Maintaining Stable Wireless Communication Connections among Multiple Collaborative Mobile Robots by Intelligent Robot Motion Control

A Dissertation Presented

by

Xu Zhong

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Mechanical Engineering

Stony Brook University

December 2014

Stony Brook University

The Graduate School

Xu Zhong

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation.

Dr. Yu Zhou – Dissertation Advisor
Associate Professor, Mechanical Engineering Department,
State University of New York, Institute of Technology

Dr. Jeffrey Q. Ge – Chairperson of Defense
Professor, Mechanical Engineering Department

Dr. Anurag Purwar – Member
Research Associate Professor, Mechanical Engineering Department

Dr. Jennifer Wong – Outside Member
Research Assistant Professor, Computer Science Department

This dissertation is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

Maintaining Stable Wireless Communication Connections among Multiple Collaborative Mobile Robots by Intelligent Robot Motion Control

by

Xu Zhong

Doctor of Philosophy

in

Mechanical Engineering

Stony Brook University

2014

This dissertation addresses the critical issue of establishing and maintaining desired wireless communication connectivity in a team of collaborative mobile robots, which is highly demanded for reliable functioning of multi-robot systems but challenging in realistic environments. The signal propagation of wireless communications among mobile robots is affected by many issues, including the transmission power, distance, obstacles, robot movement, and other environmental conditions, which result in signal loss, attenuation, multi-path fading and shadowing. Consequently, the communication condition among mobile robots in a physical environment is usually unstable, and it is difficult to accurately predict the actual communication ranges of robots.

We propose decentralized control strategies which, based on perceived link quality, adopts artificial intelligence schemes to accommodate the fluctuating communication condition, and approach and maintain desired and reliable communication connections among neighboring robots. The intelligent control strategies, such as fuzz logic and artificial neural network, are introduced as our control frameworks, which are tuned and adjusted by back propagation or reinforcement learning. These intelligent schemes are defined for each robot as an onboard controller to adapt to the quality of the wireless links with its neighbors. The combined effect of all robots' movement allows the MRS to achieve the desired communication connections. The effectiveness of proposed intelligent controllers are verified by the simulations or experiments. The advantages and disadvantages of these controllers are compared.

Pioneer P3Dx and Amigobot mobile robots are adopted as our experiment platform, which provides dedicated motion controller with encoder feedback. A real-time robust localization algorithm is proposed for mobile robots to avoid accumulate errors from onboard encoders in indoor environments. Proposed localization algorithm is designed based on the recognition of artificial landmarks captured by a single onboard camera. The artificial landmark identity are encoded with nested circles in black and white. The center of landmarks are located by a fast two-phase recognition algorithm in image frame. The absolute position of the camera/robot in the environment is estimated using the geometric mapping between the image and global frames.

An application of proposed wireless maintaining controller is explored in a task of deploying multiple robots into a realistic environment. Each robot can perform two categories of behaviors: 1) fundamental behaviors, e.g. forming sensory coverage, maintaining wireless communication connections, and avoiding collisions, are necessary for reliable functioning of the MRS; 2) application behaviors, e.g. search and exploration, are defined to fulfill the goals of specific deployment tasks. In the proposed control architecture, these behaviors are implemented by appropriately-defined parallel fuzzy controllers with different priorities, and behaviors can easily be added or removed by adding or removing corresponding parallel fuzzy controllers without affecting other behaviors.

I dedicate this dissertation to my beloved family.

Table of Contents

List of Figures

List of Tables

ACKNOWLEDGEMENTS

First and foremost, I would like to express my special appreciation and thanks Professor Yu Zhou, my academic advisor, for your guidance, encouragement and patience throughout the work of this dissertation. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. I appreciate all your time and contribution on coaching me in my Ph.D. experience.

I would like to thank my committee members Professor Q. Jeffrey Ge, Professor Anurag Purwar, and Professor Jennifer Wong for serving as my committee members. I also want to thank you for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions.

Special thanks are given to my lab mates Xionghui Lu, Hanyu Liu, Chuanchai Zou, Lukasz Krzeminski and Xiang Cui, and my friends Zhihua Gan for their assistance in the process of controller design and experimental data acquisition; and department staff Mayra Santiago and Melissa Castelbuono for their valuable assistance.

Chapter 1　Introduction and Background

This dissertation focuses on multi-robot systems (MRS) which consist of multiple mobile robots collaborating in task execution. We address the communication maintenance problem in the context of a fundamental task required by many MRS applications. That is to deploy a team of mobile robots into an unknown environment to form a communication coverage for supporting system management, data exchange and task coordination. It is a multi-robot deployment problem. This chapter presents the research motivation, related works and dissertation structure.

1.1 Research Motivation

In the past decade, a dramatically development of mobile robot has explored a mature capability of MRS, which expanses its applications in many useful fields to improve our life. Some well-known robots, including the Da Vinci robot for surgical operation in medical field, the FANUC robot for automation production and service in industry, the LEGO robot for assisting the teacher to train students in education, iRobot for helping the householder to clean the floor in home applicant, and etc. However, people are not satisfied with a single mobile robot due to its limited capability. At the same time, MRS has the following advantages：

1) MRS may provide a solution to accomplish more complicated tasks, which are far beyond the capability of an individual robot, such as robot soccer, environment monitoring and etc.

2) MRS may perform the assigned task in more reliable, faster, or cheaper way, such as disaster rescue, planetary exploration, and etc.

The performance gain of an MRS over multiple independent robots relies on the efficient coordination among robots [1], which in turn depends on reliable communication connections to support efficient data exchange among robots [2]. Thus, in study of multi-robot system, maintaining the communication is a fundamental problem. It is critical for many applications of multi robot system.

However, less attention has been paid to controlling robot motion to establish and maintain communication connections which enable the coordination. To address the inadequacy in the current research of establishing and maintaining communication connectivity in an MRS, we propose five distributed intelligent deployment control schemes to spread a team of mobile robots into an unknown environment to establish and maintain the desired communication coverage, which takes advantage of robot motion to adapt to the real-time fluctuation of wireless link quality.

The effectiveness of all proposed schemes has been verified in several simulated environments with different signal propagation conditions based on a probabilistic signal propagation model. Based on Pioneer P3Dx and Amigobot robots, a real experiment is carried out to show that the fuzzy logic control scheme can lead a team of mobile robots into a desired and reliable wireless communication coverage.

In the experiment, one important issue is addressed: robot localization. Given initial positions and orientations, on-board encoders of Pioneer P3DXs and Amigobot are able to calculate robot locations based on the relative localization. However, accumulative errors limit encoders' usability in demanding applications and a global localization method is desirable. This dissertation proposes a real-time robust positioning algorithm for mobile robots in indoor environments, based on the recognition of artificial landmarks captured by a single onboard webcam.

1.2 Related Works

In this section, we briefly discuss related works about MRS communication deployment, intelligent control and individual robot localization.

1.2.1 Multi-robot Communication Deployment

In principle, the deployment of multiple mobile robots can be controlled in either a centralized or a distributed manner. However, the unstable communication conditions in many realistic environments require prompt adaptation of robot positions in order to maintain communication connectivity in MRS. Centralized control [3-7], which relies on an established network for information collection and/or command delivery from a leader, is slow in response to the changes in the system/environment and thus not suitable for this mission. Instead, distributed control allows each robot to control its motion according to local information. It is thus highly adaptive to the changes in the system and environment, robust to the failure of individual robots and scalable to large MRS, and hence provides a feasible solution to the problem of communication connectivity in MRS.

Existing distributed deployment methods in MRS mainly focus on coordinating robots' motion to accomplish sensor coverage, e.g. using artificial potential / force fields [8-21], Voronoi tessellation [22-31], diffusion models [32-35], etc. Existing works on multi-robot coordination mostly maintain communication connections based on estimated circular communication ranges [9,11,14,16-18, 32-33,36-44], and reliable connections are widely assumed inside the range.

However, it is challenging to maintain communication connections among mobile robots in many realistic situations. The signal propagation distance, obstacles, robot motion and various environmental factors cause the signal power decay, reflection/diffraction/scattering, shadowing and fading [45]. These signal propagation conditions have highly significant impact on the performance of wireless communications [45], resulting in not only limited but also unstable robot communication ranges. Using an ideal fixed-range communication model without considering the channel uncertainty may lead to communication failure, and hence reduce the reliability of multi-robot coordination and degrade the MRS performance.

The unstable communication conditions in realistic environments call for more flexible online adaptation to the *in situ* varying wireless link quality. Several measures have been defined to capture the quality of a wireless link in different types of wireless networks [46-51], such as Received Signal Strength Indication (RSSI) -- the signal strength observed at the receiver's antenna during packet reception, Signal-to-Interference-plus-Noise Ratio (SINR) -- the extent to which the power of the received signal exceeds the sum of noise plus interference at the receiver, Packet-Delivery Ratio (PDR) -- the ratio of the correctly received packets at the receiver to the total number of packets sent by the sender, Bit-Error Rate (BER) -- the ratio of bits with errors to the total number of bits that have been received over a given time period, and Link Quality Indicator (LQI) -- the quality of the data packets received by a receiver. For two widely-adopted types of wireless networks IEEE 802.11 (Wi-Fi) and IEEE 802.15.4 (ZigBee), RSSI, SINR, PDR and BER are used for both, while LQI is used only for IEEE 802.15.4.

Based on the perceived link quality, a number of reactive communication maintenance approaches have been proposed. Approach of internalized plans have been proposed in [52] for robots to adapt to the communication condition by grid map of the environment. Value-Based

Communication Preservation (VBCP) has been proposed in [53] to maintain communications between members of robot teams based on a priori map of environment and the current position of all the robot's teammates. Pre-constructed signal strength map of the underlying environment has also been used to guide the navigation of the robots [54-56]. Online perceived signal strength and network bandwidth [54-56] have been used as the feedback for reactive communication maintenance. Moreover, [57] addresses the issue of maintain a robust high-bandwidth RF communication link between a mobile robot and its remote control station by utilizing a number of autonomous mobile relay nodes.

The work of this proposal focuses on the distributed reactive scheme for communication maintenance, because 1) it depends on only the locally perceived link quality and is thus suitable for distributed multi-robot task performance, 2) the signal strength map is not always available in advance, particularly for urgent tasks in unknown environments, and *in situ* communication conditions may change and thus deviate from the signal strength map due to the multiple previously-mentioned affecting factors. Meanwhile, though the basic concept of reactive communication maintenance has been introduced, e.g. behavior-based control, there is still a lack of robust solutions to maintaining communication connections with multiple neighbors in realistic environments.

With the intention to seek a more adaptive solution to the communication maintenance problem, we notice that fuzzy logic, providing a sound method to deal with the uncertainty in modeling [58-60], has been used to address different problems in MRS. Jahangir designed a control scheme, based on centroid Voronoi configuration integrated with robust adaptive fuzzy control techniques, for multi-robot coverage [61]. An adaptive fuzzy logic system is used to approximate the unknown system dynamics for formation control problem in MRS [62]. In [63], Bazoula present a controller based on fuzzy logic for the leader following formation of multiple nonholonomic mobile robots. A new approach to cooperative localization in MRS is achieved by using fuzzy logic to combine unreliable information [64].

Considering the requirement of the communication maintenance problem, we found that artificial neural networks [65, 66] have been widely adopted in multi-robot field. Jolly et al. applied neural networks to trajectory planning [67] and small-team decision making [68]. They also applied a fuzzy neural network to robot action selection [69]. Kanaya and Tanaka developed a cellular neural network for path planning in an environment containing multiple autonomous robots [70]. Lee proposed a neural network as a decision controller of robot behavior in a multi-robot pursuit problem [71]. Quinn et al. designed a neural network controller for formation movement in small, homogeneous teams of autonomous mobile robots [72].

We also notice that reinforcement learning [73] has been adopted in MRS to facilitate the adaptation to specific systems and realistic environments. Li et al. used reinforcement learning to choose the best action strategies for a cooperative pursuit problem with multiple mobile targets [74]. Wang and Silva developed a multi-robot object transportation system, integrating reinforcement learning and genetic algorithms [75]. Gu and Yang applied a gradient reinforcement learning algorithm with fuzzy policy to a multi-robot flocking problem [76]. Zhang et al. presented an adaptive task assignment method, based on a self-reinforcement learning model inspired by the behavior of social insects, for a team of fully distributed mobile robots with initially identical functionalities in unknown environments [77]. Azouaoui et al. proposed a real-time multi-robot group navigation algorithm based on reinforcement learning [78]. Yasuda et al. used a neural network to predict robots’ status in a reinforcement learning process for robot behavior control [79].

Fuzzy neural networks and reinforcement learning have also been combined to attain adaptive intelligent control of robots. Poramate et al. applied correlation learning and continuous actor-critic reinforcement learning to neural network control of mobile robot system to achieve a goal-directed behavior control manner [80]. Miljkovic et al. implemented neural network reinforcement learning technique into the visual control of robot manipulators [81]. Desouky and Schwartz used Q-learning to tune the parameters of a fuzzy logic controller for different pursuit-evasion differential games [82].

1.2.2 Localization of Individual Robot

Extracting the knowledge of the position and orientation of a vehicle is a fundamental problem in mobile robot applications [83]. This dissertation introduces a real-time indoor mobile robot localization scheme based on an artificial landmark design and a corresponding vision-based recognition algorithm.

In the past decades, remarkable efforts have been made to study mobile robot localization techniques, and numerous approaches have been proposed to recover the global position and orientation of a mobile robot in an indoor environment.

Ranging based on infrared [84], ultrasound[85] and radio-frequency (RF) signals have been widely applied, using range estimation schemes such as time-difference-of-arrival (TDOA) estimation [86], time-of-arrival(TOA) estimation [87], angle-of-arrival(AOA) estimation [88] and received signal strength indicator (RSSI) [89]. Some representative systems include RADAR [90], DOLPHIN [91], and Cricket [92]. However, the range-based localization has several limitations, including the interference and collisions among signals, signal cross-talking at receivers, and inaccuracy in range estimation, which may significantly affect the positioning performance.

Meanwhile, vision-based localization by capturing landmarks at known positions is also widely studied. The use of artificial landmarks simplifies the recognition procedure and compensates for the instability of natural landmarks [93]. Wen et al. attached paper-made landmarks onto ceilings or walls, and developed a topological algorithm to help the mobile robots navigate. [94]. Li and Yang used numbers as their landmarks, as the shape of the numbers are different, which can be seen and recognized easily by robot vision and thus guide the robot [95]. Kim and Lyou presented an indoor navigation scheme for a skid-steering mobile robot using a gyro sensor and a monocular camera. The position of the robot is estimated from ceiling landmark images, and is combined with the odometry information by an extended kalman filter (EKF) [96]. Greggio et al. used circular shaped landmarks to position the iCub humanoid robotics platform simulator and real robots [97]. Choi et al. introduced a hierarchical algorithm for mobile robots based on ultrasonic sensors and RFID tags [98]. Fukuda developed a vision based navigation system for autonomous mobile robots that recognizes outlets of air conditioning system located on the ceiling as landmarks. They used neural network (NN) to detect landmarks quickly and applied the fuzzy template matching (FTM) method to detect the landmarks [99]. Liu and Zhou purposed a visual trilateration based method to recover the 3D position of a mobile robot from a single image of landmarks taken by an onboard camera [100].

Recently, people pay more attention to the vision-based localization since it has more advantage over the range-based. Firstly, vision-based localization is more robust to the dynamics of robots and environments. Secondly, it provides more accurate position results upon successful landmark recognition. However, it remains challenging to balance between the landmark design and recognition to accomplish reliable and accurate landmark recognition.

A low-cost real-time robot localization system with high accuracy is highly demanded. In this dissertation, the artificial landmarks are encoded with nested circles in black and white. This design reduces the complexity of algorithm and lowers the requirement of the camera by maximizing the contrast ratio of landmarks and providing the omni-directionally identical property. The localization system uses a single upward-facing webcam as the vision sensor to capture landmarks attached to the ceiling. The recognition algorithm determines the identity of the landmarks based on a fast image processing and matching procedure. A geometric mapping between the image and global coordinates is used to decide the absolute position of the camera in real time.

1.3 Dissertation Structure

Chapter 2 presents a real-time robust localization algorithm for mobile robots in indoor environments, based on the recognition of artificial landmarks captured by a single onboard camera. Considering the disturbance of lighting condition, we encode the landmark identity with nested circles in black and white. The recognition algorithm consists of a global and a local recognition parts. The global recognition is a fast overall recognition process, including light detection, image clustering, region of interest (ROI) extraction, and ROI identification. If the number of identified ROIs does not meet the requirement of the localization algorithm, the local recognition will process those unidentified region of interest (ROI) through adaptive ROI expansion and template cutting. Based on the landmark recognition, the absolute position of the camera in the environment is estimated using the geometric mapping between the pixel frame and global frame. The proposed approach is tested via experiments in a real indoor environment, and the result reveals high localization robustness and consistency to the lighting condition.

Chapter 3 proposes five distributed intelligent control schemes to spread a team of mobile robots into an unknown environment to establish and maintain the desired communication coverage. The intelligent control schemes are proposed based on fuzzy logic (FL), artificial neural network (ANN), fuzzy neural network (FNN), radial basis network (RBN), back propagation learning (BP), and reinforcement learning (RL). FL, ANN, FNN and RBN are intelligent control frameworks, of which the parameters are adjusted and tuned by BP or RL. In this dissertation, we introduce the following control schemes: FL, BP-ANN, BP-FNN, RL-RBN and RL-FNN. The simulation results show that all proposed schemes can establish and maintain effective communication coverage under the documented path loss exponents and uncertainties, and cause the average RSSI to converge towards the desired range. The effectiveness of fuzzy logic controller are also verified by experiments under different initial setups.

Chapter 4 discusses a behavior-based parallel fuzzy control framework to approach desired sensory coverage in an unknown target environment while maintaining wireless communication connections and avoiding obstacles among a multi-robot system. The behaviors of each robot are separated into two parts based on priority level, fundamental with higher priority and application behaviors. Fundamental behaviors, i.e. forming sensory coverage, maintaining wireless communication connections, and avoiding collisions, are necessary for reliable functioning of the MRS and implemented by three parallel fuzzy controllers. Application behaviors are decided by the specific deployment tasks, such as search, exploration and etc., and also can be realized by parallel fuzzy controllers but based on the fundamental fuzzy controllers due to different priority. Proposed control architecture can easily remove or add behaviors by removing/adding corresponding parallel fuzzy controllers without any effect to other behaviors.

The effectiveness of the proposed fuzzy control framework has been verified through a series of simulated deployment and destination approaching tasks over unknown environments.

Chapter 5 focuses on the conclusion and contribution of this dissertation. In summary, proposed distributed intelligent controllers are capable of achieving the control objectives, i.e. maintaining stable wireless communication connections among multiple collaborative mobile robots.

Chapter 2 Indoor Robot Localization

This chapter proposes a real-time robust localization algorithm for mobile robots in indoor environments, based on the recognition of artificial landmarks captured by a single onboard camera. The pixel coordinates and identity of landmarks are decided by the recognition algorithm. The localization algorithm can estimate the global orientation and position of camera based on the pixel coordinates and identity of landmarks.

Considering the need of recognition from different directions and the disturbance of lighting conditions, we encode the landmark identity with nested circles in black and white alternately. The recognition algorithm consists of a global and a local recognition processes. The global recognition is a fast overall recognition process, including light detection, image clustering, region of interest (ROI) extraction, and ROI identification. The number of identified ROIs has to meet the requirement of the localization algorithm. Otherwise, the local recognition will process those unidentified region of interest (ROI) through adaptive ROI expansion and template cutting. Based on the landmark recognition, the absolute position of the camera in the environment is estimated using the geometric mapping between the image and global frames. The proposed approach is evaluated via experiments in a real indoor environment, and the result reveals high localization robustness and consistency to various lighting condition.

The layout for the rest of this chapter is organized as follows. Section 2.1 will describe our localization system and typical landmark design. Section 2.2 will propose our real-time robust landmark recognition scheme. Section 2.3 will introduce the geometric localization method and report the experimental results. Section 2.4 will summarize the contribution of this chapter.

2.1 System Overview and Landmark Design

This section will first introduce the localization system, which consists of a low-cost vision sensor carried by a mobile platform and a set of artificial landmarks attached to the ceiling of an indoor environment, and then present the design of the artificial landmarks, which consists of concentric nested circles in white and black.

2.1.1 System Overview

The proposed real-time localization system computes the absolute position and orientation of a mobile robot by detecting the pre-installed landmarks in an indoor environment. Fig. 2.1 shows the hardware of the localization system. The image on the left shows the onboard hardware of the proposed localization system; the image on the right shows a typical indoor environment with the proposed landmarks and localization system. A mobile robot is controlled by a laptop and equipped with a webcam. The camera is mounted on the robot with a wood frame and faces upward to the ceiling. A number of artificial landmarks are installed at known locations on the ceiling, which could provide more tolerance to obstacles than other layouts [101-103]. The images captured by the onboard camera are sent to the laptop for robot position and orientation estimation.

When an image of the environment, containing a few artificial landmarks, is received by the onboard computer, at first a landmark detection and recognition procedure is applied to find all the captured landmarks in the image and recognize their identities. Then, given the intrinsic parameters of the camera and the global landmark positions in the environment, a geometric mapping is used to estimate the global position and orientation of the camera, which is equivalent to the position and orientation of the robot. While the position of the landmarks are mapped when they are installed to the environment, the camera parameters are obtained through calibration [104], and a popular camera calibration toolbox can be found at [105].

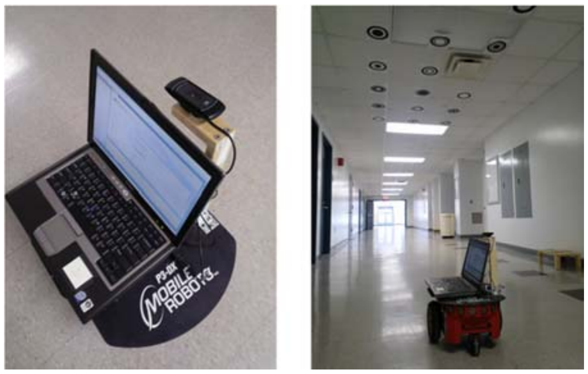


Figure 2.1: Localization System.

2.1.2 Landmark Design

A well-designed landmark will simplify the landmark recognition algorithm, which directly dominate the performance of a localization system. Through the qualitative analysis of practical experiments, we proposed a set of concentric circular rings with associated diameters as the ideal landmarks.

In Fig. 2.2, our design of a landmark includes two components, the outer ring and identity code (inner rings). The red dashed square denotes the bounding box. The outmost black circular ring denotes the outer ring. The black or white circular rings inside the outer ring are the inner rings. Landmarks are printed on a regular paper that does not contain self-illuminant materials. This simple design reduces the cost of the system and the complexity of the recognition algorithm, and facilates recognition from different dircetions under different circumferences with various lighting conditions.

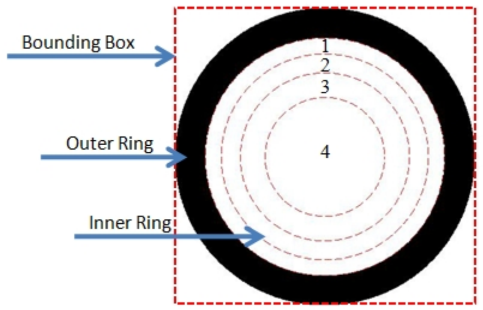


Figure 2.2: Design of artificial landmarks.

1) *Outer ring:* To address the lighting variation in complicated real indoor environments, black color is used to identify landmarks, which is more reliable than other colors under different lighting conditions. In our landmark design, each landmark has an outer ring in black color with a fixed diameter, which defines a uniform boundary for the landmarks. The bounding box of the landmark is the external tangent square.

2) *Identity code:* The identity of each landmark is encoded into a base-C numeral system. Letter C is a positive integer that corresponds to the number of colors of landmark configuration. Typical examples of numeral systems include base-2(binary), base-8(octal) and base-16(hex). The number of bits, denoted by K, needed to encode N different landmarks can be determined by Eq. (2.1),

$$K = \mathrm{Floor}\ (\ \mathrm{Log}_c N\ ) + 1 \qquad (2.1)$$

Where Floor (X) implies the largest integer less than or equal to X, and C is the base of the numeral system. In our case, C=2 because only the black and white colors are used. In another word, our landmark identity is binarily encoded with either 1 (white) or 0 (black). Fig. 2.3 shows an example with C=2, K=4, where black denotes 0, and white denotes 1. The landmarks represent 0001 to 1111 in binary system from left to right within each row and from top to bottom. Landmark (r, c) denotes the landmark located in row r and column c.

The letter C in Eq. (2.1), the based number of the coding system, has significant influence on the identity code and inner rings. The number of fields of landmark needed to be encoded is reduced as the number of color selection increases. Although it simplifies the recognition procedure, the color information is not stable at different illuminations during experimental tests; it turns out to be more challenged for system to discriminate the colors with various lighting conditions in an environment.

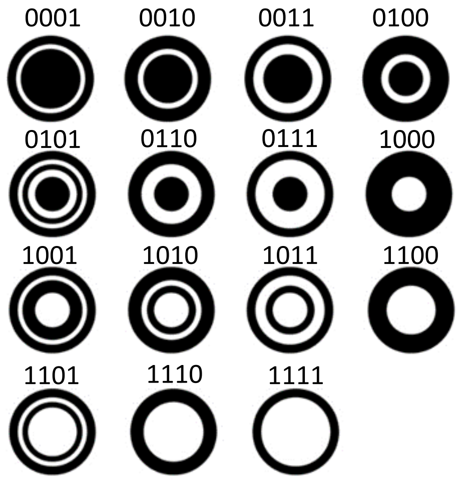


Figure 2.3: A whole set of landmarks with C=2 and K=4.

In order to simplify the recognition process, we use K concentric rings to represent the K-bit identity code, where Fig. 2.3 shows a whole set of landmarks with 4-bit identity code. Each ring is an encoding field, and is, from outer to inner, associated with a bit of the identity code. If the corresponding bit is 0, the field is colored with black; if 1, colored with white. Moreover, the width, W_i, of each encode field is defined to maintain the same area for each ring as Eq. (2.2),

$$W_i = R_i - R_{i+1} = \left(\sqrt{K-i+1} - \sqrt{K-i}\right) \times R_1 \big/ \sqrt{K} \tag{2.1}$$

Where K denotes the total number of bits, i denotes the index of bit (i=1 to K), R_i denotes the outer radius of i^{th} inner ring. Thus, a unique landmark code is generated to present an identical landmark configuration correspondingly. Since the total number of needed landmarks for an indoor environment is limited, the width of each field is large enough to support reliable landmark recognition. These rings bring the desired characteristic, omni-directional, to the landmark. That is, when the camera looks at such a landmark from different orientations, the landmark shows in the images with the similar appearance and limited distortions.

2.2 Proposed Recognition Scheme

Reliable and robust landmark recognition scheme makes the foundation of valid and accurate robot localization. As described in the Section 2.1.1, the positions of the captured landmarks in both the image and global frame are required in our localization process to estimate the real position of the camera/robot. To extract these needed information, proposed recognition algorithm can locate the position of landmarks in image frame and identify the landmarks to obtain the global coordinates from a pre-defined map of identify and global coordinates. Our recognition algorithm consists of two main steps 1) landmark or region of interest (ROI) detection, which locates the center and outer ring of each captured landmark, and 2) landmark/ROI identification, which determines the identity of each captured landmark.

Considering the requirement of real-time localization, proposed recognition algorithm is designed into a two-layer sequential structure, i.e. global and local recognition, shown in Fig. 2.4. Firstly, a fast global detection and identification are applied to recognize the ROIs. Secondly, if the number of identified landmarks cannot meet the requirement of localization algorithm, the local recognition scheme will adaptively re-identify those unidentified ROIs based on the information obtained from global layer. The local layer can identify those incomplete landmarks due to the extreme lighting condition or the position of landmarks (on the boundary of image).

The recognition process is shown in Fig. 2.4, where ROI denotes region of interest, CLR candidate landmark region, ICLR identified CLR, UCLR unidentified CLR, DL the shortest distance between an UCLR and a light source, DLThr the threshold of DL to find the UCLRs close to the light sources, DI the shortest distance between an CLR and the image boundaries, DIThr the threshold of DI to find the UCLRs located on the boundaries of the image, and DUCLRs the UCLRs with DL<=DLThr or DI<=DIThr.

The input of the algorithm is a RGB picture captured by a single vision sensor, e.g. a webcam. Then the RGB image is converted to grayscale image by eliminating the hue and saturation information while retaining the luminance. The rest of this section will explain the details of the global and local recognition processes.

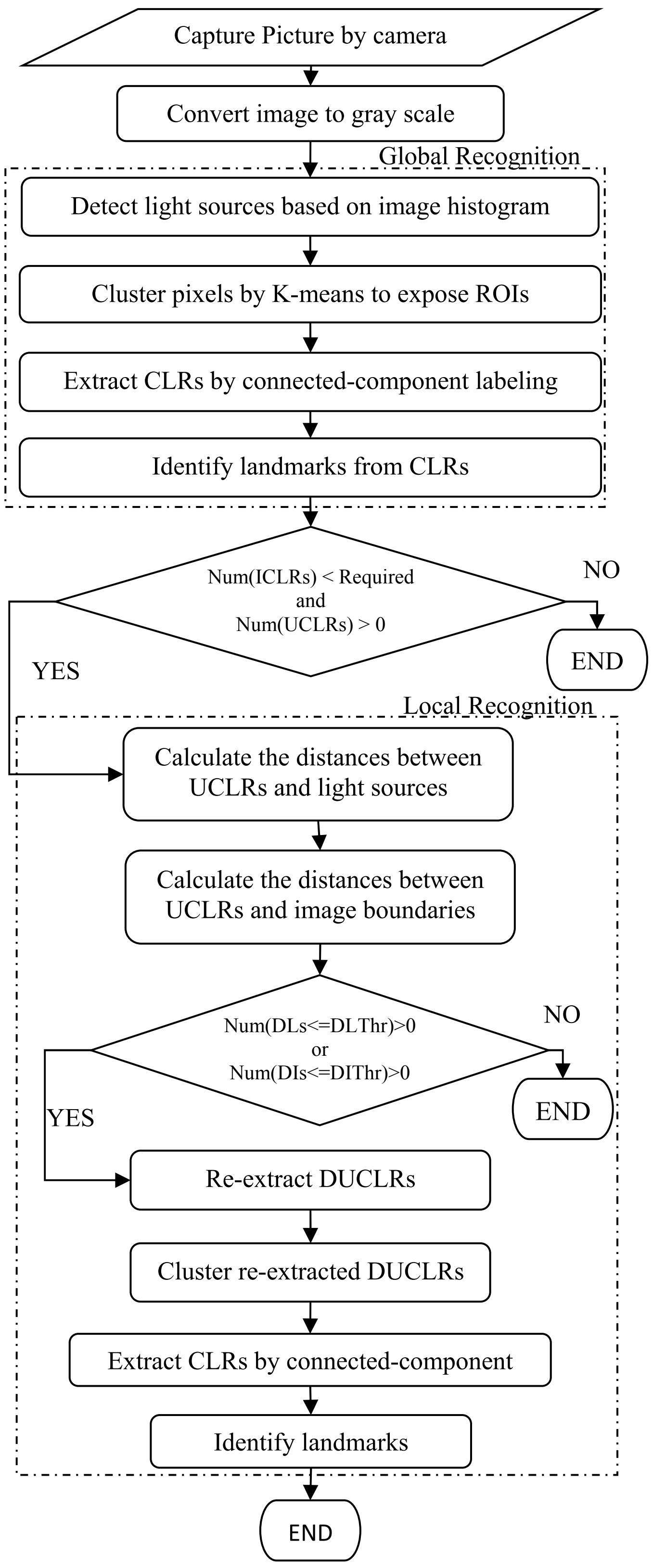

Capture Picture by camera
Convert image to gray scale
Global Recognition
Detect light sources based on image histogram
Cluster pixels by K-means to expose ROIs
Extract CLRs by connected-component labeling
Identify landmarks from CLRs
Num(ICLRs) < Required and Num(UCLRs) > 0
NO
END
YES
Local Recognition
Calculate the distances between UCLRs and light sources
Calculate the distances between UCLRs and image boundaries
Num(DLs<=DLThr)>0 or Num(DIs<=DIThr)>0
NO
END
YES
Re-extract DUCLRs
Cluster re-extracted DUCLRs
Extract CLRs by connected-component
Identify landmarks
END

Figure 2.4: The process of landmark recognition algorithm.

2.2.1 Global Recognition

The global recognition process is a fast overall recognition process, which provides information of identified regions of interest (ROIs) to local recognition and localization algorithm. The global operations are executed in the following order: light detection, image clustering, and extraction and identification of ROIs. Fig. 2.5 shows two input images to the global recognition process under different lighting conditions. In left picture, light source is small at the lower right corner. In right picture, light source is big in the mid-right region.

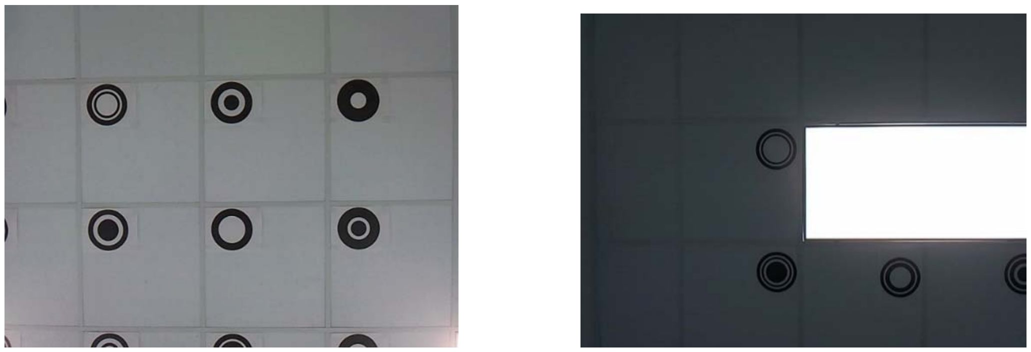


Figure 2.5: Examples of different environment illuminations.

1) Detect light sources based on image histogram:

Light source detection is a pre-process for image clustering. The environment illumination affects the white balance of the vision sensor, which makes the pixels surrounding the light source appear darker, shown in Fig. 2.5. It has a direct influence on the process and performance of landmark recognition. The knowledge of the locations of light sources in the image facilitates the landmark recognition process to deal with the effect of lighting and improve the rate and reliability of landmark recognition.

We do a fast light source detection based on the image histogram, as showed in Fig. 2.6. An image histogram presents the distribution of pixels across intensity values in a grayscale image. From the image histogram, the total number of light source pixels, whose intensity values are greater than a threshold, red lines shown in Fig. 2.6 is obtained. Comparing with the threshold, red lines shown in Fig. 2.6, all the light source pixels are labeled one by one. All the light source pixels are labeled for future operations.

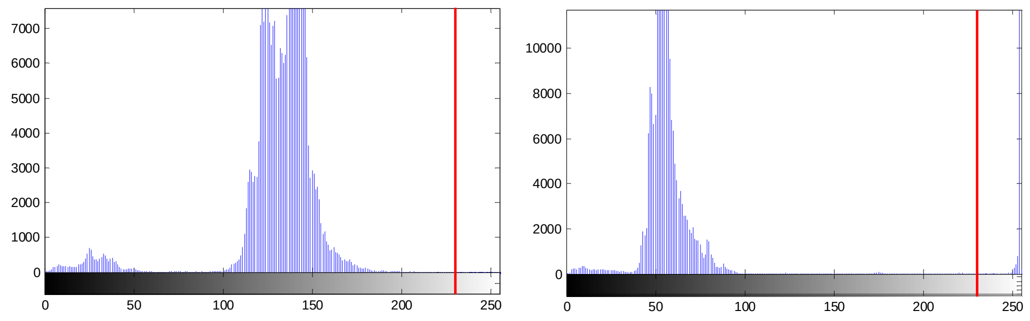


Figure 2.6: Comparison of two image histogram.

2) Cluster pixels by K-means to expose ROIs:

The grayscale image is converted to a binary image using an image clustering method K-means. It partitions all the pixels in the grayscale image, excluding the light source pixels, into two clusters, "region of interest" (ROI) and "background", by iteratively minimizing within-cluster sum of point-to-centroid differences of gray value. Besides, the start points of k-means are estimated via the corresponding image histogram, which improves accuracy of clustering and increases the speed of convergence. The image clustering results of Fig. 2.5 are shown in Fig. 2.7, where the white region represents the light source, the gray region represents the background, and the black regions are those of interest.

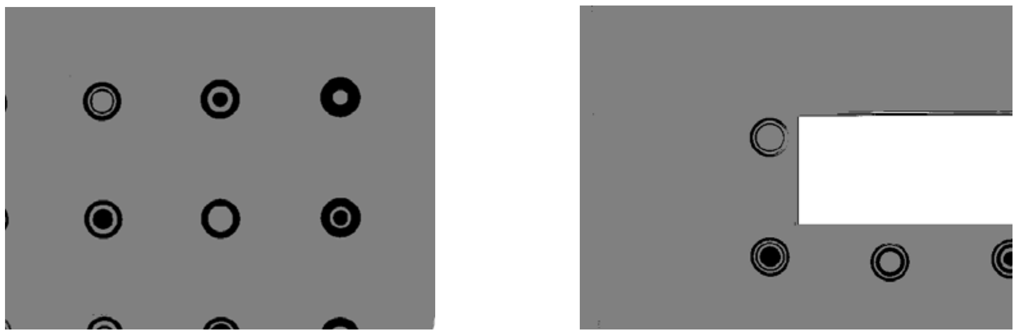


Figure 2.7: K-means results of Fig. 2.5.

3) Extract CLRs by connected-component labeling:

The CLRs are extracted by scanning the image pixel by pixel to identify connected pixel regions. Tiny components, considered as noises, are removed from the CLR list. Internal CLRs, such as the inner rings of landmarks, are removed from the CLR list by calculating the relation of inclusion. Fig. 2.8 shows the CLRs extracted from Fig. 2.7.

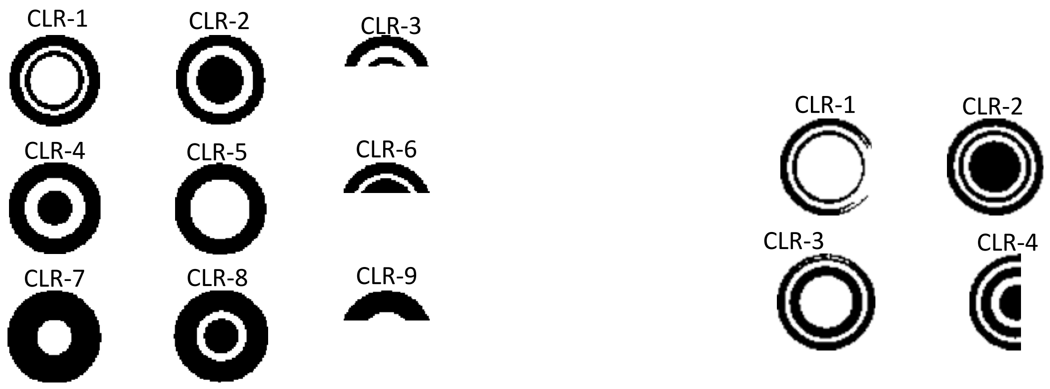


Figure 2.8: CLRs extracted from the Fig. 2.7.

4) Identify landmarks from CLRs:

The identity of CLRs are determined by the correlation coefficient defined in Eq. (2.3). First, we calculate all the correlation coefficients between a CLR and all templates. Second, if the maximum of the calculated correlation coefficients is greater than the given threshold, we mark this CLR as identified, save its index and estimate its position in pixel coordinate system; otherwise, we mark the CLR as unidentified for the local recognition process. We apply these two steps to all CLRs. Fig. 2.9 shows the matching results for Fig. 2.8 on the grayscale images. A yellow dot denotes the calculated center of a landmark. A green number denotes the index of an identified landmark.

$$r = \frac{\sum_m \sum_n \left(A_{mn} - \overline{A}\right)\left(B_{mn} - \overline{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \overline{A}\right)^2\right)\left(\sum_m \sum_n \left(B_{mn} - \overline{B}\right)^2\right)}} \tag{2.3}$$

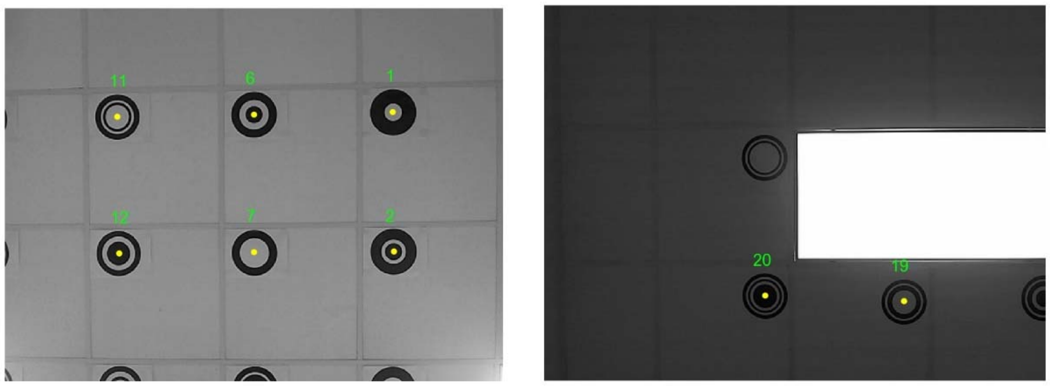


Figure 2.9: Identified CLRs for Fig. 2.8.

2.2.2 Local Recognition

If the number of identified CLRs is less than the required for the localization algorithm and the number of unidentified CLRs (UCLRs) is greater than zero, the local recognition will process those UCLRs which are marked in the global recognition process. As shown in Fig. 2.8, the main reason of the global recognition failure is that UCLRs are incomplete due to their positions. Three kinds of positions lead to this incompleteness of UCLRs: 1) on the boundary of an image, 2) close to a light source, 3) both 1) and 2). To deal with it, we first calculate the distances between the UCLRs and the light source, and between the UCLRs and the image boundaries. If one or more distances of the UCLRs satisfy one of these three conditions (DUCLRs), we will proceed to the following steps to identify these DUCLRs: resizing those

DUCLRs, partitioning each DUCLRs into two groups ("interest" and "ceiling"), extracting ROIs and identifying DUROIs. The details of these steps are explained as follows:

1) Calculate the distances between UCLRs and light sources:

We define the smallest distance of two pixels as the distance between UCLRs and light sources. Considering the speed of the algorithm, we adopt canny edge detection [106] to detect the outer edge of the UCLRs and light sources. Then, the distance is decided by the minimum distance between the outer edge of UCLRs and light sources.

2) Calculate the distances between UCLRs and image boundaries:

We define the distance of a bounding box of a UCLR to an image boundary as the distance between the UCLR and image boundary. And the distance to 4 image boundaries is calculated.

3) Re-extract DUCLRs:

Considering the influence of light sources, an adaptive re-extracting algorithm, which will extend the UCLRs along vertical (height) and horizontal (width) directions, is proposed based on the distance between the UCLRs and light sources and the size of the identified CLRs. Algorithm 1 shows the iterative process for each DUCLR, which consists of three steps:

a) Calculate standard size of landmarks, shown in Line 1. The standard size is decided by the mean of size of identified CLRs. If no identified CLRs, the standard size is equal to the default size, which is given based on the environment.

b) Determine the increment, shown in Lines 3-17. Lines 3-5 exclude those DUCLRs, of which the size ratio is greater than the threshold. Lines 6-10 and 11-15 calculate the increments of the width and height based on the distance between a DUCLR and light sources (DistU2L), where the ratio is an adjustment coefficient. Lines 18-20 avoid crossing the border of the image based on the distance between a DUCLR and each boundary ($DistF2I_{i,j}$).

c) Re-extract DUCLRs, shown in Lines 23-25. The bounding box of DUCLRs is extended in the directions of left, right, up and down based on the calculated increments.

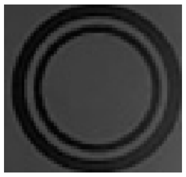
Algorithm 1. Re-extract Unidentified CLRs

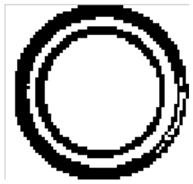
```
0.  Inputs: MeanIdentifiedCLRHeight, MeanIdentifiedCLRWidth
1.  [StandardHeight, StandardWidth] = [MeanIdentifiedCLRHeight, MeanIdentifiedCLRWidth]
                                   or = [DefaultHeight, DefaultWidth]
2.  for I = 1to Num( DUCLRs ) do
3.     if Max( Size( $DUCLR_i$ ) ) / Min( Size( $DUCLR_i$ ) ) > RatioThr then
4.        [VertIncrement, HoriIncrement ]=[0,0]
5.     else
6.        if $DUCLR_i$ .Height < StandardHeight then
7.           VertIncrement = StandardHeight - $DUCLR_i$ .Height
8.           VertIncrement = Min(VertIncrement , DistU2L* LightDistRatio)
9.        else   VertIncrement = 0
10.       end if
11.       if $DUCLR_i$ .Width < StandardWidth then
12.          HoriIncrement = StandardWidth - $DUCLR_i$ . Width
13.          HoriIncrement = Min( HoriIncrement , DistU2L* LightDistRatio )
14.       else   HoriIncrement =0
15.       end if
16.    end if
17.    Increment = [-VertIncrement, VertIncrement, -HoriIncrement, HoriIncrement]
18.    for j = 1:4 do
20.       Increment(j) = Min(Increment(j), $DistF2I_{i,j}$*sign(Increment(j)))
22.    end for
23.    for j = 1:4 do
24.       $DUCLR_i$ .Rect(j) = $DUCLR_i$ .Rect(j)+Increment(j)
25.    end for
26. end for
```

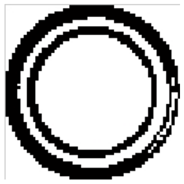
Fig. 2.10 continues the example of the right picture in Fig. 2.9, where (a) shows the resized DUCLR on the grayscale image for CLR-1 on right picture in Fig. 2.8. Fig. 2.11 shows a typical process of local recognition, where (a) shows the result of CLR extraction of global recognition, (b) shows the resized DUROI on the grayscale image for (a).

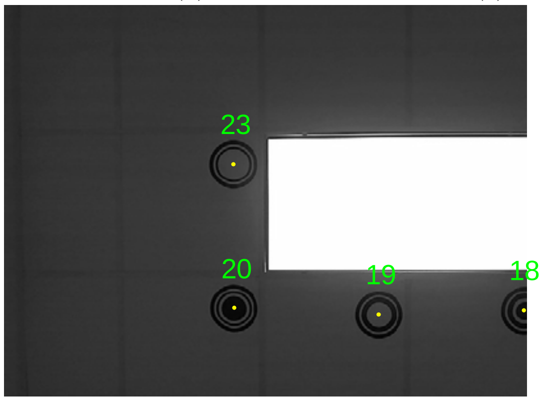
4) Cluster re-extracted DUCLRs:

The grayscale images of the extended DUCLRs are converted to binary images using the K-means method mentioned in the global recognition step 2). Two examples are shown in Fig. 2.10 (b) and Fig. 2.11 (c), which are the results of Fig. 2.10 (a) and Fig. 2.11 (b) respectively.


(a)


(b)


(c)



(d)

Figure 2.10: The local recognition process of ROI-1 on the right picture in Fig. 2.8.

In Fig. 2.10, (a) shows the resized DUROI on grayscale image, (b) shows the result of image clustering, (c) shows the results of DUROI extraction, and (d) shows the result of global and local recognition on grayscale image. The process of ID 19 on (e) is shown in Fig. 2.11, where (a) shows result of ROI extraction of global recognition, (b) shows the resized DUROI on grayscale image, (c) shows the result of image clustering, (d) shows the result of DUROI extraction, (e) shows the result of global and local recognition on grayscale image.

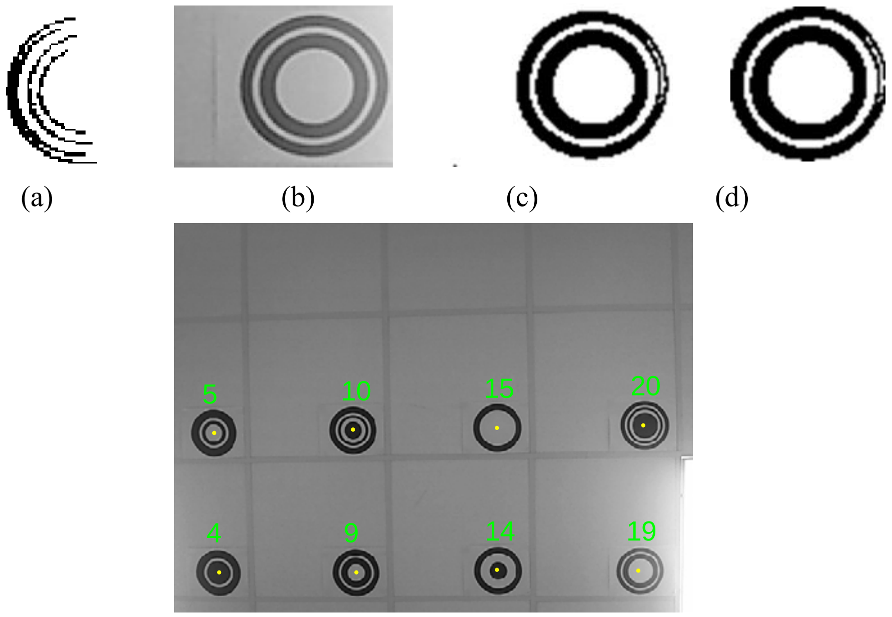


Figure 2.11: An example of local recognition process.

5) Extract CLRs by connected-component:

We apply the connected-component labeling method mentioned in the global recognition step 3) to extract CLRs from the resized binary images. Fig. 2.10 (c) and Fig. 2.11 (d) show the extracted binary images for Fig. 2.10 (b) and Fig. 2.11 (c) respectively.

6) Identify landmarks by template cutting

To recognize the DUCLRs on the boundary of image, the algorithm will cut all the templates to the same size of DUCLRs on the same side, which only applies to UCLRs with less than half incompleteness. The pixel position of UCLRs will be determined by Eq. (2.4), where u, v denote the horizontal and vertical pixel coordinate, u', v' denote the center of bounding box of UCLRs, mv, mu denote the width of missing part, +/- correspond to the right/left side of u and the bottom/top side of v.

$$u = u' \pm mu/2, \quad v = v' \pm mv/2\,. \tag{2.4}$$

Fig. 2.11 (d) and 11 (e) show the indices and positions of the landmarks resulting from the global and local recognition.

2.3 Experiments

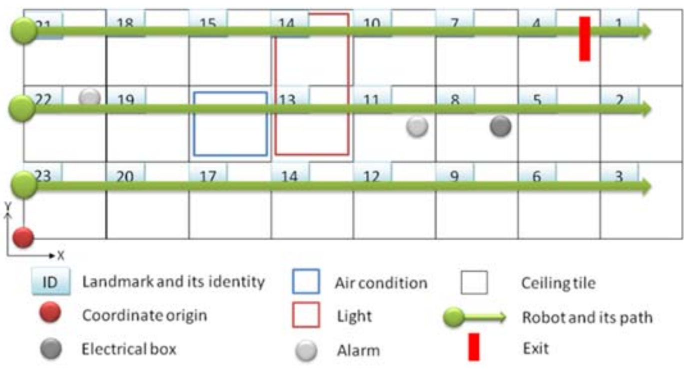
An experimental system (Fig. 2.1) has been built for testing our landmark design and recognition algorithm in an indoor environment. A simple real-time localization scheme is adopted in our experiments. The proposed landmark design and recognition algorithm has been verified through several experiments.

2.3.1 Experiment Setting

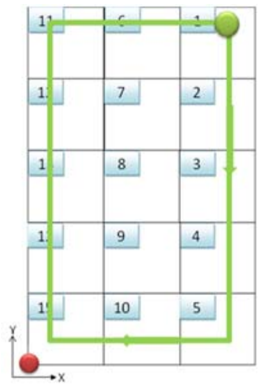
A Pioneer3-DX mobile robot is used as the carrier. A Logitech HD Pro C910 webcam is mounted on the top of the robot to capture the landmarks attached to the ceiling for robot localization. An onboard laptop computer, with Intel Core 2 Due 2.0 GHz CPU and 2G RAM, is used to control the movement of the mobile robot, receive and process the images from the camera, and estimate the robot's position.

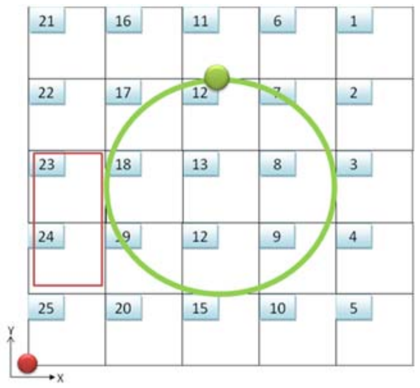
The experiment is performed in a real indoor environment with a set of 5-bit landmarks. Each landmark is printed on a piece of regular paper, attached on the ceiling and facing towards the floor. The distance between each pair of neighboring landmarks is 2 feet, measured carefully by hand. The ceiling of the testing environment is considered a plane and parallel to the flat floor.

To evaluate the performance of proposed localization scheme, we conduct experiments with different settings. The positions of the landmarks follow a grid pattern as shown in Fig. 2.12, and the tests have been conducted with three different types of robot paths, i.e. linear, rectangle and circular.



(a)





(b) (c)

Figure 2.12: Experiments setting.

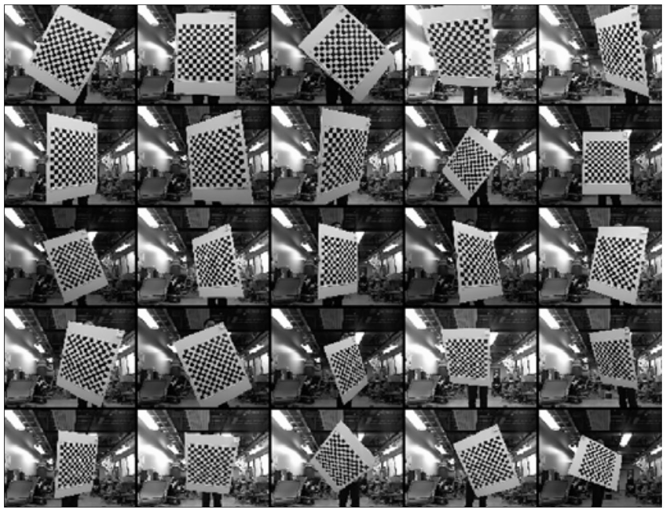


Figure 2.13: Calibration images.

The camera is calibrated before the experiments to provide the parameters needed by the localization algorithm, using the calibration toolbox mentioned in the section of Section 2.1.1

with a printed black/white checkerboard pattern. Fig. 2.13 shows a set of images for the webcam calibration. The calibration results can be found in Table 2.1 in the unit of pixels.

Table 2.1: Calibration results of camera.

Parameter	Focal Length	Image Center	Image Size
Value (Pixels)	531.5	(319, 228.5)	(640, 480)

2.3.2 Localization Technique

In order to achieve real-time localization, we use an intuitive, but effective, approach to recover the robot global coordinates. We notice that the coordinates in the image plane and the ceiling follow a consistent geometric mapping relationship.

The orientation of the camera is decided by the geometric relationship between the image and ceiling plane. As mentioned in Section 2.1.1, the camera is fixed on the robot and faces upward to the ceiling. Thus, the image and ceiling planes are parallel to each other. We can use the ceiling plane as the reference for the orientation calculation. Fig. 2.14 shows the orientation of camera is decided by the rotation of a vector between any two landmarks from the image to global (ceiling) coordinates, where the blue dots denote two landmarks L1 and L2, X'Y' denotes the global coordinate system translated to landmark L1, U'V' denotes the pixel coordinate system translated to landmark L1, ∠X'L1L2 denotes the global orientation, ∠U'L1L2 denotes the local orientation, and ∠X'L1U'denotes the orientation of camera.

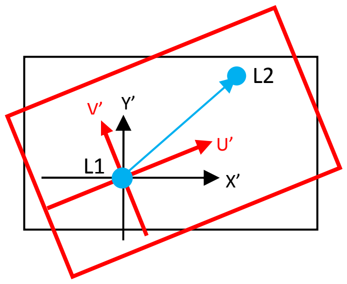


Figure 2.14: Relationship between the image and global coordinate system.

Algorithm 2 shows the process of calculating the orientation, where LDPixelCoor denotes the pixel coordinates of landmarks, LDGlobalCoor denotes the global coordinates of landmarks, and LDNum denotes the number of landmarks. The input of this algorithm is the pixel and global coordinates of landmarks. Line 4 calculates the global and pixel orientations of the vector, from ith to jth landmark, by the arctangent function.

Algorithm 2. Orientation Calculation

```
0.  Inputs: LDPixelCoor, LDGlobalCoor,
1.  Initialize: k=1
2.  for i=1 to LDNum-1 do
3.      for j=i+1 to LDNum do
4.          Decide GlobalOri and PixelOri for landmark i and j
5.          Ori(k) = ConvertToPI(GlobalOri - PixelOri )
6.      end for
7.      k=k+1
8.  end for
9.  return Mean(Ori)
```

The offset from the principle point to a landmark is determined by the image coordinates and the orientation of the camera. As shown in Fig. 2.15, O denotes the optical center of the camera, UV denotes the image coordinate system, XY denotes the global coordinate system (ceiling), C denotes the principle point, A denotes a landmark, and A' denotes the projection of A.

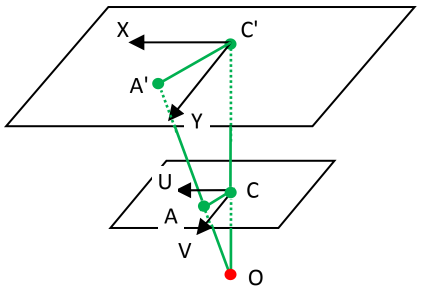


Figure 2.15: Relationship between the image and global coordinate system.

Given the pixel coordinates of the landmarks, the real offsets from the landmarks to the camera are estimated by the offset maps, including x and y offset maps. Each map is a 2-D matrix with the same dimension of the image. Each position in both offset matrices corresponds to the position of pixel in the image. Each value in both offset matrices denotes the real offset from the corresponding pixel to the principle point.

These two offset maps are generated by the following steps. First, we mount a number of markers within the field view of camera on the ceiling at known positions. Then we fix the camera at a known position beneath those markers and capture one image. Finally, the positions of those markers are tagged in the image coordinate system. Thus the real offsets from those markers to the principle point are estimated by the positions of markers in the pixel and global frames.

However, the total number of pixels for each image captured by the camera is much larger than the markers we tagged. A simple linear interpolation is applied to generate the offset maps. A triangle mesh is created by Delaunay triangulation [107], where those tagged markers are adopted as the vertices. Then the Cartesian coordinates of each pixel are transformed to barycentric coordinates [108], which represent the Cartesian coordinates of each pixel by the Cartesian coordinates of the vertices of the triangle, based on the Delaunay triangle mesh. The values of the barycentric coordinates are limited to [0, 1], which in turn limit that each pixel is only represented by the vertices of the triangle which contains the pixel. The real offsets are easily decided by the barycentric coordinates.

The position of the camera is estimated by the relationship between the image and global frames. Algorithm 2 decides the orientation between the pixels and global coordinates. The position of the principle point in the image frame is decided by the camera calibration toolbox mentioned in Section 2.3.1. The positions of the landmarks in the image frame are calculated using the method presented in Section 2.2. The positions of the landmarks in the global frame are pre-defined in Section 2.3.1. Thus, this geometric problem can be described as follow:

Given
1) The position of the principle point in the image frame,
2) The positions of the landmarks in the image and global frames,
3) The angle between the image and global frames;
Find
The position of the principle point in the global frame.

Algorithm 3 shows the process to calculate the absolute position of the camera, where PP denotes the principle point, MapX denotes the offset map of x, and MapY denotes the offset map of y. Line 3 decides the pixel offsets from the landmark to principle point in UV frame. Line 4 finds out the real offsets from the landmark to principle point by offset maps. Line 5 calculates the real offsets from the landmark to principle point in the global frame by rotation. Lines 6-7 estimate the position of the camera based on the position of the landmark in the global frame. Lines 2-8 will iterate all the identified landmarks. Line 9 returns the mean value of all the calculated positions of the camera.

Algorithm 3. Position Calculation

```
0. Inputs: LDPixelCoor, LDGlobalCoor, Ori, PP, MapX, MapY
1. CoorRotation= [ cos(-Ori), sin(-Ori); -sin(-Ori),cos(-Ori) ]
2. for i = 1 to LDNum do
3.     Get UVoffset from LDPixelCoor, LDGlobalCoor and PP
4.     Xyoffset = [ MapX(UVoffset(1)), MapY(UVoffset(2))]
5.     XyoffsetGlo = CoorRotation * XYoffset
6.     Get position of current landmark from LDGlobalCoor
7.     Pos = CurrLD + XYoffsetGlo;
8. end for
9. return Mean(Pos)
```

2.3.3 Experimental Results

A series of indoor-environment experiments have been conducted with our localization system to validate the proposed landmark design, recognition algorithm and localization technique. Fig. 2.16 and 2.17 show two typical results of our landmark recognition algorithm. In each figure, the left image shows the recognition result on grayscale image, including the identities and centers; the right image shows the result of image clustering, where the white region represents the light source, the gray region represents the background, and the black regions are regions of interest. Fig. 2.16 and 2.17 proves the effectiveness of the proposed landmark design and recognition scheme under extreme lighting condition and disturbance of other objects.

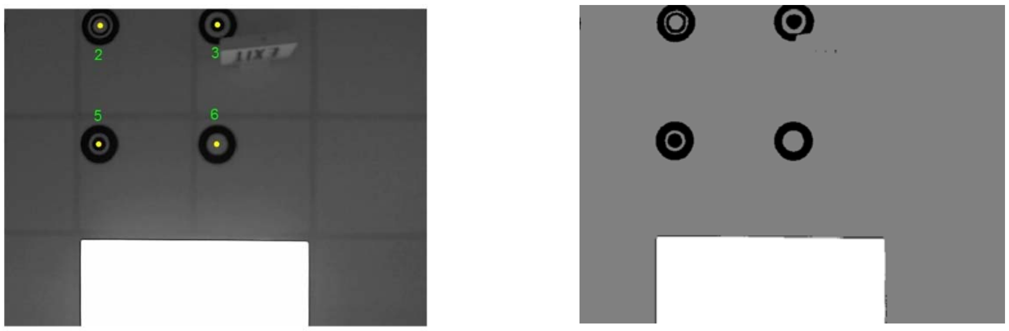


Figure 2.16: Landmark recognition with lighting affection and incomplete landmark.

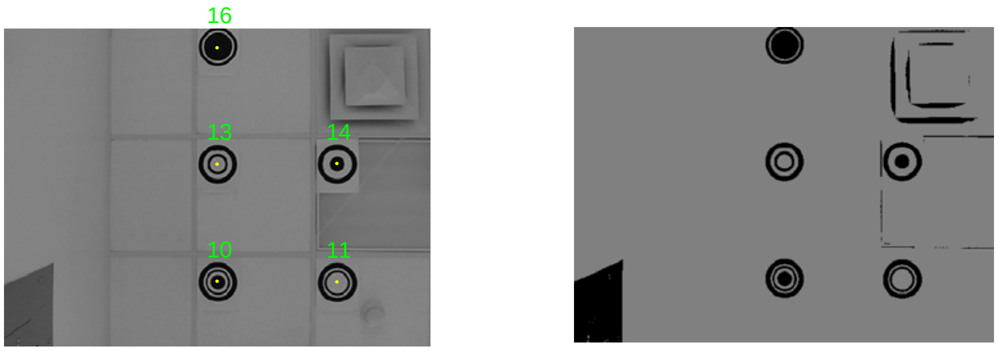


Figure 2.17: Landmark recognition with the disturbance of other objects.

To evaluate the performance of proposed localization technique, three sets of experiments are conducted through different robot paths. The results are shown in Fig. 2.18-2.20, where the green dots denote the true positions of the robot/camera, the blue dots denote the position estimated by the localization system and the red lines denote the robot/camera orientations.

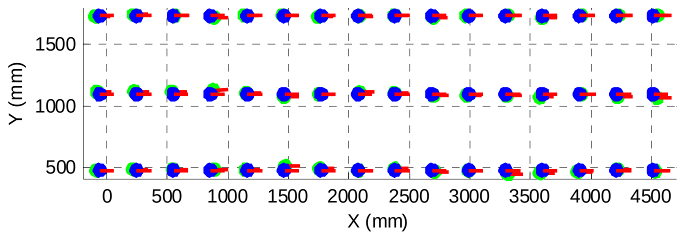


Figure 2.18: Localization errors from the linear paths.

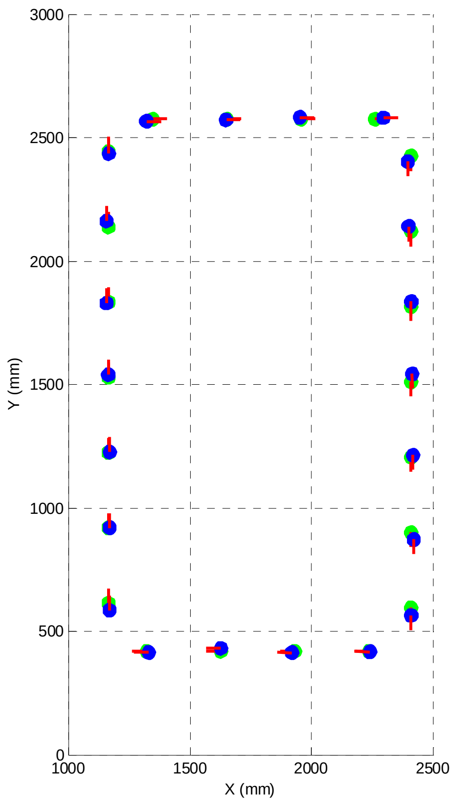


Figure 2.19: Localization errors from the rectangular path.

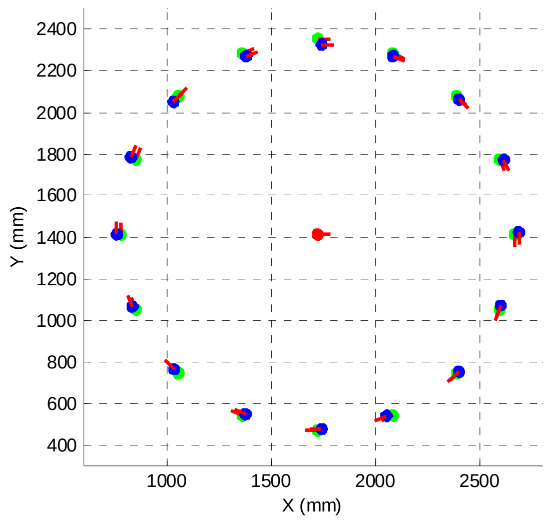


Figure 2.20: Localization errors from the circular path.

The overall accuracy of detection and identification is shown in Table 2.2. The detection accuracy is calculated by the number of landmarks detected divided by total number of landmarks captured by camera. The identification accuracy is calculated by the number of landmarks correctly identified divided by the number of landmarks detected. Those landmarks with more than half incompleteness are not included.

Table 2.2: Detection and identification accuracy.

Detection Accuracy	98%
Identification Accuracy	99%

Fig. 2.18-2.20 show that our proposed real-time low-cost localization scheme provides reliable and accurate pose estimation for the mobile robot. The average and standard deviation of the position and the orientation errors are listed in the Table 2.3. The overall average accuracy of localization is 20.2 mm and 0.5 degree based on the current system, which is sufficient for most of the regular-scale indoor mobile robot tasks.

Table 2.3: The position and orientation errors.

Experiment		Lines	Rectangle	Circle
Position Error	Average (mm)	20.6	17.4	23
	STD (mm)	9.8	10.6	7.2
Orientation Error	Average (deg)	0.3	0.5	0.9
	STD (deg)	0.2	0.4	1.1

2.4 Conclusion

This chapter presents a vision/artificial landmark-based real-time low-cost mobile robot localization scheme. In order to minimize the influence of different lighting conditions, we encode identities of the landmarks with the concentric rings in black and white. Based on the stable encoding information and nature omni-directionality of the concentric rings, a fast two-layer landmark recognition algorithm is proposed to detect and identify the landmarks captured by a single regular camera. The reported experimental results show that the proposed approach facilitates fast accurate localization for mobile robots in indoor environments.

Chapter 3 Maintaining Wireless Communication Coverage

This chapter addresses the critical issue of maintaining desired wireless communication connectivity in a team of collaborative mobile robots, which is highly demanded for reliable functioning of multi-robot systems but challenging in realistic environments. The signal propagation of wireless communications among mobile robots is affected by not only the transmission power and distance but also obstacles and other environmental conditions as well as robot movement, which result in signal loss, attenuation, multi-path fading and shadowing. Consequently, the communication condition among mobile robots in a physical environment is usually unstable, and it is difficult to accurately predict the actual communication ranges of robots.

Decentralized control strategies are proposed, which are based on perceived link quality, to maintain desired wireless communication connections among mobile robots. Proposed controllers are designed based on intelligent control frameworks, fuzzy logic, artificial neural network and radial basis network. Back propagation and reinforcement learning are presented to provide the ability of self-tuning.

The layout of this chapter is as follow. Section 3.1 will introduce the fundamental considerations and simulation setups. Section 3.2 will present proposed fuzzy logic controller. Section 3.3 will discuss artificial neural network controller based on back propagation learning. Section 3.4 will define the fuzzy neural network based on back propagation learning. Section 3.5 will explain the specific definition of the proposed reinforcement learning radial basis network based on online link quality measurement. Section 3.6 will describe the fuzzy neural network based on reinforcement learning. Section 3.7 will compare proposed intelligent controller and summarize the contribution of this chapter.

3.1 Fundamental Considerations and Simulation Setups

In this section, we discuss the fundamental considerations and simulation setups for all proposed control schemes.

3.1.1 Fundamental Considerations

It is challenging to maintain wireless communication connections among mobile nodes in many realistic situations. The unstable link quality of wireless communication in realistic environments can be captured by several metrics in different types of wireless networks [46,50], such as Received Signal Strength Indication (RSSI) -- the signal strength is observed at the receiver's antenna during packet reception, Signal-to-Interference-plus-Noise Ratio (SINR) -- the extent to which power of the received signal exceeds sum of noise plus interference at the receiver, Packet-Delivery Ratio (PDR) -- the ratio of the correctly received packets at receiver to the total number of packets sent by the sender, Bit-Error Rate (BER) -- the ratio of bits with errors to the total number of bits that have been received over a given time period, and Link Quality Indicator (LQI) -- the quality of the data packets received by a receiver.

In this dissertation, we use the Received Signal Strength Indicator (RSSI) as the metric of the wireless link quality, which is a measurement of the power that presents in a received radio signal. RSSI value fluctuates in a real environment due to the signal power decay, reflection/diffraction/scattering, shadowing and fading. RSSI is one of the several common metrics for wireless link quality. RSSI is used in this experiment simply because it is directly available for wireless networks of IEEE 802.11 (Wi-Fi) and IEEE 802.15.4 (ZigBee) protocols, which provides the convenience for experimental verification. Though we derive the following discussions based on RSSI, the proposed intelligent controllers present the general control structure in which RSSI can be replaced with different metrics of wireless link quality.

Using RSSI as the indicator of the wireless link quality, we consider the range of all possible RSSI values as the input space of proposed control schemes. We recognize that different communication settings or environments result in different ranges of RSSI values, and different applications may have different desired ranges of RSSI. In order to device the intelligent controllers in a unified manner, we transform the RSSI values under a specific scenario to the same range, which is then taken as the actual input space.

The neighborhood of a robot R_i includes those robots that have direct influence on R_i's motion control. One necessary condition is that R_i must be able to communicate with its neighbors to exchange the data of motion states. The graph of Delaunay Triangulation for a set of points [13, 19, 22, 107], in which no point is inside the circumcircle of any triangle defined by any three other points, is used to form a topologically hole-less coverage of the environment and assist the selection of a small number of most relevant robots in order to limit the communication complexity. We define the Delaunay neighborhood of a robot R_i as the set of its one-hop topological neighbors on the Delaunay graph. In this way, the neighborhood relationship among the robots is symmetric, i.e. if R_j is R_i's one-hop neighbor, then Ri must be R_j's one-hop neighbor. Moreover, by using one of the distributed graph generation algorithms [109], a partial Delaunay graph for R_i and neighboring robots, which is generated onboard by Ri, is indeed part of the global graph on the whole multi-robot system. All these features facilitate the realization of distributed control.

3.1.2 Simulation Setups

All proposed distributed control schemes for maintaining the desired communication coverage in a multi-robot deployment scenario have been evaluated through simulations. The settings of the simulations are described as below.

We simulate the uncertain wireless communication conditions in an environment with a widely-adopted probabilistic wireless signal propagation model, the log-distance path loss model [45], which states that the transmitted power of the signal can in general be expressed as

$$P_r(d) = P_t - PL(d_0) - 10n\log(\frac{d}{d_0}) - X_\sigma \tag{3.1}$$

where $P_r(d)$ is the received signal power at the distance d, and P_t is the transmitted power of signal, $PL(d_0)$ is the path loss at d_0, d is the distance between the transmitter and receiver, n is known as the path loss exponent, and X_σ is a Gaussian random variable with zero mean and standard deviation σ, which reflects the attenuation caused by flat fading. Here, P_t is constant, given the signal transmission power; and $PL(d_0)$ is constant, given the environment and reference distance.

By defining $P_r'(d) = P_r(d) - P_t(d) + PL(d_0)$ as the relative received signal power, we obtain from Eq. (3.1).

$$P_r'(d) = -10\ n\ \log(\frac{d}{d_0}) - X_\sigma \tag{3.2}$$

Since Pr'(d) is well represented by RSSI, to test the proposed control scheme, we simulate RSSI in Eq. (3.2) directly instead of Pr(d). We set the minimum distance between robots as the reference distance d_0.

To evaluate the performance of the proposed control schemes in different signal propagation conditions, we conduct the simulations in two groups of path loss exponent (n) and standard deviation (σ), shown in Table 3.1.

Table 3.1: Simulation signal propagation conditions.

Lines	Set 1			Set 2		
path loss exponent (n)	3.0	3.0	3.0	1.6	2.2	3.0
standard deviation (σ)	3.0	7.0	9.6	7.0	7.0	7.0

Besides that, we recognize that different communication settings or environments result in different ranges of RSSI values, and different applications may have different desired ranges of RSSI. In order to compare the results in a same range, we normalize the RSSI values under a specific scenario into the same range.

In the simulations of this chapter, we consider the scenario that 50 mobile robots deploy into each of the environments mentioned above to establish and maintain the desired communication coverage. These robots initially gather with random positions and orientations in a 5-by-5 square planar region, following the uniform distribution.

3.2 Fuzzy Logic

In the proposed approach of this section, each robot is controlled by a local fuzzy logic controller; a rule is defined for each robot to map the wireless link quality with a neighbor to its communication-maintaining movement relative to the neighbor; the final decision of robot movement is made by aggregating the outputs of the rules for all its neighbors. The establishment and maintenance of the system-wide communication connectivity result from the combined effect of all the local fuzzy logic controllers on individual robots.

The layout of this section is as follows. To be self-contained, Section 3.2.1 will briefly review the fuzzy inference system structure adopted by our work. Section 3.2.2 will explain the specific definition of the proposed fuzzy logic controller for communication maintenance based on online link quality measurement. Section 3.2.3 will report the simulation results based on a well-adopted probabilistic communication signal propagation model. Section 3.2.4 will show the experiment results by using Pioneer P3Dx and Amigobot robots. Section 3.2.5 will summarize the contribution of this section.

3.2.1 Review of Fuzzy Inference

Fuzzy inference is a method that interprets the values in the input vector and, based on a set of rules, assigns values to the output vector [58-60]. The basic components of a fuzzy inference system include fuzzy sets, membership functions and rules:

1) A fuzzy set is a set without a clearly defined boundary. It can contain elements with only a partial degree of membership. This degree of membership is also called the membership value, and defined between 0 and 1.
2) The mapping of each point in the input space to a membership value defines a membership function.
3) Each rule is defined in the form of an if-then statement, and decides an output fuzzy set based on the input vector.

A typical fuzzy inference process consists of five steps in the following order:

1) Fuzzification of inputs: To start a general fuzzy inference process of multiple inputs, one takes each input and determines the degree to which it belongs to each relevant fuzzy set, based on the definition of the associated membership function. In this manner, each input is fuzzified over all the associated membership functions required by the rules.

2) Fuzzy operation for each rule: If the antecedent of a given rule involves more than one fuzzy set, the fuzzy operators are applied to obtain one number out of all the relevant membership values to represent the result of the antecedent for that rule.
3) Implication for each rule: The consequent of a rule is a fuzzy set represented by a membership function which is reshaped using a function associated with the outcome of the antecedent.
4) Aggregation of outputs: The fuzzy sets that represent the outputs of all the rules are combined into a single fuzzy set.
5) Defuzzification: A single output value is resolved from the output fuzzy set as the final decision.

Under the envelope of the above general fuzzy inference process, in this paper we use the so-called Sugeno, or Takagi-Sugeno-Kang, method of fuzzy inference [58-60] to define the proposed fuzzy logic controller, due to its match with the intended control scheme. The operation flow of a Takagi-Sugeno rule is shown in Fig. 3.1. The key characteristics of the Takagi-Sugeno fuzzy inference include:

1) The output membership function of each rule is downsized from a fuzzy set to a singleton in the form of Eq. (3.3), Which can be either a linear function of the inputs or a constant, where y_k denotes the output of the kth rule, x_i denotes the ith input, and a_i and c_k are constant.

$$y_k = \sum_i a_i x_i + c_k \tag{3.3}$$

2) The output level of each rule is weighted by a firing strength of the rule, w_k, which is defined by the fuzzy operation result of the antecedent.
3) The final output of the system is the weighted average of all the rule outputs, i.e.

$$z = \sum_k (w_k y_k) \Big/ \sum_k w_k \tag{3.4}$$

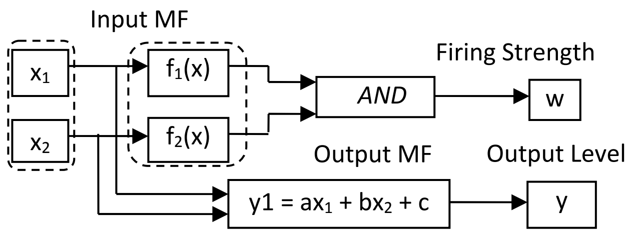


Figure 3.1: Operation flow of a Takagi-Sugeno rule with 2 Inputs.

3.2.2 Fuzzy Controller for Communication Maintenance

The control objective of this work is to deploy a multi-robot system into an unknown environment to establish and maintain desired wireless communication coverage. Due to its high adaptability and scalability, we take a distributed scheme to control the deployment process. That is, we define a local fuzzy logic controller on each robot to control its movement to adapt to the quality of the wireless links with its neighbors; it is the combined effect of all robots' movement that will allow the multi-robot system to approach and maintain the desired communication connectivity. The following discussions will focus on the definition and functionality of the local, onboard fuzzy logic controller, including fuzzy sets, fuzzy rules and fuzzy inference process.

3.2.2.1 Fuzzy Sets

As mentioned above, the control objective is to approach and maintain a desired level of wireless link quality among neighboring robots. In general, poor wireless link quality means the risk of disconnection or packet loss, while overly good wireless link quality often corresponds to small space coverage. We define five fuzzy sets out of the universe of discourse, from the losing to crowding RSSI, corresponding to a linguistic division of the RSSI levels:

1) Desired RSSI—the range of RSSI indicating that the wireless quality is desirably good for the underlying application.

2) Losing RSSI—the range of RSSI indicating that the wireless quality is poor. If the RSSI is lower than this level, the received data packets may not be decoded, which leads to the loss of the associated neighbor.
3) Weak RSSI—the range of RSSI indicating that the wireless quality is getting poor. It is the transitional range between the desired and losing RSSI.
4) Crowding RSSI—the range of RSSI indicating that the wireless quality is overly good. It means that the robots are probably crowded together.
5) Strong RSSI—the range of RSSI indicating that the wireless quality is above the desired level and getting overly good. It is the transitional range between the desired and crowding RSSI.

Arranging these fuzzy sets in the order of ascending RSSI strength, we have losing, weak, desired, strong and crowding RSSI. Corresponding membership functions for these fuzzy sets can be defined according to the functionality of onboard wireless transceiver and requirements of applications. One example is provided in Section 3.2.3.1.

3.2.2.2 Fuzzy Rules

The core of fuzzy inference is to map an input space to an output space, and the primary mechanism for doing this is a list of if-then statements known as rules. To design the proposed fuzzy logic controller, we need to map the control laws to the rules.

The proposed fuzzy logic controller for each robot contains a set of individual rules for the robot to react to each specific RSSI level with each neighbor; the consequents of all the rules will be combined to decide the comprehensive response that the robot should make to adapt to the quality of all the neighboring wireless links, which is flexible to the number of neighbors.

In the proposed fuzzy logic controller, each individual rule has the form of "If the value of RSSI that robot R_i senses with a neighbor R_j is at the level L_R, then R_i should move towards R_j with a certain step length D", where L_R denotes any of the previous-defined five RSSI fuzzy sets. The step length of the robot movement towards a neighbor is chosen according to the level of RSSI with that neighbor. In order to maintain the desired RSSI with a neighbor, a robot should move away from the neighbor with large RSSI and move close to the neighbor with small RSSI. Correspondingly,

1) If RSSI is at the losing level, then R_i moves towards R_j with D = L;
2) If RSSI is at the weak level, then R_i moves towards R_j with D = 0.3L;
3) If RSSI is at the desired level, then R_i does not move;
4) If RSSI is at the strong level, then R_i moves towards R_j with D = - 0.3L;
5) If RSSI is at the crowding level, then R_i moves towards R_j with D = - L.

Thus, the consequent of each rule is a motion decision vector **D** which requires R_i to move a distance D along the straight line connecting R_i and R_j. Specifically in the planar environment, $\mathbf{D} = [D * \cos(\theta_{ij}), D * \sin(\theta_{ij})]^T$, where θ_{ij} denotes the orientation angle of the line connecting R_i and R_j in the frame of reference of the environment.

The above five rules are defined for R_i to react to the RSSI with R_j. When R_i has multiple neighbors, these five rules apply to each of the neighbor.

3.2.2.3 Fuzzy Inference Process

The proposed fuzzy logic controller adopts the Takagi-Sugeno fuzzy inference method, and consists of the following steps:

1) Fuzzifying Inputs: The measured RSSI values between a robot R_i and all its neighbors will be at first transformed into the same universe of discourse. Then each

transformed RSSI will be used as the input to the five membership functions to obtain the corresponding membership values, with $f_{i,j,k}(RSSI_{i,j})$ denoting the value of the kth membership function associated with R_i and R_j.

2) Calculating firing strengths: The firing strength for each rule is the result of the fuzzy operation on the antecedents of the rule. In our case, the antecedent of each rule has only one part, thus the firing strength $w_{i,j,k} = f_{i,j,k}(RSSI_{i,j})$.

3) Implicating consequents: As defined in Section 3.2.2.3, the consequent of each rule is a motion decision vector $\mathbf{D}_{i,j,k}$ based on the measured RSSI.

4) Aggregating consequents: The comprehensive motion decision vector for R_i to adapt to all the neighboring wireless links is calculated by the weighted average across all the rules as Eq. (3.5), which defines the next movement that R_i should take.

$$\mathbf{D}_i = \sum_{j,k} [w_{i,j,k}\mathbf{D}_{i,j,k}] \Big/ \sum_{j,k} w_{i,j,k} \tag{3.5}$$

Since $\mathbf{D}_i$ is a single vector, no further defuzzification is needed. Fig. 3.2 shows the fuzzy inference diagram of a robot with six neighbors. In Fig. 3.2, $f_k(RSSI)$ denotes the kth input membership function, $y_k(RSSI)$ denotes the kth output membership function, $w_{i,j,k}$ denotes the firing strength for the kth membership function associated with R_i and R_j, $\mathbf{D}_{i,j,k}$ denotes a motion decision vector as the output of the kth membership function associated with R_i and R_j, and $\mathbf{D}_i$ denotes the final output for robot R_i. In this specific case, totally thirty rules are applied and then aggregated.

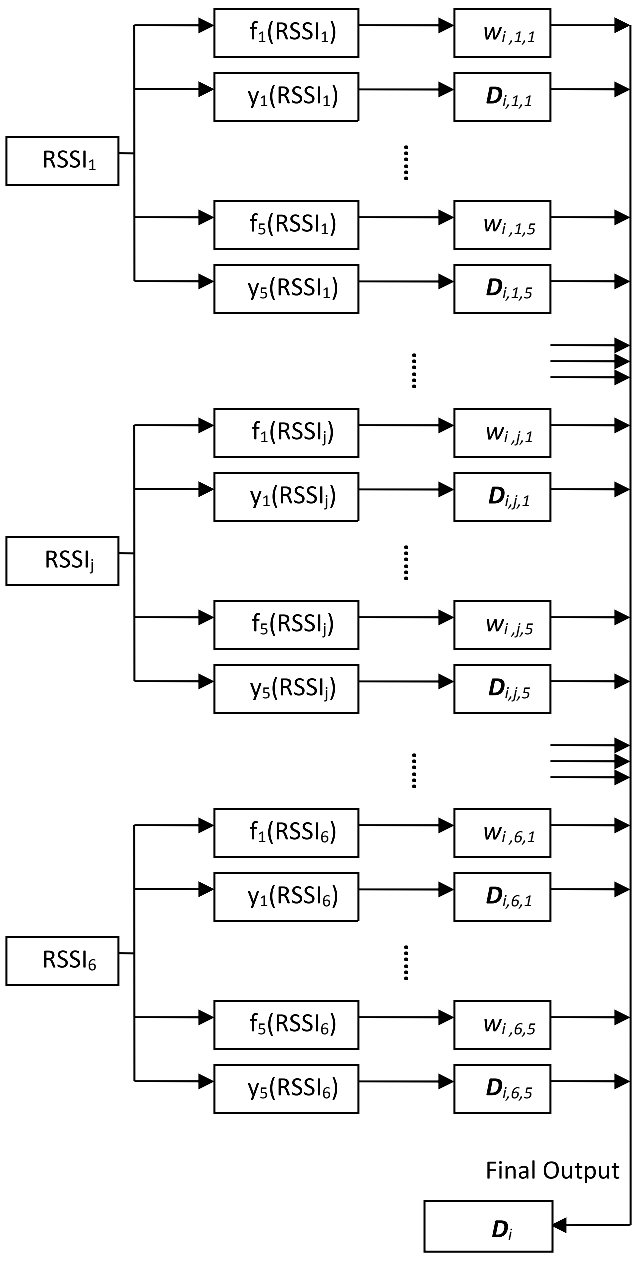


Figure 3.2: Fuzzy inference of a robot with six neighbors.

3.2.3 Simulations

The proposed fuzzy logic controller, together with the associated distributed fuzzy inference process, for establishing and maintaining communication coverage in a multi-robot deployment scenario has been evaluated through simulations. The definition of membership function and results of the simulations are reported as follows.

3.2.3.1 Definition of Membership Functions

Based on the log-distance path loss model mentioned above, we define a membership function for each fuzzy set (Fig. 3.3). Here trapezoid-shaped membership functions are adopted, because they provide an effective and simple description of the fuzziness at which we judge the RSSI level. In particular,

1) L (Losing): An RSSI $\in$[0,0.7] is considered as a losing RSSI. The membership degree of [0, 0.2] is set to 1, while [0.2, 0.7] is the transitional part.
2) W (Weak): An RSSI $\in$[0.2,2] is considered as a weak RSSI. The membership degree of [0.7, 0.8] is set to 1, while [0.2, 0.7] and [0.8, 2] are the transitional parts.
3) D (Desired): An RSSI $\in$[0.8, 2.9] is considered as a desired RSSI. The membership degree of [2, 2.4] is set to 1, while [0.8, 2] and [2.4, 2.9] are the transitional parts.
4) S (Strong): An RSSI $\in$[2.4, 5] is considered as a strong RSSI. The membership degree of [2.9, 3.7] is set to 1, while [2.4, 2.9] and [3.7, 5] are the transitional parts.
5) C (Crowding): An RSSI $\in$[3.7, 10] is considered as a crowding RSSI. The membership degree of [5,10] is set to 1, while [3.7,5] is the transitional part.

The peak and transitional ranges of these membership functions are chosen based on the RSSI resulting from the log-distance path loss model. The simulated RSSI values with different path loss exponent n and standard deviation σ are scaled into the same range [0, 10] for the proposed fuzzy logic control.

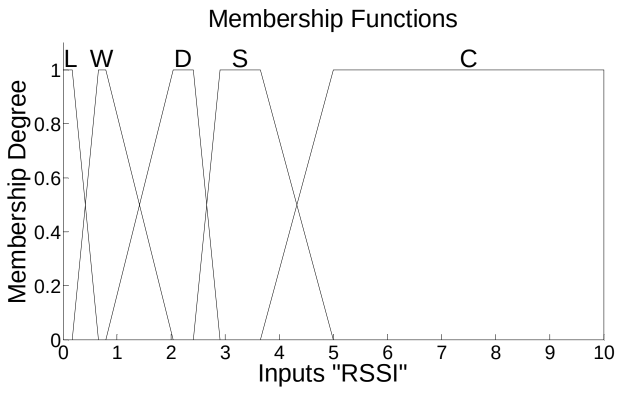


Figure 3.3: Membership Functions.

3.2.3.2 Simulation Results

Fig. 3.4 show the configurations of the 50-robot system after 2000 deployment steps under n=3 and σ=7. The system configuration and the underlying neighborhood relationship are visualized using Delaunay graph, where the robots are identified with numbers. It shows that our proposed scheme is capable of forming an effective network coverage under different levels of uncertainty in signal propagation.

Fig. 3.5 show the evolution of the average RSSI which is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig 3.5 also shows that the average RSSI in general converges towards the range of desired RSSI [2, 2.4] as the MRS approaches a communication coverage. Furthermore, it indicates that the average RSSI also has some uncertainty, reflected by the fluctuation of the average RSSI as time goes on. This fluctuation results from the reactive response of the fuzzy logic controller to the randomness in signal propagation, and the level of fluctuation increases as the uncertainty in signal propagation increases.

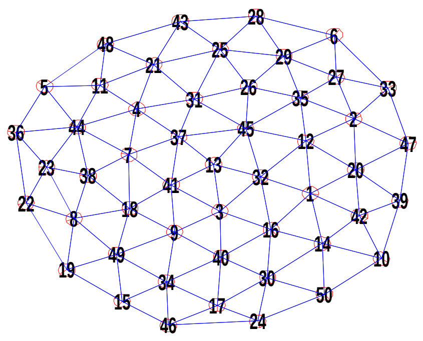


Figure 3.4: Deployment configuration when n=3.0 and σ=7.0.

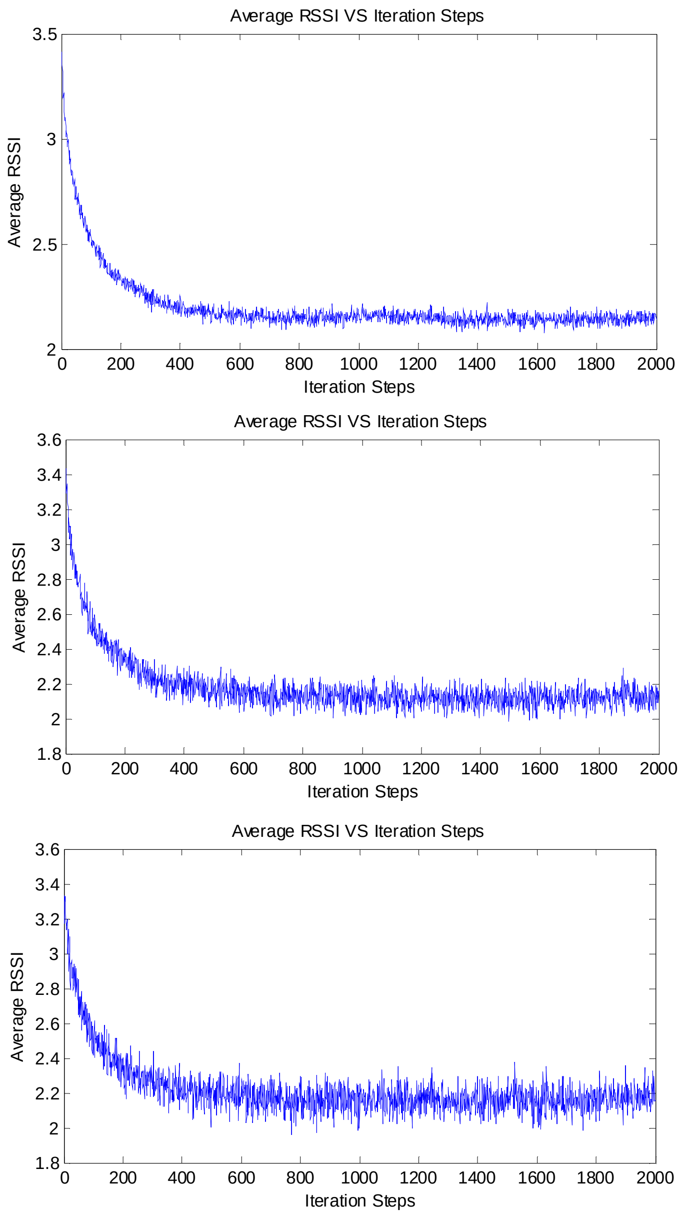
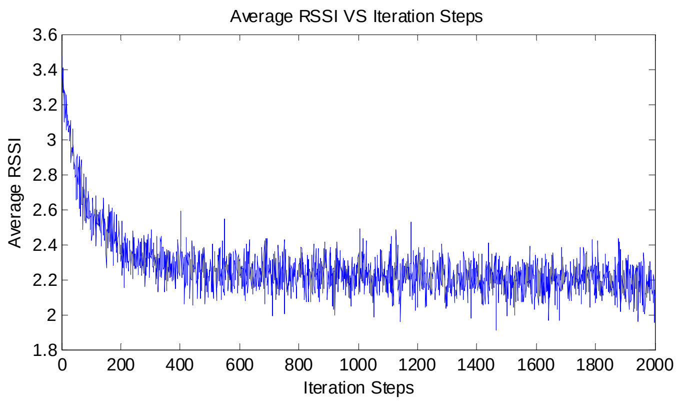
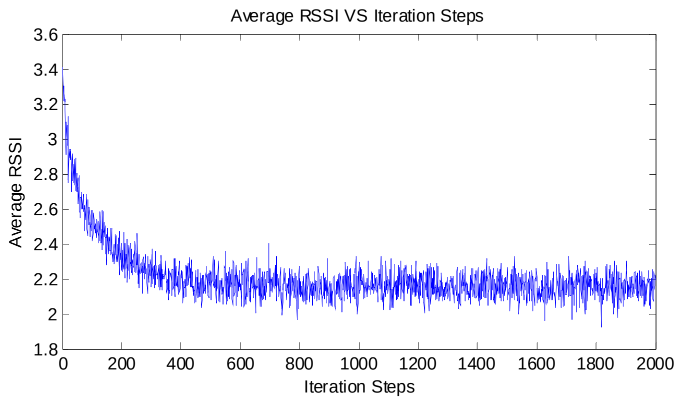


Figure 3.5: Average inter-robot RSSI when n=3.0 and σ=3.0, σ=7.0 and σ=9.6.

Fig. 3.6 show the evolution of the average RSSI under different path loss exponents. All of three figures show that the average RSSI in general converges to the range of desired RSSI, and the fluctuation of the average RSSI decreases as the path loss exponent increases.





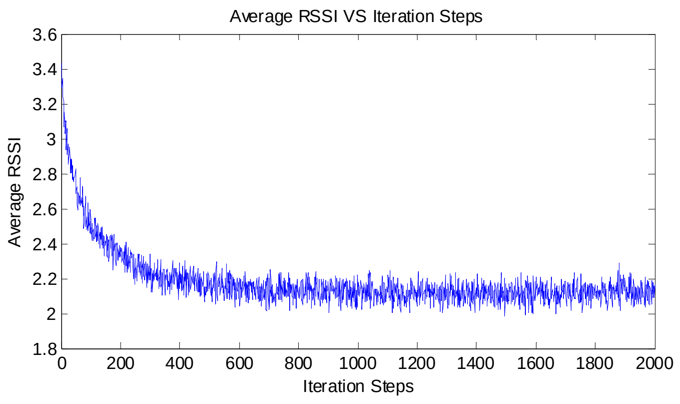
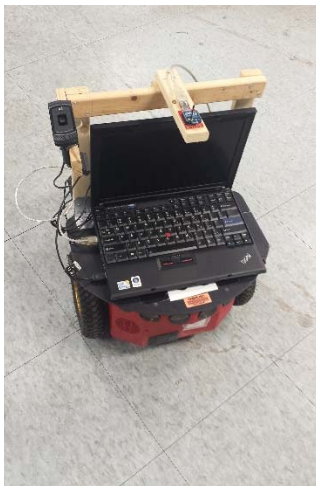
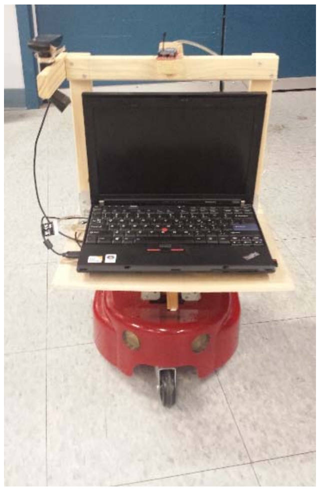


Figure 3.6: Average inter-robot RSSI when σ=7.0 and n=1.6, n=2.2 and n=3.0.

In summary, Fig. 3.4-3.6 show that, with the proposed control scheme, the MRS can form and maintain a communication coverage with desired average RSSI in different environments. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.2.4 Experiments

In this subsection, experiments are conducted to verify the effectiveness of proposed distributed fuzzy control schemes. We consider the scenario that 7 robots, include 4 Pioneer P3Dx and 3 Amigobot robots, deploy into an indoor environment to maintain desired communication coverage. Pioneer P3Dx (Fig. 3.7 (a)) and Amigobot (Fig. 3.7 (b)) robots are controlled by an onboard laptop, which is connected to a Logitech C910 webcam and an Xbee wireless module. The webcam, faces upward to the ceiling, is used to capture the picture for the localization algorithm, mentioned in Chapter 2. The wireless module (Xbee) is utilized to deliver the location (position and orientation) of robot to neighbors, and receive the information and collect RSSI values from neighbors. Fig. 3.8 shows our experiment system in an indoor environment, in which a set of artificial landmarks are attached to the ceilings for the localization algorithm.

(a) (b)

Figure 3.7: Pioneer P3Dx and AmigoBot robot with laptop, webcam and Xbee.

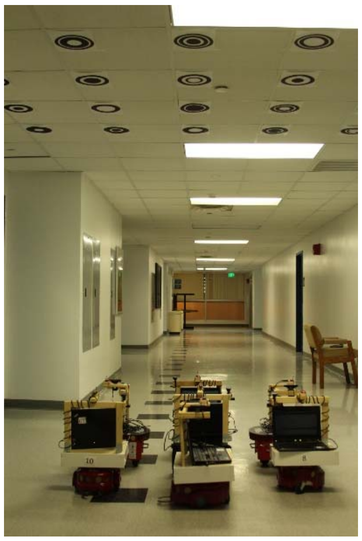


Figure 3.8: Experiment system.

The architecture of control system for the onboard laptop is shown in Fig. 3.9, which consists of three threads (yellow rectangles): robot task thread, Xbee (wireless communication) thread, and localization thread. After connected to the mobile robot, the thread of robot task receives the information packet from the mobile robot and sends the motion commands to it. The structure of robot task is pre-defined by the manufacturer as following: packet handler, sensor interpretation task, action task, state reflection task and user task. We insert our tasks into corresponding pre-defined tasks. The current information (position and orientation) of robot, obtained in packet handler, are broadcasted by Xbee wireless module. The list of neighbor is updated through the list of packets, which are received from Xbee module. The fuzzy controller of wireless communication maintaining decides the next motion vector based on the RSSI and orientation from the list of neighbor. In the task of state reflection, the absolute location are updated by the localization thread; and next motion vector are sent to the mobile robot. The above information is saved into onboard laptop to adjust the parameters of fuzzy controller.

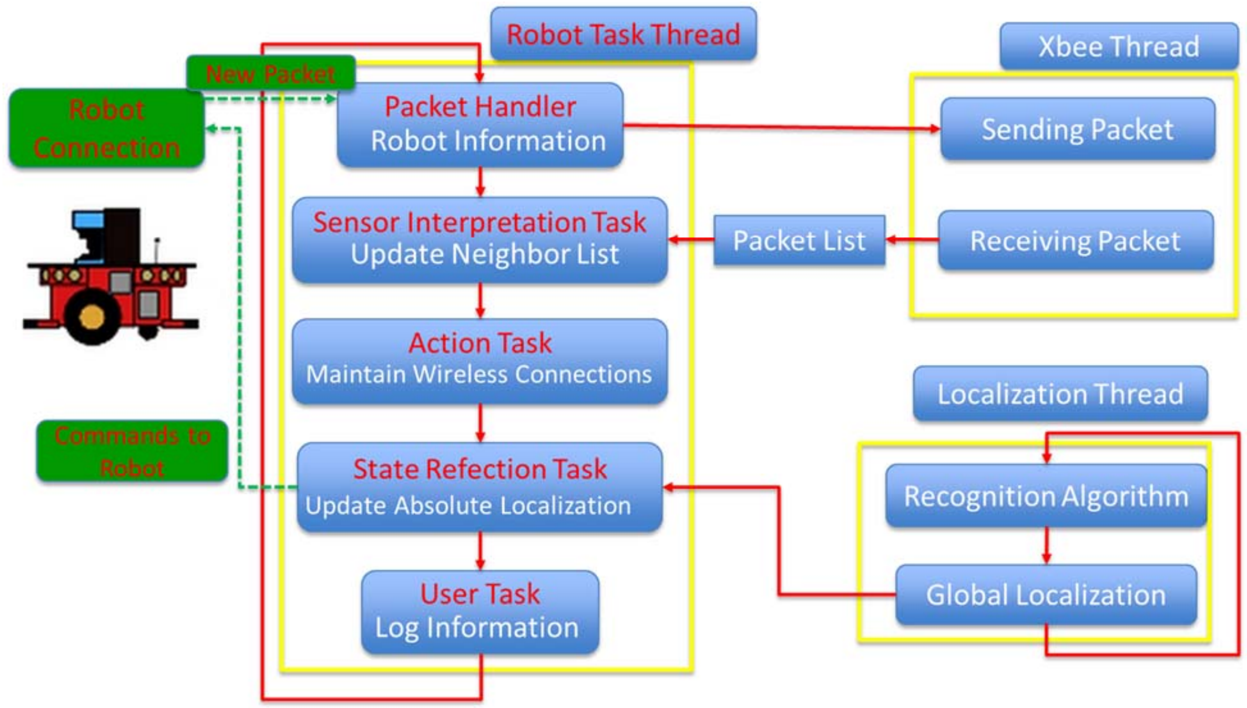


Figure 3.9: The architecture of control system.

The fuzzy controller is tuned manually by the method of try and error. A set of initial values for proposed fuzzy controller is selected based on the priori knowledge of target environment. The parameters of fuzzy controller are iteratively adjusted by the performance of controller, which is applied to three mobile robots to from a desired wireless coverage. The process of adjustment is terminated until the fuzzy controller can maintain the desired wireless communication connections among three mobile robots.

To evaluate the performance of the proposed control schemes in different initial configurations, we conduct the experiments in two sets of initial positions and orientations, shown in Fig. 3.10 and Fig. 3.11. Initial positions and orientations of setup2 is generated by random within 2-by-2 meters square planar region, following the uniform distribution.

The configurations of the 7-robot system after 30 seconds is shown Fig. 3.12 and 3.13, which indicate that proposed controller is capable of forming an effective wireless coverage under different initial conditions.

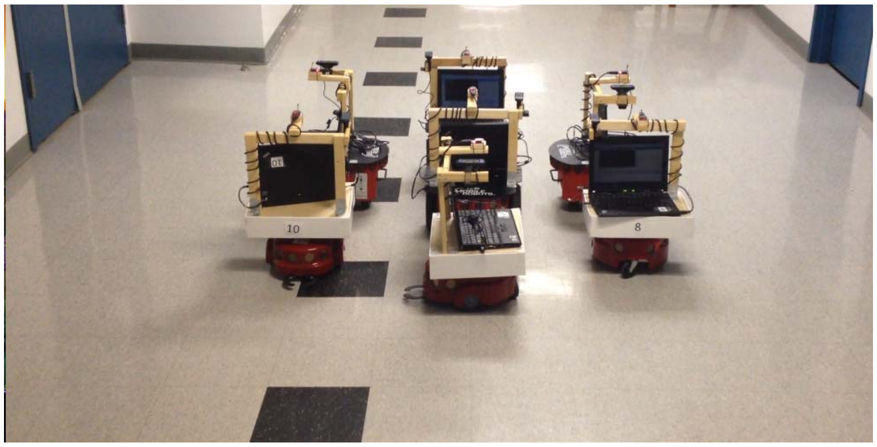


Figure 3.10: Initial configuration setup1.

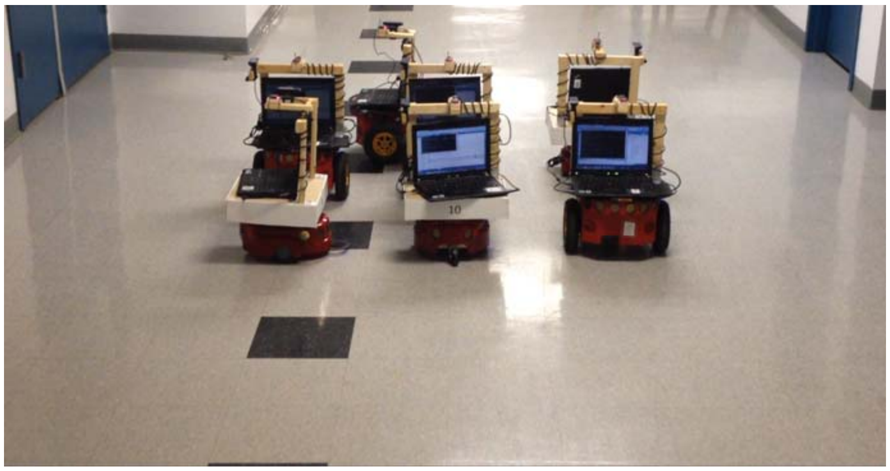


Figure 3.11: Initial configuration setup2.

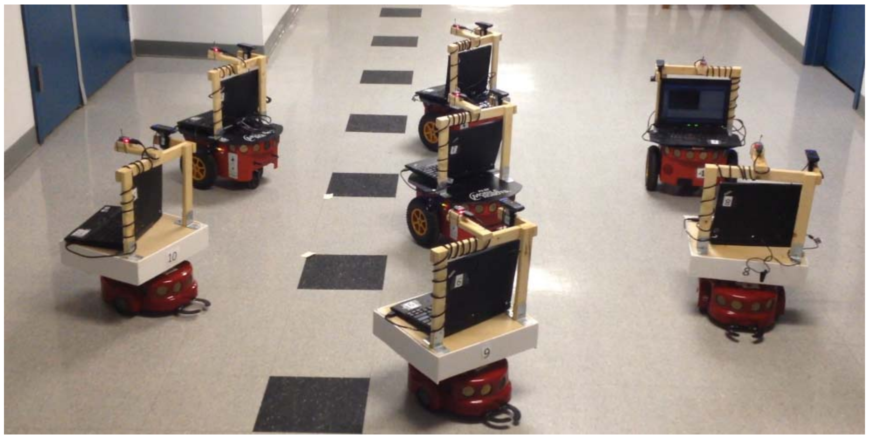


Figure 3.12: Deployment configuration after 30 seconds with initial setup1.

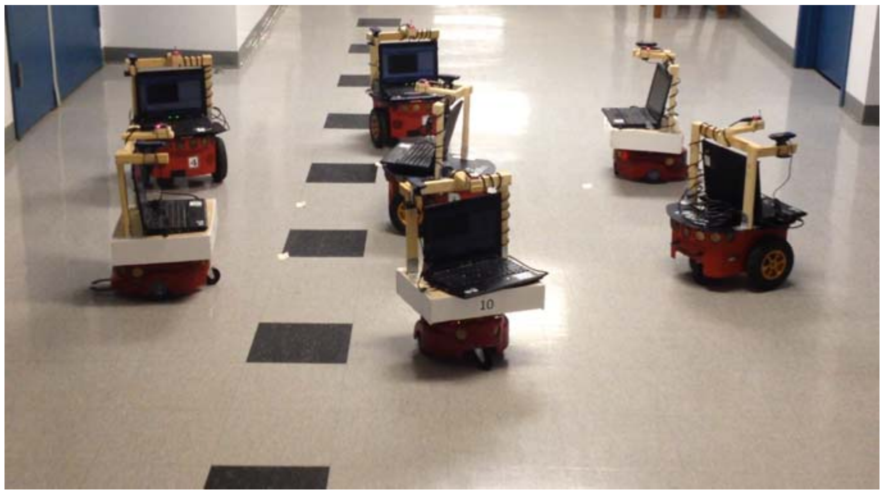


Figure 3.13: Deployment configuration after 30 seconds with initial setup2.

Fig 3.14 and 3.15 show the initial and deployment configurations in top view. The system configuration and the underlying neighborhood relationship are visualized using Delaunay graph, where the robots are identified with numbers. The evolution of the average RSSI is shown in Fig. 3.16 and 3.17, which is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig. 3.16 and 3.17 also shows that the average RSSI in general converges towards the range of desired RSSI [2, 2.4] as the MRS approaches a communication coverage.

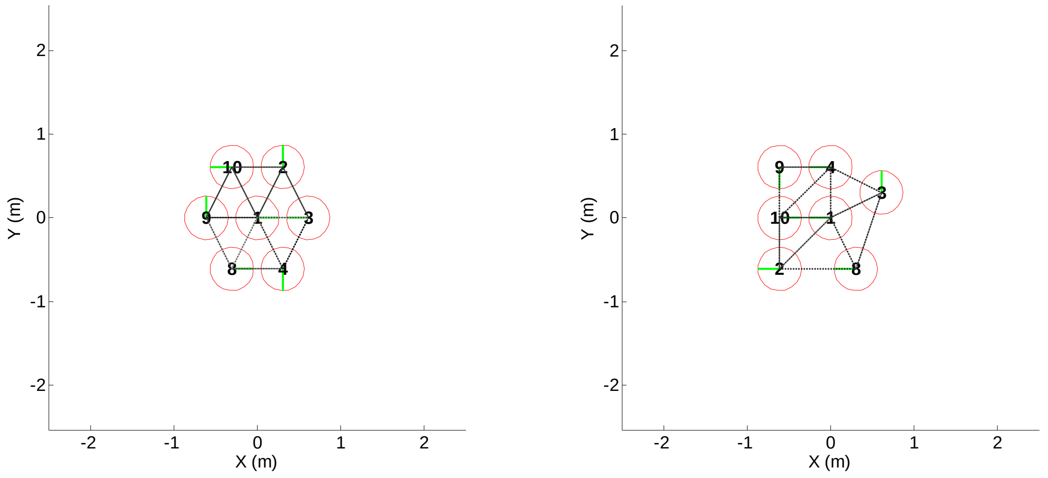


Figure 3.14: Initial configuration setup1 and setup2 in top view.

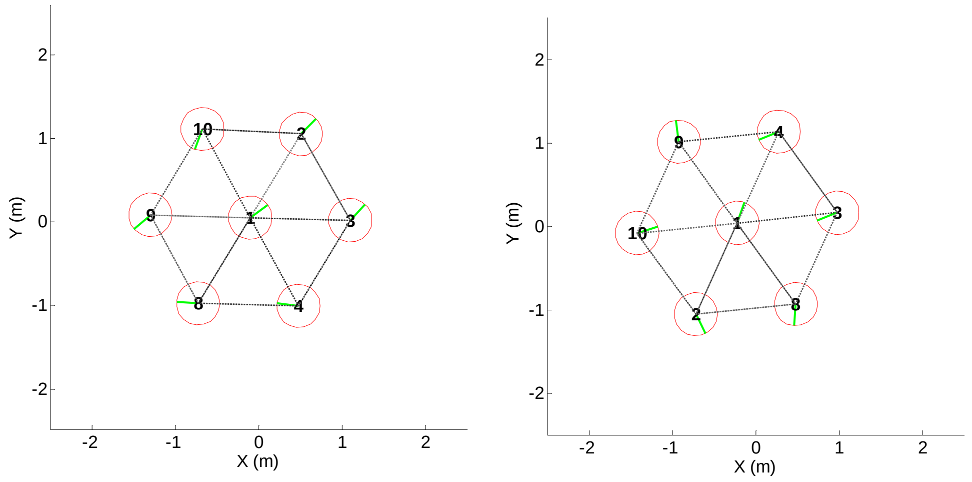


Figure 3.15: Deployment configuration after 30 seconds with initial setup1 and setup2.

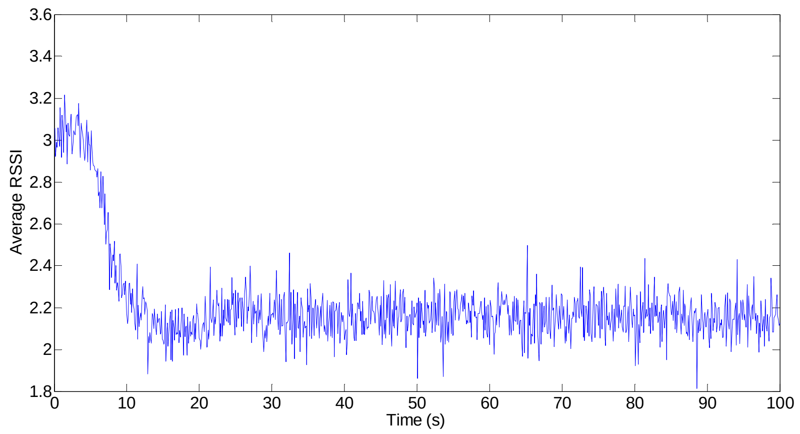


Figure 3.16: Average inter-robot RSSI with initial setup1.

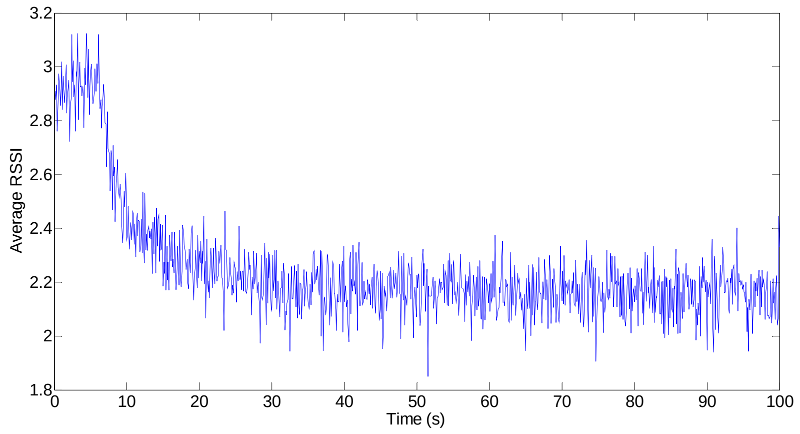


Figure 3.17: Average inter-robot RSSI with initial setup2.

In summary, Fig. 3.10-3.17 show that, with the proposed control scheme, the MRS can form and maintain a communication coverage with desired average RSSI in different initial configurations. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.2.5 Conclusions

This section proposes a distributed control strategy to establish and maintain wireless communication connections while forming a system coverage to an unknown environment. The proposed strategy defines a local fuzzy logic controller to onboard make decisions on the movement of each mobile robot to adapt to the real-time wireless link quality with its neighbors. The combined effect of all the local fuzzy control processes results in the desired communication connectivity across the whole multi-robot system. The reported simulation and experiment results show that the proposed control strategy in general guides the whole multi-robot system to converge to the desired connectivity while forming a coverage across the environment.

3.3 Artificial Neural Network Trained by Back Propagation

This section introduces a distributed motion control scheme, using an artificial neural network, to adapt to the fluctuating wireless communication conditions in realistic environments and establish and maintain desired wireless communication connections for multi-robot systems. A neural network controller is designed for each robot, trained with sample data, and applied to multi-robot deployment for communication coverage. The simulation results show that the proposed scheme can establish and maintain effective communication coverage under different path loss exponents and uncertainties, and the average RSSI converges to the desired range.

The layout for the rest of the section is as follows. To be self-contained, Section 3.3.1 will briefly review the basics of artificial neural networks. Section 3.3.2 will define the proposed neural network controller for communication coverage. Section 3.3.3 will report the simulation results based on the well-adopted probabilistic log-distance path loss model. Section 3.3.4 will summarize the contribution of this section.

3.3.1 Review of Artificial Neural Network

Inspired by biological neural networks, an artificial neural network is a computational structure consisting of a collection of interconnected elements, known as neurons, to define a function [65, 66]. The network function is largely determined by the nature of the connections, which can be adjusted to map an input space to the corresponding desired output space.

$$c = F\,(\,w^T I + b\,) \tag{3.6}$$

Neuron is the elemental component of an artificial neural network (Fig. 3.18). The core of a neuron is a transfer function F which maps the sum of the weighted input $\mathbf{w}^T\mathbf{I}$ and the bias b to the output c, i.e. Eq. (3.6) where **I** denotes the vector of all the inputs I_i to the neuron, and **w** denotes the vector of all the weights w_i of the connections between the inputs and the neuron.

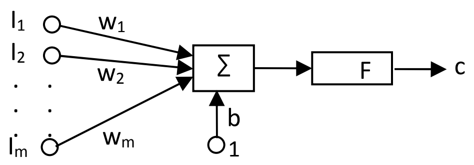


Figure 3.18: Typical neuron.

A neural network is formed by layers of neurons, where the outputs of one layer become the inputs of next layer. A typical neural network architecture is shown in Fig. 3.19. The weights between the kth and (k+1)th layers are defined in a weight matrix $\mathbf{W}^{k+1,\,k}$ whose (j,i) element represents the weight of the connection between the ith output of the kth layer and the jth neuron of the (k+1)th layer. Consequently the output vector of the (k+1)th layer becomes Eq. (3.7).

$$\mathbf{c}^{k+1} = \mathbf{F}^{\mathbf{k+1}}(\,\mathbf{W}^{k+1,k}\,\mathbf{I}^{k} + \mathbf{b}^{k+1}\,) \tag{3.7}$$

In order to let a neural network map an input space to the desired output space, the weights and biases are often adjusted through an iterative training process until the resulting outputs match the targeted outputs. The training process can be either incremental, where the weights and biases are updated after each input-target pair, or batch, where the weights and biases are updated after all the inputs and targets are provided.

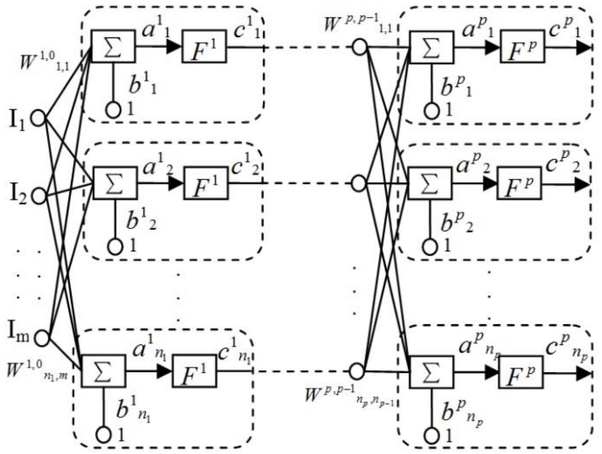


Figure 3.19: Typical neural network architecture.

In this section, we use the widely-adopted back propagation (BP) algorithm [66] to train a neural network to fit the input-output relationship embedded in the sample data. BP is a supervised learning method which infers a function based on samples of input-output data pairs. For each input vector, the algorithm estimates the error between the actual and desired network outputs, and back propagates it from the output layer to hidden neurons to estimate the contribution of each hidden neuron to the output error. It calculates the gradient of each weight, which indicates the direction of error increase, and updates the weight in the opposite direction of the gradient.

3.3.2 Proposed Neural Network Controller

In this section, we train a local neural network for each robot to control its movement to adapt to the quality of the wireless links with its neighbors; the combined effect of all robots' movement will allow the multi-robot system to approach and maintain the desired communication coverage. The following discussions will focus on the definition and structure of the local, onboard neural network controller.

The proposed neural network controller is designed to decide the communication-adaptive motion of a robot according to the real-time feedback of the quality of its wireless links with neighboring other robots. We adopt a two-layer feed forward network architecture (Fig. 3.20), which in principle can be trained to approximate any function with a finite number of discontinuities [65, 66].

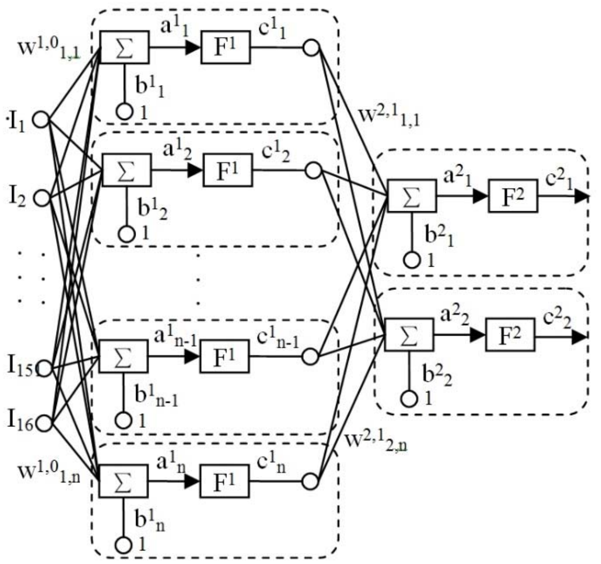


Figure 3.20: Neural network with eight neighbors.

In particular,

1) Input: The input vector **I** consists of the measured wireless link quality (RSSI in this work) and bearing between the robot and each of its neighbors. Based on our observation of simulated deployment processes, we set the input vector as sixteen dimensional, which can accommodate eight neighbors at most. If at any instant the number of neighbors is above eight (which is very rare with Delaunay triangulation), only the eight closest neighbors are considered.
2) Output: The output of the neural network is a motion decision vector $D = [D_x, D_y]$ (for 2D environments), where D_x and D_y denote the desired next-step robot displacements in x and y directions of the environment frame.
3) Hidden layer: Connecting to the inputs of the network, the hidden layer has 90 neurons. The number of neurons is determined based on a sequence of tests of the output error with different numbers of neurons. The tangent sigmoid transfer function is adopted by this layer of neurons, and the output of this layer is obtained as Eq. (3.8).

$$\mathbf{c}^1 = 2\Big/[1 + e^{-2(\mathbf{W}^{1,0}\mathbf{I}+\mathbf{b}^1)}] - 1 \tag{3.8}$$

4) Output layer: The linear transfer function is adopted by the output layer of neurons, and the output of this layer, i.e. the output of the network, is obtained as Eq. (3.9).

$$\mathbf{D} = \mathbf{c}^2 = \mathbf{W}^{2,1}\mathbf{c}^1 + \mathbf{b}^2 \tag{3.9}$$

We use back propagation algorithm to batch train the neural network such that an appropriate motion decision can be made for each robot according to current measurement of the quality of the wireless links with its neighbors. The details of training data will be discussed in Section 3.3.3.1.

3.3.3 Simulations

3.3.3.1 Training Data

To train the neural network controller, we need to come up with a collection of sample data pairs including the input, RSSI, and desired output, next step motion, which fit with the control objective of approaching and maintaining a desired level of wireless link quality among neighboring robots. In general, poor wireless link quality means the risk of disconnection or packet loss, while overly good wireless link quality often corresponds to small space coverage. In our simulation, we use the following feedback control, based on probability-laws model to generate the training data sets for the neural network controller. The feedback controller includes three steps:

1) Calculate the probability of the RSSI belonging to signal strength set_i for each neighbor.
2) Choose the control laws based on the probability from 1) and calculate the motion decision from control laws for each neighbor.
3) Calculate the linear combination of motion decisions of all neighbors.

Moreover, we recognize that the concept of set with probability provides a good classification of RSSI which contains uncertainty. A signal strength set, defined in this section, is a set with a probability for each element in the set. It can contain elements, which are also contained in other set, except the elements with probability 100%. The probability, defined between 0 and 1, calculated by an associated distribution function. This distribution function defined by f(RSSI|set_i)=p, where p denotes the probability of RSSI belonging to probability set i. We define five signal strength sets corresponding to a linguistic division of the RSSI levels: Desired RSSI, Losing RSSI, Weak RSSI, Crowding RSSI and Strong RSSI. Arranging these signal strength sets in the order of ascending RSSI level, we have losing, weak, desired, strong and crowding RSSI.

These signal strength sets are in general defined according to the functionality of onboard wireless transceivers and requirements of applications. We define these sets and associated distribution functions in our simulations (Fig. 3.21) based on the log-distance path loss model mentioned in Section 3.1.2. The simulated RSSI values with different path loss exponent n and standard deviation σ are scaled into the same range [0, 1] for input to the proposed neural network. Here trapezoid-shaped cumulative distribution functions are adopted, because they provide an effective and simple description of how we judge the RSSI level. In particular, we have

1) L (Losing): An RSSI$\in$[0,0.07] is considered as a losing RSSI. The probability of [0, 0.02] is set to 1, while [0.02, 0.07] is the transitional part.
2) W (Weak): An RSSI$\in$[0.02,0.2] is considered as a weak RSSI. The probability of [0.07, 0.08] is set to 1, while [0.02, 0.07] and [0.08, 0.2] are the transitional parts.

3) D (Desired): An RSSI$\in$[0.08, 0.29] is considered as a desired RSSI. The probability of [0.2, 0.24] is set to 1, while [0.08, 0.2] and [0.24, 0.29] are the transitional parts.

4) S (Strong): An RSSI$\in$[0.24, 0.5] is considered as a strong RSSI. The probability of [0.29, 0.37] is set to 1, while [0.24, 0.29] and [0.37, 0.5] are the transitional parts.

5) C (Crowding): An RSSI$\in$[0.37, 1] is considered as a crowding RSSI. The probability of [0.5,1] is set to 1, while [0.37,0.5] is the transitional part.

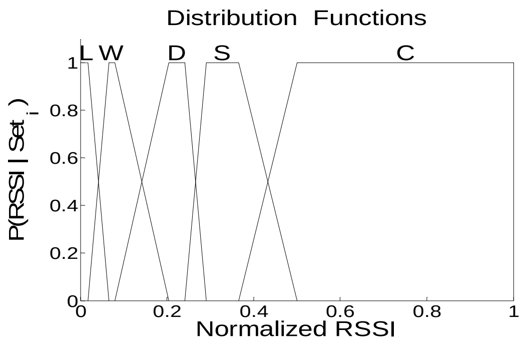


Figure 3.21: Signal strength sets and distribution functions of RSSI.

In order to maintain the desired RSSI with a neighbor R_j, a robot R_i should move away from R_j with large RSSI and move close to R_j with small RSSI. Correspondingly, we define

1) If RSSI is at the losing level, then R_i moves towards R_j with a distance D = L;
2) If RSSI is at the weak level, then R_i moves towards R_j with D = 0.3L;
3) If RSSI is at the desired level, then R_i does not move;
4) If RSSI is at the strong level, then R_i moves towards R_j with D = - 0.3L;
5) If RSSI is at the crowding level, then R_i moves towards R_j with D = - L.

After one control law chosen from above five control laws, based on the probability of RSSI belonging to signal strength set_i, the consequent of the feedback controller is a motion decision vector **D** which requires R_i to move a distance D along the straight line connecting R_i and R_j. Specifically in the planar environment, **D** = $[D * \cos(\theta_{ij}), D * \sin(\theta_{ij})]^T$, where θ_{ij} denotes the orientation angle of the line connecting R_i and R_j in the environment frame. The above five control laws are defined for R_i to react to the RSSI with R_j. When R_i has multiple neighbors, these five control laws and associated probability apply to each of the neighbors, and the comprehensive motion decision for R_i, which adapts to all the neighboring wireless links, is a linear combination of the motion decisions of all the neighbors.

To generate the training data sets for the proposed neural network controller, we randomly position 1 to 8 neighbors around a robot with uniform distribution inside a circle with relatively reliable packet receiving condition. We use Eq. (3.2) to generate RSSI among the robots and normalize them to the range of [0, 1], and follow the above feedback controller to generate the corresponding motion decisions for each robot. The RSSI inputs (along with the bearings) between each robot and its neighbors and the associated motion decision are taken as a sample data pair for the training of the neural network. The trained neural network is then used as the onboard controller for each mobile robot in a multi-robot system to establish and maintain the desired communication coverage.

3.3.3.2 Simulation Results

Fig. 3.22 and 3.23 show that the average RSSI in general converges towards the range of desired RSSI [0.2, 0.24] as the MRS approaches a communication coverage. They also indicate that the average RSSI has some uncertainty, reflected by the fluctuation of the average RSSI as the time goes on. This fluctuation results from the reactive response of the neural network controller to the randomness in signal propagation. Fig. 3.22 shows that the level of fluctuation in the average RSSI increases as the uncertainty in signal propagation increases, and Fig. 3.23 shows that the fluctuation decreases as the path loss exponent increases.

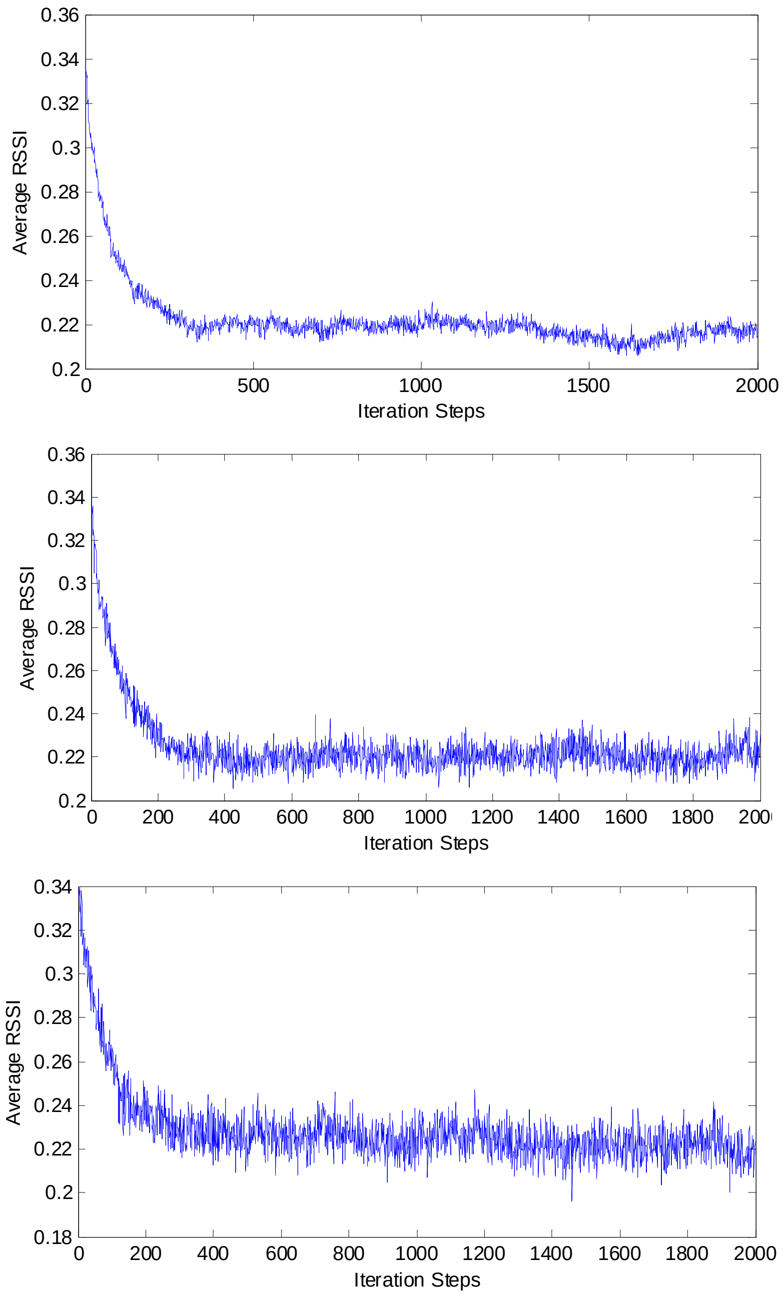


Figure 3.22: Average inter-robot RSSI when n=3.0 and σ=3.0, σ=7.0 and σ=9.6.

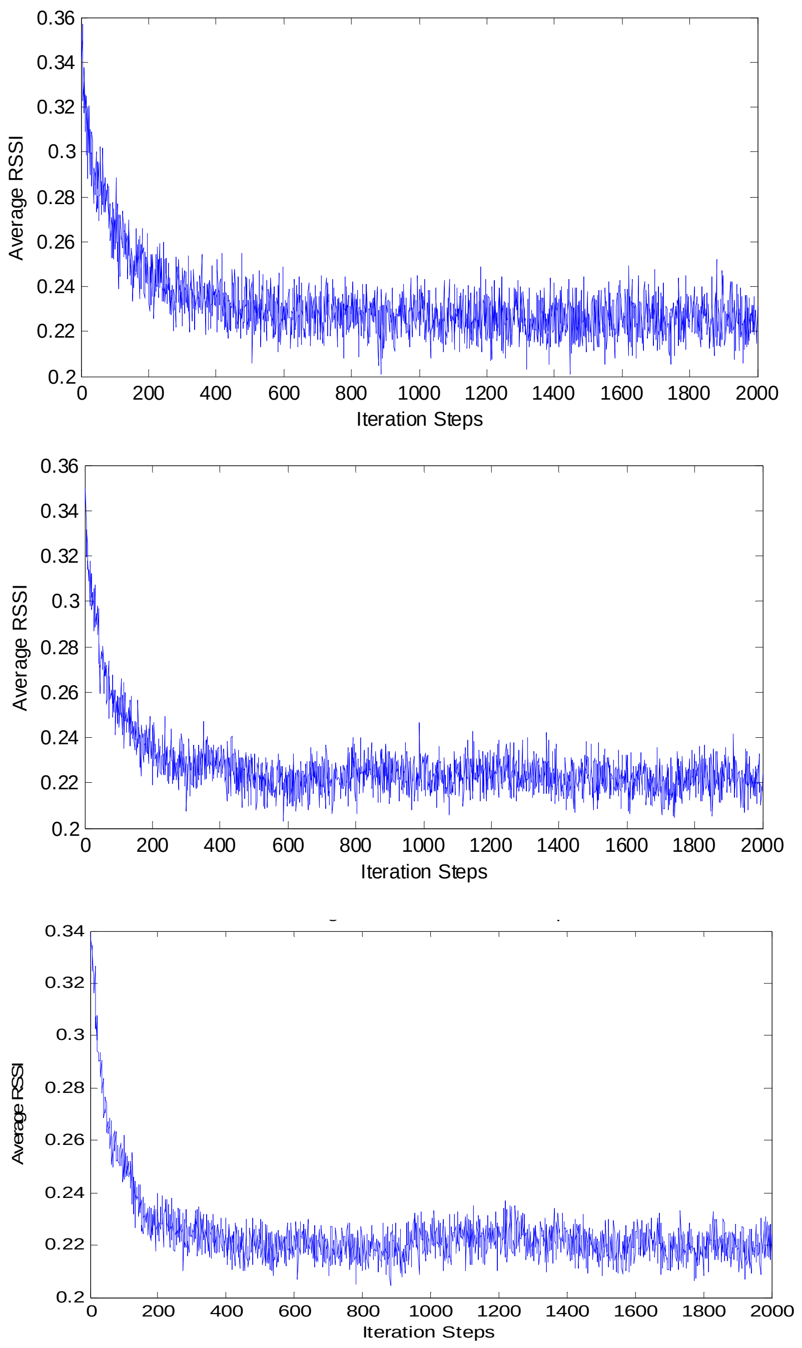


Figure 3.23: Average inter-robot RSSI when σ=7.0 and n=1.6, n=2.2 and n=3.0.

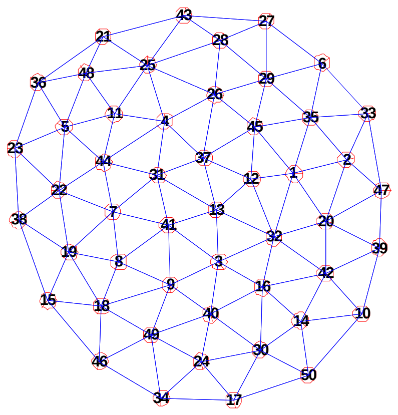


Figure 3.24: Deployment configuration when n=3.0 and σ=7.0.

We also look at the evolution of the system configuration during the deployment process under different values of n and σ. Fig. 3.24 presents a typical system configuration of the 50-robot system after 2000 deployment steps, where the underlying neighborhood relationships are visualized using Delaunay graph. It shows that the proposed neural network control scheme is capable of forming an effective network coverage under different levels of uncertainty in signal propagation.

3.3.4 Conclusion

This section proposes a distributed control scheme to establish and maintain wireless communication connections while forming a system coverage to an unknown environment. The proposed approach defines a local neural network controller to onboard make motion decisions for each mobile robot to adapt to the real-time wireless link quality with its neighbors. The combined effect of all these neural network controllers results in the desired communication coverage. The simulation results show that, with the proposed neural network control scheme, the MRS can form and maintain an effective communication coverage with desired average RSSI in different environments. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.4 Fuzzy Neural Network Trained by Back Propagation

This section presents a decentralized control scheme, based on the fuzzy neural network, to achieve the control objective, which is to establish and maintain desired wireless communication connections for multi-robot systems. The whole control process consists of:

1) Defining the fuzzy neural network,

2) Training the neural network with error function,

3) Applying trained neural network to multi-robot deployment.

The simulation results show that the proposed scheme can establish and maintain effective communication coverage under documented path loss exponents and uncertainties, and the average RSSI converges to the desired range.

The layout for the rest of this section is as follows. To be self-contained, Section 3.4.1 will briefly review the structure of the fuzzy neural network adopted by our work. Section 3.4.2 will define the proposed fuzzy neural network for communication maintenance. Section 3.4.3 will report the simulation results based on the well-adopted probabilistic log-distance path loss model. Section 3.4.4 will summarize the contribution of this section.

3.4.1 Review of fuzzy-Neural Network

A fuzzy neural network is a hybrid system which combines the fuzzy logic and artificial neural network [110]. The structure is similar to a neural network but follows a fuzzy inference process, which maps inputs through input membership functions and rules to outputs. The parameters of the fuzzy inference are tuned through the learning process of the neural network.

We adopt the Takagi-Sugeno fuzzy neural network (Fig. 3.25), which consists of an antecedent (lower) network, corresponding to the fuzzification of the inputs and the calculation of the fire strengths, and a consequent (upper) network, corresponding to the implication of the consequents. The final output is determined by the linear combination of the consequents weighted by the fire strengths.

The antecedent network consists of three layers:

Layer 1 determines the membership degree of each input x_i based on each membership function f_j, i.e. $\mu_i^j = f_j(x_i)$.

Layer 2 determines the fire strength of each rule w_k by applying the fuzzy operation to the antecedent of each rule.

Layer 3 normalizes the fire strength as Eq. (3.10).

$$\overline{w}_k = w_k \Big/ \sum_k w_k \ , \tag{3.10}$$

The consequent network consists of two layers:

Layer 1 calculates the consequent of each rule as a linear combination of the inputs, i.e. Eq. (3.11) where a_{ski} denotes the weight between x_i and y_{sk}, and c_{sk} denotes the weight between 1 and y_{sk}.

$$y_{sk} = \sum_i a_{ski} x_i + c_{sk} \tag{3.11}$$

Layer 2 calculates the final output as a linear combination of the consequents of the rules, weighted by the fire strengths Eq. (3.12).

$$z_s = \sum_k y_{sk} \overline{w}_k \ , \tag{3.12}$$

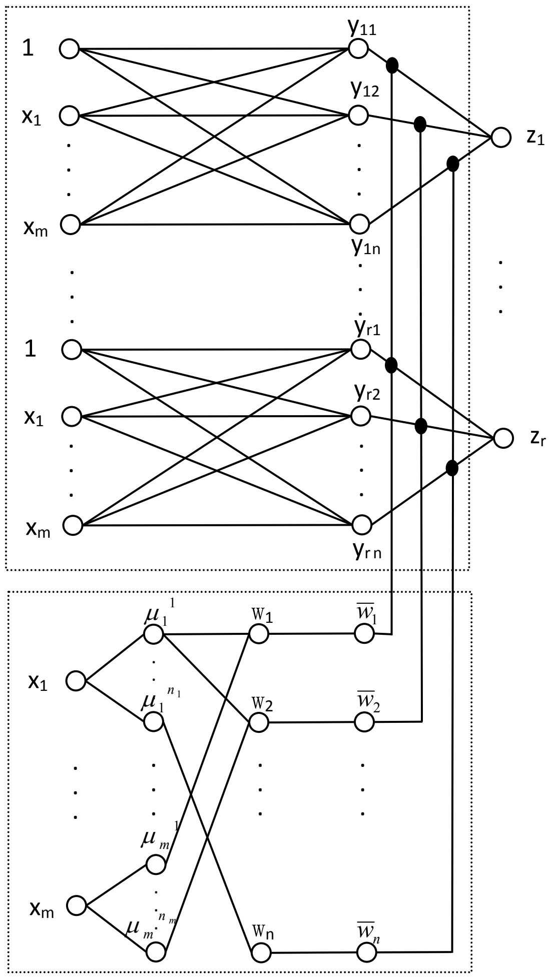


Figure 3.25: Typical fuzzy neural network architecture.

3.4.2 Proposed Fuzzy Neural Network

The proposed fuzzy-neural network controller, based on proposed fuzzy logic in Section 3.2.2, is designed to make decision for the communication-adaptive motion of a robot according to the real-time feedback of the wireless link quality with its neighbors.

Following the fuzzy neural network structure described in Section 3.4.1, the proposed fuzzy neural network consists of an antecedent (lower) network and a consequent (upper) network (Fig. 3.25). The inputs of the proposed fuzzy neural network are divided into two separate portions: the antecedent network inputs [x_1^a,…, x_n^a], which are the RSSI values between the robot and its neighbors, and the consequent network inputs [x_1^c,…, $x_{(2n)}^c$], which are the cosine and sine of the bearing angles between the robot and its neighbors, where n denote the number of neighbor.

The antecedent network maps the RSSI to the firing strength of each rule:

Layer 0 (Input): The input vector is eight dimensional, which can accommodate the RSSI from at most eight neighbors. Each input $x_{i,j}{}^{a}$ is a measured RSSI between the robot R_i and a neighbor R_j. If at any instant the number of neighbors is above eight (which is very rare with Delaunay triangulation), only the eight closest neighbors are considered.

Layer 1 (Fuzzification): The inputs are fuzzified using the transfer functions ($f_{i,j,k}$) of this layer, of which example is defined in Section 3.4.3.2, with $f_{i,j,k}$ denoting the value of kth membership function associated with robot R_i and its neighbor R_j, to determine the membership degree of each input in each fuzzy set, i.e. $\mu_{i,j}{}^{k} = f_{i,j,k}(x_j{}^{a}) = f_{i,j,k}(RSSI_{i,j})$.

Layer 2 (Normalization): Layer 2 defines the fire strengths of the corresponding rules by normalizing the outputs of layer 1 using Eq. (3.10). Since the antecedent of each rule has only one part, there is no need for a separate layer for fuzzy operation, which is needed in the general case (Section 3.4.1). So, the inputs of this layer can be determined by $w_{i,j,k} = \mu_{i,j}{}^{k}$, where $w_{i,j,k}$ is the fire strength associated with R_j through the kth rule.

The consequent network maps the bearing angles between robot R_i and its neighbors to a motion decision for R_i:

Layer 0 (Input): The input vector is sixteen dimensional, which can accommodate at most eight neighbors. Each pair of inputs, $x_{i,(2j-1)}{}^{c}$ and $x_{i,(2j)}{}^{c}$, are the cosine and sine values of the bearing angle $\theta_{i,j}$ between robot R_i and a neighbor R_j.

Layer 1 (Implication): This layer defines the motion decision towards each neighbor under each level of wireless link quality. The transfer functions are defined as Eq. (3.13), where

$L_{i,j,k}$ denote the step length associated with its neighbor R_j by kth fuzzy rules. And $Dy_{i,j,k}$ is defined in same way.

$$y_{1,(2j-1,k)} = Dx_{i,j,k} = \cos\theta_{i,j} \cdot L_{i,j,k} \tag{3.13}$$

Layer 2 (aggregation): The final output of the fuzzy neural network is a motion decision vector **D** = [Dx, Dy], refer to [z_1, z_2] in Fig. 3.25, obtained as a linear combination of the outputs of layer 1 of the consequent network (the consequents of the rules) weighted by the outputs of the antecedent network (the fire strengths of the rules) using Eq. (3.12).

3.4.3 Simulations

3.4.3.1 Training Data

The proposed fuzzy neural network maps the RSSI and bearings between a robot R_i and its neighbors to the motion decision of R_i. The involved parameters of fuzzy inference, including the step length L (as the parameter of the transfer function of layer 1 in the consequent network) and the parameters of the membership function (as the transfer function of layer 1 in the antecedent network), are tuned using the back propagation (BP) algorithm [65, 66, 110]. For each input vector, the BP algorithm estimates the error between the actual and desired network outputs, and back propagates it from the output layer to hidden neurons to estimate the contribution of each hidden neuron to the output error. It calculates the gradient of each parameter, which indicates the direction of error increase, and updates the parameter in the opposite direction of the gradient.

In particular, matching the goal of maintaining the desired link quality, we set the objective of the network training as minimizing the error between desired and actual link quality. Correspondingly we define the error function as Eq. (3.14), where n denotes the number of the neighbors, $RSSI_d$ denotes the desired RSSI value, $RSSI_i$ denotes the RSSI value with the ith neighbor after the robot R_i moves a step **D**. And $RSSI_i$ can be calculated by Eq. (3.2), in which distance d is determined by the motion decision vector **D** (Dx, Dy).

$$E = \frac{1}{2} \sum_{i=1}^{n} (RSSI_i - RSSI_d)^2 \tag{3.14}$$

To generate the training data sets for the proposed fuzzy neural network controller, we randomly position a team of robots, use Eq. (3.2) to generate RSSI among the robots and normalize them to the range of [0, 1]. After the robot moves a step **D**, the error gradients with respect to the parameters of fuzzy inference, are calculated based on the error function defined in Eq. (3.14). Here, we conduct the batch training in which the involved parameters are updated after all the input-output samples are presented.

The trained fuzzy neural network is then used to decide the motion for each robot according to the current quality of the wireless links with its neighbors.

3.4.3.2 Simulation Results

The RSSI value is normalized into the range of [0, 1], and the desired RSSI is set to R_d= [0.2, 0.24] according to the log-distance path loss model. The associated membership functions are defined in Fig. 3.26, where L, W, D, S and C correspond to the fuzzy sets of Losing, Weak, Desired, Strong and Crowding RSSI respectively, which are defined in Section 3.2.2.1.

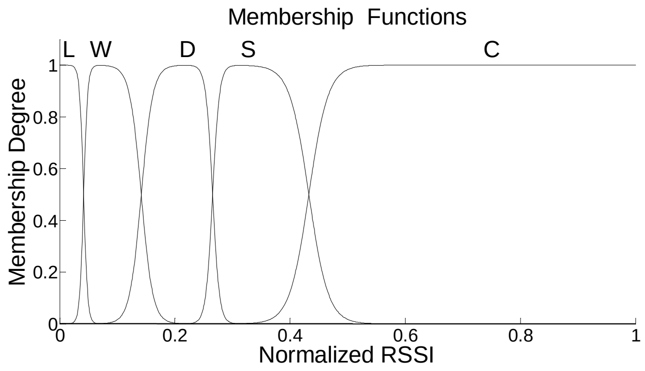


Figure 3.26: Membership functions.

The corresponding membership functions for these fuzzy sets are defined in the following form of Eq. (3.15), where a1, a2, c1 and c2 are constants of the function.

$$f(RSSI) = \frac{1}{1+e^{-a1(RSSI-c1)}} - \frac{1}{1+e^{-a2(RSSI-c2)}} \tag{3.15}$$

A typical deployment configuration resulting from the proposed fuzzy neural network control process is shown in Fig. 3.27, including 50 robots after 2000 deployment steps, where the underlying neighborhoods are visualized using Delaunay graph. It shows that the proposed fuzzy neural network control scheme is capable of forming an effective network coverage under different levels of uncertainty in signal propagation or different path loss exponents.

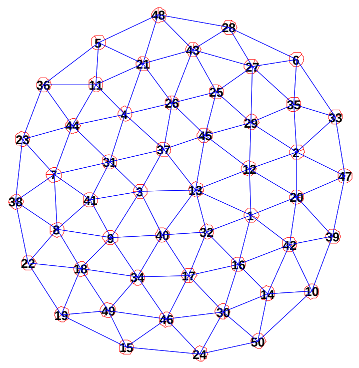


Figure 3.27: Typical deployment configuration.

The evolution of the average RSSI is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig. 3.28 and 3.29 show that the average RSSI in general converges towards the range of desired RSSI [0.2, 0.24] as the MRS approaches a communication coverage. The figures also indicate the fluctuation of the average RSSI. It results from the reactive response to the randomness in signal propagation. Fig. 3.28 shows that the level of fluctuation increases as the uncertainty in signal propagation increases; while Fig. 3.29 shows that the fluctuation of the average RSSI decreases as the path loss exponent increases.

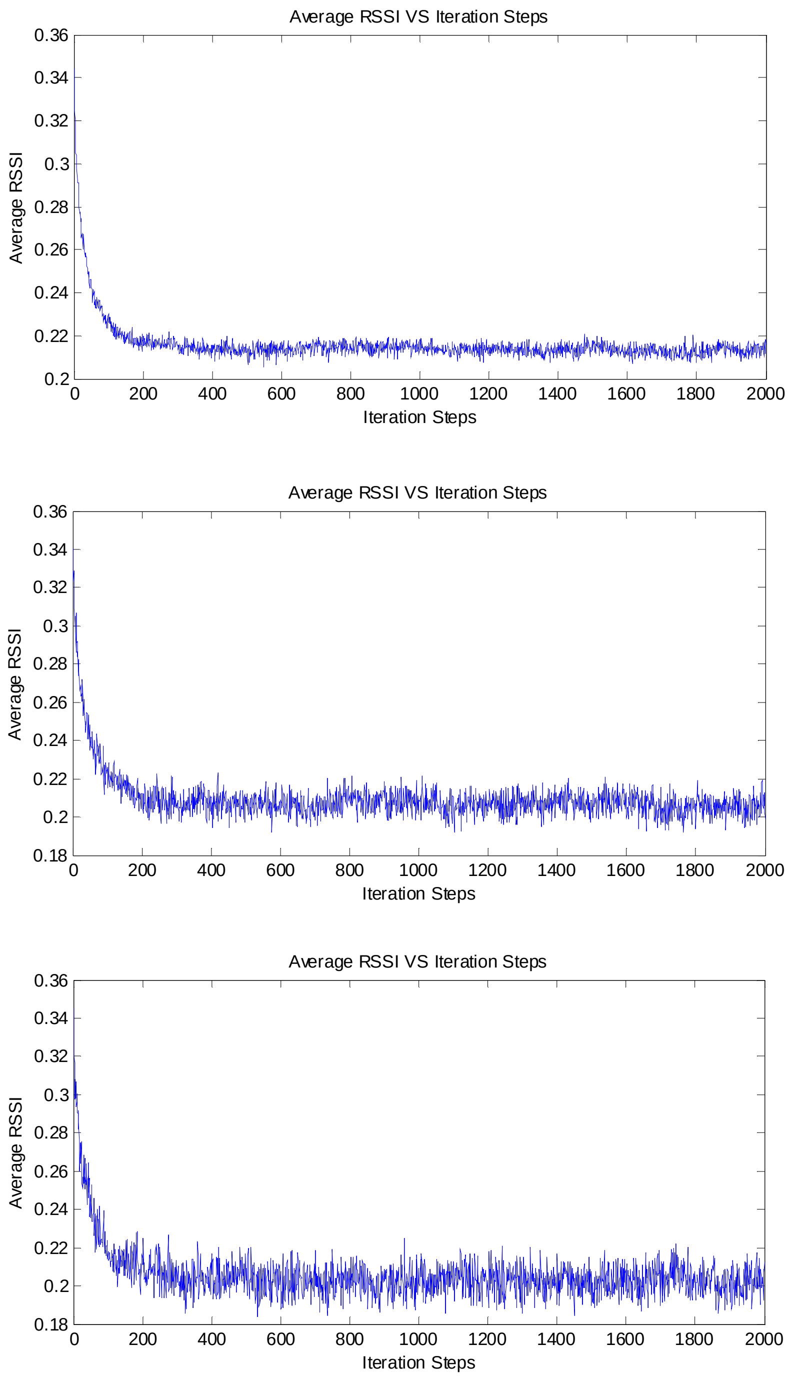


Figure 3.28: Average inter-robot RSSI when n=3.0 and σ=3.0, σ=7.0 and σ=9.6.

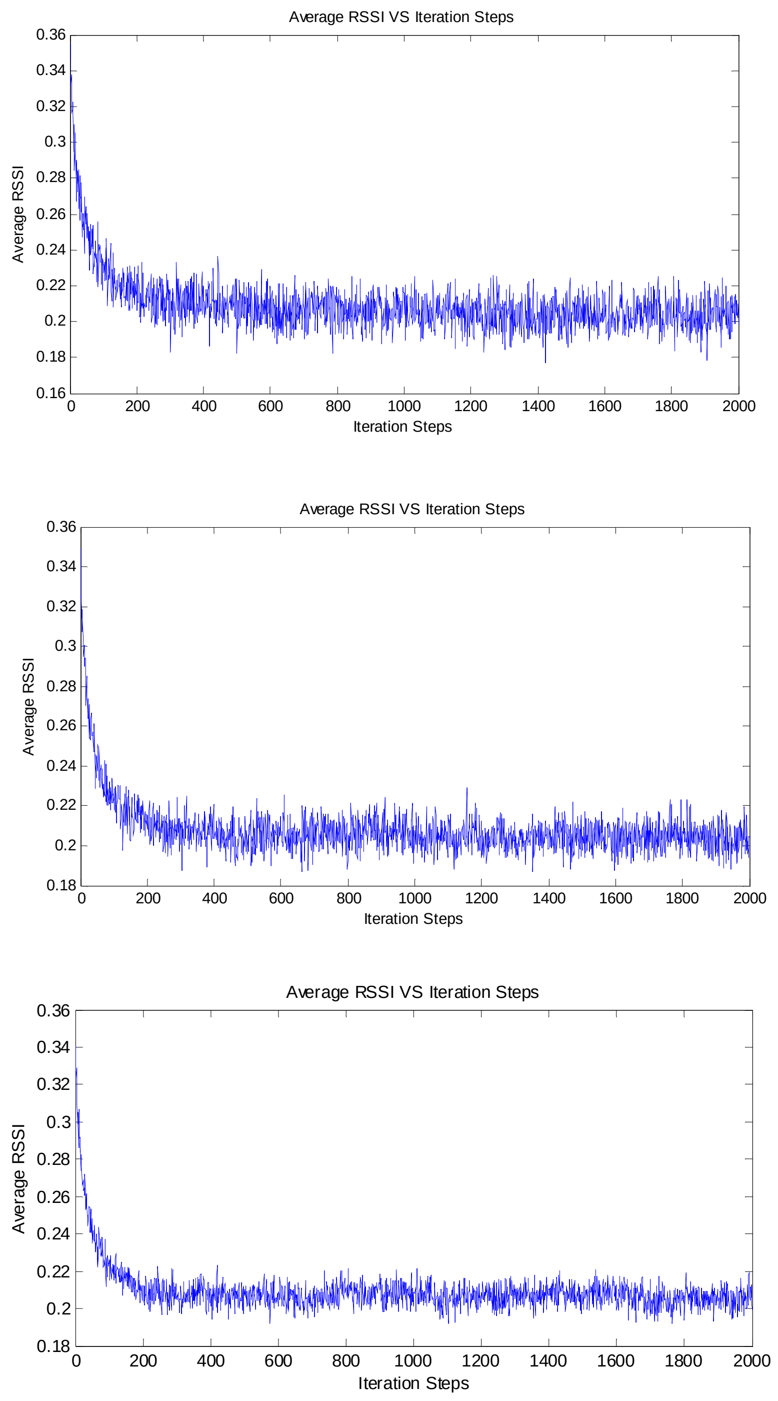


Figure 3.29: Average inter-robot RSSI when σ=7.0 and n=1.6, n=2.2 and n=3.0.

In summary, Fig. 3.27-3.29 show that, with the proposed fuzzy neural network scheme, the MRS can form and maintain an effective communication coverage with desired average RSSI in different environments. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of the average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.4.4 Conclusion

This section presents a distributed control scheme to establish and maintain wireless communication connections while forming a system coverage to an unknown environment. The proposed approach defines a fuzzy neural network controller to onboard make motion decisions for each mobile robot to adapt to the real-time wireless link qualities with its neighbors. The simulation results show that the proposed control scheme effectively guides the whole multi-robot system to converge to the desired communication coverage.

3.5 Radial Basis Network Trained by Reinforcement Learning

This section presents a decentralized control strategy, using the technique of reinforcement learning artificial neural network, to learn and approach a desired wireless communication coverage in a realistic environment for a team of collaborative mobile robots. A reinforcement learning neural network, based on the radial-basis function (RBF), is designed for each robot to learn the control law of maintaining the wireless link quality in a target environment and apply to multi-robot deployment process to form communication coverage. The learning process is performed for a robot through consecutive interactions between controller and environment to establish relationship between wireless link quality and robot motion decision. In several environments simulated with the probabilistic log-distance path loss model, the simulation results show that the proposed reinforcement learning neural network based control approach leads to a desired and reliable multi-robot wireless communication coverage.

The layout for the rest of the section is as follows. To be self-contained, section 3.5.1 will briefly review the radial basis network and reinforcement learning method adopted by our work. Section 3.5.2 will discuss the specific definition of the proposed reinforcement learning radial basis network based on online link quality measurement. Section 3.5.3 will report the simulation results based on a well-adopted probabilistic communication signal propagation model. Section 3.5.4 will summarize the contribution of this section.

3.5.1 Review of Radial Basis Network and Reinforcement Learning

An artificial neural network is a computational structure consisting of an interconnected group of processing elements to model the function relationship between the inputs and outputs. A radial basis network (RBN) is a type of artificial neural network, which uses radial basis functions as activation functions and determines the outputs of the network based on the linear combination of those functions [66]. Reinforcement learning (RL) is a goal-directed learning approach. It provides a learning paradigm to train a neural network to adapt to the environment through trial-and-error search and reward. The following two subsections briefly summarize the key points of RBN and RL.

3.5.1.1Radial Basis Network

A neural network is formed by layers of neurons, where the outputs of one layer become the inputs of next layer. In general, a radial basis network (RBN) consists of two layers:

1) A radial basis hidden layer.
2) A linear output layer.

The elemental component of each layer is an artificial neuron which maps the input vector into an output by using an activation function.

A neuron of a hidden layer is shown in Fig. 3.30. Its mathematical form is $y_j = F^1(\|\boldsymbol{w}^{1,0}{}_j - \boldsymbol{x}\|\ b^1{}_j)$, where y_j denotes the output of neuron j of hidden layer, F^1 denotes the distance activation function $F^1 = e^{-n^2}$ with n as the independent variable, $\boldsymbol{w}^{1,0}{}_j$ denotes the vector of the weights of all the connections between the inputs and neuron j of the hidden layer, $\boldsymbol{x}$ denotes the vector of all the inputs, $\|\boldsymbol{w} - \boldsymbol{x}\|$ denotes the vector distance between the weight vector $\boldsymbol{w}$ and input vector $\boldsymbol{x}$, and $b^1{}_j$ denotes the bias of neuron j of the hidden layer. In practice, a radial basis neuron acts as a detector of the difference between the input vector x and the weight vector w, where b adjusts the sensitivity of the activation function.

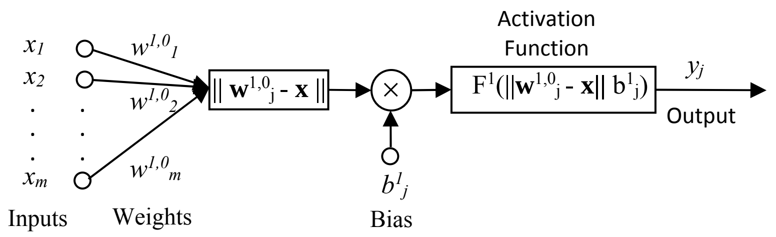


Figure 3.30: Typical model of a neuron in the hidden layer.

A neuron of an output layer (Fig. 3.31) has the similar structure as the neuron of the hidden layer. The mathematical form is $\boldsymbol{z}=\boldsymbol{F}^2(\boldsymbol{W}^{2,1}\boldsymbol{y}+\boldsymbol{b}^2)$, where $\boldsymbol{z}$ denotes the vector of all the outputs, $\boldsymbol{F}^2$ denotes a linear activation function, $\boldsymbol{W}^{2,1}$ denotes the weight matrix between the hidden and output layer, $\boldsymbol{y}$ denotes the vector of all the outputs of the hidden layer, and $\boldsymbol{b}^2$ denotes the bias vector of all the outputs.

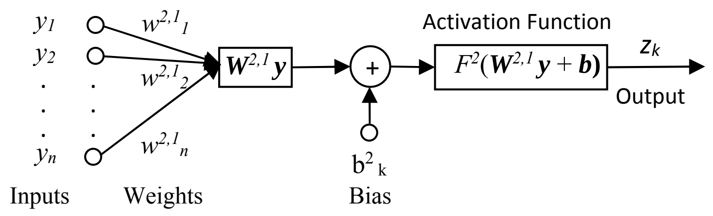


Figure 3.31: Typical model for a neuron in the output layer.

The free parameters of a radial basis network, such as the weight and bias, are frequently adjusted to map an input space to a desired output space or adapt to the environment in which the network is embedded through a learning process. There are three major learning paradigms:

1) Supervised learning – inferring a mapping between inputs and outputs from labeled training data, where the cost function is related to the mismatch between the mapping and the data;
2) Reinforcement learning – discovering a policy for selecting actions to maximize cumulative rewards, generated by interactions between the agent and the environment;
3) Unsupervised (self-organized) learning -- finding the hidden relationship in unlabeled data, where errors or reward signals cannot be determined to evaluate or improve the performance of algorithm.

In the proposed control scheme, reinforcement learning is adopted due to the need of learning communication adaption policy from interaction with the environment.

3.5.1.2Reinforcement Learning

Reinforcement learning (RL) learns how to map situations to actions through continuous interaction with the environment to minimize a scalar index of performance.

A general framework of RL, as shown in Fig. 3.32, consists of:

1) Agent/controller which determines the actions/control signals.
2) Environment/plant which decides the states and rewards.

During the interaction, after the agent selects its actions, the environment makes a response consequently and results in new situations/states and rewards. Through continuous iterations over time, the learning process tries to maximize the rewards for the actions.

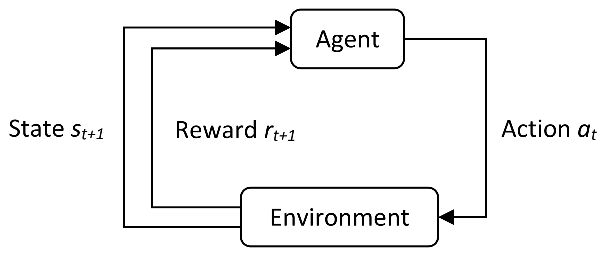


Figure 3.32: General framework of reinforcement learning.

Specifically, the interaction between the gent and environment happens at a sequence of discrete time, t=0, 1, 2…. At each time t, the agent receives some feedback of the environment's state, $s_t \in S$, where S is the set of possible states, and selects an action, $a_t \in A(s_t)$, where $A(s_t)$ is the set of actions available for state s_t. Then at next time t+1, as a result of its previous action a_t, the agent receives a numerical reward r_{t+1}, and the environment evolves into a new state s_{t+1}.

At each time, the agent maps the state to an action with a probability. This mapping relationship is known as the agent's policy, denotes as $\pi_t(s, a)$, which gives the conditional probability of $a_t = a$ when $s_t = s$. The policy evolves as the learning process goes, with the goal to maximize the total amount of reward received over a long run.

The RL framework in Fig. 3.32 is an abstraction of the problem of goal-directed learning from interaction. It proposes that, whatever the details of the sensory, memory and control apparatus are, and whatever objective one tries to achieve, any goal-directed learning process can be abstracted to one with three signals passing back and forth between an agent and its environment: one signal to represent the choice made by the agent (the action), one signal to represent the basis on which the choice is made (the state), and one signal to define the agent's goal (the reward).

3.5.2 Proposed Radial Basis Network Trained by Reinforcement Learning

This section addresses the communication maintenance problem for a team of mobile robots in a realistic environment. Since the wireless communication condition is completely unknown in such an environment, a radial basis network is designed for a robot to adapt to the environment and maintain the desired wireless link quality with one neighbor. A local controller for each robot is defined by the aggregation of all the RBNs, corresponding to all its neighbors, to adapt to the quality of the wireless links with all its neighbors. The combined effect of all the robots' movement will allow the multi-robot system to approach and maintain the desired communication connectivity. Besides, we take a distributed manner to control the deployment process in order to attain high adaptability and scalability.

The control objective is to establish and maintain a desired wireless communication coverage for a team of collaborative mobile robots in a realistic environment. In this section, a distributed adaptive control scheme (Fig. 3.33) is proposed to achieve our intention. An RBN, designed for a robot to maintain the wireless link quality for one neighbor, is trained using RL through the interaction with an unknown environment and used to decide the motion for a robot.

The proposed control scheme consists of two parts:

1) Learning: The RBN controller of a robot interacts with an unknown environment and learns the communication-adaptive motion policy according to the real-time feedback of the quality of its wireless links with its neighbors. In order to keep the complexity of network reasonable, the RBN is designed for a robot to maintain the desired wireless communication with one neighbor.

2) Decision making: when a robot has multiple neighbors, a RBN controller is assigned to each neighbor. The motion of each robot is decided by an onboard local controller which aggregates the RBNs for all the neighbors.

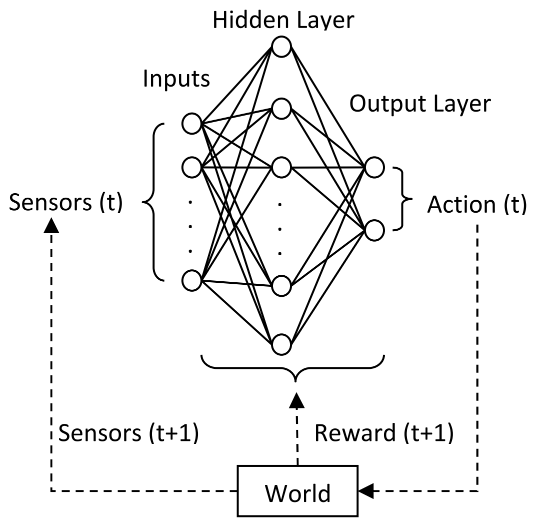


Figure 3.33: Proposed framework of RBN and RL.

In particular, the RBN (Fig. 3.33) is designed to receive the feedbacks from sensors, e.g. wireless unit and encoders, and decide the action, i.e. the robot's next-step displacement. It consists of the following layers:

1) Input: The input vector I consists of the wireless link quality, bearing between the robot and its neighbor and other information measured through sensors, which may affect the wireless link quality.
2) Hidden layer: Following the definition in section II.A, the output of the hidden layer is $\mathbf{H} = \mathbf{F}\ (\ \| \mathbf{W}^{1,0}$ - ReMat($\mathbf{I}$, 1, n) $\| \odot \mathbf{b}^{1}$), where $\mathbf{F}$ denotes the activation function, $\mathbf{W}^{1,0}$ denotes the weights matrix of all the connections between the inputs and hidden layer, $\mathbf{I}$ denotes the vector of all inputs, ReMat($\mathbf{I}$, 1, n) creates a matrix consisting of an 1-by-n copies of $\mathbf{I}$, n denotes the length of the input vector, operator $\odot$ denotes

element-by-element multiplication, and $\mathbf{b}^1$ denotes the bias vector of all the neurons in the hidden layer.

3) Output layer: The output of the radial basis network is a motion vector $\mathbf{D} = [D_x, D_y]$ (for 2D environments), where D_x and D_y denote the desired next-step robot displacements in x and y directions of the environment frame. The action for the next step can be calculated by $\mathbf{D} = \mathbf{W}^{2,1}\,\mathbf{H} + \mathbf{b}^2$, where $\mathbf{W}^{2,1}$ denotes the weights matrix of all the connections between the hidden layer and output layer, and $\mathbf{b}^2$ denotes the bias vector of all the neurons in the output layer.

The proposed learning process (Fig. 3.33), based on the general framework of RL (Fig. 3.32), is designed to adaptively learn the control policy through the interaction between the controller and environment. Within the general RL framework, the corresponding components are defined in following:

1) The "agent" consists of the hidden and output layer of the RBN.
2) The "environment" includes the real world and the sensors. In fact, anything that cannot be changed arbitrarily by the agent is considered to be exclusive, thus becoming part of its environment.
3) The "states" of the environment include the measured wireless link quality, and the bearing between the robot and its neighbor.

During the learning process, the interaction between the RBN and environment occurs at a sequence of discrete time.

1) At the beginning of learning process, the RBN randomly initializes all the network parameters, i.e. the weight and the bias, and the initial position of robot is randomly

picked, following the uniform distribution, within the communication range of a neighbor which does not move during the learning process.

2) At time t, the position of the robot is updated by previous action(t-1). The RBN perturbs all the parameters by Eq. (3.16) and Eq. (3.17), i.e. make random small changes to all the weights and biases, where ***W*** denotes the weights matrix, α denotes the learning rate, randn(size(***W***)) generates a matrix which contains pseudorandom values drawn from the standard normal distribution and has the same size as ***W***, ***b*** denotes the bias vector, and ERR_c denotes the current difference between the desired RSSI and the RSSI measured at the output position after each action. The perturbations of the parameters can be adaptively adjusted by introducing the ERR_c to the perturbations, where the perturbations will decrease/increase as the ERR_c decreases/increases. The RBN then decides the next motion of the robot (action (*t*)) according to the wireless link quality and bearing (sensors (*t*)).

$$\mathbf{W} = \mathbf{W} + \alpha \times ERR_c \times randn\ (size\ (\mathbf{W})) \tag{3.16}$$

$$\mathbf{b} = \mathbf{b} + \alpha \times ERR_c \times randn\ (size\ (\mathbf{b})) \tag{3.17}$$

3) At time step t+1, as the consequence of action (t), the RBN receives a numerical reward (reward (t+1)) determined by the reward function F^r, and chooses whether or not to update all the parameters from the perturbations. Here, F^r is defined as Eq. (3.18), where abs() is the absolute value operator, $RSSI_c$ denotes the RSSI measured after current action, and $RSSI_d$ denotes the desired RSSI. If F^r is greater than 0, RBN will update all the weights and biases with the perturbations. Otherwise RBN will not update any weights and biases. The strategy, indicated by the definition of reward function, is that the controller will find the weights and biases that can produce the

actions leading towards the desired RSSI. At the same time, the RBN receives sensors ($t + 1$) as the new input, and continues with the learning process.

$$F^r = ERR_c - \mathrm{abs}(RSSI_c - RSSI_d) \qquad (3.18)$$

Through the learning process, the RBN is trained to establish a motion decision policy for the robot to adapt to the wireless link quality with each neighbor, and a motion decision can be made for each robot according to the wireless link quality and the bearing between the robot and its neighbor.

3.5.3 Simulations

The proposed reinforcement learning radial basis network controller for establishing and maintaining communication coverage in a multi-robot deployment scenario has been evaluated through simulations. The results of the learning process and simulations are reported as below.

3.5.3.1 Result of Learning Process

The control objective is to approach and maintain a desired wireless communication coverage for MRS in an unknown environment, which is achieved in this work by approaching and maintaining a desired level of RSSI. Since RSSI is the indicator of the wireless link quality, the proposed scheme takes RSSI as the input to the RBN. Thus, each RBN has one input (RSSI) and one output (motion decision $\mathbf{D} = [\, D_x = d\cos(\theta), D_y = d\sin(\theta) \,]$, where θ denotes the bearing between the robot and its neighbor. When a robot has multiple neighbors, one RBN is needed to deal with each neighbor. The final motion decision is the sum of the outputs of all the RBNs.

We train the proposed RBN controller for a robot to adapt to each of the environments defined in Section 3.1 through a sequence of interaction with the environment, using the reinforcement learning process described in Section 3.5.2. At each moment, a robot moves by following the motion decision made by the current RBN based on the current RSSI condition with a fix neighbor, and then adjusts the parameters of its onboard RBN controller based on the reward feedback. Fig. 3.34 shows a few typical learning curves in a representative environment, Fig. 3.35 shows the mean of these learning curves, and Fig. 3.36 shows the standard deviation of these learning curves. While the difference and fluctuation among the learning curves result from the randomness of the wireless communication condition, the simulation results indicate that in average the RBN controller converges through the learning process (in the case of Fig. 3.34-3.36, the learning process converges after 128 steps in average). The resulting controller will move the robot to approach the desired RSSI with the neighbor.

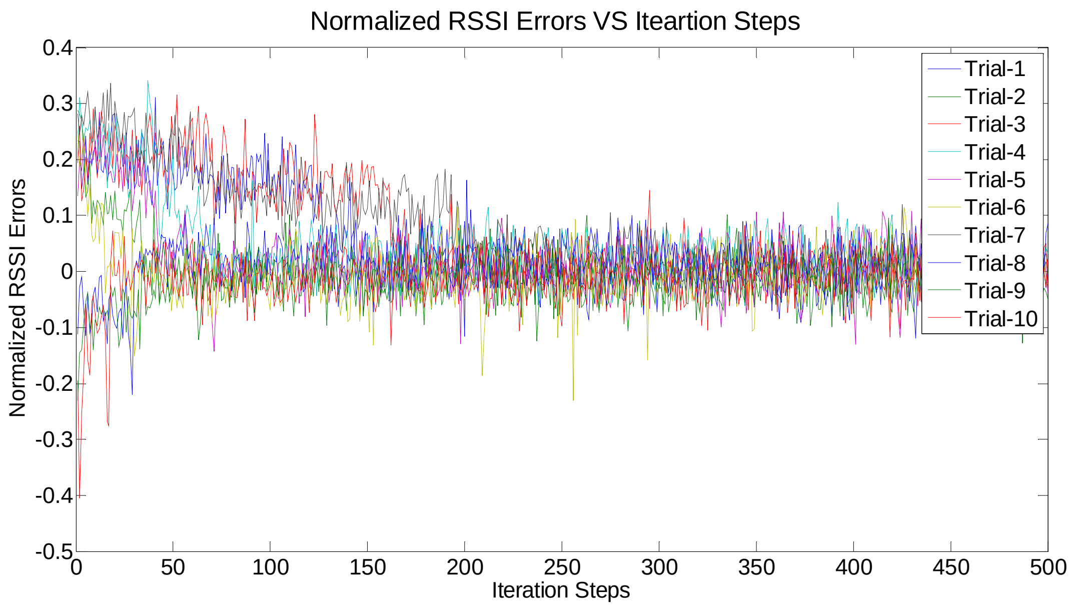


Figure 3.34: Evolution of the difference between the desired and measured RSSI when n=3.0 and σ=7.0.

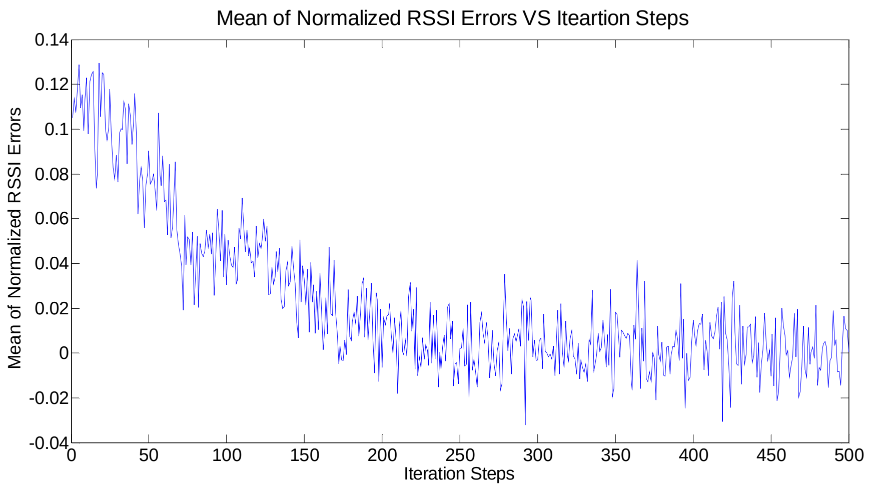


Figure 3.35: Evolution of the mean of the difference between the desired and measured RSSI when n=3.0 and σ=7.0.

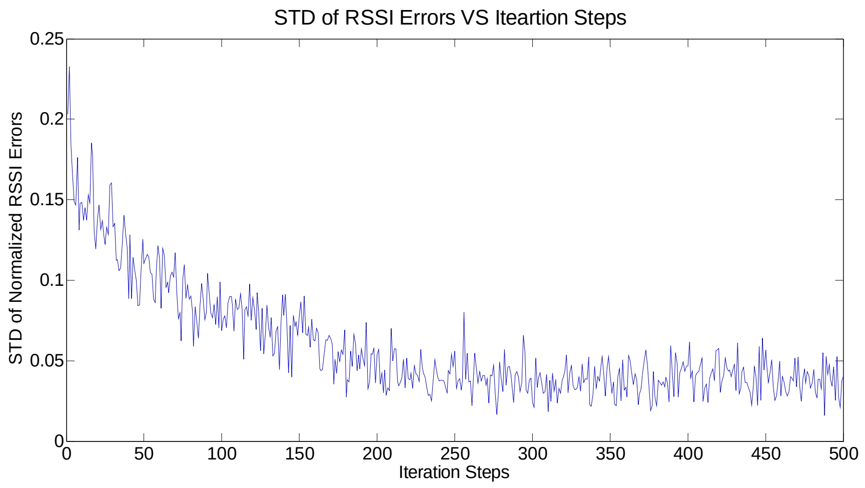


Figure 3.36: Evolution of the standard deviation of the difference between the desired and measured RSSI when n=3.0 and σ=7.0

3.5.3.2 Results of the Deployment Process

A typical deployment configuration resulting from the proposed control scheme is shown in Fig. 3.37, including 50 robots after 2000 deployment steps, where the underlying neighborhoods are visualized using Delaunay graph. It shows that the proposed control scheme is capable of forming an effective network coverage under different levels of uncertainty in signal propagation or different path loss exponents.

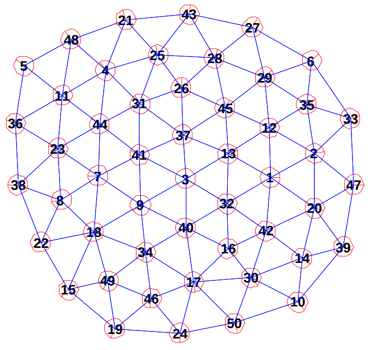


Figure 3.37: Typical deployment configuration.

The evolution of the average RSSI is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig. 3.38-3.39 show the average RSSI in general converges towards the desired RSSI value 0.22 as the MRS approaches a communication coverage. The fluctuation of the average RSSI results from the reactive response to the randomness in signal propagation. Fig. 3.38 presents the level of fluctuation increases as the uncertainty in signal propagation increases; meanwhile Fig. 3.39 provides that the fluctuation of the average RSSI decreases as the path loss exponent increases.

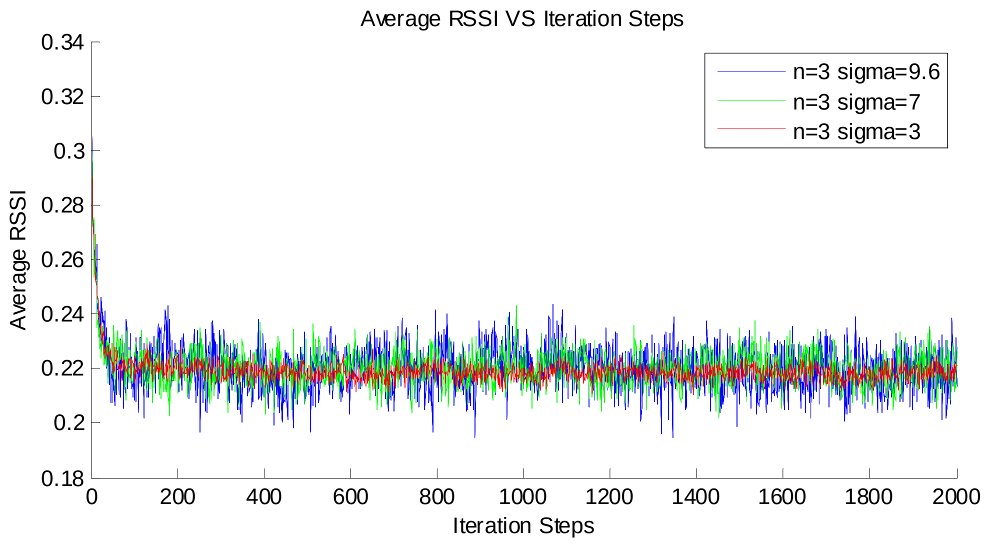


Figure 3.38: Average inter-robot RSSI when n=3.0 and σ=3.0, σ=7.0 and σ=9.6.

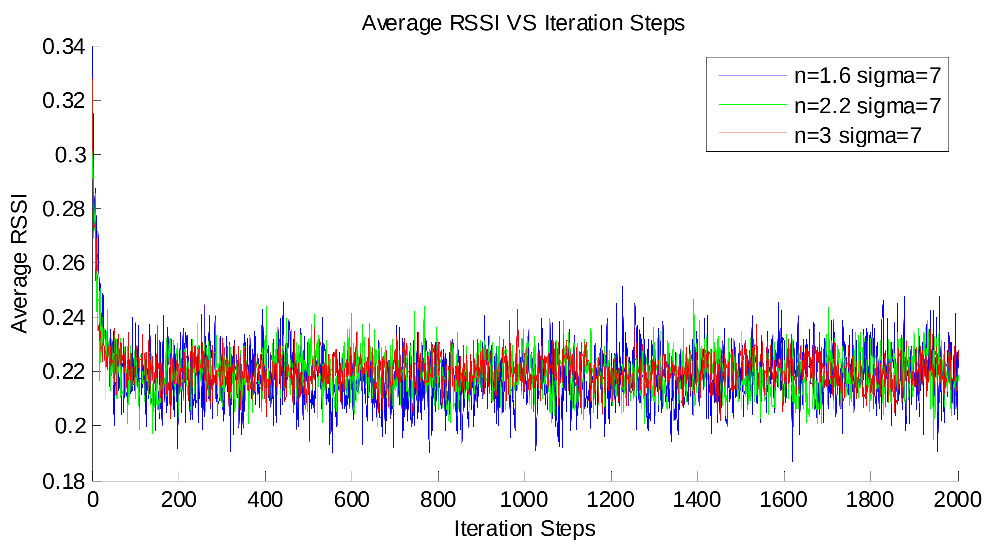


Figure 3.39: Average inter-robot RSSI when σ=7.0 and n=1.6, n=2.2 and n=3.0.

In summary, Fig. 3.37-3.39 show that, with the proposed control scheme, the MRS can form and maintain an effective communication coverage with desired average RSSI in different environments. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of the average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.5.4 Conclusion

A distributed adaptive control scheme, based on the radial basis network (RBN), is proposed to establish and maintain wireless communication connections among multiple mobile robots. The proposed strategy trains an onboard RBN controller through a reinforcement learning process to make it adapt to an unknown environment, and uses the resulting controller to onboard decide the communication-adaptive motion of a robot with respect to one neighbor. When a robot has multiple neighbors, one RBN is defined to deal with each neighbor. The decisions on the movement of a robot to adapt to the real-time wireless link quality with all its neighbors are made by aggregating the RBNs corresponding to all the neighbors. The combined effect of the movements of all the robots results in a desired communication connectivity across the whole multi-robot system. The simulation results show that the proposed control strategy in general guides the whole multi-robot system to converge to the desired connectivity while forming a coverage across the environment.

3.6 Fuzzy Neural Network Trained by Reinforcement Learning

This section presents a decentralized multi-robot motion control strategy to facilitate a multi-robot system, comprised of collaborative mobile robots coordinated through wireless communications, to form and maintain desired wireless communication coverage in a realistic environment with unstable wireless signaling condition. A fuzzy neural network controller is proposed for each robot to maintain the wireless link quality with its neighbors. The controller is trained through reinforcement learning to establish the relationship between the wireless link quality and robot motion decision, through consecutive interactions between the controller and environment. The tuned fuzzy neural network controller is applied to a multi-robot deployment process to form and maintain desired wireless communication coverage.

The layout for the rest of the section is as follows. Section 3.6.1 will explain the specific definition of the proposed reinforcement learning fuzzy neural network controller based on real-time link quality measurement. Section 3.6.2 will report the simulation results based on a well-adopted probabilistic communication signal propagation model. Section 3.6.3 will summarize the contribution of this section.

3.6.1 Proposed Fuzzy Neural network Trained by Reinforcement Learning

The objective of this section is to deploy a team of robots into an unknown environment to establish and maintain desired wireless communication coverage. Due to its high adaptability and scalability, we design a distributed scheme to control the deployment process, i.e. a local controller for each robot. Since the wireless communication condition is completely unknown in such an environment, reinforcement learning is applied to train a fuzzy neural network controller for a robot to adapt to the environment and maintain the desired wireless link quality with a neighbor. A local controller for each robot is defined by the aggregation of all the FNNs, corresponding to all its neighbors, to adapt to the quality of wireless links with all its neighbors. The combined effect of all the robots' movements will allow the multi-robot system to approach and maintain the desired communication connectivity.

The proposed control scheme consists of two parts:

1) Learning: The FNN controller of a robot interacts with an unknown environment and learns the communication-adaptive motion policy according to the real-time feedback of quality of its wireless links with its neighbors. In order to keep the complexity of network reasonable and the scalability of the control scheme, the FNN is designed for a robot to maintain the desired wireless communication with one neighbor.
2) Decision making: when a robot has multiple neighbors, a FNN controller is assigned to each neighbor. The motion of each robot is decided by an onboard local controller which aggregates the FNNs for all neighbors.

A fuzzy neural network controller is designed, based on the proposed fuzzy logic controller in Section 3.2.2, to decide the communication-adaptive motion of a robot according to the real-time feedback of the wireless link quality with one neighbor. The final movement of a robot is generated by merging all neighbors' decisions. Following the fuzzy neural network structure described in Section 3.4.1 (Fig. 3.25), the proposed fuzzy neural network consists of an antecedent (lower) network and a consequent (upper) network (Fig. 3.40).

The antecedent network maps the RSSI to the firing strength of each rule:

Layer 0 (Input): Input $RSSI_{i,j}$ is a measured RSSI between the robot R_i and a neighbor R_j.

Layer 1 (Fuzzification): The inputs are fuzzified using the transfer functions ($f_{i,j,k}$) of this layer, where $f_{i,j,k}$ denoting the value of kth membership function associated with robot R_i and its neighbor R_j, to determine the membership degree of each input in each fuzzy set, i.e. $\mu^k_{i,j} = f_{i,j,k}(RSSI_{i,j})$.

Layer 2 (Normalization): Layer 2 defines the fire strengths of the corresponding rules by normalizing the outputs of layer 1 using Eq. (3.10). Since the antecedent of each rule has only one part, there is no need for a separate layer for fuzzy operation, which is needed in the general case (Section 3.4.1). So, the inputs of this layer can be determined by $w^k_{i,j} = \mu^k_{i,j}$, where $w^k_{i,j}$ is the fire strength associated with R_j through the kth rule.

The consequent network maps the bearing angles between robot R_i and its neighbors to a motion decision for R_i:

Layer 0 (Input): The pair of input, $\cos\theta_{i,j}$ and $\sin\theta_{i,j}$, are the cosine and sine values of the bearing angle $\theta_{i,j}$ between robot R_i and a neighbor R_j.

Layer 1 (Implication): This layer defines the motion decision towards each neighbor under each level of wireless link quality. The transfer functions are defined as $Dx^k_{i,j} = \cos\theta_{i,j} \bullet L^k_{i,j}$, $Dy^k_{i,j} = \sin\theta_{i,j} \bullet SL^k_{i,j}$ where $SL^k_{i,j}$ denote the step length associated with its neighbor R_j by kth fuzzy rules. And $Dx^k_{i,j}$ and $Dy^k_{i,j}$ denote the motion decisions along x-axis and y-axis.

Layer 2 (aggregation): The output of FNN is a motion decision vector $\mathbf{D}_{\mathbf{i,j}}$ = [Dx$_{\mathbf{i,j}}$, Dy$_{\mathbf{i,j}}$], obtained as a linear combination of the outputs of layer 1 of the consequent network (the consequences of the rules) that is weighted by the outputs of the antecedent network (the fire strengths of the rules) using Eq. (3.12).

The final motion decision of a robot decided by the sum of all motion decision vectors with its neighbor, i.e. $\mathbf{D}_{\mathbf{i}}$ = [Dx$_{\mathbf{i}}$, Dy$_{\mathbf{i}}$]= [Σ_1^nDx$_{\mathbf{i,j}}$, Σ_1^nDy$_{\mathbf{i,j}}$], where n denotes the number of its neighbor.

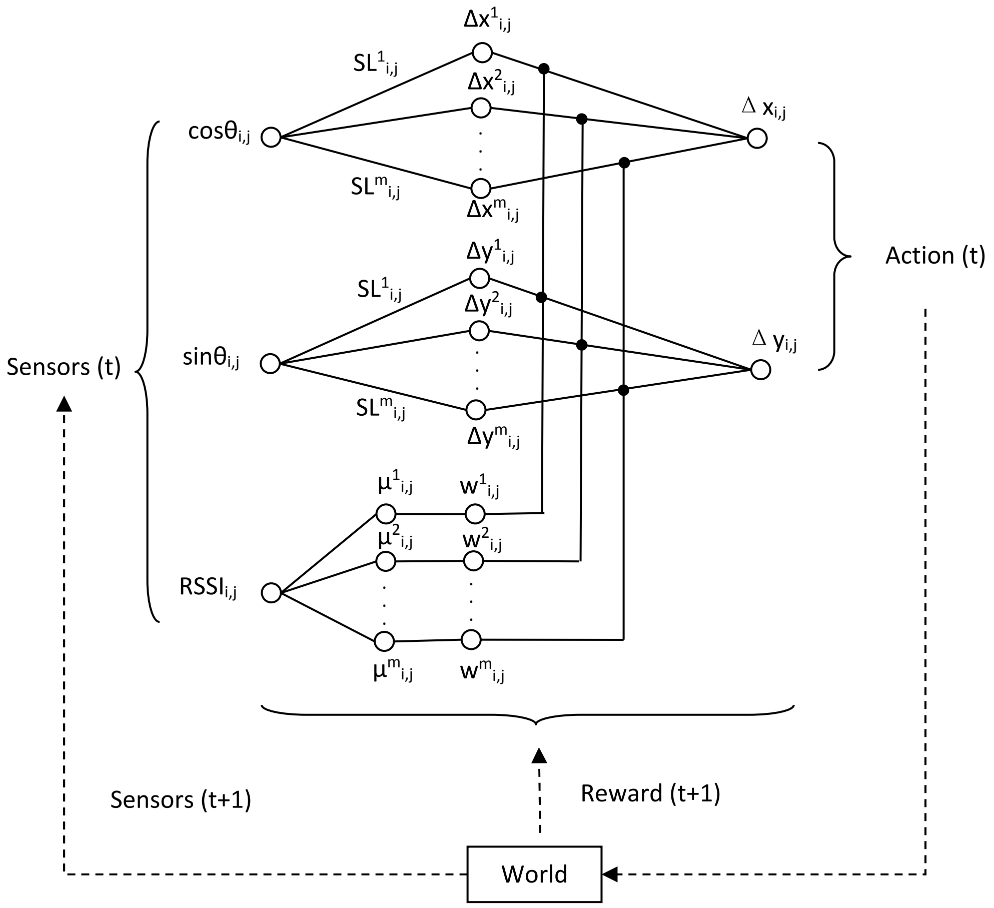


Figure 3.40: Proposed framework of fuzzy neural network and reinforcement learning.

In this section, reinforcement learning is applied to proposed FNN controller to adapt to the wireless link quality for a neighbor through the interaction with an unknown environment. The proposed learning process (Fig. 3.40), based on the general framework of reinforcement learning (Fig. 3.32), is designed to adaptively learn the control policy by tuning the parameters of each membership function (transfer function of layer 1 in antecedent network) and step length of each rule (weights between input and layer 1 in consequent network). Within the general reinforcement learning framework, the corresponding components are defined in following:

1) The "agent" consists of antecedent and consequent network of the proposed fuzzy neural network controller.
2) The "environment" includes real world and sensors. In fact, anything that cannot be changed arbitrarily by the agent is considered to be exclusive, thus becoming part of the environment.
3) The "states" of the environment includes measured wireless link quality, and bearing between the robot and its neighbor.

During the learning process, the interaction between FNN and environment occurs at a sequence of discrete time.

1) At the beginning of learning process, FNN randomly initializes parameters of the membership function and step lengths; and the initial position of robot is randomly picked, following the uniform distribution, within the communication range of a neighbor which does not move during the learning process.
2) At time t, the position of the robot is updated by previous action (t-1). The FNN perturbs related parameters by Eq. (3.19) and (3.20), i.e. make random small changes to all the membership functions and step lengths,

$$MF = MF + \alpha \times ERR_c \times randn\left(size\left(MF\right)\right) \tag{3.19}$$

$$SL = SL + \alpha \times ERR_c \times randn\left(size\left(SL\right)\right) \tag{3.20}$$

Where MF denotes the parameters of membership functions, α denotes the learning rate, randn (size (mf)) generates a matrix which contains pseudorandom values drawn from the standard normal distribution and has same size as mf, SL denotes the step lengths, and ERR_c denotes the current difference between the desired RSSI and the RSSI measurement at the output position after each action. The perturbations of the parameters can be adaptively adjusted by introducing the ERR_c to the perturbations, where the perturbations will decrease/increase as the ERR_c decreases/increases. The FNN then decides the next motion of the robot (action (t)) according to the wireless link quality and bearing (sensors (t)).

At time step t+1, as the consequence of action (t), the FNN receives a numerical reward (reward (t+1)) determined by the reward function F^r, and chooses whether or not to update the parameters from the perturbations.

$$F^r = ERR_c - abs(RSSI_c - RSSI_d) \tag{3.21}$$

Where abs() is the absolute value operator, $RSSI_c$ denotes the measured RSSI after current action, and $RSSI_d$ denotes the desired RSSI. If F^r is greater than 0, FNN will update MF and SL with the perturbations. Otherwise FNN will not update any parameters. The strategy, indicated by the definition of reward function, is that the controller will find the membership functions and step lengths that can produce the actions leading towards the desired RSSI. At the same time, the FNN receives sensors (t+1) as the new input, and continues with the learning process.

Through the learning process, FNN is trained to establish a motion decision policy for the robot to adapt to the wireless link quality with each neighbor, and a motion decision can be made for each robot

3.6.2 Simulations

The proposed distributed control scheme for maintaining desired communication coverage in a multi-robot deployment scenario has been evaluated through simulations.

3.6.2.1 Result of learning process

The goal for this experiment is to maintain desired wireless communication coverage for MRS in an unknown environment with unpredictable interference. Doubtlessly, the quality of wireless connection is directly proportional to the Received Signal Strength Indication which higher RSSI value is preferable for a stable data exchange among robots. However, the physical range of wireless coverage also needs to be optimized which definitely will reduce the RSSI value, and a desired range of RSSI value is defined to maximize the physical range of the coverage of MRS. Thus, the proposed scheme takes RSSI as the input to the FNN. Each FNN has one input (RSSI) and one output (motion decision **D** = [$D_x = d\cos(\theta)$, $D_y = d\sin(\theta)$], where θ denotes the bearing between the robot and its neighbor. When a robot has multiple neighbors, one FNN is needed to deal with each neighbor. The final motion decision is the sum of the outputs of all the FNNs.

The proposed fuzzy neural network controller, defined in Section 3.6.1, is trained through a sequence of interaction with the environment, using the reinforcement learning process described in Section 3.5.1. Be more specifically, during an arbitrary learning process, a robot follows the motion decision made by the FNN based on the current RSSI condition with a fixed neighbor, and then adjusts the parameters of its onboard FNN controller through the reward feedback after executive movement. Fig. 3.41 shows a few typical learning curves in a representative environment, Fig. 3.42 shows the mean and standard deviation of these learning curves. While the difference and fluctuation among the learning curves result from the randomness of the wireless communication condition, the simulation results indicate that in average the fuzzy neural network controller converges through the learning process. In the case of Fig. 3.41-3.42, the learning process converges after 70 steps in average. The resulting controller will move the robot to approach the desired RSSI with the neighbor.

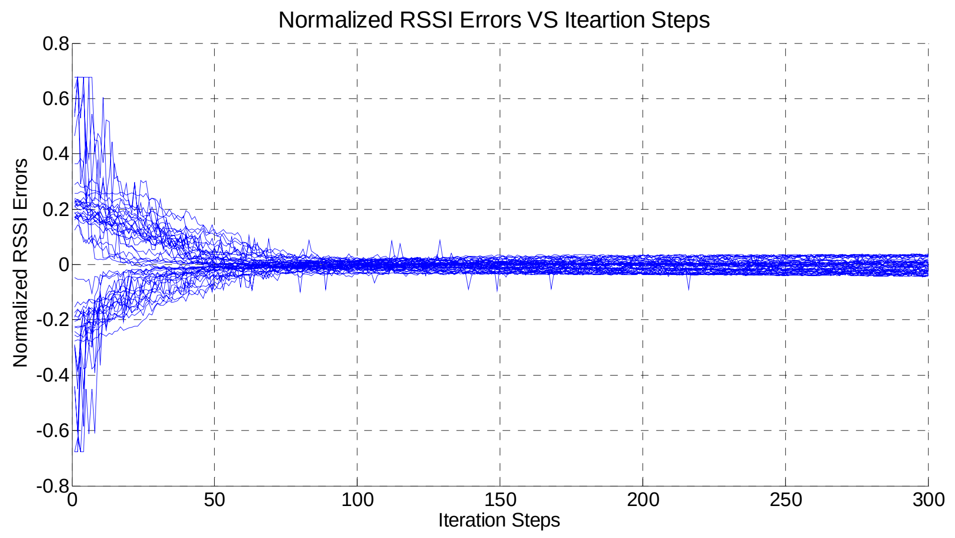


Figure 3.41: Evolution of the difference between the desired and measured RSSI when n=3.0 and σ=7.0.

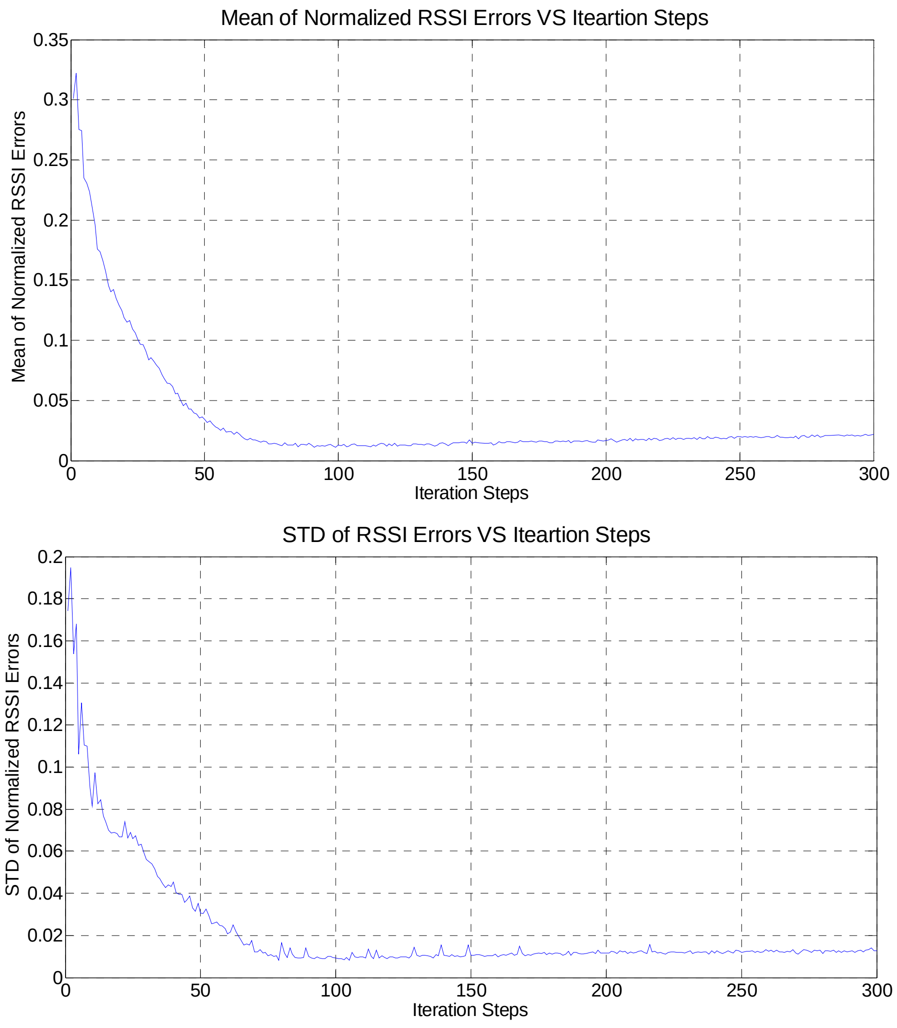


Figure 3.42: Evolution of the mean and standard deviation of the difference between the desired and measured RSSI when n=3.0 and σ=7.0.

3.6.2.2 Results of the deployment process

A typical deployment configuration resulting from the proposed control scheme is shown in Fig. 3.43, including 50 robots after 2000 deployment steps, where the underlying neighborhoods are visualized using Delaunay graph. It shows that the proposed control scheme is capable of forming an effective network coverage under different levels of uncertainty in signal propagation or different path loss exponents.

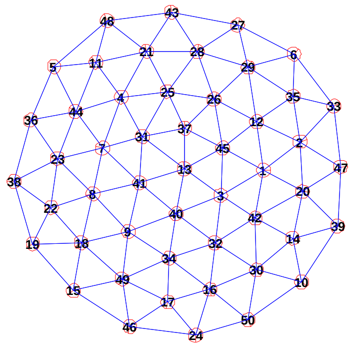


Figure 3.43: Typical configurations when n=3.0 and σ=7.0.

The evolution of the average RSSI is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig. 3.44 and 3.45 show the average RSSI in general converges towards the desired RSSI value 0.22 as the MRS approaches a communication coverage. The fluctuation of the average RSSI results from the reactive response to the randomness in signal propagation. Fig. 3.44 presents the level of fluctuation increases as the uncertainty in signal propagation increases; meanwhile Fig. 3.45 provides that the fluctuation of the average RSSI decreases as the path loss exponent increases.

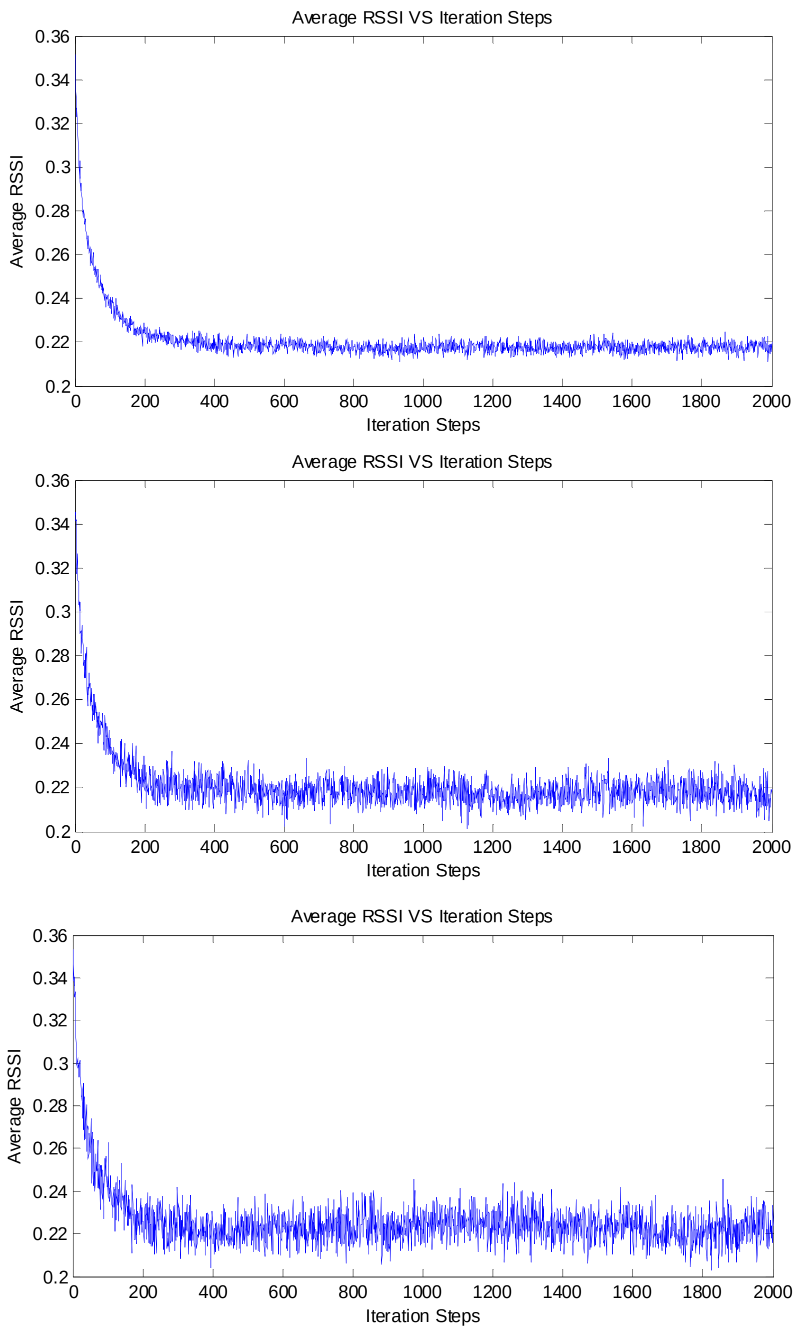


Figure 3.44: Average inter-robot RSSI when n=3.0 and σ=3.0, σ=7.0, and σ=9.6.

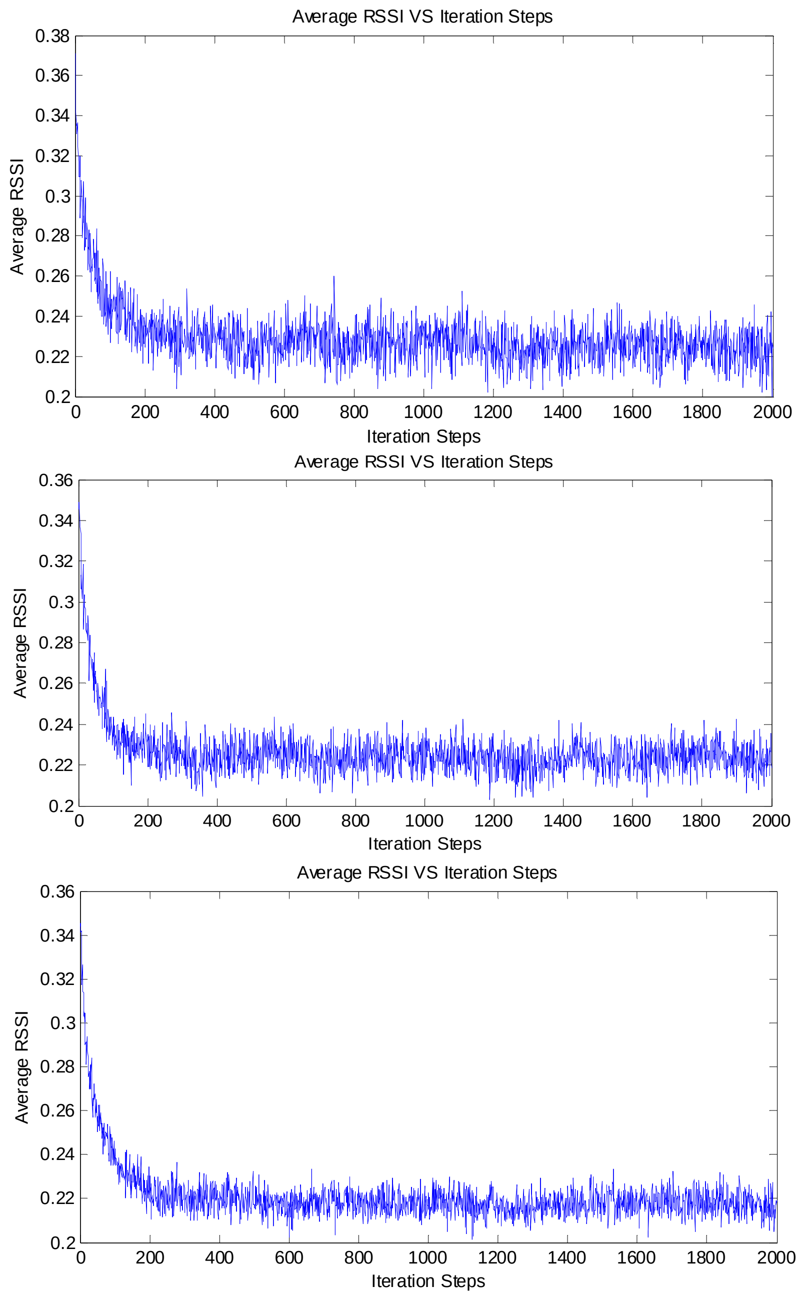


Figure 3.45: Average inter-robot RSSI when σ=7.0 and n=1.6, n=2.2, and n=3.0.

In summary, Fig. 3.43-3.45 show that, with the proposed control scheme, the MRS can form and maintain an effective communication coverage with desired average RSSI in different environments. In spite of the fluctuation in average RSSI due to the uncertainty in signal propagation, the evolution of the average RSSI, in particular the mean of the average RSSI, does show a general trend of convergence to the desired wireless link quality.

3.6.3 Conclusion

The proposed strategy trains the onboard FNN controller through a reinforcement learning process to make it adapt to an unknown environment, and uses the resulting controller to onboard decide the communication-adaptive motion of a robot in a two-robot system. Same terminology is applied to simulation system that one FNN is defined to deal with each neighbor. The movement decision of a robot would be executed by aggregating the FNNs corresponding to all the neighbors according to the Received Signal Strength Indication value through continuous iteration, until a satisfied range of communication connectivity is achieved, where MRS would maintain the best performance. Our simulation result indicated that the proposed strategy maintains multiple robot system in optimal wireless network connectivity coverage.

3.7 Comparison and Conclusion

The objective of this chapter is to maintain the desired wireless communication coverage among a team of mobile robot. Based on local perceived wireless link quality, we propose several decentralized intelligent control schemes to approach a desired wireless communication coverage in a target environment. In this chapter, five intelligent controllers are defined, i.e. fuzzy logic (FL), artificial neural network trained by back propagation algorithm (ANN BP), fuzzy neural network trained by back propagation algorithm (FNN BP), radial basis network trained by reinforcement learning (RBN RL) and fuzzy neural network trained by reinforcement learning (FNN RL). These intelligent schemes are introduced for each robot as an onboard controller to adapt to the quality of the wireless links with its neighbors. The combined effect of all robots' movement allows the MRS to achieve the desired communication connections. The effectiveness of the proposed control schemes are verified through simulations under different wireless signal propagation conditions. The characteristics of these controllers are listed in Table 3.2.

Table 3.2: Comparison of intelligent controllers.

Method	Architecture	Self-tuning	Training Data	Training Mode
FL	Human knowledge	N/A	N/A	N/A
ANN BP	Try and error	Y	Off line	Batch
FNN BP	Human knowledge	Y	Off line	Batch
RBN RL	Try and error	Y	On line	Incremental
FNN RL	Human knowledge	Y	On line	Incremental

In Table 3.2, the first column indicates the control method, which include all proposed control schemes in this chapter. The second column indicates the control framework. The control frameworks of FL, FNN BP and FNN RL are designed based on expert knowledge. Unlike other three control frameworks, ANN BP and RBN RL are designed by the method of try and error, such as the number of nodes in hidden layer. Self-tuning denotes the ability of adjust its own parameters to achieve the control objective. Training data denotes the acquisition of training data. Training mode denotes the way of using training data. In incremental training, the parameters of the network are updated each time an input is presented to the network. In batch training, the parameters of network will update after all data pairs presented.

As shown in Table 3.2, each control scheme has advantages and disadvantages. Nevertheless, any proposed control schemes can maintain the desired wireless communication coverage in a multi-robot system. And the selection of control method depend on the specific applications. In chapter 4, we will apply fuzzy logic controller to a behavior-based control framework for a multi-robot system.

Chapter 4 Application of Wireless Communication Maintaining

This chapter proposes a behavior-based fuzzy control framework for controlling a multi-robot system (MRS) to approach desired sensory coverage in an unknown environment while maintaining wireless communication connections and avoiding collisions. Such an MRS consists of multiple mobile robots with limited onboard sensing and wireless communication capabilities. Each robot can perform two categories of behaviors: 1) fundamental behaviors, e.g. forming sensory coverage, maintaining wireless communication connections, and avoiding collisions, are necessary for reliable functioning of the MRS; 2) application behaviors, e.g. search and exploration, are defined to fulfill the goals of specific deployment tasks. In the proposed control architecture, these behaviors are realized by appropriately-defined parallel fuzzy controllers with different priorities, and behaviors can easily be added or removed by adding or removing corresponding parallel fuzzy controllers without affecting other behaviors.

The layout for the rest of the chapter is as follows. Section 4.1 will introduce the control framework and define the proposed behavior-based fuzzy controllers. Section 4.2 will report the simulation results based on the well-adopted probabilistic log-distance path loss model. Section 4.3 will summarize the contribution of this chapter.

4.1 Proposed Behavior-based Motion Control Framework

The fundamental considerations and the simulation setups are referred to Section 3.1. The knowledge of fuzzy inference can be found in Section 3.2.1.

The objective of the presented control scheme is to establish the sensory coverage and the wireless communication without any collision for specific deployment tasks, such as search, exploration, monitoring etc. Behaviors of each robot are defined into two groups with different priorities, fundamental (with higher priorities) and application behaviors (Fig. 4.1). Fundamental behaviors include:

1) Sensory coverage: moving the robot relative to its neighbors to approach desired coverage according to the sensor range.

2) Collision avoidance: preventing the robot from colliding with other robots and obstacles according to the relevant distances.

3) Wireless connection maintenance: maintaining the robot's wireless communication connections with neighboring robots according to the received signal strength indication (RSSI).

In this chapter, we define an application behavior as destination approaching which moves the robot from an initial position into a target environment.

Four parallel fuzzy controllers are designed to realize the three fundamental behaviors and one application behavior (Fig. 4.1). The combined effect of the three fundamental controllers lead to a fundamental objective, i.e. to maintain a desired distance and wireless link quality without any collision, while the combination of all the four controllers will guide the multi-robot system from an initial position into a target area and approach the fundamental objective at the same time. The reason for us to design the fuzzy controllers in a parallel way is because:

1) each behavior can be easily removed or added by removing or adding the corresponding fuzzy controller;

2) each behavior can be easily adjusted to adapt to different robot systems and environments by adjusting the corresponding controller without any influence on other behaviors.

Collision avoidance and communication maintenance have the highest priorities due to the risk of the damage of robots and the loss of connections. Since fundamental behaviors are the necessary condition for reliable functioning of MRS, the application behaviors have lower priorities in general.

All proposed fuzzy controllers are defined by the Sugeno-type fuzzy inference [58-60], because it matches with the intended control framework. The inputs for each fuzzy controller are the robot's distances from its neighbors and/or the RSSI with its neighbors, while the output of each fuzzy controller is a motion decision vector for next movement that the robot should take according to this controller (behavior). The final motion decision for this robot is the sum of all the motion decision vectors resulting from all the fuzzy controllers of this robot.

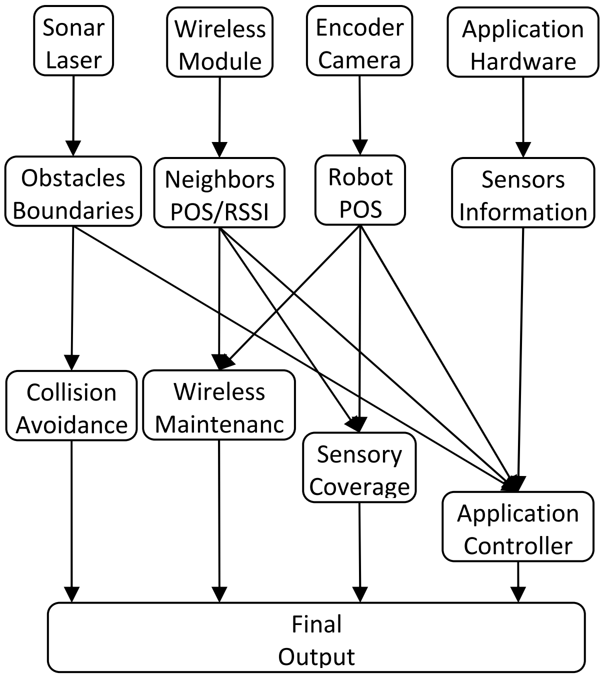


Figure 4.1: Priority-based behavior fuzzy control diagram.

To adjust the priority easily, the fuzzy rules are defined in the same format, i.e. "If the value of the distance and/or RSSI that robot R_i with a neighbor R_j is at the level F, then R_i should move towards R_j with a certain step length L". The final output for each behavior-based fuzzy controller is determined by the weighted average of all the rules. The behavior priority can be tuned by the step length L. In this manner, the overlap parts of different behaviors are dominated by the behaviors with higher priorities.

The following subsections describe the four fuzzy controllers separately by fuzzy sets, rules and process.

4.1.1 Sensory Coverage Fuzzy Controller

The control objective of sensory coverage for a robot R_i is to approach and maintain a desired distance with its neighboring robots.

Fuzzy sets (F^S): We define five fuzzy sets out of the distance range, from the uncovering to crowding distance, corresponding to a linguistic division of the distance levels:

1) Desired distance—indicating that the distance is desirably good for the sensory coverage behavior.
2) Uncovering distance—indicating that the robot is out of the sensor range.
3) Pro-uncovering distance (transitional part)—indicating that the robot is getting out of the sensor range.
4) Crowding distance—indicating that the robots are crowding close together.
5) Pro-crowding distance (transitional part)—indicating that the robots are getting crowding together.

The corresponding membership functions for these fuzzy sets can be defined by the functionality of the onboard sensor or the requirements of specific task, with an example provided in Section 4.2.1.

Fuzzy rules: The rules are set to reduce the uncovering distance, maintain the desired distance, and enlarge the crowding distance. Each individual rule has the form of " If the value of distance $d_{i,j}$ that robot R_i with a neighbor R_j is at the level F^S, then R_i should move towards R_j with a certain step length L^S", where F^S denotes any of the above-defined five fuzzy distance sets for the sensory coverage control. The step length L^S of the robot movement towards a neighbor is defined according to the level of distance with that neighbor and the priority level of the behavior.

The consequent of each rule is a motion decision vector $\mathbf{D}^S$ which requires R_i to move a distance L^S along the straight line connecting R_i and R_j. Specifically in a planar environment, $\mathbf{D}^S = [\ L^S * \cos(\theta_{ij}),\ L^S * \sin(\theta_{ij})\ ]^T$, where θ_{ij} denotes the orientation angle of the line connecting R_i and R_j in the frame of reference of the environment. When R_i has multiple neighbors, these five rules apply to each neighbor.

Fuzzy interference: all the fuzzy controllers are following the similar process, but with different inputs and outputs.

1) Fuzzifying inputs and calculating firing strengths: The distance with each neighbor is used as the input to the membership functions to obtain the corresponding membership values, with $f_{i,j,k}(\ d_{i,j}\ )$ denoting the value of the kth membership function associated with R_i and R_j, which are used as the firing strengths of the rules.
2) Implicating consequents: The consequent of each rule is a motion decision vector $\mathbf{D}_{i,j,k}$ based on the distance.
3) Aggregating consequents: The comprehensive motion decision for R_i to adapt to all the neighboring distances is calculated by the weighted average across all the rules as Eq. (4.1), which defines next movement that R_i should make, where $w_{i,j,k} = f_{i,j,k}(\ d_{i,j}\ )$ is the fire strength.

$$\mathbf{D}_i = \sum_{j,k} [w_{i,j,k} \mathbf{D}_{i,j,k}] \Big/ \sum_{j,k} w_{i,j,k} \tag{4.1}$$

4.1.2 Collision Avoidance Fuzzy Controller

The collision avoidance controller aims at maintaining a safe distance among robots and obstacles, which means that it only works in the range of colliding/dangerous distance. Compared with sensory coverage controller, it has a higher priority and can avoid any objects, such as obstacles and other robots in the environment. In this subsection, a neighbor for each robot refers to any object.

Fuzzy sets (F^C): Three fuzzy sets are defined out of the distance range, from the colliding to safe distance, corresponding to the following division of the distance levels:

1) Safe distance—the range of distance indicating that the distance is long enough to avoid collision.

2) Colliding distance—the range of distance indicating that robot is colliding with its neighbor.

3) Alerting distance (transitional range)—the range of distance indicating that robot is getting colliding with others.

Fuzzy rules: Three fuzzy rules are defined, corresponding to each fuzzy set, to move away from a neighbor with colliding/alerting distance. Each individual rule has the same format defined in Section 4.1.1.

4.1.3 Wireless Maintenance Fuzzy Controller

In this part, we present our fuzzy control scheme for maintaining wireless communication quality in a MRS.

Fuzzy sets (F^W): Poor wireless link quality means the risk of disconnection or packet loss. We define three fuzzy sets out of the RSSI range, from the losing to desired RSSI, corresponding to the following division of the RSSI levels:

1) Desired RSSI—indicating that the wireless quality is desirably good for the underlying application.

2) Losing RSSI—indicating that the wireless quality is poor. If the RSSI is lower than this level, the received data packets may not be decoded, which leads to the loss of the associated neighbor.

3) Weak RSSI (transitional range)—indicating that the wireless quality is getting poor.

Fuzzy rules: The rules are set to improve the poor, and maintain the desired wireless link quality. Each individual rule has the same format defined in Section 4.1.1.

4.1.4 Application Fuzzy Controller

Application controller is designed to meet the requirement of specific deployment application. In this chapter, the destination approaching is defined to guide the multi-robot system to a target area.

Fuzzy sets: We define two groups of fuzzy sets for distance and one group of fuzzy set for RSSI, which are same as the definitions in Sections 4.1.1-4.1.3.

Fuzzy rules: The rules are set to move the robot to the target area when both the wireless link quality and distance for each neighbor are close to the desired. Each individual rule has the form of " If the value of distance $d_{i,j}$ and $RSSI_{i,j}$ that robot R_i sense with a neighbor R_j is at the level F^S , F^C and F^W, then R_i should move towards the target area with a certain step length L^A ", where F^S, F^C and F^W denotes any of the previous-defined fuzzy sets in Sections 4.1.1-4.1.3, and L^A denotes the step length for the destination approaching behavior. And the consequent of each rule is a motion decision vector $\mathbf{D}^A$. Since each rule has three inputs (five, three and three fuzzy sets), we define forty-five rules total for all the combinations. When R_i has multiple neighbors, the final outputs will be the "and" operate on each neighbor's output, which is calculated by applying these forty-five rules to each neighbor.

4.2 Simulations

The proposed behavior-based fuzzy control framework for establishing desired sensory coverage and maintaining wireless communication in a multi-robot deployment scenario has been evaluated through simulations.

4.2.1 Definition of Membership Functions

The four fuzzy controllers, defined in Section 4.1, are designed to realize the fundamental and application behaviors. Following the format of fuzzy sets defined in Sections 4.1.1-4.1.3, we define a membership function (MF) for each fuzzy set. Here, the trapezoid-shaped MFs are adopted, because they provide an effective and simple description of the fuzziness at which we judge the distance and RSSI. Each MF consists of a central part with full membership (set to 1) and boundary parts with partial membership between 0 and 1. The MFs for the fundamental behaviors are listed as follows.

1) Sensory coverage

We define five MFs in Fig. 4.2 corresponding to the fuzzy sets for sensory coverage (Section 4.1.1): C (crowding distance), PC (pro-crowding distance), D (desired distance), PU (pro-uncovering distance) and U (uncovering distance). The central boundary parts are determined by sensory range.

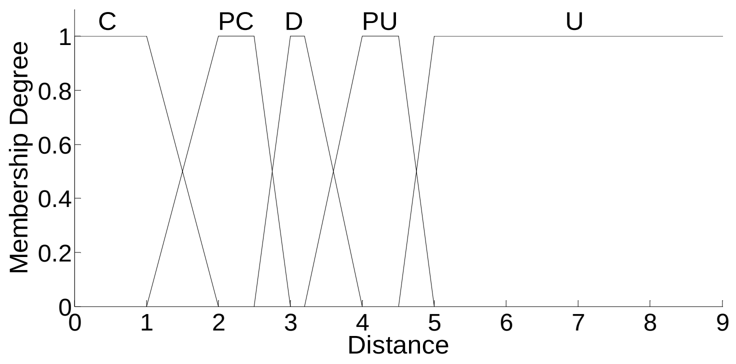


Figure 4.2: Membership functions of sensory coverage.

2) Collision avoidance

Three MFs are defined in Fig. 4.3 corresponding to the fuzzy sets for collision avoidance (Section 4.1.2): C (colliding distance), D (dangerous distance) and S (safe distance). The central and boundary parts are determined by the radius of each robot.

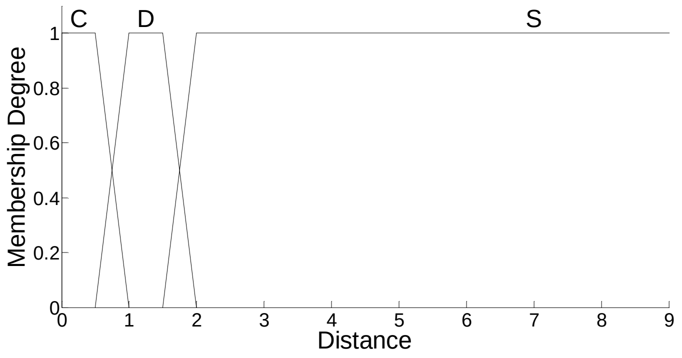


Figure 4.3: Membership functions of collision avoidance.

3) Wireless maintenance

Three MFs are defined in Fig. 4.4 corresponding to the fuzzy sets for wireless maintenance (Section 4.1.3): L (losing RSSI), W (weak RSSI), D (desired RSSI). The central and boundary ranges of these MFs are chosen based on the RSSI resulting from the log-distance path loss model. The simulated RSSI values are scaled into the range [0, 1].

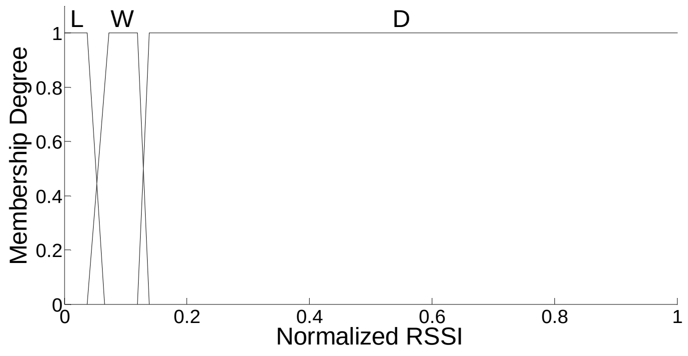


Figure 4.4: Membership functions of wireless maintenance.

4.2.2 Simulation Results

The simulations are designed to verify the effectiveness of the behavior-based fuzzy control scheme for the deployment of MRS. We conduct the simulations for the fundamental behaviors and application behaviors in an open environment and a constrained environment as following:

1) fundamental behaviors in an open environment;
2) fundamental and application behaviors in an open/constrained environment with/without obstacles.

Each simulation starts with 50 robots initially gathering with random positions and orientations in a 5-by-5 square planar region, following the uniform distribution.

Fig. 4.5 shows a system configuration for the fundamental behaviors, i.e. sensory coverage, collision avoidance and wireless maintenance, in an open environment, where the underlying neighborhood relationships are visualized using Delaunay graph. It shows that the proposed fundamental behavior fuzzy controllers are capable of forming an effective network coverage in an open environment.

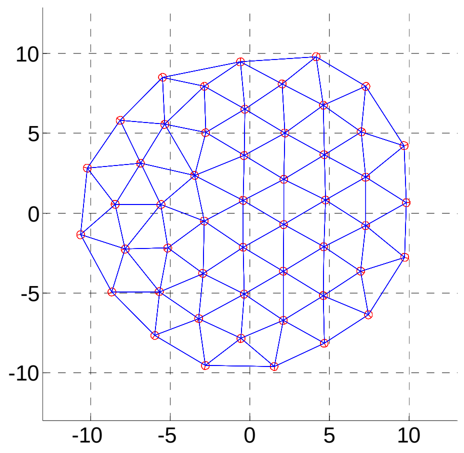


Figure 4.5: Configuration for the fundamental behaviors in an open environment.

Fig. 4.6 shows the system initial, intermediate and final configuration in the free space for the fundamental with the exploration behaviors, i.e. sensory coverage, collision avoidance, wireless maintenance and destination approaching in the constrained environment with obstacles. Fig. 4.7-4.8 show the intermediate configuration in the indoor environment without/with obstacles for the fundamental with the exploration behaviors, where the underlying neighborhood relationships are visualized using Delaunay graph. Fig. 4.6-4.8 show that the presented fundamental and destination approaching fuzzy controllers are capable of deploying a team of robot into an unknown target environment to establish sensory coverage for the destination approaching tasks.

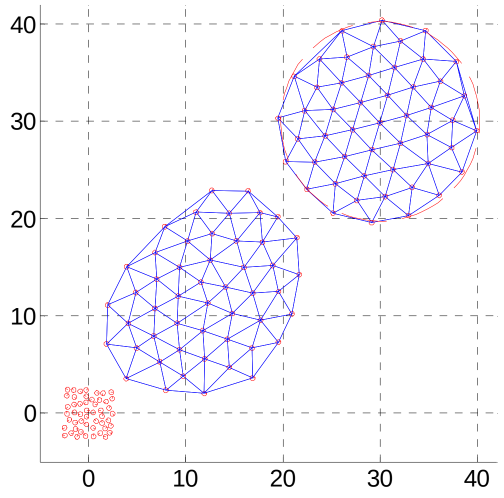


Figure 4.6: Free space configuration for exploration.

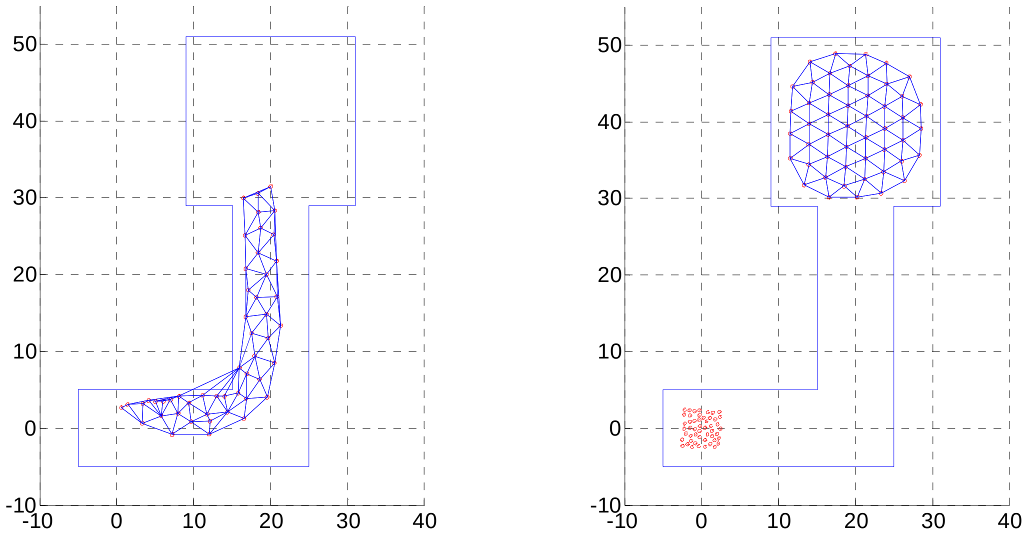


Figure 4.7: Indoor configuration for exploration.

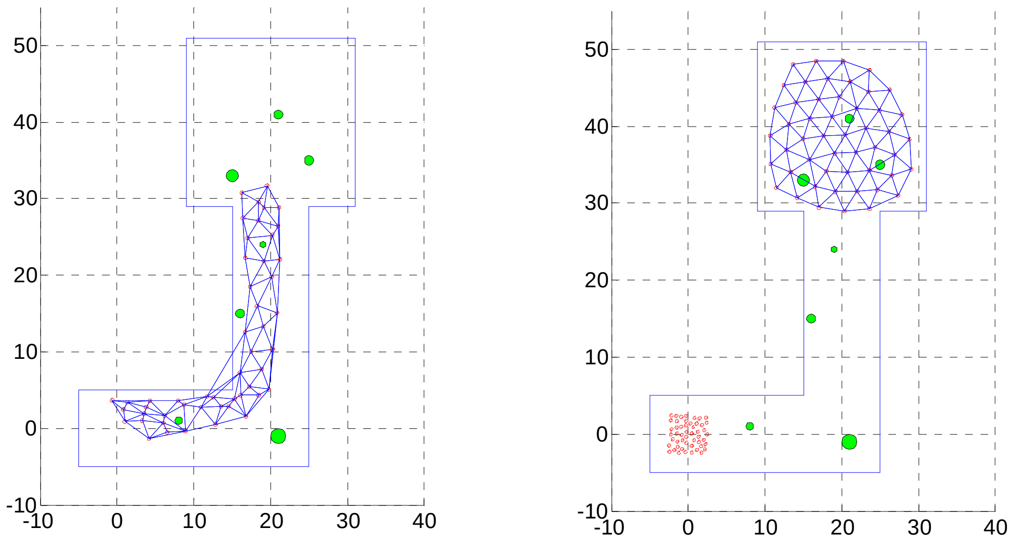


Figure 4.8: Indoor configuration with obstacles for exploration.

To check the performance of wireless communication maintenance, the evolution of the average RSSI is calculated by averaging the RSSI among neighboring pairs of robots at each moment. Fig. 4.9-4.12 show that the average RSSI in general converges towards the range of desired RSSI [0.15, 1] as the MRS maintains the wireless communication.

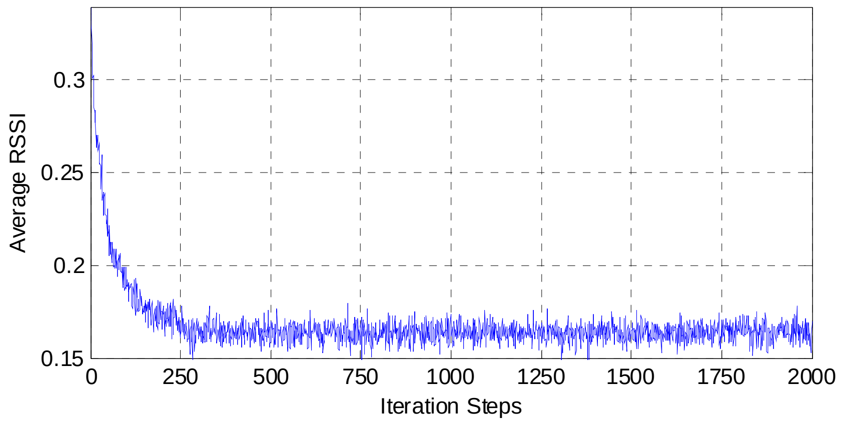


Figure 4.9: Average inter-robot RSSI for fundamental behaviors.

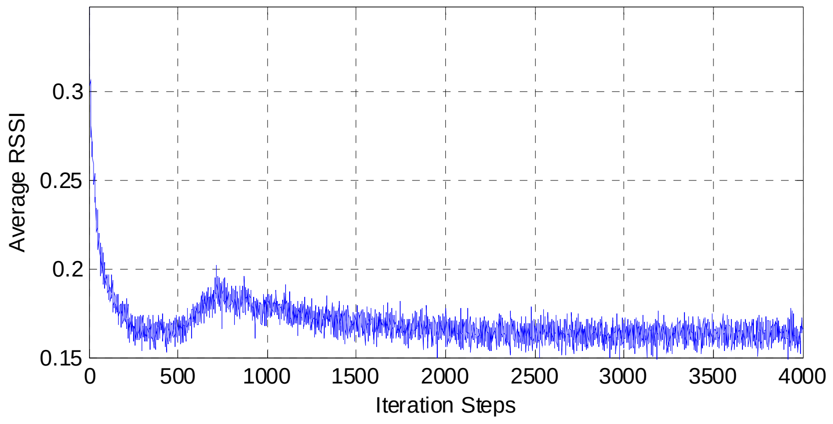


Figure 4.10: Average inter-robot RSSI for free space exploration.

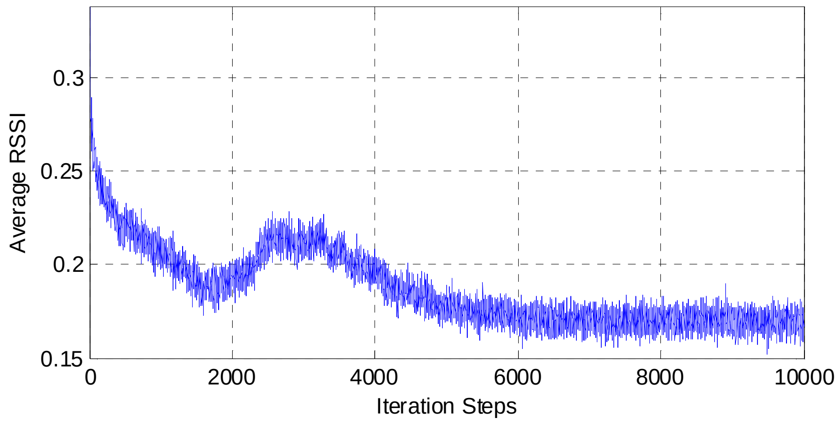


Figure 4.11: Average inter-robot RSSI for indoor exploration.

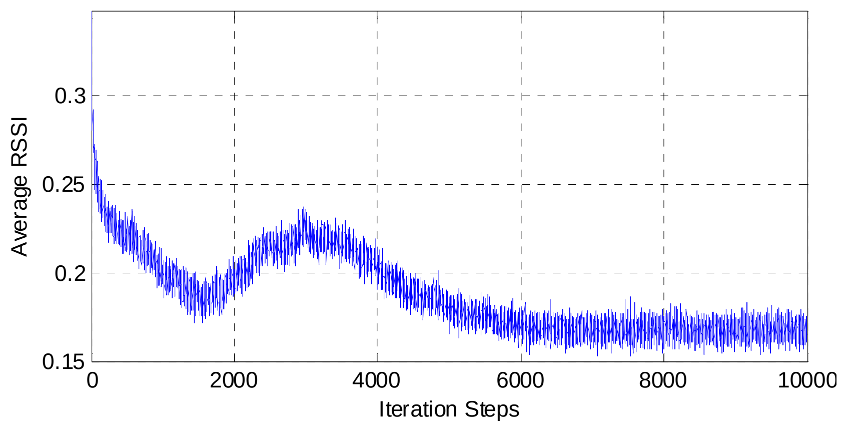


Figure 4.12: Average inter-robot RSSI for indoor exploration with obstacles.

To check the performance of sensory coverage, the evolution of the average distance is calculated by averaging the distance among neighboring pairs of robots at each moment. Fig. 4.13-4.16 show that the average distance in general converges towards the range of desired distance [3, 3.2] as the MRS approaches the desired sensory coverage.

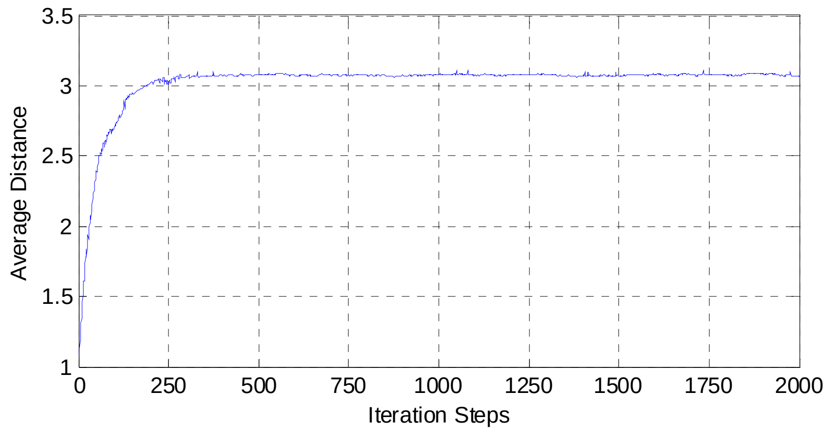


Figure 4.13: Average inter-robot distance for fundamental behaviors.

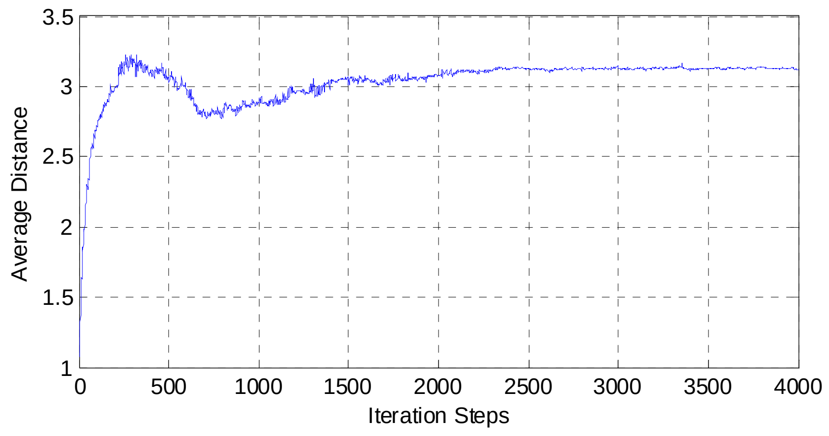


Figure 4.14: Average inter-robot distance for free space exploration.

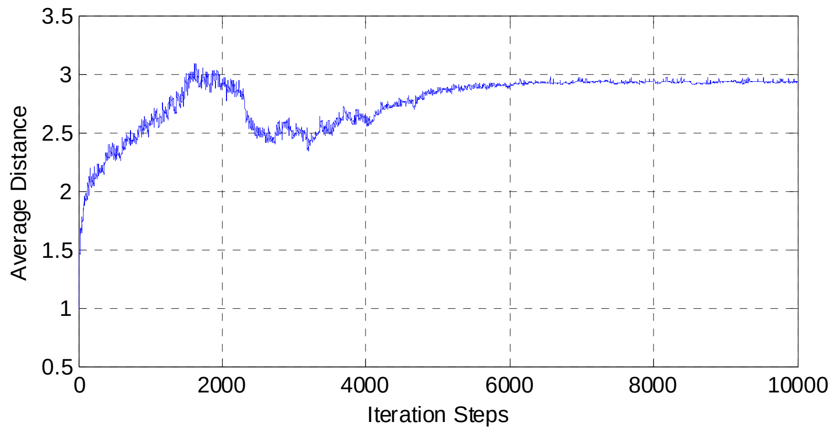


Figure 4.15: Average inter-robot distance for indoor exploration.

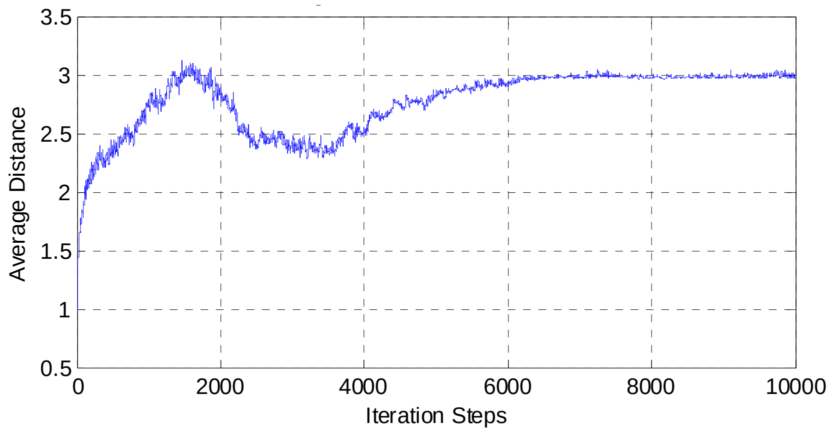


Figure 4.16: Average inter-robot distance for indoor exploration with obstacles

In summary, Fig. 4.6-4.16 show that, with the proposed behavior-based parallel fuzzy architecture, the MRS can form sensory coverage with desired distance while maintaining wireless communication with desired RSSI for the deployment tasks. Moreover, during these deployment processes, robot-robot and robot-obstacle collisions are avoided with the proposed controllers.

4.3 Conclusion

This chapter proposes a behavior-based control scheme to establish a sensory coverage to an unknown target environment while maintaining wireless communication connections and avoiding obstacles for specific deployment tasks. The proposed approach defines a set of parallel fuzzy controllers for each behavior to onboard make motion decisions for each mobile robot to adapt to the real-time distances and wireless link quality with its neighbors. The combined effect of all these fuzzy controllers results in the realization of the desired deployment goals. The simulation results show that, with the proposed fuzzy control scheme, the MRS can form an effective sensory coverage with wireless communication connections without any collision into a destination environment. The work of this paper provides a general behavior-based fuzzy control framework, which can be easily extended to various deployment applications and tasks.

Chapter 5 Conclusion

This chapter summarizes all controllers on maintaining desired wireless communication connections among multiple mobile robots in an unknown environment. Five decentralized intelligent motion control schemes are proposed based on local perceived wireless link quality.

Considering the uncertainty of wireless link quality, fuzzy logic controller is discussed and designed to approach a desired wireless communication coverage. The effectiveness of proposed fuzzy inference are verified by simulations and experiments, where the results show that proposed fuzzy logic controller is capable of forming an effective network coverage under different levels of uncertainty in signal propagation.

The structure and procedure of fuzzy logic is well defined. However, the parameters of fuzzy logic are defined based on the knowledge of target environment. To seek a self-tuning method, we found that the parameters of artificial neural network can be tuned by learning algorithm through input-output data pairs. Based on back propagation algorithm, an artificial neural network controller is defined and presented. The disadvantage of artificial neural network is that its structure, such as the number of layers, the number of nodes in each layer, the transfer functions for each node, is designed based on the method of try and error. To incorporate the well-defined structure and the ability of self-tuning, we present a fuzzy neural network controller, trained by back propagation to adapt to the wireless link quality. Simulations are conducted to show that both back propagation neural network controllers can achieve the control objective, i.e. leading a team of robot to desired wireless communication.

Although the back propagation algorithm can tune the parameters though data pairs, such data pairs are difficult to collect under some circumstance. We notice that a learning method, reinforcement learning, can on-line adjust parameters. Based on reinforcement learning, radial and fuzzy neural network controllers are defined to adapt to the desired wireless link quality in different unknown environment. The results of learning process indicate that proposed control schemes are converge to desired wireless link quality. And the effectiveness of control schemes are illustrated by the simulations.

Real experiments are carried out to show that the on board fuzzy logic controller can lead a team of robots to maintain the desired wireless communication. Each Pioneer P3DX (mobile platform) is controlled by an onboard laptop by client-server control architecture. A ZigBee module is mounted on each P3DX to communicate with other robots and collect the received signal strength. To avoid the accumulated errors of encoders, a real-time low-cost mobile robot localization scheme is proposed for mobile robots in indoor environments, based on the recognition of artificial landmarks captured by a single onboard camera. The reported experimental results revels that high localization robustness and accuracy to various lighting condition.

An application of proposed wireless maintaining controller is explored in a task of deploying multiple robots into a realistic environment. A behavior-based fuzzy control framework is proposed to achieve the control objective. Behaviors are realized by appropriately-defined parallel fuzzy controllers with different priorities, and can easily be added, removed or adjusted by the same operation of corresponding fuzzy controllers without affecting other behaviors. The effectiveness of the proposed fuzzy control framework has been verified through a series of simulated deployment and destination approaching tasks over unknown environments.

Bibliography

[1] T. Arai, E. Pagello, and L. E. Parker, “Guest editorial advances in multirobot systems,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 655-661, 2002.

[2] T. Balch and R. C. Arkin, “Communication in reactive multiagent robotic systems,” *Autonomous Robots*, vol. 1, no. 1, pp. 27-52, 1994.

[3] R. Grabowski, L. E. Navarro-Serment, C. J. J. Paredis, and P. K. Khosla, “Heterogeneous teams of modular robots for mapping and exploration,” *Autonomous Robots*, vol. 8, pp. 293-308, 2000.

[4] Y. Zou and K. Chakrabarty, “Sensor deployment and target localization based on virtual forces,” in Proceedings of INFOCOM 2003: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, pp. 1293-1303, 2003.

[5] S. Chellappan, X. Bai, B. Ma, D. Xuan, and C. Xu, “Mobility limited flip-based sensor networks deployment,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 2, pp. 199-211, 2007.

[6] S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang, “Deploying wireless sensor networks under limited mobility constraints,” *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1142-1157, 2007.

[7] M. Jenkin and G. Dudek, “The paparazzi problem,” in Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 2042-2047, 2000.

[8] A. Howard, M. J. Mataric, and G. S. Sukhatme, “Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem,” in Proceedings of the 6th International Conference on Distributed Autonomous Robotic Systems, pp. 299-308, 2002.

[9] S. Poduri and G. S. Sukhatme, “Constrained coverage for mobile sensor networks,” in Proceedings of 2004 IEEE International Conference on Robotics and Automation, vol. 1, pp. 165-171, 2004.

[10] D. O. Popa, H. E. Stephanou, C. Helm, and A. C. Sanderson, “Robotic deployment of sensor networks using potential fields,” in Proceedings of the 2004 IEEE International Conference on Robotics and Automation, vol. 1, pp. 642-647, 2004.

[11] N. Heo and P.K. Varshney, “Energy-efficient deployment of intelligent mobile sensor networks,” *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans,* vol. 35, no. 1, pp. 78–92, 2005.

[12] M. L. Lam and Y. H. Liu, "Isogrid: an efficient algorithm for coverage enhancement in mobile sensor networks," in Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1458-1463, 2006.

[13] J. Tan, "A scalable graph model and coordination algorithms for multi-robot systems," in Proceedings of 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 1529-1534, 2005.

[14] K. Akkaya and S. Janapala, "Maximizing connected coverage via controlled actor relocation in wireless sensor and actor networks," *Computer Networks*, vol. 52, no. 14, pp. 2779–2796, 2008.

[15] R. S. Chang and S. H. Wang, "Self-deployment by density control in sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1745-1755, 2008.

[16] L. Parker, "Cooperative robotics for multi-target observation," *Intelligent Automation and Soft Computing*, vol. 5, no. 1, pp. 5-19, 1999.

[17] L. Parker, "Distributed algorithms for multi-robot observation of multiple moving targets," *Autonomous Robots*, vol. 12, no. 3, pp. 231-255, 2002.

[18] Z. Butler and D. Rus, "Event-based motion control for mobile sensor networks," *IEEE Pervasive Computing*, vol.2, no. 4, pp. 34-42, 2003.

[19] P. V. Sander, D. Peleshchuk, and B. J. Grosz, "A scalable, distributed algorithm for efficient task allocation," in Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1191-1198, 2002.

[20] J. H. Reif and H.Wang, "Social potential fields: a distributed behavioral control for autonomous robots," *Robotics and Autonomous Systems*, vol. 27, no. 3, pp. 171-194, 1999.

[21] L. Chaimowicz, N. Michael, and V. Kumar, "Controlling swarms of robots using interpolated implicit functions," in Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 2487-2492, 2005.

[22] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 1327-1332, 2004.

[23] J. Cortes, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESIAM: Control, Optimization and Calculus of Variations*, vol. 11, no. 4, pp. 691-719, 2005.

[24] J. Tan, N. Xi, W. Sheng, and J. Xiao, "Modeling multiple robot systems for area coverage and cooperation," in Proceedings of the 2004 IEEE International Conference on Robotics and Automation, 2004.

[25] M. Schwager, J. McLurkin, and D. Rus, “Distributed coverage control with sensory feedback for networked robots,” in Proceedings of Robotics: Science and Systems, vol. 3, pp. 2568-2573, 2006.

[26] M. Schwager, J. Slotine, and D. Rus, “Decentralized, adaptive control for coverage with networked robots,” in Proceedings of the 2007 IEEE International Conference on Robotics and Automation, pp. 3289-3294, 2007.

[27] M. Schwager, J. Slotine, and D. Rus, “Consensus learning for distributed coverage control,” in Proceedings of the 2008 IEEE International Conference on Robotics and Automation, pp. 1042-1048, 2008.

[28] M. Schwager, F. Bullo, D. Skelly, and D. Rus, “A ladybug exploration strategy for distributed adaptive coverage control,” in Proceedings of the 2008 IEEE International Conference on Robotics and Automation, pp. 2345-2353, 2008.

[29] S. Kataoka and S. Honiden, “Multi-robot positioning model: multi-agent approach,” in Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, pp. 112-119, 2006.

[30] Q. Jiang, “An improved algorithm for coordinated control of multi-agent system based on r-limited voronoi partitions,” in Proceedings of the 2006 IEEE International Conference on Automation Science and Engineering, pp. 667-671, 2006.

[31] S. Martinez and F. Bullo, “Optimal sensor placement and motion coordination for target tracking,” *Automatica*, vol. 42, no. 4, pp. 661-668, 2006.

[32] A. Winfield, “Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots,” *in Distributed Autonomous Robotic Systems 4*, L. E. Parker, G. Bekey, and J. Barhen, Eds. Springer-Verlag, 2000.

[33] M. R. Pac, A. M. Erkmen, and I. Erkmen, “Scalable self-deployment of mobile sensor networks: a fluid dynamics approach,” in Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1446-1451, 2006.

[34] W. Kerr, D. Spears, W. Spears, and D. Thayer, “Two formal gas models for multi-agent sweeping and obstacle avoidance,” *Lecture Notes in Artificial Intelligence*, vol. 3328, pp. 111-130, 2005.

[35] W. Kerr and D. Spears, “Robotic simulation of gases for a surveillance task,” in Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2905-2910, 2005.

[36] V. Isler, S. Khanna, and K. Daniilidis, “Sampling based sensor- network deployment,” in Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 1780-1785, 2004.

[37] G. Antonelli, F. Arrichiello, S. Chiaverini, and R. Setola, “A self-confguring manet for coverage area adaptation through kinematic control of a platoon of mobile robots,” in Proceedings of 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1332-1337, 2005.

[38] W. H. Fan, Y. H. Liu, F. Wang, and X. P. Cai, “Multi-robot formation control using potential field for mobile ad-hoc networks,” in Proceedings of the 2005 IEEE International Conference on Robotics and Biomimetics, pp. 133-138, 2005.

[39] B. S. Pimentel and M. F. M. Campos, “Cooperative communication in ad hoc networked mobile robots,” in Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2876-2881, 2003.

[40] M. Ji and M. Egerstedt, “Distributed coordination control of multiagent systems while preserving connectedness,” *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 693-703, 2007.

[41] M. N. Rooker and A. Birk, “Combining exploration and ad-hoc networking in RoboCup rescue,” *Lecture notes in computer science*, vol. 3276, pp. 236-246, 2005.

[42] M. N. Rooker and A. Birk, “Communicative Exploration with Robot Packs,” *Lecture notes in computer science*, vol. 4020/2006, pp. 267-278, 2006.

[43] M. N. Rooker and A. Birk, “Multi-robot exploration under the constraints of wireless networking,” *Control Engineering Practice*, vol. 15, pp. 435-445, 2007.

[44] J. Vazquez and C. Malcolm, “Distributed multirobot exploration maintaining a mobile network,” *Intelligent Systems*, vol. 3, pp. 113-118, 2004.

[45] M. Schwartz, *Mobile Wireless Communications,* Cambridge University Press, Cambridge, UK, 2005.

[46] IEEE std. 802.15.4, *Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area networks,* pp. 65-66, 2003.

[47] Texas Instruments, *CC2420 datasheet*, pp. 48-49, 2007.

[48] A. Vlavianos, L. K. Law, I. Broustis, S. V. Krishnamurthy, and M. Faloutsos, “Assessing link quality in IEEE 802.11 wireless networks: which is the right metric,” IEEE 9th International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 1-6, 2008.

[49] K. Srinivasan, and P. Levis, “RSSI is under appreciated,” in Proceedings of the Third Workshop on Embedded Networked Sensors, 2006.

[50] S. Farahani, *ZigBee Wireless Networks and Transceivers,* Elsevier, Burlington, MA, 2008.

[51] L. Tang, K. Wang, Y. Huang, and F. Gu, “Channel characterization and link quality assessment of IEEE 802.15.4-compliant radio for factory environments,” *IEEE Transactions on Industrial Informatics*, vol. 3, no. 2, pp. 99-110, 2007.

[52] A. R. Wagner and R. C. Arkin, "Internalized plans for communication-sensitive robot team behaviors," in Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2480–2487, 2003.

[53] M. Powers and T. Balch, "Value-Based Communication Preservation for mobile robots," *Distributed autonomous robotic systems 6 part VIII*, pp. 327-336, 2007.

[54] M. A. Hsieh, R. V. Kumar and C. J. Taylor, "Constructing radio signal strength maps with multiple robots," in Proceedings of the 2004 IEEE International Conference on Robotics and Automation, vol. 4, pp. 4184-4189, 2004.

[55] M. A. Hsieh, A. Cowley, V. Kumar and C. J. Taylor, "Towards the deployment of a mobile robot network with end to end performance guarantees," in Proceedings of the 2006 IEEE International Conference on Robotics and Automation, pp. 2085-2090, 2006.

[56] M. A. Hsieh, A. Cowley, V. Kumar and C. J. Taylor, "Maintaining network connectivity and performance in robot teams," *Journal of field robotics*, vol. 25, pp. 111-131, 2008.

[57] H. G. Nguyen, N. Pezeshkian, A. Gupta and N. Farrington, "Maintaining communication link for a robot in a hazardous environment," in Proceedings of the 10th International Conference on Robotics and Remote Systems for Hazardous Environments, 2004.

[58] K. M. Passino and S. Yurkovich, *Fuzzy Control*, Addison Wesley Longman, USA, 2010.

[59] S. N. Sivanandam, S. Sumathi and S. N. Deepa, *Introduction to Fuzzy Logic using Matlab*, Springer, 2007.

[60] T. J. Ross, *Fuzzy Logic with Engineering Applications, 2nd edition,* John Wiley & Sons, USA, 2004.

[61] M. Jahangir, S. Khosravi and H. Afkhami, "A robust-adaptive fuzzy coverage control for robotic swarms," *Nonlinear Dynamic*, vol. 69, no. 3, pp. 1191-1201, 2012.

[62] B. Ranjbar-Sahraei, F. Shabaninia, A. Nemati, and S.D. Stan, "A novel robust decentralized adaptive fuzzy control for swarm formation of multiagent systems," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 8, pp. 3124-3134, 2012.

[63] A. Bazoula, M. S. Djouadi and H. Maaref, "Formation control of multi-robots via fuzzy logic technique," *International Journal of Computers Communications & Control*, vol. 3, pp. 179-184, 2008.

[64] J. P. Canovas, K. LeBlanc and A. Saffiotti, "Robust multi-robot object localization fusing fuzzy logic," *Lecture Notes in Computer Sciences*, vol. 3276, pp. 247-261, 2005.

[65] T. J. Anastasio, *Tutorial on Neural Systems Modeling*, Massachusetts, USA: Sinauer Associates Inc, 2010.

[66] S. Haykin, *Neural Networks and Learning Machines,* Prentice Hall, New Jersey, USA, 2008.

[67] K. G. Jolly, R. Sreerama Kumar and R. Vijayakumar, “An artificial neural network based dynamic controller for a robot in a multi-agent system,” *Neurocomputing*, vol. 73, no. 1-3, pp. 283-294, 2009.

[68] K. G. Jolly, K. P. Ravindran, R. Vijayakumar and R. Sreerama Kumar, “Intelligent decision making in multi-agent robot soccer system through compounded artificial neural networks,” *Robotics and Autonomous System*, vol. 55, no. 6, pp. 589-596, 2007.

[69] K. G. Jolly, R. Sreerama Kumar and R. Vijayakumar, “Intelligent task planning and action selection of a mobile robot in a multi-agent system through a fuzzy neural network approach,” *Engineering Applications of Artificial Intelligence*, vol. 23, no. 6, pp. 923-933, 2010.

[70] M. Kanaya and M. Tanaka, “Path planning method for multi-robots using a cellular neural network,” *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 81, no. 3, pp. 10-21, 1998.

[71] M. Lee, “Evolution of behaviors in autonomous robot using artificial neural network and genetic algorithm,” *Information Sciences*, vol. 155, no. 1-2, pp. 43-60, 2003

[72] M. Quinn, L. Smith, G. Mayley and P. Husbands, “Evolving controllers for a homogeneous system of physical robots: structured cooperation with minimal sensors,” *The Royal Society*, vol. 361, no. 1811, pp. 2321-2343, 2003.

[73] R. S. Sutton and A. G. Barto, *Reinforcement Learning : An Introduction,* MIT Press, Massachusetts, USA, 1998.

[74] J. LI, Q. Pan and B. Hong, “A new approach of multi-robot cooperative pursuit based on association rule data mining,” *International Journal of Advanced Robotic Systems*, vol. 6, no. 4, pp. 329-336, 2009.

[75] Y. Wang and C. W. de Silva, “A machine-learning approach to multi-robot coordination,” *Engineering Applications of Artificial Intelligence*, vol. 21, no. 3, pp. 470-484, 2008.

[76] D. Gu and E. Yang, “Fuzzy policy reinforcement learning in cooperative multi-robot systems,” *Journal of Intelligent & Robotics Systems*, vol. 48, no. 1, pp. 7-22, 2007.

[77] D. Zhang, G. Xie, J. Yu and L. Wang, “Adaptive task assignment for multiple mobile robots via swarm intelligence approach,” *Robotics and Autonomous Systems*, vol. 55, no. 7, pp. 572-588, 2007.

[78] O. Azouaoui, A. Cherifi, R. Bensalem, A. Farah and K. Achour, “Reinforcement learning-based group navigation approach for multiple autonomous robotic systems,” *Advanced Robotics*, vol. 20, no. 5, pp. 419-442, 2006.

[79] T. Yasuda, K. Ohkura and K. Ueda, "A homogenous mobile robot team that is fault-tolerant," *Advanced Engineering Informatics*, vol. 20, pp. 301-311, 2006.

[80] M. Poramate, K. Christoph and W. Florentin, "Combining correlation-based and reward-based learning in neural control for policy improvement," *Advances in Complex Systems*, vo. 16, no. 2-3, 2013.

[81] Z. Miljkovic, M. Mitic and M. Lazarevic, "Neural network reinforcement learning for visual control of robot manipulators," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1721-1736, 2013.

[82] S.F. Desouky and H.M. Schwartz, "Self-learning fuzzy logic controllers for pursuit-evasion differential games," *Robotics and Autonomous Systems*, vol. 59, no. 1, pp. 22-33, 2011.

[83] J. Borenstein, H. R. Everett, L. Feng and D. Wehe, "Mobile robot positioning - sensors and techniques," *Jounrany of Robotic Systems*, vol. 14, no. 4, pp. 231-249, 1997.

[84] J. Paradiso, J. Gips, M. Laibowitz, S. Sadi, D. Merrill, R. Aylward, P. Maes and A. Pentland, "Identifying and facilitating social interaction with a wearable wireless sensor network," *Personal and Ubiquitous Computing*, vol. 14, pp. 137-152, 2010.

[85] M. Hazas and A. Hopper, "Broadband ultrasonic location systems for improved indoor positioning," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 536-547, 2006.

[86] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in Proceedings of the 19th IEEE Conference of the Computer and Communications Societies, vol. 2, pp. 775-784. 2000.

[87] L. Dai, Z. Wang, C. Pan and S. Chen "Wireless positioning using TDS-OFDM signals in single-frequency networks," *IEEE Transactions on Broadcasting*, vol. 58, no. 2, pp. 236-246, 2012.

[88] A. Kangas and T. Wigren, "Angle of arrival localization in {LTE} using {MIMO} pre-coder index feedback," *IEEE Communications Letters*, vol. 17, no. 8, pp. 1584-1587, 2013.

[89] R. Parker and S. Valaee, "Vehicular node localization using received-signal-strength indicator," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 6, pp. 3371-3380, 2007.

[90] D. Orlando and G. Ricci, "Adaptive radar detection and localization of a point-like target," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4086-4096, 2011.

[91] Y. Pailhas. C. Capus and K. Brown, "Dolphin-inspired sonar system and its performance," *IET Radar, Sonar& Navigation*, vol. 6, no. 8, pp. 753-763, 2012.

[92] N. S. Zhou, H. Feng and R. Yuan, "Error compensation for cricket indoor location system," in Proceedings of the 2009 International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 390-395, 2009.

[93] D. Meyer-Delius, M. Beinhofer, A. Kleiner and W. Burgard, “Using artificial landmarks to reduce the ambiguity in the environment of a mobile robot,” in Proceedings of the 2011 IEEE International Conference on Robotics and Automation, pp. 5173-5178, 2011.

[94] F. Wen, K. Yuan, W. Zhou, X. Chai and R. Zheng, “Visual navigation of an indoor mobile robot based on a novel artificial landmark system,” in Proceedings of the 2009 International Conference on Mechatronics and Automation, pp. 3775-3780, 2009.

[95] H. Li and S. X. Yang, “A behavior-based mobile robot with a visual landmark-recognition system,” *IEEE/ASME Transactions on Mechatronics*, vol. 8, no. 3, pp. 390-400, 2003.

[96] T. Kim and J. Lyou, “Indoor navigation of skid steering mobile robot using ceiling landmarks,” in Proceedings of the 2009 IEEE International Symposium on Industrial Electronics, pp.1743-1748, 2009.

[97] N. Greggio, A. Bernardino, C. Laschi, J. Santos-Victor and P. Dario, “Real-time 3D stereo tracking and localizing of spherical objects with the icub robotic platform,” *Journal of Intelligent and Robotic Systems*, vol. 63 no. 3-4, pp. 417-446, 2011.

[98] B. Choi, J. Lee, J. Lee and K. Park, “A hierarchical algorithm for indoor mobile robot localization using RFID sensor fusion,” *IEEE Transactions on Industrial Electronic*, vol. 58, no. 6, pp. 2226-2235, 2011.

[99] T. Fukuda, S. Ito, F. Arai, Y. Yokoyama, Y. Abe, K. Tanaka and Y. Tanaka, “Navigation system based on ceiling landmark recognition for autonomous mobile robot-landmark detection based on fuzzy template matching (FTM),” in Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 150-155, 1995.

[100] W. Liu and Y. Zhou, “Recovering the position and orientation of a mobile robot from a single image of identified landmarks,” in Proceeding of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems , pp. 1065-1070, 2007.

[101] Y. Lee, B. Yim and J. Song, “Mobile robot localization based on effective combination of vision and range sensors,” *International Journal of Control, Automation, and Systems,* vol. 7, pp. 97-104, 2009.

[102] I. Shimshoni, “On mobile robot localization from landmark bearings,” *IEEE Transactions on Robotics and Automation,* vol. 3, no. 18, pp. 971-976, 2001.

[103] N. M. Kwok, Q. P. Ha, S. Huang, G. Dissanayake and G. Fang, “Mobile robot localization and mapping using a gaussian sum filter,” *International Journal of Control, Automation, and System,* vol. 3, no. 5, pp. 251-268, 2007.

[104] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities,* Morgan Kaufmann, San Francisco, 2005.

[105] http://www.vision.caltech.edu/bouguetj/calib_doc

[106] J. Canny, “A Computational Approach To Edge Detection,” *IEEE Transction Pattern Analysis and Machine Intelligence,* vol. 8, no. 6, p. 679–698, 1986.

[107] M. I. Shamos and F. Preparata, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.

[108] P. J. Schneider and D. H. Eberly, *Geometric Tools for Computer Graphics*, Morgan Kaufmann, San Francisco, 2003.

[109] Q. Li and D. Rus, “Navigation protocols in sensor networks,” *ACM Transactions on Sensor Network*, vol. 1, no. 1, pp. 3-35, 2005.

[110] P. Liu and H. Li, *Fuzzy Neural Network Theory and Applicaion,* World Scientific, NJ, USA, 2010.