Master's Thesis
Computer Science
Thesis no: MCS-2011-05
January 2011

Asymmetric potential fields

Implementation of Asymmetric Potential Fields in Real Time Strategy Game

Muhammad Sajjad
Muhammad Mansur-ul-Islam

School of Computing
Blekinge Institute of Technology
SE – 371 79 Karlskrona
Sweden

This thesis is submitted to the School of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:

Authors:
Muhammad Sajjad
sajjad81@gmail.com

Muhammad Mansur-ul-Islam
mansurulislam@gmail.com

University advisor:
Dr. Stefan Johansson, PhD.
School of Computing,
Blekinge Institute of Technology

School of Computing
Blekinge Institute of Technology
SE – 371 79 Karlskrona
Sweden

Internet : www.bth.se/com
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ACKNOWLEDGEMENT

بسم الله الرحمن الرحيم

First of all we would like to thanks almighty ALLAH for his blessings in the completion of our thesis. We would like to thank our parents and family who supported us morally and encouraged us to complete our task.

We are thankful to Stefan Johansson, who guided us in the right way, supported us and gave us positive feedback. It was really difficulty to complete the thesis in a good way without the guidance and support of Stefan Johansson.

We are also thankful to J. Hagelbäck, who helped us to understand the structure of BWAPI. We are also thankful to our friends in BTH who supported us morally and had best wishes for us regarding the completion of thesis.

ABSTRACT

Context. In eighties, the idea of using potential fields was first introduced in the field of the robotics. The purpose of using potential fields was to achieve the natural movement in robotics. Many researchers proceeded this idea to enhance their research. The idea of using potential fields was also introduced in real time strategy games for the better movement of objects.

Objectives. In this thesis we worked on the idea of using asymmetric potential fields in the game environment. The purpose of our study was to analyze the affect of asymmetric potential fields on unit's formation and their movement in game environment. In this study performance of asymmetric potential fields was also compared with symmetric potential fields.

Methods. By literature review the potential field and its usage in RTS games were studied. The methodology to implement the potential fields in RTS game was also identified in literature review. In experimental part the asymmetric potential fields implemented by using the methodology proposed by Hagelbäck and Johansson. By following that methodology asymmetric potential field was applied on StarCraft bot by using the BWAPI. Experiment was also designed to test the asymmetric potential field bot.

Results. Asymmetric potential field bot was tested on the two maps of StarCraft: Brood War game. On these two maps, bot implemented with asymmetric potential field and the bot implemented with symmetric potential field competed with four bots. Three bots were selected from StarCraft competition and one was built-in bot of this game. The results of these competition shows that asymmetric potential field bot has better performance than symmetric potential field bot.

Conclusions. The results of experiments show that the performance of bot implemented with asymmetric potential fields was better than symmetric potential field on single unit type and two unit types. This study shows that with the help of asymmetric potential fields interesting unit formation can be formed in real time strategy games, which can give better result than symmetric potential fields

Keywords: Potential fields, asymmetric potential fields, Real time strategy games.

ABBREVIATIONS

RTS	Real time strategy
PF	Potential fields
ORTS	Open Real Time Strategy
BWAPI	Brood War Application Programming Interface
SPF	Symmetric Potential Fields
ASPF	Asymmetric potential Fields
API	Application Programming Interface
DLL	Dynamic-link library
AI	Artificial Intelligence
MSD	Maximum Shooting Distance

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 BACKGROUND

Computer games are emerging as a most growing part of entertainment world. With the passage of time video games have got popularity in the all ages ranging from children to adult. Due to this popularity people demands more complexity in challenges and the real behavior of the environment of the game [26]. Due to this demand AI researcher has also started to concentrate on real time strategy (RTS) games instead of turn based games [27].

RTS games are the special type of games in which players interact in real time without waiting for another player's turn [1][9]. These games are based on military simulation and abstraction [2].Player behaves as a leader of the tribe and gathers resources scattered over 2D terrain to increase its domestic and military power. Player increases its technology power and manages units to defeat another tribe and defense its building and resources [3]. In RTS games all decisions are taken at run time. In RTS game generally user has a top down approach and multiple users can interact with game, independently of each other and without waiting for turn [3].

The main theme of the RTS games is to provide an environment to the players in which they can manage resources and take decision in any circumstances and move accordingly with the collaboration of units in real time. In RTS games several teams struggle to gather resources in order to get the military superiority and territorial control over each other [10]. Age of empire and StarCraft are the most popular examples of RTS games.

The concept of potential fields is very useful to avoid collision and achieve human like movement of objects in RTS games. The usage of potential fields really affects the performance of a RTS game [8]. The idea behind potential fields has some similarity with the influence maps [4].Influence map is a widely used technology in robotic and game environment to create artificial intelligence agent. [29].This technology is mainly used in strategy games but also useful in games where tactical analysis was required [30]. Knowledge of artificial agent about the game environment is represented by influence map [30]. The location of enemy position, food, weapon or own forces were indicated in it.

In the potential fields charges are put on the certain points in the game. Positive charges in most cases represent the attraction field that attracts an object to the destination. Negative charges in most cases used for the repelling force that avoids an object from the obstacles [4].

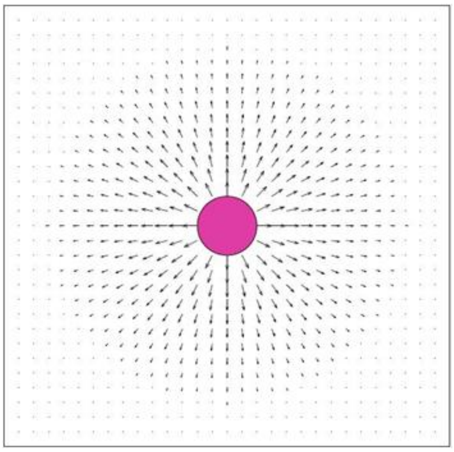


Figure 1: Repulsive potential fields

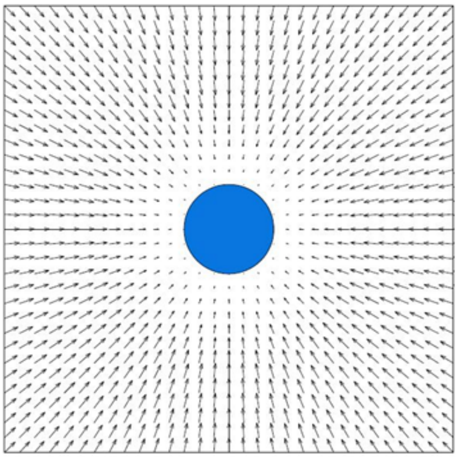


Figure 2: Attractive potential Fields

Figure 2 shows the attractive potential fields around a particular point that can be a destination or goal where an object wants to reach. The attractive force will attract object towards this certain point. Figure 1 shows the repulsive potential fields around a point. This point can be an obstacle or enemy units. This repelling force will repel object away from the obstacle so that an object can follow right path to reach their destination.

1.2 RELATED WORK

Ossama Khatib was the very first person who gave the concept of potential fields in robotics in 1985. He named that potential field as artificial potential field [7]. He used artificial potential field in robotics to avoid obstacles in the object movement [6]. The idea of potential field is to put charges on the goal and obstacles. Obstacles have repelling charges and goal has attractive charges. [4]

Later on Arkin [11] came up with another technique which is based on the spatial navigation of vector fields and proved better than the previous ones. He introduced this technique with name of motor schema. This technique was used as the alternative of different potential methods to produce suitable speed and intelligent moves for the robot. Basically motor schema provides a robot simple rules or information to take an intelligent decision. [11]

In game the idea of potential fields is used to apply charges at interesting point in the game world and charges generate a field which gradually fades to zero [5]. If these charges are positive (attractive) then object moves toward it. If it has negative (repelling) charges object move away from it. Applying potential fields in games is new idea in gaming industry.

In 2006 research conduct by Hector conducted a research, in which multiple potential fields were applied to Quake 2 game [12]. The results of his research shows that potential fields give the good result and agent in this game explores the map of whole virtual world and also fight with the enemy very well by positioning itself in good position. This results show that potential fields should be consider for using in gaming industry.

Another researcher, Johan Hagelbäck [8] implemented open real time strategy (ORTS) based on potential fields. ORTS is a game engine developed particularly for researchers in game artificial intelligence. In this research each unit assigned a set of charges which generates potential fields around it. To take decision for any unit, the potential fields for current and surrounding was calculated. If the surrounding potential field is higher it moves toward that area otherwise it remains idle. In that implementation no path finding algorithm was used. The implementation was focus on two types of games that was tank battle and tactical combat [8]. The implemented client takes part in the competition; the result was not so impressive. But it shows that potential field is an alternative solution of path finding algorithms and it need more research to implement potential fields.

Potential Fields is a highly parallel, fast, reactive alternative to parallel planning e.g. using A* in games. It calculates the utilities of the possible next move by evaluating them with one step look-ahead only [3]. Usually attractive and repulsive fields have the behavior similar to the behavior of fields in real physics i.e. generate a field that is symmetric around a given point. The goal of this research is to see how asymmetric fields can be used to create dynamic unit formations in RTS games. Previously some work has done in using potential fields in games. But there is no research done on behavior of RTS game when asymmetric potential fields is applied on it.

1.3 PROBLEM DOMAIN

Potential fields were being used in robotic successfully. Now some researcher has done some study about using potential fields in games [8] [12] [33]. Hagelbäck and Johansson proposed methodology to implement symmetric potential fields in RTS games [3]. But we did not found any research related to using of asymmetric potential fields in RTS games. We have done research in that area and implemented asymmetric potential fields in RTS games. We also tried to find how interesting formation can be created in RTS game by using asymmetric potential fields.

1.4 AIM AND OBJECTIVES

The aim of our research is to implement the asymmetric potential fields to create units formation in RTS games. There are following objectives that will help us to achieve our goal.

- Working knowledge of symmetric potential fields in RTS games.
- How asymmetric potential fields can apply on single unit type.
- Apply asymmetric potential fields on units in RTS games.
- Comparison of asymmetric with symmetric potential fields.

1.5 RESEARCH QUESTIONS

RQ1: How can be asymmetric potential fields used in RTS games?

RQ2: How to implement asymmetric potential fields that protect a leader with dynamic unit formations while combating enemies in RTS games?

RQ3: Is asymmetric potential fields more effective as compared to symmetric potential fields?

1.6 RESEARCH METHODOLOGY

The main research method used for this study is experiment, which is quantitative approach.

1.6.1 LITERATURE REVIEW

As a prerequisite we have performed literature review and studied books, articles and journals related to the usage of potential fields. The purpose of the study was to analyze the previous work done in this field and to find the procedures and methods in order to carry out this research.

1.6.2 EXPERIMENT

In the next step we have implemented asymmetric potential fields in the RTS game StarCraft. Purpose of the study is to test the effectiveness of asymmetric potential fields in RTS games in comparison to symmetric potential fields. To achieve mentioned purpose we have also developed bot with implementation of symmetric potential fields. And finally the experiments were designed and conducted for comparison of symmetric potential field bot with asymmetric potential field bot.

1.6.2.1 Experiment Overview

We have conducted two experiments to evaluate our study. We have selected two different maps from the StarCraft AI Competition AIIDE2010 [19]. These maps are Dragoon Battle and Dragoon Air Battle. The Dragoon Battle map is available on the competition website [19]. We have also modified the Dragoon Battle map to two units type map and named that map Dragoon Air Battle map. We have done experiments by testing bots (symmetric and asymmetric) on these two maps.

More about experiment is presented in Chapter 5.

1.7 RELATION BETWEEN RESEARCH METHODOLOGY AND RESEARCH QUESTIONS

RQ1 is related to our theoretical research. To answer RQI we have gone through the material regarding previous studies about the usage of artificial potential fields in the game environment. On the basis of previous study we designed the methodology for the implementation of asymmetric potential fields in RTS games.

We did empirical research to find answer of RQ2. In RQ2 we have done experiment and implemented the asymmetric potential fields in an RTS game according to the methodology which we found from RQ1. In order to answer the RQ3 the results were analyzed that were obtained from experiment in empirical evaluation. The overall research plan is shown in Figure 3.

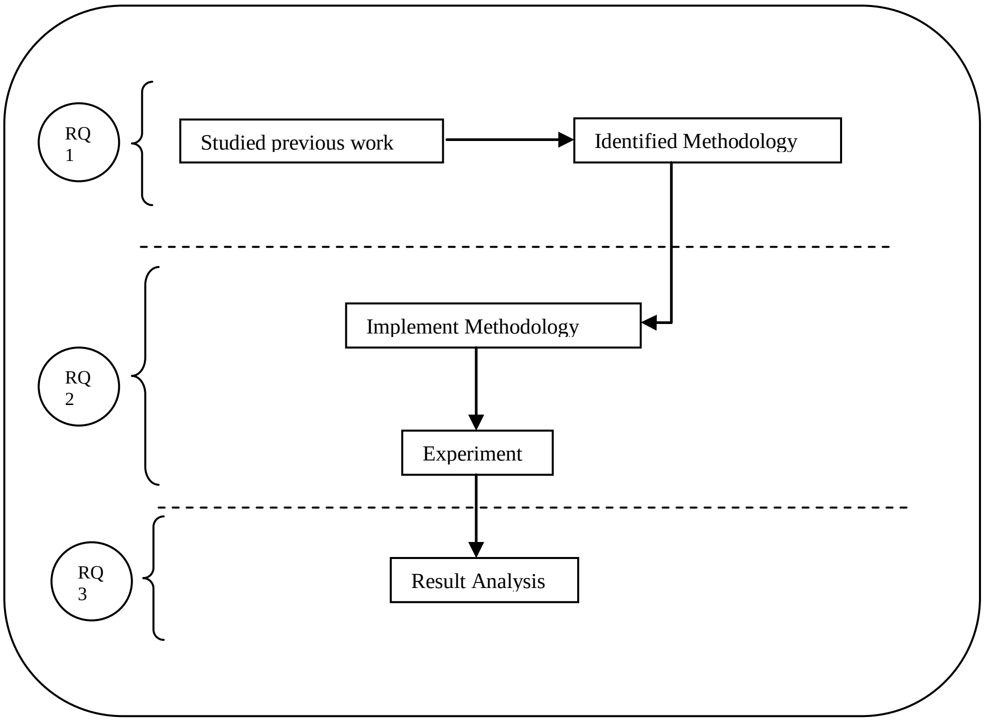


Figure 3: Research Plan

2 POTENTIAL FIELDS

2.1 ARTIFICIAL POTENTIAL FIELDS CONCEPT

A particular behavior is the result of a change that an agent experiences in a particular environment. There can be more than one behavior in the response of a stimulus. There are two popular techniques that deal with the multiple behaviors [13]:

- Always –on
- Sometimes-on

In the always-on technique, an agent always looks on the environment and performs actions accordingly. The sometimes-on technique is different compared to always-on and only comes in action where there is some specific change in the environment [13].

Potential Field is one of the behavior based techniques [13]. Potential field is based on always-on technique always looking on the changes in the environment. Potential field does not need any particular event to perform for the behavior. In 1985, Ossama gave the idea of artificial potential fields in robotics to avoid the obstacles and achieve human like movements [7].

Potential field has some similarity with the influence maps. In influence map, numerical values are used with positive and negative signs to represent area occupied by own unit and by enemy units [4]. In Figure 4, positive values show field of attraction and also indicate that area is occupied by own units and negative values shows repelling field and also indicate that area is occupied by enemy units. 0 shows that region has no field.

0	2	6	2	0	0	0	0	0
2	6	8	6	2	0	0	0	0
6	8	10	8	6	2	0	0	0
2	6	8	6	2	0	-2	0	0
0	2	6	2	0	-2	-6	-2	0
0	0	2	0	-2	-6	-8	-6	-2
0	0	0	-2	-6	-8	-10	-8	-6
0	0	0	0	-2	-6	-8	-6	-2

Figure 4: Influence map [4]

The idea of working influence map is that put some positive numeric value on own unit, i.e. is 10 in Figure 4. This value is decreased gradually in nearby cells and shows the influence of the units. This value converges to zero.

2.2 HOW DO POTENTIAL FIELDS WORK

Action vector is the representation of each behavior and the combination of action vectors creates potential fields accordingly [13]. Here we will take an example of a robot that has to reach the destination. The action vector in this case will point the robot to the specific destination. The robot will have to follow a route to reach the destination and during this there will be action vectors that will guide the robot toward destination. Collectively these all action vectors will generate attractive potential fields as shown in Figure 5.

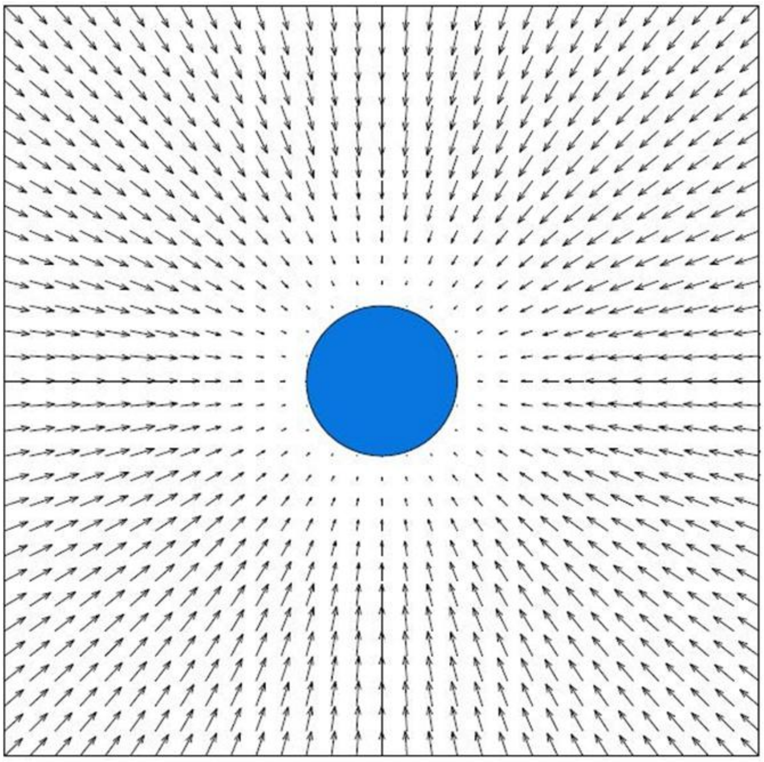


Figure 5: Representation of attractive potential fields [13]

To understand how to calculate the potential fields in above figure, we can think of mapping of one vector in another vector [13].

To calculate the action vectors of the field in which agent attracted toward the goal, let us suppose goal is placed at coordinate (xg,yg) and agent which is attracted to the goal is placed at (xa,ya).

Formula used for distance (d) calculation between agent and goal is [13],

$$d= \sqrt{(xg - xa)^2 + (yg - ya)^2} \qquad (1)$$

Angle between agent and goal is calculated with formula to find in which direction agent has to travel to reach goal. The formula is given below: [13].

$$\theta = \tan^{-1}\left(\frac{yg-ya}{xg-xa}\right) \qquad (2)$$

According to distance d and angleθ, Δx and Δy can be calculated as [13]:

$$\Delta x = \begin{cases} 0 & if\ d < r \\ \propto (d-r)\cos\theta & if\ r \leq d \leq s+r \\ \propto s\cos\theta & if\ d > s+r \end{cases} \qquad (3)$$

$$\Delta y = \begin{cases} 0 & if\ d < r \\ \propto (d-r)\sin\theta & if\ r \leq d \leq s+r \\ \propto \sin\theta & if\ d > s+r \end{cases} \qquad (4)$$

Where r is the radius of the goal, s describes the spherical field around the goal on which it has an influence, and α is the constant which describes the scale, containing the values greater than 0, which shows the strength of field.

The action vector Δx and Δy describe the behavior of agent on the base of values describe above. If agent reaches in the radius of goal then Δx and Δy becomes zero. At this point no force will apply on it because it reaches the goal. If the agent is outside the radius of goal but inside the spherical region of field then the magnitude of action vector is proportional to the distance between agent and goal and its direction is toward the goal.

If the agent is outside the radius and sphere of field then the maximum value is assigned to magnitude of vector.

Now consider the situation when there is an obstacle and field is repelling as shown in Figure 6.

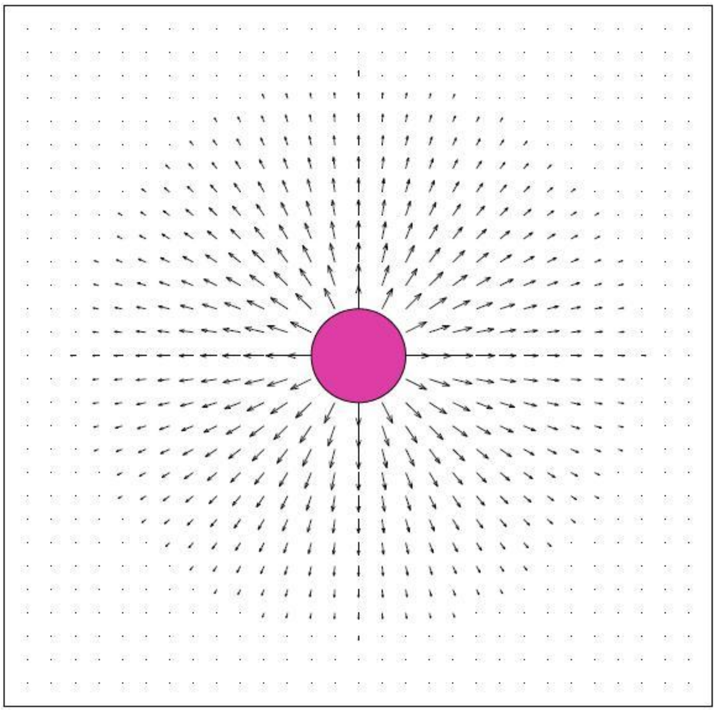


Figure 6: Representation of repulsive potential field [13]

In this condition the action vector is calculated in same manner as describe for the Figure 5. The calculated values of resultant vector Δx and Δy [13] are

$$\Delta x = \begin{cases} \infty & if\ d < r \\ -\beta(s+r-d)\cos\theta & if\ r \le d \le s+r \\ 0 & if\ d > s+r \end{cases} \quad (5)$$

$$\Delta y = \begin{cases} \infty & if\ d < r \\ -\beta(s+r-d)\sin\theta & if\ r \le d \le s+r \\ 0 & if\ d > s+r \end{cases} \quad (6)$$

If the agent is in radius of goal then the values of resultant vector which shows the potential field is infinite. If agent is within the sphere s of obstacle then the magnitude of vector is equal to distance between the agent and obstacle. The negative sign of the Δx and Δy shows that it is repelling and its direction is outside from sphere s. If agent is outside radius and sphere then there will be no repelling force on it so the magnitude of vector is 0.

The above described equations for attractive and repulsive forces are probably good in a robotics setting where the robot moves fast toward the goal when it is far away from it. The speed of robot decreases when it gets closer in order to move slowly toward the goal with higher accuracy.

In most of the scenario the attractive and repulsive potential field can be applied on different objects at same time. To deal with this situation, both potential fields are combined and have impact on agent movement.

2.3 COMBINATION OF POTENTIAL FIELDS

In previous section the method for calculating the action vector in presence of attractive or repulsive forces is defined. But in real scenarios most of the times both type of forces are working together, as shown in Figure 7.

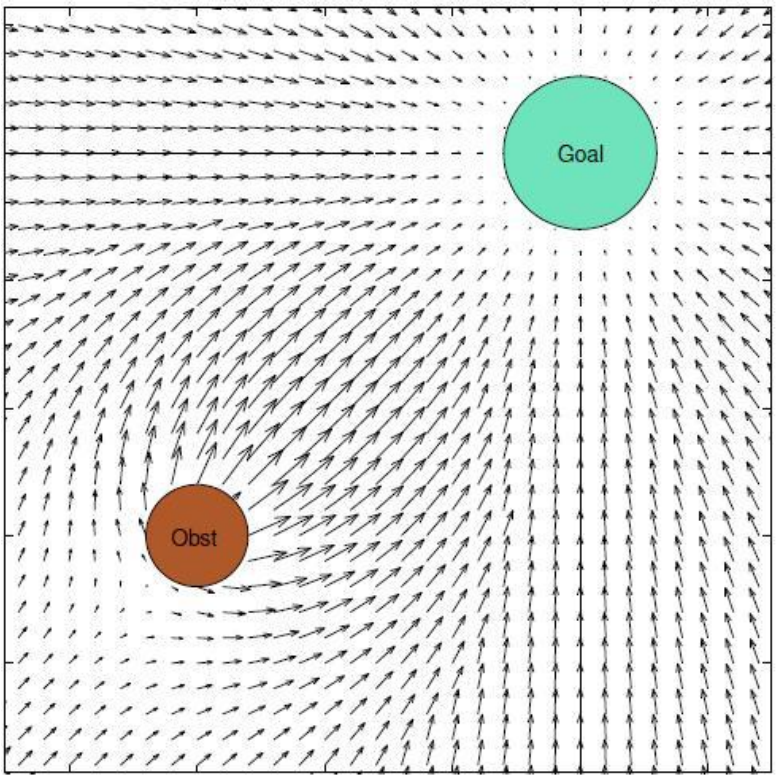


Figure 7: Representation of combination of two fields [13]

In Figure 7, both types of forces act upon agent. So how will agent behave in this condition? The action vector in this scenario can be calculated by adding both repulsive and attractive force [16]. Let suppose the Δxr and Δyr are the vectors of repulsive forces and Δxa and Δya are the vectors of attractive forces. The action vectors Δx and Δy that will guide an agent to avoid obstacle and to reach the goal can be expressed as [13]:

$$\Delta x = \Delta xr + \Delta xa \qquad (7)$$

$$\Delta y = \Delta yr + \Delta ya \qquad (8)$$

2.4 THE APPLICATION OF POTENTIAL FIELDS IN GAME ENVIRONMENTS

To understand how this action vector is implemented in a game environment to produce artificial potential fields, keep the Figure 4 of influence map in mind. High charges are assigned to interesting points which gradually become zero. Agent looks at its adjacent positions and moves toward high charges (if charges are attractive) to reach its destination. Here an agent can be described as the entities in the game environment and are assigned to do different tasks as per their unit type. Position with neutral charges shows the neutral part of map. If there are any obstacles in the map, high repelling values are assigned to it which also gradually becomes zero. If agent is near negative charges, it repels from negative charges and step toward positive charges. It can also be explained as potential fields work in the radius and out site this radius there will be no charge, if agent comes into this radius it will be attracted or repelled from it according to the charge nature.

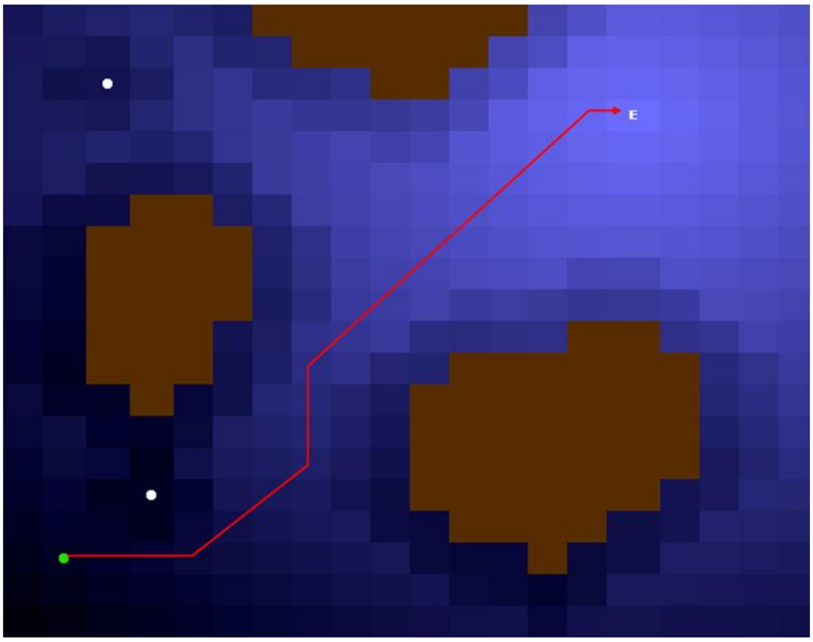


Figure 8: Behavior of potential field in game environment [4]

In Figure 8 the charge is placed at destination E, which spreads in the map. The agent will move towards high charges and following the path to reach at destination. Some obstacle mountains (brown color) and other agents (white circles) assign repelling charges. So agent will move away from it to avoid collision and move toward the attractive charge of E.

2.5 TYPES OF POTENTIAL FIELDS

Potential fields can be of different types depending upon the situation and the requirement of a programmer. There are some useful potential fields [12]. Some of them are mentioned below:

- Uniform Potential Fields
- Perpendicular Potential Fields
- Attractive Potential Fields
- Repulsive Potential Fields
- Asymmetric Potential Fields

2.5.1 UNIFORM POTENTIAL FIELDS

This potential field will guide an agent to move in a specific or desired direction depending on situation. This field may help an agent to follow a wall but cannot guide towards goal or enemy unit [13]. In the scenario where agent is exploring the map this field can be used. Agent will follow a constant field beside the wall or boundary and can explore whole area.

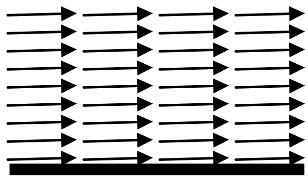


Figure 9 : Uniform Potential Field

2.5.2 PERPENDICULAR POTENTIAL FIELDS

This potential field is very simple in behavior. It is used to guide an object to prevent itself from the walls [13]. The field can guide the agent to move away from any obstacle, or also can used in moving toward enemy.

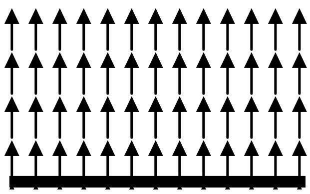


Figure 10: Perpendicular Potential Field

2.5.3 ATTRACTIVE POTENTIAL FIELDS

This potential field is the most important one and is used to attract an agent [5]. A programmer uses potential field at certain points in an environment and due to this field an object gets attracted towards that point. We will use attraction potential field in the StarCraft game on certain points depending upon our requirements e.g. Enemy units, goal.

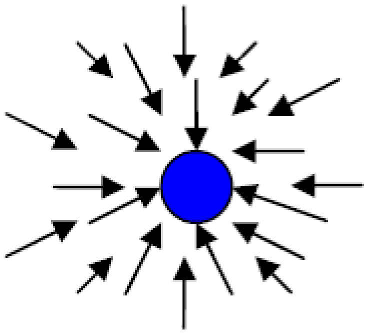


Figure 11: Attractive potential field [12]

2.5.4 REPULSIVE POTENTIAL FIELDS

This potential field also plays an important role in a certain environment. It behaves opposite to the attraction field It generates repulsive field that prevents an object from obstacles [5]. A programmer uses repulsive potential fields on different points according to need. We will use this repulsive potential field in StarCraft game on our own units specially to avoid them to collide with each other.

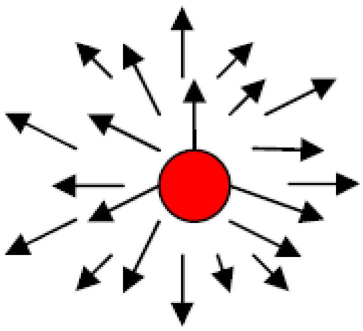


Figure 12: Repulsive potential field [12]

2.5.5 ASYMMETRIC POTENTIAL FIELDS

In asymmetric potential fields attractive and repulsive forces are generated from the same object at the same time. The asymmetric potential field can be useful for the formation of units in the game environment. Units can perform better against enemy units by possessing the attractive and repulsive fields at the same time. The representation of asymmetric potential fields is shown in Figure 13. If we consider a scenario that there is a gun in the enemy area.

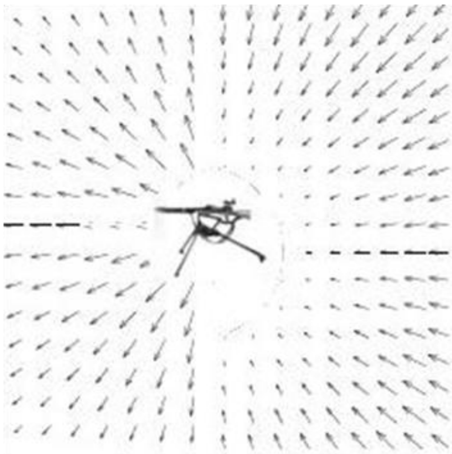


Figure 13: Asymmetric Potential Field

The mission of own unit has to reach the gun and destroy it. But if the unit will attack the gun from front it can easily be killed within the range of gun. To avoid killing of own unit and putting the enemy in worst case asymmetric fields is used. The unit is attracted towards the gun but to avoid the damage repelling field was generated at the MSD of enemy gun in front so own unit will be repelled from front as shown in Figure 13. At the back of the gun attractive potential fields was generated so own unit will attract toward back and can destroy gun without damaging itself as shown in Figure 14.

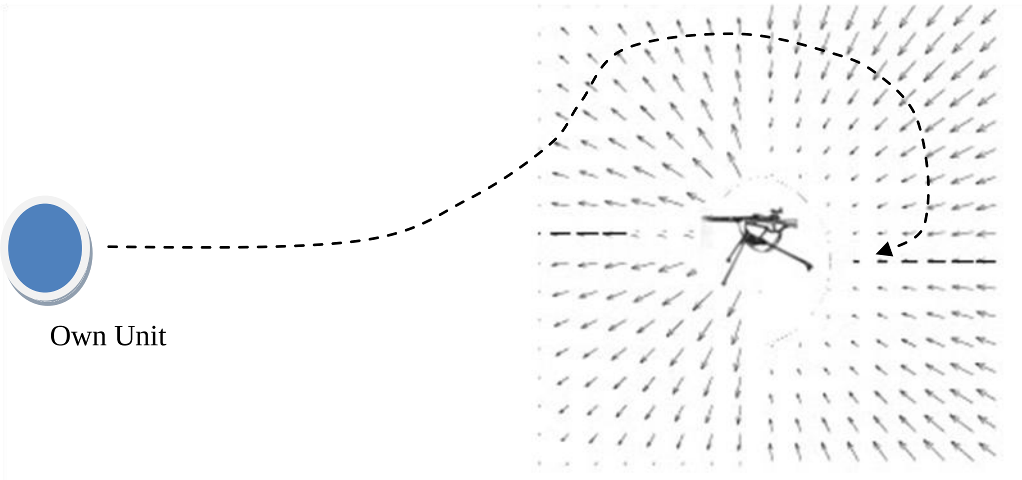


Figure 14: Representation of Asymmetric Potential Field

2.6 LOCAL MINIMA

An agent in a game environment moves toward a point which has highest value than its current position. Sometimes agent reaches a point which has the highest value than the surrounding position which leads an agent in an awkward position where agent cannot move forward neither can move back to its previous position [15]. It mostly happens when an agent moves toward a goal and it faces obstacle like wall or mountain in its way. The agent reaches a point where it cannot find the higher charges for the movement. At the time the repulsive and attractive forces becomes equal and the agent circulates among the adjacent positions in the same area.

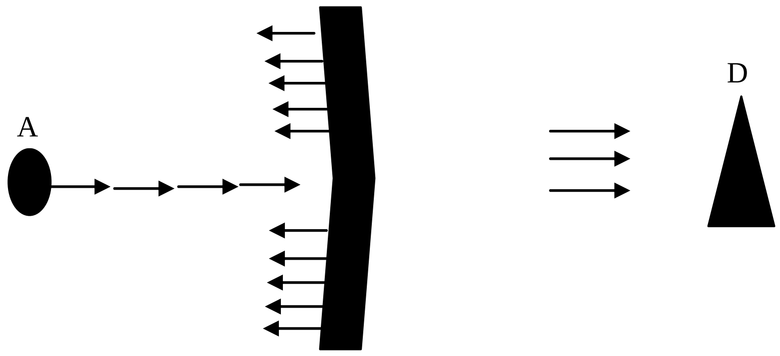


Figure 15: Representation of Local Minima

In Figure 15 an agent “A” is heading towards goal or destination “D” with the force of attraction, but there is an obstacle in agent’s way and has repelling force which will prevent an agent from the collision with the obstacle. An agent can be trapped in the local minima if magnitude of repelling and attractive forces becomes equal. In this situation an agent do not find a highest value than its current position to move on. There are several methods to avoid local optima. Here we will discuss some of them.

2.6.1 RANDOM POTENTIAL FIELD

An agent can come out of the local minima and can find better position by moving in different random positions [13].

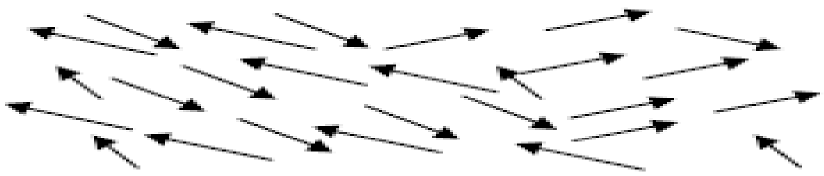


Figure 16: Random potential field [33]

2.6.2 ADDING TRAIL

Local minima can be avoided by pushing the agent to closest nodes by adding trail. Each agent will add the trail at last n positions that a unit has visited including current position of the unit [14]. Trail will produce some repelling force which will help unit to get out of local optima and unit will move forward to make its way towards a goal as explained in Figure 17. But if there is a complex obstacle then a unit can stuck in local minima even by adding trail. This is a flaw of using potential field methods for path finding but can be avoided by using best path finding techniques.

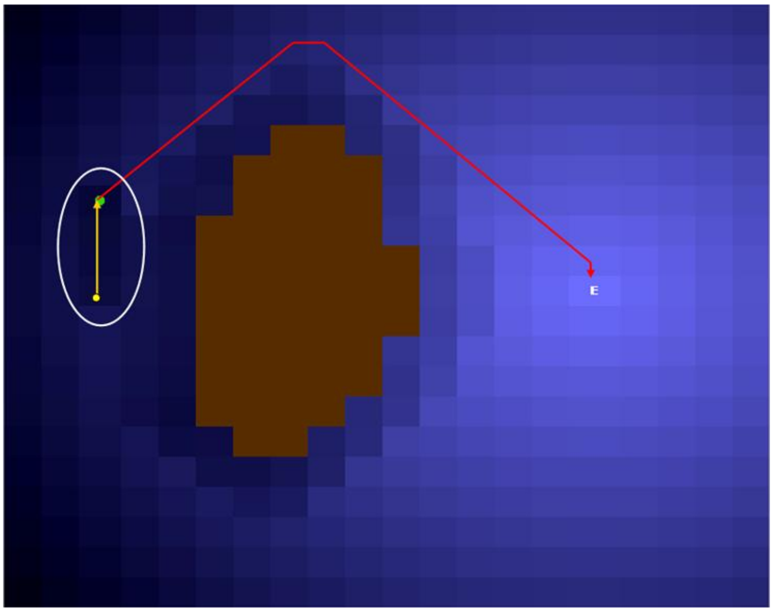


Figure 17: Agent is escaping from local minima with the pushes of trail [4].

2.7 ADVANTAGES OF PF:

- Potential field applied on multiple agents in ORTS provides more flexibility and efficiency as compared to other path planning methods [8].
- Potential field based solution behaves well in the changing environment and adopt changes in a game scenario [3].
- The efficiency of game regarding object movement can be increased by applying different potential fields together [8] [3].

3 STARCRAFT

The StarCraft game is one of the most popular RTS games [17]. It was developed and released by Blizzard Entertainment on 31st March 1998. It has been one of the best selling computer game, as 11 Million copies were sold worldwide up to February 2009[18][17].

The StarCraft is a military science fiction game, consists of the three races Terrans, Zerg and Protoss. Terrans are humans, who have left the earth and traveled to the galaxy. Zerg consists of several types of creatures, they are like insectoid creatures. They don't have technology power, but have natural weapons [17] [19]. Protoss has very advanced technology and it is based on humanoid race. In our research we have used the Dragoons and Corsair units which belong to the Protoss race. The characteristics of these two units are mentioned in Table1.

Characteristics	Dragoon	Corsair
Race	Protoss	Protoss
Role	Ground Unit	Space Fighter
Speed	5.25	6.67
Hit point	100	100
Ground Attack	Yes	No
Air Attack	Yes	Yes
Shields	80	80

Table 1: Characteristics of Dragoon and Corsair

In this StarCraft players plan and build structure to develop their technology. Players need to gather resources in order to develop structures. Each structure has its purpose. Different units are created to attack the enemy base and the enemy units and also for defending the own bases.

There are three main steps that players follow in the StarCraft game like a typical RTS game [32]. The first step is to gather resources and information in a game environment, the second step is to strengthen their unit's technology by using gathered resources and constructing refineries and base stations. The third step is to initiate attack against enemy by managing units in an effective way [32].

This game can be played by single player or it can also be played in multiplayer environment where players battle against human player and computers.

3.1 STARCRAFT BROOD WAR

The StarCraft: Brood War is the expansion of StarCraft and consists of number of new maps and campaigns [20]. It was released in USA on November 30, 1998. It can be played on the Windows and Mac operating systems. This expansion pack was developed with the collaboration of two popular video game companies (Saffire and

Blizzard Entertainment) [20]. The expansion pack is the advancement in StarCraft that introduced extra unit, new maps, and music. The StarCraft: Brood War brings in some good tweaking to units and best skills to make better strategy. These new features do not affect the main theme of the game that was based on the resources gathering, upgrading technologies and combating with enemies.

StarCraft: Brood War is a popular game among the players of all ages. In South Korea, StarCraft: Brood War matches are being conducted in a professional way among teams and players. The professionals in the form of teams take part in competition. These players get sponsorships to compete with their opponents [21].

3.2 BROOD WAR API

The Brood War Application Programming Interface (BWAPI) is an open source framework, which helps in communicating with StarCraft: Brood War game engine. This framework is developed in C++. This framework helps in creating AI modules, which communicate with StarCraft to retrieve information from game about units [22]. It also issue commands to the StarCraft. Bots for StarCraft can be developed by using BWAPI, which enabled developer to create AI based bots. Figure 18 shows the basic interface of BWAPI [22].

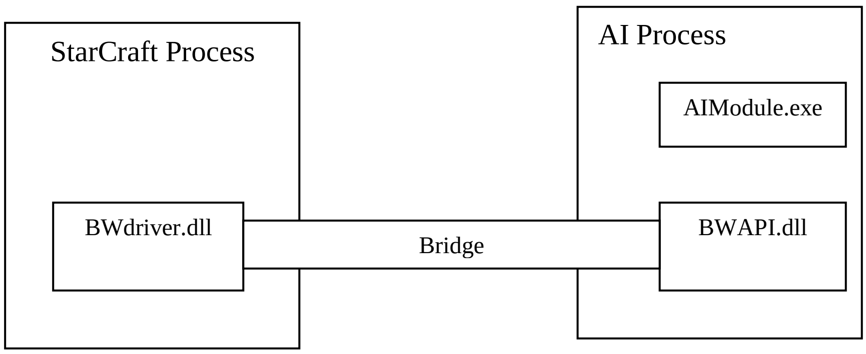


Figure 18: Interface of BWAPI

3.3 BTHAI

This is a computer program (bot /computer player) that plays StarCraft automatically without human involvement. The BTHAI communicates with the StarCraft engine through BWAPI. The BTHAI has a multi-agent architecture using built in path finding for the navigation in the game environment and potential fields in fighting with enemy [23]. This bot also participated in StarCraft AI competition [28]. In our research we decided to use BTHAI because in this project potential fields have been implemented and tested. We extended the BTHAI project for the implementation of asymmetric potential fields. Architecture of BWAPI is attached in appendix A.

Multi-agent System

BTHAI has multi-agent based architecture, so each unit in this project is represented as an agent [23]. Base agent is an abstract class for all agents. Structure Agent and Unit Agent are the two agents that control the game environment and they inherit from the Base Agent class. All buildings in the game environment are Structure

Agents and all units are Unit Agents weather they are attacking units or non attacking units.

Managers

All active agents are listed in the AgentManager. The numbers of agent which are attacking and defending are also contained in the AgentManager.

The Exploration Manager is responsible for all the activities performed to explore game world and keeps track of the movement of explorer units and decide the next exploring tasks.

SquadCommander acts as the leader of the force and takes decisions regarding attacking and retreating.

Potential Fields

PFManager facilitates the agent for the navigation in the game environment. The PF Manager uses pathfinder or potential field automatically for the navigation of agents towards their goals.

3.4 MOTIVATION FOR USING BWAPI

The BWAPI provides the interface to communicate with StarCraft: Brood War and bots for StarCraft: Brood War can be developed by using it. It provides the test bed for Artificial Intelligence research workers to evaluate their research on robust commercial RTS environment. In year 2010, StarCraft AI Competition was held, in which worldwide AI researchers participated to evaluate their bots against others. Some more motivation factors are:

- It can disable the StarCraft GUI so result of fight between AI bots is based on their AI techniques, rather than human assistance.
- It provides frame by frame replay so each fight can be analyzed thoroughly.
- It has access to all unit types and their weapons.

4 BOT IMPLEMENTATION

Potential fields have been used before in first person shooter game [31] and RTS game [8]. We have not found any research regarding the use of asymmetric potential s in RTS games. So we decided to apply asymmetric potential field on a RTS game bots using Brood War API. Symmetric potential fields have already been applied in the BTHAI project. We have extended the implementation of the BTHAI bot by applying asymmetric potential fields to it. We used Brood War API for the communication of bot with the StarCraft game engine.

4.1 CREATION OF BOT

As we have described before we have extended the BWAPI bot and implemented asymmetric potential fields on it. To apply the asymmetric potential fields on that multi-agent structure we followed the methodology described by Hagelbäck and Johansson [5]. The methodology consists of six steps.

1. Identification of units.
2. Identification of field in scenario.
3. Assigning charges to object.
4. Granularity of the system.
5. Main agent of the system
6. Multi-agent System Architecture

4.1.1 IDENTIFICATION OF UNITS

In this phase we identified the units in scenario. Either these objects are static or dynamic.

In first scenario of our experiments there were 24 units. 12 were own unit and 12 were enemy unit. These units are Dragoons which belongs to Protoss race.

In second scenario of experiment each team has two types of units. These two unit types are Dragoons and Corsair. There are total 14 units on each side, in which 8 units are Dragoons and 6 are Corsairs.

4.1.2 IDENTIFICATION OF FIELDS

We identified three tasks in our experiments.

- Leader guide units to strategically position
- Defend leader
- Destroy enemy

To fulfill these tasks we identified four types of potential fields. These fields are strategic field, leader field, own unit field and enemy field.

Leader field is an asymmetric potential field and generated by leader. This field helps the units to follow leader and because of asymmetric nature of field units stay behind of leader.

Strategic field is an attracting field and generated by specific position. It attract leader to that specific terrain. The purpose is that the leader will guide units to that position and units have enough time to create formation and ready to face enemy.

Own unit field is an asymmetric potential field and generated by own units. This field attracts the leader from the back side when leader is in the MSD of the enemy. The purpose of that field is that leader stay behind of own units and remain safe at the time of fight.

Enemy unit field is generated by enemies. This field is repelling for leader when leader was in MSD of enemy. For remaining own units, this field was attractive when enemies are in MSD. The purpose of that field is that leader remains stay away from enemy and own units attract toward enemy when enemies are in MSD.

4.1.3 ASSIGN CHARGES

In this phase the potential fields of unit were pre calculated. It was also notified that which objects belong to which field.

We had selected four types of fields, so units were assigned charges as described below.

Charges on leader

Leader was generating the asymmetric potential fields. Own units were attracted toward the leader from backside rather than enemy unit.

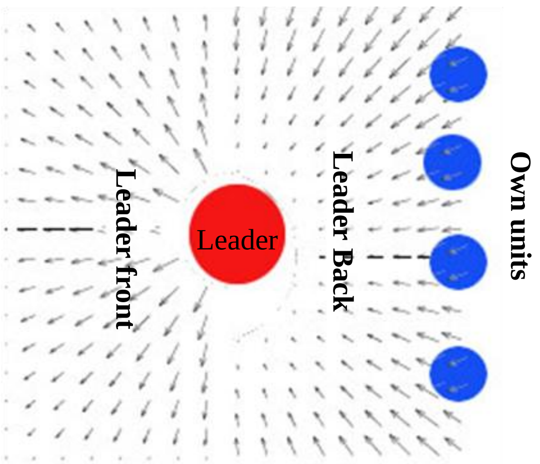


Figure 19: Asymmetric Potential Field on leader

Between leader and own unit there was attractive potential fields which was generated at back side of leader as shown in Figure 19. In Figure19, leader is shown in red colour and own units in blue colour. When leader is not in the MSD of enemy then there was field of attraction on the back of leader. Because of this field of attraction, own unit was attracted toward leader until leader reaches in the MSD of enemy. This field can be calculated as:

$$p(leader) = \begin{cases} 50 * l \ \ if\ d \geq MSD\ of\ enemy \\ 0\ if\ d < MSD\ of\ enemy \end{cases} \quad (4.1)$$

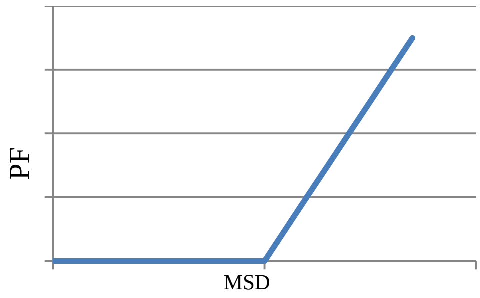


Figure 20: Graphical representation of Formula 4.1

Between the leader and own units repelling potential fields was applied which was generated at front side of leader as shown in Figure 19. When leader is not in the MSD of enemy then there was repelling potential fields in front of leader. This field repelled the own unit so they stayed behind of leader until leader come into the range of MSD. This field can be calculated as:

$$p(leader) = \begin{cases} 0\ \ if\ \ d \leq MSD\ of\ enemy \\ -50 * l\ \ if\ d > MSD + 100 \end{cases} \quad (4.2)$$

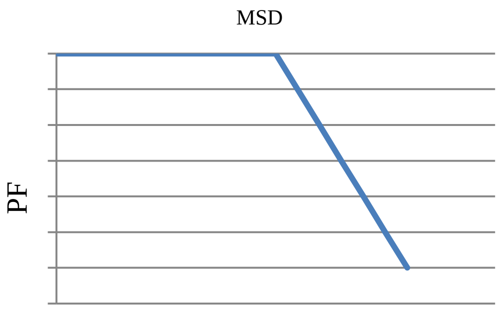


Figure 21: Graphical representation of Formula 4.2

Where d= distance between enemy and leader

MSD= maximum shooting distance of enemy

L = distance between the leader and units.

Strategic Field

In the selected scenario the specific position has attractive potential fields when enemy units were away from the leader of our unit. This field of attraction will lead leader towards the specific position. The force of attraction became zero when leader reached within the range of maximum shooting distance of enemy unit. At that time own units stopped following leader and attracted towards the enemy unit

$$p(strategic) = \begin{cases} 1000 \; if \; d > \; MSD \; of \; enemy \\ 0 \; if \; d \leq MSD \end{cases} \qquad (4.3)$$

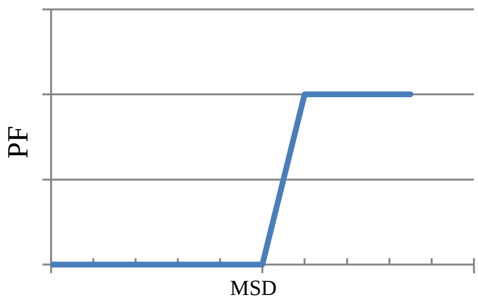


Figure 22: Graphical representation of Formula 4.3

Where d= distance between enemy and leader

MSD= Maximum Shooting Distance of enemy.

p= potential field

Charges on Own Unit

Own unit have repulsive charges, so they will avoid colliding with each other. When leader is within maximum shooting distance of the enemy and the leader starts retreating, then the own units generate an attractive potential fields for leader as shown in Figure23.

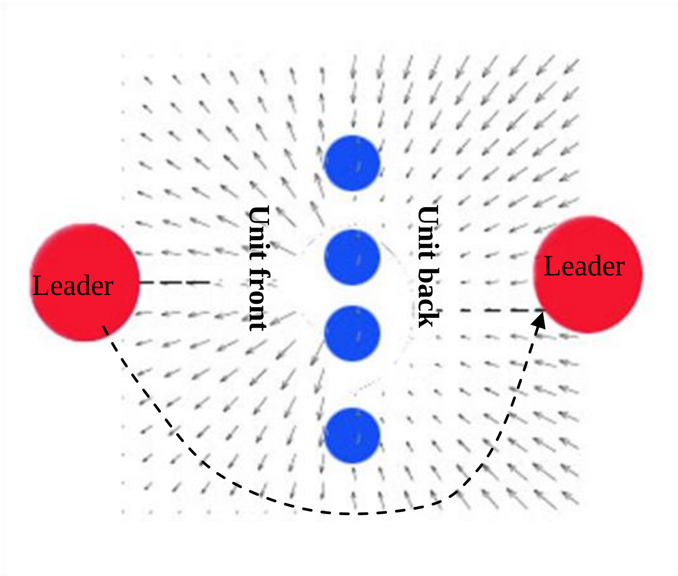


Figure 23: Asymmetric Potential Fields generated by own units

This field of attraction is generated at the backside of the own units, so the leader stays behind its own units until the end of fight.

The attractive potential field for the leader is calculated as:

$$p(own\ unit) = \begin{cases} 0 \quad d \leq MSD\ of\ enemy \\ \frac{l}{50}\ if\ d > \ MSD\ of\ enemy \end{cases} \qquad (4.4)$$

Where l= distance between own unit and leader

d= distance between enemy and leader

Charges on Enemy

Enemy has attractive and repulsive charges on it. When own unit come in the MSD of the enemy then own units attract toward the enemy. The purpose of that field is that own unit attract toward enemies instead of the leader. When leader come in MSD of enemy it repel from enemy so it can move away from enemy.

4.1.4 GRANULARITY OF THE SYSTEM

The resolution of the map is described in this phase. What should be the best pixel size of frame that results in better performance of algorithms?

In the StarCraft default size of tiles is 16*16 (pixel square). In BTHAI 8*8 tiles were used to calculate the potentials on it. So in our both scenarios (Experiment 1 and experiment 2), we used 8*8 tiles and potential fields was updated on every frame.

4.1.5 MAIN AGENTS OF THE SYSTEM

In first experiment, the leader of Dragoons was considered as the main agent and in second experiment Corsair considered as leader. If leader was near enemy then potential values changes. So that’s why we consider leader as the main agent in both experiments.

4.1.6 MULTI-AGENT SYSTEM ARCHITECTURE

Multi-agent architecture was designed in this phase. In this phase more agent were identified which will used to control movement.

In addition to leader, we had introduced one more agent that was between the leader and game server. That agent receives the information about the map and position of enemy, which helped in deciding the movement of leader.

4.2 SOFTWARE REQUIREMENTS FOR DEVELOPING BOT

BTHAI was developed by using the BWAPI 3.0.3. We have decided to use the latest version of BTHAI 1.00 and also latest version of BWAPI which is version 3.3. BTHAI was developed in C++ 2008, so we used visual C++ 2008 express edition to extend that bot.

4.3 Software Requirements for Running Bot

The main requirement which is necessary for run that bot is StarCraft: Brood War expansion, with version 1.16.1, Chaos launcher and BWPAI version 3.3. Chaos launcher is the loading tool. This tool is used to inject the dll (Dynamic-link library) file into the StarCraft process. So we used that tool to inject our bot in StarCraft process.

4.4 Architecture of Bot

We have used the multi-agent based architecture given in Appendix A [23] where we extended its PFManager function to use asymmetric potential fields. The over view of the architecture is given in Section 3.3.

4.4.1 Conceptual View of Architecture

The conceptual view gives the basic idea of the domain of an application with the representation of connectors and conceptual components. The conceptual components are used to represent the different functions of an application and coordination of data and its flow is represented by connectors [24]. Figure 24 represents the conceptual view of our bots.

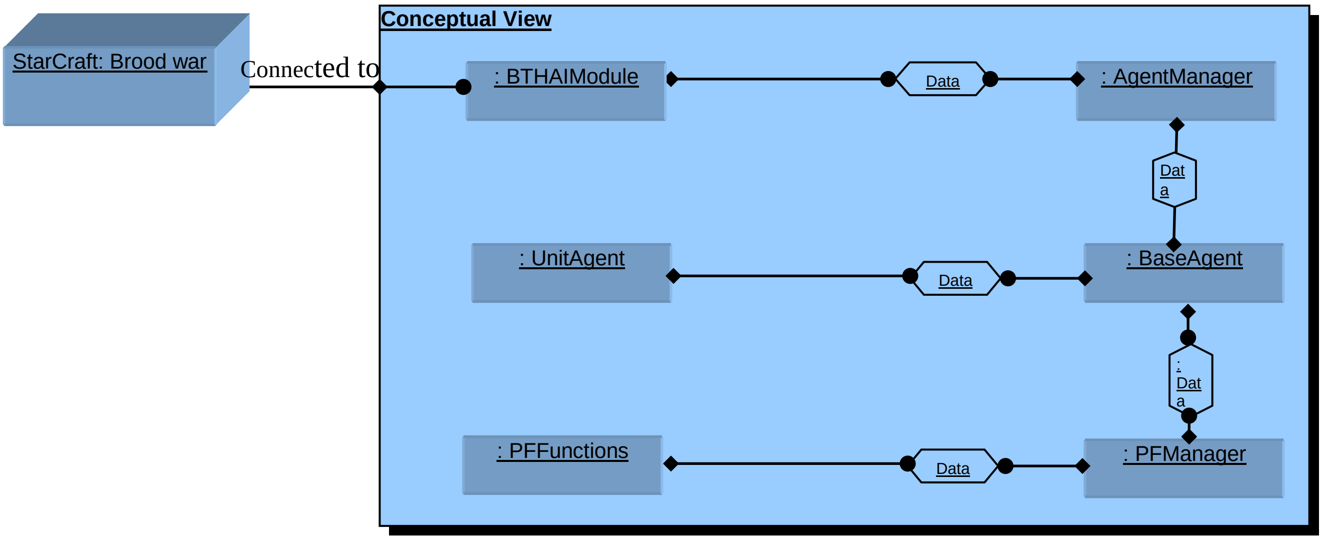
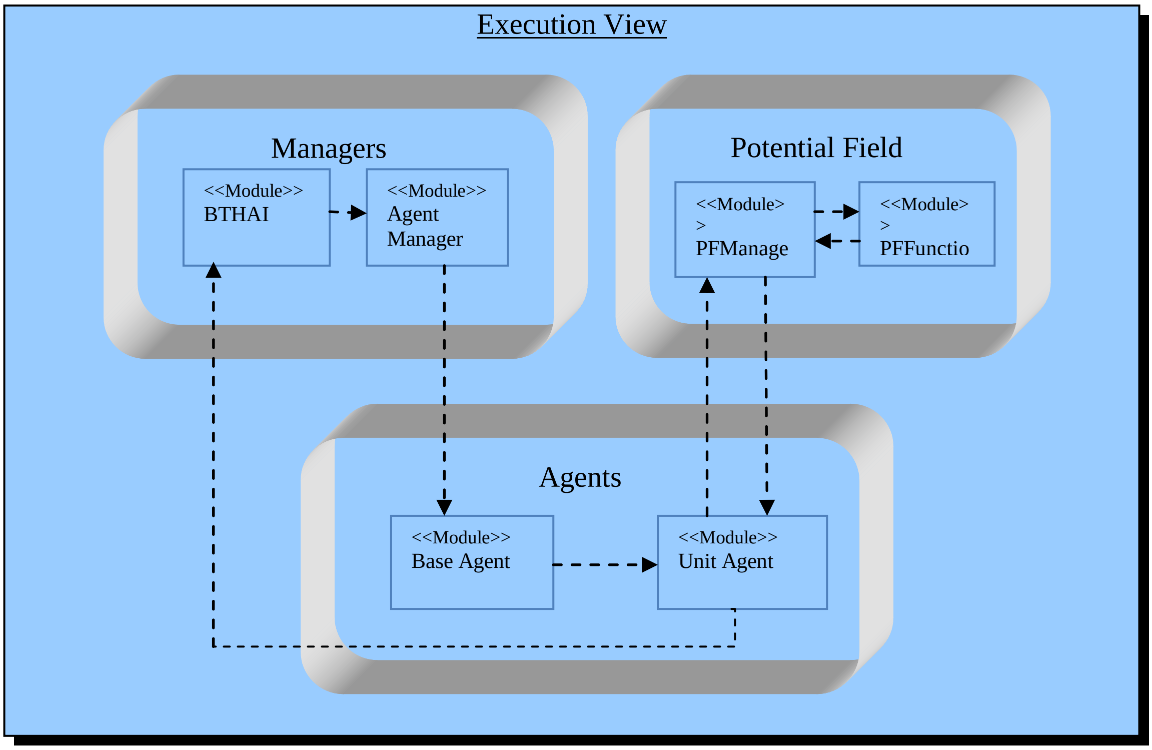


Figure 24: Conceptual View

4.4.2 Execution View of Architecture

The execution view describes the logical flow of controls in an application at run time [24]. The execution view of our bots is mentioned in the Figure 25.


Execution View
Managers
<<Module>>
BTHAI
<<Module>>
Agent
Manager
Potential Field
<<Module>
>
PFManage
<<Module>
>
PFFunctio
Agents
<<Module>>
Base Agent
<<Module>>
Unit Agent

5 EMPIRICAL WORK

Description and motivation regarding experiment as methodology is discussed in methodology Section 1.6.2. Two experiments are conducted. Explanation of all steps of experiments and execution is discussed in this chapter. Results are presented in Chapter 6.

5.1 EXPERIMENT OVERVIEW

We have conducted two experiments to evaluate our study. We have selected two different maps from the StarCraft AI Competition AIIDE2010. These maps are Dragoon Battle and Dragoon Air Battle. The Dragoon Battle map is available on the competition website [19]. We have also modified the Dragoon Battle map to two units map and named that map Dragoon Air Battle map. We have done experiment by developing bots for theses two maps.

5.2 EXPERIMENT NO.1

5.2.1 EXPERIMENT EXPLANATION

To conduct first experiment Dragoon Battle map is used. This map was consist of two units, own unit and enemy unit. Each unit contains 12 Dragoons. In this experiment, we have implemented asymmetric and symmetric potential fields on the bots to compete with other bots on Dragoon Battle map. The Dragoon Battle map is shown in Figure 26.

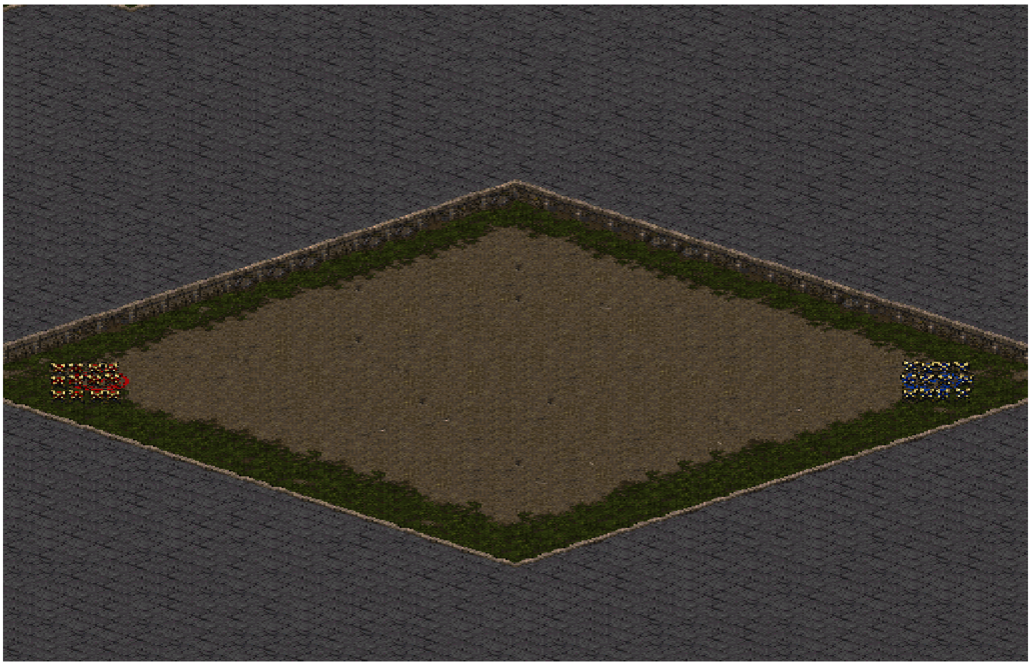


Figure 26: Dragoon Battle Map [28]

The experiment was divided in to two parts. In first part, we used a strategy based on PSO swarm. In order to apply symmetric potential fields we selected one of the Dragoon agents as a leader and that was attracted towards the enemy. The other agents followed the leader by being attracted towards the leader rather than enemy. The agents follow the leader until the enemy becomes visible. Once enemy is visible, our unit agents are attracted by them and start fighting with enemy.

The leader remained in the front of our units. If the leader dies then the agent with more shield and hit points among remaining agents will become the new leader of the group. So in symmetric potential fields there was only field of attraction for the leader and for the agents.

Pseudo Code for Symmetric Potential Fields

```
If (not visible (enemy unit))
{
    Leader attract to (enemy field)
    Agents attract to (leader field)
}
Else if (visible (enemy unit) )
{
    Leader attract to (enemy field)
    Agents attract to (enemy field)
}

If (leader dies)
{
    New leader = own unit (highest hit point +shield)
}
```

In second part of the experiment we applied asymmetric potential field, we again adopted the same strategy in this scenario. A Dragoon agent selected as a leader and attracted toward the enemy. The other agents followed the leader due to force of attraction on leader. The unit attracted towards the leader rather than enemy until enemy unit can be seen. Once enemy unit was seen by agents they stopped following leader and attracted towards the enemy unit and started fighting. At that time the leader of the tribe possessed the repelling field from enemy, so leader kept MSD (Maximum shooting distance) from the enemy unit. At the same time own unit generate attractive field on their back side. This field attracts the leader toward own unit backside, so leader stays behind of the own unit during fight. Asymmetric field for this bot was described in more detail in Section 4.1.

Pseudo Code for Asymmetric Potential Fields

```
If (not visible (enemy unit))
{
   Leader attract to (strategic field)
   Agents attract to (leader back field)
   Agent repelled to (leader front field)
}
```

```
Else if (visible (enemy units))
{
    Leader repelled to (enemy)
    Leader attract to (own unit backward field)
    Agents attracted to (enemy field)
}

If (leader dies)
{
     New leader = own unit (highest hit point +shield)
}
```

Finally we had two bots after the implementation of symmetric and asymmetric potential fields. We named both bots as ASPF and SPF. ASPF bot was with the asymmetric potential fields. We conducted matches to measure the performance of asymmetric potential fields as compared to symmetric potential fields. In matches SPF and ASPF bot competed against the four other bots. One of these bot is built-in computer AI bot of StarCraft game and other three are from the StarCraft competition AIIDE 2010. We used bots from StarCraft competition as the bench mark to validate our results. The names of the bots that used for the matches with ASPF and SPF are Chaos Neuron, Windbot and MSAILBOT.

5.2.2 EXPERIMENT NO.2

In the next experiment we extended our study from one to two units. For this purpose we extended map of Dragoon Battle for the second experiment. The extended map consists of two units from Protoss race. These two units are Dragoon and Corsair. We named this map as Dragoon Air Battle. Figure 27 represents the Dragoon Air Battle map.

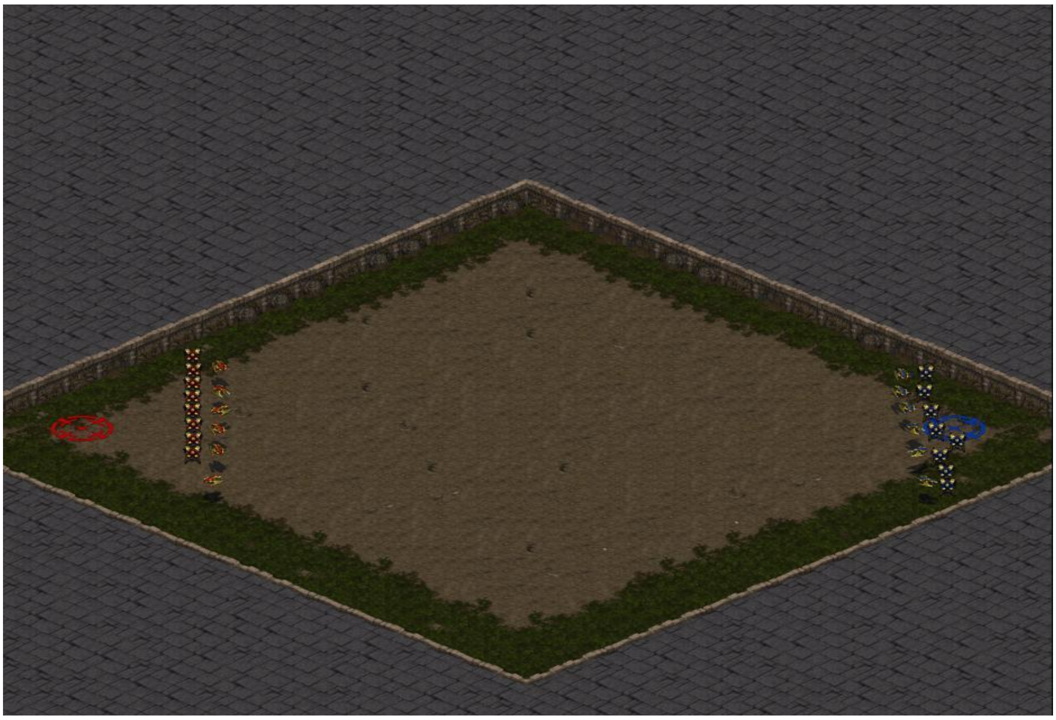


Figure 27 Dragoon Air Battle Map

In this experiment, we examined the asymmetric and symmetric potential field bots on two units. The experiment is divided in to two parts. In first part, we used symmetric potential field bot. In order to apply symmetric potential fields we selected one of the Corsairs as the unit leader and that unit leader attracted towards the enemy units. The remaining Dragoon and Corsair followed the leader until the enemy unit is visible. Once enemy unit can be seen, the Dragoons and Corsairs attracted towards enemy units and started fighting with them. In the second part of this experiment we applied asymmetric potential fields. In order to apply the asymmetric potential fields, we used the same strategy as we adopted in the previous experiment. One of the agents picked as a leader for both units (Dragoons and Corsairs) and both started following the leader. Asymmetric potential field behavior was same as we had discussed before in previous experiment and also explained Section 4.1.in more detail. In Chapter6, we will present the results of experiments and analysis of the results gathered from both experiments and will also discuss the validation of the results.

5.3 Hypothesis Formulation

The hypothesis is the basic idea of the expected outcome of a research. Experiments are being conducted to test the hypothesis. After analyzing the results from experiments, the conclusion can be made that whether our hypothesis is true or it is rejected. The hypothesis formulation of our research is as follows:

5.5.1 Hypothesis

H0: Performance of ASPF bot = Performance of SPF bot.

H1: Performance of ASPF bot > Performance of SPF bot.

5.5.1.1 Performance attributes

We have tested the hypothesis twice using two sets of data:

1. Hypothesis testing using the performance data of the games won.
2. Hypothesis testing using the performance data of the games lost.

In our case performance is based on attributes presented below in Table 2:

Performance Attributes (Games Won)	**Performance Attributes (Games lost)**
Avg. number of games won	Avg. number of games lost
Avg. number of leader killed	Avg. number of leader killed

Table 2 : Performance Attributes

Based on the attributes presented in the table our hypothesis will be divided into following sub-hypothesis.

Hypothesis testing using data from win games:

H0.1 Number of games won by ASPF bot = Number of games won by SPF bot

H1.1 Number of games won by ASPF bot > Number of games won by SPF bot

H0.2 Avg. number of leader killed in ASPF bot= Avg. number of leader killed in SPF bot

H1.2 Avg. number of leader killed in ASPF bot < Avg. number of leader killed in SPF bot

Hypothesis testing using data from lost games:

H0.1 Number of games lost by ASPF bot = Number of games lost by SPF bot

H1.1 Number of games lost by ASPF bot< Number of games lost by SPF bot

H0.2 Avg. number of leader killed in ASPF bot= Avg. number of leader killed in SPF bot

H1.2 Avg. number of leader killed in ASPF bot < Avg. number of leader killed in SPF bot

5.4 VARIABLES SELECTION

There are two types of variables that are used in the experiment. These are independent and dependant variables. Independent variables are those variables that can be changed and controllable in the experiment environment [25].The variables that present the effects of treatments in order to evaluate the performance of techniques used in experiment are termed as dependent variables [25]. In our experiment, the independent and dependant variables are as follows:

5.4.1 INDEPENDENT VARIABLES

- Bots
- Force of potential fields
- Attack strategy

5.4.2 DEPENDANT VARIABLES

- Game completion time
- Number of Enemies Left
- Number of own units Left
- Number of leader killed

5.5 SELECTION OF SUBJECTS

The built-in AI StarCraft bot and three bots that took part in the StarCraft competition were selected as the subjects for the experiment.

Chaos Neuron

Chaos Neuron participated in tournament of the StarCraft and secured third position. This bot used some interesting movement to circle around enemy [28].

Windbot

This bot also participated in the first tournament of StarCraft and well familiar with the StarCraft environment. This is a simple bot and was built on the base of genetic algorithm and give optimistic solution by using the training results [28].

MSAILBOT

MSAILBOT was introduced by the Michigan Student Artificial Intelligent (AI) Laboratory and this bot also participated in the first tournament of StarCraft. MSAILBOT was built with the state based micromanagement artificial intelligence [28].

Built-in AI Bot

This bot is built-in feature of StarCraft. The developers can test their innovation in the AI modules by competing their bots with the built in AI bot.

5.6 EXPERIMENT DESIGN

The experiment design is based on the factors and treatments. The mostly used experiment designs are of four types and are mentioned below [25].

- One factor with two treatments.
- One factor with more than two treatments.

- Two factors with two treatments.
- More than two factors each with two treatments.

The design of our experiment is based on one factor with two treatments. In our experiment Performance is the factor and treatments are ASPF and SPF.

5.7 ENVIRONMENT SETUP

We used visual C++ interface to develop asymmetric potential fields in order to control the movement of units by implementing our strategy. We used StarCraft: Brood War 1.16.1. We developed these bots by using the Brood War API. To carry out experiments we used two personal computers with same specification. The specification of both computers is mentioned below.

- Operating System: Windows XP
- Memory: 3GB RAM
- Hard Disk: 320HD
- Processor Speed: 2.1 GHz

Both computers were connected through LAN. We installed StarCraft: Brood War expansion pack with updated version of 1.16.1 on both computers. Chaos launcher was also downloaded. The computers were configured to run bot by using the instruction given on BWAPI website [22].

5.8 EXECUTION OF EXPERIMENT

There were four bots that participated in the competition with SPF and ASPF. These bots were Chaos neuron, Windbot, MSAIL and built-in AI. Our experiments were conducted in two parts. In the first part we conducted matches of the single unit type on the Dragoon battle map and in the second part we conducted the matches of the two unit types on the Dragoon air battle map. Multiplayer match was created on the maps, so two bots can join game. We divided our experiment in two parts to clarify the purpose of experiment. The details regarding execution of experiments are mentioned below.

In the first part of the experiment, we first injected the SPF bot (implemented with symmetric potential fields) in one computer and conducted matches with the selected bots by injecting these bots one by one in another computer. After the completion of matches of SPF with other bots, we replaced SPF with the ASPF bot (implemented with asymmetric potential fields) and again conducted matches with the other bots by adopting same procedure. There were 100 matches conducted in the first part for both ASPF and SPF bots .The details regarding competition among bots on one unit map are mentioned below.

- 20 matches between SPF bot and Built in AI.

- 10 matches between SPF bot and Chaos Neuron.
- 10 matches between SPF bot and Windbot.
- 10 matches between SPF bot and MSAILBOT
- 20 matches between ASPF bot and Built-in AI.
- 10 matches between ASPF bot and Chaos Neuron.
- 10 matches between ASPF bot and Windbot.
- 10 matches between ASPF bot and MSAILBOT.

In the second phase we conducted matches of SPF and ASPF on the two unit map by following the same procedure adopted in first part of experiment. The details of these matches are mentioned below.

Total 100 matches were conducted in second experiment.

- 20 matches between SPF bot and Built in AI.
- 10 matches between SPF bot and Chaos Neuron.
- 10 matches between SPF bot and Windbot.
- 10 matches between SPF bot and MSAILBOT.
- 20 matches between ASPF bot and Built-in AI.
- 10 matches between ASPF bot and Chaos Neuron.
- 10 matches between ASPF bot and Windbot.
- 10 matches between ASPF bot and MSAILBOT.

In next section, we will discuss the results obtained from the experiment and will interpret results graphically.

6 RESULTS

Results collected from first and second experiments are mentioned in Appendix B and C. Each experiment consists of four competitions. In each competition ASPF and SPF competed with selected bot.

6.1.1 RESULTS OF EXPERIMENT NO.1

First competition

In the first competition ASPF and SPF bot competed with built-in AI bot. The results of matches conducted for ASPF and SPF with built-in AI bot are presented in Tables 1and2 of Appendix B. Summary of the results of first competitions is given in the Table 3.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of enemy unit left	Average number of own unit left
ASPF	70%	1.16	58.60 sec	3.16	4.35
SPF	35%	2.15	50.89 sec	4.61	3.0

Table 3: Overall performance of ASPF and SPF with built in AI

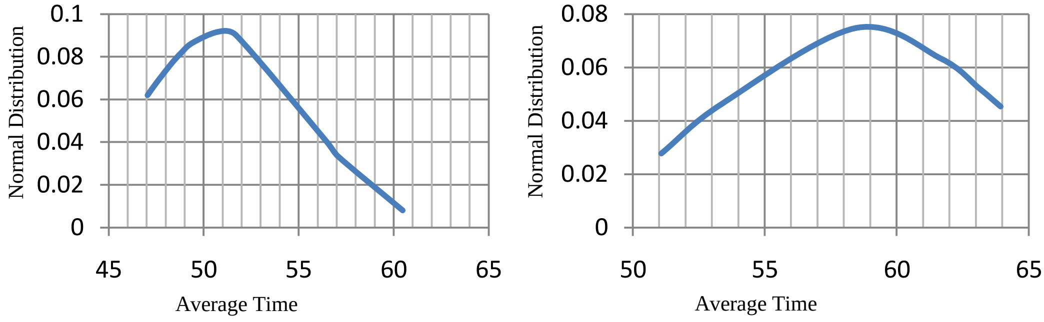


Figure 28: Normal Distribution of Time of lost matches in SPF (left) and ASPF (right)

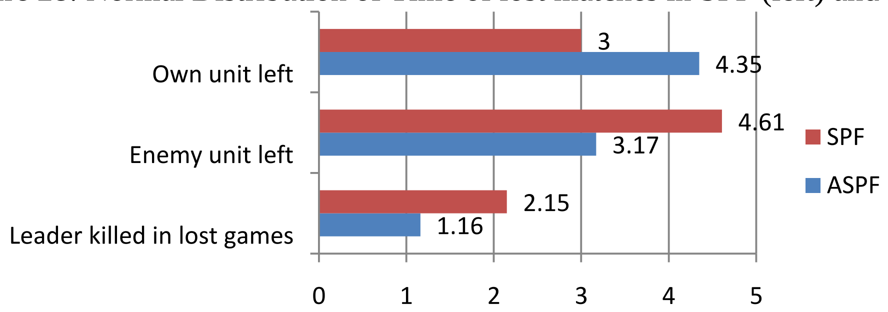


Figure 29: Comparison of ASPF and SPF in lost matches

Summary of Results

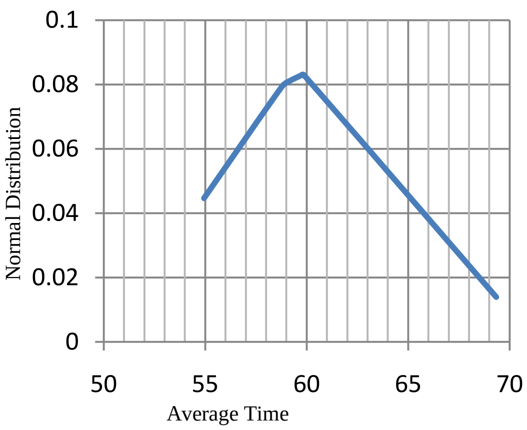
The summarized result of the competition of ASPF and SPF with built in AI mentioned in the Table3 and graphical representation of results shows that the ASPF bot gave better results as compared to the SPF bot against built-in AI. According to Table 3 the win percentage was 70% for ASPF bot and 35% for the SPF bot. From the Figure 29 it was seen that because of the affect of asymmetric potential fields the average number of leader killed for the ASPF bot in lost matches was lower as compared to SPF bot. From Table 3 it was also analyzed that the matches in which ASPF bot could not win against built-in AI bot, it engaged the enemy units for longer time as compared to SPF bot. The average completion times of ASPF and SPF bots in the lost games were 58.60 sec and 50.89 sec shown in Figure 28.

Second competition

In the second competition ASPF and SPF bot competed with Chaos Neuron bot. The data sets of results of second competition are presented in Tables 3and 4 of Appendix B. The summary of the second competition is given in Table 4 and the graphical representation of results is also given below.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of enemy unit left
ASPF	40%	1.33	60.29 sec	5
SPF	0%	2.7	55.4 sec	5.9

Table 4: Overall performance of ASPF and SPF with Chaos Neuron



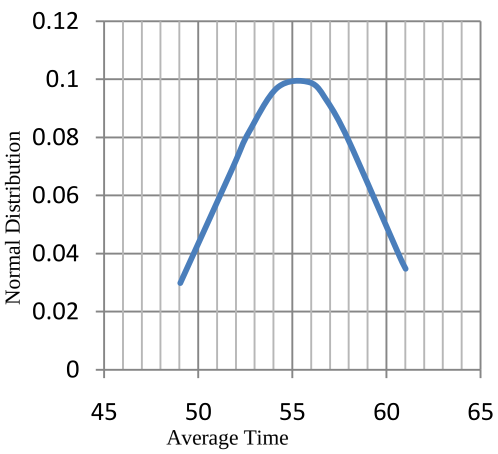


Figure 30: Normal Distribution of Time of lost matches in ASPF (left) and SPF (right)

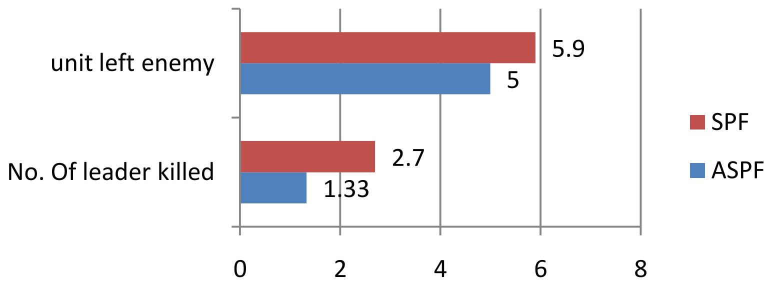


Figure 31: Comparison of ASPF and SPF in lost matches

Summary of Results

The statistics given in Table 4 showed that the performance of the ASPF bot was relatively better than the SPF bot, in the second competition. The ASPF bot won 40% of its matches against Chaos Neuron bot, however SPF bot could not win even a single match. We have noticed that in this competition that the leader affects the overall performance of the unit. It was also noticed from Figure 31 that the average number of leader killed in lost matches for the ASPF bot was 1.33 that was lower as compared to SPF where average number of leader killed in lost matches was 2.7. The normal distribution of average time in lost matches was shown in Figure30 that interprets that bot implemented with asymmetric potential fields engaged enemy unit for longer time as compared to SPF bot.

Third competition

In third competition ASPF and SPF bot competed with Windbot. The data sets of result of matches in the fourth competition are presented in Tables 5 and 6 of Appendix B. The Table 5 presents the summary of the results and Figure 32 and Figure 33 represents the results in graphical form.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of enemy unit left
ASPF	1%	1.77	50.28 sec	6.11
SPF	0	2.7	46.70 sec	7.5

Table 5: Overall performance of ASPF and PSF with Windbot

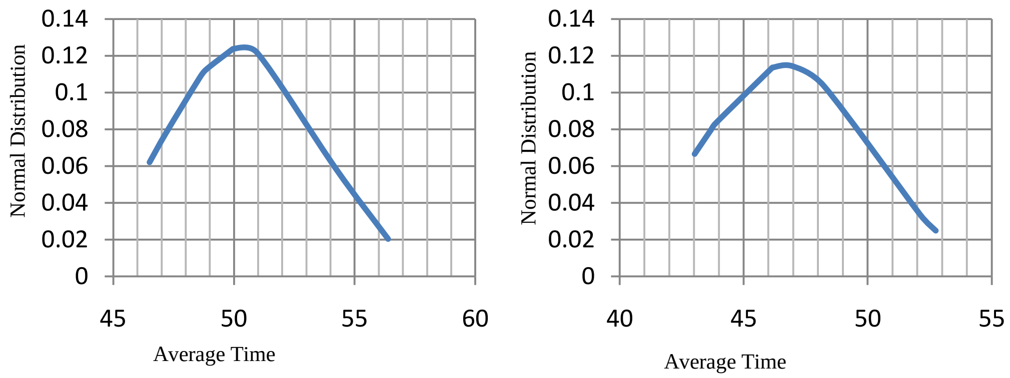


Figure 32: Normal Distribution of Time of lost matches in ASPF (left) and SPF (right)

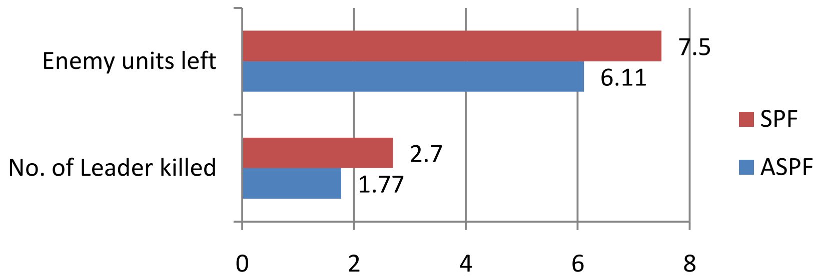


Figure 33: Comparison of ASPF and SPF in lost matches

Summary of Results

We can see from the Figures (32 and 33) and Table 5 containing results that average number of leader killed in lost games for the ASPF are lower than the average number of leader killed in the lost games for the SPF. By analyzing average number of leader killed, time average and average number of enemy unit left shown in Table 5 we concluded that performance of ASPF bot is better than SPF bot against Chaos Neuron.

Fourth Competition

In fourth competition ASPF and SPF bot competed with MSAILBOT. The data sets of results of these competitions are presented in Tables 7 and 8 of index B. The Table 6 shows the summary of the result.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of enemy unit left
ASPF	0	2	51.92 sec	7.9
SPF	0	3.2	47.98 sec	9.6

Table 6: Overall performance of ASPF and PSF with MSAILBOT

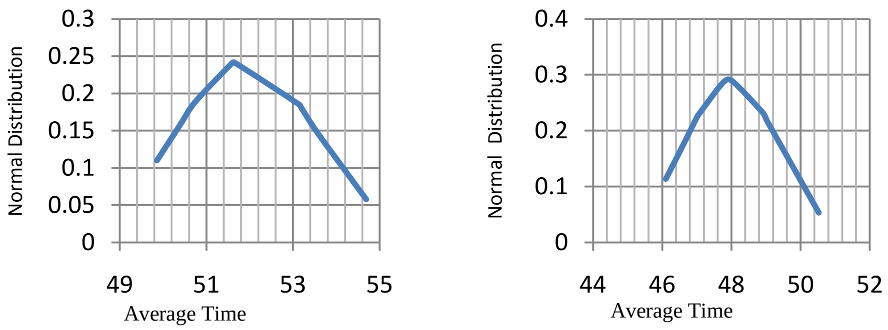


Figure 34: Normal Distribution of Time of lost matches in ASPF (left) and SPF (right)

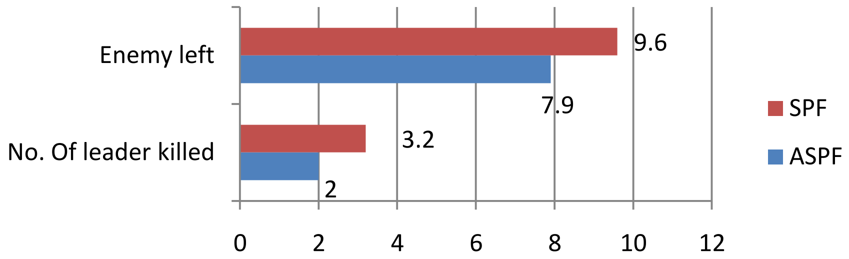


Figure 35: Comparison of ASPF and SPF in lost matches

Summary of Results

In the fourth competition both bots (ASPF and SPF) could not win any match against MSAIL bot. But by analyzing the statistics given in Table 6, it has been noticed that even though ASPF bot could not win the matches against MSAIL bot but it engaged the enemy unit for longer time as compared to SPF. It was also noticed from Figure 35 that average of leader killed in the ASPF bot is less as compared to the average of leader killed of SPF. The average numbers of enemy units left in the winning matches against ASPF are also lesser than the enemy units left in the winning matches against SPF.

6.1.2 Results of Experiment No.2

In the second experiment, we conducted matches of SPF and ASPF bots on the map containing two units. The data set containing results of competitions are presented in Appendix C. The summary and analysis of results is mentioned below.

First competition

In first competition ASPF and SPF bot competed with built-in AI bot. The data sets containing result of first competition are presented in Tables 1and2 in Appendix C. The Table 7 presents the summary of results of first competition.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of own unit left	Average number of enemy unit left
ASPF	65%	1.0	60.25 sec	4.69	2.6
SPF	20%	2.5	49.08 sec	4.0	5.5

Table 7 : Overall performance of ASPF and SPF with built in AI

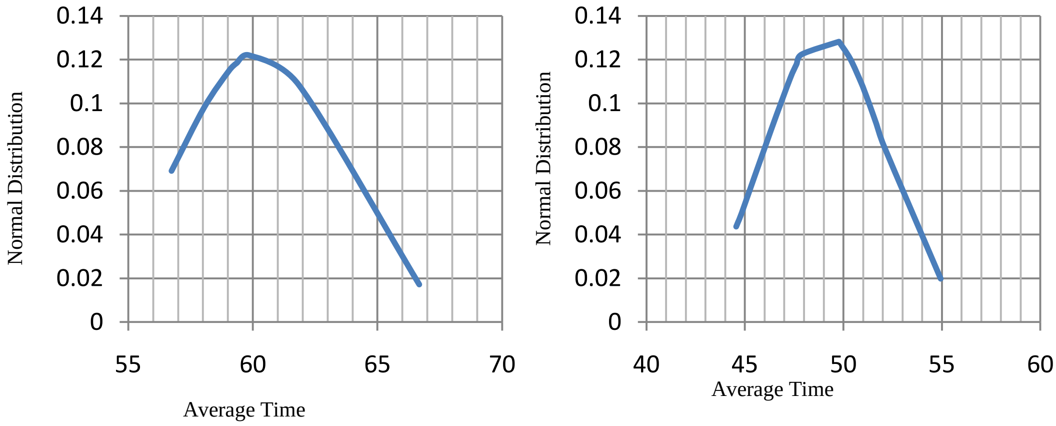


Figure 36: Comparison of average completion time of ASPF (left) and SPF (right) in lost matches.

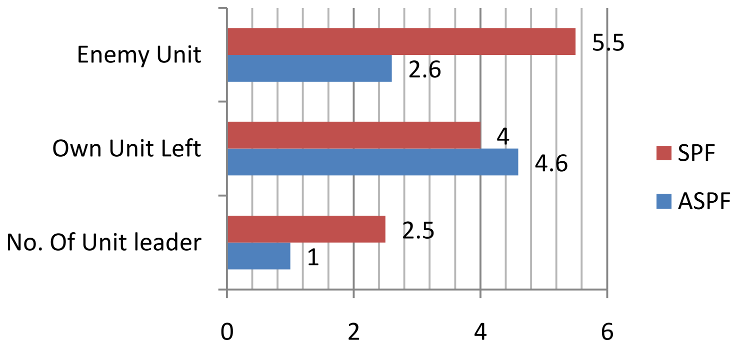


Figure 37: Comparison of ASPF and SPF in lost matches.

Summary of Results

The summarized result of the competition of ASPF and SPF with built in AI mentioned in the Table 7 shows that Asymmetric approach has better results and it won 65% of its matches against built-in AI. Due to asymmetric potential fields the average number of leader killed in ASPF matches was 1.0 as shown in Figure 37 which was lower as compared to the average number of leader killed in the SPF matches against built in AI, where average number or leader killed was 2.5. It was also noticed from the Figure 36 that in those matches where ASPF bot could not win against built-in AI bot, it engaged the enemy units for longer period of time i.e. 60.25 sec than the SPF bot i.e. 49.08 sec.

Second competition

In the second competition ASPF and SPF bot competed with Chaos Neuron bot. The results of second competition are presented in Tables 3and4 of Appendix C. The Table 8 shows the summary of these results.

Bots	Win ratio	Average no. of leader killed in lost matches	Time average in lost time	Average number of enemy unit left
ASPF	50%	1.0	66.29 sec	5.2
SPF	20%	2.9	56.33 sec	6.8

Table 8 : Overall performance of two ASPF and PSF with Chaos Neuron

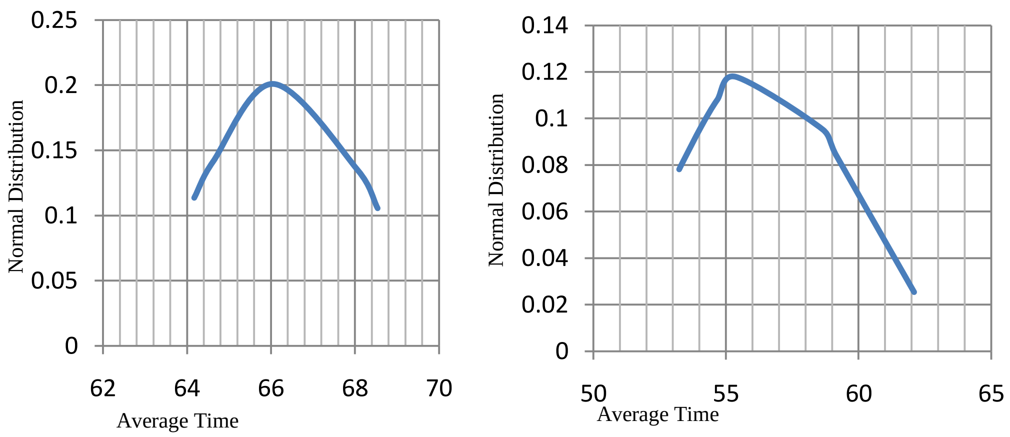


Figure 38: Comparison of ASPF (left) and SPF (right) according to average completion time of match

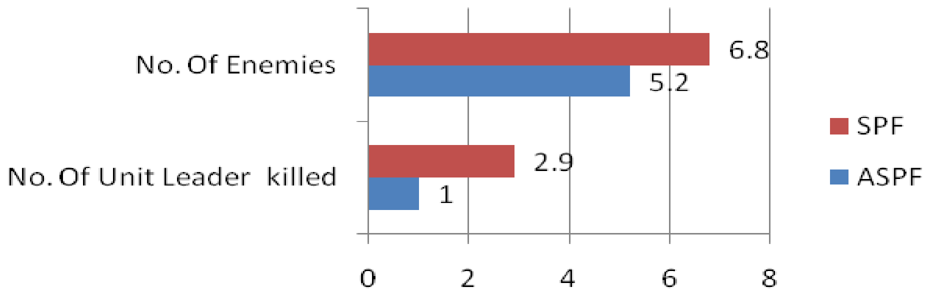


Figure 39: Comparison of ASPF and SPF in lost matches

Summary of Results

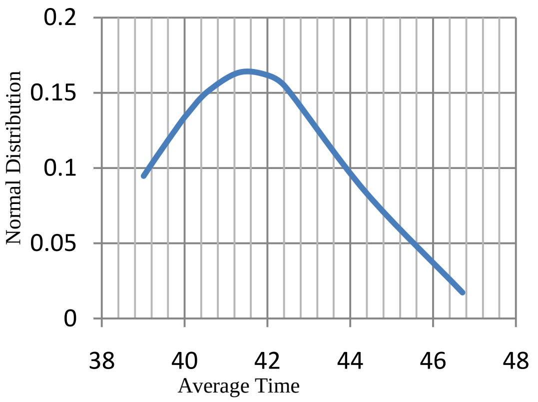
The results of second competition show that ASPF performance is quite better than SPF as shown in Table 8. ASPF won 50% of its matches against the Chaos neuron however SPF won 20% of its matches. Figure 38 shows that the average time of lost matches was also greater in ASPF compared to SPF. The average completion time for the ASPF bot was 66.29 sec in the lost matches against chaos neuron. However in the lost matches against chaos neuron the average time of the SPF bot was 56.33 sec. We can say that overall performance of ASPF was better than SPF against Chaos neuron as ASPF bot won more games compared to SPF bot and also performed well in the matches where ASPF bot could not win as shown in Figure 39.

Third competition

In the third competition, we conducted matches of ASPF and SPF against Windbot. The data sets of results of these competitions are presented in Tables 5and6 of Appendix C. The summary of the results is shown in the Table 9.

Bots	Win ratio	Average no. of leader killed in win matches	Time average in win matches	Average number of unit left in win matches
ASPF	100%	0.4	41.55 sec	7.1
SPF	50%	1.2	47.94 sec	4.4

Table 9 : Overall performance of ASPF and PSF with Windbot



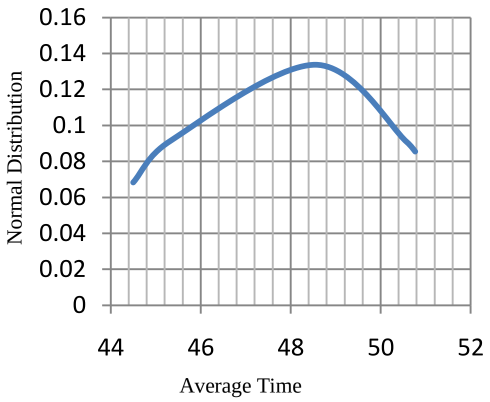


Figure 40: Comparison of average completion time of ASPF (left) and SPF (right) in won matches.

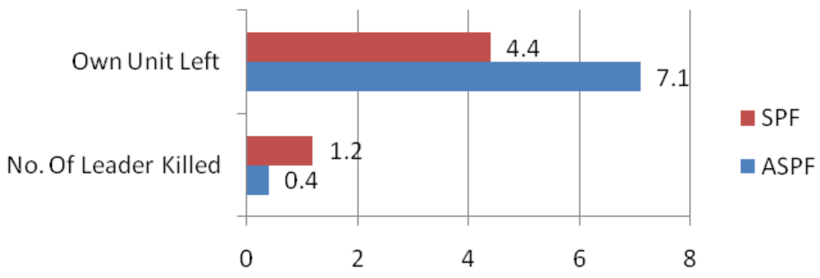


Figure 41 : Comparison of ASPF and SPF in won matches

Summary of Results

Table 9 shows that in third competition ASPF bot won its all matches and SPF managed to win only 50% of its matches against Windbot. It was also noticed that ASPF bot average completion time of winning games was 41.55 sec, which was better than SPF's average completion time 47.94 sec of winning games. We also analyzed that ASPF bot has less average number of leader killed 0.4 in win matches as compared to average number of leader killed 1.2 of SPF matches as shown in Figure 41.

Fourth competition

In fourth competition ASPF and SPF bot competed with MSAILBOT. The data sets of result of these competitions are presented in Tables 7and8 of Appendix C. The Table 10 shows the summary of that result and the graphic representation of comparison of results are shown in the Figure 42 and Figure 43.

Bots	Win ratio	Average no. of leader killed in win matches	Time average in win matches	Average number of unit left in win matches
ASPF	100%	0.90	46.85 sec	5.7
SPF	60%	1.5	52.34 sec	4.7

Table 10 : Overall performance of ASPF and PSF with MSAIL bot

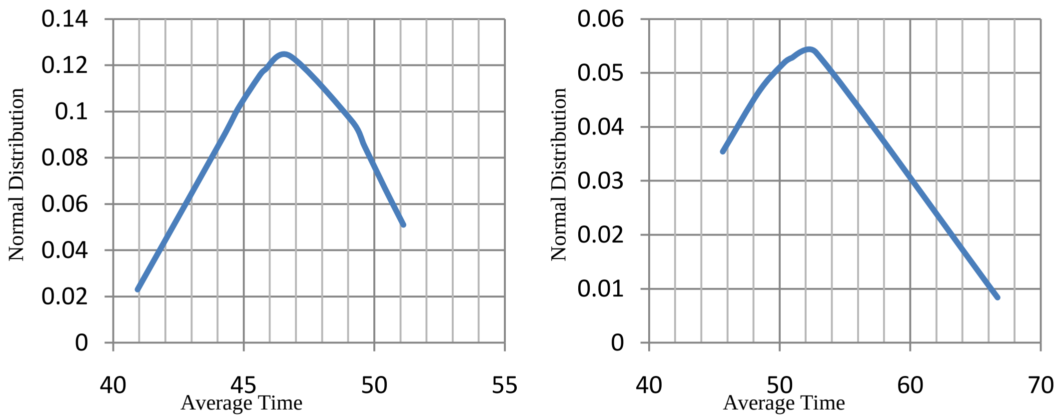


Figure 42: Comparison of Average time in won matches of ASPF (left) and SPF (right)

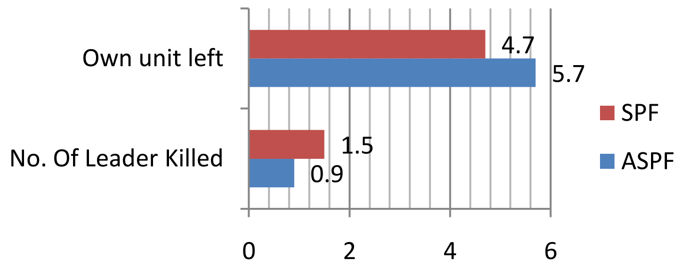


Figure 43: Comparison of SPF and ASPF in won matches

Summary of Results

The result of fourth competition in Table 10 shows that performance of ASPF bot is better than SPF in matches against MSAILBOT. ASPF won all its matches. The win percentage of ASPF bot was 100% whereas the win percentage of SPF bot was 60% as shown in the Table 10 In these matches average number of leader killed shows that leader was almost killed in every match in ASPF, but still its average was better than SPF as shown in Figure 43 .We can see from the Figure 42 that the average time of completion for the ASPF bot in win matches was 46.85 sec, where the average time of completion for SPF bot in win matches was 52.34 sec.

6.2 HYPOTHESIS TESTING

The t-test was used to verify the hypothesis. This test was used for finding the significant difference between the mean of two sample values [25] [34]. The reason of selecting the t-test is that our experiment design is one factor and two treatments. According to Wohlin [25] for this type of design t-test is suitable.

		Win Matches		Loss Matches	
	Bots	Average Matches win	Average no. of leader killed	Average No. of Matches lost	Average no. of leader killed
Competition 1	ASPF	0.70	0.07	0.30	1.16
	SPF	0.35	0.57	0.65	2.15
Competition 2	ASPF	0.40	0.25	0.60	1.33
	SPF	0	-	1.00	2.7
Competition 3	ASPF	0.10	1	0.90	1.77
	SPF	0	-	1.00	2.7
Competition 4	ASPF	0	-	1.00	2
	SPF	0	-	1.00	3.2

Table 11: Summary of experiment 1 results

		Win Matches		Lost Matches	
	Bots	Average Matches win	Average No. of leader killed	Average No. Of Matches lost	Average no. Of leader killed
Competition 1	ASPF	0.65	0	0.35	1
	SPF	0.20	2.25	0.80	2.56
Competition 2	ASPF	0.50	0	0.50	1
	SPF	0.20	2	0.80	2.87
Competition 3	ASPF	1.00	0.4	0	-
	SPF	0.50	1.2	0.50	3
Competition 4	ASPF	1.00	0.90	0	-
	SPF	0.60	1.5	0.40	2.25

Table 12: Summary of experiment 2 results

Result of t-test on selected performance attributes from experiment 1 is mentioned in Appendix D. The summary of t-test result is presented in Table 13 and Table 14 .The summary of t-test in lost matches shows that there is significant difference between ASPF and SPF in average number of leader killed in lost matches in competition 1and 2. In competition 3 and 4 ASPF and SPF fail to win any match. But number of leader killed shows that there was significance different in ASPF and SPF performance. In Table 14 summary of t-test on the result of experiment is presented. This result shows that there is a significant distance between ASPF and SPF in competition 1. In competition 2 SPF could not win any single match, due to this reason t-test was not applied on it. From Table 11 it can be noticed that ASPF won a match in second competition, on basis of this result we can conclude that ASPF performance is better than SPF. In competition 3 and 4 ASPF and SPF could not win single match.

		Competition 1	Competition 2	Competition 3	Competition 4
Average no. of leader killed in lost matches	**T stat**	4.352	4.555	2.760	3.674
	P Two Tail	0.0007	0.0005	0.0162	0.0017
	T critical	2.160	2.160	2.160	2.100
lost	**T stat**	2.306	2.449	0.1717	-
	P Two Tail	0.0266	0.0367	0.3434	-
	T critical	2.024	2.262	2.262	-

Table 13: Summary of t-test in loss matches in experiment 1

		Competition 1	Competition 2	Competition 3	Competition 4
Average no. of leader killed in Win matches	**T stat**	2.333	-	-	-
	P Two Tail	0.047	-	-	-
	T critical	2.306	-	-	-
Won	**T stat**	2.306	2.449	0.1717	-
	P Two Tail	0.0266	0.0367	0.3434	-
	T critical	2.024	2.262	2.262	-

Table 14: Summary of t-test in win matches in experiment 1

Result of t-test on selected attributes from experiment 2 is mentioned in Appendix E. In Table 15 and Table 16, summary of t-test result shows that there is significant difference in means of selected attributes. From Table 15 and Table 16 we can notice that there was significant difference in ASPF and SPF in the first competition. In competition 2 there is no significant difference in number of won and lost matches. But number of leader killed in lost matches has significant difference in competition 2. In competition 2, the result of leader killed in lost matches has standard deviation 0, so t-test cannot be applied on it. But from Table 12 it can be noticed that in ASPF leader was not killed even for a single time and in SPF its average is 2. It can also be noticed from Table 15 and 16 that there is a significant difference in ASPF and SPF bots in competition 3 and 4.

T-test was performed on 0.05 probability level, so we can conclude that the chance of error in our result was less than 5%. We can say with 95% level of confidence that there was significant difference in mean of attributes in both experiments. Hence we reject null hypothesis in competition 1 and 2 in experiment 1. For competition 3 and 4 we reject null hypothesis that average number of leader killed in lost matches are equal, but we accept the null hypothesis that number of lost matches are equal. In experiment 2 we reject null hypothesis in competition 1, 3, and 4. In competition 2 we accept null hypothesis that there is no significant difference between the number of matches win in ASPF and SPF. But number of leader killed shows that there is a significant difference in performance of ASPF and SPF. On the basis of results presented in Tables 11 and 12, we can accept alternative hypothesis that ASPF performance has been increased as compared to SPF.

		Competition 1	**Competition 2**	**Competition 3**	**Competition 4**
Average no. of leader killed in lost matches	**T stat**	12.198	8.275	-	-
	P Two Tail	3.45×10^{-9}	7.34×10^{-5}	-	-
	T critical	2.0342	2.364	-	-
lost	**T stat**	5.94	1.405	3	2.449
	P Two Tail	1.02×10^{-5}	0.177	0.0149	0.0367
	T critical	2.093	2.109	2.262	2.262

Table 15: Summary of t-test in loss matches in experiment 2

		Competition 1	**Competition 2**	**Competition 3**	**Competition 4**
Average no. of leader killed in won matches	**T stat**	4.700	-	3.098	2.449
	P Two Tail	0.0182	-	0.0127	0.044
	T critical	3.182	-	2.2621	2.364
Won	**T stat**	3.151	1.405	3	2.449
	P Two Tail	0.003	0.176	0.0149	0.0367
	T critical	2.024	2.1009	2.262	2.262

Table 16: Summary of t-test in win matches in experiment 2

6.3 VALIDITY THREATS

According to Cook and Campbell there are four types of validity threats for experiment [25]. These threats are conclusion, internal, construct and external validity.

6.3.1 CONCLUSION THREAT

This threat is related to the act of drawing conclusion that is not backed up by the experimental results [25].

In our experiment we made a performance comparison between two bots implemented with symmetric and asymmetric potential field. There was a threat that the faulty outcome of results may affect the conclusion of the experiments. There was also a threat that we got that result by chance. We performed t-test to avoid this threat and to check the significance difference between mean of performance both techniques.

6.3.2 Internal Threat

These threats are usually related to the execution of experiment and can disturb the experiments if these are not in the researcher's knowledge [25].

We have identified that system configuration and CPU utilization can also affect the bot performance during the competitions. To minimize this threat we configured both bots on the computers of same specification with the same configuration. We also monitored CPU performance during the experiment. We excluded all those result in which processor was consuming 100% of its processing power.

6.3.3 Construct Validity

This threat is related with the experiment setting and choices of parameters in experiment [25].

In this experiment the bot development language can effect on the performance of bot and can cause the delay in issuing the commands. The StarCraft bot can be developed in many languages, but they required proxy server to communicate with BWAPI which was developed in the Visual C++. So this communication can result in slow performance of bot due to compatibility issue. To avoid this threat we decided to develop bot in the Visual C++ environment.

One more threat which can arise is related with the selection of bots from which our ASPF bot has to fight. There are many bots available for StarCraft, but randomly selection of bots could affect the validation of results. To reduce this threat we decided to select the bot from official StarCraft AI competition [28]. All the bot we selected are developed by universities

6.3.4 External Validity

The threats related to the factor that can affect the experiment result outside the experiment environment [25] .

In this experiment, the bot that was implemented with asymmetric potential fields was capable to compete for specific type of race and map. This bot can be configured to adjust with other races, but this can effect on the result of experiment.

7 DISCUSSION

In the first experiment we have analyzed the affects of asymmetric potential fields on the single unit type to compare its performance against the symmetric potential fields. Figure 44 shows that the win percentage of ASPF in experiment one was significantly better compared to the win percentage of SPF. In the first experiment, we have noticed that because of asymmetric potential fields leader moved to safe location and remaining unit cover the leader. Because of that movement in asymmetric potential field average number of leader killed in all matches of ASPF bot was lower than average number of leader killed in SPF bot as shown in Figure 45.

In first experiment ASPF bot could not win matches against the Windbot and MSAILBOT. This is because of the better target picking strategy used by Windbot and MSAILBOT. In our asymmetric potential fields bot we focused on the movements of unit, and didn't apply any target picking strategy. From Table 3 it was analyzed that ASPF bot won 40% of its matches against chaos neuron bot. Chaos neuron was using special tactics, and try to circle the opposition. We have noticed that in those matches in which ASPF bot won against Chaos Neuron, ASPF bot was able to break the formation of Chaos Neuron units due to the movement of leader before they surround ASPF bot. But in case of SPF bot, SPF bot failed to break the formation of Chaos Neuron and could not win even a single match against it.

Number of matches won and average number of leader killed shows that asymmetric bot performance is better than symmetric potential bot in experiment 1. Beside these two factors some other factors which are mentioned in summary of results in Chapter 6 are game completion time and average number of units left in loss matches. In all results the average time completion in ASPF is better than time in SPF. It shows that even in loss matches ASPF bot engaged enemy for more time as compared to SPF bot.

.

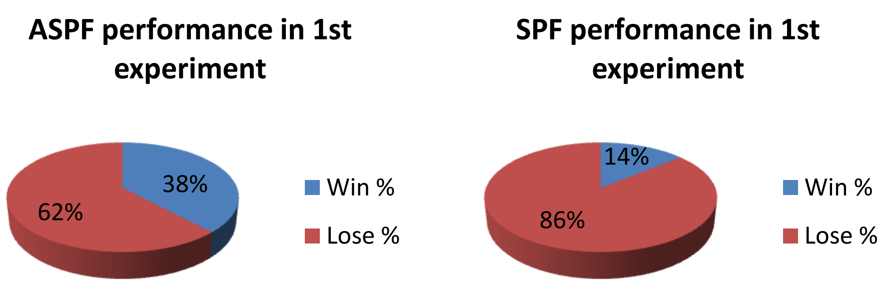


Figure 44 : Comparison of win percentage of ASPF and SPF

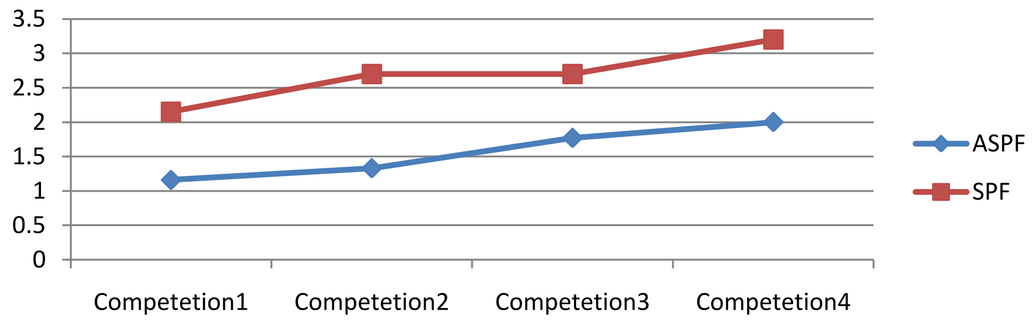


Figure 45: Comparison of ASPF and SPF on average no. of leader killed in Experiment 1

In the second experiment the affect of asymmetric potential fields was analyzed on two unit types. We analyzed that the performance of the ASPF bot on two unit types was better regarding the win ratio, the average number of leader killed, the average number of units left and the average completion time as compared to SPF bot. The win comparison of ASPF and SPF against other bots in the second experiment is shown in the Figure 46. In Figure 47 comparison of average number of leader killed in ASPF bot and SPF bot is shown. This figure shows that in ASPF bot leader killed less number of time as compared to SPF bot. ASPF managed to win all matches against MSAILBOT and Windbot and also showed good results against built-in AI and Chaos Neuron bot. MSAILBOT used the strategy of attacking the first unit it saw. In ASPF and SPF leader was air unit type. The leader is in the front and it was the first unit which is detected by MSAIL BOT. So the units of the MSAIL bot attracted toward the leader. In ASPF because of asymmetric potential fields leader move back and try to reach safe position that is behind the own units. Units of MSAIL bot follow the leader and neglect other units of ASPF. At this stage ASPF bot got benefit and can easily attack and destroy the MSAILBOT unit. On other hand SPF bot also gave good result against MSAILBOT but could win only 50% of its matches. Windbot was using the strategy of attacking the nearest units first. During the fight air unit types (Corsair) were closer to the enemy compared to the ground unit types (Dragoons), so all units of Windbot became busy with the ASPF air unit type, which gave advantage to ASPF ground unit types to fire on enemy freely.

The results of both experiments show that performance of ASPF bot has been increased as compared to SPF bot.

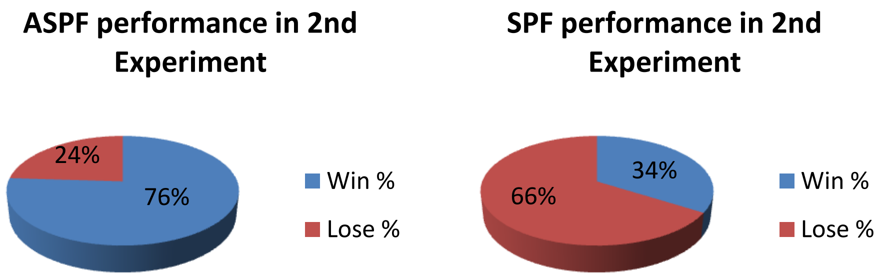


Figure 46 : Comparison of win percentage of ASPF and SPF

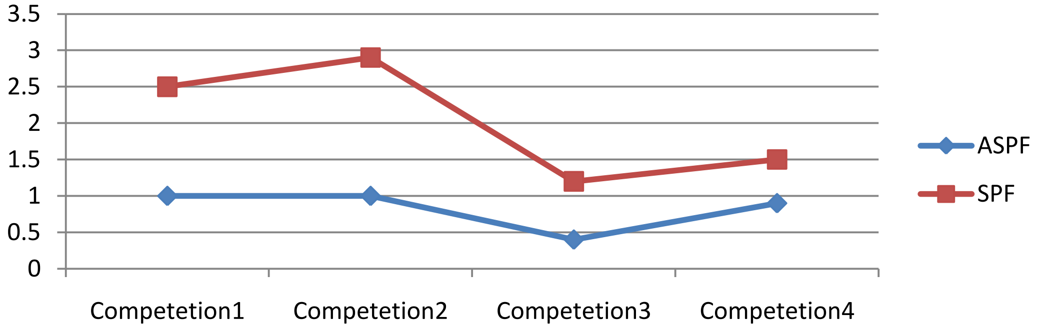


Figure 47: Comparison of ASPF and SPF on average no. of leader killed in Experiment 2

8 CONCLUSION

We presented our study regarding the implementation of asymmetric potential fields in RTS games in this report. We used the visual C++ for the implementation and used Brood War API for the communication of ASPF bot with the StarCraft environment.

The intention behind study was to use asymmetric potential fields in the game environment to observe their effects in comparison to symmetric potential fields regarding the unit's formation.

Potential field is a technique that is used for the navigation of objects in the game environment in order to achieve the natural movements. We extended this work by using the asymmetric potential fields in scenarios of one and two types of units. By using asymmetric potential fields we achieved the repelling and attracting behavior of units at the same time. We conducted matches of bots implemented with the symmetric and asymmetric potential fields against the bots who participated in the first tournament of StarCraft competition. We used these bots as a bench mark for the validation of results. We analyzed that the bot with asymmetric potential fields performed better as compared to the bot with symmetric potential fields regarding the completion time, win ratio and number of leader killed. We validated the result by t-test which proves that there was significance difference between the performance of the asymmetric potential field bot and symmetric potential field bot.

In our experiments we used the same strategy for the single unit type and two unit type. Our strategy was based on selecting an agent from the units as a leader. The leader guides the other agents toward enemy units. This research can be enhanced by making a leader from each unit type, this may results in better unit's formation and results can be improved.

8.1 CHALLENGES IN IMPLEMENTATION OF ASPF

- The study regarding the usage of potential fields in the RTS games is novel. There was a limited research done in this area. A very few people had research regarding the usage of potential field in the real time strategy games. As we have discussed before that there was not any research regarding the implementation of asymmetric potential field. So there was a challenge for us to find out the methodology for the implementation of asymmetric potential fields.
- Another challenge was to find a formation of units to protect leader by using asymmetric potential fields.
- The third challenge was to check the effectiveness of the asymmetric potential fields in the robust commercial environment to validate the study.

8.2 HOW TO OVERCOME CHALLENGES IN IMPLEMENTATION OF ASPF

- We thoroughly studied the articles related to the implementation of symmetric potential fields. Symmetric potential field is related to the distribution of same charges on a single object in the game environment. However the asymmetric potential field is related to the distribution of positive and negative charges on the single object at the same time. We found the methodology for the implementation of asymmetric potential fields in the RTS games with the help of methodology proposed by Hagelbäck and Johansson for the symmetric potential fields in RTS games. The methodology used for the implementation of asymmetric potential field is presented in Chapter 4.

- We needed a strategy to implement the asymmetric potential fields on the units in order to obtain the interesting unit formation. It was a big challenge to find out a way to make interesting unit formations because in asymmetric potential field negative and positive charges are used at a same time on the single object. The challenge was to find out the strategy and by using that we can put the positive and negative charges on agents at same time and can obtain unit formations to protect leader in fighting with enemy.
- We came across a strategy to pick up the main agent as a leader from a unit type. The leader possesses the asymmetric potential field at the same time. When the enemy is visible to leader and units, leader will be repelled from the front of the unit is and will be attracted from the back of units. The units will be attracted towards the enemy and the leader will be protected in this way. This strategy resulted interesting formation of the units while fighting with the enemies. We used the same strategy on the single and two unit types. We have explained in detail in the Chapter 5 under sections 5.2.1 and 5.2.2.
- In matches SPF and ASPF bot competed against the four other bots. One of these bot is built-in computer AI bot of StarCraft game and other three are from the StarCraft competition AIIDE 2010. We used bots from StarCraft competition as the bench mark to validate our results. The names of the bots that used for the matches with ASPF and SPF are Chaos Neuron, Windbot and MSAILBOT.

8.3 ANSWERS TO RESEARCH QUESTIONS:

RQ1: How can be asymmetric potential fields used in RTS games?

To answer this question we gone through the related studies and found methodology by which we can implement asymmetric potential field in RTS game. Chapter 4 presents the methodology to use asymmetric potential field in the RTS games.

RQ2: How to implement asymmetric potential fields that protect a leader with dynamic unit formations while combating enemies in RTS games?

The answer of RQ2 is given in detail in the Chapter 4 and Section 5.5. We have used the methodology of Hagelbäck and Johansson to implement asymmetric potential fields in the game environment. We implemented a strategy to protect leader and also applied asymmetric fields on the single and two types of units.

RQ3: To what degree asymmetric potential fields is effective as compared to symmetric potential fields?

By analyzing the results collected from the experiment 1 and 2, we conclude that Asymmetric potential fields has better effect on the objects regarding their movement and formation in the game environment as compared to the symmetric potential fields. The answer of this research question is given in more detail in Chapter no.6

8.4 CONTRIBUTION

Asymmetric potential fields were not previously implemented in the RTS games. In our research we found out the methodology to implement the asymmetric potential fields in the real time strategy game. We implemented asymmetric potential fields on the single and two unit types of StarCraft game. We also implemented strategy to assign leader and how leader can be protected while combating with enemies. It was also stated in our research that because of asymmetric potential field leader moved to safe location that resulted in better performance compared to symmetric potential field bot.

9 FUTURE WORK AND LIMITATION

As discussed before the concept of using potential fields in the RTS game is quite new and limited research had been done. We have implemented the asymmetric potential fields on small scenario with limited functions. We did not work on target picking so result can be improve by working on it. Because of limited resources we did not apply asymmetric potential field on a full Real time strategy game scenario.

There is more research required in the area and following are some suggestions:

- Asymmetric potential fields can be used in producing strategies at runtime by analyzing enemy formation, so we can attack on enemy's weak points. Currently most of the bots are using static strategies so it is a good idea to create strategies by analyzing the enemy movements. By adopting this type of strategy may result into some interesting behavior which may close to human like behavior.
- In our study, we applied asymmetric potential fields on the one and two type of units of the same race. In future interesting results can be obtained by implementing the asymmetric potential fields on the more than two types of units. Some interesting behavior of units can be achieved by applying asymmetric potential fields on the complete scenario of RTS strategy.
- The affect of asymmetric potential fields can be checked in the game environment by implementing it with the some target picking strategy.
- In our strategy we have assigned one leader which guided all units. In future this strategy can be expanded and can assign leader from each unit type, which guides the unit of its type. This can improve the result and more interesting movement can be noticed.

10 REFERENCES

[1]. M. Buro. “Real-time strategy games: A new AI research challenge,” in *Proc. of the Eighteenth International Joint Conference on Artificial Intelligence*, 2003, pp.1534-1535.

[2]. S. Ding, N. Tang, T. Lin and S. Zhao. “RTS-GameFlow: A New Evaluation Framework for RTS Games,” *International Conference on Computational Intelligence and Software Engineering*, 2009, pp.1-4.

[3]. J. Hagelbäck and S.J. Johansson. “The rise of potential fields in real time strategy bots,” *In Proc. Artificial Intelligence and Interactive Digital Entertainment*, 2008.

[4]. J. Hagelbäck. “Using Potential Fields in a Real-time Strategy Game Scenario (Tutorial),” Internet:*http://aigamedev.com/open/tutorials/potential-field*, last visit [May. 6, 2010].

[5]. J. Hagelbäck. “Dealing with fog of war in a Real Time Strategy game environment,” in *Proc. of IEEE Symposium On Computational Intelligence and Games*, 2008. pp.55-62.

[6]. A. Poty, P. Melchior and A. Oustaloup. “Dynamic path planning for mobile robots using fractional potential field,” in *First International Symposium on Control, Communications and Signal Processing*, 2004, pp. 557- 561.

[7]. O. Khatib. “Real-time obstacle avoidance for manipulators and mobile robots,” *The International Journal of Robotics Research*, vol. 5, n 1, pp. 90–98, spring 1986.

[8]. J. Hagelbäck. “A Multi-Agent Potential Field based approach for Real-Time Strategy Game bots.” Licentiate Dissertation Series, Blekinge Institute of Technology, Sweden, 2009.

[9]. M.Buro and T. Furtak. “RTS Games as Test-Bed for Real-Time AI Research,”in *Proc. of the 7th Joint Conference of Information Science, Department of Computing Science, University of Alberta, Canada,* 2003.

[10]. R.K. Balla and A. Fern. “UCT for Tactical Assault Planning in Real-Time Strategy Games,” *in Proc. of the International Joint Conference on Artificial Intelligence*, 2009, pp. 40–45.

[11]. R.Arkin. “Motor Schema based mobile robot navigation,” *International Journal of Robotics Research*,vol.8, n 4, pp. 92-112, Aug. 1989.

[12]. H.V.Cazorla. “Multiple Potential Fields in Quake 2 Multiplayer.” *Master thesis in computer science, BTH*, Sweden, 2006.

[13]. M.A. Goodrich. “Potential Fields Tutorial.” Internet: *http://students.cs.byu.edu/~cs470ta/readings/Pfields.pdf*, last visit [Dec. 3, 2010].

[14]. C. Thurau, C. Bauckhage, and G. Sagerer.” Learning Human-like Movement Behavior for Computer Games,” *International Conference on the Simulation of Adaptive Behavior (SAB)*, 2004.

[15]. F. Janabi-Sharifi and D. Vinke. "Integration of the artificial potential field approach with simulated annealing for robot path planning," in *Proc.of the 1993 IEEE International Symposium on Intelligent Control*, 1993, pp. 536-541.

[16]. S.Viorel, I.Mircea, S. Elena and P.Cristina. "Using Artificial Potential Field Methods and Fuzzy Logic for Mobile Robot Control,*"12the International Conference on Power Electronics and Motion Control*, 2006, pp.385-389.

[17]. J.Togelius, M.Preuss, N.Beume, S. Wessing, J.Hagelback and G.N.Yannakakis. "Multiobjective exploration of the StarCraft map space,"*IEEE Symposium on Computational Intelligence and Games (CIG), 2010,* pp.265-272.

[18]. "StarCraft" Internet: http://en.wikipedia.org/wiki/StarCraft, last visit [Dec.8, 2010].

[19]. "StarCraft Compendium" Internet: http://classic.battle.net/scc/, last visit [Dec.12, 2010].

[20]. "StarCraft: Brood War" Internet: http://en.wikipedia.org/wiki/StarCraft:_Brood_War, last visit [Dec.8, 2010].

[21]. J.L.Hsieh and C.T.Sun. "Building a player strategy model by analyzing replays of real-time strategy games," in *Proc. of IEEE International Joint Conference on Neural Networks*, 2008, pp.3106-3111.

[22]. "Bwapi" Internet: http://code.google.com/p/bwapi/, last visit [Dec. 8, 2010].

[23]. "Bthai" Internet: http://code.google.com/p/bthai/, last visit [Dec. 8, 2010].

[24]. C. Hofmeister, R. Nord and D. Soni. "*Applied Software Architecture*," Addison-Wesley, 2003.

[25]. C.Wohlin; PRuneson; M.Höst; M.C.Ohlsson and A.Wesslen, "*Experimentation in Software Engineering*", U.S.A, 2000.

[26]. J.Hagelback and S.J. Johansson. "Measuring player experience on runtime dynamic difficulty scaling in an RTS game," in *Proc. of IEEE Symposium on Computational Intelligence and Games, 2009,* pp.46-52.

[27]. R.Metoyer, S.Stumpf, C.Neumann, J.Dodge, J.Cao and A.Schnabel . "Explaining how to play real-time strategy games," *Know.-Based Syst*., vol.23, pp. 295-301, 4 May 2010.

[28]. "StarCraft AI Competition" Internet: http://eis.ucsc.edu/Tournament1Results, last vist [Dec.18,2010].

[29]. N.Wirth and M.Gallagher. "An influence map model for playing Ms. Pac-Man ,"in Proc. of *IEEE Symposium on Computational Intelligence and Games, 2008.* pp.228-233.

[30]. P. Tozour. "Influence mapping," *in Game Programming Gems 2*, M. DeLoura, Ed. Charles River Media, 2001, pp. 287-297.

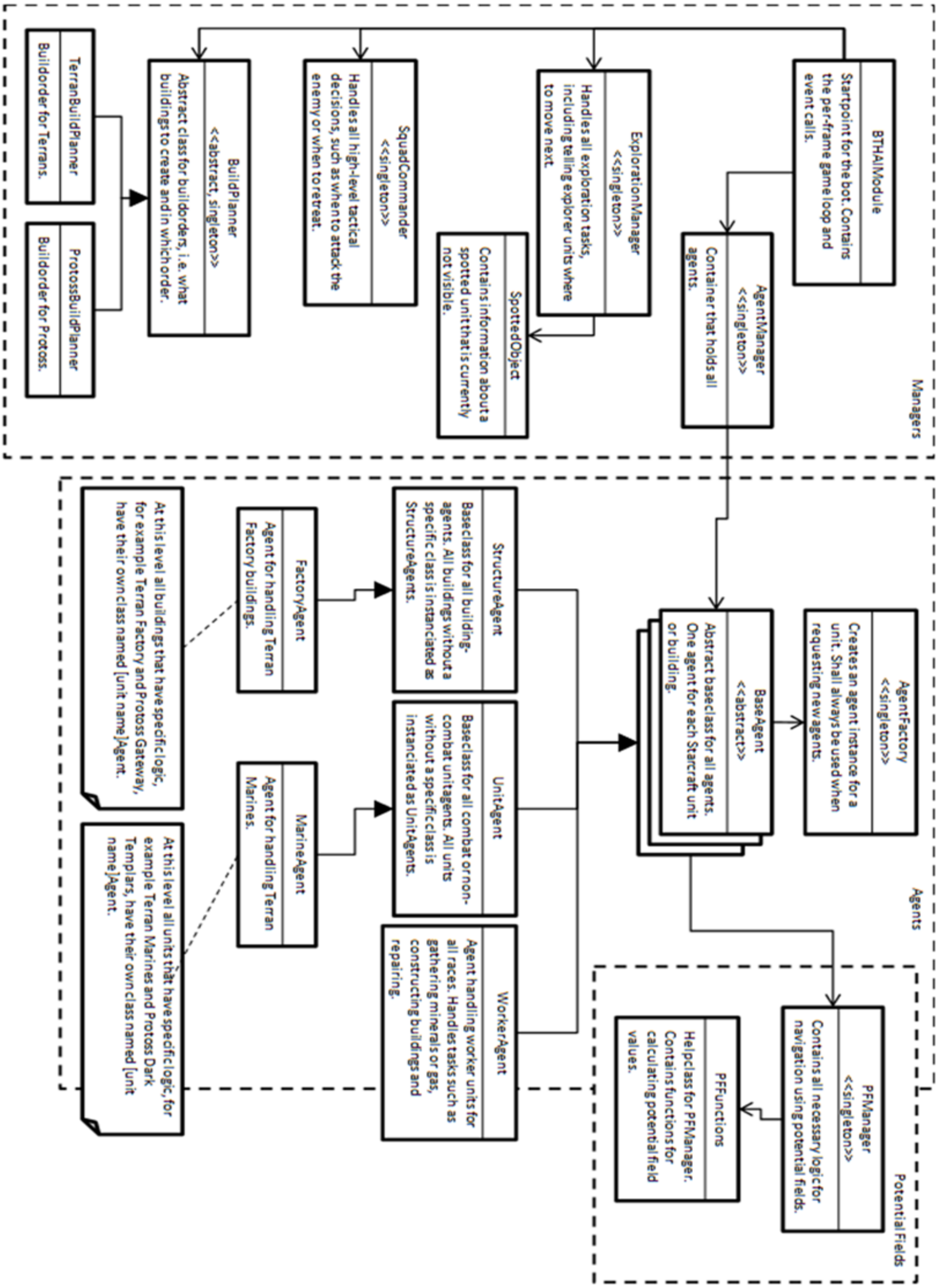
[31]. M.Mamei and F.Zambonelli. "Motion Coordination in the Quake 3 Arena Environment: A Field-Based Approach," *AAMAS'04,* July 19-23, 2004.

[32]. J.Togelius; M.Preuss; N.Beume; S.Wessing; J.Hagelbacka nd G.N.Yannakakis. "Multiobjective exploration of the StarCraft map space," *2010 IEEE Information Theory Workshop (ITW 2010)*, 2010, pp. 265-72.

[33]. R.T.Butt "Performance Comparison of AI Algorithms Anytime Algorithms" Master thesis in computer science, BTH, 2008.

[34]. "Student's t-test" Internet:http://www.biology.ed.ac.uk/research/groups/jdeacon/statistics/tress4a.html#Student's t-test, last visit [Jan. 02, 2011].

APPENDIX A


Managers
BTHAIModule
Startpoint for the bot. Contains the per-frame game loop and event calls.
AgentManager
<<singleton>>
Container that holds all agents.
ExplorationManager
<<singleton>>
Handles all exploration tasks, including telling explorer units where to move next.
SpottedObject
Contains information about a spotted unit that is currently not visible.
SquadCommander
<<singleton>>
Handles all high-level tactical decisions, such as when to attack the enemy or when to retreat.
BuildPlanner
<<abstract, singleton>>
Abstract class for buildorders, i.e. what buildings to create and in which order.
TerranBuildPlanner
Buildorder for Terrans.
ProtossBuildPlanner
Buildorder for Protoss.
Agents
AgentFactory
<<singleton>>
Creates an agent instance for a unit. Shall always be used when requesting new agents.
BaseAgent
<<abstract>>
Abstract baseclass for all agents. One agent for each Starcraft unit or building.
StructureAgent
Baseclass for all building-agents. All buildings without a specific class is instanciated as StructureAgents.
UnitAgent
Baseclass for all combat or non-combat unitagents. All units without a specific class is instanciated as UnitAgents.
WorkerAgent
Agent handling worker units for all races. Handles tasks such as gathering minerals or gas, constructing buildings and repairing.
FactoryAgent
Agent for handling Terran Factory buildings.
MarineAgent
Agent for handling Terran Marines.
At this level all buildings that have specific logic, for example Terran Factory and Protoss Gateway, have their own class named [unit name]Agent.
At this level all units that have specific logic, for example Terran Marines and Protoss Dark Templars, have their own class named [unit name]Agent.
Potential Fields
PFManager
<<singleton>>
Contains all necessary logic for navigation using potential fields.
PFFunctions
Helpclass for PFManager. Contains functions for calculating potential field values.

APPENDIX B

Table1: ASPF with Computer (Built in AI)

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	0	6	46.39	Won
2	0	3	1.03.72	Won
3	0	6	1.05.84	Won
4	0	3	48.29	Won
5	0	3	50.01	Won
6	2	1	1.03.13	Lost
7	1	5	58.44	Lost
8	0	6	54.27	Won
9	1	2	1.1.68	Lost
10	0	5	52.24	Won
11	1	5	53.38	Lost
12	1	4	51.08	Lost
13	0	6	49.18	Won
14	1	2	1.03.94	Lost
15	0	5	58.45	Won
16	0	6	1.011	Won
17	0	4	54.63	Won
18	1	3	56.25	Won
19	0	3	54.56	Won
20	0	4	53.12	Won

Table 2: SPF with Computer (Built in AI)

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	3	4	50.74	Lost
2	1	3	50.13	Won
3	2	3	47.04	Lost
4	2	2	57.08	Lost
5	0	2	49.95	Won
6	2	3	56.46	Lost
7	1	3	57.73	Won

8	2	5	48.29	Lost
9	0	3	48.73	Won
10	2	6	49.39	Lost
11	1	7	48.78	Lost
12	2	4	48.46	Lost
13	3	3	47.55	Lost
14	1	3	46.23	Won
15	2	1	1.00.48	Lost
16	0	3	58.41	Won
17	2	7	51.70	Lost
18	1	4	52.96	Won
19	3	8	47.05	Lost
20	2	7	48.61	Lost

Table3: ASPF with Chaos Neuron

Match no.	**Leader Killed**	**Winning team units left**	**Completion time (in sec)**	**Results**
1	1	6	54.94	Lost
2	0	3	1.03.00	Won
3	0	2	1.09.09	Won
4	1	2	59.85	Lost
5	2	6	58.77	Lost
6	1	5	59.17	Lost
7	0	1	1.05.00	Won
8	2	5	59.71	Lost
9	1	6	1.09.34	Lost
10	1	3	59.80	Won

Table 4: SPF with Chaos Neuron

Match no.	**Leader Killed**	**Winning team units left**	**Completion time (in sec)**	**Results**
1	2	2	1.03	Lost
2	3	8	49.05	Lost

3	3	6	57.96	Lost
4	3	9	51.79	Lost
5	2	6	52.69	Lost
6	3	7	54.22	Lost
7	2	3	1.00.75	Lost
8	3	7	52.11	Lost
9	2	5	56.88	Lost
10	4	6	55.93	Lost

Table5: ASPF with Windbot

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	1	5	56.38	Lost
2	2	9	48.55	Lost
3	1	6	48.84	Lost
4	2	4	49.95	Lost
5	2	8	49.89	Lost
6	2	2	46.50	Lost
7	1	7	50.64	Won
8	2	9	50.95	Lost
9	2	4	47.19	Lost
10	2	8	54.22	Lost

Table 6: SPF with Windbot

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	2	10	48.23	Lost
2	2	8	46.61	Lost
3	2	10	43.87	Lost
4	2	3	52.15	Lost
5	3	6	46.98	Lost
6	5	8	46.17	Lost
7	3	9	43.69	Lost
8	3	6	52.74	Lost
9	2	9	43.03	Lost
10	3	6	43.61	Lost

Table 7: ASPF with MSAILBOT

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	2	7	51.57	Lost
2	2	9	53.19	Lost
3	2	10	49.86	Lost
4	2	8	50.75	Lost
5	1	7	50.44	Lost
6	2	9	51.65	Lost
7	3	7	54.69	Lost
8	2	7	50.37	Lost
9	3	7	53.14	Lost
10	1	8	53.57	Lost

Table 8: SPF with MSAILBOT

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	5	9	47.74	Lost
2	3	10	49.01	Lost
3	3	10	47.01	Lost
4	3	11	46.10	Lost
5	3	9	50.53	Lost
6	3	10	48.01	Lost
7	3	9	48.90	Lost
8	3	9	47.02	Lost
9	4	8	49.04	Lost
10	2	11	46.52	Lost

APPENDIX C

Table 1: ASPF with Built in AI

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	1	2	56.74	Lost
2	0	3	54.49	Won
3	0	7	46.84	Won
4	1	3	58.05	Lost
5	0	1	1.00.79	Won
6	0	7	51.65	Won
7	1	3	59.05	Lost
8	0	4	53.77	Won
9	0	8	41.73	Won
10	0	2	56.53	Won
11	1	2	59.53	Lost
12	1	2	1.01.85	Lost
13	0	4	47.66	Won
14	0	8	42.35	Won
15	0	4	49.90	Won
16	0	7	40.88	Won
17	1	2	1.06.68	Lost
18	0	4	50.13	Won
19	0	4	46.57	Won
20	1	4	59.86	Lost

Table 2: SPF with Built in AI

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	3	2	54.94	Lost
2	1	4	46.08	Won
3	2	8	51.67	Lost
4	3	4	50.89	Lost
5	3	7	44.64	Lost
6	2	6	50.35	Lost
7	2	2	51.50	Lost
8	2	5	49.77	Lost

9	3	4	46.68	Won
10	2	5	46.30	Lost
11	3	6	51.29	Lost
12	3	5	50.59	Won
13	3	8	44.55	Lost
14	3	6	49.78	Lost
15	3	8	47.61	Lost
16	2	3	48.05	Won
17	3	3	47.88	Lost
18	2	8	44.83	Lost
19	2	6	52.06	Lost
20	3	5	47.30	Lost

Table 3: ASPF with Chaos Neuron

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	1	4	1.04.62	Lost
2	0	2	1.11.95	Won
3	1	6	1.08.07	Lost
4	1	5	1.04.17	Lost
5	0	4	1.02.46	Won
6	1	7	1.06.06	Lost
7	0	1	1.18.25	Won
8	1	4	1.08.54	Lost
9	0	1	1.22.17	Won
10	0	2	1.20.28	Won

Table 4: SPF with Chaos Neuron

Match no.	Leader Killed	Winning units left	Completion time (in sec)	Results
1	3	8	55,38	Lost
2	3	8	1.02.09	Lost
3	2	3	1.2.18	Won
4	4	6	54.68	Lost
5	2	7	58.65	Lost
6	3	8	53.37	Lost
7	2	5	54.07	Lost
8	3	7	59.16	Lost
9	2	2	1.02.95	Won

10	3	6	53.24	Lost

Table 5: ASPF with Windbot

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	0	8	39.87	Won
2	1	7	44.37	Won
3	0	8	41.30	Won
4	0	8	39.01	Won
5	0	8	40.54	Won
6	0	8	39.33	Won
7	0	8	40.05	Won
8	1	6	46.71	Won
9	1	6	41.98	Won
10	1	4	42.43	Won

Table 6: SPF with Windbot

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	4	6	54.58	Lost
2	1	4	50.77	Won
3	3	8	1.01.51	Lost
4	1	6	44.50	Won
5	1	2	50.58	Won
6	4	10	54.72	Lost
7	3	7	51.16	Lost
8	2	4	48.55	Won
9	1	6	45.31	Won
10	1	2	56.20	Lost

Table 7: ASPF with MSAILBOT

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	1	7	50.40	Won
2	0	7	44.20	Won

3	1	5	49.41	Won
4	1	5	49.63	Won
5	1	6	45.81	Won
6	1	6	44.77	Won
7	1	4	46.76	Won
8	1	7	40.93	Won
9	1	5	51.12	Won
10	1	5	45.56	Won

Table 8: SPF with MSAILBOT

Match no.	Leader Killed	Winning team units left	Completion time (in sec)	Results
1	1	5	45.63	Won
2	1	5	48.35	Won
3	2	2	1.07.05	Lost
4	2	4	50.94	Won
5	2	3	59.76	lost
6	2	3	1.06.70	Won
7	3	2	1.03.45	Lost
8	2	1	59.13	Lost
9	1	5	52.78	Won
10	2	6	50.08	Won

APPENDIX D

t-Test: No of leader killed in lost matches In Competition 1		
	Variable 1	*Variable 2*
Mean	2,153846154	1,166666667
Variance	0,307692308	0,166666667
Observations	13	6
Hypothesized Mean Difference	0	
df	13	
t Stat	4,352297326	
P(T<=t) one-tail	0,000391832	
t Critical one-tail	1,770933383	
P(T<=t) two-tail	0,000783664	
t Critical two-tail	2,160368652	

t-Test: Number of leader killed in lost matches in Competition 2		
	Variable 1	Variable 2
Mean	2,7	1,333333333
Variance	0,455555556	0,266666667
Observations	10	6
Hypothesized Mean Difference	0	
df	13	
t Stat	4,555555556	
P(T<=t) one-tail	0,000269824	
t Critical one-tail	1,770933383	
P(T<=t) two-tail	0,000539647	
t Critical two-tail	2,160368652	

t-Test: Number of leader killed in lost matches in Competition 3		
	Variable 1	*Variable 2*
Mean	2,7	1,777777778
Variance	0,9	0,194444444
Observations	10	9
Hypothesized Mean Difference	0	
df	13	
t Stat	2,760538937	
P(T<=t) one-tail	0,008104048	
t Critical one-tail	1,770933383	
P(T<=t) two-tail	0,016208096	
t Critical two-tail	2,160368652	

t-Test: Number of leader killed in lost matches in Competition 4		
	Variable 1	*Variable 2*
Mean	3,2	2
Variance	0,622222222	0,444444444
Observations	10	10
Hypothesized Mean Difference	0	
df	18	
t Stat	3,674234614	
P(T<=t) one-tail	0,000867715	
t Critical one-tail	1,734063592	
P(T<=t) two-tail	0,00173543	
t Critical two-tail	2,100922037	

t-Test: Average Number of lost matches in Competition 1		
	Variable 1	*Variable 2*
Mean	0,65	0,3
Variance	0,239473684	0,221052632
Observations	20	20
Hypothesized Mean Difference	0	
df	38	
t Stat	2,306512519	
P(T<=t) one-tail	0,013314456	
t Critical one-tail	1,685954461	
P(T<=t) two-tail	0,026628912	
t Critical two-tail	2,024394147	

t-Test: Average Number of lost matches in Competition 2		
	Variable 1	*Variable 2*
Mean	1	0,6
Variance	0	0,266666667
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	2,449489743	
P(T<=t) one-tail	0,018393749	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,036787498	
t Critical two-tail	2,262157158	

t-Test: Average Number of lost matches in Competition 3		
	Variable 1	*Variable 2*
Mean	1	0,9
Variance	0	0,1
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	1	
P(T<=t) one-tail	0,1717182	
t Critical one-tail	1,8331129	
P(T<=t) two-tail	0,3434364	
t Critical two-tail	2,2621572	

t-Test: Average Number of leader killed in win matches in Competition 1		
	Variable 1	*Variable 2*
Mean	0,571428571	0,071428571
Variance	0,285714286	0,071428571
Observations	7	14
Hypothesized Mean Difference	0	
df	8	
t Stat	2,333333333	
P(T<=t) one-tail	0,023955863	
t Critical one-tail	1,859548033	
P(T<=t) two-tail	0,047911726	
t Critical two-tail	2,306004133	

t-Test: Average Number of Win matches in Competition 1		
	Variable 1	*Variable 2*
Mean	0,7	0,35
Variance	0,221052632	0,239473684
Observations	20	20
Hypothesized Mean Difference	0	
df	38	
t Stat	2,306512519	
P(T<=t) one-tail	0,013314456	
t Critical one-tail	1,685954461	
P(T<=t) two-tail	0,026628912	
t Critical two-tail	2,024394147	

t-Test: Average Number of Win matches in Competition 2		
	Variable 1	*Variable 2*
Mean	0,4	0
Variance	0,266666667	0
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	2,449489743	
P(T<=t) one-tail	0,018393749	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,036787498	
t Critical two-tail	2,262157158	

t-Test: Average Number of Win matches in Competition 3		
	Variable 1	*Variable 2*
Mean	0,1	0
Variance	0,1	0
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	1	
P(T<=t) one-tail	0,171718198	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,343436396	
t Critical two-tail	2,262157158	

APPENDIX E

t-Test: Average Number of leader killed in lost matches in Competition 1		
	Variable 1	*Variable 2*
Mean	2,5625	1
Variance	0,2625	0
Observations	16	7
Hypothesized Mean Difference	0	
df	15	
t Stat	12,19875091	
P(T<=t) one-tail	1,72617E-09	
t Critical one-tail	1,753050325	
P(T<=t) two-tail	3,45234E-09	
t Critical two-tail	2,131449536	

t-Test: Average Number of leader killed in lost matches in Competition 2		
	Variable 1	*Variable 2*
Mean	2,875	1
Variance	0,410714286	0
Observations	8	5
Hypothesized Mean Difference	0	
df	7	
t Stat	8,275159266	
P(T<=t) one-tail	3,67008E-05	
t Critical one-tail	1,894578604	
P(T<=t) two-tail	7,34017E-05	
t Critical two-tail	2,364624251	

t-Test: Average Number of loss matches in Competition 1		
	Variable 1	*Variable 2*
Mean	0,8	0,35
Variance	0,168421053	0,239473684
Observations	20	20
Hypothesized Mean Difference	0	
df	37	
t Stat	3,151036696	
P(T<=t) one-tail	0,001608698	
t Critical one-tail	1,687093597	
P(T<=t) two-tail	0,003217396	
t Critical two-tail	2,026192447	

t-Test: Average Number of loss matches in Competition 2		
	Variable 1	*Variable 2*
Mean	0,8	0,5
Variance	0,177777778	0,277777778
Observations	10	10
Hypothesized Mean Difference	0	
df	17	
t Stat	1,405563857	
P(T<=t) one-tail	0,088933164	
t Critical one-tail	1,739606716	
P(T<=t) two-tail	0,177866328	
t Critical two-tail	2,109815559	

t-Test: Average Number of loss matches in Competition 3		
	Variable 1	*Variable 2*
Mean	0,5	0
Variance	0,277777778	0
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	3	
P(T<=t) one-tail	0,007478182	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,014956364	
t Critical two-tail	2,262157158	

t-Test: Average Number of loss matches in Competition 4		
	Variable 1	*Variable 2*
Mean	0,4	0
Variance	0,266666667	0
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	2,449489743	
P(T<=t) one-tail	0,018393749	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,036787498	
t Critical two-tail	2,262157158	

t-Test: Average Number of leader killed in Win matches in Competition 1		
	Variable 1	*Variable 2*
Mean	2,25	0
Variance	0,916666667	0
Observations	4	13
Hypothesized Mean Difference	0	
df	3	
t Stat	4,700096711	
P(T<=t) one-tail	0,009109927	
t Critical one-tail	2,353363435	
P(T<=t) two-tail	0,018219855	
t Critical two-tail	3,182446305	

t-Test: Average Number of leader killed in Win matches in Competition 3		
	Variable 1	*Variable 2*
Mean	1,2	0,4
Variance	0,2	0,266666667
Observations	5	10
Hypothesized Mean Difference	0	
df	9	
t Stat	3,098386677	
P(T<=t) one-tail	0,006377811	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,012755621	
t Critical two-tail	2,262157158	

t-Test: Average Number of leader killed in Win matches in Competition 4		
	Variable 1	*Variable 2*
Mean	1,5	0,9
Variance	0,3	0,1
Observations	6	10
Hypothesized Mean Difference	0	
df	7	
t Stat	2,449489743	
P(T<=t) one-tail	0,02207014	
t Critical one-tail	1,894578604	
P(T<=t) two-tail	0,04414028	
t Critical two-tail	2,364624251	

t-Test: Average Number of Win matches in Competition 1

	Variable 1	Variable 2
Mean	0,65	0,2
Variance	0,239473684	0,168421053
Observations	20	20
Hypothesized Mean Difference	0	
df	37	
t Stat	3,151036696	
P(T<=t) one-tail	0,001608698	
t Critical one-tail	1,687093597	
P(T<=t) two-tail	0,003217396	
t Critical two-tail	2,026192447	

t-Test: Average Number of Win matches in Competition 2

	Variable 1	Variable 2
Mean	0,5	0,2
Variance	0,277777778	0,177777778
Observations	10	10
Hypothesized Mean Difference	0	
df	17	
t Stat	1,405563857	
P(T<=t) one-tail	0,088933164	
t Critical one-tail	1,739606716	
P(T<=t) two-tail	0,177866328	
t Critical two-tail	2,109815559	

t-Test: Average Number of Win matches in Competition 3

	Variable 1	Variable 2
Mean	1	0,5
Variance	0	0,277777778
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	3	
P(T<=t) one-tail	0,007478182	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,014956364	
t Critical two-tail	2,262157158	

t-Test: Average Number of Win matches in Competition 4		
	Variable 1	*Variable 2*
Mean	1	0,6
Variance	0	0,266666667
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	2,449489743	
P(T<=t) one-tail	0,018393749	
t Critical one-tail	1,833112923	
P(T<=t) two-tail	0,036787498	
t Critical two-tail	2,262157158	