A POTENTIAL FIELD APPROACH TO MULTIPLE ROBOT FORMATION CONTROL

by

ROHIT SANTOSH TALATI

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2007

To my family for all their love and support.

ACKNOWLEDGEMENTS

I would like to acknowledge the following people who have helped me make this thesis possible. In no particular order:

My supervising professor, Dr. Dan O. Popa for his valuable guidance and constant motivation. I am grateful for his deep involvement in my thesis work and constant demand for the best.

My supervising committee members Dr. Harry Stephanou and Dr Frank Lewis for gracefully agreeing to contribute their valuable time, advice and support at ARRI.

My family whom I could ask of nothing more - Mr. Santosh G. Talathi ('Baba'), Mrs. Sushama S. Talathi ('Mama') and brother Vinod S. Talati for the unending, selfless motivation and support.

All the ARRI-DIAL members, employees and colleagues for making me feel comfortable and sharing their knowledge and experience during my work on this thesis.

My friends for their patience with my random sleeping patterns and usually unshaven face.

July 23, 2007

ABSTRACT

A POTENTIAL FIELD APPROACH TO MULTIPLE ROBOT FORMATION CONTROL

Publication No. _____

ROHIT SANTOSH TALATI

The University of Texas at Arlington, 2007

Supervising Professor: Dan Popa

A special case of cooperative control for mobile robots is considered - formation control. A potential field based algorithm is developed in which geometric, communication and information centric influences are considered and allowed to deform the formation. Control is accomplished in a leader-follower(s) method. One leader robot defines the overall trajectory and follower nodes individually and autonomously maintain the formation while simultaneously moving toward a goal position. The particular nodes considered are the ARRIbots developed at the Distributed and Intelligence and Autonomy Lab (DIAL) in the Automation and Robotic and Research Institute (ARRI), which are non-holonomic differential-drive wheeled robots. In order to facilitate the above, a kinematic model for the ARRIbot mobile robot that accounts for its non-holonomic constraint and particular inputs is presented. Based on this model, a trajectory tracking controller (LQR based) is detailed assuming a given reference trajectory. Finally additional artificial forces are added to account for additional constraints on the optimal node paths and positions. Obstacle avoidance is added to the formation by repulsive forces and an artificial communications force is added in order to optimize the wireless communications channel between nodes. These results are validated using computer simulations and experiments with the ARRIbots on our mobile robot platform.

It was found that the potential field algorithm was successful in maintaining the required node formation. The position error for follower nodes was found to decay as required. In addition, a simulation of mine-field detection scenario was shown to successfully combine the three formation influences. The LQR trajectory tracker was found to satisfy the requirements of the formation control algorithm. The state estimate error was low for both straight line and turning motions while the tracking error was low for straight line and high for turning motions. This was attributed to the short duration of the latter and amplification of calculation and processing delay errors. The accuracy of the state estimate errors was shown to be useful in reducing the tracking error over multiple trajectories.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Mobile robots can be used in a variety of situations. Applications include sensing and performing tasks where it is too dangerous, expensive or simply impossible to send humans. Some practical uses could be mine-field detection, toxic gas sensing, unmanned vehicle missions, sensor nets, satellite synchronization and more. All of the above require many separate aspects of mobile robotics - mechanical hardware design, communications, path planning, trajectory tracking, sensing, information processing and much more. In addition, multiple robots can work together to achieve a common goal - this is called cooperative control. A troop of robots like this can reduce the time required to complete a task as they collectively have more processing power, can sample a greater physical area, share resources, and perform tasks in parallel. With the falling cost of hardware, it is desirable to have a large number of inexpensive 'nodes' that are capable of cooperative control.

The following sections will include a literature review of relevant previous work done in the areas of trajectory tracking and formation control, followed by a mechanical and mathematical model of the ARRIbot. Then the method of LQR trajectory tracking and Potential Field formation control are discussed along with the specific ARRIbot implementation details. Finally, simulation and experimental results for our methods shown.

1.1 Problem Statement

A cooperative system is a collection of nodes which communicate and cooperate to perform a common objective or goal. This cooperation may be in a centralized manner or in a distributed manner. In this thesis, a special case of cooperative control is considered - formation control. It is desired to have a formation of mobile nodes that can be

commanded to move as a group. The nodes must individually and autonomously (thus distributed) decide the micro-movements required to maintain the formation while simultaneously moving toward a goal position. A few specific applications where formation control is required include:

- Terrain Mapping
- Reconnaissance and Search & Rescue
- Surveillance
- Environment Monitoring (Sampling and Sensing)
- Synchronization (satellite, vehicles)
- Vehicle Platoons

Formation shape and position of nodes can depend on many factors. In this thesis, we focus on three influences:

- Geometric.

  This is an artificial constraint on formation shape imposed by a higher level mission strategy. The particular shape of the formation and the orientation of the robots within is chosen to optimize the mission goal. For example, mine-field detection would employ a horizontal chain formation, agricultural harvesting (raster scanning) would employ an equidistant formation, satellite synchronization would require a mesh-like formation. Applications are also possible for combinations of land, sea and air mobile robots. An upside down pyramid formation could be used with four Unmanned Aerial Vehicles (UAV) and one ground base vehicle. The UAVs would perform scouting and inform the ground vehicle of upcoming targets.

- Communications centric.

  It is desirable for mobile nodes to remain in communications range of the other nodes in order to allow exchange of data. Since the capacity of a wireless channel is dependant upon the distance between nodes, the communications centric control will accomplish optimal placement of the nodes so that the data rate of the wireless communications channel is maximized. In addition, this control should strive to keep the network connected while moving. A network with better connectivity

between nodes will have less communication delay and less energy consumption due to a reduction in lost wireless packets and number of hops. This type of formation control can be applied in any Wireless Sensor Network(WSN) [9] where multiple mobile robots are required to communicate.

- Information Centric.

  This stems from the need to alter the formation geometry and position based on an external algorithm. An example of an information gathering algorithm is adaptive sampling of a spatio-temporal field. The aim is to regenerate the field and represent it using parameters. The sampling locations are chosen to maximally reduce the uncertainty in the estimate of these parameters [10]. These sampling points are updated every iteration and the robots must change formation to reach these points.

The particular nodes considered are the ARRIbots, which are non-holonomic differential-drive wheeled robots. The nodes also require some method of following a specified trajectory in order to fulfil the requirements of the formation control strategy.

1.2 Contributions

Research work at ARRI includes mobile robot localization, monitoring, formation control and navigation. This thesis continues and extends the research work in [8], [11],[2] and [10]. During the course of research, the author participated in:

- ARRIbot localization without the use of pervasive and persistent fixed nodes. The aim was to self localize nodes with respect to each other after an initial localization of a few nodes. Movement of nodes was restricted so that enough nodes were available to localize them afterward.
- Implementation of LQR Trajectory Tracking for the ARRIbots.
- Simulations of formation control using potential fields (Geometric).
- Addition of obstacle avoidance to formation control algorithm.
- Implementation of a Leader-Follower chain formation with two ARRIBots.
- Simulation of a Mine Field detection process. This involved the use of geometric and communication centric formation control.

Linear Quadratic Regulator(LQR) trajectory tracking and Potential field formation control have already been demonstrated in the field as described in [11] and [12]. In [8], a communications centric approach is taken to node placement. This research seeks to combine all three approaches - geometric, communication and information - into a single resultant formation. As can be expected, the geometric (ideal) formation will be deformed due to the influence of the communication and information factors.

The main contribution of this thesis is the formation control algorithm by the use of potential fields. The algorithm employs leader-follower(s) resulting in a control law that allows a command station to specify a formation shape (via geometry), the position of each node in the formation, and the trajectory the group must describe. A virtual leader in general provides the overall objective to the formation.

Ideally, the formation control is decentralized in the sense that the base station does not provide the path for any follower robot. Each individual robot plans it's own path using relative distance and angle measurements. These measurements should be obtained using sensors local to the robot - for example ultrasound rangers (as in the 1 Dimensional implementation case), radar or laser range finders could be used. For our 2 Dimensional implementation, however, these measurements are centralized - the base station is equipped with an overhead IR camera and is capable of absolute global localization of all the robots. The individual robots query the base station for the locations it requires in order to plan its path. The LQR trajectory tracking controller is decentralized and relies on the dead-reckoning to track the desired trajectory.

Simulations were performed in MATLAB. These included adding separately the geometric, communications and information forces along with an idealized model of the ARRIbots. Obstacles such as walls and other robots were added to demonstrate the obstacle avoidance. The communication and sensing delay was incorporated using an appropriate sampling time. A test-bed composed of an overhead infrared camera, projector and base-station was created in order to test the formations. The ARRIbots were programmed and the algorithm was distributed to the the individual nodes with communication with the base station necessary only for sensing other robot poses.

In order to facilitate formation control in practice, we also present a kinematic model for the ARRIbots that accounts for its non-holonomic constraint, linear and angular speeds, and wheel speeds as control inputs. Based on this model, we also design a trajectory tracking controller (LQR based) assuming a given reference trajectory. The validity of the approach is illustrated via simulations performed in MATLAB. Various cases such as straight-line, turn and arbitrary smooth paths were chosen. Experimental tests were performed at the ARRI Dial Lab using the ARRIbots. The controller was implemented using the Javelin stamp micro-controller and a projector/infrared sensing test-bed was used to capture and present results.

In order to test our formation algorithm we propose a general experimentation structure, which can be used in several different scenarios. This general structure addresses some important issues about the communication effects over a networked system, the information flow between nodes such that data can be easily collected and analyzed, and the programming style that can influence the tests results.

The flexibility of potential field based control is demonstrated by additional artificial forces. Obstacle avoidance is added to the formation by repulsive forces. This enables the formation to deform as required in the presence of obstacles and re-group when the obstacles have been passed. Another factor influencing the optimal position of a node is the wireless communications channel. A simplified potential is derived based on the a channel capacity model and an artificial communications force is added in order to improve communications between nodes.

Finally an advanced simulation was performed to simulate a mine field detection scenario. 20 nodes in a horizontal chain formation sweep and sample a mine field and relay the results back to a (far) base station. Several intermediate 'hop' nodes are placed in between the base and the field. These 'hop' nodes continuously change their formation to optimize the use of the communications channel.

1.3 Thesis Organization

The thesis is organized as follows. In chapter 2, first relevant previous work in the fields of trajectory tracking and formation control are discussed. Next, we describe the hardware configuration of the ARRIbot, and propose it's kinematic model. Also, the physical parameters and the method of measurement of the ARRIbots are given along with a state estimator to measure the position and orientation.

In chapter 3 a trajectory tracking control algorithm is proposed along with implementation details for the ARRIbot. An algorithm for formation control using potential fields is developed using a leader-follower approach. One of the nodes is defined as a leader and the others as followers. The geometry of the formation is known apriori, and the followers use a potential field to converge to the expected formation position. The advantage of potential fields is that other path-influencing factors such as obstacle-avoidance and communication constraints can easily be factored into the overall path. This control law allows us to determine the formation shape, the position of each node within the formation, and the trajectory that the group can describe.

Chapter 4 presents MATLAB simulation results for the LQR trajectory tracking controller and the formation control. The effects of varying the controller parameters for the LQR controller are presented.

Chapter 5 presents experimental results for the LQR trajectory tracking controller and the formation control using the ARRIbots available at the ARRI Dial Lab. Comparisons are made between uncontrolled and controlled motions.

Finally Chapter 6, will outline conclusions and proposes possible future work.

1.4 Summary of Results

It was found that a potential field approach to path-planning is suited to the problem of formation control. It can incorporate the node position influencing factors through the use of artificial forces. Geometric formation was accomplished with an attractive force toward the ideal formation position, obstacle avoidance was accomplished

using repulsive forces and communications constraints were satisfied using artificial forces derived from maximizing a utility function with respect to the network and capacity.

The advantages of the potential field approach include:

- Combine multiple objectives into one field
- Parabolic Potentials have easy derivatives, thus implementation is easy.

The disadvantages of potential fields are:

- Tuning of weights is required to get best behavior
- Arbitrary Potential functions may have difficult or non-analytic derivatives.
- Local minima can trap robots, thus necessitating higher level heuristics.

LQR trajectory tracking was found to satisfy the requirements of the formation control algorithm. The tracking error ($desired - actual$) error for the straight line motion was low, however the tracking error for the axis turn motions were high. This was due to the errors accumulating from processing time and fixed point calculation errors. The small duration of the turning motion tended to amplify the errors, thus only the turning case was affected. The state estimate error ($actual - estimated$) was very low in both cases. Thus the dead-reckoning localization was accurate and could be used to correct the tracking error.

LQR trajectory tracking was found to be advantageous in many ways:

- Calibration of motors is un-necessary.
- There are no PID gains that need to be tuned.
- The final controller is simply a P-controller with time varying gains resulting in less storage required.
- An estimate of the robot state ($[x, y, \varphi]^T$) is always available as a direct result of LQR. This is very useful to higher level algorithms.
- Any smooth trajectory can be tracked given enough processing power and storage.
- Special Cases such as 'Straight Line' and 'Axis Turn' have constant LQR gains resulting in great simplification.

The disadvantages of LQR include:

- LQR gains are trajectory dependant.

- Q and R matrices need to be tuned.
- Needs a very accurate timer.
- A fast processor is required
- Storage of pre-calculated gains takes up a lot of memory ($3x3xN$, where N is the number of steps).

While testing our formation algorithm important general issues that affect mobile robots came to light such as:

- Communication effects over a networked system. Lost packets during wireless transmission are a major concern and a scheme for guaranteed message delivery had to be used.
- Embedded micro-controller programming. Smaller scale processors typically include only a subset of the features of general purpose processers (floating point math, available memory, interrupts, programming libraries). These feature had to be worked around by implementing them using lower lever instructions.
- Hardware reliability - some parts were more prone to failure than others.
- Hardware compatibility - some sensors and actuators are designed with a bus system design (for example I^2C) instead of direct pin wiring via a multiplexer as in the ARRIbots.

Much additional research is possible and required in the field of formation control. A through study of communication delay effects on the formation is one area that could be expanded upon. The stability of the formation (string, mesh) also needs to be analyzed along with addition of addition of robustness to modeling and localization errors. Finally, a higher level control needs to be developed in order to define the leader trajectories (mission control, task/research allocation) and formation geometry. An interface to easily specify these would also enable further testing and possible commercial deployment.

CHAPTER 2

LITERATURE REVIEW AND BACKGROUND

2.1 Wireless Sensor Network

A Wireless Sensor Network (WSN) is a collection of nodes that have both sensing and wireless communication ability. They are spatially distributed and autonomous and have the task of cooperatively monitoring environmental conditions [2]. Typical applications of a WSN are in monitoring and surveillance - for example habitat monitoring, acoustic detection, seismic detection, military surveillance, inventory tracking, medical monitoring, smart spaces, process monitoring and more. A wireless sensor network can be made up of both static and mobile nodes. The presence of mobile nodes is advantageous in scenarios such as adaptive exploration and search and rescue missions where the network topology is constantly changing [13] [9].

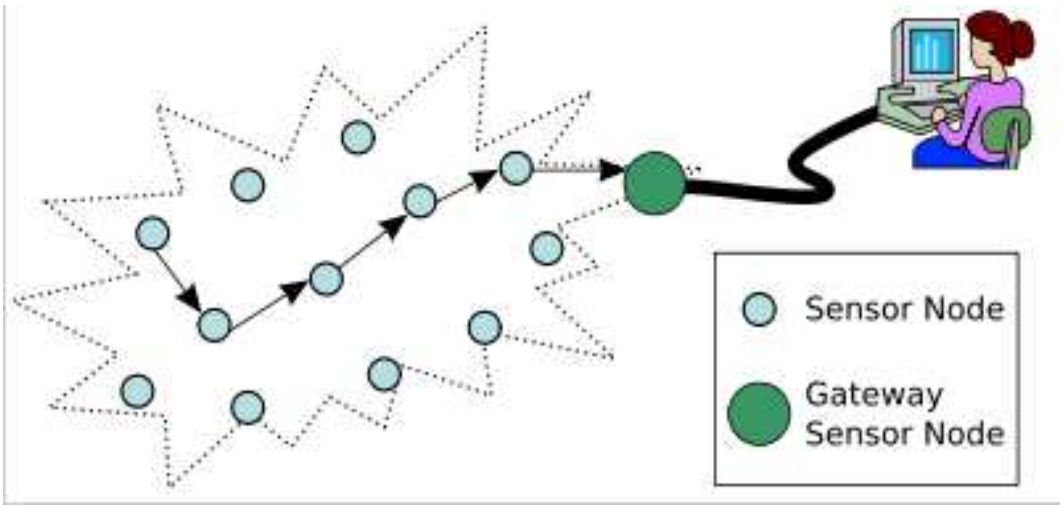

Figure 2.1. Wireless Sensor Network [1].

2.2 ARRIBot

The ARRIbot is a low cost mobile wireless sensor node that was developed in the Distributed and Intelligence and Autonomy Lab (DIAL) at the Automation and Robotic and Research Institute (ARRI). It is meant to form, along with other mobile as well as static nodes, part of an indoor Wireless Sensor Network. The objectives were to have have mobility and sensing and localization functionalities while keeping the cost low. For this research, the ARRIBot was used as designed by [10] [2] and [13] with minor modifications.

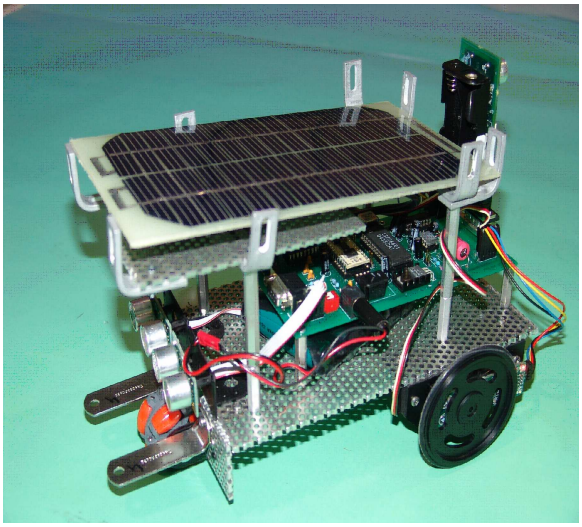

Figure 2.2. The ARRIBot [2].

The hardware on the ARRIbot includes [2]:

- Cricket, wireless module
- Solar panel
- Electronic board
- 2x Ultrasonic, ranging Module
- Optical Wheel Encoders
- Color Sensor

The features of the ARRIbot include:

- Compact electrical design achieved by the use of multilayer PCB layout and on-board micro-controller circuits
- High Precision maneuverability of resolution less than 2mm achieved by continuous servo motors and optical encoders
- Precision control over speed and direction achieved using Pulse Width Modulation (PWM) controlled servo motors
- Onboard sensor package comprising of an inbuilt color sensor and cricket sensor board
- Wireless communication protocol incorporated using cricket as transceiver
- Collision avoidance module implemented using ultra sonic range finders
- Continuous power level monitoring and range updating
- Alternate source of energy such as solar cells for charging the on-board battery.
- Light weight, low cost and compact design ideal for safe, large scale deployment

2.2.1 Modelling the ARRIBot

The Arribots were modeled as a differential drive system with state:

$$\vec{x} = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} \tag{2.1}$$

where x is the x-coordinate, y is the y-coordinate and φ is the orientation of the robot in a reference frame T as shown in Figure 2.3.

From the geometry of the problem,

$$\begin{aligned} x' &= -v \sin \varphi \\ y' &= -v \cos \varphi \\ \varphi' &= \omega \end{aligned} \tag{2.2}$$

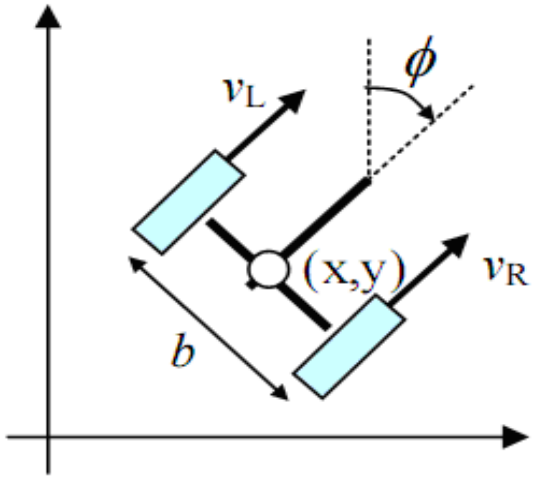

Figure 2.3. ARRIBot Model [3].

Where v is the vehicle linear velocity and ω is the vehicle angular velocity. 2.2 can be arranged to isolate the inputs v and ω:

$$\begin{bmatrix} x' \\ y' \\ \varphi' \end{bmatrix} = \begin{bmatrix} -\sin\varphi \\ \cos\varphi \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{2.3}$$

$$= \begin{bmatrix} -\sin\varphi & 0 \\ \cos\varphi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{2.4}$$

From the equations of motion of a wheel:

$$v = \frac{v_R + v_L}{2} = \frac{\omega_R r_R + \omega_L r_L}{2} \tag{2.5}$$

and

$$\omega = \frac{v_R - v_L}{l} = \frac{\omega_R r_R - \omega_L r_L}{l} \tag{2.6}$$

Substituting the above into the model gives:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \frac{r_R}{2}\cos\phi & \frac{r_L}{2}\cos\phi \\ \frac{r_R}{2}\sin\phi & \frac{r_L}{2}\sin\phi \\ -\frac{r_R}{l} & \frac{r_L}{l} \end{bmatrix} \bullet \begin{bmatrix} u_R \\ u_L \end{bmatrix} \tag{2.7}$$

A discrete version of Equation 2.7 is used for our case [2]:

$$\begin{aligned} x_{k+1} &= x_k + \frac{K_{drvR} r_R \Delta\theta_R + K_{drvL} r_L \Delta\theta_L}{2} \cos(\varphi_k) \\ y_{k+1} &= y_k + \frac{K_{drvR} r_R \Delta\theta_R + K_{drvL} r_L \Delta\theta_L}{2} \sin(\varphi_k) \\ \varphi_{k+1} &= \varphi_k + \frac{K_{turnR} r_R \Delta\theta_R - K_{turnL} r_L \Delta\theta_L}{2l} \end{aligned} \tag{2.8}$$

Where K_{drv} is defined the reciprocal of the ratio of the encoder counts to degrees. i.e.

K_{drv} = revolutions/encoder count and $\Delta\theta$ represents the change in encoder counts right and left respectively.

2.2.2 Cricket Wireless Module

The cricket (v2) is a combination device that serves as the wireless link in a wireless sensor network. In addition to operating as a wireless data communications device through Radio Frequency(RF), it includes localization capability using Ultrasound signals in combination with RF[14]. The radio operates at a frequency of 433 MHz, with a range of about 30 meters indoors in the absence of obstacles. The maximum ultrasound range is 10.5 meters when two crickets are facing each other and are in direct line of sight. Mobile or static crickets localize by transmitting both a RF and ultrasound pulse with a defined time interval in between. The receiver obtains a distance estimate for the corresponding sender beacon by using the known difference in propagation speeds between RF (speed of light) and ultrasound (speed of sound). This localization capability is accurate to an error of 1cm to 3cm [2]. The cricket is interfaced to the ARRIbot using the available serial port (RS 232).

2.3 Trajectory Tracking

The subject of trajectory tracking was pursued as it is an essential part of the formation control. Trajectory tracking is the ability of a mobile robot to move along a desired path $\vec{x}_{desired}(k)$ which has been generated from a path planing stage (the formation control algorithm in this case). The performance of the trajectory tracking can be measured using the tracking error $\vec{x}_{desired}(k) - \vec{x}_{actual}(k)$ where $\vec{x}_{actual}(k)$ is the actual

Figure 2.4. Crossbow Cricket Module.

state (position) at time k. Open loop control of robot motors usually does not yield good results in practice due to un-calibrated motors, unbalanced left and right motor characteristics, inaccuracies in wheel radii and axle length. State feedback is required to minimize the tracking error and can be provided using wheel odometry sensors. Trajectory tracking for mobile robots is an area of research in its own right and has been extensively studied [15].

2.3.1 Available Approaches

Since our model of the ARRIbot is kinematic, this review of previous approaches does not include those that require a dynamic model. A kinematic model takes into account the linear and angular velocities of the robot as control inputs [16]. Although this approach simplifies the discussion, it will not be accurate in cases where there is friction, acceleration and wheel-slip. However, several results validate the use of a kinematic model [16].

The particular model of the ARRIbot is also one that is non-holonomic. A non-holonomic system is one that has a restricted space of achievable velocities but an unrestricted space of achievable configurations [17]. In case of the ARRIbot, only movement along the heading direction is possible along with turning motions. Lateral movements are not possible. These restrictions however do not impede the robot from achieving any particular state, only the path to that state will differ.

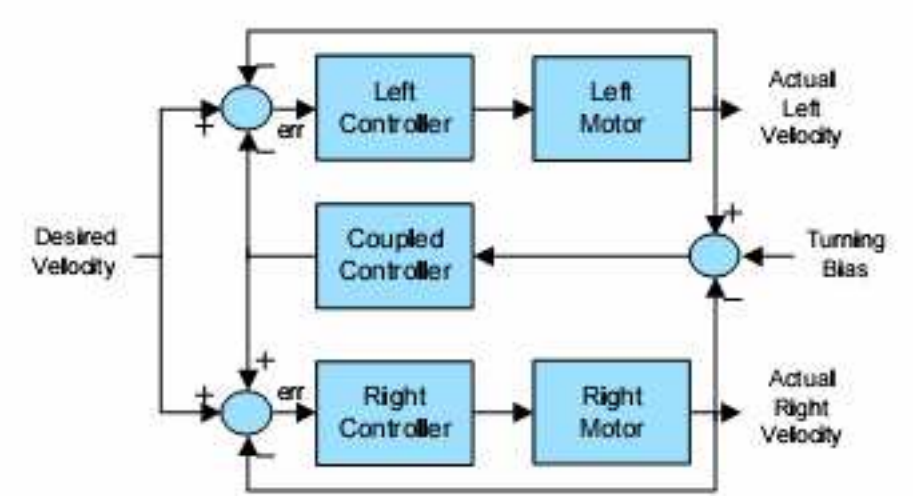

Figure 2.5. PID controller for Trajectory Tracking [4].

A popular approach for trajectory tracking is PID control [4]. Optical encoder data was used along with two PID controllers for feedback to control the speed of each motor - this ensured straight line movement. Trajectory tracking was accomplished by a third PID controller which controls the commutative centerline offset. Figure 2.5 shows a block diagram of the PID trajectory tracking algorithm.

Non-linear control theory is used in [18] for tracking an articulated vehicle. It allows following arbitrary paths. The model is linearized about the trajectory and a linear controller is designed for the linearized system. [19] investigates tracking for non-holonomic chains of order one. Drift-less systems are considered, with two inputs and a discontinuous switching feedback law is used.

The regulation and tracking problems of a unicycle mobile robot (kinematic model) are tackled in [20]. A feasible reference trajectory is assumed and a controller is designed

using the linear and angular speeds as control inputs. Various simulations and experimental results are presented in the article to demonstrate controller performance. In addition, [10] presents a controller that considers bounded kinematic model disturbances. Again, simulation results are presented to demonstrate controller performance. Stability of a unicycle robot is considered in [21] using the global exponential stability (GES) of the tracking error. GES in this case is assured by using a persistently existing input in the angular velocity. Thus the controller was shown to be suitable for any kind of trajectory except for a straight line.

Dynamic-feedback linearization is another method of trajectory tracking. In [22], the authors develop a algorithm via dynamic-feedback linearization along with a standard proportional-derivative (PD) controller. The result is then extended for the point stabilization problem. Implementation of this algorithm was accomplished and the performance was compared to other existing controllers.

In [23] and [24] a time varying coordinate transformation is introduced and smooth time-varying controllers are designed using a cascaded systems approach. [24] develops both full and reduced order observers along with an observer based controller. The tracking error of the resulting system is shown to be K-exponential convergent.

2.4 Formation Control

Formation control involves having a group of mobile nodes that maintain a particular spatial configuration. The nodes should (in a decentralized manner) perform the path-planning required to maintain the formation. The performance of a formation control algorithm can be measured by the total position error $\sum_{robots} (\vec{x}_{expected} - \vec{x}_{actual})$. The stability of the formation can be defined in terms of string stability [25]. For a system to be string stable, the position error must decrease along the formation away from the leader. For example in a chain formation,

$$(||x_{expected} - x_{actual}||)_i < (||x_{expected} - x_{actual}||)_j \text{ for} i > j$$

Where i and j are the ordered spatial positions of the nodes. The algorithm is also expected to allow for movement of the entire group as a single entity.

2.4.1 Available Approaches

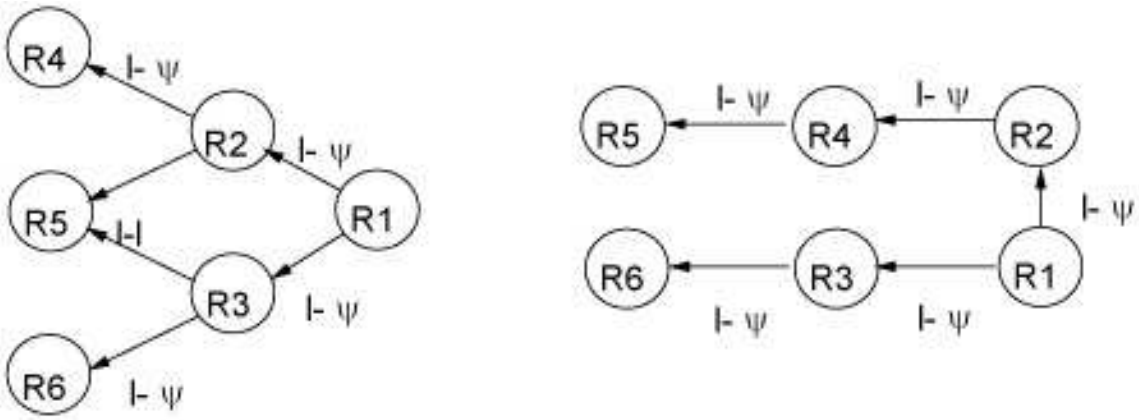

Figure 2.6. Leader-Follower formations [5].

One approach uses leader-follower patterns [12] in formation control. The sensing of other robots is decentralized in that only local sensor-based information is available for each robot. Two types of feedback controllers for maintaining formations of multiple robots were discussed - an $l-\psi$ controller (Figure 2.6) and an $l-l$ controller. The $l-\psi$ controller strives to maintain a desired length l and a desired angle ψ between the leader and the follower. This is done using input/output feedback linearizaiton. The $l-l$ control similarly tries to control the distance l_1 and l_2 from one follower robot to two leader robots.

In [26], a controller based on generalized coordinates was presented. Generalized coordinates are parameters that characterize the vehicles location(L), orientation(O) and its shape(S) with respect to a reference point within the formation. The trajectories of the group can then be specified in terms of L, O and S coordinates. A feedback control law was developed for asymptotic tracking of such trajectories while maintaining a desired formation geometry. A similar idea was described in [27] and [28] where the geometry of the formation is specified in shape coordinates.

A virtual structure method was used in [29] where the controller strives to force a group of robots to behave in a rigid formation. The controller is derived in three steps - defining the desired dynamics of the virtual structure, translation of the 'macro' desired motion of the virtual structure into desired motions for each robot and finally tracking controllers for each node are employed. This method is combined with a leader-follower method along with a behavioral approach for formation control of spacecraft. A similar method was applied in [30] for spacecraft formation control.

Figure 2.7. Flocking (V shaped formation) and Swarming of birds [6].

Another important approach to multi-agent coordinated navigation is based on the imitation of animal flocking and swarming (Figure 2.7). Much of this work is based on Reynolds result [31], where he develops three basic rules - separation, alignment and cohesion for simulation of flocking behavior. In [32] the authors develop a decentralized control law that emulates the three flocking rules. The algorithm is also designed to navigate through obstacles using split, rejoin and squeezing maneuvers, maintaining the group together. Simulation results are presented for a flock of 100 agents. Swarming is a special case of flocking where alignment is not considered.

A Potential field path-planning based algorithm was described in [33] and [34]. Zones of influence were defined around the nodes such as 'Sensing Zone', 'Safety Zone' and 'Influence Communciation Zone'. These zone parameters are used to define artificial

forces on the node in order to maintain sensing, obstacle avoidance and communications range respectively. An artificial attractive potential was defined using quadratic functions of the euclidian distance between the robot and it's desired position. Obstacle avoidance was also defined using a similar repulsive potential. At every iteration, the gradient of the corresponding field provided the driving force of the robot. A stability analysis was also provided in [34] in an obstacle free environment by looking at the conditions for force equilibrium.

2.5 Communication Restraints

2.5.1 Network Flow Model

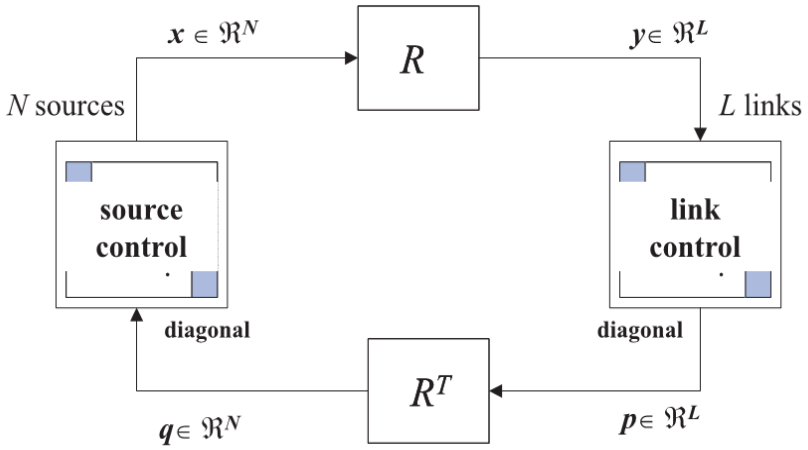

Figure 2.8. Network Flow Control Model [7].

The wireless network is defined as a set of sources and L links as shown in 2.8. Each link has a finite capacity c_l in packets per second and is shared by N sources. Each source i can use a subset $L_i \subseteq L$ of links. Thus a routing matrix $(L \times N)$ can be defined [35] [7]:

$$R_{li} = \left\{ \begin{array}{ll} 1 & \text{if} l \in L_i \\ 0 & otherwise \end{array} \right\} \tag{2.9}$$

There are many utility functions that can be chosen to represent different network models: such as $\arctan(x_i)$ for TCP Reno and $\log(x_i)$ for TCP Vegas[35].

2.5.2 Capacity

The Capacity of a communications link depends on the distance between the two nodes as is consistent with Shannon's theorm [10] which gives the maximum theoretical rate at which error-free bits can be transmitted.

$$C = W \log_2(1 + \frac{S}{N}) \tag{2.10}$$

where C is the channel capacity in bits per second, W is the signal bandwidth in Hertz and $\frac{S}{N}$ is the power signal to noise ratio. Alternatively [8],

$$C = W \log_2(1 + \frac{K' P_t}{W N_0 d^\alpha}) \tag{2.11}$$

where P_t is the power, d the distance between nodes, α the path loss coefficient, F the fading margin and K' the propagation constant. This equation shows that as the distance increases the capicity decreases. However, for our purposes, since this equation is invalid for $d = 0$, we use a modified version [8]:

$$c(d) = \left\{ \begin{array}{cc} c_0 \left(\frac{1}{d^\alpha} - \frac{1}{r_{zone}^\alpha} \right) & , \text{if} r_{\min} < d < r_{zone} \\ c_0 \left(\frac{1}{r_{\min}^\alpha} - \frac{1}{r_{zone}^\alpha} \right) & , \text{if} d \leq r_{zone} \\ 0 & , \text{if} d \geq r_{zone} \end{array} \right\} \tag{2.12}$$

where d is the distance between the nodes, r_{min} is the distance at which the capacity decay begins and r_{zone} is the distance at which it decays to zero.

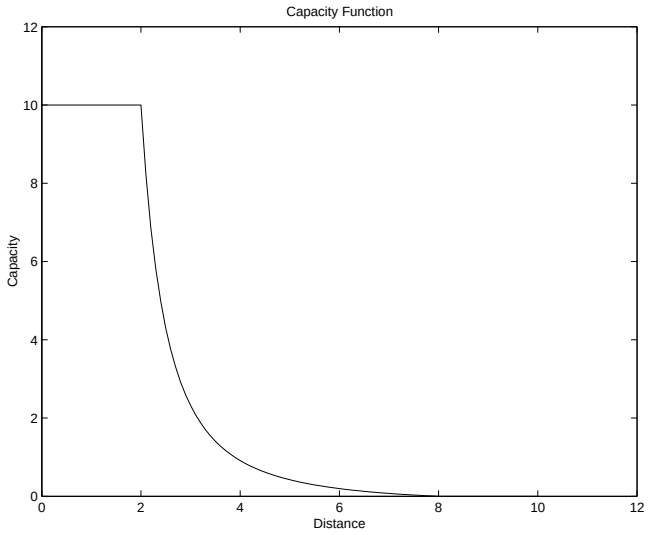

Figure 2.9. Wireless capacity as a function of distance [8].

CHAPTER 3

ALGORITHMS

3.1 LQR Trajectory Tracking

Linear Quadratic Regulator (LQR) trajectory tracking was used for path planning. LQR is based on optimal control theory [11] [8] [36]. The desired trajectory is assumed to be known from some path-planning stage and the tracking controller then guarantees that the system will follow the desired trajectory. The gains for the tracking controller are obtained via LQR.

The non-holonomic system is linearized about the desired open-loop path and a controller is used to keep the system on that path. A time-varying LQR controller is used to regulate the linearized system to zero.

3.1.1 LQR Controller

The discrete form of the linearized system equation is:

$$\delta x(k+1) = A_d(k)\delta x(k) + B_d(k)\delta u(k), \delta x(0) = 0 \tag{3.1}$$

where A_d and B_d are the discrete versions of the state and input matrices.

The total control law can be expressed as a feed forward control (which would drive an ideal system along the desired trajectory) and a feedback part that regulates the (non ideal) linearized system to zero. This gives the total control law to be:

$$u(k) = u_f(k) + K(k)(x_d - \hat{x}(k)) \tag{3.2}$$

where $K(k)$ is the LQR gain. This gain has to be calculated apriori as follows:

Let a cost function be defined as J

$$J = \frac{1}{2}\sum_{k=0}^{\infty}\left(x_k^T Q x_k + u_k^T R u_k\right) \tag{3.3}$$

We wish to optimize J resulting in:

$$K(k) = \left[B_d(k)P(k+1)B(k) + R(k)\right]^{-1} B_d(k)P(k+1)A(k) \tag{3.4}$$

$$P(k) = A_d(k)P(k+1)\left[A(k) - B(k)K(k)\right] + Q(k) \tag{3.5}$$

B_d is found by separating variables

$$\dot{x} = B_d \bullet u \tag{3.6}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \frac{r_R}{2}\cos\phi & \frac{r_L}{2}\cos\phi \\ \frac{r_R}{2}\sin\phi & \frac{r_L}{2}\sin\phi \\ -\frac{r_R}{b} & \frac{r_L}{b} \end{bmatrix} \bullet \begin{bmatrix} u_R \\ u_L \end{bmatrix} \tag{3.7}$$

$$A_d = J_1 u_r + J_2 u_l \tag{3.8}$$

where J_i is the Jacobian for the ith column of B_d. Thus

$$A_d = \begin{bmatrix} 0 & 0 & -\frac{r}{2}\sin\phi(ur + ul) \\ 0 & 0 & -\frac{r}{2}\cos\phi(ur + ul) \\ 0 & 0 & 0 \end{bmatrix} \tag{3.9}$$

$P(N)$ is the end configuration weighting matrix, $Q(k)$ is the configuration weighting matrix and $R(k)$ is the control weighting matrix.

3.1.2 Choosing initial conditions

K(k) must be obtained before hand and is done by working backward in time from the P(k) and K(k) equations. Initial condions must be chosen for P(N) and Q and R and this is a challenging task. Simulations and trial and error were used to find the best values.

From experimentation is is known that P(N), Q and R are path dependant, so each different path will require re-calculation of these values. Thus for each path, it is required to simulate and choose appropriately these values. [3,4]

Q is the configuration (3x3) matrix. The physical significance of Q is the amount of control effort or importance given to each state (in our case, the x, or the y or the phi). R is the control (2x2) matrix which allows us to choose the control effort diverted to the inputs. Since all of our inputs are equally important, this matrix is initialized to eye(2)

3.1.3 Special Cases

Although the LQR controller can handle arbitrary trajectories, much computing power is required to pre-calculate the feed-back gains $K(k)$. Thus we treat individually the special cases of 'straight line' and 'axis turn' and attempt to find optimizations.

3.1.3.1 Straight Line Trajectory Tracking

Moving in a straight line is a special case as only one of the states is changed - x. From the K(k) and P(k) equations the K(k) = constant for all k. This stems from:

$$B_d = h \times \begin{bmatrix} \frac{r_R}{2}\cos\varphi & \frac{r_L}{2}\cos\varphi \\ \frac{r_R}{2}\sin\varphi & \frac{r_L}{2}\sin\varphi \\ \frac{r_R}{l} & -\frac{r_L}{l} \end{bmatrix} \tag{3.10}$$

Since φ is always 0, $\cos\varphi = 1$ and $\sin\varphi = 0$.

Thus B_d reduces to

$$B_{d(straightLine)} = h \times \begin{bmatrix} \frac{r_R}{2} & \frac{r_L}{2} \\ 0 & 0 \\ \frac{r_R}{l} & -\frac{r_L}{l} \end{bmatrix} \tag{3.11}$$

which is a constant matrix.

$$A_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + h \begin{bmatrix} 0 & 0 & -\frac{r}{2}\sin\phi(u_R + u_L) \\ 0 & 0 & -\frac{r}{2}\cos\phi(u_R + u_L) \\ 0 & 0 & 0 \end{bmatrix} \tag{3.12}$$

This again reduces to

$$A_{d(straightLine)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + h \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{r}{2}(u_R + u_L) \\ 0 & 0 & 0 \end{bmatrix} \tag{3.13}$$

Thus A_d is always a constant and for straight line movements, the LQR gain $K(k)$ is a constant for all K.

3.1.3.2 Axis-Turn Trajectory Tracking

Turning a set angle also has only one of the states change - phi. However the K(k) is not constant contrary to what one would expect.

$$A_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + h \begin{bmatrix} 0 & 0 & -\frac{r}{2}\sin\phi(u_R + u_L) \\ 0 & 0 & -\frac{r}{2}\cos\phi(u_R + u_L) \\ 0 & 0 & 0 \end{bmatrix} \tag{3.14}$$

Since $u_R = -u_L$, this reduces to

$$A_{d(turn)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.15}$$

$$B_d = h \times \begin{bmatrix} \frac{r_R}{2}\cos\varphi & \frac{r_L}{2}\cos\varphi \\ \frac{r_R}{2}\sin\varphi & \frac{r_L}{2}\sin\varphi \\ \frac{r_R}{l} & -\frac{r_L}{l} \end{bmatrix} \tag{3.16}$$

Thus, $A_d = I$ and B_d does not remain constant. This makes K variable with k.

In the interest of implementation of a working turning algorithm via LQR algorithm, the error in x and y is ignored. This allows us to have a constant LQR gain K and is easier to implement. With better hardware, the error in x and y can also be controlled.

3.2 Formations with Potential Fields

The aim is to have one leader node and several follower nodes maintain a formation. The follower ARRIbots must follow the leader (i.e. maintain a specified distance and

angle with respect to the leader) in a given formation. The leader is the node that defines the overall movement and orientation of the formation.

3.2.1 Defining the formation

A higher level algorithm must decide the shape of the formation. For example, a raster sampling application will require a horizontal chain formation, a gas field may require a diamond formation, and others may require any arbitrary formation. Here we provide a method for specifying the formation in terms of relative distances and angles.

The formation is defined by the location in a formation reference frame $'T'$. Figure 3.1 shows an example of a triangle formation with the leader (Robot 1) at $(50, 0, 0)$, Robot 2 at $(0, 28.87, 0)$ and Robot 3 at $(0, -28.87, 0)$. The three robots from an equilateral triangle formation.

Similarly, Figures 3.2 and 3.3 show a diamond formation and a vertical chain formation respectively.

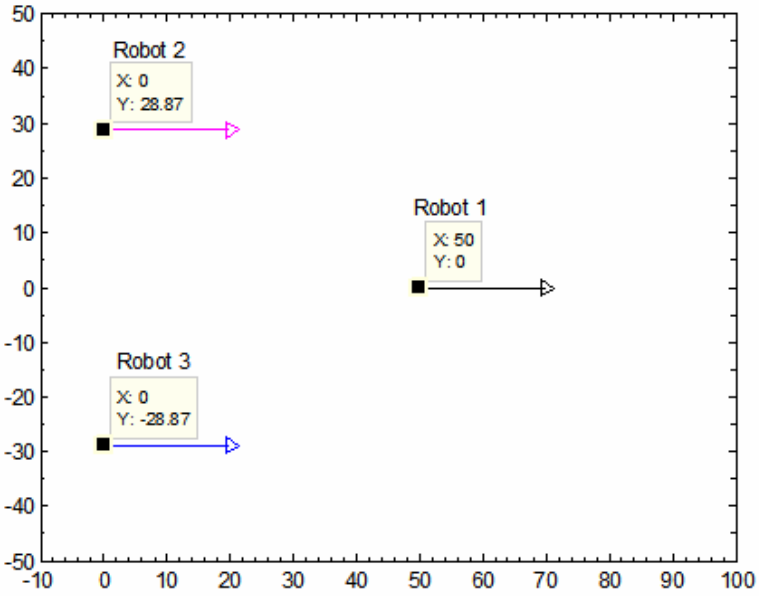

Figure 3.1. Triangle Formation.

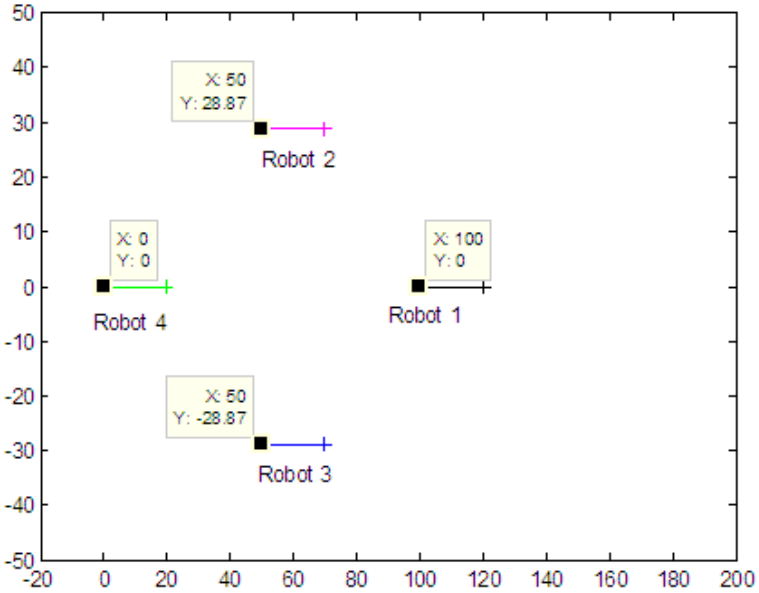

Figure 3.2. Diamond Formation.

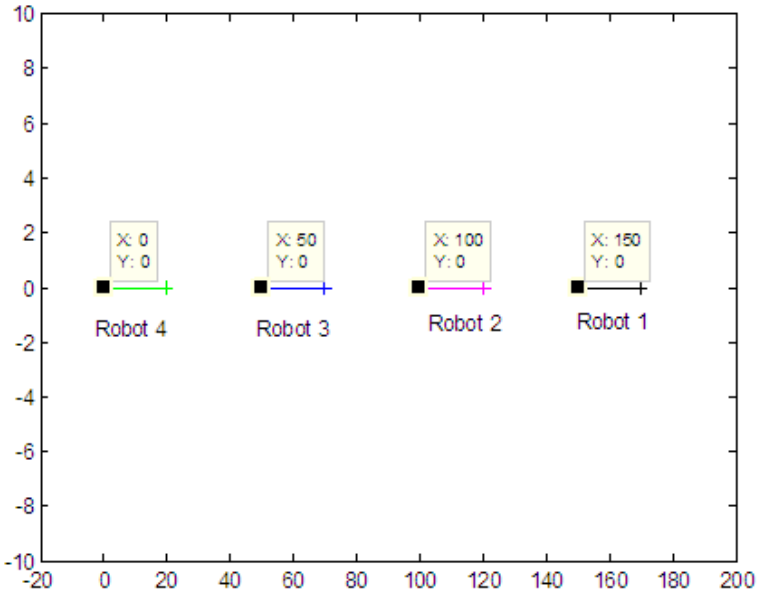

Figure 3.3. Chain Formation.

3.2.2 Leader Path planning

We again assume that the leader path has already been chosen by a higher level algorithm. One method for leader path planning is with the use of a potential field made up of an attractive field toward the goal U_{goal} and a net repulsive field $U_{obstacle}$ away from all obstacles [37].

$F_{goal}(x)$ is an attractive force that allows the leader to reach the goal. Let $z := (x_z, y_z)$ be the desired goal position. We can then choose a parabolic potential

$$U_{goal}(x) = \frac{1}{2}\zeta \left\| x - \hat{z} \right\|^2 \tag{3.17}$$

The resultant artificial goal force at any point x defined as:

$$F_{goal}(x) = -\nabla U_{goal}(x) \tag{3.18}$$

$$= \left(-\frac{\partial}{\partial x} U_{goal,}(x,y), -\frac{\partial}{\partial y} U_{goal,}(x,y) \right) \tag{3.19}$$

$$= \zeta \left(x_z - x, y_z - y \right) \tag{3.20}$$

$F_{obstacle}(x)$ is a repulsive force that allows the leader to avoid the other robots and obstacles. Let H be the set of all obstacles (including other robots) in the environment. For robot i for all points $x \epsilon \mathbb{R}^2 - H$ we can define a repulsive obstacle potential $U_{obstacle}(x)$, [37] where

$$U_{obstacle,i}(x) = \left\{ \begin{array}{c} \frac{1}{2}\eta \left(\frac{1}{d(x)} - \frac{1}{d_0} \right)^2, \text{ if } d(x) \leqslant d_0 \\ 0 \text{ , if } d(x) > d_0 \end{array} \right\} \tag{3.21}$$

where $d(x)$ is the distance from the robot to the obstacle and d_0 is the radius of the range of effect of the obstacle. This is needed so that the obstacle has no effect on nodes that are sufficiently far away. $U_{obstacle}$ tends to infinity as the robot gets closer to the obstacle and becomes zero when it leaves the range of effect of the obstacle.

The artificial repulsive obstacle force at any point x defined as:

$$F_{obstacle,i}(x) = -\nabla U_{obstacle}(x) \tag{3.22}$$

$$F_{obstacle,i}(x) = \left\{ \begin{array}{cc} \eta \left(\frac{1}{d(x)} - \frac{1}{d_0} \right) \frac{1}{d^2(x)} \angle d(x) & \text{, if} d(x) \leq d_0 \\ 0 & \text{, if} d(x) > d_0 \end{array} \right\} \tag{3.23}$$

where $\angle d(x)$ is the unit vector in the direction of the repulsive force.

The resultant obstacle force is simply a summation of all the repulsive forces acting from all the obstacles:

$$F_{obstacle}(x) = \sum_{i \in H} F_{obstacle,i}(x) \tag{3.24}$$

Finally the resultant force is a vector sum of the the goal and obstacle forces:

$$F_{leader}(x) = F_{goal}(x) + F_{obstacle}(x) \tag{3.25}$$

Other methods of leader path planning may also be used. Some examples are Shortest Path algorithms, A^*, Voronoi Graph reduction [38], etc. For the purposes of simulation and experimental testing for this thesis, a fixed leader trajectories (such as straight line, curved, and discrete path) were simply pre-selected to demonstrate the benefits of the potential field formation control method.

3.2.3 Follower Artificial Forces

The follwer robots are required to maintain the formation with respect to three criteria:

- Geometric Centric
- Communications Centric
- Information Centric

The geometric centric requirement will help the robots maintain a formation as specified in Section 3.2.1. The artificial force F_{form} is introduced in order to satisfy this.

The Communications Centric formation will seek to optimize the placement of the nodes to maximize the capacity of the wireless channels. In particular, here we look at the middle node configuration as described in Section 2.5. The artificial force $F_{communications}$ is introduced in order to satisfy this.

The information centric requirement stems from the need to alter the formation geometry and position based on an external information gathering algorithm. An example would be adaptive sampling of a linear field - the sampling points are updated every iteration and the robots must change formation to reach these points. This requirement can

be satisfied by changing the desired formation expected positions in a dynamic manner as in Section 3.2.1.

The final force on the follower robot(s) is a vector sum of all the individual component forces:

$$F_{follower}(x) = F_{form}(x) + F_{obstacle}(x) + F_{communications}(x) \tag{3.26}$$

3.2.3.1 Formation Force

The formation force is described as an attractive force $F_{form}(x)$ toward the expected formation position $p_{L,i}$. The formation is maintained by the use of artificial forces. Each force contributes to the final direction and speed of the movement of each robot. Let n = number of robots

p_i = position of robot i p_L = position of Leader robot

The expected position of robot i with respect to the leader L is:

$$\hat{p}_{L,i} = p_L + \begin{bmatrix} \cos(\phi_L) & -\sin(\phi_L) \\ \sin(\phi_L) & \cos(\phi_L) \end{bmatrix} (p_{i_T} - p_{L_T}) \tag{3.27}$$

where p_{i_T} and p_{L_T} are the positions of robot i and the leader respectively in the formation definition frame T as in Section 3.2.1 and ϕ_L is the current orientation of the leader robot.

The attractive potential $U_{form,i}$ is defined as parabolic:

$$U_{form,i}(p) = \frac{1}{2}\xi \left\| p_i - \hat{p}_{L,i} \right\|^2 \tag{3.28}$$

With the potential Force at any point p defined as:

$$F_{form,i}(p) = -\nabla U_{form,}(p) \tag{3.29}$$

$$F_{form,i}(p) = \xi \begin{bmatrix} (\hat{x}_{L,i} - x) \\ (\hat{y}_{L,i} - y) \end{bmatrix} \tag{3.30}$$

3.2.3.2 Obstacle Avoidance Force

The individual robots also need to avoid obstacles that may be in the environment. $F_{obstacle}(x)$ is a repulsive force that allows the leader to avoid the other robots and obsta-

cles. Let H be the set of all obstacles (including other robots) in the environment. For robot i for all points $x\epsilon\mathbb{R}^2 - H$ we can define a repulsive obstacle potential $U_{obstacle}(x)$, [37] where

$$U_{obstacle,i}(x) = \left\{ \begin{array}{c} \frac{1}{2}\eta\left(\frac{1}{d(x)} - \frac{1}{d_0}\right)^2, \text{ if } d(x) \leqslant d_0 \\ 0 \text{ , if } d(x) > d_0 \end{array} \right\} \tag{3.31}$$

$d(x,i)$ is the distance from the robot i at x to the obstacle and d_0 is the radius of the range of effect of the obstacle. This is needed so that the obstacle has no effect on nodes that are sufficiently far away. $U_{obstacle}$ tends to infinity as the robot gets closer to the obstacle and becomes zero when it leaves the range of effect of the obstacle.

The artificial repulsive obstacle force at any point x defined as:

$$F_{obstacle,i}(x) = -\nabla U_{obstacle}(x) \tag{3.32}$$

$$F_{obstacle,i}(x) = \left\{ \begin{array}{cc} \eta\left(\frac{1}{d(x)} - \frac{1}{d_0}\right)\frac{1}{d^2(x)}\angle d(x) & \text{, if} d(x) \leq d_0 \\ 0 & \text{, if} d(x) > d_0 \end{array} \right\} \tag{3.33}$$

where $\angle d(x)$ is the unit vector in the direction of the repulsive force. η is the weight that we can assign to scale the magnitude of the force. This particular value will depend of the size of the obstacle and it's 'threat' level. The resultant obstacle force is simply a summation of all the repulsive forces acting from all the obstacles:

$$F_{obstacle}(x) = \sum_{i \in H} F_{obstacle,i}(x) \tag{3.34}$$

With the use of the obstacle avoidance force, the formation can now:

- Go around obstacles
- Deform to squeeze through narrow openings

3.2.3.3 Communications Forces

The network flow model was described in Section 2.5 along with the routing matrix R which describes the network sources, sinks and links.

Optimizing the network flow control problem is divided into two parts: a static optimization problem and a dynamic stabilization problem [8].

The static optimization of the network flow control problem consists of maximizing a chosen utility function for all of the sources for a fixed capacity constraint,

$$\max_{x \geq 0} \sum_{i=1}^{N} U_i(x_i) \text{ subject to } Rx \leq c \tag{3.35}$$

where U_i is the strictly concave utility function and R is the routing matrix for the network. There are various utility functions for different network models: $\arctan(x_i)$ for TCP Reno and $\log(x_i)$ for TCP Vegas [35].

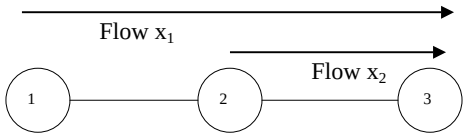

Figure 3.4. Middle Node Configuration [8].

We consider a middle node configuration in this thesis. Here, one node is linked with two other nodes on the network (Figure 3.4) and we aim to find the force on this middle node due to the locations of the other two nodes [8].

As mentioned in Section 2.5 the data rate between nodes varies with the distance between the nodes. As the distance increases between nodes the signal-to-noise power decreases which in turn decreases the data rate. The optimal utility function value will vary with node location. This problem can be posed as an unconstrained optimization problem using Lagrange multipliers:

$$U^*(r_{i=1\ldots N}) = \min_{p_l>0} \max_{x_o>0} (\sum_{i=1}^{N} w_i U(x_i) + \sum_{l=1}^{L} p_l(c_l(r_i) - y_l)) \tag{3.36}$$

Since capacity depends upon the distance between nodes a second maximization is performed to find the optimal node positions.

$$\max_{\mathbf{r}} \max_{x \geq 0} \sum_{i=1}^{N} U_i(x_i) \text{ subject to } Rx \leq c(\mathbf{r}) \tag{3.37}$$

$$\begin{aligned} \max_{x \geq 0} \quad & \sum U_1(x_1) + U_2(x_2) \\ \text{subject to} \quad & Rx \leq c \end{aligned} \tag{3.38}$$

where $R = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ and $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and c is given from the capacity model. The utility function is taken to be $log(x_i)$, which is a variant of TCP Vegas protocol [35].

This optimized utility function is used as the potential field $U_{communications}$. The artificial communications force $F_{communications}(x)$ can be defined as the derivative of the potential field:

$$F_{communications}(x) = -\nabla U_{communications}(x) \tag{3.39}$$

This force is dependent on the capacity of the wireless communication link between the nodes, which is in turn dependent on the distance between the nodes.

To solve this maximization a relaxation method was used in [8]. The results are directly used here as as look up tables in one and two dimensions. Figure 3.5 shows the potential and forces in 1D and Figure 3.6 shows the potential for the 2D case.

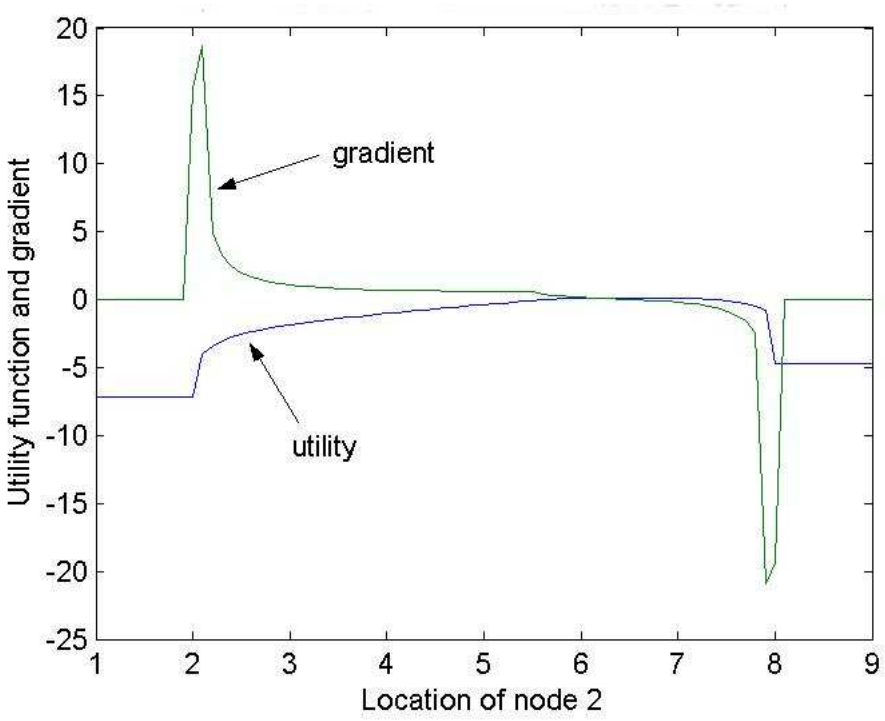

Figure 3.5. 1D utility function and gradient for middle node configuration [8].

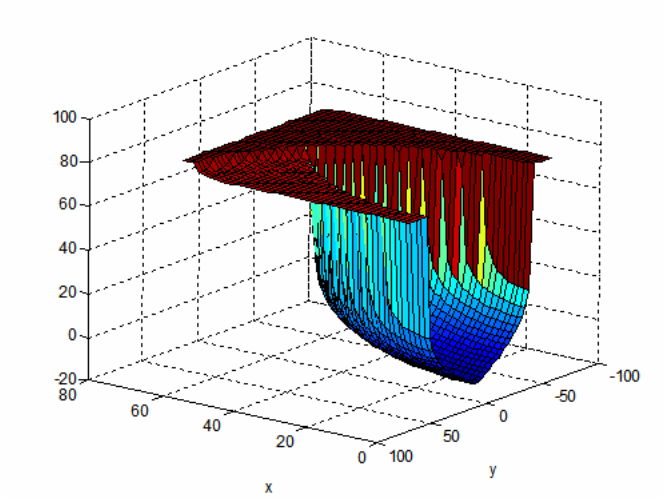

Figure 3.6. 2D potential function for middle node configuration [8].

Routing

The Mine-Field Sweep simulation (Section 4.4) required transmission of sampling data from the testing field to a sufficiently far away base station. A modified version of the Zone Routing Protocol (ZRP) was used as in [8]. Individual nodes are modeled to have a finite transmission distance r_{zone}. This necessitated a routing protocol to ensure that messages are delivered in a multi-hop fashion. The objective is to come up with routing paths for each node that began at the node, pass through middle nodes as required and finally end at the base station.

For this simulation a shortest path approach - $A*$ is used. We consider each node to be a graph vertex and the wireless link to be the edge. The edge weights are initialized to be equal as a wireless transmission can be considered to have the same cost over different distances as long as the nodes are within range. The heuristic used is the cartesian distance from the node to the base station. Figure 3.7 shows an example route for Robot 2.

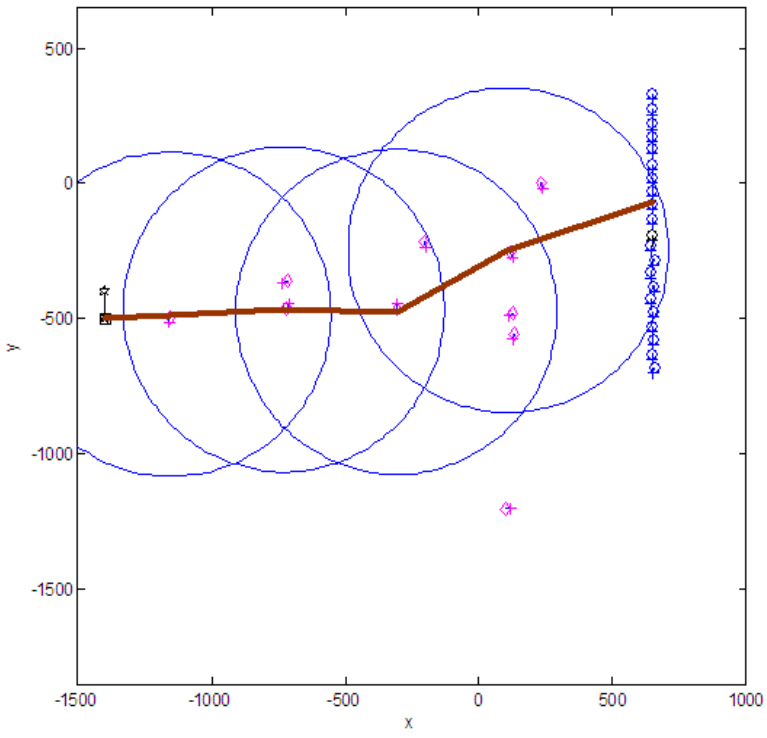

Figure 3.7. Communications Route Example.

A routing table is generated and maintained for every node at every iteration. The purpose of the routing table is to represent the graph for the A* algorithm. Table 3.1 shows an example of the routing matrix for node 10 for the scenario shown in Figure 3.8. The table (or matrix) is $N \times N$ where N is the total number of nodes. The element in the ith row and jth column is 1 if node j is within communications range r_{zone} and 0 otherwise.

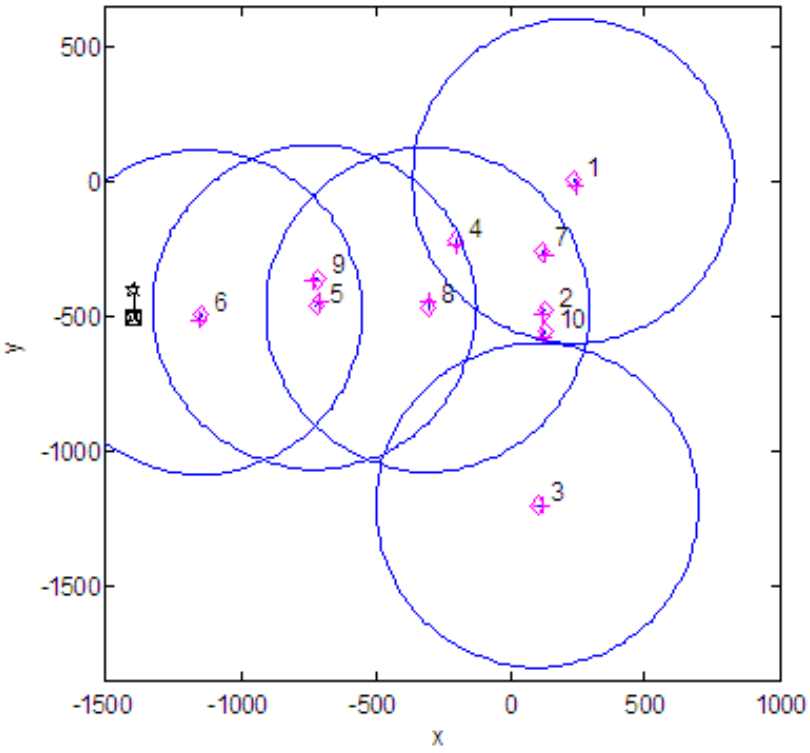

Figure 3.8. Routing Matrix Example Scenario.

The next node in a route is chosen from the list of nodes that are in range (given by the routing matrix). For each potential next node, a heuristic is calculated - the cartesian distance from the potential node to the base station and the node with the minimum heuristic is chosen. However, if the heuristic is greater that that of the previous node in the routing path, a stop condition is reached and there is no possible route through this node. The process is repeated with the next best potential node until either:

- A better node is found that has a lower heuristic that the previous node

Table 3.1. Routing Matrix for Node 10

	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9	Node 10
Node 1	0	1	0	1	0	0	1	0	0	1
Node 2	1	0	0	1	0	0	1	1	0	1
Node 3	0	0	0	0	0	0	0	0	0	0
Node 4	1	1	0	0	1	0	1	1	1	1
Node 5	0	0	0	1	0	1	0	1	1	0
Node 6	0	0	0	0	1	0	0	0	1	0
Node 7	1	1	0	1	0	0	0	1	0	1
Node 8	0	1	0	1	1	0	1	0	1	1
Node 9	0	0	0	1	1	1	0	1	0	0
Node 10	1	1	0	1	0	0	1	1	0	0

- The potential next nodes are exhausted. The algorithm takes one step back and chooses the next best node for the previous step.

If the algorithm back-tracks all the way to the source node, there is no possible path to the base from the source node.

$A*$ is an example of an informed search algorithm, thus it is dependent on the heuristic function to lead it closer to the goal node [39] [40]. The routing path already traveled is used as a testing criteria thus the algorithm is complete and optimal; i.e. the shortest path will always be found given one exists and that the heuristic never overestimates the actual cost.

3.2.3.4 Generating Movement from Forces

Once the artificial force $F(x)$ acting on the node has been calculated, the node must move accordingly. One method of doing this is to use a mass damper model [8]:

$$m_i\ddot{\mathbf{r}}_i + \nu_i\dot{\mathbf{r}}_i = \mathbf{F}_i \tag{3.40}$$

where m and ν are mass and damping terms.

However for our case, we have only a kinematic model and are ignoring acceleration effects. Thus we have

$$\nu_i\dot{\mathbf{r}}_i = \mathbf{F}_i \tag{3.41}$$

Where direction of the force $\angle F(x)$ will give the new direction of the node, while the magnitude $||F(x)||$ will determine the speed. ν is chosen by trial and error.

$$v \propto F(x) \Rightarrow v = \mu F(x) \tag{3.42}$$

and

$$\phi = \angle F(x) \tag{3.43}$$

This definition of robot path planning using potential fields could result in the the robot being trapped in a local minima. To avoid this, these should be combined a higher level algorithm that could be based on heuristics, graph methods or similar.

3.3 ARRIBot Implementaion

3.3.0.5 LQR Controller

Figure 3.9 shows a block diagram of the controller operation. A desired path $\vec{x}_d = \begin{bmatrix} x & y & \varphi \end{bmatrix}^T$ is fed as input and the difference from the estimate and desired is multiplied by the LQR gain $K(k)$ to produce the offset to the Feed-forward input (u_f) calculated using the formulas resulting from the model. The feedforward inputs U_L and U_R for straight line are :

$$\mathrm{u_L} = \mathrm{r_L} \bullet \mathrm{MoveSpeed} \bullet \frac{180}{\pi} \tag{3.44}$$

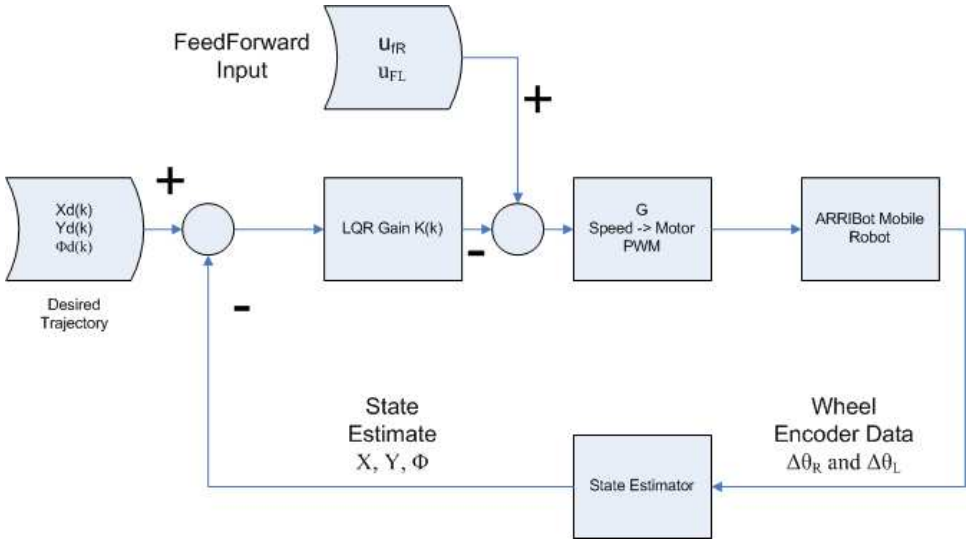

Figure 3.9. LQR Controller block diagram.

$$u_R = r_R \bullet \text{MoveSpeed} \bullet \frac{180}{\pi} \tag{3.45}$$

The feedforward inputs U_L and U_R for turning are :

$$u_L = l \bullet \frac{\text{TurnSpeed}}{2r_L} \tag{3.46}$$

$$u_R = l \bullet \frac{\text{TurnSpeed}}{2r_R} \tag{3.47}$$

This input is then converted to PWM values required by the motors (through G).

The system is measured through encoder counts $\delta\theta_l$ and $\delta\theta_r$) and these are used along with the discrete model of the system defined in Chapter 2 to get the estimated state $\hat{\vec{x}}$.

Figure 3.10 shows a simplified overall structure of the ARRIBot motion. After initializing the hardware, the ARRIbot waits for either a move or a turn command. The command is executed and the ARRIbot returns to the wait state.

Figure 3.11 shows a detailed flow of executing a move or turn command.

Multiple Steps

The controller (and the requirement for constant K) has been designed so that it all actions are assumed to start from (0,0) and that the x and y directions are those in the traditional sense (East and North).

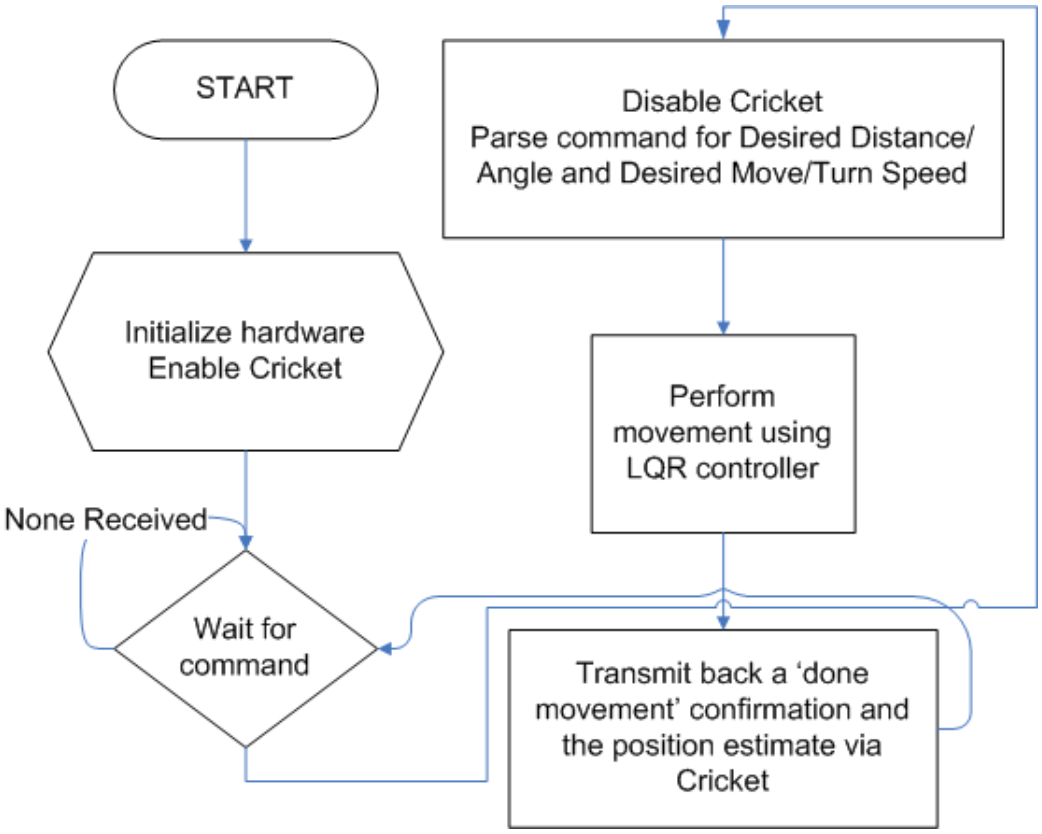

Figure 3.10. LQR Controller overall flow diagram.

An issue that arises is that after the first step, the origin for the next one is no longer (0,0) and the orientation also may not be 0. Thus a transformation is required to convert from the original frame of reference to the new frame of reference. As figure 3.12 shows we start with the robot in initial frame x, y with state $[x_o, y_o, \varphi_o]^T$. After the move, the robot is in it's new frame x', y' and has differential new local estimates $dx, dy, d\varphi$ in this new frame. From the geometry of the problem:

$$d\theta = \tan^{-1}\left(\frac{dy}{dx}\right) \tag{3.48}$$

$$d = \sqrt{dx^2 + dy^2} \tag{3.49}$$

$$\tilde{y} = d \bullet \sin(d\theta + d\varphi) \tag{3.50}$$

$$\tilde{x} = d \bullet \cos(d\theta + d\varphi) \tag{3.51}$$

Giving the position in the original global frame x, y of:

$$x = x_o + \tilde{x} \tag{3.52}$$

$$y = y_o + \tilde{y} \tag{3.53}$$

$$\varphi = \varphi_o + d\varphi \tag{3.54}$$

This transformation must be done after every move in order to preserve the frame of reference of the robot. A consequence of this transformation is that the *arctan* function is required. This function is not available in the Javelin Stamp on the ARRIBot, thus a piece-wise linear approximation was created by splitting the function into three parts. The final obtained approximation was:

$$\tan^{-1}(\theta) \approx \left\{ \begin{array}{cc} 16(\theta) + 22 & 0 \leq \theta \leq 4 \\ 1.2(\theta) + 73 & 4 < \theta \leq 11 \\ 0.066(\theta) + 86 & \theta > 11 \end{array} \right\} \tag{3.55}$$

R/C servo characteristics

System G from Figure 3.9 represents the conversion from angular velocity to the Pulse Width Modulation(PWM) value required by the servos. The servos were each run under load for various PWM values and time delays and the encoder counts were then normalized to 1 second. A linear curve fit was applied to the regions that are not saturated. Figure 3.16 and 3.17 show the motor calibration results for the Right and Left motors respectively.

The data was fit to a linear equation:

$$PWM = \Delta\theta * K_m + K_b \tag{3.56}$$

The resulting values (for both motors) were $Km = 0.6$ and $Kb = 173$. This was expected as 173 is the stop value for the motors.

3.3.0.6 1 Dimensional Leader-Follower

A one dimensional formation consists of a leader robot followed by single file robots. The leader is assumed to move only in the x direction and the follower robot(s) must maintain a formation (in this case defined by a desired distance $d_{desired}$).

From Figure 3.18,

$$\Delta d = d_{measured} - d_{desired} \tag{3.57}$$

where $d_{measured}$ is $x_2 - x_1$. This is used to calculate the formation potential field U_{form} and the corresponding force F_{form}. The ARRIbot then will move with a speed v and distance Δd.

The sampling time was chosen to be 1 second. Thus the ARRIbot will take a distance measurement every 1s and recalculate the desired speed and distance to move. The distance measurement itself was obtained using the onboard ultrasound ranger (SRF04) (Figure 3.19) which was calibrated using a linear fit (Fig 3.20). Table 3.2 shows the relationship between ultrasound counts and the actual distance. The final formula was found to be

$$ActualDistance = 0.148 \times UltraSoundCount + 14.63 \tag{3.58}$$

Table 3.2. Ultrasound Calibration

Actual Distance (cm)	Ultrasound Counts
30	103
40	170
60	312
80	444
100	580
120	706
140	844
160	976
190	1184
220	1388
250	1610
260	1650
270	1722
280	1789

Figure 3.21 shows a flowchart of the leader-follower behavior.

3.3.0.7 2 Dimensional Leader-Follower

In the 2-D case, the operation is similar but the calculations are more involved. Relative distance and orientation cannot be obtained using one ultrasound ranger so an overhead infrared camera system was used in conjunction with image processing techniques was used.

Figure 3.22 shows a block diagram of the 2-D follower algorithm. The sampling time was chosen as 3s. On every iteration, the camera was used to obtain the leader and follower state vector. A potential field U_{form} is generated using the expected position from Section 3.2.1.2. along with the obstacle potential field $U_{obstacle}$from section 3.2.1.3. The two fields are vectorially added and the gradient is taken to get the net artificial force $F(x)$.

This force is used to plan a path to the desired position with speed proportional to $||F(x)||$. This path will consist of 2 separate actions: A turn to face the correct direction and a straight line move to the desired location.

The loop continues until one of two stopping conditions is met. If $D_{toMove} < d_{deadzone}$, then just turn the angle required. Also, in order to prevent small orientation changes making a lot of stopping and starting motion, there is a second rule that - If $\angle_{toTurn} < \angle_{deadzone}$, then move the required distance. The Motors are never stopped unless either of the two stopping conditions are met.

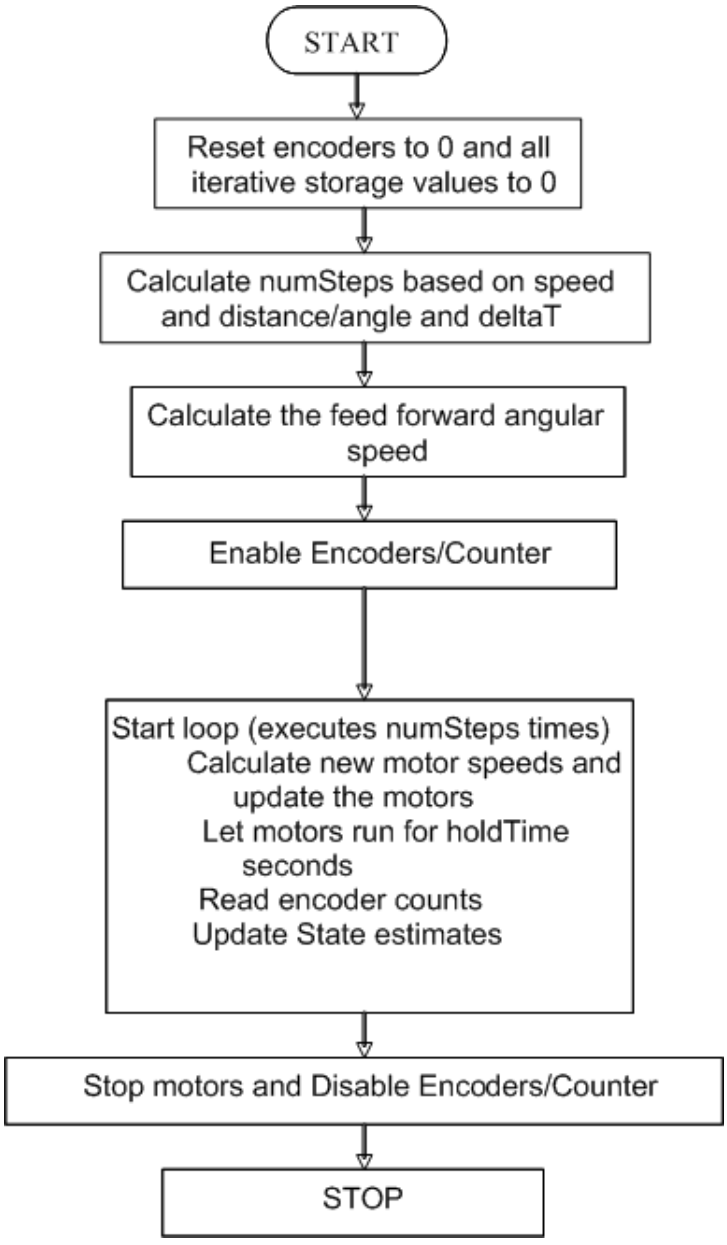

Figure 3.11. LQR Controller single action flow diagram.

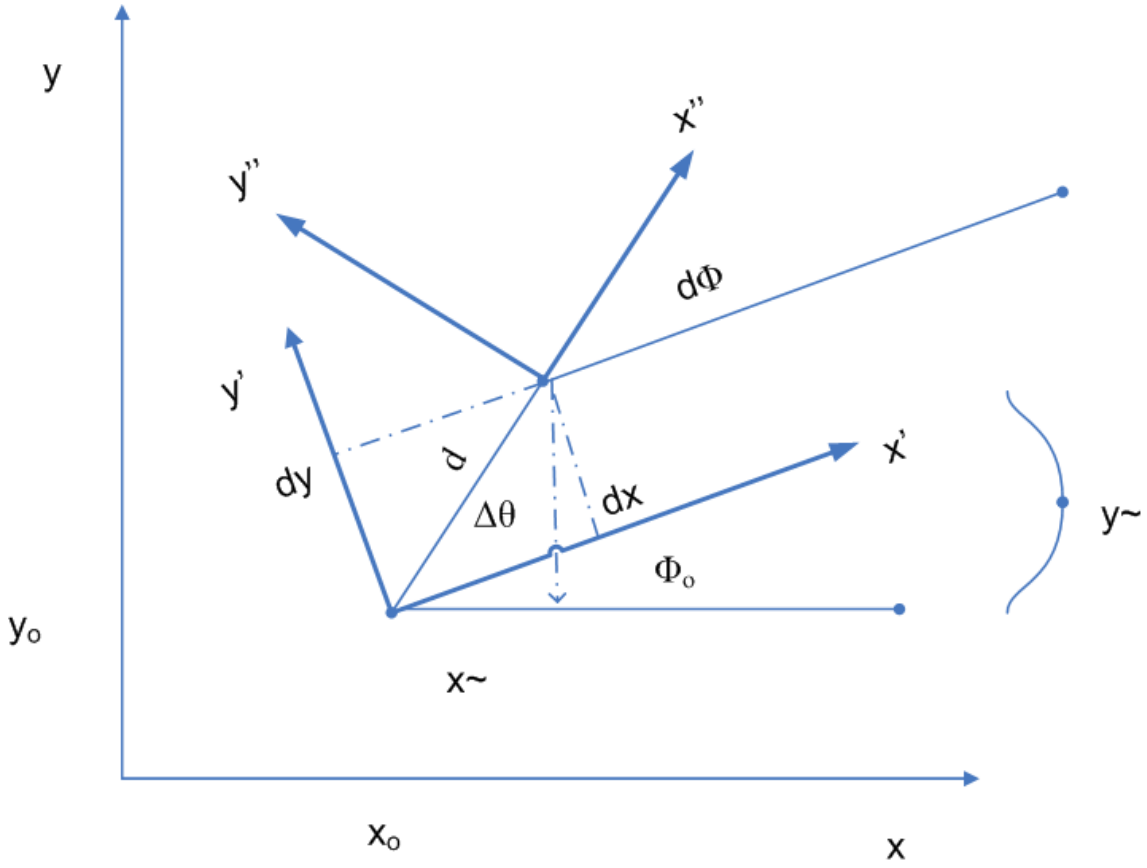

Figure 3.12. Multiple Step Reference frame transformation.

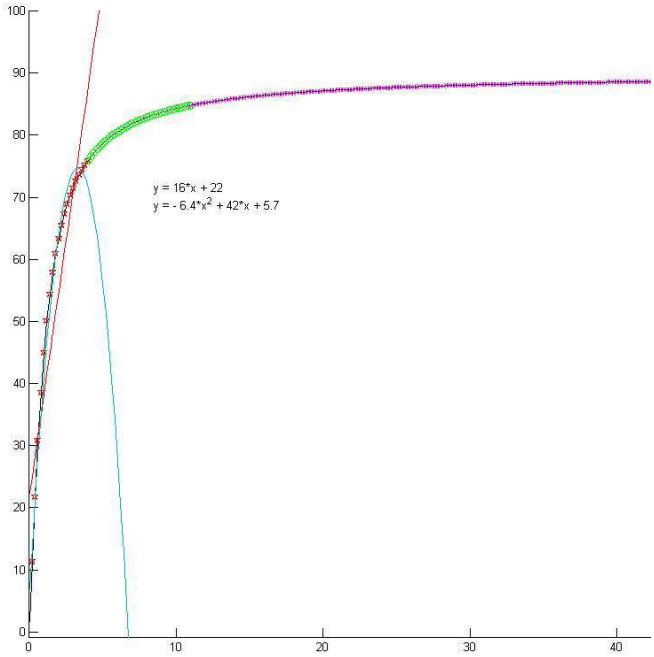

Figure 3.13. Arc Tangent Piece-wise approximation($0 \leq \theta \leq 4$).

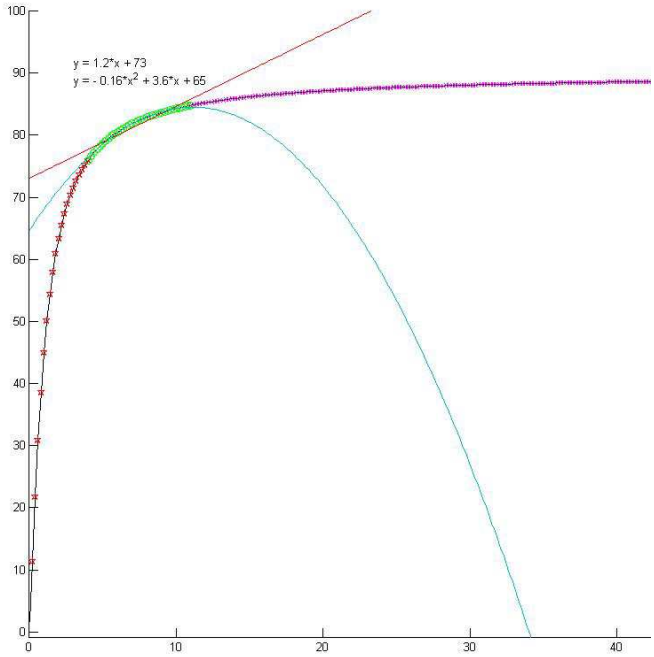

Figure 3.14. Arc Tangent Piece-wise approximation($4 < \theta \leq 11$).

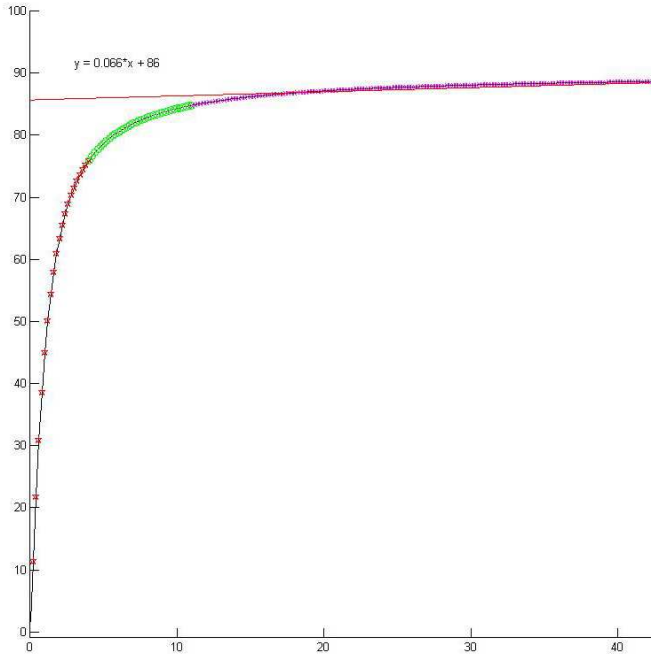

Figure 3.15. Arc Tangent Piece-wise approximation($\theta > 11$).

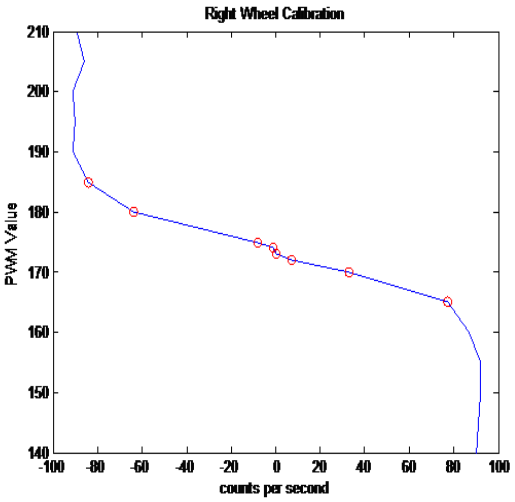

Figure 3.16. Motor Calibration (Right Wheel).

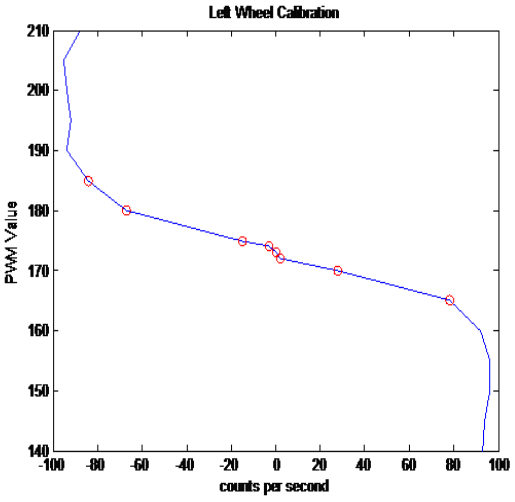

Figure 3.17. Motor Calibration (Left Wheel).

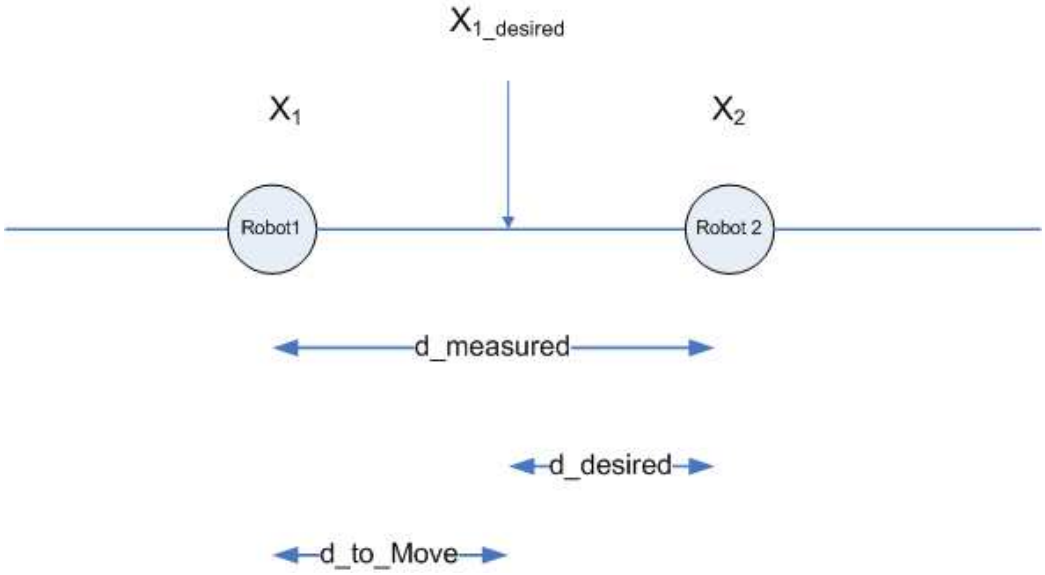

Figure 3.18. 1-D Leader-Follower Formation.

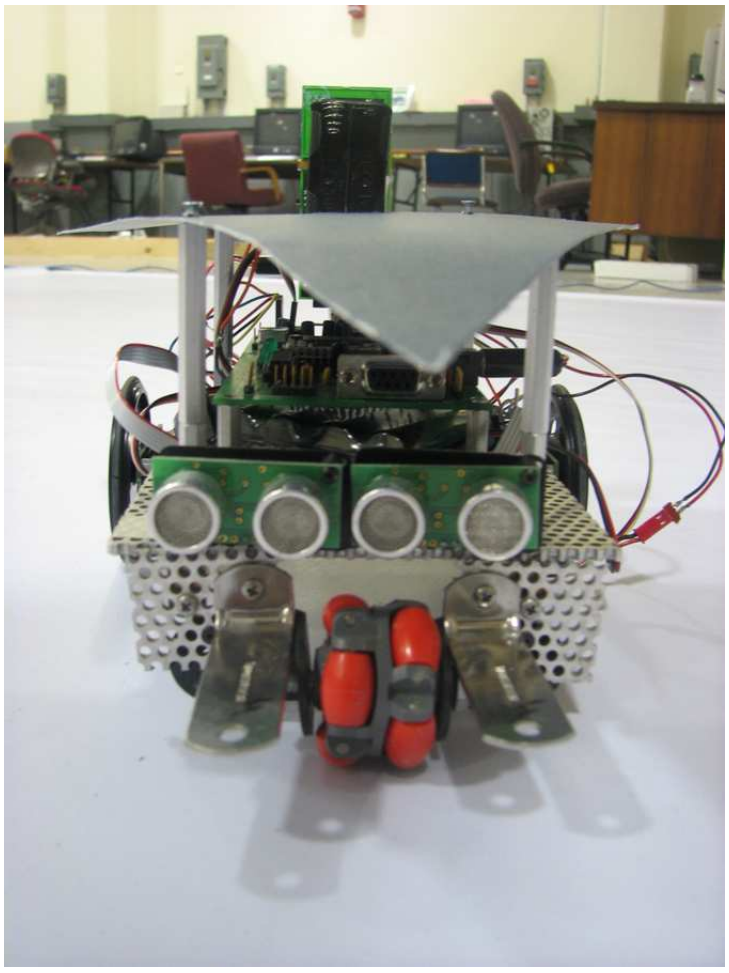

Figure 3.19. ARRIBot Ultrasound Sensor (SRF04)).

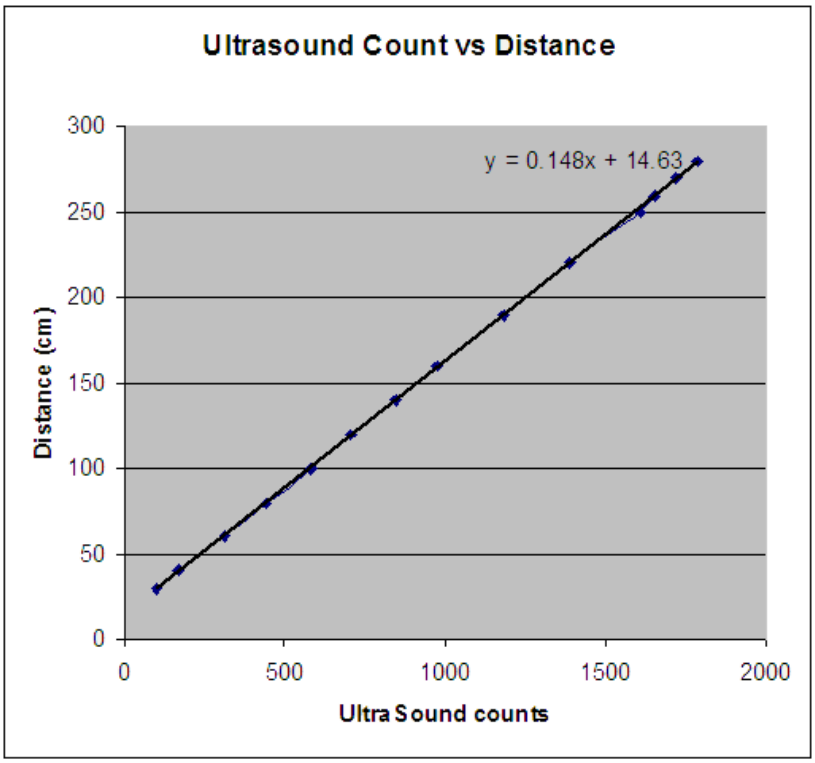

Figure 3.20. Ultrasound Counts vs Distance (cm)).

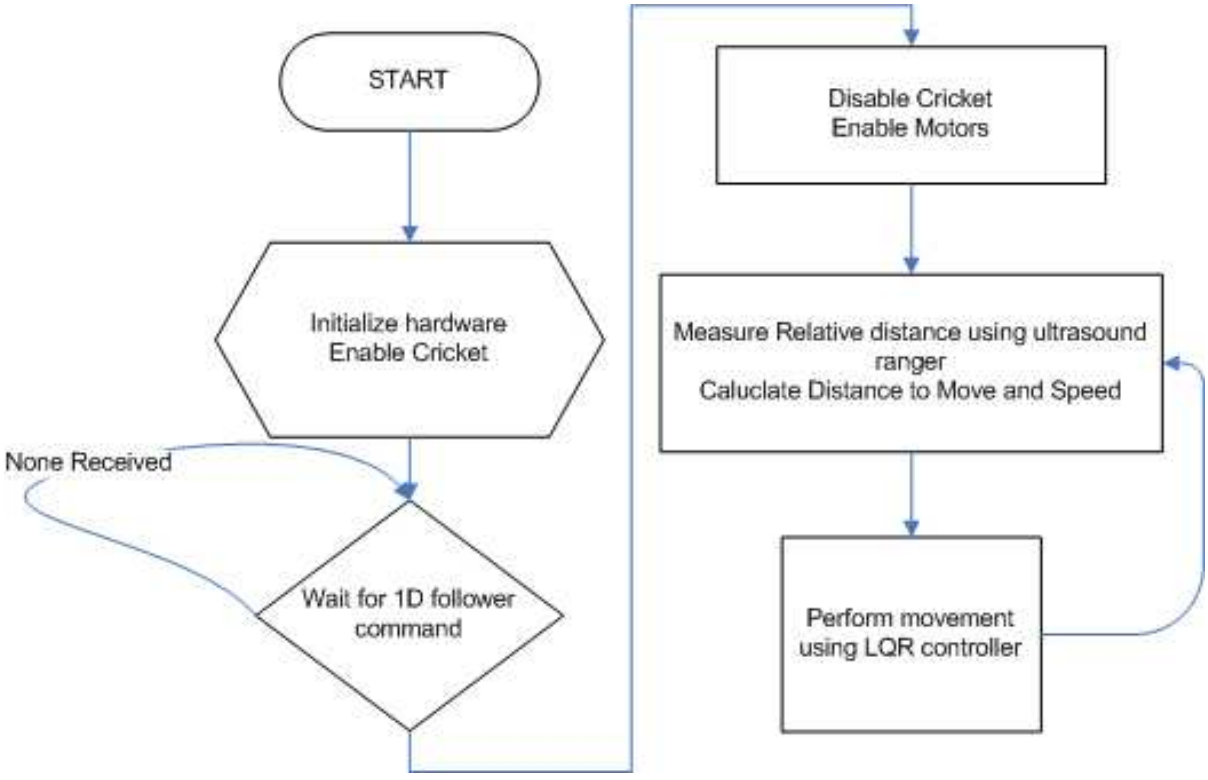

Figure 3.21. 1D Formation Flowchart.

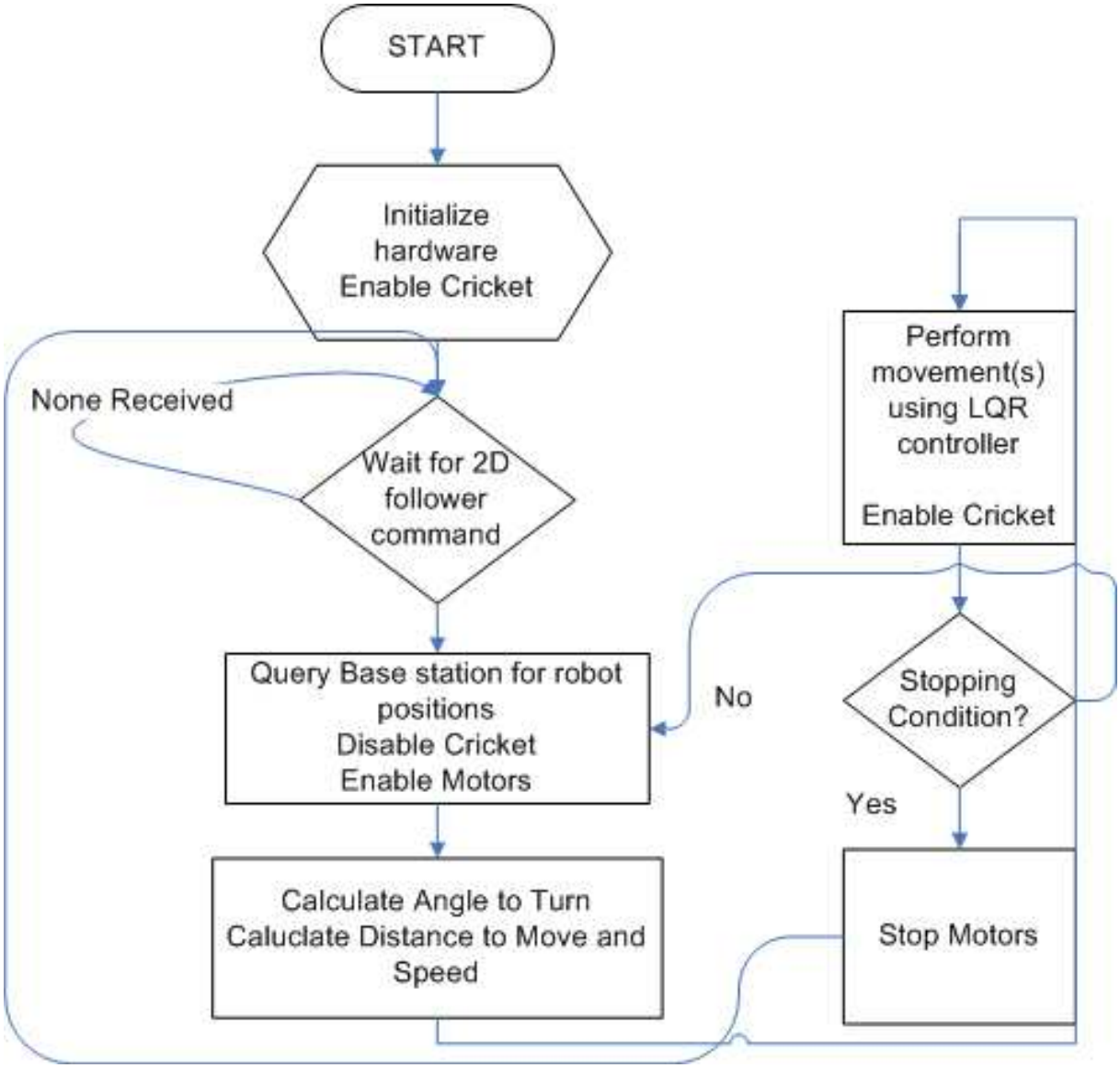

Figure 3.22. 2D Formation Flowchart.

Infrared Camera Localization

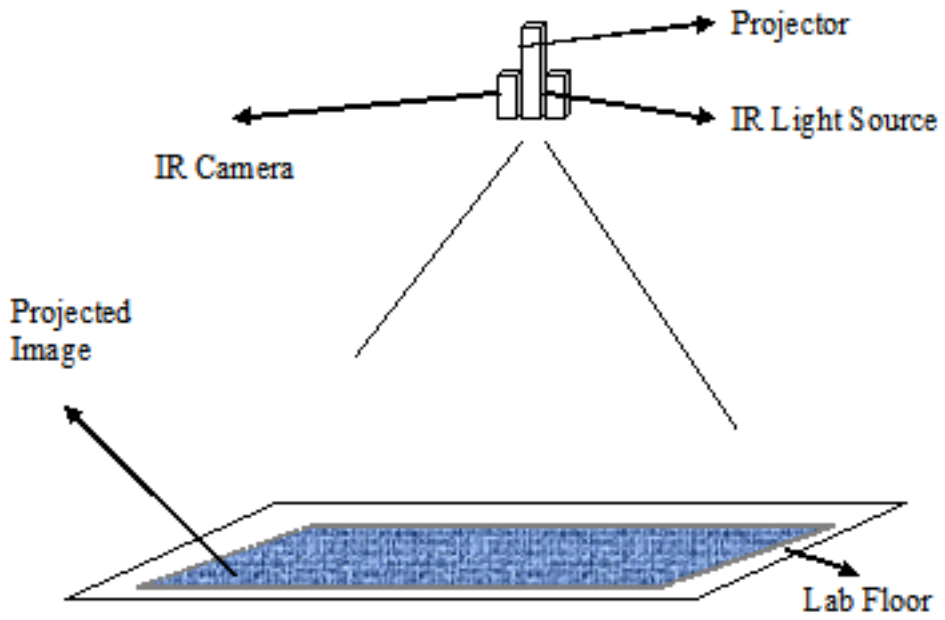

Figure 3.23. Infrared Camera Localization setup.

An infrared camera was used to localize and determine the poses of the ARRIbots. Infrared cameras have the advantage (as opposed to visible light cameras) of filtering out shadows - thus motion capture is easier. Figure 3.23 shows the lab setup. Access to the camera image is obtained in MATLAB through the 'videoinput' object, while the Cricket wireless mote is used through a serial port object.

The projector display was coded in MATLAB. The output of the infrared camera was not perfectly straight but skewed due to the positioning of the mount. This was fixed by introducing a transformation from 'virtual space' within MATLAB to the 'display space' on the lab floor. This transformation was found and implemented and is based upon a polynomial fit of the error along the image. A second transformation was required to correct the camera distortion and skew. This transformation was also found and implemented in a similar manner.

Localization was performed by using the difference in light intensity levels due to the presence of the ARRIbots. In order to aid the process, an infrared absorbing paper

was attached to the top of each robot as seen in 3.24. In addition these papers were cut into isosceles triangles of different sizes for each robot so that orientation could be determined and the the robots could be differentiated easily.

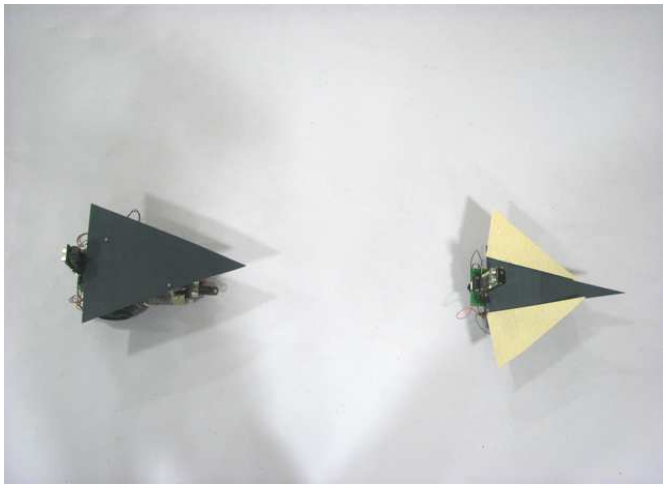

Figure 3.24. Camera Localization robot setup.

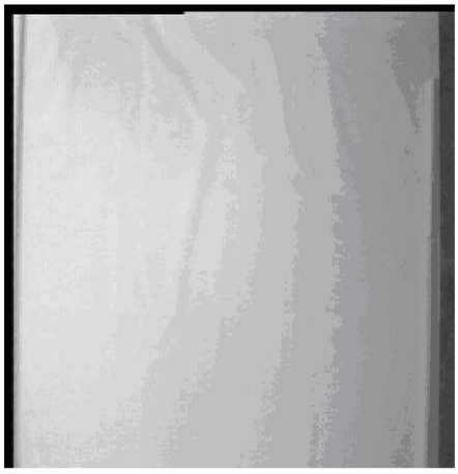

Figure 3.25. Infrared Camera Background image.

First, an image was taken without any robots on the lab floor. This serves as the background image 3.25. In order to detect the position of the robot, the background image is subratacted from the current image 3.26 to obtain a noise free image 3.27.

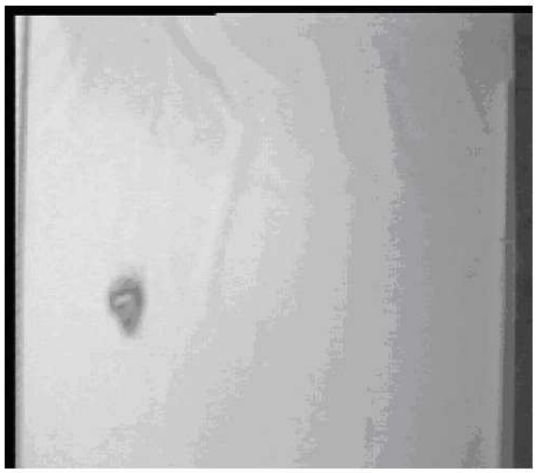

Figure 3.26. Infrared Camera image with robot.

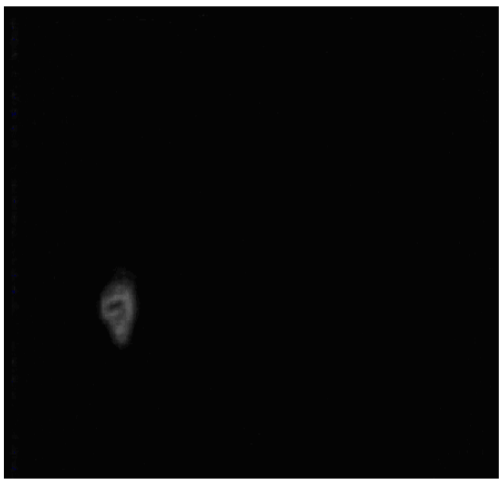

Figure 3.27. Infrared Camera subtracted image with robot.

Since the focus and resolution of the camera is not ideal, the robot outline is not sharp and there may be holes in the picture. Various routines from the MATLAB Image Processing toolbox are used to fill in the holes imfill() and to sharpen the image im2bw().

Finally, the MATLAB image processing function regionprops() was used to get the attributes of the robot's image such as position (in pixels), size, major axis and minor axis. The code automatically filters out objects that are not within a size range that is appropriate for the ARRIbots.

Figure 3.28. Infrared Camera threshold adjusted image.

Orientation was obtained through a routine that measured the distance from the centroid of the shape to all of the outer points and chose the maximum. This works as the shape of the ARRIbots is triangular with the longer sides pointing to the front.

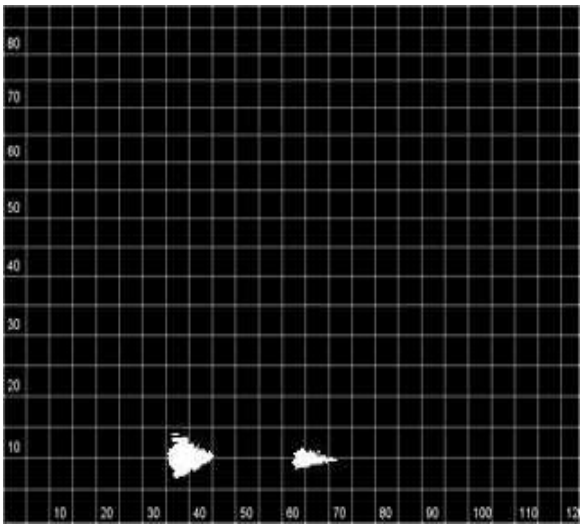

Figure 3.29. Camera Localization Example with 2 robots.

Figure 3.29 shows an example of camera localization of the leader and follower. Table 3.3 shows the errors obtained.

Table 3.3. Camera Localization Errors for Leader, Follower (5 Trials)

Trial	X Actual (in)	Y Actual (in)	Angle Actual φ (^{o})	X Error (in)	Y Error (in)	Angle φ Error(^{o})
Leader,1	65	10	0	1.22	1.94	4.2
Leader,2	65	10	0	1.52	1.32	4.5
Leader,3	65	10	0	1.65	1.52	3.8
Leader,4	65	10	0	1.86	0.36	3.6
Leader,5	65	10	0	1.64	1.45	4.6
Follower,1	35	10	0	1.34	1.85	-6.4
Follower,2	35	10	0	1.47	1.37	-6.31
Follower,3	35	10	0	1.14	1.85	-6.0
Follower,4	35	10	0	1.84	1.85	-6.3
Follower,5	35	10	0	1.33	1.24	-5.5

Cricket Communication Module

It was desired to analyze the data of any particular motion; this required sending back to the base station various data (for each step) such as State estimates, and acknowledgement messages. In addition, the ARRIbot was also required to receive all the various commands such as move, turn, follow leader.

The cricket wireless mote (Figure 2.4) was used for the required wireless communication capability. The cricket was set to forward all messages received from the serial port on the radio. Figure 3.30 shows a flowchart of the cricket operation which was programmed in nesC. This behavior was derived by modifying code already written by previous members of the ARRI DIAL lab [13]. The packet format for cricket commands was chosen as shown in table 3.4. The first row shows the elements of the packet while the second row shows the data size.

Table 3.4. Cricket Packet Format with Byte Sizes

Destination Node ID	:	Command	:	Data	:	...	;\ r \ n
3	1	2 or 3	1	variable	1	variable	3

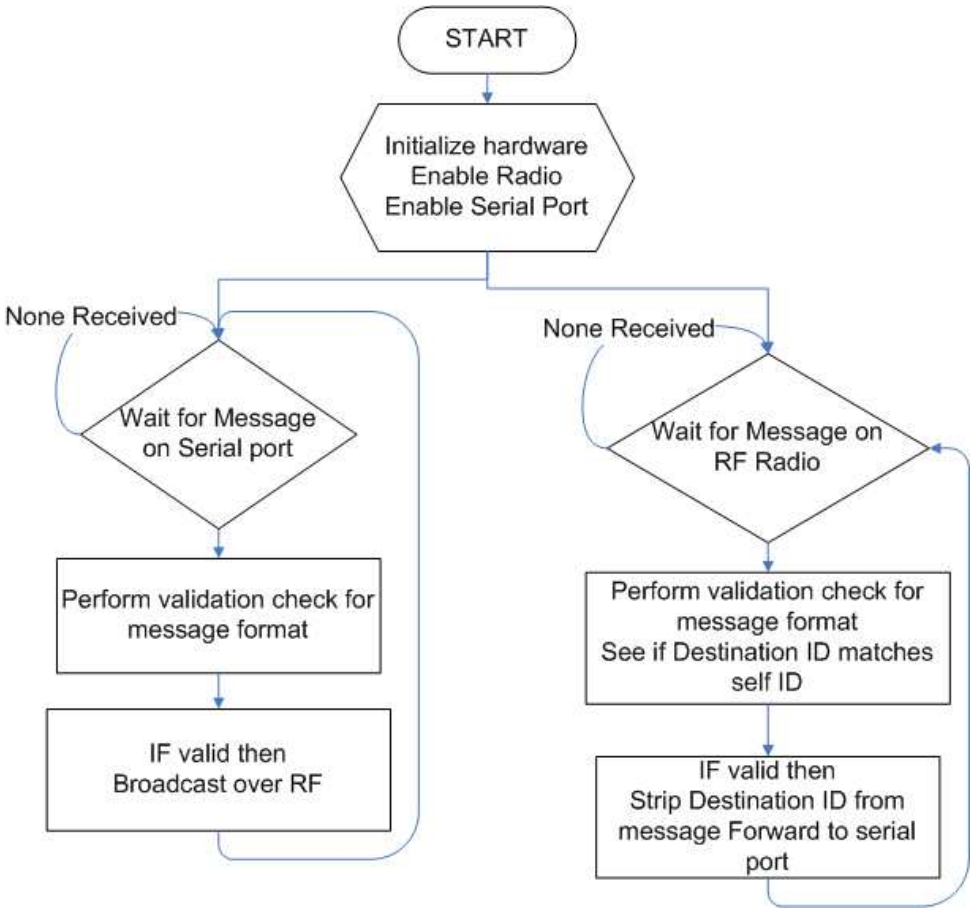

Figure 3.30. Wireless Cricket Module Flowchart.

The ':' char is used as the separator character while the ';' char is used as the terminating value. The maximum packet size is 32 characters.

During testing, it was found that as the distance between nodes increased, packet loss occurred. Thus an acknowledgement scheme for more reliable communications was designed and implemented where:

- All incoming messages have to be acknowledged with a return message.
- A timeout of 0.5s where no acknowledgement is received causes the message to be re-sent.
- After 5 times unsuccessful attempts, a failure to send notification is generated.

CHAPTER 4

SIMULATION RESULTS

4.1 LQR Trajectory Tracking

Simulations were performed with a model of the ARRIbot as given in Equations 2.8 however imperfections were added in order to model the inconsistent parameters of the actual hardware:

- Unequal motor strengths were simulated by a scaling factor of 1.1 added to the right wheel angular velocity input. This caused the robot to veer to the left when given equal left and right wheel velocities.
- The axle length was given less than actual.
- Unequal left and right wheel radii were specified.

Wheel Encoders were not directly modeled, it was assumed that the robot has direct access to the wheel angular velocity. No noise was assumed in the wheel velocity measurement. Note that the ideal model is still used to calculate LQR gains. These LQR gains are applied via the controller in an attempt to regulate the imperfect model.

4.1.1 Arbitrary Path

This simulation was done to investigate the performance of tracking an arbitrary path. In this case, the path was chosen to be part of an ellipse [8].

The configuration matrix Q and the control matrix R used in Equations 3.4 and 3.5 (as discussed in section 3.1.2) have been chosen as:

$$Q = \begin{bmatrix} 1000 & 0 & 0 \\ 0 & 1000 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, h = 0.5s$$

the gain pre-calculating stage gives a continuously time varying gain. The figure below shows the results of the simulation. We can see from Figure 4.1 and 4.2 that the desired trajectory is tracked as well as the orientation.

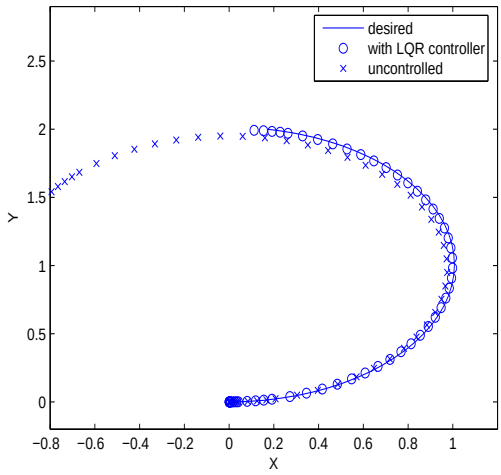

Figure 4.1. LQR Arbitrary path tracking Simulation (Position) [8].

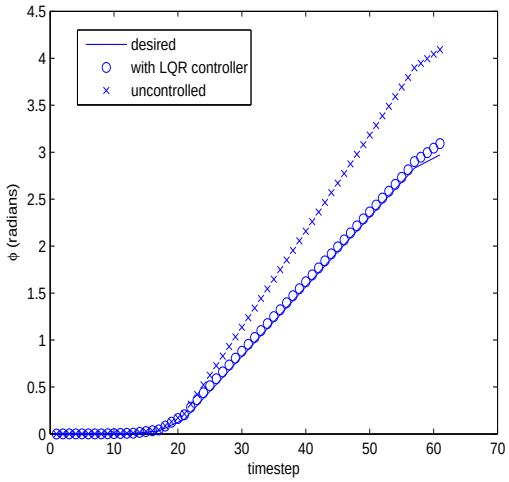

Figure 4.2. LQR Arbitrary path tracking Simulation (Angle) [8].

4.1.2 Straight Line Path

A MATLAB simulation was done to investigate the performance of the LQR controller. Here, we look at the special case of straight line, with a $d_{desired} = 200$ and $v = 10cm/s$. The feed-forward motor velocities were calculated as in Equations 3.44 and 3.45.

As before, with

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100000 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, h = 0.5s$$

the gain pre-calculating stage gives

$$K = \begin{bmatrix} 0.3940 & 0.0160 & 9.1429 \\ 0.3940 & -0.0160 & -9.1429 \end{bmatrix}$$

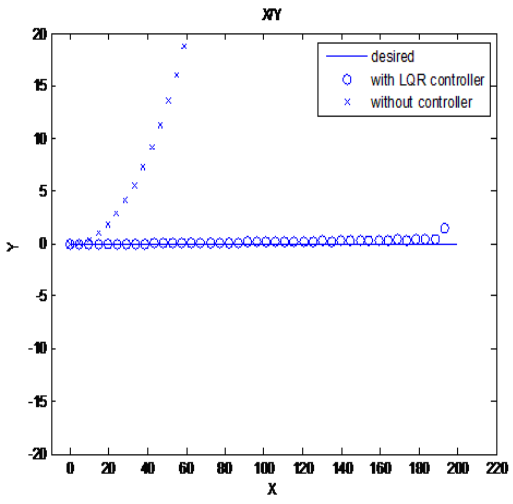

Figure 4.3. LQR Straight-line simulation Position(1).

The figures below show the result of the above configuration. The importance was placed on the [y] state and thus the x position and the phi position is not very accurate.

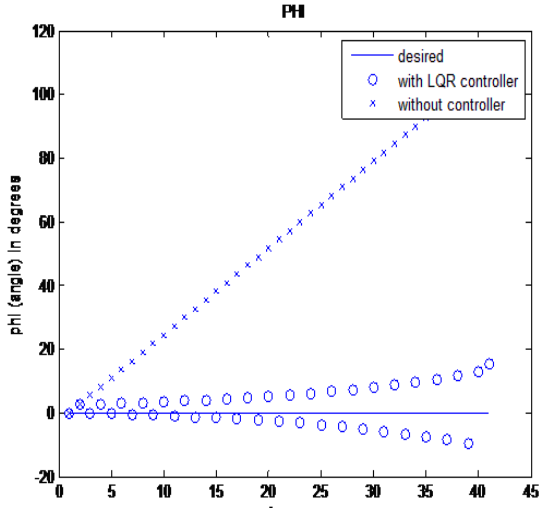

Figure 4.4. LQR Straight-line simulation Orientation(1).

With

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1e9 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, h = 0.5s$$

the gain pre-calculating stage gives

$$K = \begin{bmatrix} 0.3340 & 3.42 & 8.1835 \\ 0.3340 & -3.42 & -8.1835 \end{bmatrix}$$

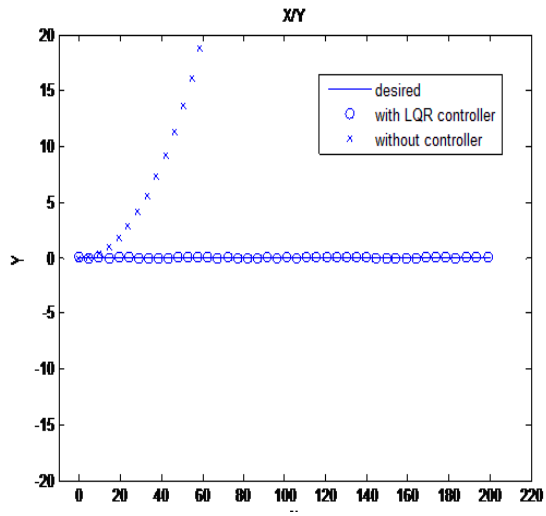

Figure 4.5. LQR Straight-line simulation Position(2).

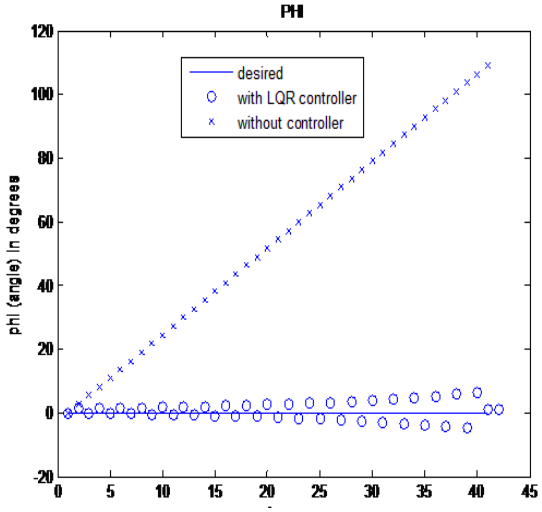

Figure 4.6. LQR Straight-line simulation Orientation(2).

we obtain the results shown in the below figure. We can see that here the gains are chosen properly. The phi angle does not deviate much from 0 (it oscillates) and the x and y are fairly on target.

4.1.3 Axis Turn

Here, we look at the special case of an on-axis turn. We set the desired turning angle of $\angle_{desired} = 700$ and $v = 18 degrees/s$. The feed-forward motor velocities were calculated as in Equations 3.46 and 3.47.

With

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1e9 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, h = 0.5s$$

the gain pre-calculating stage gives a variable K with time. However as mentioned before we ignore the deviation in the x and y and only controll the φ. Thus the first 2 columns of K are set to zero and thus:

$$K = \begin{bmatrix} 0 & 0 & 4.524 \\ 0 & 0 & -4.524 \end{bmatrix}$$

We can see from the figures that the position does not change as required and that the orientation angle is tracked well.

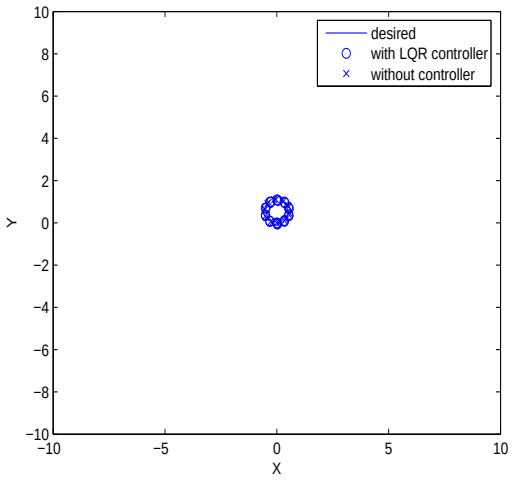

Figure 4.7. LQR Axis Turn simulation (Position).

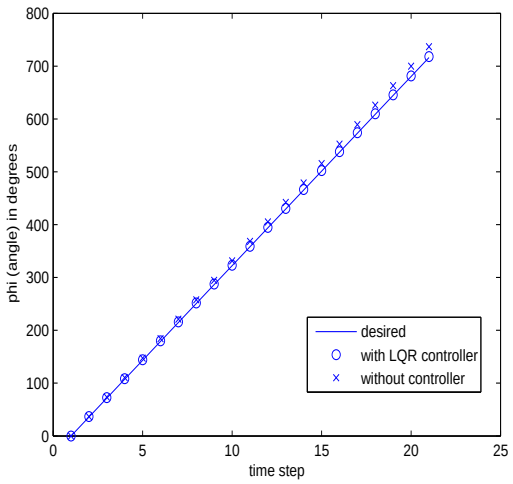

Figure 4.8. LQR Axis Turn simulation (Orientation).

4.2 Formations via Potential Field

Simulations for formation control were performed again with multiple model of the ARRIbot as given in Equations 2.8. The ideal models were used without any added

imperfections. For each of the simulations, a 150s duration was chosen with a 0.1s time-step. The motion of the robots was propagated every time-step..

4.2.1 1 Dimensional Case

A one dimensional follower was simulated using MATLAB. The sampling time for this case is simulated to be every 0.5 second and the actual position values are taken to be the measurement of relative distance. Figure 4.9 shows a leader-follower simulation of 2 ARRIbots while Figure 4.10 shows the formation position error as a function of time.

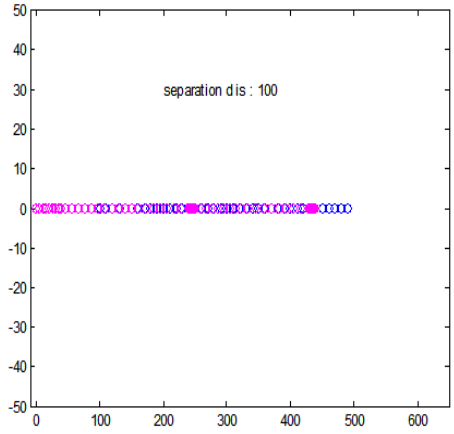

Figure 4.9. One Dimensional Leader-Follower.

4.2.2 2 Dimensional Case

A two dimensional follower was simulated using MATLAB. The sampling time is simulated to be every 1 second and the actual position values are taken to be the measurement of relative distance. Figures 4.11 and 4.12 shows a leader-follower simulation of 3 ARRIbots in a triangle formation.

A similar leader trajectory was simulated however a wall with a narrow gap was introduced as an obstacle. For this case the weight of F_{form} was set to be 1, the weight of $F_{obstacle}$ was set to 100 so that the obstacles were properly avoided. d_{zone} was set to be 5m. The obstacle avoidance artificial force causes the formation to deform in order

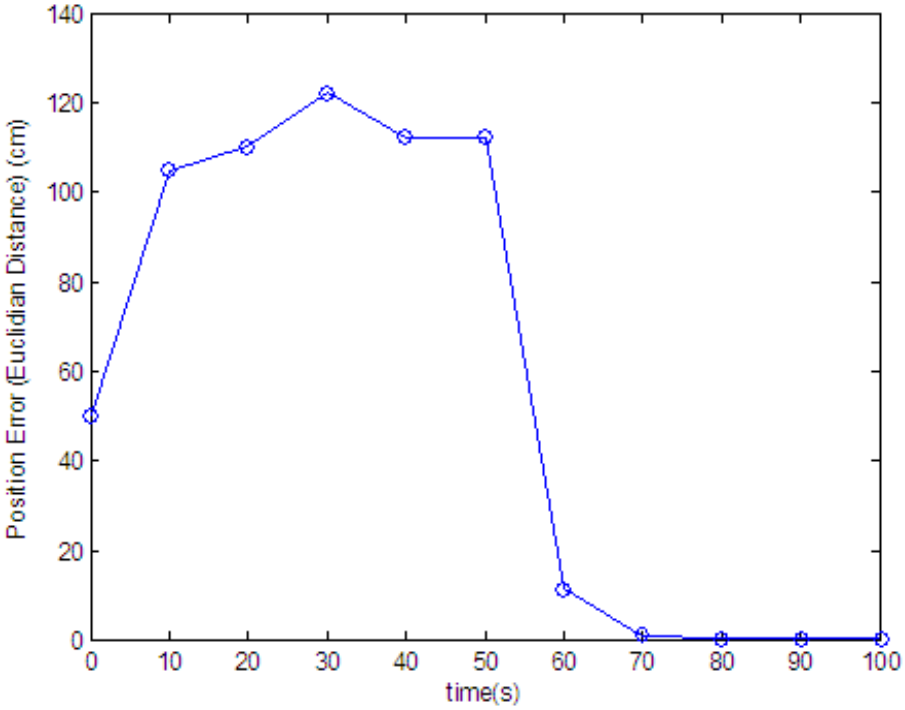

Figure 4.10. One Dimensional Leader-Follower (Position Error).

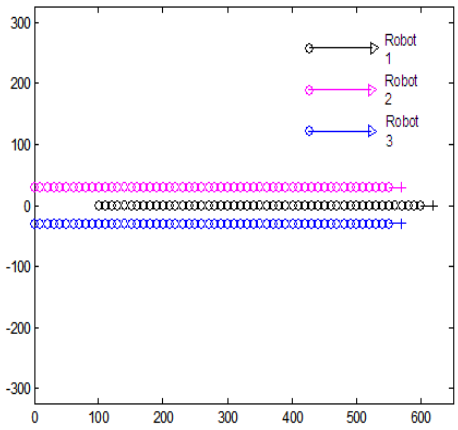

Figure 4.11. 2 Dimensional Formation (Straight Line Triangle).

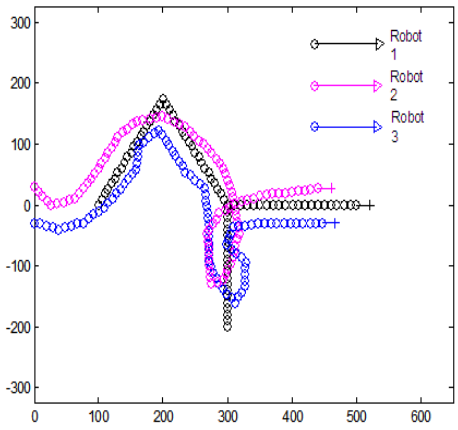

Figure 4.12. 2 Dimensional Formation (Random Path Triangle).

to squeeze through the entrance as we can see in Figures 4.13 and 4.14. Figure 4.15

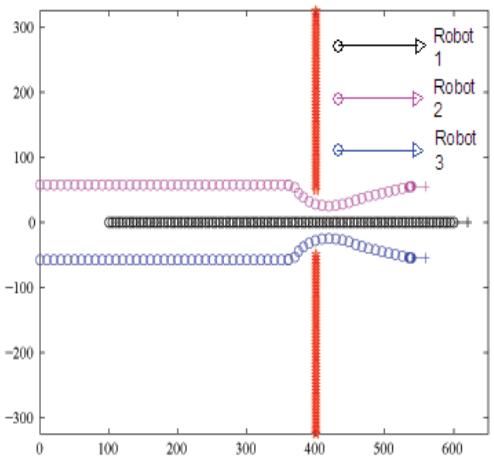

Figure 4.13. 2 Dimensional Formation with Wall Obstacle (Straight Line Triangle).

shows the individial forces that act on Robot 2 at one point of the simulation. We can see a Formation force toward the expected formation position, and repulsive forces from the obstacles, the leader and Robot 1.

Figures 4.16 and 4.17 shows the position and orientation error respectively for both robots as a function of time. The erratic path of the leader causes many spikes in the

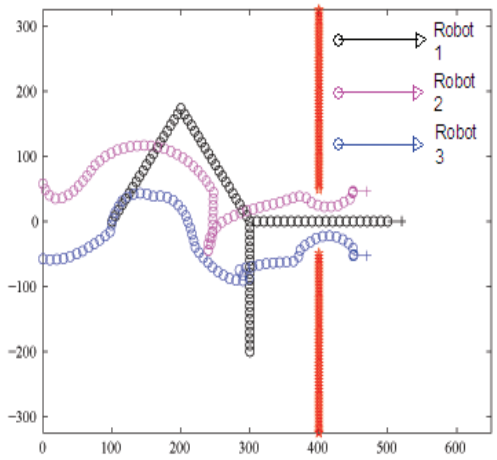

Figure 4.14. 2D Formation with Wall Obstacle (Random Path Triangle).

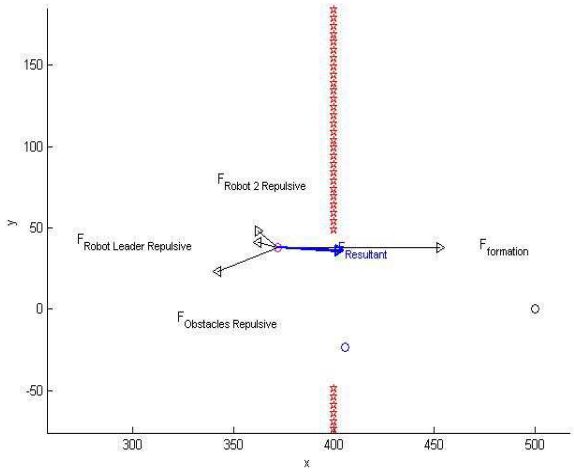

Figure 4.15. 2D Formation with Obstacle - Forces view.

error as do the presence of the wall obstacle. The errors are finally reduced to zero at the end of the simulation.

4.3 Communications Constraint

We simulate how a middle node configuration affects the formation. A diamond formation is chosen with the leader robot as an information source and the follower as the information sink. The middle two robots are pass through nodes for information (middle nodes in the wireless packet route). For this case the weight of F_{form} was set to

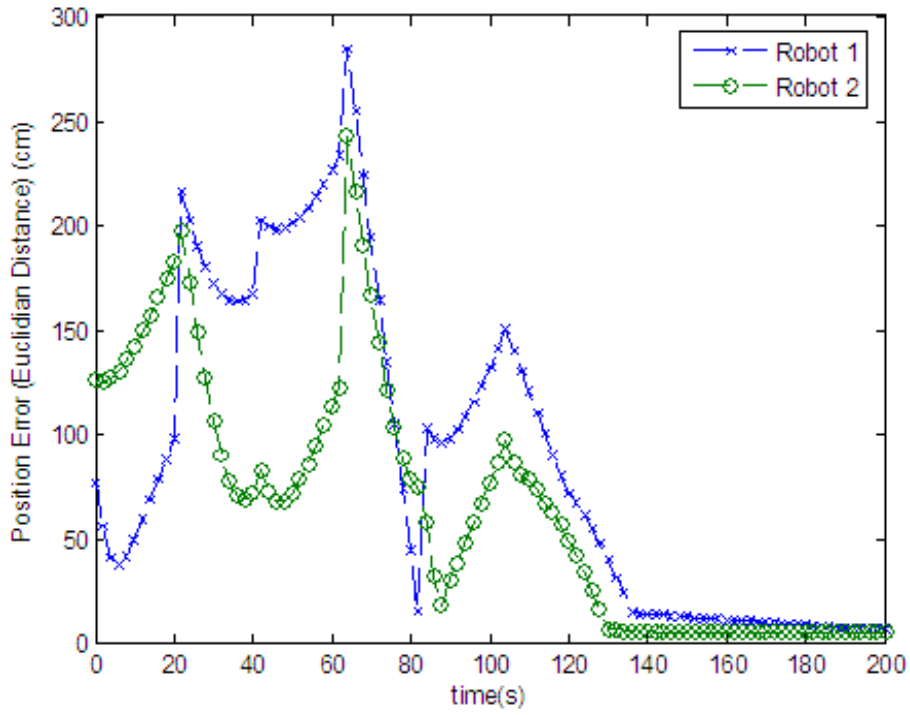

Figure 4.16. 2D Formation with Obstacles - Position Error .

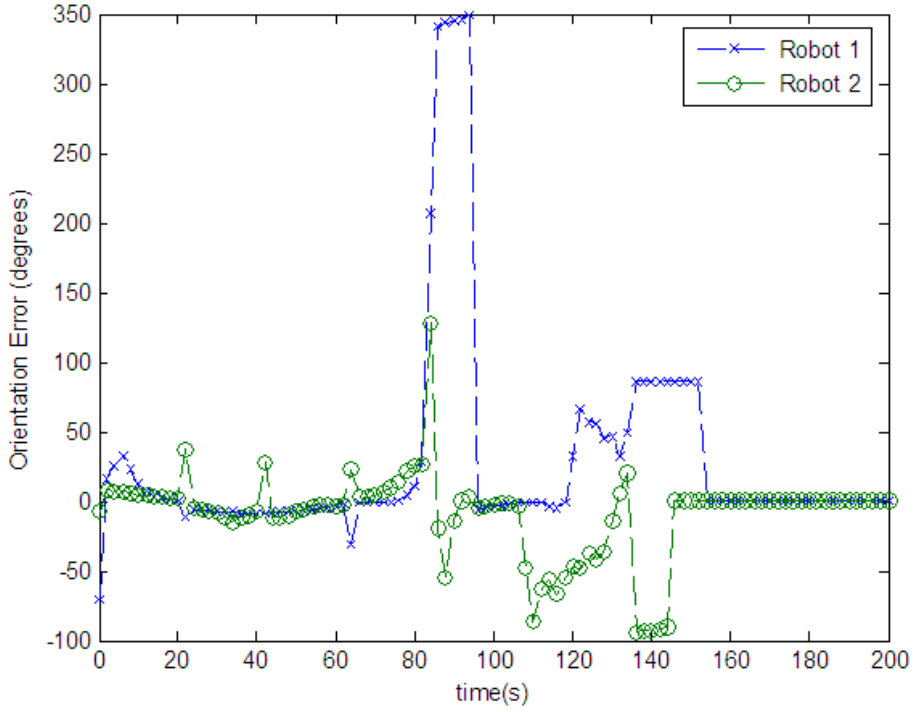

Figure 4.17. 2D Formation with Obstacles - Orientation Error .

be 1 and the weight of $F_{communications}$ was set to $\frac{1}{3}$ so that the communications force was not dominant over the formation force. Figure 4.19 shows the individual forces that

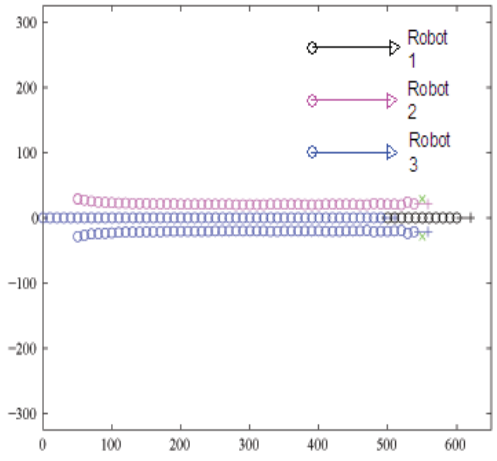

Figure 4.18. 2D Formation with Communications Force (Straight Line Diamond).

act on Robot 2 at one point of the simulation. We can see a Formation force toward the expected formation position and the communications force toward the axis connecting the leader and Robot 3.

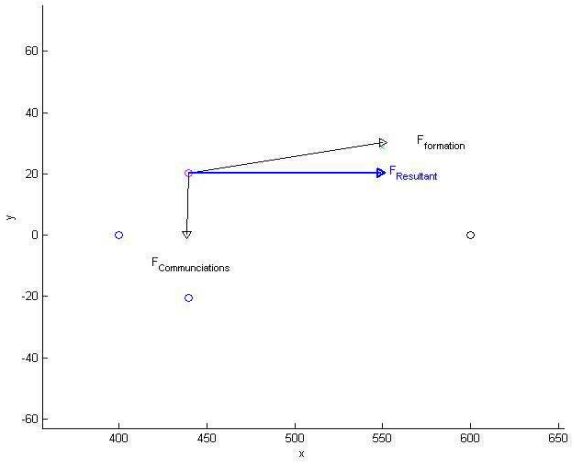

Figure 4.19. 2D Formation with Communications Force - Forces view.

We can see that the formation ideal position (shown by the x) is not achieved by the two middle node robots. This is due to the squeezing effect of the communication force to bring the robots directly in between then end nodes.

4.4 Minefield Sweep Scenario

An more advanced simulation was performed to simulate a mine field detection scenario. The setup consists of 20 robots who must sweep and sample a minefield and return the data wirelessly to a base station as shown in Figure 4.20.

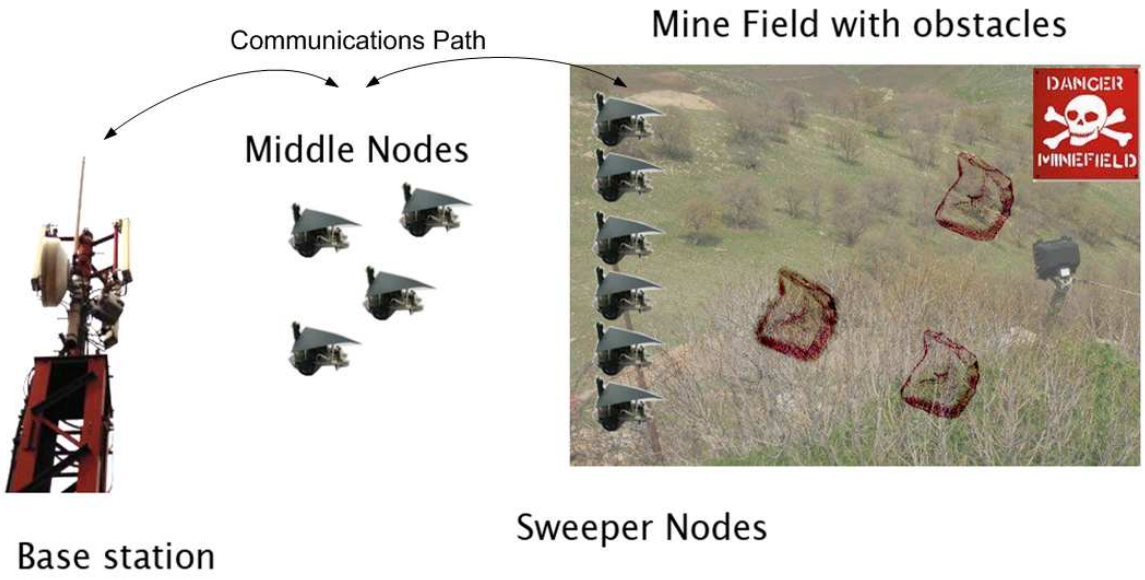

Figure 4.20. Mine Field Sweep Scenario.

The 20 sampling robots are given a horizontal chain formation so that the field can be raster scanned. 6 obstacles in the terrain exist, the robots must navigate around these obstacles.

The base station is considered to be sufficiently far from the sampling field that intermediate 'hop' or middle nodes are required. Thus, several intermediate middle nodes are placed in between the base and the field. These must continuously change their formation to optimize the use of the communication channels. The aim is to demonstrate a combination of geometric and communication centric formation control to accomplish a real-world mission.

4.4.1 Sampling Robots

The 20 Sampling ARRIbots were again modeled using an ideal two wheeled differential drive robots along with the ARRIbot specific parameters such as wheel radius and axle length.

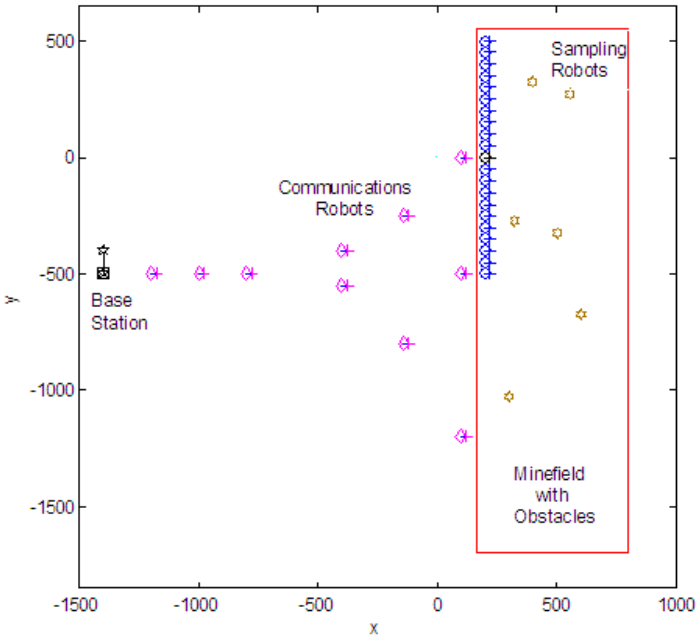

Figure 4.21. Mine Field Detection MATLAB Simulation.

The robots are required to raster sample a field measuring $63m \times 115m$ (Figure 4.21) while simultaneously avoiding the obstacles in their path. A formation consisting of one leader and 20 followers was used and 6 obstacles (modeled as point obstacles with a range of influence d_0 as in Section 3.2.3.2). The formation chosen was a horizontal chain formation with 10 robots to the left of the leader and 10 to the right each spaced $5m$ apart. The leader trajectory was chosen to be a multi-step 'U' shape with 5 steps (Figure 4.22):

- Straight path of $45m$
- Right turn of 90^o

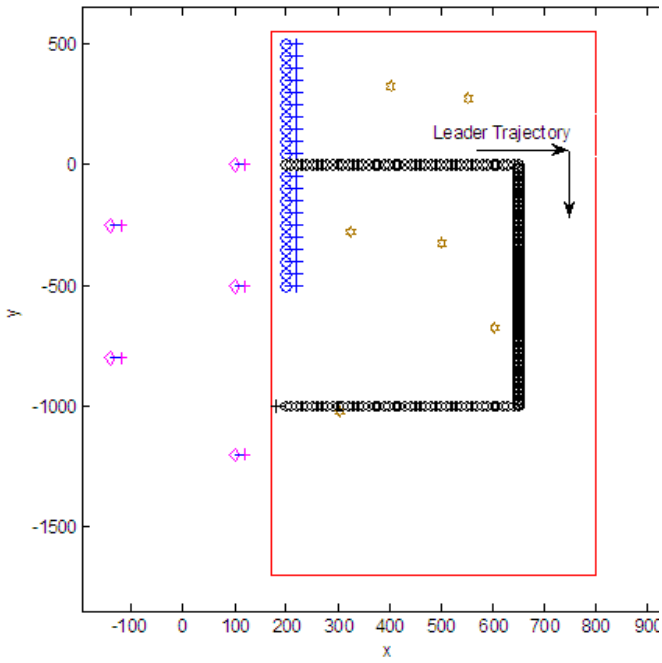

Figure 4.22. Mine Field Detection - Leader Trajectory.

- Straight path $100m$
- Right turn of 90^o
- Straight path of $45m$

Note that the sampling robots were not used as hop nodes themselves. This is valid extension and should be considered in future work.

4.4.1.1 Dynamic formation switching

After the robots have swept the upper half of the field, they are required to turn and move toward the bottom in order to sweep the lower half. However the sudden change in orientation of the leader (right turn of 90^o) at this point will cause a major disturbance as the robots try to keep in formation. Thus at this point, a new formation is enforced - a vertical chain. A vertical chain formation is similar to the the horizontal chain except that robots are 'back-to-back' as opposed to 'side-by-side' (Figure 4.23).

Switching to this formation keeps the robots in a vertical line and allows for a smooth motion for the formation.

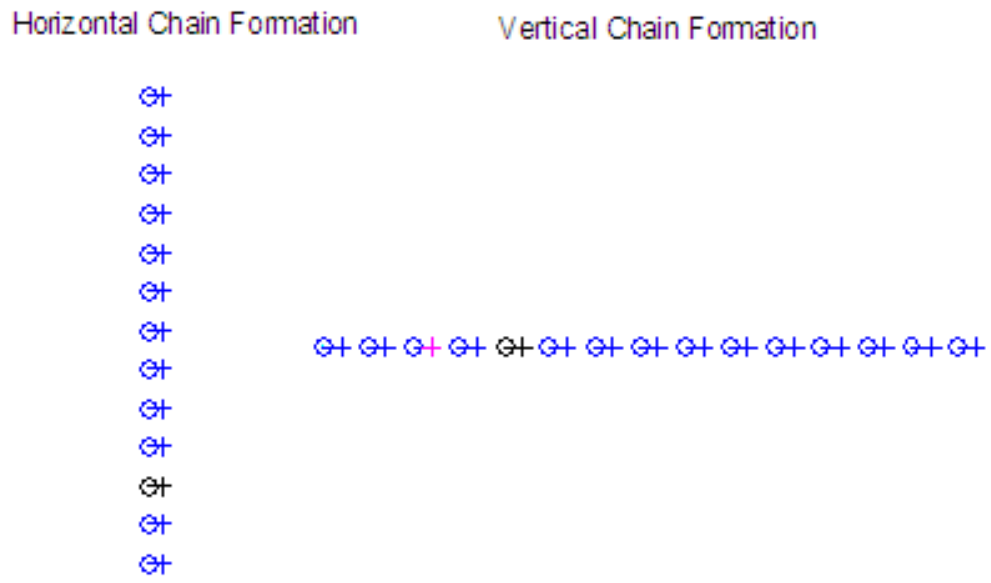

Figure 4.23. Chain Formations.

Similarly, when the robots have reached the lower half, a 'side-by-side' formation is again required. Thus at this point, the formation is switched back to the horizontal chain and the sweeping continues.

This demonstrates the ability to specify formations dynamically. The only information required by individual follower robots is the new position with respect to the leader in the formation reference frame T as described in Section 3.2.1.

4.4.2 Communication Robots

10 Communications robots are used in order to transmit sampled data back to the base station. The Communications Robots are required to:

- Maintain a multi-hop link between all sampling nodes and the base station
- Maintain a spread formation.

- Change position in order to optimize the utility of the communications channels along currently used communications routes.

The formation used is shown in Figure 4.21. This particular formation was chosen so that most parts of the field were in range of at least one communication robot and at the staring position, all sampling nodes are within range of at least one communication robot.

The formation was maintained using the formation control algorithm developed in 3.2. Thus each robot experience an artificial attractive F_{form} that pulled it toward it's formation position. By choosing a strong weight for this force, the communication nodes can be forced to remain within communications range of each other and the base station.

A second artificial force on each communications robot was $F_{communications}$. This force is as defined in section 3.2.3 and pushes the robot toward it's ideal middle node configuration location. $U_{communications}$ was used as a lookup table directly from C. Helm's work [8]. The gradient of this along with an appropriate weighting value was used to obtain the force.

For this simulation, a weight of 1 was used for F_{form} while a weight of $\frac{1}{3}$ was used for $F_{communications}$. The two were vectorially added to give the resultant force on the communication robots.

Figures 4.24, 4.25 and 4.26 show the simulation at various stages. We can observe that

- The field is sampled as required (each sample is shown by an 'x').
- The sampling robots successfully avoid obstacles.
- Nodes that are not part of any communications routing path are attracted back to their formation position.
- Even though the original placement of the communications nodes did not cover the entire field, the $F_{communications}$ has moved some of the nodes so that this is accomplished.
- There is always a path from all source nodes (the sampling robots) to the sink (the base station).

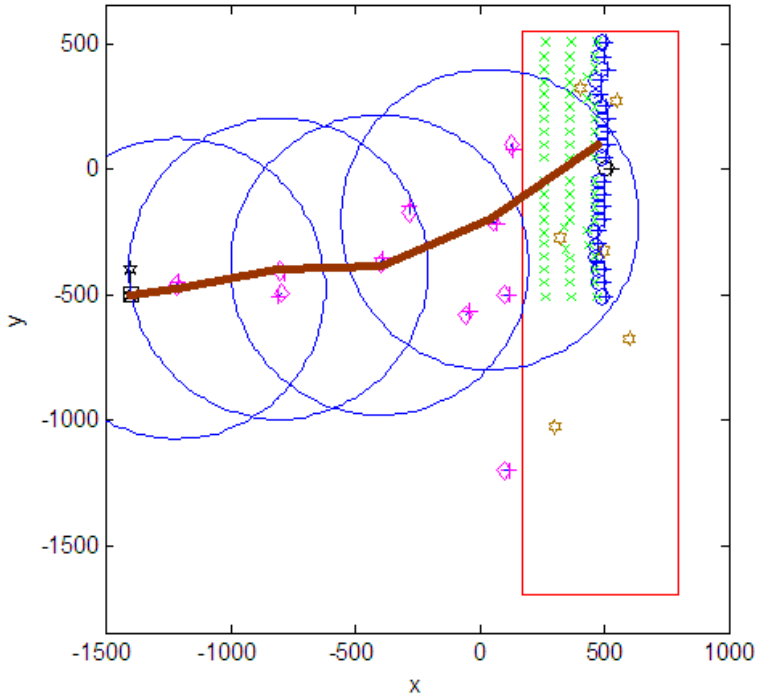

Figure 4.24. Mine Field Detection Simulation - Result (1/3).

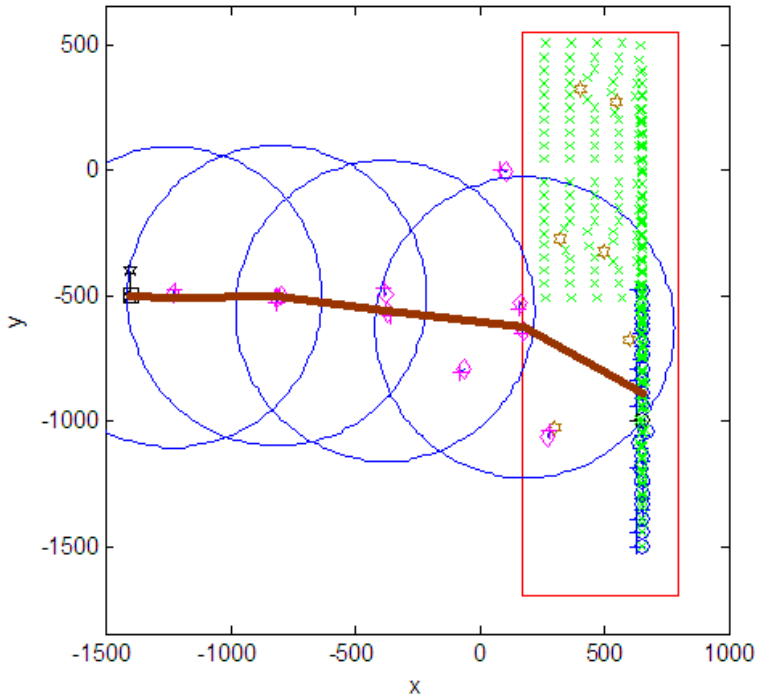

Figure 4.25. Mine Field Detection Simulation - Result (2/3).

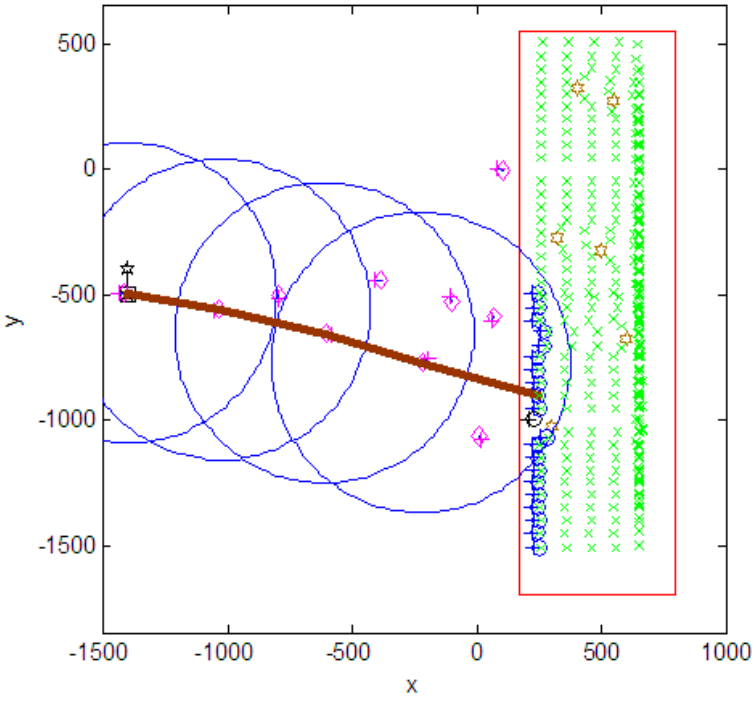

Figure 4.26. Mine Field Detection Simulation - Result (3/3).

CHAPTER 5

EXPERIMENTAL RESULTS

5.1 LQR Trajectory Tracking

An experimantal setup was chosen consisting of the ARRIbot running the LQR trajectory tracking controller along with wireless communications through the cricket and base station. The ARRIbot was given commands to move and turn through various values and the motion was captured by the camera. A comparison was made between desired, actual and estimated values for the robot state.

5.1.1 Straight Line Path

Table 5.1 shows the results for the straight line trials with the ARRIbot. We can see that the end pose is very close to the desired distance with small errors in X, Y and φ.

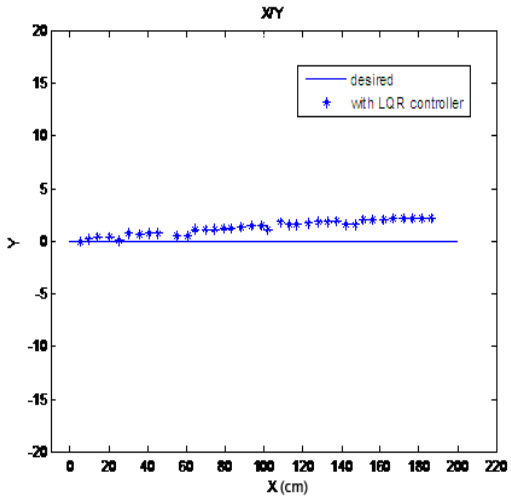

Figure 5.1. LQR Straight-line experimental Position(1).

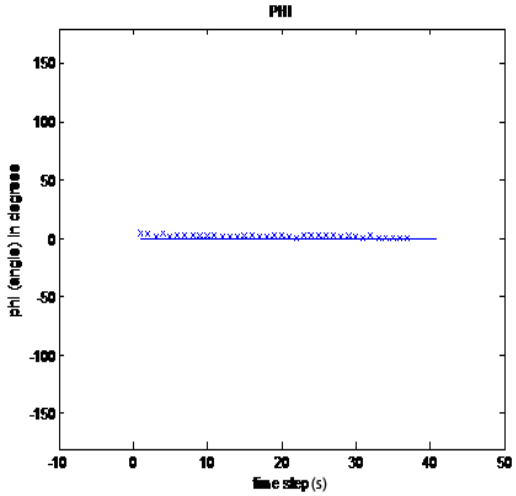

Figure 5.2. LQR Straight-line experimental Orientation(1).

The figures 5.1 and 5.1 show the result of the straight line trial for $d_{desired} = 200$ and $v = 10cm/s$. The trials were conducted with the straight line gain from the simulation phase:

$$K = \begin{bmatrix} 0.3340 & 3.42 & 8.1835 \\ 0.3340 & -3.42 & -8.1835 \end{bmatrix}$$

The figures 5.3 and 5.3 show the result of the straight line trial for $d_{desired} = 100$ and $v = 10cm/s$. Again the same K was used from the simulation phase:

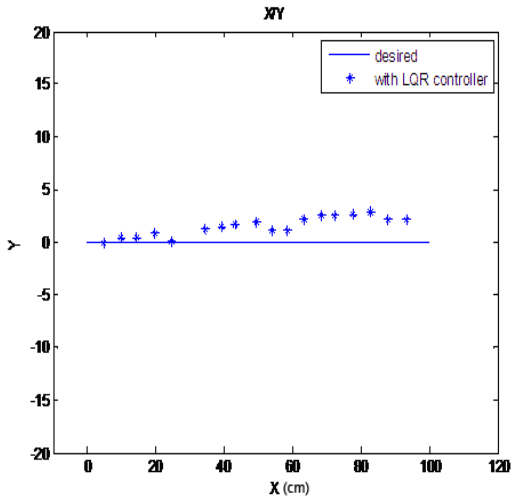

Figure 5.3. LQR Straight-line experimental Position(2).

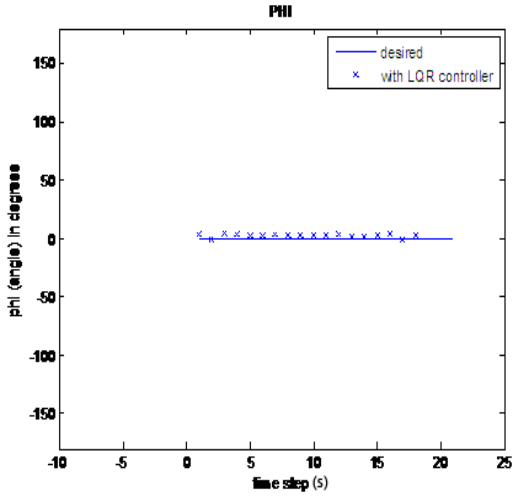

Figure 5.4. LQR Straight-line experimental Orientation(2).

We can see that here the gains are chosen properly. The phi angle does not deviate much from 0 (it oscillates) and the x and y are fairly on target.

5.1.2 Axis Turn

As in the simulation, the gain pre-calculating stage gives a variable K with time and we ignore the deviation in the x and y and only control the φ. Thus:

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1e9 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, h = 0.5s$$

$$K = \begin{bmatrix} 0 & 0 & 4.524 \\ 0 & 0 & -4.524 \end{bmatrix}$$

A large value of $1e9$ for the configuration matrix was chosen so that the desired φ angle was tracked as closely as possible. The x and y diagonal terms are zero as we wish to ignore those errors. This is justified because the duration of axis turn motions are very small and thus the errors in x and y also remain small. Also in terms of robot localization, an error in orientation is much worse than one in position as the dead-reckoning error grows very fast when long distances concerned.

Table 5.1. Straight Line Motion Results

Distance Desired (along X) (cm)	Actual Final X Position (cm)	Actual Final Y Position (cm)	Actual Final φ Position (^{o})	Final X Estimate (cm)	Final Y Estimate (cm)	Final φ Estimate (^{o})	X Error(Desired - Actual) (cm)	X Error(Desired - Estimated) (cm)	X Error(Actual - Estimated) (cm)
50	52.4	2.2	5	48.543	1.378	5.532	-2.40	1.46	3.86
50	54.2	4.8	5	47.372	0.285	6.253	-4.20	2.63	6.83
100	103.4	3.3	5	97.342	2.457	5.453	-3.40	2.66	6.06
100	106.2	5.7	4	98.264	1.456	4.763	-6.20	1.74	7.94
200	198.1	4.4	4	195.368	2.356	3.782	1.90	4.63	2.73
200	193.8	3.2	4	194.385	2.785	3.329	6.20	5.62	-0.58

Table 5.2. Axis Turn Motion Results

Angle φ Desired (^{o})	Actual Final X Position (cm)	Actual Final Y Position (cm)	Actual Final φ Position (^{o})	Final X Estimate (cm)	Final Y Estimate (cm)	Final φ Estimate (^{o})	φ Error(Desired - Actual) (^{o})	φ Error(Desired - Estimated) (^{o})	φ Error(Actual - Estimated) (^{o})
45	2.4	-1.2	42	2.294	-2.735	41.248	3	3.752	0.752
45	2.2	-1.8	46	2.383	-2.528	43.568	-1	1.432	2.432
90	3.4	-3.2	95	4.176	-3.295	94.622	-5	-4.622	0.378
90	3.5	-5.5	97	3.732	-4.542	96.348	-7	-6.348	0.652
180	4.1	-4.2	185	4.637	-3.235	188.652	-5	-8.652	-3.652
180	4.8	-4.9	185	7.3245	-5.272	185.322	-5	-5.322	-0.322
-45	-3.5	2.1	-47	-4.293	3.145	-46.451	2	1.451	-0.549
-45	-3.2	2.3	-48	-3.826	2.956	-47.154	3	2.154	-0.846
-90	-5.2	4.2	-95	-5.452	5.237	-96.235	5	6.235	1.235
-90	-3.7	5.5	-96	-4.348	5.732	-98.197	6	8.197	2.197

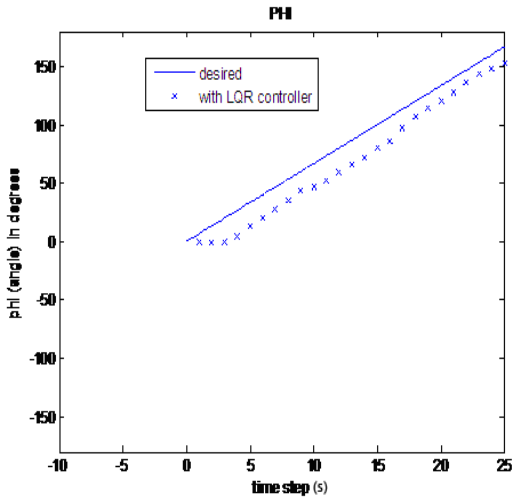

Figure 5.5. LQR Axis Turn Experimental Position(1).

We can see from figures 5.5 (180^o turn) and 5.6 (90^o turn) that the angle is tracked as required.

5.1.3 Multi-Step Square Trajectory

In order to test the accuracy of the ARRIbot straight line and turning, a square pattern was chosen. This is a multi-step trajectory with 4 straight line paths of 40 inches each and 4 90^o turns. Three measurements were taken - desired path, actual path and estimated path. The estimated path was generated using the state estimator in the LQR algorithm and transmitted back to the base station via the cricket wireless communications link.

Figure 5.7 shows a trial run without 'on the fly' adjustment of the path. The ARRIbot was simply commanded to execute the 8 steps in order as if it perfectly executed all prior steps i.e. dead reckoning was not used to adjust the next step. Tables 5.3, 5.4 and 5.5 show the errors obtained. The errors are quite substantial, this shows that for multiple steps, the tracking ability of the LQR controller alone cannot be relied upon.

Figure 5.8 shows a trial run with 'on the fly' adjustment of the path. The ARRIbot was simply commanded to execute the 8 steps in order as if it perfectly executed all prior

Table 5.3. Multiple step square trajectory error results (without dead-reckoning adjustment) (Desired - Actual)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	0	0	0.4855
1	-3.0384	-3.0783	-3.0050
2	-3.7726	-1.1108	-5.2762
3	1.3038	-0.6907	-1.7235
4	2.0075	-2.4754	-3.1366
5	3.2171	3.9882	-7.5226
6	3.5869	4.9131	-2.8966
7	-4.2512	4.0559	-4.8076
8	-4.5199	3.8929	-9.3196

Table 5.4. Multiple step square trajectory error results (without dead-reckoning adjustment) (Actual - Estimated)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	0.1623	-0.1182	-0.1915
1	-1.8727	0.8691	1.7585
2	0.2620	0.7839	2.0882
3	-3.9346	-2.0095	-1.7438
4	-3.1346	-0.9241	-5.0995
5	-2.0299	1.9339	-2.2237
6	-1.9898	1.4000	-2.8385
7	-2.4053	-2.0072	-3.9918
8	-2.2766	-0.7350	-3.6512

Table 5.5. Multiple step square trajectory error results (without dead-reckoning adjustment) (Desired - Estimated)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	-0.0178	0.3239	-0.0398
1	-5.4018	-1.9838	-1.4385
2	-4.7832	-1.2162	-3.1145
3	-2.1962	-5.2249	-5.1955
4	-1.1377	-3.6529	-10.0894
5	2.2679	6.3875	-8.7836
6	1.3182	5.8270	-5.5255
7	-6.5826	1.7488	-7.9541
8	-7.4012	4.2573	-13.5533

steps i.e. dead reckoning was not used to adjust the next step. Tables 5.6, 5.7 and 5.8 show the errors obtained. The errors can be reduced to almost a constant value after each step, this shows that for multiple steps, the dead-reckoning estimate is accurate and can be used to correct for the tracking errors.

Table 5.6. Multiple step square trajectory error results (with dead-reckoning adjustment)(Desired - Actual)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	0.0537	0.2762	0.0443
1	-3.8750	-1.7360	-4.2749
2	-2.2095	-3.1819	-1.1679
3	2.3071	-3.4611	-2.0417
4	1.0869	-1.6423	-2.8488
5	1.6166	2.0782	-5.9249
6	3.0940	4.0041	-3.2691
7	-1.7451	1.7188	-5.7036
8	-1.3723	1.1930	-4.9935

Table 5.7. Multiple step square trajectory error results (with dead-reckoning adjustment) (Actual - Estimated)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	0.1623	-0.1182	-0.1915
1	-1.8727	0.8691	1.7585
2	0.2620	0.7839	2.0882
3	-3.9346	-2.0095	-1.7438
4	-3.1346	-0.9241	-5.0995
5	-2.0299	1.9339	-2.2237
6	-1.9898	1.4000	-2.8385
7	-2.4053	-2.0072	-3.9918
8	-2.2766	-0.7350	-3.6512

Table 5.8. Multiple step square trajectory error results (with dead-reckoning adjustment) (Desired - Estimated)

Step No.	X Position (cm) Error	Y Position (cm) Error	Angle φ Error(^{o})
initial	-0.0178	0.3239	-0.0398
1	-5.4018	-1.9838	-1.4385
2	-4.7832	-1.2162	-3.1145
3	-2.1962	-5.2249	-5.1955
4	-1.1377	-3.6529	-10.0894
5	2.2679	6.3875	-8.7836
6	1.3182	5.8270	-5.5255
7	-6.5826	1.7488	-7.9541
8	-7.4012	4.2573	-13.5533

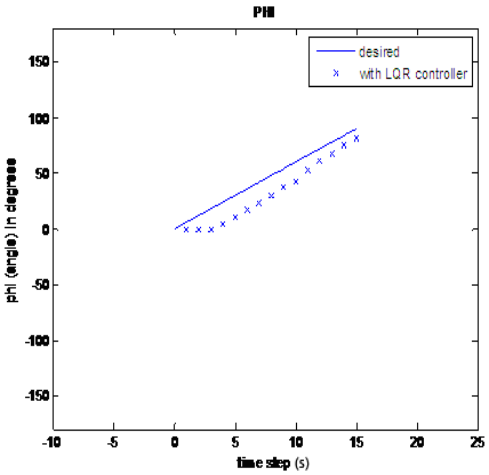

Figure 5.6. LQR Axis Turn Experimental Orientation(1).

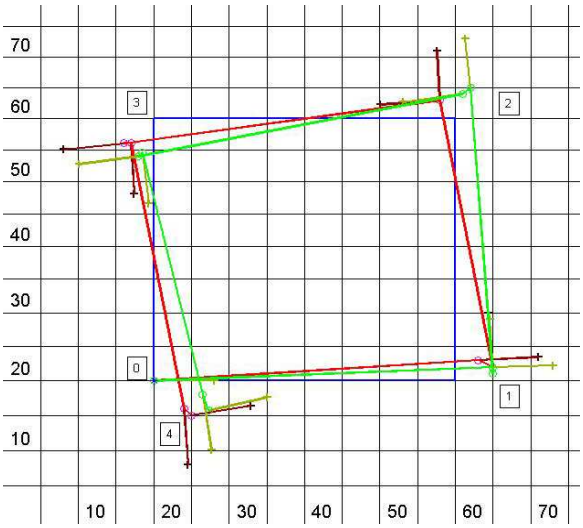

Figure 5.7. LQR Axis Turn Experimental Position(2).

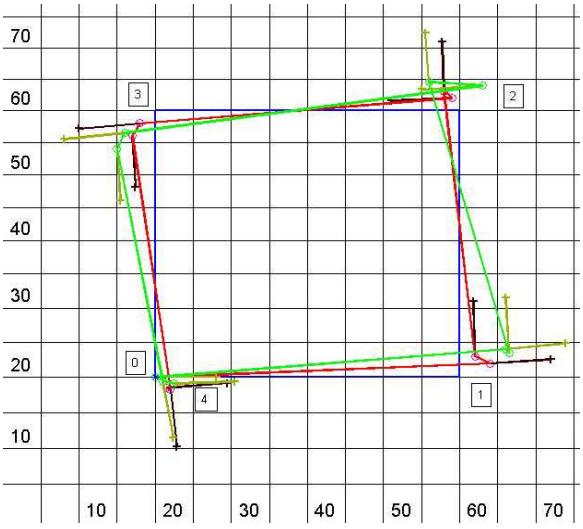

Figure 5.8. LQR Axis Turn Experimental Orientation(2).

5.2 Formations via Potential Field

A two dimensional follower was tested using the DIAL lab setup. A test-bed composed of an overhead infrared camera, projector and base-station was created in order to test the formations. The ARRIbots were programmed and the algorithm was distributed to the the individual nodes with communication with the base station necessary only for sensing other robot poses. The sampling time is 3 seconds and the measurement of relative distance and angle are found using the overhead infrared cameras. The time for taking a measurement was $> 1s$. This was due to repeated passes being required to localize all robots in case of over exposed camera images. The formation used was a chain formation with two robots. The distance of separation was 30 inches. Three leader trajectories were tested - straight line, curved and discrete step.

Straight line Leader Trajectory

Table 5.9. Straight line Final Pose Error Results

Trial No.	X Position (in) Error	Y Position (in) Error	Angle φ Error$(^o)$
1	-5	-2.5	-3
2	-4.5	3	-5
3	2.1	6.2	-2
4	5.1	3.1	-6
5	-5.5	-4.5	-8

Figure 5.9 shows the time lapse positions of the robots where the leader was given a straight line path of 150cm. Figure 5.10 shows the actual test setup. Table 5.9 shows the errors in x, y, φ for 5 test runs. We can see that the final pose of the follower is very close to the ideal position.

Figure 5.11 shows the position error vs. time for the straight line formation. The error grows initially because of the delay in the sensor measurement (camera), however

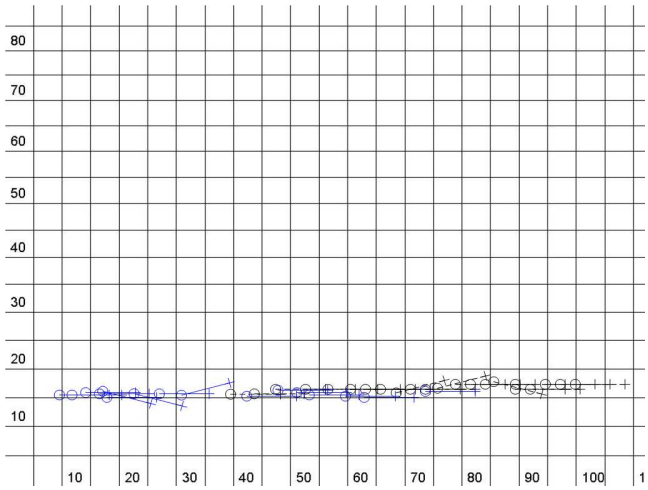

Figure 5.9. 2D Formation - Straight Line.

Figure 5.10. 2D Formation of ARRIbots - Test Setup.

it is reduced to 6.3in by the end. Similarly, Figure 5.12 shows the orientation error vs. time. We can see that it remains small and is -3^o at the end.

Curved Leader Trajectory

Figure 5.13 shows the time lapse positions of the robots where the leader was given a curved path of 200cm.Table 5.10 shows the errors in x, y, φ for 5 test runs. We can see that the final pose of the follower is very close to the ideal position.

Figure 5.14 shows the position error vs. time for the straight line formation. The error grows initially because of the delay in the sensor measurement (camera), however

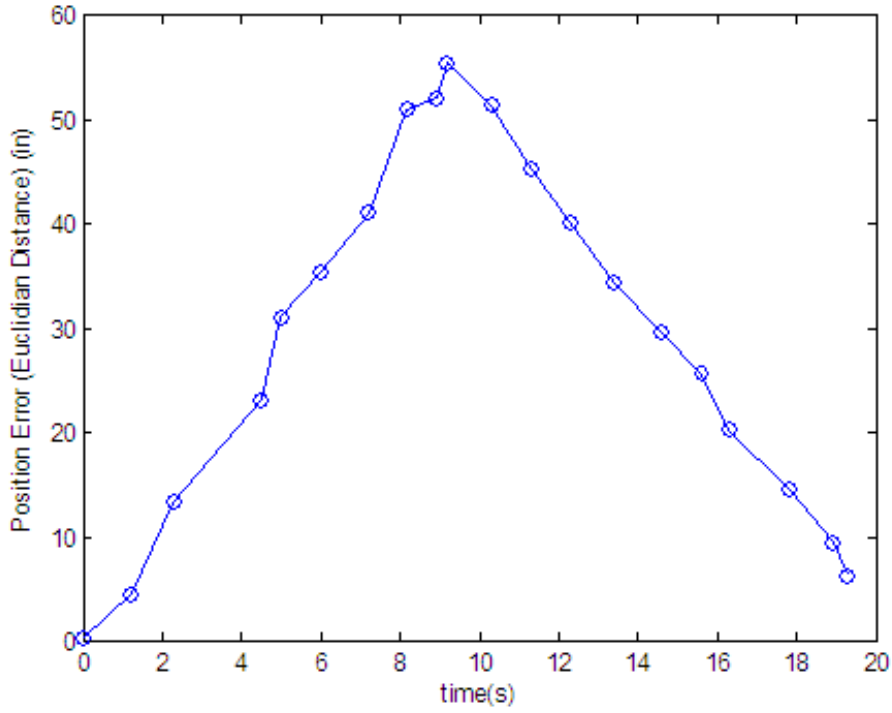

Figure 5.11. 2D Formation - Straight Line Position Error.

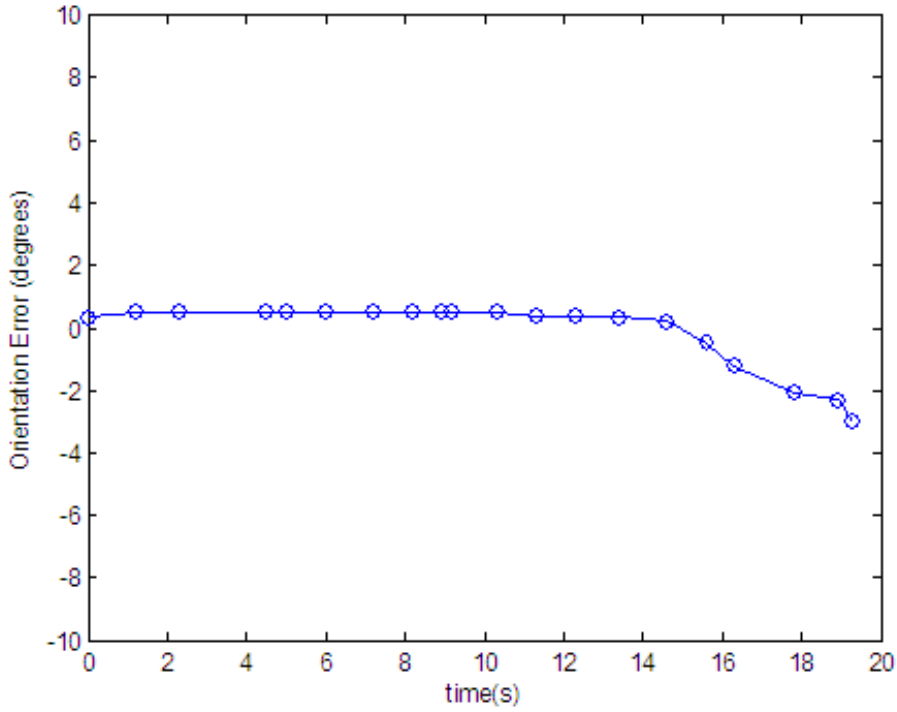

Figure 5.12. 2D Formation - Straight Line Orientation Error.

Table 5.10. Curved Line Final Pose Error Results

Trial No.	X Position (in) Error	Y Position (in) Error	Angle φ Error$(^o)$
1	3	5.2	-4
2	-2.4	-3.4	-11
3	4.2	-4.8	8
4	1.4	2.8	-6
5	-4.6	-5.2	-7

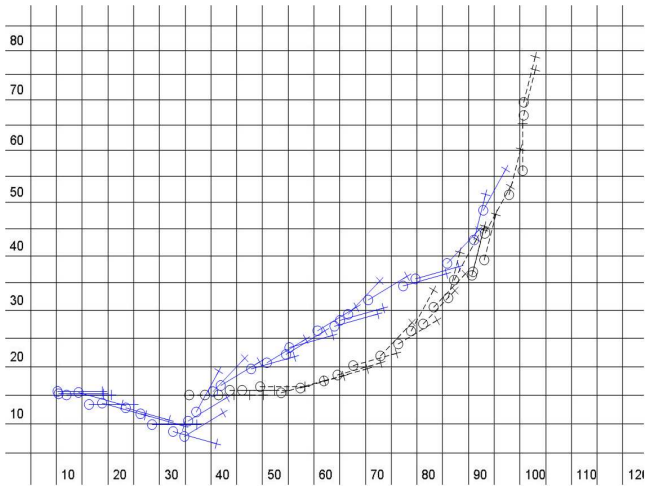

Figure 5.13. 2D Formation - Curved.

it is reduced to $4.2in$ by the end. Similarly, Figure 5.15 shows the orientation error vs. time. We can see that it remains small and is -4^o at the end.

Multiple Step Leader Trajectory

Figure 5.16 shows the time lapse positions of the robots where the leader was given a multi-step path. This consisted of:

- Straight path of $40cm$
- Turn of 45^o
- Straight path $130cm$
- Turn of -45^o
- Straight path of $40cm$
- Turn of 90^o
- Straight path of $40cm$

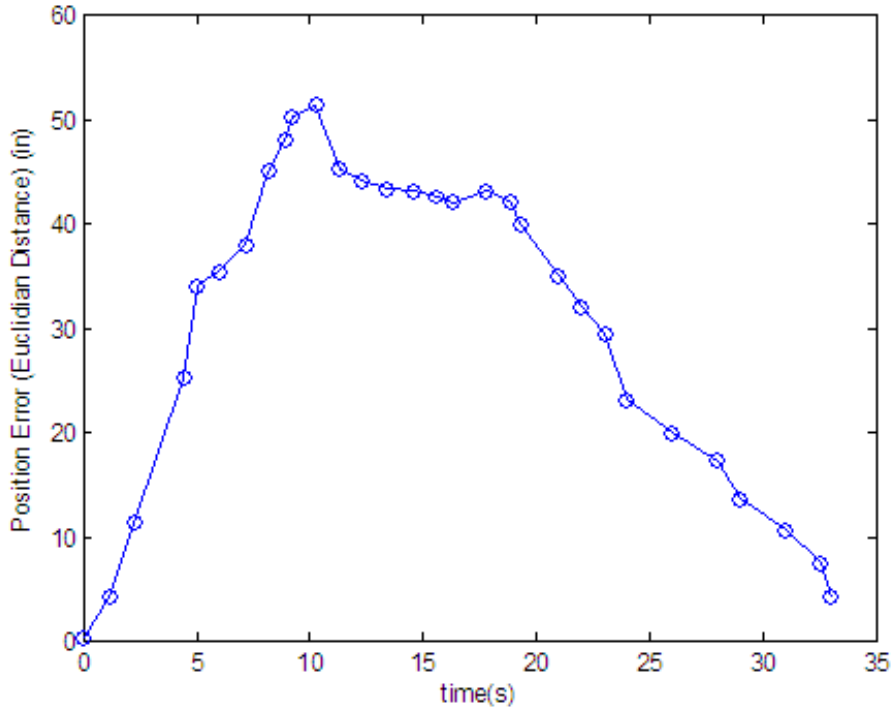

Figure 5.14. 2D Formation - Curved Line Position Error.

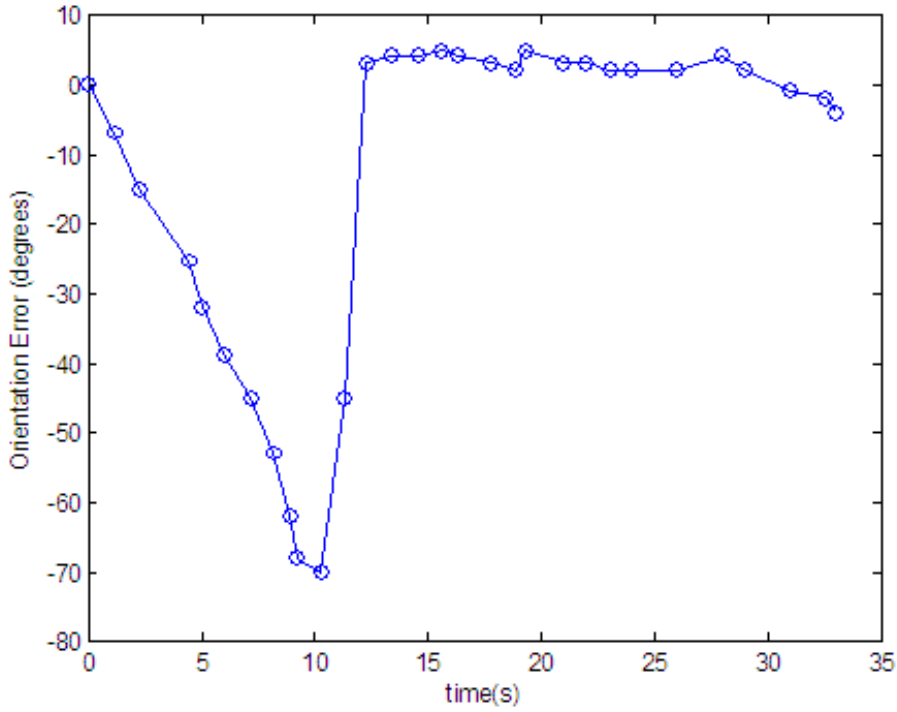

Figure 5.15. 2D Formation - Curved Line Orientation Error.

Table 5.11. Multiple Step Final Pose Error Results

Trial No.	X Position (in) Error	Y Position (in) Error	Angle φ Error$(^o)$
1	2.5	-0.5	6
2	-3.4	-5.1	-8
3	5.2	2.8	3
4	5.4	2.2	5
5	2.6	-0.9	-9

Table 5.11 shows the errors in x, y, φ for 5 test runs. We can see that the final pose of the follower is very close to the ideal position.

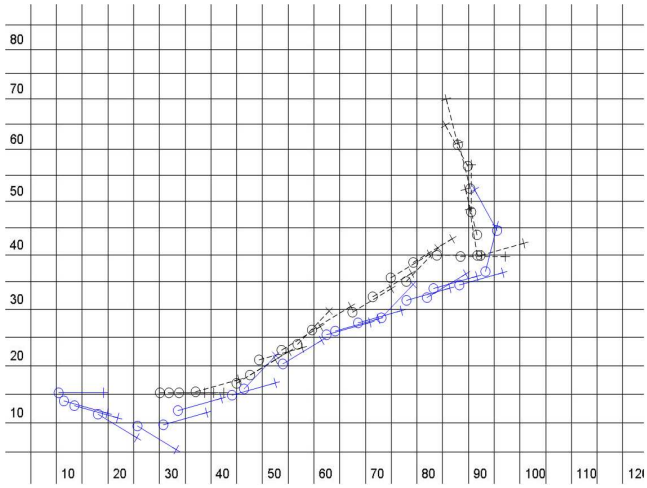

Figure 5.16. 2D Formation - Multiple Step.

CHAPTER 6

CONCLUSION

In this thesis the problem of formation control has been addressed. Specifically, it has been simulated for and implemented on a non-holonomic differential drive vehicle - the ARRIbot. The issue of formation control is formulated as a potential field navigation problem with the additional artificial force due to formation requirements. The overall trajectory is defined by a leader and the followers seek to maintain the formation. This allows us to determine the shape of the formation, the position of each node within the formation, and the trajectory the group must describe.

The advantages of the potential field approach include:

- Combine multiple objectives into one field. In our case, this included:
  - Geometric (referring to desired robot positions and orientations),
  - Communications centric(referring to optimal placement for maximizing the utility of a wireless channel, and
  - Information centric (referring to formation changes specified by an external algorithm)
- The various attractive and repulsive potentials used were of a Parabolic form. This form has an easy calculated analytic derivative. Thus, implementation on actual hardware is possibles.

The disadvantages of potential fields are:

- Tuning of weights is required to get best behavior. For the simulations in this thesis, the weights were decided first by choosing an appropriate damping value ν and then choosing by trial and error the weights on individual forces. Some forces were weighed stronger than others in order to obtain the desired behavior.
- Arbitrary Potential functions may have difficult or non-analytic derivatives. The choice of potential function depends on the application.

- Local minima - a condition where the presence of 'U' shaped obstacles, or a combination of multiple obstacles can trap robots due to the gradient always pointing toward the minima. Higher level heuristics or global path-planners can be used to overcome this.

In order to have the above formation control, an algorithm for trajectory tracking was also developed and implemented. A kinematic model for the ARRIbots that account for their non-holonomic constraint, linear and angular speeds, and wheel speeds as control inputs was developed and based on this model, a trajectory tracking controller (LQR based) was implemented assuming a given reference trajectory.

LQR trajectory tracking was found to satisfy the requirements of the formation control algorithm. Although the $desired - actual$ error was high, the $actual - estimated$ error was very low. Thus the dead-reckoning localization was accurate and could be used to correct the tracking error. LQR trajectory tracking was found to be advantageous in many ways:

- Calibration of motors is un-necessary. This should allow brand new robots to track a trajectory without any motor characterization except for 'zero-ing' of the stop value.
- There are no PID gains that need to be tuned. This is very time consuming for robots that use PID velocity controllers for trajectory tracking
- The final controller is simply a P-controller with time varying gains. This means that less storage is required because no Integral or Derivative terms are required.
- An estimate of the robot state ($[x, y, \varphi]^T$) is always available as a direct result of LQR. This is very useful to higher level algorithms.
- Any smooth trajectory can be tracked given enough processing power and storage.
- Special Cases such as 'Straight Line' and 'Axis Turn' have constant LQR gains resulting in great simplification.

The disadvantages of LQR include:

- LQR gains are trajectory dependant.
- Q and R matrices need to be tuned.

- Needs a very accurate timer.
- A fast processor is required.
- Storage of pre-calculated gains takes up a lot of memory ($3x3xN$, where N is the number of steps)

These results were validated and tested using computer simulations and experiments on our mobile robot platform at ARRI. The formation control was found to work well for the leader trajectories tested. The trajectory tracking controller was found to perform as well as a calibrated robot, but has the advantages of feedback and not requiring calibration.

While testing our formation algorithm important general issues that affect mobile robots came to light such as:

- Communication effects over a networked system. Lost packets during wireless transmission are a major concern and a scheme for guaranteed message delivery had to be used.
- Embedded micro-controller programming. Smaller scale processors typically include only a subset of the features of general purpose processers (floating point math, available memory, interrupts, programming libraries). These feature had to be worked around by implementing them using lower lever instructions.
- Hardware reliability - some parts were more prone to failure than others.
- Hardware compatibility - some sensors and actuators are designed with a bus system design (for example I^2C) instead of direct pin wiring via a multiplexer as in the ARRIbots.

Much additional research is possible and required in the field of formation control. A through study of communication delay effects on the formation is one area that could be expanded upon. The stability of the formation (string, mesh) also needs to be analyzed along with addition of addition of robustness to modeling and localization errors. A thoroughly decentralized implementation of formation control (with local path-planning and sensors) would be a continuation of this research. Sources of error should be reduced by the use of more powerful hardware. Finally, a higher level control needs to be devel-

oped in order to define the leader trajectories (mission control, task/research allocation) and formation geometry. An interface to easily specify these would also enable further testing and possible commercial deployment.

REFERENCES

[1] Wireless sensor network. [Online]. Available: http://en.wikipedia.org/wiki/Sensor_network

[2] J. Ghadigaonkar, "Arribot - an enabler for wireless sensor networks," Master's thesis, The University of Texas at Arlington, Arlington TX, 2006.

[3] J. Mireles, "Kinematic models of mobile robots," The University of Texas at Arlington, Tech. Rep., Aug. 2004.

[4] C. Batten, "Control for mobile robots," Mobile Autonomous System Laboratory, Massachusetts Institute of Technology, Tech. Rep., 2007. [Online]. Available: www.mit.edu/ cbatten/work/maslab/maslab-control-talk.pdf

[5] T. Sugar and V. Kumar. (2000) Control and coordination of multiple mobile robots in manipulation and material handling tasks. [Online]. Available: http://www.eas.asu.edu/ tsugar/exec/iserpaper.html

[6] (2000) Algorithms for control and interaction of large formations of robots. [Online]. Available: http://roboti.cs.siue.edu/projects/formations/

[7] J. Wen and M. Arcak, "A unifying passivity framework for network flow control," 2002. [Online]. Available: citeseer.ist.psu.edu/wen02unifying.html

[8] C. Helm, "Robotic sensor deployment using potential fields," Master's thesis, Rensselaer Polytechnic Institute, New York, 2004.

[9] F. L. Lewis, "Wireless sensor networks," The University of Texas at Arlington, Arlington TX, Tech. Rep. [Online]. Available: http://arri.uta.edu/acs/

[10] K. Sreenath, "Adaptive sampling with mobile wsn," Master's thesis, The University of Texas at Arlington, Arlington TX, 2005.

[11] D. Popa, "Path-planning and feedback stabilization of nonholonomic control systems," Ph.D. dissertation, Rensselaer Polytechnic Institute, Newyork, Feb. 1999. [Online]. Available: http://arri.uta.edu/popa/

[12] J. Desai, J. Ostrowski, and V. Kumar, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 905–908, Dec. 2001.

[13] S. Gorthi, "Real time data monitoring and manipulation for wireless sensor networks," Master's thesis, The University of Texas at Arlington, Arlington TX, 2006.

[14] CrossBow, *Cricket v2 User Manual*. MIT CSAIL, 2004.

[15] Z. Li and J. F. Canny, "Robot motion planning with nonholonomic constraints," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M89/13, 1989. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/1989/1165.html

[16] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[17] F. Pin, "Motion planning for non-holonomic wheeled vehicles," Oak Ridge National Laboratory, TN, Tech. Rep.

[18] M. Sampei, T. Tamura, T. Kobayashi, and N. Shibui, "Arbitrary path tracking control of articulated vehicles usingnonlinear control theory," *IEEE Transactions on Control Systems Technology*, vol. 3, pp. 125–131, Mar. 1995.

[19] M. Hannah, "Approximate tracking for nonholonomic chains of order one," *In Proceedings of the 2nd IEEE Symposium on New Directions in Control and Automation*, Mar. 1994.

[20] T.-C. Lee, K.-T. Song, C.-H. Lee, and C.-C. Teng, "Tracking control of unicycle-modeled mobile robots using a saturation feedback controller," *IEEE Transactions on, Vol.9, Iss.2, Mar 2001 Control Systems Technology.*

[21] E. Lefeber, J. Jakubiak, K. Tchon, and H. Nijmeijer, "Observer based kinematic tracking controllers for a unicycle-type mobile robot," *Proceedings 2001 ICRA. IEEE International Conference on, Vol.2, Iss., 2001 Robotics and Automation, 2001.*

[22] G. Oriolo, A. De Luca, and M. Vendittelli, "Wmr control via dynamic feedback linearization: design, implementation, and experimental validation," *IEEE Transactions on, Vol.10, Iss.6, Nov 2002 Control Systems Technology.*

[23] K.-C. Cao and Y.-P. Tian, "A time-varying cascaded design for trajectory tracking control of nonholonomic systems," *Chinese Control Conference, 2006*, pp. 2058–2063, Aug. 2006.

[24] E. Lefeber, J. Jakubiak, K. Tchon, and H. Nijmeijer, "Observer based kinematic tracking controllers for a unicycle-type mobile robot," *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation, 2001.*, vol. 2, pp. 2084–2089.

[25] X. Liu, A. Goldsmith, S. Mahal, and J. Hedrick, "Effects of communication delay on string stability in vehicle platoons," *Proceedings. 2001 IEEE, Vol., Iss., 2001 Intelligent Transportation Systems, 2001.*

[26] S. Spry and J. Hedrick, "Formation control using generalized coordinates," *CDC. 43rd IEEE Conference on, Vol.3, Iss., 14-17 Dec. 2004 Decision and Control, 2004.*

[27] F. Zhang, M. Goldgeier, and P. Krishnaprasad, "Control of small formations using shape coordinates," *Proceedings. ICRA '03. IEEE International Conference on, Vol.2, Iss., 14-19 Sept. 2003 Robotics and Automation, 2003.*

[28] M. Yamakita and M. Saito, "Formation control of smc with multiple coordinate systems," *(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, Vol.1, Iss., 28 Sept.-2 Oct. 2004 Intelligent Robots and Systems, 2004.*

[29] K.-H. Tan and M. Lewis, "Virtual structures for high-precision cooperative mobile robotic control," *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on, Vol.1, Iss., 4-8 Nov 1996.*

[30] R. Beard, J. Lawton, and F. Hadaegh, "A coordination architecture for spacecraft formation control," *IEEE Transactions on, Vol.9, Iss.6, Nov 2001 Control Systems Technology.*

[31] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, M. C. Stone, Ed., vol. 21, no. 4. New York, NY: ACM Press, July 1987, pp. 25–34. [Online]. Available: http://portal.acm.org/citation.cfm?id=37406

[32] R. Saber and R. Murray, "Flocking with obstacle avoidance: cooperation with limited communication in mobile networks," *Proceedings. 42nd IEEE Conference on, Vol.2, Iss., 9-12 Dec. 2003 Decision and Control, 2003.*

[33] W. Fan, Y. Liu, F. Wang, and X. Cai, "Multi-robot formation control using potential field for mobile ad-hoc networks," *Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on*, pp. 133–138.

[34] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," *IEEE Int'l. Conf. on Robotics and Automation, 1217–1222.*, 2002.

[35] S. Low, F. Paganini, and J. Doyle, "Internet congestion control," *IEEE, Vol.22, Iss.1, Feb 2002 Control Systems Magazine.*

[36] A. Divelbiss, "A global approach to nonholonomic motion planning," Master's thesis, Rensselaer Polytechnic Institute, New York, 1992.

[37] J. Latombe, *Robot Motion Planning.* Bostion, MA: Kluwer Academic Publishers, 1991.

[38] H. Choset and J. Burdick, "Sensor based planning, part i: The generalized voronoi graph," 1995. [Online]. Available: citeseer.ist.psu.edu/choset95sensor.html

[39] R. E. Tarjan, *Data Structures and Network Algorithms.* Society for Industrial and Applied Mathematics, 1983.

[40] R. Sedgewick, *Algorithms*, 2nd ed. Addison-Wesley, 1988.

[41] F. Lizarralde and J. Wen, "Feedback stabilization of nonholonomic systems in presence of obstacles," *IEEE International Conference on, Vol.3, Iss., 22-28 Apr 1996 Robotics and Automation, 1996. Proceedings., 1996.*

[42] A. Divelbiss and J. Wen, "Trajectory tracking control of a car-trailer system," *IEEE Transactions on Control Systems Technology*, May 1997.

[43] M. Egerstedt, X. Hu, and A. Stotsky, "Control of mobile platforms using a virtual vehicle approach," *IEEE Transactions on, Vol.46, Iss.11, Nov 2001 Automatic Control.*

[44] Y. Li and X. Chen, "Leader-formation navigation with sensor constraints," *IEEE International Conference on, Vol., Iss., 27 June-3 July 2005 Information Acquisition, 2005.*

[45] J. Fredslund and M. Mataric, "A general algorithm for robot formations using local sensing and minimal communication," *IEEE Transactions on, Vol.18, Iss.5, Oct 2002 Robotics and Automation.*

[46] G. Kim, D. Y. Lee, Lee, and Kyungno, "Formation of mobile robots with inaccurate sensor information," *Transactions on Control, Automation and Systems Engineering, vol.3 no.4 pp.203-209, 2001.*

[47] Parallax, *Javelin Stamp Manual.* Parallax Inc, 2004.

[48] (2003) Tinyos tutorial. [Online]. Available: http://www.tinyos.net/tinyos-1.x/doc/tutorial/

BIOGRAPHICAL STATEMENT

Rohit S. Talati was born in Pune, India and completed his schooling in Abu Dhabi, United Arab Emirates. He received his Bachelor of Science degree in Electrical Engineering from The University of Toronto, Canada in 2005. He then joined the Master of Science Program in Electrical Engineering at University of Texas at Arlington in Spring 2006. During his time, he worked as a mentor at IEEE student branch and as a research assistant at the Automation and Robotics Research Institute (ARRI). Motivated by his interest in the field of system controls and robotics, he pursued hexapod walking gaits, mobile robot localization, interface design for overall mobile robot control, and finally took up formation control under the supervision of Dr. Dan O. Popa. This thesis is reflection of his work on research study and implementation done on formation control.