i(t) - \frac{\delta}{2}, \theta_i(t) + \frac{\delta}{2}].
\]
In fact, the real measurement from the ultrasonic sensor along the cone edge needs the priori knowledge of the environment, like the roughness of the walls.
Figure 4.4: Sensor locations on the experimental robot and its diagram. (a) Sensor positions on the experimental robot; (b) Schematic diagram of the robot with sensors.
4.2 The Proposed Method

In 1960, Kalman published his famous paper describing a recursive solution to the linear discrete-data filtering problem (Kalman, 1960). Since that, the Kalman filter has become a tool that can estimate the variables of a wide range of process.

4.2.1 Extended Kalman Filter

In mathematical terms, we would say that a Kalman filter estimates the states of a linear system. It is also the one that minimizes the variance of the estimation error (Simon, 2007). For linear dynamic systems with process noise and the measurement noise that are white and uncorrelated, the Kalman filter is known to be an optimal estimator (Gelb, 1974). In order to use a Kalman filter to remove noise from a signal, the measuring process should be described with the state equation

\[ X(t + 1) = f(X(t), u(t), w(t)), \]  

(4.14)
and the measurement equation

\[ Z(t + 1) = h(X(t), r(t)), \]  

(4.15)

where \( X \) is the system state vector; \( f \) is the nonlinear state function at the current step \( t + 1 \) related to the previous time step \( t \); \( u \) is the robot driven input; \( w \) and \( r \) are the process and measurement noise, respectively, which are assumed to be zero mean white noise with covariance matrix; \( Z \) is a measurement vector; and \( h \) is the nonlinear measurement function of the system.

For the mobile robot, with its state vector \( X = [x, y, \theta]^T \), according to Eqs. (4.2)-(4.8), the nonlinear dynamic model can be defined as

\[
f(X(t), u(t), w(t)) = \begin{bmatrix} x(t-1) \\ y(t-1) \\ \theta(t-1) \end{bmatrix} + \begin{bmatrix} \cos \theta(t-1) & 0 \\ \sin \theta(t-1) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} + w(t). \tag{4.16}\]

The measurement matrix \( h(X(t)) \) is consisted of two sub-vectors: the odometric measurements from Eqs. (4.6)-(4.8) and the sonar measurement function of Eq. (4.13), which is given as

\[
h(X(t), r(t)) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \\ d_1(t) \\ d_2(t) \\ \vdots \\ d_n(t) \end{bmatrix} + r(t), \tag{4.17}\]

where \( n \) is the number of the sensors employed on the robot. Its dimension is depended on the number of sensors used in the system.

The Kalman filter estimates the process by two steps: the filter takes advantage of the process function and error covariance to predict the current state of the process as the priori estimation and then incorporates the newly measurements into the priori state to correct it and obtain an improved state estimation. The prediction step
can be also thought as “time update”, and the correction step can be thought of as “measurement update”.

The Jacobian equations that linearize the process function \( f(X(t), u(t), w(t)) \) and the measurement function \( h(X(t), r(t)) \) are taken place first.

- Jacobian matrix of partial derivatives of \( f \) with respect to \( X \)
  \[
  A[i, j] = \frac{\partial f[i](X(t), u(t), 0)}{\partial X[j]} \tag{4.18}
  \]
  \[
  A[i, j] = \begin{bmatrix}
  1 & 0 & -T \sin \theta \\
  0 & 1 & T \cos \theta \\
  0 & 0 & 1
  \end{bmatrix} . \tag{4.19}
  \]

- Jacobian matrix of partial derivatives of \( f \) with respect to \( w \)
  \[
  W[i, j] = \frac{\partial f[i](X(t), u(t), 0)}{\partial w[j]} . \tag{4.20}
  \]

Thus the error covariance \( Q \) is

\[
Q = \sigma^2 \begin{bmatrix}
T + \frac{T^3}{3} v(t)^2 \sin^2 \theta(t) & -\frac{T^3}{3} v(t)^2 \sin \theta(t) \cos \theta(t) & -\frac{T^2}{2} v(t) \sin \theta(t) \\
-\frac{T^3}{3} v(t)^2 \sin \theta(t) \cos \theta(t) & T + \frac{T^3}{3} v(t)^2 \cos^2 \theta(t) & -\frac{T^2}{2} v(t) \cos \theta(t) \\
-\frac{T^2}{2} v(t)^2 \sin \theta(t) & -\frac{T^2}{2} v(t)^2 \cos \theta(t) & T
\end{bmatrix} . \tag{4.21}
\]

- Jacobian matrix of partial derivatives of \( h \) with respect to \( X \)
  \[
  H[i, j] = \frac{\partial h[i](X(t), 0)}{\partial X[j]} . \tag{4.22}
  \]
Thus the measurement function of the filter is

\[
H[i, j] = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
-P_v^i \cos P_n^i & -P_v^i \sin P_n^i & x_i' \sin(\theta(t) - P_n^i) - y_i' \cos(\theta(t) - P_n^i)
\end{bmatrix},
\]

\[i = 1, 2, 3, \ldots, n.\] (4.23)

At the beginning of the EKF estimation, the filter needs to be initialized for \(t = 0\). The initial states are given as

\[
\hat{X}(0) = E[X(0)],
\]

\[
P(0) = E[(X(0) - E[X(0)])(X(0) - X(0))^T],
\]

where the \(\hat{X}\) is the state of the process and \(P\) is the error covariance of the process. Then the state is estimated at each sampling time according to the following two groups of equations.

- **Group one: EKF time update equations**

  \[
  \hat{X}(t + 1)^- = f(\hat{X}(t), u(t), 0),
  \]

  \[
P(t + 1)^- = A(t + 1)P(t)A(t + 1)^T + Q(t),
  \]

  where \(\hat{X}(t + 1)^-\) and \(P(t + 1)^-\) are the priori state of the process and the error covariance at current time \((t + 1)\), respectively. The time update equations project the state \(\hat{X}\) and the error covariance \(P\) from the previous time step \(t\) to the current time step \(t + 1\). Function \(f\) comes from Eq. (4.16), and \(A(t + 1)\) is the Jacobian matrix of the process at step \(t + 1\).

- **Group two: EKF measurement update equations**

  \[
  K(t + 1) = P(t + 1)^-H(t + 1)^T[H(t + 1)^-P(t + 1)^-H(t + 1)^T + R(t)]^{-1}.
  \]

  Then the Kalman gain \(K(t + 1)\) will be used to correct the priori states

  \[
  \hat{X}(t + 1) = \hat{X}(t + 1)^- + K(t + 1)[Z(t + 1) - H\hat{X}(t + 1)^-],
  \]

  \[
P(t + 1) = [I - K(t + 1)H(t + 1)]P(t + 1)^-.
  \]
In measurement update, the first task is to compute the Kalman gain $K$ as in Eq. (4.28). This gain indicates how reliable the measurement is. Then the posteriori state of the process is generated by integrating the measurements as in Eq. (4.29). The final step is to update the error covariance according to Eq. (4.30). Again $H$ is the Jacobian matrix of the measurement at current step $t + 1$, and $R$ is the measurement noise covariance.

The EKF can be implemented to fuse signals in several schemes (Green and Sasiadek, 1998). The first method is called direct pre-filtering method, which filters all the measurement signals via EKF prior to comparison. The differences from these EKFs are then used to update the priori prediction as the posteriori state. The second scheme, which is called direct state space EKF, is by applying the state space model into the filter. The measurement signals are combined before feed into the EKF. The third method is indirect feed forward EKF. In this method, the signals are compared before feed into the EKF. The EKF estimated error is then incorporated into one of the measurements. All the three schemes are shown in Figure 4.7. In the thesis, the second layout, that is the direct EKF method, is used. The measurement signals, which come from the odometry and sonar sensors, are combined into one measurement vector and then fed into the EKF.

For many applications, the statistic noise $Q$ and $R$ are predefined and will maintain unchanged during the whole process. While set different values for errors, the performance of the filter will be completely different. For example, in Figure 4.8 shows the performance of the filter at various $R$ value (Welch and Bishop, 2006). While the measurement error $R$ is increased from 0.01 to 1, the measurement is “trusted" less, so that the filter is affected less by the noisy measurement. The filter converges to the actual position slower. While $R$ is decreased from 1 to 0.0001, the filter trusts the measurement more. It converges to the actual position faster, but oscillates more frequently because of the noise. From the figure, it is easy to draw the conclusion that the performance of the EKF is largely dependent on the predefined model.

Commonly, this priori information is based on the certain knowledge about the observation process. However, if priori information is not sufficient to define the real
Figure 4.7: The EKF schemes. (a) Direct Pre-filtering scheme; (b) Direct EKF scheme; (c) Indirect feed forward EKF scheme.
Figure 4.8: The EKF performance at various $R$ values. (a) $R$ is 0.01; (b) $R$ is 1; (c) $R$ is 0.0001.
statistic model, the Kalman filter may result in unreliable estimation, even leads to filter divergence (Reina et al., 2007). If the process that feeds the information into the Kalman filter is behaved differently against the Kalman filter, it causes the filter to continue to act in the wrong way. Therefore, a system with constant noise variances is impossible to satisfy all the situations and also difficult to define.

The Kalman filtering estimation at a given time \( t \) can be considered as a weighted combination between the new measurement and the predicted state space vector according to the dynamic model and all previous measurements. The term \( Z(t) - H\hat{X}(t)^- \) is called the “measurement innovation or residual”, which is referred as the discrepancy between the output of the actual measurement \( Z(t) \) and the measurement prediction \( H\hat{X}(t)^- \). Under an assumption of the completely accurate vehicle model, the statistical properties of the innovation are assumed to be similar to the theoretical prediction. If too many weights (Kalman gain) are assigned to the dynamic model, which means the predicted measurement is trusted more and the sensory reading from the real measurement is trusted less, it could lead to the poor position estimation and even divergence. The innovation based detection is employed to determine whether the Kalman filter is needed to be corrected by the mean of adjusting the error covariance \( Q(t) \) and \( R(t) \).

In this chapter, the fuzzy logic is proposed to adjust the state error covariance \( Q \) and the measurement error covariance \( R \) based on the equations

\[
Q(t) = \eta_Q(t)Q_0, \tag{4.31}
\]
\[
R(t) = \eta_R(t)R_0, \tag{4.32}
\]

where \( \eta_R, \eta_Q \) are the scaling factors; and \( R_0 \) and \( Q_0 \) are the initial value of \( Q \) and \( R \), respectively. The tuning process of \( Q(t) \) and \( R(t) \) is achieved via \( \eta_R, \eta_Q \) to tune its initial value. The scale factors \( \eta_R, \eta_Q \) are based on the size of the predicted residual. The predicted residual vector, or innovation vector, is expressed as

\[
\varepsilon(t) = Z(t) - H(t)\hat{X}(t)^-.
\]

The hypothesis is that the diversion of the innovation vector far from a certain range demonstrates a reduction of the filter performance. The predicted theoretical
range is from the equation
\[ \hat{r}(t) = H(t)P(t)H(t)^T + R(t), \] (4.33)
where error covariance \( P(t) \) is regarding the state noise \( Q(t) \). The good estimated \( Q(t) \) and \( R(t) \) could produce innovation process consistent within their theoretical range. If the filter diverges, the residual will no longer be zero mean. The mismatch between the actual innovation process and its theoretical value is defined as
\[ \xi(t) = \varepsilon(t)\varepsilon(t)^T, \] (4.34)
\[ \alpha(t) = \frac{1}{n} \sum_{i=1}^{n} \hat{r}_{(i,i)}(t), \] (4.35)
where \( n \) is the dimension of the matrix, \( \alpha(t) \) are the average of the diagonal elements.

The working mechanism of the proposed fuzzy logic tuned EKF is shown in Figure 4.9.

A fuzzy system is employed to minimize the mismatch given in Eq. (4.35). The overall fuzzy system employs a one-input-two-outputs fuzzy system. The fuzzy system utilize the average of the mismatch between the actual value and the theoretical value, \( \alpha(t) \), as input and the outputs are the scaling factors, an adaptation recommended for the corresponding mismatch. The model of the fuzzy logic controller to tune the EKF is shown in Figure 4.10.

### 4.2.2 Fuzzy Logic-aided EKF

The design of the fuzzy logic controller is composed of three parts: fuzzification, inference mechanism, and defuzzification.

**Fuzzification**

The fuzzification process involves mapping the crisp input values into the linguistic fuzzy terms with the membership functions (Cao et al., 2008). In this section, triangle and trapezoidal functions are implemented to represent the fuzzy membership functions. The input and output membership functions are shown in Figures 4.11 and 4.12, respectively.
Figure 4.9: The flow diagram of the proposed EKF tuned by fuzzy logic.

Figure 4.10: The employed fuzzy architecture.
Figure 4.11: Input membership function.

Figure 4.12: Output membership functions. (a) Output of scaling factor $\eta_R(t)$; (b) Output of scaling factor $\eta_Q(t)$. 
The input membership functions are defined as five levels according to the value of $\alpha$: “NZ” (near to zero), which means the actual innovation vector is even smaller than the theoretical range and very near to zero; “S” (small), which indicates that the observed innovation vector is close to but still smaller than the theoretical prediction; “M” (medium), which illustrates that the observed innovation vector is close to but a little larger than the theoretical value; “L” (large), which demonstrates that the observed innovation vector is far beyond the theoretical estimation.

The output membership functions include 5 states: “NZ” (near to 0), the output scaling factors are close to zero; “N1” (near to 1), the scaling factors are around one; “LL1” (little larger than 1), the scaling factors are a little larger than 1; and “L” (large), the scaling factors are larger than one. These values are all obtained from the experiments.

**Inference Mechanism**

The fuzzy inference engine is a Mamdani type inference which is based on a set of “IF-THEN“ rules. By selecting appropriate factors, the fuzzy logic controller will adapt the Kalman filter and try to keep the innovation sequence acting as zero mean white noise.

When the observed residual samples are moving away from the innovation theoretical prediction, the Kalman filter is becoming unstable. In this case,

- IF innovation process samples (actual) are observed very near to zero, which indicates that they lie inside the range with a smaller $Q$ and $R$, THEN reduce the $Q$ and $R$;

- IF the value of the observed sample is neither beyond the range predicted by the theory nor very near zero, which means the current $Q$ and $R$ are appropriate, THEN leave $Q$ and $R$ almost unchanged;

- IF the observed samples is so far from zero as to exceed the predicted range, THEN $Q$ and $R$ should be increased.
Table 4.1: Rule table for the scaling factors.

<table>
<thead>
<tr>
<th>α</th>
<th>η_R</th>
<th>η_Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>NZ</td>
<td>NZ</td>
<td>NZ</td>
</tr>
<tr>
<td>S</td>
<td>N1</td>
<td>N1</td>
</tr>
<tr>
<td>M</td>
<td>LL1</td>
<td>LL1</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
</tbody>
</table>

The fuzzy logic rule table is illustrated in the Table 4.1.

With the current \( Q(t) \) and \( R(t) \) tuned at the current sampling step according to the adaption mechanism, as the consequence, the error covariance \( P(t) \) and the gain \( K(t) \) are also adapted. While large factors are applied, it means that process noise \( Q \) and the measurement noise \( R \) are added. But when \( \eta_R \) is very large and \( R \) tends to be very large, that makes the Kalman gain \( K = 0 \) which means the measurement cannot be count any more.

### Defuzzification

The defuzzification procedure maps the fuzzy output from the inference mechanism to a crisp signal (Cao et al., 2008). There are many methods that can be used to convert the conclusions of the inference mechanism into an actual output of the fuzzy controller (Kermiche et al., 2007). In this system, the centroid average method is implemented which is perhaps the most popular method and returns the centre of the area under the curve.

### 4.3 Simulation Results

The proposed method for estimating the position of the driving robot by fusing the dead reckoning data with ultrasonic observations has been implemented and tested via computational simulations.

In the simulation, the robot is setting to track a cycle. There is no control system
worked on the robot, because the goal of the experiment is to test the localization system but not the control part. Thus, because of the noise existence and no control adjustment, the motion of the robot cannot be a perfect cycle. The performance of the robot tracking will reflect the efficiency of the localization method. During the simulation, the precise localization system makes the estimated position close to the actual position.

The simulation program runs in the MATLAB. The cycle path is with a radium of 20 m. The initial linear velocity and angular velocity are set as 0.35 m/s and 1°/s. Noise of the velocities are assumed as zero mean Gaussian distribution with \( \sigma_d = 0.1 \text{ m/s} \) and \( \sigma_\theta = 0.5^\circ \). The sampling period \( \Delta t = 0.1 \text{ s} \). The environment is relative large about 50 m x 50 m. There are 5 ultrasonic sensors being used in the simulation.

### 4.3.1 Typical EKF

Firstly, the efficiency of the typical EKF is tested to see how it performed in the localization issue. The results of the typical EKF algorithm illustrated in Figures 4.13-4.16, which are compared with the dead reckoning method, are the robot moving trajectory within the environment, the position estimation on X coordinate, the position estimation on Y coordinate and the orientation value of the robot, respectively. With the comparison to dead reckoning, it is easier to figure out that how the EKF improves the accuracy of the localization.

In Figure 4.13, the performance of the dead reckoning is limited to error accumulation. At the beginning of the process, the estimation of the dead reckoning is acceptable which keeps close to the actual position of the robot. With the time elapse, since the previous accumulated error never be eliminated from the estimation, the performance of the dead reckoning is diverged gradually. Finally, it goes far away from the robot actual position. However, during the whole process, the EKF almost precisely estimates the actual position of the robot. The estimation of the EKF and the “actual position” is almost overlapped from the beginning till the end. No matter how the robot is affected by the process uncertainty, the EKF still be able to reduce the noise and estimate its current position. From the figures which illustrate
the value of $x$ coordinate, $y$ coordinates and the orientation, this conclusion will be verified again.

Errors between the robot actual position and the estimated position is shown in Figures 4.14-4.16. The error reflects the accuracy of the state estimation of the approach. In these three figures, error of the dead reckoning is no bound, because priori errors are accumulated into the estimation at each sampling time. The EKF is able to eliminate the error accumulation and also reduce the signal errors. Errors of the EKF estimation are following the Gaussian distribution with the boundary of $3\sigma$. However, there is divergence of the EKF occurred around 34 s till the end of the process. In the simulation, an abrupt error intend to be involved around the 35 s. Divergence is the result of abrupt error changes which are beyond the predefined error covariance, and will cause the filter deviated from the robot actual position. Although divergence appeared once per 10-20 times experiments, it leads the filter to go to crash each time. The fuzzy logic-aided EKF with the capability of tuning the error covariance is more flexible while face the sudden uncertainty.
Figure 4.14: Estimation of $X$ coordinate of the robot centre. (a) $x$ coordinate of the algorithm; (b) Error of $x$ between the actual position and the estimated position of the robot centre.
Figure 4.15: Estimation of $Y$ coordinate of the robot centre. (a) $y$ coordinate of the algorithm; (b) Error of $y$ between the actual position and the estimated position of the robot centre.
Figure 4.16: Estimation of the robot orientation. (a) orientation of the robot; (b) Errors between the robot actual orientation and the estimated orientation of the robot.
4.3.2 Fuzzy Logic-aided EKF

To check the estimation given by the FL-aided EKF, the localization and the error of the localization are compared with the one given by a typical EKF under the same initial condition. The state of the robot is shown in the Figure 4.17.

![Figure 4.17: Robot localization in the environment.](image)

From Figures 4.18(b) and 4.19(b), errors come from the FL-aided EKF are smaller which guarantee the filter to work with less oscillation and reduce the chance of divergence occurrence. In addition, the smaller error indicates that the FL-aided EKF is working more precisely on localization.

In Figures 4.17-4.19, it is easy to figure out that the divergence of the typical EKF occurred because of the intentionally added error. However, fuzzy logic tuned EKF is able to overcome this abrupt changes during the tracking easily.

The adjusted results of scaling factors at each samples are displayed in Figure 4.24. With the adjustment of scaling factors $\eta_R(t)$ and $\eta_Q(t)$, the error covariance will always be kept similar to the actual error in order to prevent the behaviour difference between the filter and the robot.
Figure 4.18: Estimation of $X$ coordinate of the robot centre. (a) $x$ coordinate of the algorithm; (b) Error of $x$ between the actual position and the estimated position of the robot centre.
Figure 4.19: Estimation of $Y$ coordinate of the robot centre. (a) $y$ coordinate of the algorithm; (b) Error of $y$ between the actual position and the estimated position of the robot centre.
Figure 4.20: Estimation of the robot orientation. (a) orientation of the robot; (b) Error of orientation between the actual position and the estimation.
Figure 4.21: The linear velocity error of the robot centre.

Figure 4.22: The angular velocity error of the robot.
Figure 4.23: The internal and external sensor errors of the robot.

However, since the fuzzy logic-aided EKF lacks the ability of learning, no matter how many times the program have been run, the fuzzy controller still has to identify the working condition first and then operate to meet the actual condition. In some cases while the error covariance needs to be repeatedly adjusted for several times, the tuning process could be very time consuming. In this thesis, the fuzzy logic is employed to train the neural network online. Then the trained neural network can work independently to achieve the purpose of tuning the scaling factor.

4.4 Concluding Remarks

In this chapter, a fuzzy controller has been proposed to tune the Kalman filter. With the variance of the scaling factor, the filter is able to precisely estimate the position of the robot. However, with the lack of the ability of learning, the fuzzy logic employed to aid the EKF could be a time consuming process. Thus the fuzzy logic would train the neural networks to “memorize” the running process. To compare with the typical EKF with constant error covariance, the proposed approach not only effectively reduces the system error and uncertainty, but also prevents the filter from divergence.
Figure 4.24: Results of the regular factors. (a) Factor $\eta_R(t)$; (b) Factor $\eta_Q(t)$.
Chapter 5

Neural Network-aided Extended Kalman Filter

The Kalman filter is a well known technique for the state estimation. It is a recursive estimation procedure that uses sequential sets of measurements. In recent years Kalman filters based localization has became common practise in the robotics literature. However, its performance still can be improved via the artificial intelligence. In the previous chapter, the designed FL controller has been proved to be able to tune the error covariance of the filter. In this chapter, the previous outputs of the fuzzy controller are employed to train a back propagation NN. The well trained NN is operated on the EKF via the same principle which controls the filter by tuning its error covariance.

5.1 The Proposed Method

For the EKF, the reliable information of the error covariance $Q(t)$ and $R(t)$ are prerequisite. However, for most of the system, it is not easy to ensure the error covariance, since it is always varied with the time. In addition, constant $R$ and $Q$ of the normal EKF do not fit to the unpredicted error changes. Thus the on-line adjustment of the error covariance of the Kalman filter is proposed in order to improve the performance and suppress the divergence of the filter.
5.1.1 The Proposed Tuning Algorithm

The covariance matrix of the Kalman filter equation is according to the following patrons:

\[ R(t) = \eta_R(t)R_0, \]  \hspace{1cm} (5.1)
\[ Q(t) = \eta_Q(t)Q_0, \]  \hspace{1cm} (5.2)

where \( R_0 \) and \( Q_0 \) are the initial noise constant matrix, and \( \eta_R(t) \) and \( \eta_Q(t) \) are the adaptive scaling factors that are used to vary \( R(t) \) and \( Q(t) \), respectively. These two factors \( \eta_R(t) \) and \( \eta_Q(t) \) are both changed with time. The generation of the \( \eta_R(t) \) and \( \eta_Q(t) \) is the key of the covariance adjustment. When they are equal to 1, a typical EKF is obtained.

The logic of adaptive algorithm can be qualitatively described as follows. As the model of the EKF is completely modelled, the innovation process of the Kalman filter \( \varepsilon(t) = Z(t) - H(t)\hat{X}(t)\) should be a zero mean white noise process. The detection of the innovation process is used to indicate whether the model of the filter needs to be tuning. In this study, a match technique called degree of mismatch, which is defined as

\[ \text{DoM} = \frac{\text{diag}(\hat{\varepsilon}_r(t))}{\text{diag}(C_r(t))}, \]  \hspace{1cm} (5.3)

is employed to test the discrepancy between the theoretical value and the actual value of the innovation process. The variable \( \hat{\varepsilon}_r(t) \) is the theoretical value of the innovation process,

\[ \hat{\varepsilon}_r(t) = HP(t) - H(t)^T + R. \]  \hspace{1cm} (5.4)

Variable \( C_r(t) \) is the actual covariance of \( \varepsilon(t) \), which is approximated by its sample covariance,

\[ C_r(t) = (\varepsilon(t)\varepsilon(t)^T), \]  \hspace{1cm} (5.5)

where \( N \) is the number of the diagonal elements. The matrix variable DoM shows the consistency of the filter. When the static properties of the filter are accurate, the innovation sequence should be a zero mean white noise. The value of the diagonal elements of the DoM should be within the range predicted by the theory but not close
to zero. If not, there is the deviation of the filter occurred. Thus, if the actual value of innovation process has discrepancies with its theoretical value, then the error $R$ and $Q$ will be adjusted by the neural network. The NN is going to take the off-line learning and then to be used to generate scaling factors $\eta R(t)$ and $\eta Q(t)$ which are used to adjust the $R$ and $Q$.

5.1.2 The Employed Neural Network

For the neural network, with the balance of saving computation and achieving training target, one hidden layer is selected. At the discrepancy appears, the values of the diagonal elements of the DoM are taken as the inputs of the NN. Since there are five sensors employed in the filter, the number of input neurons are eight. The neurons of the hidden layer are selected from the repeating tests. The initial number of neurons of the hidden layer is three. In order to obtain the best performance of the NN, the number of neurons are increased gradually by trail and error. Finally the number of neurons are set at eight. The outputs are the scaling factors $\eta R(t)$ and $\eta Q(t)$. The layout of the neural network is shown in Figure 5.1. The adjustments of $R(t)$ and $Q(t)$ make the DoM back to the acceptable range to reduce the discrepancy. The architecture of NN employed in the project with only one hidden layer.

The outputs of hidden neurons (hidden layer) are

$$h_k = g\left(\sum_{i=0}^{8} w_{ki} x_i\right), \quad k = 1, 2, ..., 8,$$

(5.6)

where $x_0 \equiv 1$; $g()$ is the input/output function for the hidden layer; and $w_{ki}$ is the weight between the $i$-th input neuron and the $k$-th hidden neuron. For the hidden layer neuron, the input/output function is a sigmoid function as

$$g(x) = \frac{1}{1 + e^{-x}}.$$

(5.7)

The outputs of output neurons (output layer) are

$$y_j = f\left(\sum_{k=0}^{8} v_{jk} h_k\right), \quad j = 1, 2,$$

(5.8)
Figure 5.1: Architecture of the employed neural network.

Figure 5.2: Sigmoid output function.
where \( h_0 \equiv 1; \) \( v_{jk} \) is the weight between the \( k \)-th hidden neuron and the \( j \)-th output neuron; and \( f() \) is the activation function of the output layer. For the output layer, the activation function is a linear function,

\[
f(x) = x.
\] (5.9)

The learning algorithm taken here is the least mean square (LMS) learning which is guaranteed to decrease the error. The definition of the error is

\[
e_j = t_j - y_j, \quad j = 1, 2,
\] (5.10)

where \( e_j \) is the error of the \( j \)-th output neuron, and \( t_j \) and \( y_j \) are the target output and the computed output of the \( j \)-th output neuron, respectively.

The LMS function of the system is \( E = \sum_{j=1}^{2} \frac{1}{2} e_j^2 \). Then we bring Eq. (5.10) into the LMS function, so that

\[
E = \sum_{j=1}^{2} \frac{1}{2} (t_j - y_j)^2.
\] (5.11)

The weights between the input layer and the hidden layer are adjusted according to

\[
\Delta w_{ki} = -\eta \frac{\partial E}{\partial w_{ki}},
\] (5.12)

where \( \eta \) is the learning rate with \( \eta > 0 \). The weights between the hidden layer and the output layer are adjusted according to

\[
\Delta v_{jk} = -\eta \frac{\partial E}{\partial v_{jk}}.
\] (5.13)

Thus Eq. (5.11) is brought into Eqs (5.12) and (5.13) to replace the error \( E \). The the \( \Delta v_{jk} \) and \( \Delta w_{ki} \) are given as follows.

- For the weight \( v \)

\[
\Delta v_{jk} = -\eta \frac{1}{2} \sum_{j=1}^{2} (t_j - y_j)(0 - \frac{\partial y_j}{\partial v_{ki}}) = \eta h_k e_j,
\] (5.14)

\[
j = 1,2, \quad k = 0, 2, \ldots, 8.
\]
• For the weight $w$

$$
\Delta w_{ki} = -\eta \frac{1}{2} \sum_{j=1}^{2} 2(t_j - y_j)(0 - \frac{\partial y_j}{\partial w_{ki}}) = \eta h_k (1 - h_k) x_i e_h^k, \tag{5.15}
$$

$$
e_h^k = \sum_{j=1}^{2} e_j v_{jk}. \tag{5.16}
$$

The learning rate is so important in the algorithm. If it is chosen too small, the learning process will be too slow, while if it is too big, the divergent oscillation may be occurred.

### 5.2 Simulations

In this study, the training samples are selected from the experiment first. Then the offline neural network training is carried out in accordance with these data. Then the outputs of the neural network, which are the regulatory factors, are employed to adjust the error covariance matrix. The performance of the proposed NNEKF method is compared with the typical EKF to see how it improves the robot localization accuracy.

The simulation program ran in the MATLAB. The initial linear velocity and angular velocity are set as 1 $m/s$ and $10^\circ/s$. Noise of control input is assumed to be the zero mean Gaussian distribution with $\sigma_d = 0.1$ $m/s$ and $\sigma_\theta = 0.5^\circ$, respectively. The sampling period $\Delta t$ is equal to 0.1 $s$. The environment is relative large about 40 $m \times 65$ $m$. There are five ultrasonic sensors being used in the simulation.

The initial weight of each neuron is random. The learning rate is at 0.05 to avoid the local minimum solution. The initial values of the scaling factors $\eta_R(t)$, $\eta_Q(t)$ are both equal to 1. There are 1000 testing cases used to train the NN. To check the estimation given by the NN-aided EKF, the localization and its error are compared with the one given by the typical EKF under the same initial condition. The localization of the robot is shown in Figure 5.3. Although the localization of the NNEKF and the actual position of the robot are almost overlapped at most of the
time, there are still some difference existed. From Figure 5.4, Figure 5.5, and Figure 5.6, it is easy to figure out that with the tuning of the error covariance, the error between the actual position and the estimated localization is smaller. In addition, NNEKF tunes the error covariance to fit to the sudden changes of the error that prevent the filter from divergence.

![Robot localization in the environment.](image)

**Figure 5.3:** Robot localization in the environment.

The errors of the robot location are illustrated in Figures 5.4(b), 5.5(b) and 5.6(b). In these three figures, the errors from the NNEKF are obviously smaller than the errors from the typical EKF, which means the localization based on the NNEKF is more precise and smarter. From the errors of the system linear velocity in Figure 5.7, the errors of the angular velocity in Figure 5.8, and the sensors measurement uncertainties in Figure 5.9, it is easy to draw the conclusion that errors and uncertainty have been more effectively reduced in the NN-aided EKF.

Meanwhile the results of the adjusted regulatory factors at each time sample are displayed in Figure 5.10. With the change of the scaling factors $\eta_R(t)$ and $\eta_Q(t)$, the
Figure 5.4: Estimation of $X$ coordinate of the robot centre. (a) $x$ coordinate of the algorithms; (b) Errors of $x$ between the actual position and the estimated position of the robot centre.
Figure 5.5: Estimation of $Y$ coordinate of the robot centre. (a) $y$ coordinate of the algorithm; (b) Errors of $y$ between the actual position and the estimated position of the robot centre.
Figure 5.6: Robot orientation error between the actual position and estimation.

Figure 5.7: The linear velocity error of the robot centre.
Figure 5.8: The angular velocity error of the robot.

Figure 5.9: The internal and external sensor errors of the robot.
models of the error $Q$ and $R$ are always maintained to be close to the actual error to provide a simultaneous fixed error covariance for the EKF.

### 5.3 Concluding Remarks

In this thesis, the NN-aided EKF is proposed to improve the performance of the EKF. The performance of the filter is tracked by the technology called DoM to see if there is a divergence occurred. Since the NN-aided EKF has the capability of tuning the error covariance, the NN-aided EKF is able to handle the abrupt error changes and bring the deviated filter back to normal. The efficiency and effectiveness of the proposed approach are demonstrated by the simulation studies.
Figure 5.10: Results of the regular factors. (a) Factor $\eta_R(t)$; (b) Factor $\eta_Q(t)$. 
Chapter 6

Experiments on a Real Mobile Robot

In this chapter, the proposed algorithm is employed on a real mobile robot to testify its performance. The experiments are designed to demonstrate whether the robot can precisely localize itself using of the on-board sensors. Two experiments are operated on the robot, Dr. Robot Sentinel at the Advanced Robot and Intelligent System Lab.

6.1 Experiment Setup Results

For the robot navigation implementations, to estimate its state precisely is the pre-request. To achieve this purpose, the robot depends on the readings provided from internal and external sensors. The robot employed in this thesis is equipped with several kinds of sensors, which are listed in Table 6.1.

A overview of the experimented mobile robot is illustrated in Figure 6.1. The top view of the robot is shown in Figure 6.1(b), which presents the positions of the employed modules. In Figure 6.1(b), the three ultrasonic sensors are used to explore the area in front of the robot. In addition, two infrared sensors, sensor \( a \) and sensor \( b \), are responsible for detecting the side area. According to this disposal, the area toward the robot movement can be fully probed. In addition, two optical encoders
are fixed to the differential wheels to measure the velocities at each sampling time. The wheel velocity is computed according to

\[ v = \frac{E}{800} \cdot 2\pi r, \quad (6.1) \]

where \( r \) is radius of wheel which is 0.07 m of the robot and \( E \) is the encoder difference of the time. The linear velocity and angular velocity of the robot are calculating based on the equation

\[ v = \frac{v_L + v_R}{2}, \quad \omega = \frac{v_R - v_L}{d}, \quad (6.2) \]

where \( d \) is the distance between two wheels, and \( v_L \) and \( v_R \) are the velocities of the two wheels, respectively. In the experiments, \( d \) of the employed robot is 0.3 m.

The robot is working at the ARIS lab which is 10 m length and 7 m width. The layout of working environment is shown in Figure 6.2.

Normally, the localization system of the robot is responsible for providing the information for the control system as inputs. The control system then makes adjustment according to the coming inputs. In this case, the performance of the robot would be affected by both control system and localization system. Since the possible poor designed control module may affect the efficiency of the localization system, in the experiments, the control module and the localization module are the two systems working separately.
Figure 6.1: The experimental robot, Dr. Robot *Sentinel*. (a) The photo of the experimental robot; (b) The top view of the robot components employed in this study.
Figure 6.2: The experimental environment.
Table 6.2: Comparison between the actual position and the estimated position.

<table>
<thead>
<tr>
<th>Point</th>
<th>Desired value</th>
<th>Estimated value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>(0.35,0.70)</td>
<td>(0.38,0.75)</td>
<td>0.06</td>
</tr>
<tr>
<td>b</td>
<td>(0.72,0)</td>
<td>(0.77,0)</td>
<td>0.05</td>
</tr>
<tr>
<td>c</td>
<td>(1.21,-0.61)</td>
<td>(1.19,-0.64)</td>
<td>0.04</td>
</tr>
<tr>
<td>d</td>
<td>(1.70,0)</td>
<td>(1.67,-0.03)</td>
<td>0.04</td>
</tr>
</tbody>
</table>

6.1.1 Localization on the Sine Wave Trajectory

In this experiment, the robot is to track a sine wave trajectory. The more wind line the robot moves, the more errors might occur. In the experiment, the robot is assigned to follow the black line started from point (0,0) to the end point (1.70, 0). The path overview and process of the experiment are illustrated in Figure 6.3.

The velocities of the experiment are illustrated in the Figure 6.4. The five sensors measurements are shown in Figure 6.5.

To bring the linear velocity, angular velocity and the measurements of the robot into the proposed mechanism, which tunes the EKF via fuzzy logic and then trains the neural network to aid the EKF, the location estimations of the robot are coming out and illustrated in Figure 6.6.

Finally, the localization estimation of the robot in the experiment is illustrated in Figure 6.7.

It is impossible to compare all the estimated points along the trajectory with the expected ones. Only some representative points, point a, b, c and d, are selected to test the accuracy.

From the results listed in Table 6.2, it is easy to find out that the errors are small enough and no error accumulation during the whole process. Because of the involvement of the tuning mechanism, the errors are limited within a boundary, which means there is no divergence happened on the EKF.
Figure 6.3: The robot follows the sine wave trajectory. (a) The robot movement at various time instances on the trajectory; (b) The corresponding robot positions on the path of the panel (a).
Figure 6.4: The velocities of the robot to follow a sine wave trajectory. (a) Velocities of the two wheels; (b) Linear velocity of the robot centre; (c) Angular velocity of the robot.
6.1.2 Localization on a Rectangular Trajectory

Long term motion is the factor that causes the dead reckoning not suitable for the robot localization. The longer the robot moving, the more errors might be accumulated into the position estimation to cause an inaccurate estimation. In this experiment, the robot is ordered to follow a black line which is composed of 4 straight lines and 4 curves, like a rectangular. Since the working environment is too large to take a photograph, the tracking path of the robot is shown as a stretch in Figure 6.8(b) and the working process is illustrated in Figure 6.8(a).

The velocities of the robot left and right wheels are illustrated in the Figure 6.9(a). The linear and angular velocities of the robot are illustrated in Figures 6.9(b) and 6.9(c).

The position of the robot is estimated based on the velocities of the robot and five sensor measurements. The velocities and measurements are brought into the proposed algorithm to produce the robot position during tracking the rectangular trajectory. The results of the robot position \((x, y, \theta)\) are illustrated in Figure 6.11.

The results of the robot localization on tracking the rectangular trajectory are illustrated in Figure 6.12. In this figure, there is almost no divergence occurred.
Figure 6.6: The robot position of the sine wave experiment. (a) $x$-value of the robot; (b) $y$-value of the robot; (c) Orientation value of the robot.
Figure 6.7: The localization estimation of the robot of sine wave experiment.
Figure 6.8: The robot follows the rectangular trajectory. (a) The robot movement at various time instances on the trajectory; (b) The corresponding robot positions on the path of the panel (a).
Figure 6.9: The velocities of the robot to follow a rectangular trajectory. (a) Velocities of the two wheels; (b) Linear velocity of the robot centre; (c) Angular velocity of the robot.
Figure 6.10: Measurements of robot sensors.

during the robot operation.

6.2 Concluding Remarks

In this chapter, two experiments are running on the real mobile robot to prove the efficiency of the proposed algorithm. These two experiments are the cases that always cause the localization system failed. However, in our approach, with the support of the fuzzy logic training and the neural network learning, the EKF is being able to deal with these two situations without divergence happened. The two experiments further prove the high performance of the fuzzy logic integrated neural network-aided EKF.
Figure 6.11: The robot position on the rectangular path. (a) $x$-value of the robot; (b) $y$-value of the robot; (c) Orientation value of the robot.
Figure 6.12: The localization of the robot on rectangular trajectory experiment.
Chapter 7

Conclusions and Future Work

7.1 Conclusion

In this thesis, a Kalman filter based approach for mobile robot localization has been investigated. The proposed fuzzy logic controller aided extended Kalman filter enables the mobile robot to localize itself in the working environment precisely. The neural network is designed to learn the tuning results of the filter. With the integration of the neural networks, the proposed extended Kalman filter is being able to adjust to the working environment sooner. The presented Kalman filter is experimented on an real mobile robot, Dr. Robot Sentinel. The results of implementations show a good performance of the proposed algorithm.

7.2 Future Work

The proposed method in this thesis is effective on the mobile robot localization. However, there are still some rooms to improve the performance of it. In the future work, there are some ways to be utilized to optimize the fuzzy logic controller, like GA and neural network. Since the neural network has the capability of saving computation, more advanced on-board sensors, like camera and laser, can be employed in the localization.
References


