Sheffield Hallam University

A Path Planned Trolley in a Supermarket

Student Name: Pandian Sri Sai Surya

Student ID: 29042165

Module ID: 55-704710-KF-20190

Supervisor: Dr Lyuba Alboul

Date of Submission: 27/05/21

Abstract

In future, the automation system will influence the whole world to make everything more efficient. In particular, these innovative technologies can be introduced in the area of digital retail, so that customers could get their goods conveniently and efficiently. Every store uses shopping trolleys to assist customers transport items to buy. The billing process is quite exhausting and takes more time, which implies that the shops have to introduce new methods getting modernized to the future world. The use of a conventional shopping trolley by the customers in till are inconvenient and time-consuming. Today's world retailing plays a significant part in our economic activity, and the retailing businesses are also not ready to lose their customers.

There are two problem statements presented in this report. One is when customers are confronted with the problem of standing in queues and finally getting disgusted to leave the shop. Secondly, for old people who find it exceedingly difficult to wait in long queues. All customers enjoy shopping but hate to wait in a long queue that ends up in a frustrated mood. So, the customers taking a numeric-labelled trolley scan the required items using a hand-held electronic gun system that is equipped to the trolley and it also displays their respective prices as well as adding the whole amount. Eventually If customers complete their shopping, they push a button on the trolley making itself prepare a path-plan to enter to the queue. The elderly can sit down in the waiting place inside the supermarket and other shoppers can roam inside looking for future purchases. Parallelly the items are billed by the cashier as their trolley enters the till after being in the queues. After that, the cashier calls the customer's trolley number, and the transaction is completed. Finally, consumers will exit in the same cheerful mood as when they arrived, without having to stand in queues.

We implement an Artificial Potential Function (APF) method path planning to an autonomous trolley to achieve the goal position which adapts to the environment having static and dynamic obstacles. APF is a simple, authentic and smoothly in algorithm for preventing obstacles for the autonomous mobile robot, but it does have some challenges. For instance, before achieving their target, the local minimum point will drag mobile robots. Even if several enhanced APF algorithms have indeed been created, few papers don’t detail the way these algorithms have been implemented. Taking the considerations this report will concentrate on the manifestation of abstract enhanced principle for APF by demonstrating some formula and parameter improvements. To overcome the GNRON problem, a repulsive potential field function is introduced by a distance correction factor. To enhance the local minima issue, the regular hexagonal method is suggested. Additionally, for the dynamic environment, the relative velocity approach for detecting and preventing moving particles is suggested. This approach analyses not just the spatial position, but rather the magnitude and path of the velocity for the moving particles, which will in fact prevent dynamic obstacles.

The aim of this paper is to show how to use MATLAB 2018a to simulate an APF Path Planning with varying degrees of difficulty to achieve a collision-free optimal path. So, our goal is to design a trolley carrying the products, path plans itself to the starting point of the queue from where the customer left the trolley and finally queues up in the billing section representing the respective customer. The customer can rest themselves upon being called.

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 History of a Supermarket

Regarding a supermarket store, every week, 32 million of us go grocery shopping. They are a part of our environment as well as our weekly schedules. They have given reasonable food to all, expanded the food we consume and shortened the time we spend buying and cooking food considerably. Grocery shopping is one of the adult population's most popular practises. Often, we are on the store, love it or hate it. During the coronavirus epidemic, grocery stores are deemed important companies and workers are working hard to keep shelves in stock. Even during expansion of online supermarkets, shoppers continue to have the visual pleasure of choosing items while wandering between display racks.

Figure 1: Typical Supermarket

In the last 30 years, major supermarkets in many countries have transformed the retail industry environment by wider shop sizes, more shelving, more product and service diversity and comprehensive commercialization strategies. Supermarkets typically claim that they simply cater to consumer desires and that if they get it wrong, customers will simply move somewhere. However, it can be stated that they often play an important part in influencing consumer appetite and that, due to their market dominance, they have a clear control on what customers want, as well as how and when they purchase it. Furthermore, rather than doing all their purchases in the same shop, shoppers waste time shopping for the grocery store that provides more benefits in a specific product. Indeed, in the intensely competitive retail market with extremely challenging customers, the pursuit of a good consumer partnership is an important way of doing business.

1.2 Problem Statement

Every supermarket seeks to improve its operating efficiency and give its consumers the best experience possible. Customers enjoy shopping in supermarkets and food stores where time does not have to be wasted in queues. It can both be a blessing and a curse if you have a long line of retail customers in your shop.

On the one side, more consumers are ready to shop and boost business revenues. Long queues, on the other hand, aim to lower perceived customer satisfaction dramatically. The factors making a frustrated customer are firstly, time without activity seems longer than time occupied - If you find your customers hanging in queues twiddling their thumbs in frustration, they will believe they have been waiting "forever."

The more a customer must wait, the less happiness he/she has. Secondly, unexplained delays tend to be longer than explained waits - Nothing irritates the clients more than joining a seemingly short queue only to be kept up for a lifetime.

When it appears that the line has been stopped and the cashier is looking out into space with no reason for the delay, you will find that the customers start tapping their toes and become more irritated. They can also believe that the delay is not worth it and leave without purchasing something.

Figure 2: Queue in a Supermarket

And the most advanced check-out supervisor cannot be all at once. The numbers of people entering or passing through the store are not measured, the checkouts cannot be seen simultaneously, and customers who go through the checkouts are not able to be precisely measured over time. On average, retailers agree that the most appropriate time they are willing to stand in a queue is 5 - 10 minutes. It could also suggest that any prospective buyers will opt to quit and leave the shop prior to selling. Furthermore, according to a new survey, when faced with long queues that seem to be stagnant, more than 20% of shoppers would cancel their orders or give up completely on having required services and walk out of the shop. With such fierce rivalry in the supermarket market, are stores prepared to lose any more customers? Customer loyalty is the secret to every company's growth. Some stores can deliver the best consumer finance items to their customers, but nothing else counts if the overall customer service is bad.

Improving the overall consumer experience is a sure way to see happy faces by removing physical lines for utilities and providing their clients with a personalised and creative experience.

1.3 Solution for the Need

I assume a Path Planned Shopping Trolley as a device can be an invention that can be tailored to the demands of its own ecosystem in order to meet the needs of both shoppers and retailers in almost ideal equilibrium and harmony. This path planned trolley is programmed and functioned to make its way to the billing section/ counter of the supermarket once the customer finishes their shopping.

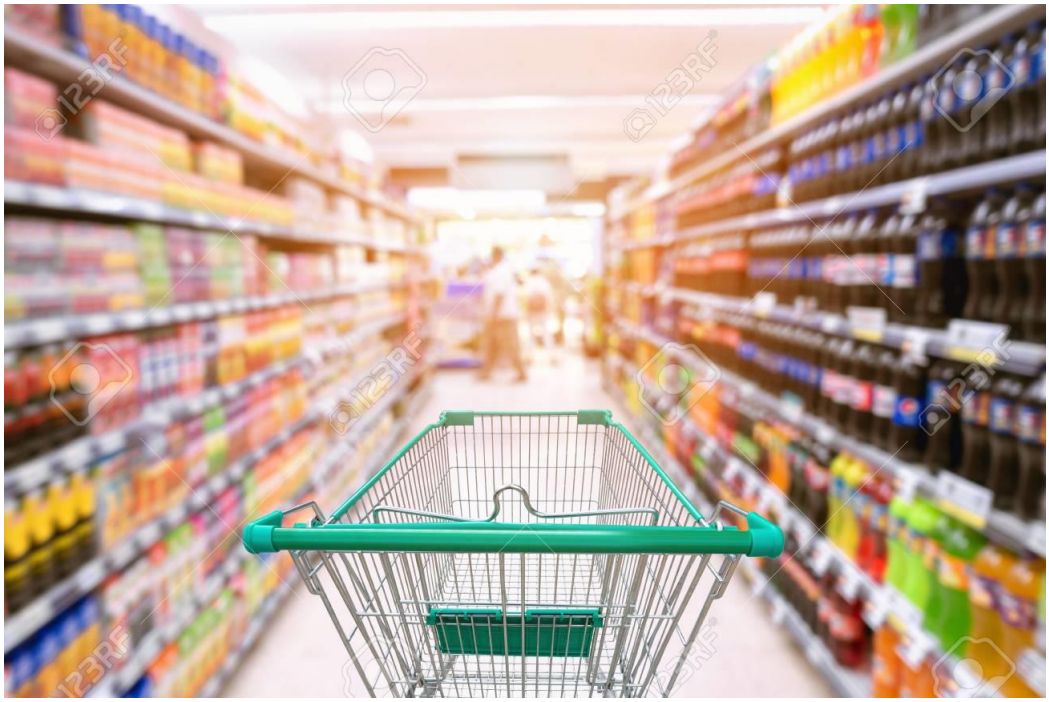

Figure 3: Shopping Trolley used in Supermarket

This is based on a problem statement of customers leaving the stores standing in huge queues as the consumers must select the items they want to buy and put those into the shopping cart and then they must enter the billing area and at last queue up for payment.

The use of robotics and other types of automation in retail stores has grown in popularity in recent years, as they not only increase the shopping experience of customers, but also attract people into the store, enabling stores to compete with online shops. The aim of all shopping robots is to make shopping in shops more convenient. As a result, retail businesses are obsessed with making the shopping experience more enjoyable and entertaining while also reducing prices and improving quality through the use of technology.

1.4 Motivation and Method Used

The Path Planning method is to search for a collision-free optimum path in an obstacle-full work spot from the point of departure to the destination, depending on one or more optimisation criteria, such as the quickest route, the shortest time of motion, and the lowest cost of work. In robotics path planning, robots who would travel easily in the world face the same challenges that humans do while walking. If people do not communicate with their surroundings while travelling, they will become disoriented in their direction and position due to flaws in their movement system. As humans determine which direction to take to avoid obstacles along the way by using their sense of the world, robots must detect the environment through sensors to make the right decision for their path to the target.

Path planning provides for an environmental map to be developed. Environmental mapping refers to an exact definition of spatial positions of the different structures in the robot environment, like barrier, road signs and more: i.e. the development of a space model or map. The objective is to assist the mobile robot design an optimised path from start of the robot to the goal point of the known prototype with obstacles. Many algorithms have been developed on robot route planning. There are two major approaches to main navigation algorithms: the classical algorithms and the heuristic algorithms.

Traditional methods like roadmap construction, artificial potential field process (APF), cell decomposition, and reactive approaches. The Voronoi diagram, visibility matrix, subgoal network, and silhouette method are examples of roadmap building formulas, which are a community of calculative geometrically based approaches to route planning.

These algorithms convert the workspace's challenges into polygons. To find the shortest path, a relation is formed between the corner points of polygonal barriers, the start point, and the target point. Dijkstra algorithm, BFS algorithm, A* algorithm, Bio-inspired algorithms, and so on are examples of heuristics algorithms. These algorithms can produce decent results, but not always the best. Each of the algorithms listed has cons and drawbacks. They are not mutually independent but are highly connected with one another. Everyone can be mixed in many implementations to obtain the most efficient route planning process.

CHAPTER TWO

LITERATURE REVIEW

Khatib et al. suggested the artificial potential field for manipulators and mobile robots in obstacle avoidance. The operator shifts in a force field in the potential field role. Attainable location for the agent is an enticing pole and repulsive surfaces to the agent are the obstacles. By integrating the enticing force with the repulsive force in the current context, it generates a possible field that enables the agent to reach the target location while avoiding obstacles. The possible area is commonly used in robot route planning and control issues, thanks to its sophisticated mathematical analysis and simplicity.

Chunshu et al. suggested an algorithm for the avoidance of obstacles in an undefined static environment using APF. This approach was developed in conjunction with grid APF and solved in a static setting to minimise the local problem and oscillations. The suggested algorithm calculates the possible function value for each robot cell and selects the cell with minimum value. The cell value is raised to escape local minima when the robot is in the minimum value.

The Adaptive APF solution for the avoidance of unmanaged aircraft has been implemented by Razee H. et al. The potential area suggested relies on the aircraft's attitude and its relative location on the obstacle. They used a revolving potential field across the obstacle and outcomes of the simulation proved that the proposed solution was feasible.

Vadakkepat P., et al. proposed an adaptive APF with a genetic algorithm to map the optimum route and an escape force method to escape local minima in order to facilitate mobile robot routing.

Chen L. implemented an enhanced UUV Route Lanning APF based virtual obstacle (Unmanned Underwater Vehicle). They first found that UUVs are stuck in local minima ranges and then added a virtual obstruction point. This simulated obstruction helps to adjust the amplitude, direction and thereby avoid the local minimums.

Cai and Ferrari suggest the cell decomposition algorithm. Its basic theory is to decompose the robot's surroundings into a collection of nonoverlapping cells. A connectivity graph is built using the cells. Reactive methods, such as neural networks, fuzzy logic, and neuro fuzzy systems, have also served as the foundation for path planning growth. It represents the robot's workspace as a network of neurons. As inputs to the neural network, use the data obtained from the sensors, as the obstacle orientation and state of motion. Consider the outputs of iterative equations to be the robot's travel trajectory and steering angle. The robot is capable of avoiding obstacles and moving toward the target stage.

Doria et al. stated a mechanism for avoiding local minima in APF influenced by the Deterministic Annealing (DA) technique, and they inserted a temperature parameter (T) into the cost function of T-APF. The addition of the T to the cost feature increases the repulsion area of the obstacle while decreasing the attraction area of the target. Unlike the virtual annealing approach, the DA approach avoids random displacement across the cost function surface. As the robot becomes trapped at a local minima point, the T value begins to rise before the robot leaves the local minima point.

Wenjie L. et al. suggested a possible means of knowledge for coordinated path planning and management. Their latest strategy uses information roadmap to avoid local minimums while increasing the chances that the sensor, subject to robot kinematics, can be measured.

Lee et al. suggested a random force-based algorithm for local minima escape in the APF method. They tackled the symmetrically balanced robot-obstacle-goal condition that causes the deadlock. When the robot becomes stuck at a nearby minima, the randomized force technique is used to free it.

Ji-Wung proposed a potential field also known as bug potential filed, that is a fusion of the potential field and the bug algorithm to solve the issue of local minima and path ineffectiveness in path planning in known environments. In order for the robot to avoid all path borders of obstacles, a vision graph and the colliding cone are used to measure the direction of the virtual powers. The curvature weighted the shortest path algorithm from Dijkstra to find the best path on its visibility graph.

Gingras D. et al. proposed a path planning method related to fluid dynamics. The finite element approach is used in this algorithm to calculate a velocity potential function that is free of local minima. As a road map, several flow patterns were estimated, and the best direction was chosen.

Ge and Cui propose a new potential solution in the field for a complex area of shifting objectives and obstacles. The latest potential feature not only takes into account the position of the robot but also its speed (relative to target and obstacles). The negative gradient of the direction and speed potential is defined as the virtual power. Then the motion of the mobile robot is determined by the virtual power.

Sun et al define a complex system adaptive potential field characteristic dependent on a higher minimum hazard index. It can build an optimal route not only based on relative data of distance and speed direction, but also on the robot's size and the obstacle speeds.

Montiel et al. propose a dynamic challenge with a concurrent evolutionary artificial potential area. If a reachable configuration range exists, it can attain stability in complex real-world scenes with dynamic obstacles.

To address the limitations of T-APF-based algorithms, Q. Song et al. introduced an updated potential field incorporating fuzzy control. Repulsive force has been updated with two regulatory variables, one of which is caused by distance and another by the robot's velocity. The fuzzy monitoring approach was used to change the regulatory factors. The computing time is one problem with this approach.

C. Ya-Chun et al. proposed an online deadlock avoidance approach for wheeled mobile robots with the existence of boxlike obstacles calculated to use the Hough transformation. The local minima issue has been resolved by reinventing the repulsive potential area. They combined several obstacles into one large obstacle, resulting in a deadlock. This allows the robot to break free again from deadlock.

A Melingui et al. introduced a new real-time navigation method that combines APF with an interval type-2 fuzzy logic scheme. The orientation angle relative to the target was included in their study to calculate the likelihood of overcoming the local minima. When the robot encounters a trapping condition, fuzzy logic is being used to help it avoid.

The advanced tangent bug (ITB) technique was proposed for EF Mohamed et al. to be combined with a future submission in order to escape local minima problems in real-time T-APF path planning. They also used switching and combining conditions in their system to ensure deadlock-free movement. Proceed to the objective using APF and boundary lead mode utilising the ITB algorithm to avoid permanent obstructions. Drive your technique in two modes. The additional switching function was utilised to address local minima problems and to discover the fastest way to travel.

CHAPTER THREE

METHODOLOGY

3. 1 Experimental Design

3.1.1 Autonomous Mobile Robotic Trolley:

Modern trolleys would be more autonomous than the trolley devices currently in use. The trolley requires many autonomous skills and the ability to act smartly. To be completely autonomous, a trolley must have the capacity to obtain environmental knowledge, operate beyond user intervention for a prolonged period of time, navigate through its working area and escape obstacles and conditions dangerous to humans.

The sensing mechanism primarily offers trolley services for the use of these environmental input to see, contact, move and implement algorithms. By using various types of sensory instruments the trolley needs to collect knowledge about its surroundings. A selection of sensors for such knowledge should be autonomous to carry out their roles and raw sensory dataset should be processed and converted into an environmentally friendly analytical form.

Motion preparation is used to perform a certain mission, such as trolley motion, to travel from one location to another (goal position). Motioning of mobile carriages enables the robot to determine how to attain a certain target location (achieve a given task). The consumer has to supply the trolley with an operation, and the trolley then decides the action it takes. Motion planning challenges can be split into two so-called route and direction planning. The trolley movement must be planned to generate an executable route. The difference between trajectory planning and path planning is significant. Path planned point to a set of no collision places and the geometrically arrangement of the carriage (orientation and position) in which trolley dynamics are ignored and planning of trajectory creates a time profile for the path with linear and angular speeds. For each scheduled task, a low level control is essential. This provides an actuator control sequence that tracks its movement along the created direction or trajectory, to execute the necessary movement of the trolley.

This follows a Robot Navigation system. To complete its functions, the autonomous robot needs autonomy and mobility. In such circumstances, it may traverse certain areas, which could be known or unfamiliar. The movement of the robot through its surroundings is referred to as robot navigation. Navigation of mobile robots is a diverse field of study that encompasses a wide range of technology and applications. Robot navigation refers to a mobile robot's ability to assess its very own location in its reference frame and find a path to the target position. In order to successfully navigate a robot from one position (beginning point) to another (goal point), the robot must overcome three prominent claims.

1. Role for localization:

   The role for localization is the method of determining from the position where, a robot is in relation to its surroundings. In way to produce useful choices, the robot must understand its own role. This issue is solved by the robot localization.
2. Role for mapping:

   When navigating, the robot must feel the environment and recognise key features that will enable the robot to collect information about its surroundings. The robot must know where it goes in order to complete the operation. It must provide a target and a course of action.

3. Role for Planning the Path:

   As robots comprehend where they are and where they have to go in their work environment, they have to decide how to do so while respecting the constraints given, such as overcoming barriers.

3.1.2 Hand-held Scanners:

Handheld scanners are small remote devices which customers can use in their basket to scan items or even skip checkout lines for the cost of their products. Despite the prevalence of portable scanners, their effect on market purchases is not clear. Although industry analysts expect the development of handheld scanners, there are contrasting findings from studies. There is also proof of this, for example, as handheld scanners guarantee that checkout times will be reduced, that shoppers will keep within their budgets and have fun shopping. Multi-sales measurements such as net sales, number of products and segments sold, and overall time spent can be increased by handheld scanners. When customers use handheld scanners, they are vigilant about the stores and touch more objects.

On the top of the trolley, we install this retailing hand held computer machine for identification of the products which the customers scans and with a consistent WiFi connection having all the database of the store. This machine is also called as pricing gun of the retailing shop and also it has an LCD display which shows the price of the product which is being scanned. Hence the customers can check the prices which they want and put it inside their trolleys. This machine is fixed on the trolley top for security purposes.

The gun to be used on the top of the trolley is shown below in figure 4,

Figure 4: Hand-held Computer gun

The specifications are:

Power: Main bat. rechargeable li-ion battery 3200mAh Pistol bat. rechargeable li-ion battery 5200mAh (with pistol grip)

CPU used: Cortex-A7 1.3GHz quad-Core

RAM used: 1GB

ROM used: 4GB Flash

The display used is 4'' TFT-LCD touch screen with backlight, 16.7M colours ,480 W× 800L (WVGA)

The barcode scanner: Symbol SE4500 and Symbol SE955.

The WLAN used for communication is the main part as all the database is saved inside the intranet of the retailing shop. Hence, the specification for the WLAN is WI-FI IEEE802.11b/g/n?internal antenna.

3.1.3 Path-planning simulating/ initializing button

Once the trolley is filled with products selected by the customers, the trolley should be ready for path-planning. To be more understandable, the trolley has a button on its top and when it is pressed it comes to knowledge that the customer has finished shopping. So, the trolley's next step is to start the APF methodised path planning. Hence this button is the path planning initializing button.

3.2 ARTIFICIAL POTENTIAL FIELD

3.2.1 Approach

In mobile robotics, it is a tough challenge to achieve an objective role without colliding with obstacles in the operating world. One basic approach used in problems such as these is artificial potential which offers a convenient way to avoid barriers. The definition of potential areas is that certain fields of virtual potential operate on the robot to monitor its movement. A future area reveals a reactive architecture as it is sensitive to environmental characteristics. The area of artificial potential is among the usual strategies for autonomous mobile robot path planning and obstacle avoidance. It can be considered an environment with many peaks created by obstacles and goal that mark the final stage. The robot is called a particle in the field of robot path design that travels from a strong potential to a low potential. Now visualizing the APF, we understand that it is a set of vectors arithmetically depicted in 2D or 3D arrows, in which the arrow length is magnitude and the arrow angle is direction. This technology is mostly shown in 2D settings when designing mobile robot routes and preventing obstacles.

Consider the robot in position (x_r, y_r) in order to explain the basic phenomenon of the artificial potential fields. Remember that in this setting the challenge (x_o, y_o) and the objective (x_g, y_g) are established and the robot moves towards the target. Figure 5 demonstrates how the barrier forces the robot.

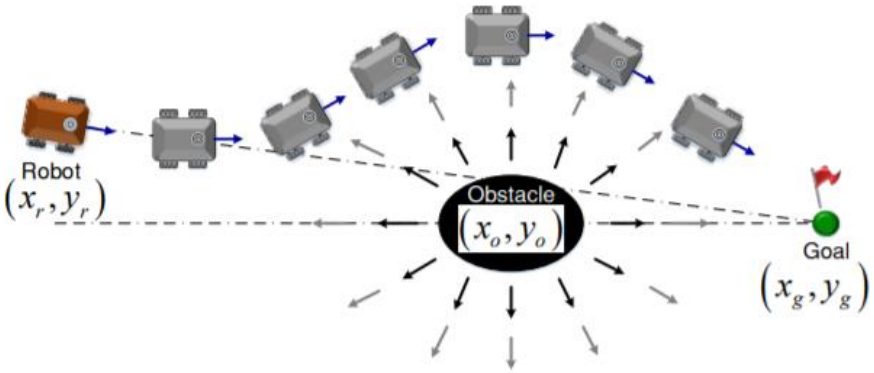

Figure 5: Approach for APF

When the robot approaches the target, it will meet the effective range of the barrier, which means that the robot would be repelled and step away from the obstacle. The robot goes straight into the target spot if beyond the effective range of the obstruction.

The advantages of APF can be said many such as the robot's behaviour, however, can be easily predicted by the designer, is implemented and visualised. APF promotes parallelism, each area is independent and can be independently applied. During the design stage or in real time, the APF method can be quickly parameterised and optimised. The mixture mechanism is versatile and can be modified to represent different sub-behaviour priority.

3.2.2 Theoretical algorithms and calculations

APF consists of two main fields: the goal-creating enticing field and the obstacle-creating repulsive field. The attractive and repulsive forces guide it to the goal when the robot is immersed in the potential area. The aim of this combination of multiple strengths is to steer the robot movement safer while keeping it away from obstacles. This statistic summarises the strength distribution of the attractive potential as well as the repulsive potential of an environmental barrier. Figure 6 shows the environment with which the algorithms are calculated.

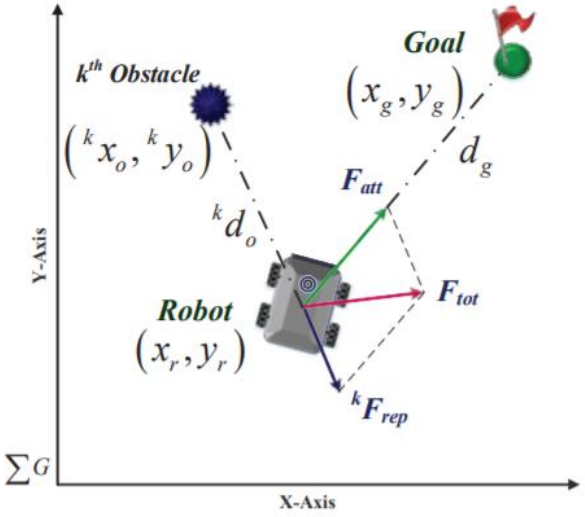

Figure 6: Theoretical Algorithm for APF

3.2.2.1. Calculation of Attractive field potential function

Before approaching the target stage, the trolley was hit by the attractive force right from the beginning. The greater the distance between the target and the attraction. When the trolley arrives at the destination, the attraction is reduced to nothing.

The robot is frequently known as a mass point to alleviate the path planning issue. The robot's and goal's positions can be represented as vectors pr = [xr, yr]T and pg = [xg, yg] T, accordingly. The goal's attractive force can be interpreted using the Gaussian function, as seen in Equation.

$F_{att} = a_g[1-\exp(-b_g * d^2_g)] * e_g$

where,

F_{att} is the force of attractive

a_g is the maximum value of F_{att}

b_g is a width constant of the distribution

d_g is a a parameter which shows the Euclidean distance between the goal and the robot.

d_g = square root of $[(x_r - x_g)^2 + (y_r - y_g)^2]$

e_g is the unit vector that is towards the goal point. It can be written as,

$e_g = 1/d_g [(\Delta x_g i + \Delta y_g j)]$

where,

$\Delta x_g = x_g - x_r$

$\Delta y_g = y_g - y_r$

The attractive force is directed down the line seen between trolley and the target point and heads to the goal point.

3.2.2.2. Calculation of Repulsive field potential function:

The repulsive potential field function, like the attractive potential field function, is actually distance-based. The repulsion could impact the trolley if it were beyond the influence zone of the obstruction. The greater the repulsion, the nearer the obstacle.

The repulsive potential force for the k^{th} obstacle is represented by,

$^kF_{rep} = a_o [\exp ((-b_0) * (^kd_0^2)] * ^ke_g$ if $^kd_0 <= d_d$

where,

$^kF_{rep}$ is the repulsive force from the k^{th} obstacle

a_o is the maximum value of $^kF_{rep}$

b_0 is a width constant of the distribution

$^kd_0$ is the parameter for the Euclidean distance between the goal and the robot.

d_d is the impact distance of the robot

e_o is the unit vector that is towards the robot from the k^{th} obstacle and is represented as,

$^{k}e_o = 1 / {}^{k}d_0 \, (\Delta \, {}^{k}x_o \, i + \Delta \, {}^{k}y_0 \, j)$

where,

$\Delta \, {}^{k}x_o = x_r - {}^{k}x_o$

$\Delta \, {}^{k}y_o = y_r - {}^{k}y_o$

3.2.2.3. Calculation of Resultant field potential function :

When heading toward the target, the trolley would be influenced by the combined behaviour of the the repulsive potential field and the attractive potential field. This is a basic scenario in which the trolley escapes the challenge island on the way to the target. The trolley is drawn by attractive force directly to the target from the start point, being within the productive region of repulsive field, the trolley is exposed to both attractive and repulsive force, respectively.

$F_{tot} = F_{att} + \Sigma \, {}^{k} F_{rep}$

There are one or more targets and obstacles in the environment in robot navigation difficulties. The forces created by the two fields must thus be combined by the barriers and objectives. The total potential range is achieved by summing the repulsive potentials of all barriers and the attractive repulsive force of the target.

3.2.3 Methodology Involved

The APF methodology in this application was extensively altered to make it more effective in static as well as dynamic trolley path planning. In addition, we propose an obstacle avoidance strategy based on the modified APF system. The GNRON problem is resolved by using a distance correction factor for the original repulsive potential field function. The regular hexagon driven system (RHG) is coupled with the APF technique to let the vehicle leave the local minimum area. The relative velocity technique uses the speed element to the APF approach to tackle the obstacle avoidance of the dynamic surrounding challenge.

3.2.3.1. GNRON Problem statement and solution

The GNRON problem happens if the location of the target is within the obstacle. In terms of attractive force, this process creates a greater rebellious force. Consequently, it is suggested that the robot achieve effectively to a stronger attractive function.

The GNRON dilemma is caused by the fact that the target point is not the minimum of the overall potential function. If the trolley hits the obstacle, $d(X,X_o)$ approaches 0 and $1/ d(X,X_o)$ approaches infinity. That is, as the trolley approaches the target, the attractive potential potential field reduces while the repulsive potential field grows rapidly.

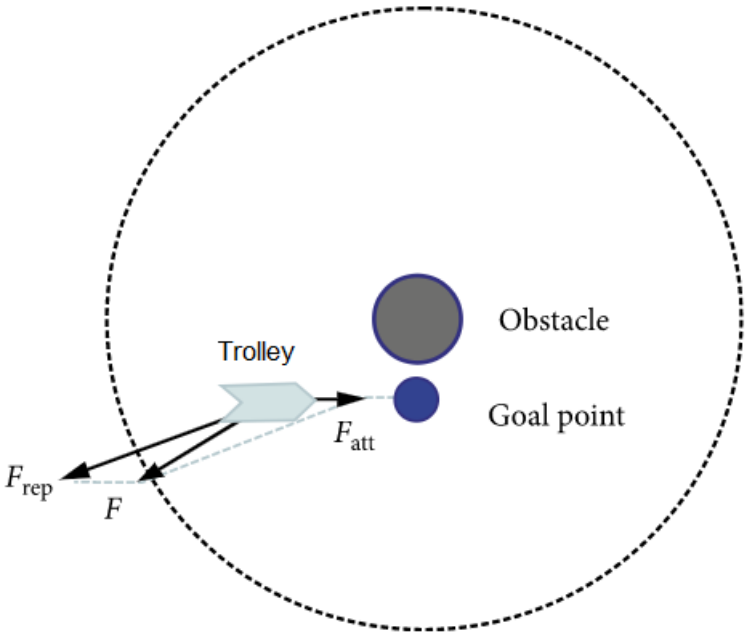

Figure 7: GNRON issue

The GNRON issue is shown in two dimensions in figure 7. The repulsive potential field's impact range is shown by the dotted circle. When the trolley is positioned as seen in the figure, it is caused by the combined action of repulsive and attractive forces. The repulsion is significantly greater than the attraction, analogous to the flow equation from a higher potential to a less potential field. The cumulative force is directed away from the target stage. The trolley is unable to cross the target stage. Hence the process of GNRON has influenced the public opinion for local minima solution as well as higher potential field materials.

To solve the GNRON issue the repulsive potential field function is corrected by a distance factor. It is capable of balancing changes in two types of force, especially the repulsive force rise. If the trolley approaches the target stage, the repulsive force will be increasingly diminished. It should therefore ensure that the possible area at the target point is the local minimum. The repulsive potential field function can be expressed as follows while the attractive potential field remains unchanged.

$$\mathrm{U_{rep}}\ (X) = \begin{cases} \left(\frac{1}{2}\right) k_{rep} \left[\frac{1}{d\ (X, X_o)} - \frac{1}{d_o}\right] \mathrm{d^n}\ (\mathrm{X}, \mathrm{X_g}), & \mathrm{d}\ (\mathrm{X}, \mathrm{X_o}) \leq \mathrm{d_o} \\ 0, & \mathrm{d}\ (\mathrm{X}, \mathrm{X_o}) > \mathrm{d_o} \end{cases}$$

dn (X,Xo) is the distance from the trolley to the target place. n is always more than 0 for any positive real number. The augmented repulsive force characteristic is:

$$\mathrm{F_{rep}}\ (X) = -\Delta\ U_{rep}\ (X) = \begin{cases} F_{rep1}\ (X) + F_{rep2}\ (X) & \mathrm{d}\ (\mathrm{X}, \mathrm{X_o}) \leq \mathrm{d_o} \\ 0, & \mathrm{d}\ (\mathrm{X}, \mathrm{X_o}) > \mathrm{d_o} \end{cases}$$

F_{rep1} and F_{rep2} are as follows:

$$\mathrm{F_{rep1}}\ (X) = k_{rep} \left(\frac{1}{d\ (\mathrm{X}, \mathrm{X_o})} - \frac{1}{d_o}\right) \frac{d^n\ (\mathrm{X}, \mathrm{X_g})}{d^2 (\mathrm{X}, \mathrm{X_o})}$$

$$\mathrm{F_{rep2}}\,(X) = \frac{n}{2} k_{rep} \left(\frac{1}{d\,(\mathrm{X}, \mathrm{X_o})} - \frac{1}{d_o} \right)^2 \left(d^{n-1}\,(X, X_g) \right)$$

The following figure 8 depicts the resulting force model of the enhanced repulsive field potential.

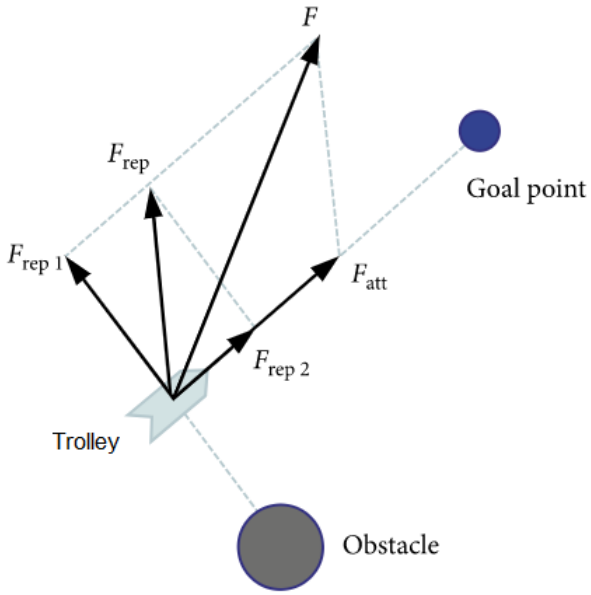

Figure 8: Resulting force model of the enhanced repulsive field potential.

The enhanced repulsive force F_{rep} is not directed in the direction of the trolley and the obstacle. This is not the same as the conventional APF system. F_{rep1} is one of the Frep components. Its path is parallel to the line between the trolley and the obstacle, and it causes the trolley to move away from the obstacles. F_{rep2} is actually a form of F_{rep} component. Its course is on line connecting the goal point and the trolley point, and it leads the trolley towards the target point.

The trolley's force function will perform additional analysis depending on the n value:

1. When n=1, $F_{rep1}(X)$ and $F_{rep2}(X)$ are expressed as follows:

$$\mathrm{F_{rep1}}\,(X) = k_{rep} \left(\frac{1}{d\,(\mathrm{X}, \mathrm{X_o})} - \frac{1}{d_o} \right) \frac{d\,(\mathrm{X}, \mathrm{X_g})}{d^2\,(\mathrm{X}, \mathrm{X_o})}$$

$$\mathrm{F_{rep2}}\,(X) = \frac{n}{2} k_{rep} \left(\frac{1}{d\,(\mathrm{X}, \mathrm{X_o})} - \frac{1}{d_o} \right)^2$$

As the trolley gets closer to the target, d decreases and reaches zero. F_{rep1} is now approaching 0, and F_{rep2} is a constant. Trolley drives toward the target point solely by the motion of attraction.

2. When n is greater than one and the trolley travels near to the target point, d^n (X, Xg) and d^{n-1} (X, Xg) reach zero. F_{rep1} and F_{rep2} reach zero, as does the overall repulsion F_{rep}. Under the impact of the attractive force, the trolley will eventually approach the target point.

3.2.3.2. Local Minimum Problem Statement and solution :

Whenever the robot, the target and the centre of the obstacle are matched, the local minimum issue arises. This leads to a cumulative potential force equal to 0 before the target location is reached. To resolve this question, the idea of triggering a virtual escape force whenever a local minimum is identified is a new repulsive potential function. This force acts as a rotary force that permits that the robot escapes from the impasse positions and efficiently moves away from the obstacles to the goal.

Since the resulting force is zero or reduced at the target stage, the trolley should come to an end. The trolley that nevertheless reaches other places where pleasant and repellent forces in same line but dislike directions are almost the same. The resulting trolley strength is so likely to be minor or nonexistent. The trolley is then trapped in a minimal local area. These figures depict three common examples of local minima issues.

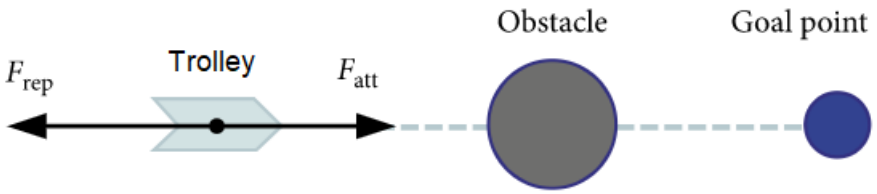

Figure 9: Local Minima Issue

Figure 9 indicates a minimal local issue in a standard APF colinear system is composed of three points (obstacle is between the target point and the trolley).

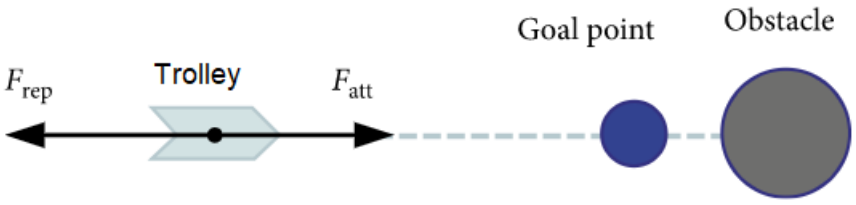

Figure 10: Minimal local issue in a standard APF colinear system

Figure 10 indicates a minimum of 3 colinear points local issue (target point is between the obstacle and the trolley).

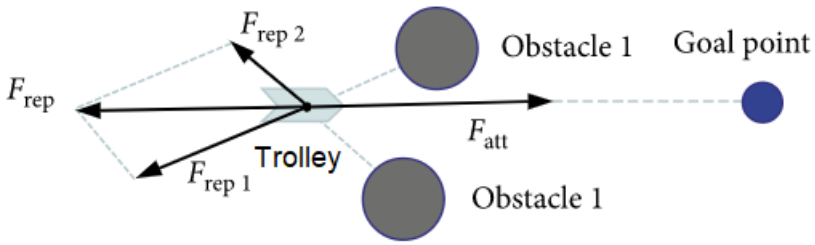

Figure 11: A Minimum of 3 colinear points local issue

Figure 11 shows the minimum local environment and many obstacles.

From the first two figures, we can say that the trolley, the goal point and the obstacle are collinear.

At first, the force of attractive outweighs the force of repulsive. Along on the line, the trolley travels toward the target. If the gap between the trolley and the obstacle reduces, so does the repulsive force. There must be a point at which the repulsive force is negligible, and the trolley comes to a halt. The third figure depicts the local minima challenge with many obstacles. The force of attractive is balanced by the force of repulsive, causing the trolley to come to a complete halt.

We now implement a regular hexagon guided (RHG) approach for solving the local minima issue.

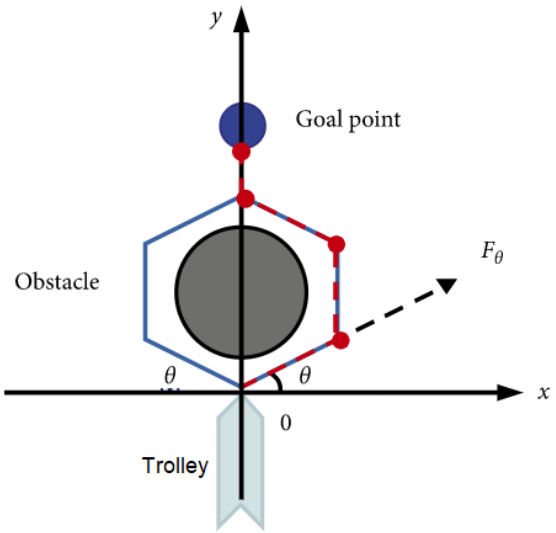

Figure 12: RHG solution with θ negative

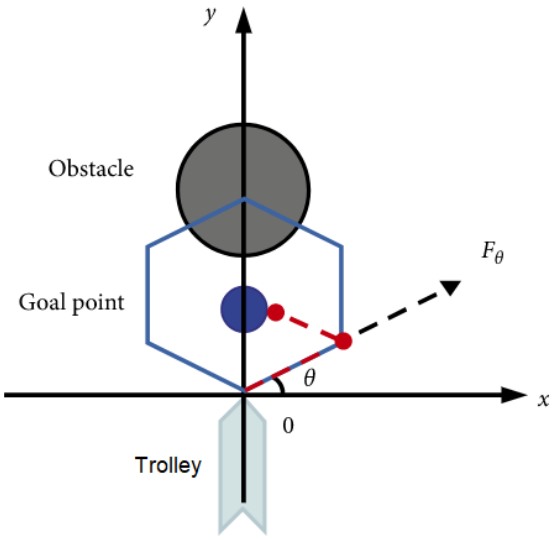

Figure 13: RHG solution with θ positive

As shown in figures 12 and 13, we conclude that the trolley is stuck in the local minimum area. Assume that the coordinate origin is the local minima location. Using the line between both the trolley and the target point as the Y-axis. View as the x-axis the line that crosses the point of origin opposite to the y-axis. Make a virtual normal hexagon with the width of the length equal of one-step position.

When the trolley catches in a local minimum area, an angle along the x-axis is created. Fθ is the next stage size's motion path, which will assist the trolley in leaving the local minimum zone. The trolley rotates (90- θ) degrees clockwise when θ is positive. When θ is negative, the trolley rotates counter-clockwise (90- θ) degrees. Since the inner angles of a typical hexagon are all 120 degrees, angle θ is a constant, θ= 30. The trolley moves in a near-determination virtual regular hexagon, but not in the area where the minimum local area is removed. The RHG solution will finish if the distance between the trolley and the target point is less than that of the obstacle when the target point goes beyond the radius of the obstacle. We analyse the RHG system further to ensure that the trolley can choose local minimal immediately and increase navigation reliability.

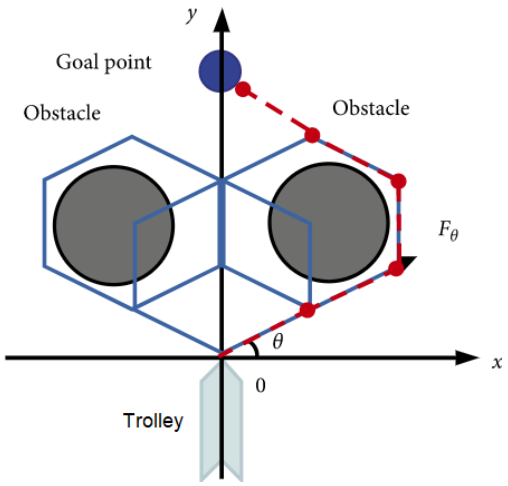

Figure 14: Trolley is centred in the minimal local area

1. This figure 14 shows that the trolley is centred in the minimal local area where the obstacles are perfectly symmetrical arranged. The trolley scans the obstacles around with sensors and finally locating the obstacles. Now we are building a two-dimension coordinate and a normal regular hexagon. The first standard hexagon will be applied to the cellular mobile network if the region of the obstacle is high. The trolley travels in the usual hexagon position until the obstacle is a long way off.

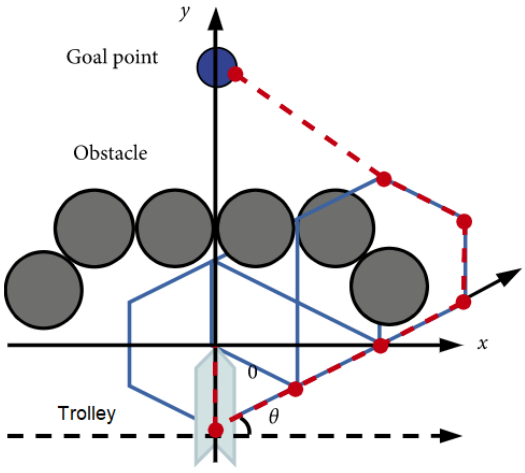

Figure 15: Trolley meets a concave obstacle

2. The local minimum is observed as the trolley meets a concave obstacle and as a consequence it is operated in a loop, as seen in figure 15. The trolley goes backwards horizontally or, if necessary, turns again. The trolley constructs a virtual regular hexagon and a two-dimension coordinate structure until it reaches enough further from the local minimum area. The trolley builds the steering system by sensing and identifying hazards and thereby removing the minimum local area. You should find the right way to clear obstacles.
   From this review it is clear that there are two clear benefits of the RHG approach. On the one hand, because of the incorrect parameter distribution, this approach should prevent the occurrence of foldback and oscillations. This approach decreases measurement stress by the fixed parameter dramatically, on the other hand.

3.2.3.3. The Obstacle Avoidance Dynamic Environment Problem Statement and Solution

The path planning can be determined by conventional APF approach if the trolley's speed and obstacles moving are slow. The trolley can collide with moving of the obstacles by the conventional APF method if the speeds of a trolley or moving obstacles are high. The explanation is that the standard APF approach does not take into account the influence of moving speed and applies static obstacles. The standard APF system is used to clear obstacles if the trolley deviates from moving obstacles. This leads to a longer route and reduces navigating speed.

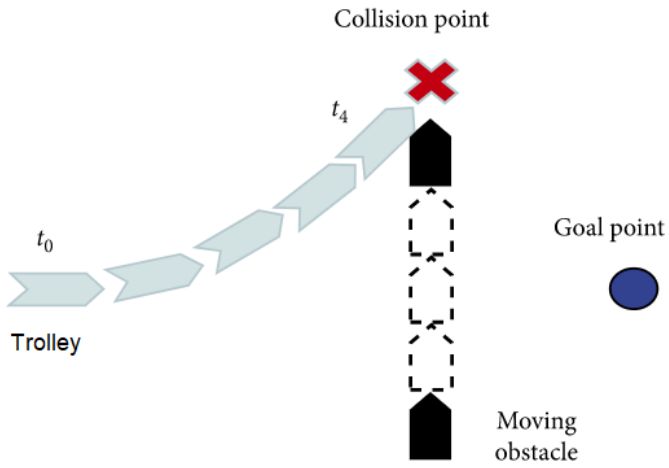

Figure 16: Trolley is influenced by the repulsive t0 area.

From the figure 16 we come to know that the trolley is influenced by the repulsive t0 area. The resulting force calculation continues to alter direction. At t4, the trolley found the location of the next step. However, due to high speed which can pose a risk of accident, the trolley did not change direction in time.

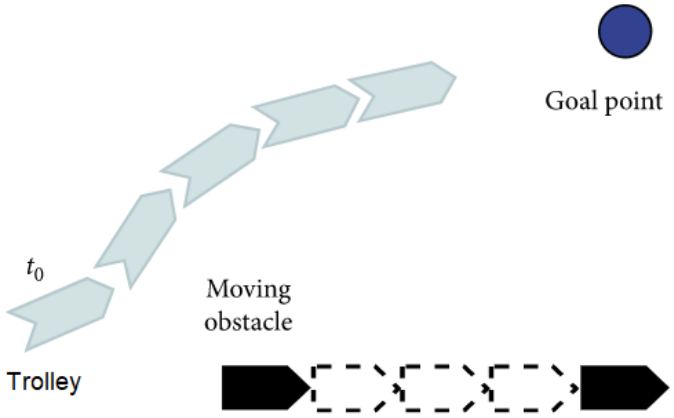

Figure 17: Trolley approaches the t0 field of impact

From the figure 17, it is seen that the trolley approaches the t0 field of impact and changes direction of a moving obstacle. In reality, the moving obstacle tends to leave from the trolley. The trolley is soon to step out of the obstacle's effect and into the target under the control of the attractive force alone. There is no chance of accident even though the trolley does not change direction.

The APF approach has to understand not just the spatial position but also the orientation and magnitude of the obstacles moving in order to solve the issue of the complex environmental obstacle avoidance. The relative speed method is proposed in this article. We also increase the repulsive function of the potential field U_{rep} (X). Use the relative speed component between the trolley and the advancing obstacle towards the obstacle as a judgement of obstacle prevention. The following are described as the current repulsive potential field function U_{rep} (X,V):

$$U_{rep}(X, V) = \begin{cases} U(X) + U(V), & \mathrm{d}(X, X_o) \le \mathrm{d}_o \; and \; v_o \ge 0 \\ 0, & else, \end{cases}$$

$$U(X) = \frac{1}{2} k_{rep} \left(\frac{1}{d(X, X_o)} - \frac{1}{d_o} \right)^2 d^n(X, X_o)$$

$$U(V) = k_v \left(\frac{(v - v_0)^T e_{ao}}{d(X, X_o)} \right) = k_v \frac{v_{ao}}{d(X, X_o)}$$

Here,

v_o is the moving obstacle's velocity

v is the trolley's velocity

U (V) is the moving obstacle's repulsive potential field.

K_v is a constant for the velocity repulsive potential field.

E_{ao} is the acceleration vector for the repulsive potential field which is pointed from the trolley to the moving obstacle.

d (X, Xo) is the distance between the moving obstacle and the trolley

$(v- v_o)^T e_{ao} = v_{ao}$ is the relative velocity between the trolley and the moving obstacle.

If $v_o >= v$, the trolley is planning a path which is behind the obstacle.

If $v_o < v$, the trolley is planning a path which is in front of the obstacle.

$v_{ao} >= 0$ describes the trolley moving towards the moving obstacles

$v_{ao} < 0$ describes the trolley deviating from the dynamic obstacle.

The improved repulsive force $F_{rep}(X,V)$ for the trolley in a dynamic environment is as given,

$$F_{rep}(X, V) = -\Delta U_{rep}(X,V)$$

$$\begin{cases} F(X) + F(V), & \mathrm{d}(X, X_o) \le \mathrm{d}_o \; and \; v_{ao} \\ 0, & else, \end{cases}$$

where,

$$F(X) = -\Delta_X U(X) = F_1(X) + F_2(X)$$

$$= K_{rep}\left(\frac{1}{d(X, X_o)} - \frac{1}{d_o}\right)\frac{d^2(X, X_g)}{d^2(X, X_o)}$$

$$+ \frac{n}{2} k_{rep}\left(\frac{1}{d(X, X_o)} - \frac{1}{d_o}\right)^2 d^{n-1}(X, X_g)$$

$$F(V) = -\Delta_V U(V) = -k_v\left(\frac{v_{ao}}{d(X, X_o)}\right) = k_v \frac{v_{oa}}{d(X, X_o)}$$

Where $F(X) = F_{rep}(X)$. $F(V)$ being the velocity of the repulsive force.

The updated resultant force for the trolley is shown in figure 18,

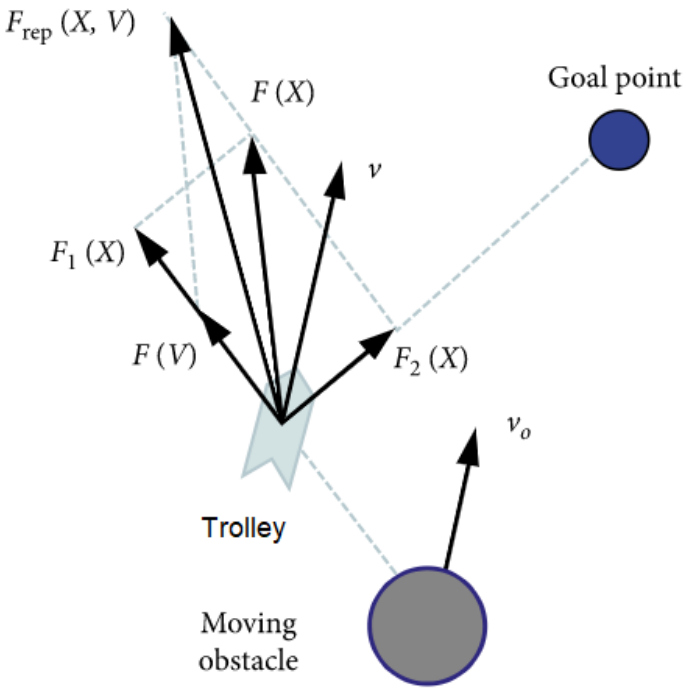

Figure 18: Updated resultant force for the trolley

The enhanced potential field for repulsive vector feature makes it possible for the trolley to become more repulsive as the obstacles are sensed. This will increase your safety considerably.

Use $(v\text{-}v_o)^T e_{ao} = v_{ao}$ to judge between trolley and travelling obstructive status in relative movement. An order for the prevention of blind obstacles.

The relative motion between the trolley and moving obstacle is shown in the following figures, l1 is the joint line between the trolley and the centre of the obstacle. l1 and l2 are parallel to each other, and l1 and l3 are perpendicular to each other. V_{ao} is the relative velocity along the direction of l1.

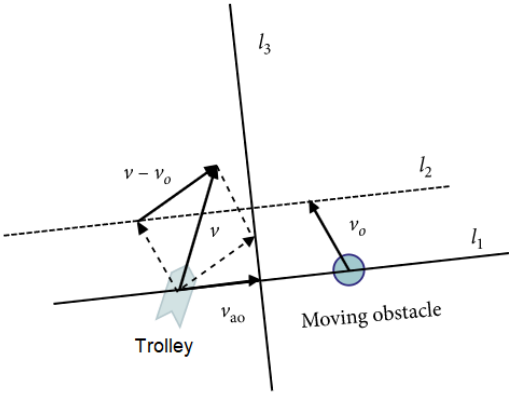

Figure 19: $V_{ao} >= 0$

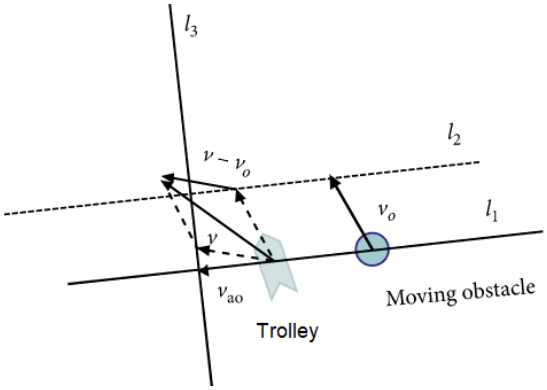

Figure 20: $V_{ao} < 0$

In figure 19, V_{ao} points to the moving obstacle and $V_{ao} >= 0$. The trolley moves further towards the moving obstacle and preventing obstacles becomes an unavoidable occurrence.

In figure 20, V_{ao} leads to the direction opposite of the moving obstacle and $V_{ao} < 0$. The trolley deviates from the moving obstacle and obstacle avoidance are not necessary.

3.3 Environment design:

An environment has a boundary, geometric surrounding, static obstacle and dynamic obstacles. Let us know about each and everything in a separate manner.

3.3.1 Boundary of the environment:

The environment should have a boundary within which the trolley has to travel considering the boundaries as static obstacles.

Here in our case study, the boundaries are the walls beneath the supermarket and the trolley must know that it should not cross the boundary and hence the environment is built with boundary as shown in figure 21. Hence, we use a 12x8 spatial dimension in a supermarket environment with boundaries at each sides.

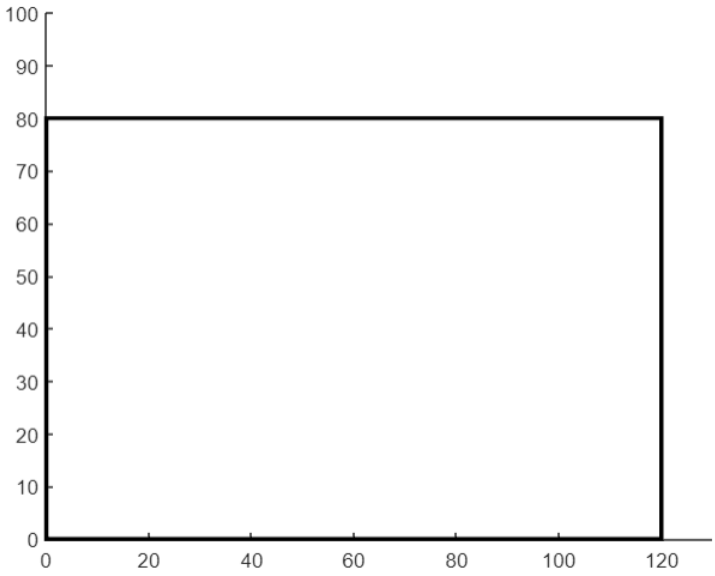

Figure 21: An environment with a boundary

Static and dynamic objects with various shapes and possible roles can be found in the field. The scale, colour, quantity, location, and degree of influence of these elements can all be easily changed. The world, on the other hand, includes the robots and their associated targets, but this does not have as much editability. It is only necessary to adjust their colour and location. The following subsections would go through all of the information concerning these elements in greater depth.

3.3.2 Geo-metrical Environment in both Static and Dynamic:

The path planning formula of a new framework with geometrically based obstacles is applied in a simplified setting, that both the conventional and proposed approaches were subjected to a simulation analysis. If more than one wall is attached to form a group, the robot may take the wall group centre into account in addition to the wall centre. The algorithm would decide the best path for the current repulsive force part of the updated Artificial Potential Field method by considering the probability of avoiding the artefacts identified by a collection of walls. These diagrams depict various scenarios in which the robot makes multiple decisions depending on section distribution.

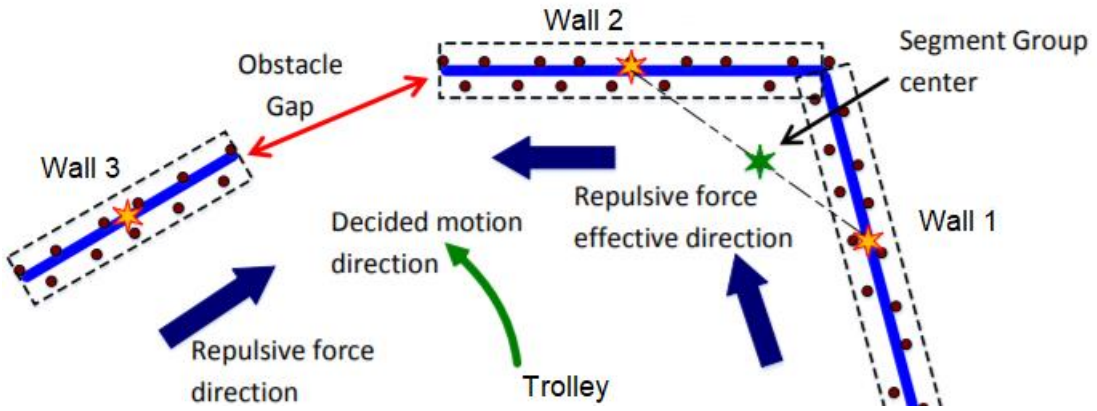

Figure 22: Case 1 for a geo-metric environment

The walls 1 and 2 in figure 22 belong to the same thing. As a result, the section group is used to determine the position of the latest repulsive force part. In this situation the space between walls 2 and 3 is necessary (obstacle gap). Walls 1 and 2, because of the repulsive force the robot is moving to the left, while wall 3 creates a force pushing the robot to the right. The overall force will take the robot to the area open for the target (direction of movement is calculated).

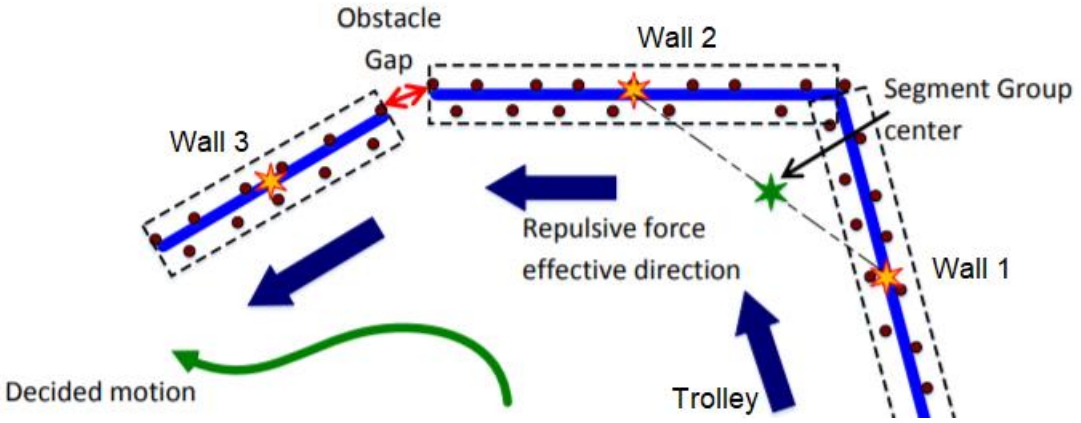

Figure 23: Case 2 for a geo-metric environment

If the distance here between walls is insufficient to allow the trolley to proceed, as seen in figure 23, the method chooses the direction of the third wall based on the direction of the first and second walls. Finally, in this case, all of the walls produce repulsive force in the same direction.

This method can be run continuously, and until the path of a segment is determined, it is saved in data and information and later used for the next phase of the motion.

3.3.3 Static Environment:

For static obstacles to be defined in an environment, we must assume some shaped obstacles say a rectangular obstacle in a regular dimension as shown in figure 24. Here in our case, the static obstacles include shelves of the supermarket, bins placed all around.

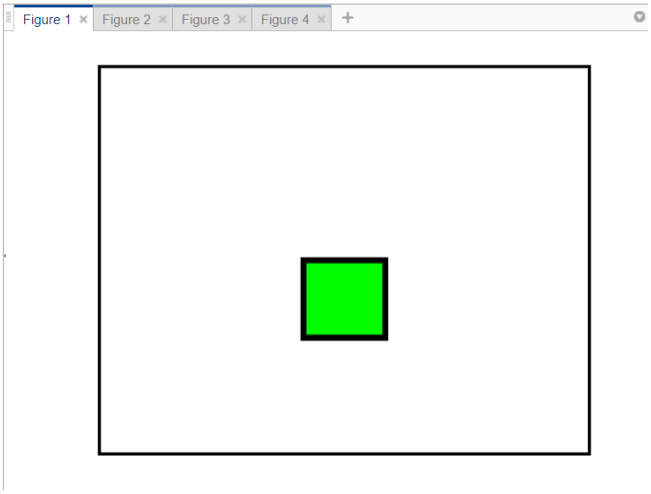

Figure 24: A static obstacle in an environment

The potential force contour for the static environment is as follows in figure 25,

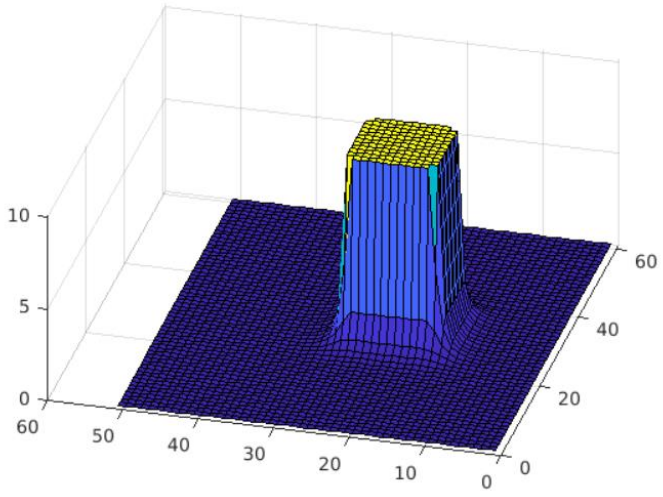

Figure 25: Potential Force Contour Diagram

3.3.4 Dynamic Environment:

The dynamic environment is created below in figure 26 with rectangular functions. The green, red and yellow rectangular blocks are the dynamic obstacles. We can say them as, people, other trolleys, etc. They are created in matlab functions and in such an environment, all the obstacles perform vertical movement from bottom to top in the simulation.

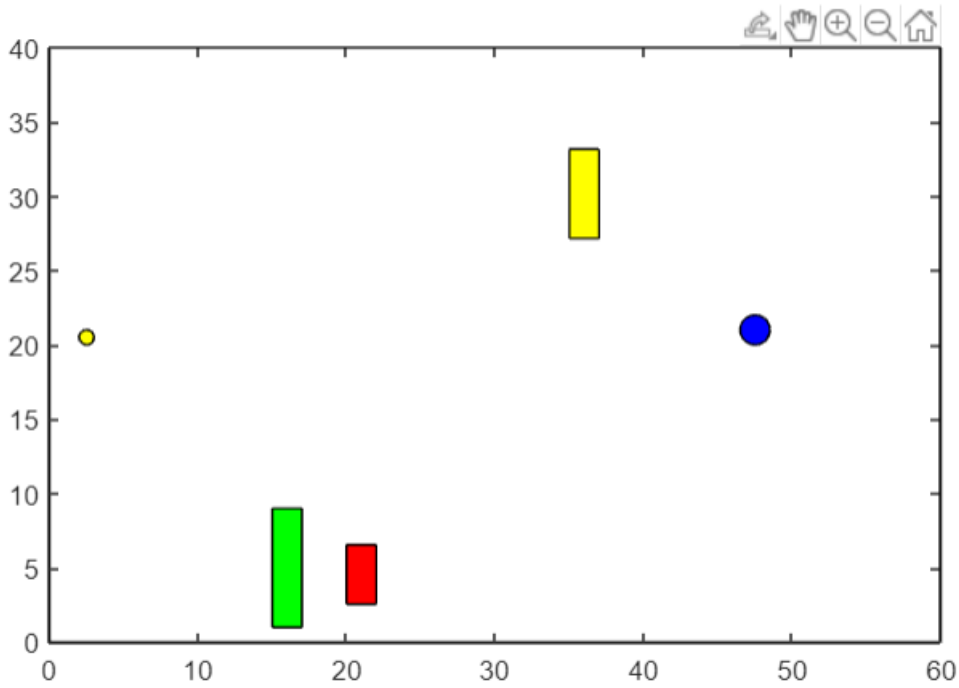

Figure 26: A dynamic environment

CHAPTER FOUR
SIMULATION AND RESULTS

The technique is tested on MATLAB R2017b in case the trolley is a particle in order to determine the performance of the proposed method. The collection and configuration of simulation parameters depends on our assumptions. To simplify simulations, no complexities including noises between the trolley and the obstacles are applied to the simulated steps.

When the trolley is released by the customer after shopping, the trolley makes itself ready for path planning and gets an idea about the path aiming at the till of the supermarket. Once it starts, it detects any obstacles which includes both static and dynamic including people inside the supermarket, other trolleys, the shelves, wall and doors of the supermarket. Now the simulation for the static and dynamic environments are displayed below.

4.1 Static Environment Simulation

Here, all the static obstacles in the following simulations indicate the shelves of the supermarket where the products are placed, other static trolleys, bins etc. The goal point is always the same which is the 'Counter/ Till of the Supermarket'.

Both environmental details in route design issues are known beforehand in static conditions, and the motion of the trolley is steady. This type of trajectory planning problem is often called track planning problems in a known environment, which establishes all information about artefacts in the workplace. There will be no modifications to the determined direction during the trolley's motion if the path is designed before the trolley begins to travel. The path planning occurs in two environments such as known and unknown static environments. In a static environment which known, the robot should be aware of the whole environment before beginning to drive. As a result, leading up to the mission, the trolley will calculate the best collision-free path offline. The trolley navigation is more complex in a static and unknown system than in a static and established environment. It is owing to the variation in the input obtained from the environment's sensors. As an outcome, the desired result cannot be achieved, and the trolley must determine the direction while in movement using local knowledge.

4.1.1 With a Single Stationary / Static Obstacle:

For this simulation analysis, the trolley began moving from point (1, 1), the goal is placed at (8, 9), and there is one obstacle on the bridge joining the trolley and the goal at (5, 6). For each move towards the target, the algorithm must decide the way points of the trolley's path. At the start of the trolley's motion, it travelled steadily and without delay until it reached a distance of 7 units in the XY direction. However, the effect of the barrier became evident after that place, and the total repulsive strength started to fall. As the trolley passed up to the barrier, the overall force approached its minimum value nearby, behind the obstacle.

The path plan is shown as follows in figure 27.

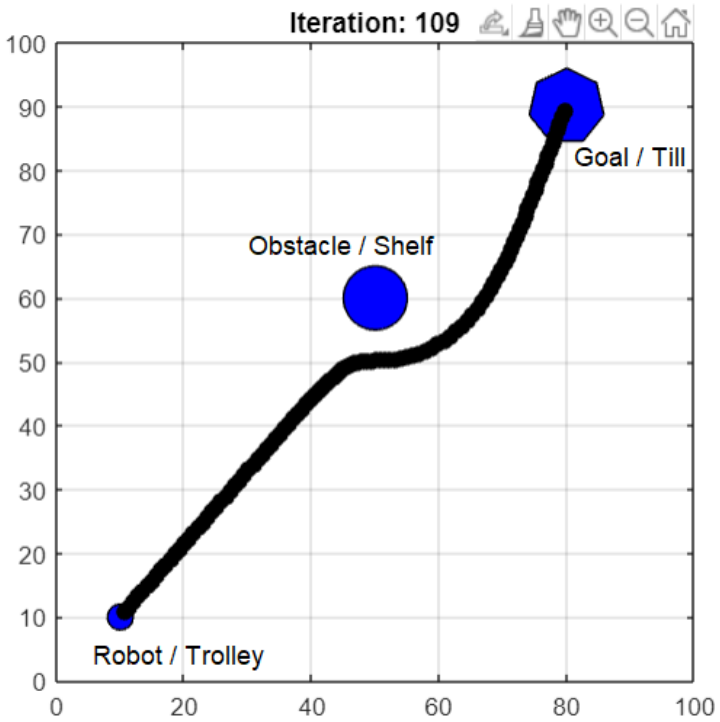

Figure 27: A single static obstacle Environment

The repulsive force is produced at the point (4.5, 5) where the repulsive force first appeared, and thus the trolley begins to create a motion of curvature rather than a straight movement. Increased force against direction is more significant in that the secondary repulsive force acts perpendicular to the principal repulsive force; therefore the route of the overarching potentials change and the trolley starts moving around the barrier and achieves its objective.

The Contour patterns we have obtained is shown in the figure 28 below,

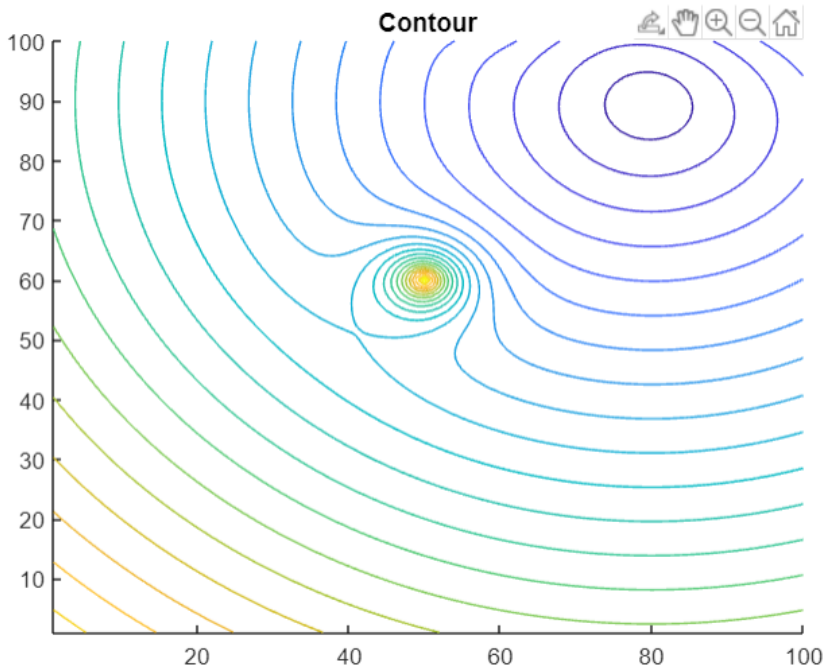

Figure 28: Contour patterns

The Potential Force Field on the Contour is shown in the figure 29 below,

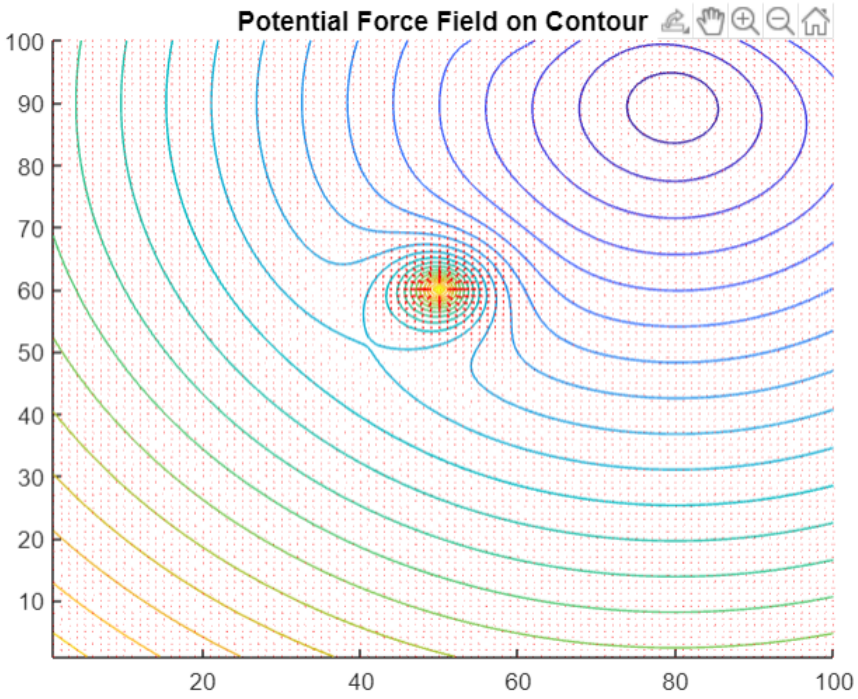

Figure 29: The Potential Force Field on the Contour

4.1.2 With Two Symmetrically placed obstacles:

For this simulation test, we have shown that the environment has two symmetrically positioned obstacles placed at points; (6, 6), (4, 7), and the goal point is positioned (7, 9). Simulations have been done as seen in the figure 30 below to solve the deadlock problem of symmetrical distribution obstacles.

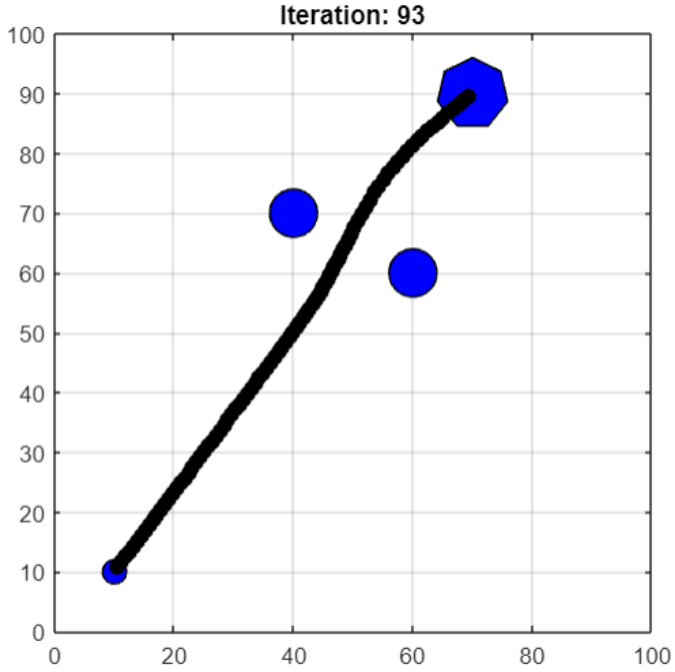

Figure 30: Two symmetrically placed static obstacles

The contour patterns for this kind of environment is shown below in figure 31,

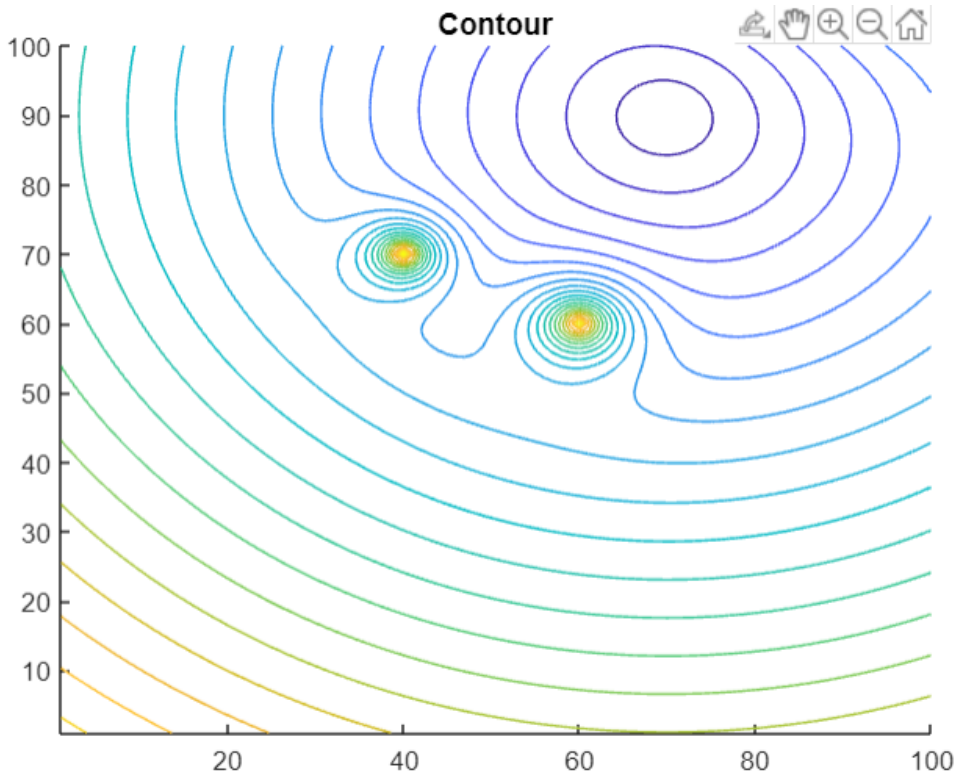

Figure 31: Contour patterns

The potential field force for the contour is shown below in figure 32,

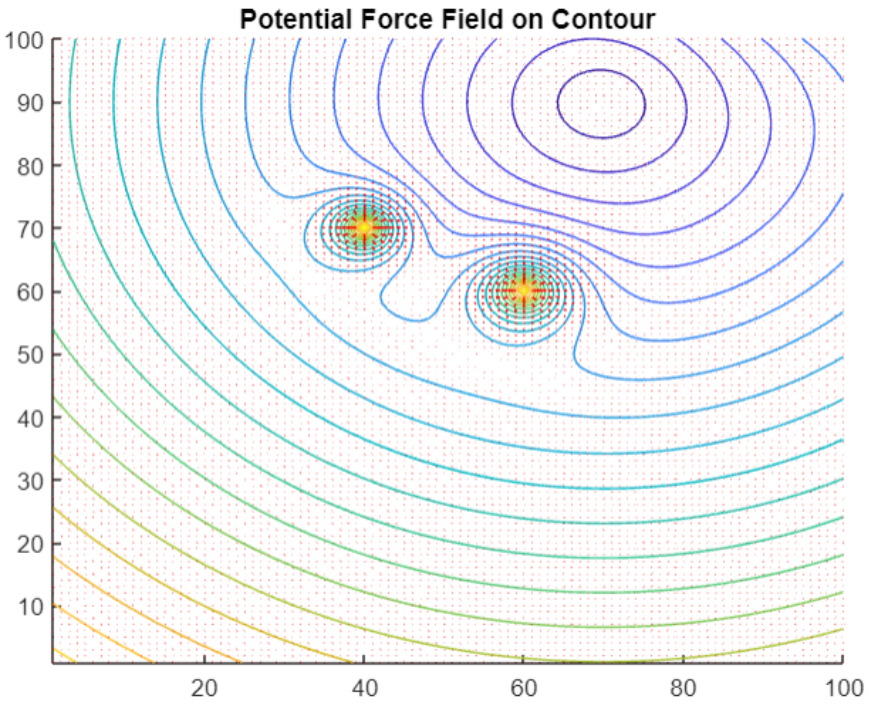

Figure 32: Potential field force for the contour

The iteration diagram is shown in figure 33 as follows,

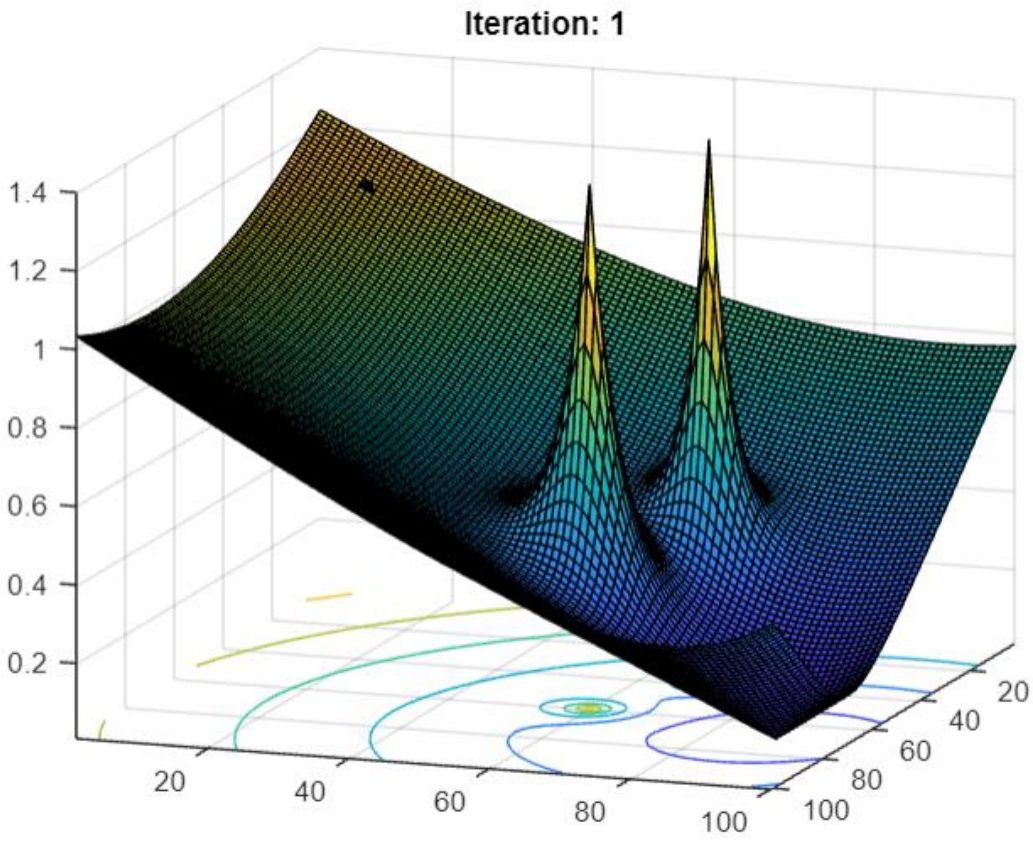

Figure 33: Iteration Diagram

The relative advantage of the proposed approach over the conventional method is clearly seen. When the robot approaches the obstacles, its potential force begins to decline, but it begins to raise due to the extreme new repulsive component while approaching zero and before hitting obstacles.

This is under any circumstances because the current force component creates a rotary force to the path of the target only and increases until the robot crosses the barrier. The robot is capable, even after the obstacles are overcome, of detecting the obstacles, and thus of repulsive force in the direction of the target.

The overall potential force has thus been raised to a limit and began to decline although after a viewing angle to the obstruction the current repulsive force portion fades. Eventually, at the finishing stage the cumulative possible power is zero. By using this simulation case study the suggested approach has been shown to be able to overcome conventional method disadvantages, such as a dead-lock for various scenarios at local minimum, including the non-reachability condition.

4.1.3 With Multi Stationary / Static Obstacle:

For this simulation analysis, we have shown that the environment has 4 static obstacles placed at points; (3, 2), (3, 4), (5, 6), (7, 7), and the line path has been changed according to the obstacles without hitting them but also following the line path after crossing the obstacle. The trolley has reached the goal (8, 9). In figure 34, black path that our latest APF is designed for is collision free and global optimisation, ideal for the management of mobile robots.

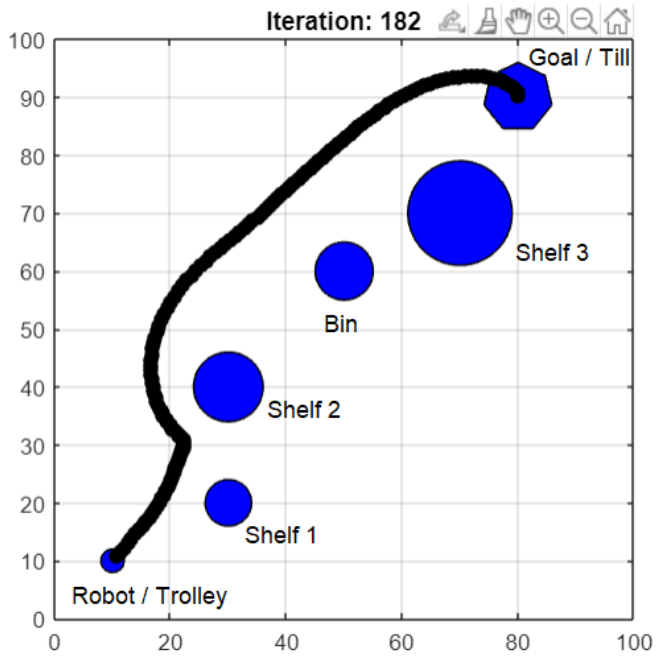

Figure 34: Multi-static obstacle Environment

At the start of the trolley's motion, it travelled steadily and without delay until it reached a distance with a gap of hitting a static obstacle. Now the path is planned immediately based on the obstacles in the environment making sure the trolley does not hit anything and finally reaches the till. The contour patterns of the environment are shown below in figure 35,

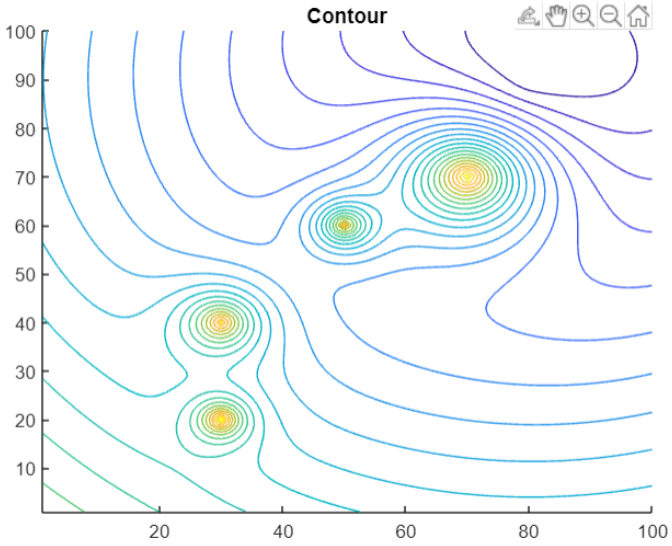

Figure 35: Contour diagram

The Potential Force Field on contour is shown below in figure 36,

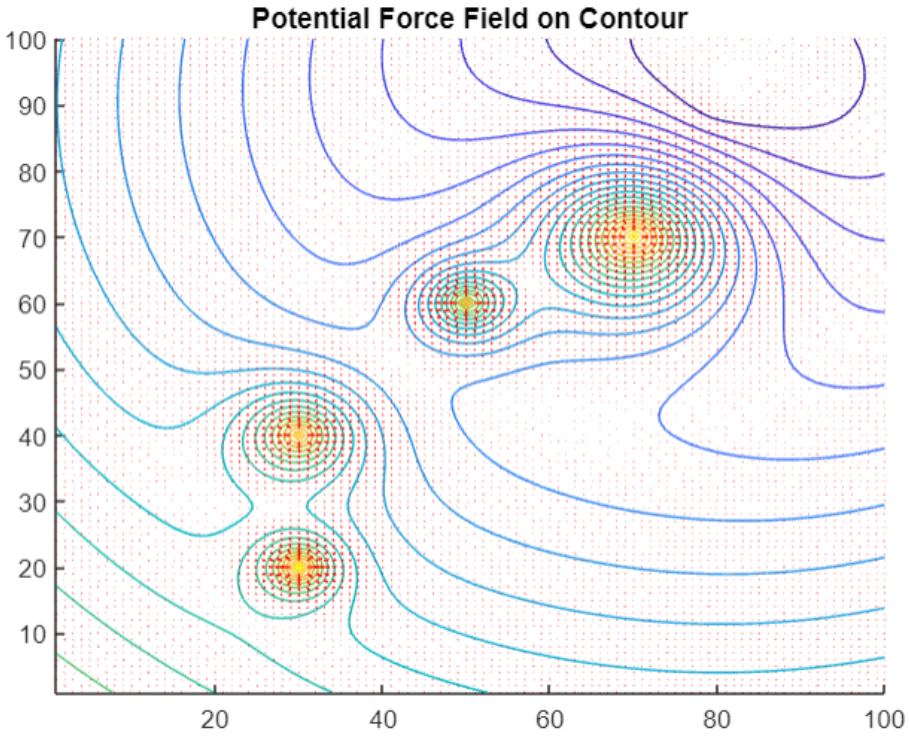

Figure 36: Potential Force Field on contour

The iteration figure is shown below in figure 37,

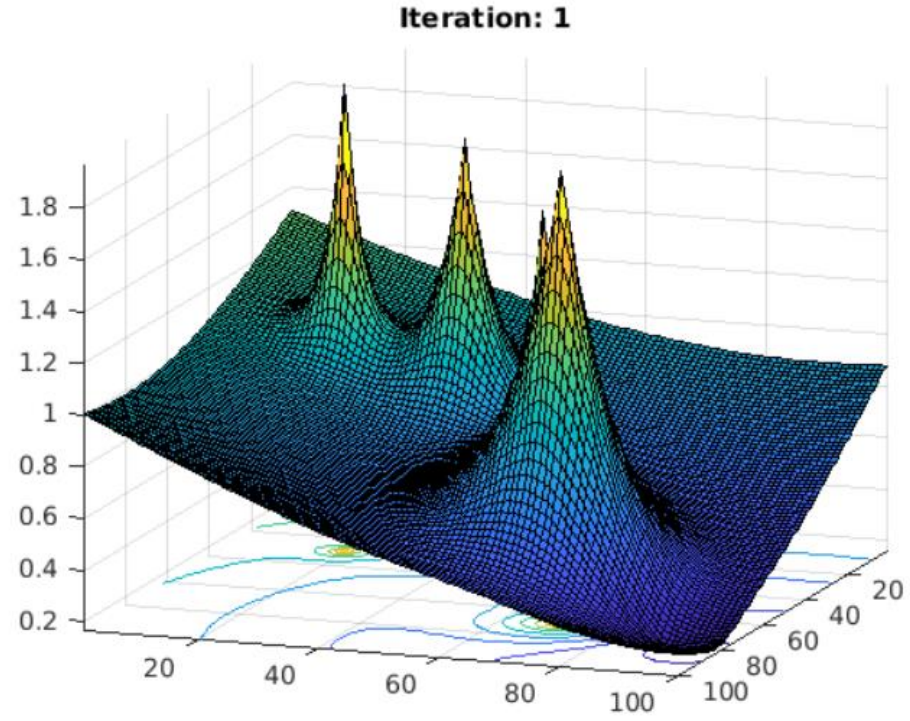

Figure 37: Iteration Diagram

4.2 Both Static and Dynamic Environment Simulation:

Here, all the dynamic obstacles in the following simulations indicate the people shopping inside the supermarket, other dynamic trolleys moving from its respective customer to the goal point.

The goal point is always the same which is the 'Counter/ Till of the Supermarket'.

In the static and dynamic path design issues, no prior knowledge about the terrain is provided or is only understood in part, i.e. the observable sections for the obstacles. As the trolley arrives at its destination, it prepares its route using the knowledge at its disposal. If the trolley progresses, further obstacles are encountered that are employed to change the landscape and schedule the route. This sequence of changes and reorganisation of environmental knowledge is replicated before the car reaches its target.

This path planning also occurs in two environments such as known and unknown dynamic environments. The trolley is conscious of other moving objects in this kind of established situation, such that it can receive information about its location, speed, acceleration and so on. Such information can be accessed from any moving object or a different central communication system. The most difficult case in trolley path planning is path planning in a complex and unknown setting. This is perhaps the most likely scenario that a mobile trolley would face. The trolley cannot use a method of global path planning and must focus on local path planning under certain diverse and unpredictable circumstances. Global optimization is hard to obtain. The trolley must be outfitted with a variety of sensors in order to gather knowledge about its surroundings and prepare its journey in real time.

The planning time of the trolley should be brief, as the trolley requires a sufficient amount of time to change its course to avoid obstacles. For the dynamic environment simulation, the following pictures depicts the path planned by the trolley and further moves without hitting the obstacles.

Both static and dynamic obstacles are in such a retail environment. Now the trolley learns that the customer has finished shopping and ready to pay. So the trolley travels to the till without any disruption in view of the obstacles. The static obstacles (blue rectangle blocks) here mentioned are the gondolas and the dynamic obstacles (red rectangles blocks) are the customers inside the supermarket. The first green circle is indicated as the starting point of the queue in the till and the second green circle is the payment point (till) of the supermarket. All the dynamic obstacles are directed with an arrow for easy understanding. The path of the trolley is indicated in yellow.

The figure 38 shows the initial condition for a both static and dynamic environment.

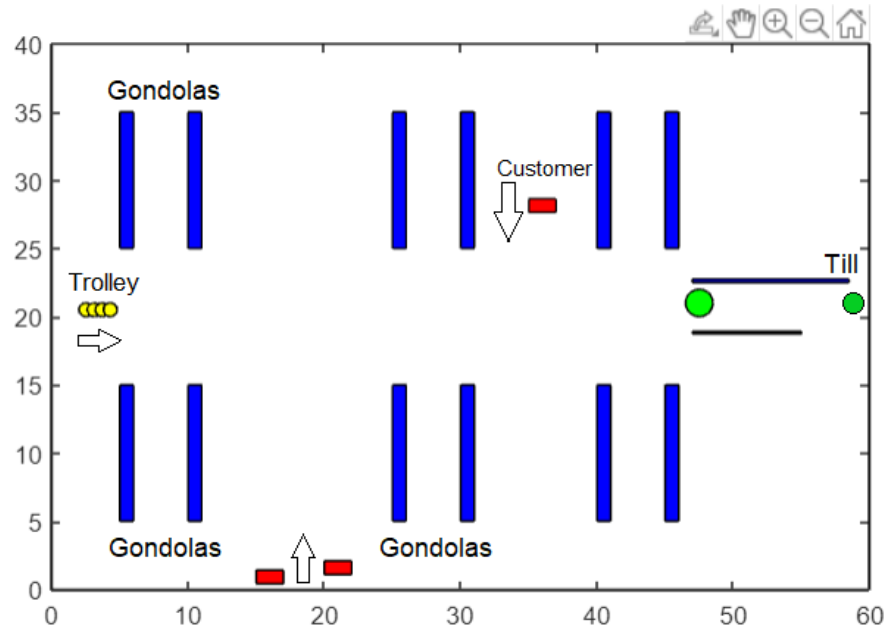

Figure 38: Initial environment with both static and dynamic obstacles

The next figure 39 now shows the state of the environment after 2 seconds. The trolley has already started to cross one of the customers who is dynamic in the environment and about to cross another customer too.

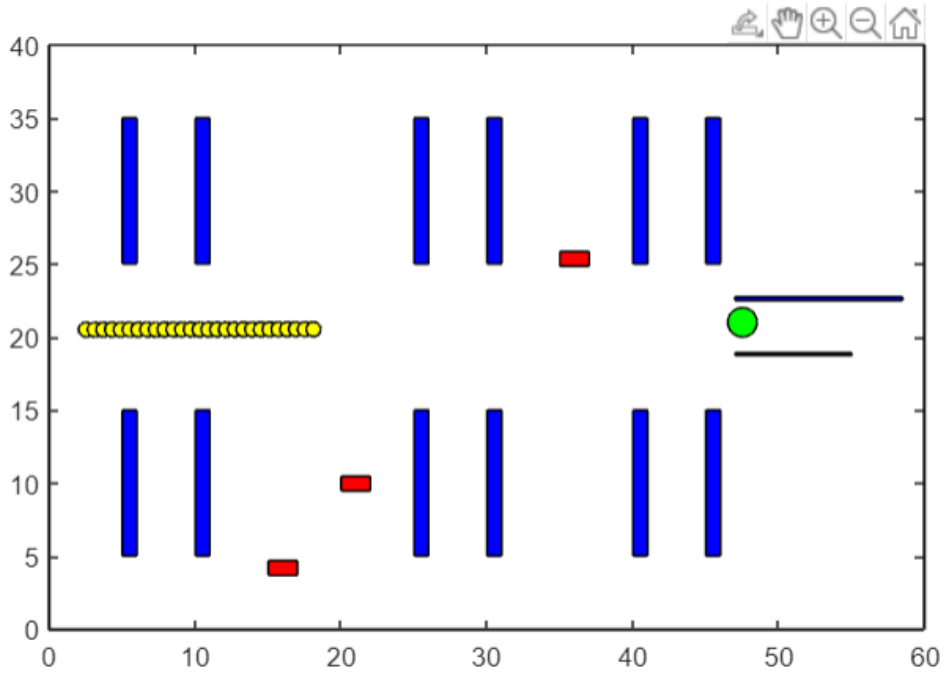

Figure 39: State of the environment after 2 seconds

After this, we could see that in figure 40, the trolley has passed the second obstacle and is heading towards another dynamic customer who's really approaching the trolley perpendicularly.

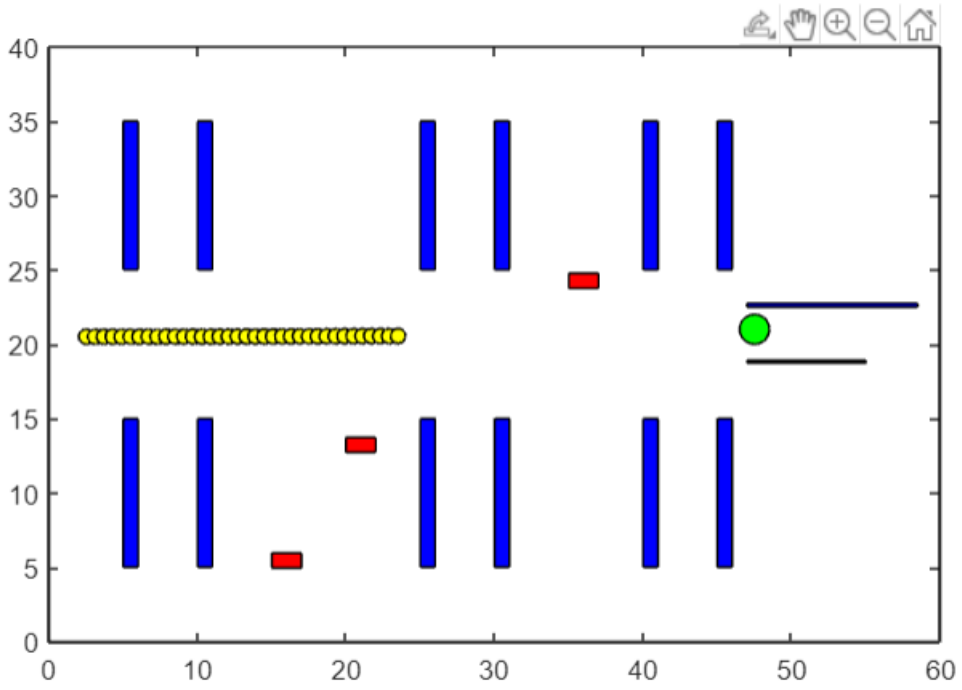

Figure 40: Trolley crossed static and 2 dynamic obstacles

Now the next figure 41 shows that after crossing the 2 dynamic obstacles, the path of the trolley has started to change a bit detecting an obstacle in front of it. The trolley immediately understands that it is going to hit an obstacle and so decides to change the path. From the algorithms simulated in Matlab, we come to know that the trolley moving with a particular velocity detects an obstacle through its path before a distance of 2 metres. Suddenly the velocity of the trolley increases and also the angle of the path is deflected, trying to avoid the obstacle.

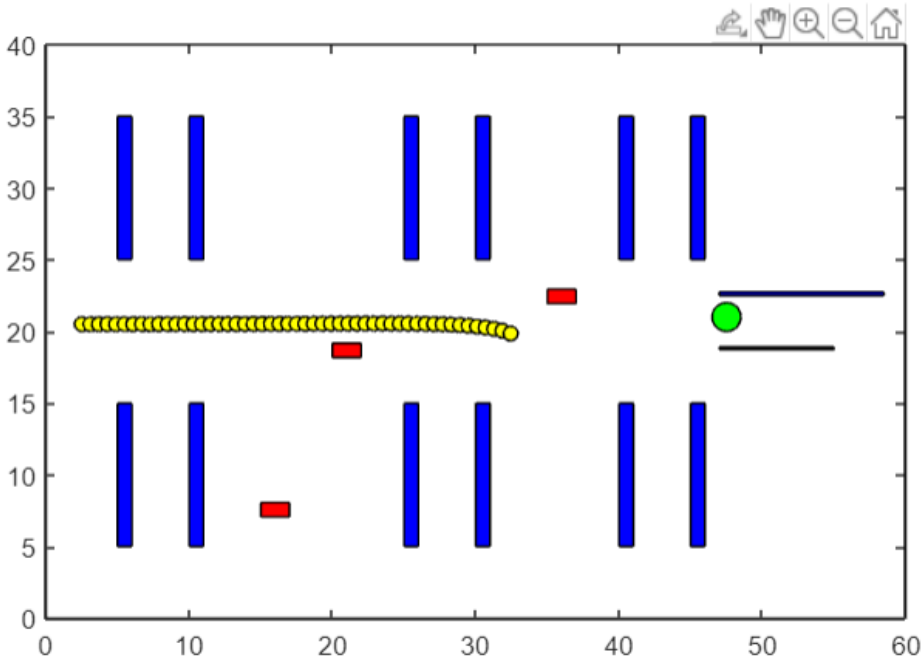

Figure 41: Trolley's path has changed

In the following figure 42, its clearly seen that the trolley has increased its velocity considerably and crossed the obstacle without hitting or without being hit.

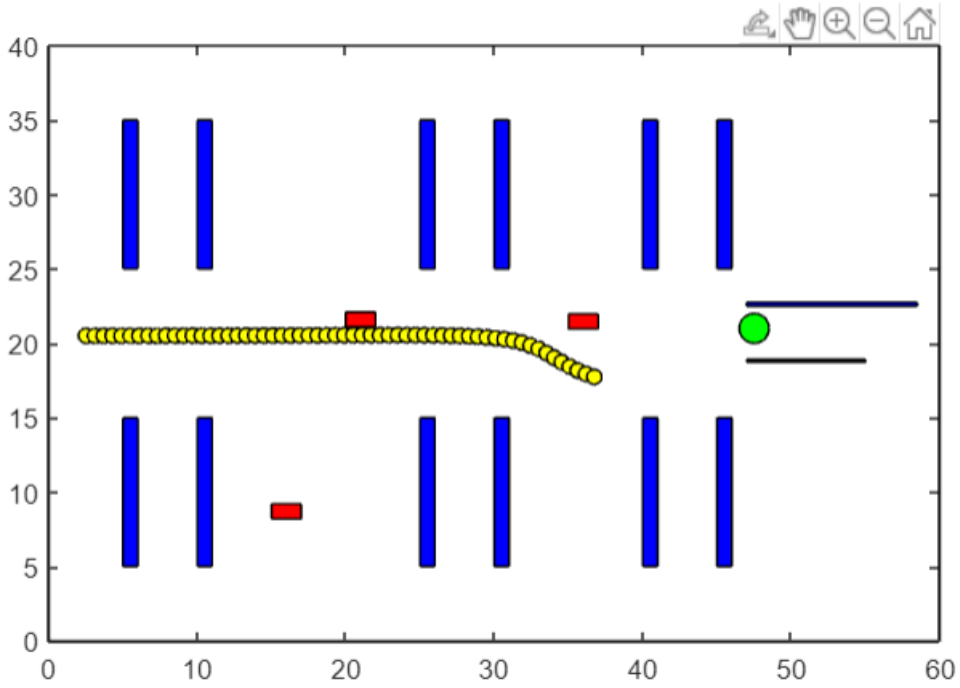

Figure 42: Trolley's velocity increases to cross the obstacle

In the next figure 43, we can expect the deflected path of the trolley to end up in its original path, trying to change its angle back again. The position of the starting point of the queue is almost near and the trolley should be back in its original path planned.

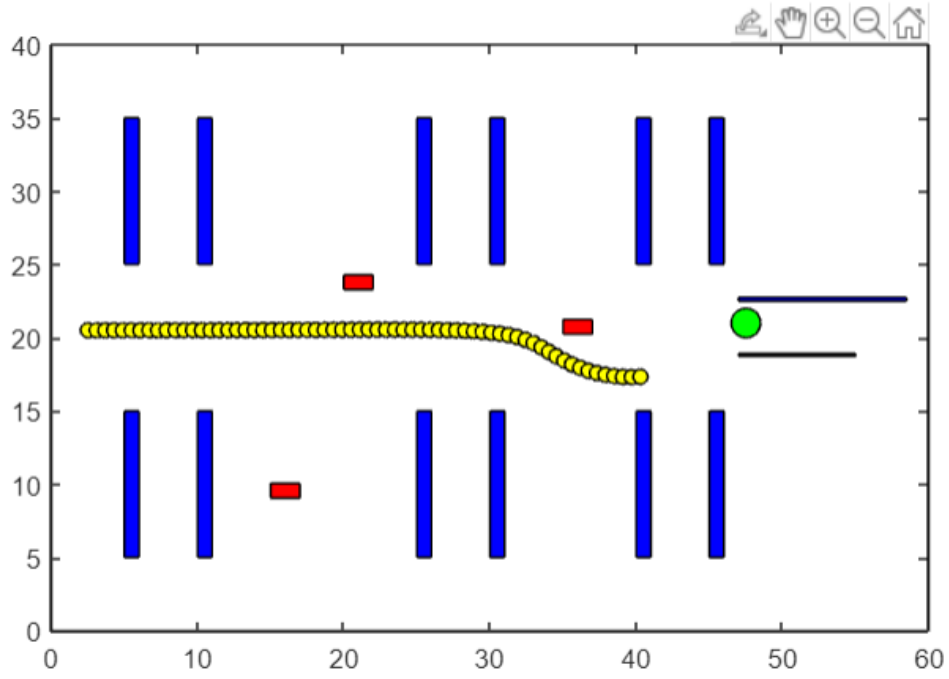

Figure 43: The deflected path changing to its original path after obstacle avoidance.

Finally, in figure 44 the trolley avoiding all the obstacles in the environment has reached the starting point of the queue in the supermarket. The moving customers, other moving trolleys and the racks of the supermarket are being avoided in the path planning method and hence the goal point is reached by the trolley. Now the trolley reaching the point, notices that there is already a trolley being billed in the till and waits till it is eliminated. The other trolley is also taken as a moving obstacle here in this environment. The algorithms are made in such a way that, when a secondary trolley is in a radius of 2 metres from the path planned trolley, it movement stops and waits for a condition with obstacles not in a 2m radius.

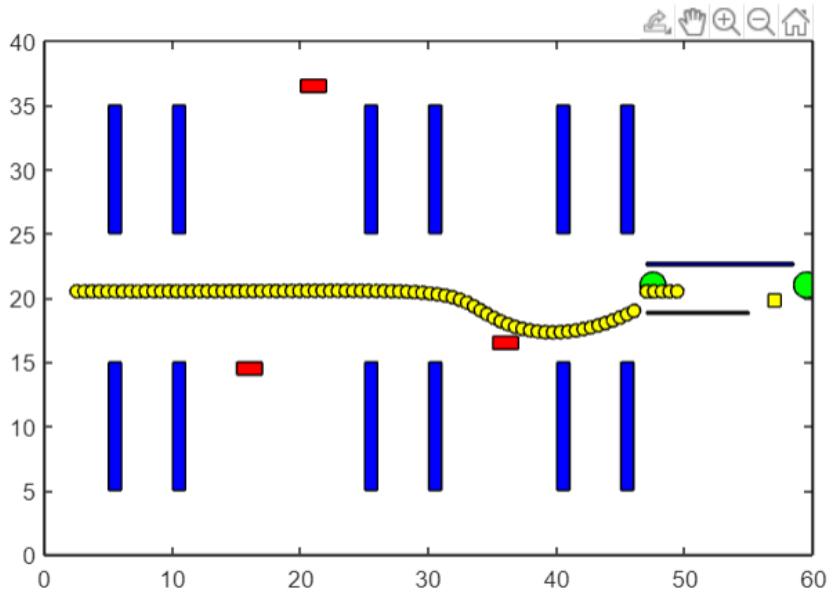

Figure 44: Trolley reaches the starting point of the queue

In fig 45,as that secondary trolley has finished payment, the path planned trolley tries to reach the till in a straight direction allowing the other trolley to move away in a 2m radius gap.

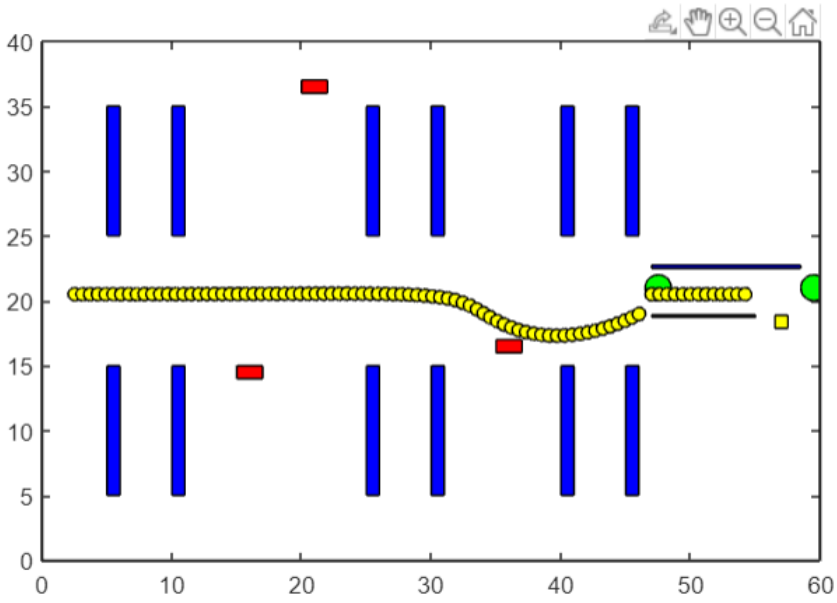

Figure 45: The trolley starts to move to the till

Finally, the path planned trolley reaches the till for payment and successfully completes an APF based path-planning. Now in this supermarket environment, the trolley has overcame both the static as well as dynamic obstacles leading its way to the till as shown in figure 46.

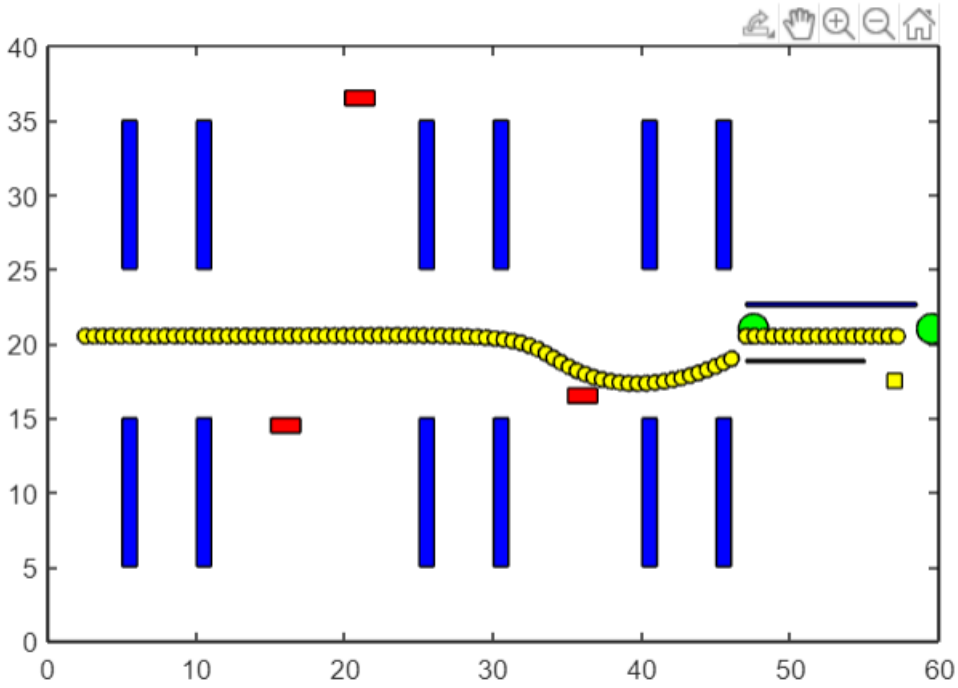

Figure 46: Entire Path Planning is completed

The outcomes of these simulation demonstrate the efficacy of the proposed algorithm in avoiding obstacles. As the suggested approach for route planning includes both improving the APF and selecting a technique for preventing obstacles, it has some benefits for seeking a trolley direction that is smooth and free. If the trolley is within the obstacle distance, the RHG approach can overcome local minimal problems effectively, escape barriers and hit the target point. If the trolley is beyond the obstacle's reach, it moves directly to the target without preventing obstacles. The trolley will achieve a smooth movement direction in the obstacle avoidance method.

According to the simulation and experimental test, this approach has been shown to be effective for the avoidance of complex obstacles in uncertain and unstructured environments, improving the efficiency of robotic path planning considerably, reducing not only time requirements, but also reducing the distance. The outcomes of the research show: 1) The approach for preparing a static and dynamic environment is efficient. 2) Only the details of the trolley condition and the environmental area features should be used to ensure that action choices are made correctly.

A significant amount of simulation shows that the enhanced APF approach escapes local minima and oscillating motions very quickly. Besides that, the simulation outcome confirms that a more global optimum, nearly ideal, collision-free and protection approach towards its target can still be calculated according to general APF with our suggested approach to path planning.

CHAPTER FIVE

CONCLUSION

The simulation results from Matlab 2018a for both static and dynamic environments show that a path-planning is done avoiding the obstacles and also overcoming from different problems such as GNRON problem, local minima. The path planning challenge for autonomous mobile robots is a remarkably interesting navigation branch. For avoiding collisions for mobile robots, APF was normally used. The trolley can then choose to prevent obstacles by applying external and self-state conditions and moving obstacles. The correction factor is introduced on the basis of the standard APF approach to overcome the GNRON problem in the repulsive field function. This will quickly allow the trolley to hit the goal. The RHG way of solving the local minimal problem is in combination with the APF method. A generic regular hexagon is formed if the trolley is stuck in a local minimum area. That will lead the trolley to get out of the local minima and hit the target point. Planning the prevention of obstacles is one of the main research for the trolley such that the robot finds a stable and collision-free route from the point of departure to the terminal based on the specified methodology while obstacles exist all around destination. APF is a relatively mature approach that is commonly utilized due to its straightforward and clear mathematical estimation. Even so, due to the intrinsic boundedness of conventional APF, the trolley cannot indeed be applied explicitly, but in order to solve the issue that the robot cannot meet the limit in the traditional APF algorithm, this report proposes an updated obstacle avoidance algorithm.

It demonstrates that the use of the path planning methods can be extended to a challenging dense environment. This proves our improved APF approach for solving path planning, which is a difficult problem for the autonomous mobile robot, with high viability and performance.

Applications created with ease of acceptance, architecture and the shopping experience can be designed to be more involved and user-friendly. This facilitates and makes it more suitable for people to use the platform in comparison to current programmes throughout the shopping method. Bills are not to be paid by customers near cash registers. Since the inventory details you have ordered is passed to a single accounting system. Via credit/debit cards, the shoppers can pay their payment. The proposed scheme is highly autonomous, independent, genuine, reliable and timely. The machine is also highly time consuming. This scheme aims to achieve a quicker processing service. It helps the purchaser to understand the information specifics beforehand so that they can prepare for the price consequently; The benefits of this project are convenient to use, and no special preparation is needed.

The item would be highly praised by the public due to its distinctive characteristics. However, certain issues and progress in our future work do need to be addressed and carried out. A dynamic world and real-time experimentation with a mobile robot will be proposed for potential work. The safety and legal security of this device must be taken into account; nevertheless, this is not possible at this point as the pandemic lockdown have not produced a prototype currently.

REFERENCE

1. Y. Qin, D. Sun, N. Li and Y. Cen, "Path planning for mobile robot using the particle swarm optimization with mutation operator," in Third International Conference on Machine learning and Cybernetics, Shanghai, 2004.

2. Q. Zhu, Y. Yan and Z. Xing, "Robot Path Planning Based on Artificial Potential Field Approach with Simulated Annealing," in IEEE Sixth International Conference on Intelligent Systems Design and Applications ISDA'06, 2006.

3. R. Siegwart, I.R.Nourbakhsh, ”Introduction to Autonomous Mobile Robots” ,Prentice Hall,2005.

4. S. Liu, L. Mao and J. Yu, "Path Planning Based on Ant Colony Algorithm and Distributed Local Navigation for Multi-Robot Systems," in Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation, 2006.

5. M.Khatib, R.Chatila, “An extended potential field approach for mobile robot sensor-based motions”, Proceedings of Intelligent Autonomous Systems IAS-4.

6. Jose Mireles Jr. EE Systems and Control Course, Path Planning Mobile Robot.

7. P. Bhattacharya and M. Gavrilova, “Roadmap-based path planning—using the voronoi diagram for a clearance-based shortest path,” IEEE Robotics & Automation Magazine, 2008.

8. Xiaojing Fan, Yinjing Guo, Hui Liu, Bowen Wei, Wenhong Lyu. "Improved Artificial Potential Field Method Applied for AUV Path Planning", Mathematical Problems in Engineering, 2020

9. D. Hahnel, W. Burgard, D. Fox K. Fishkin and M. Philipose, "Mapping and localization with RFID technology", Proc. IEEE Int. Conf Robot. Autom, 2004.

10. Tharindu Weerakoon, Kazuo Ishii, Amir Ali Forough Nassiraei. "An Artificial Potential Field Based Mobile Robot Navigation Method to Prevent From Deadlock", Journal of Artificial Intelligence and Soft Computing Research, 2015.

11. Pu Shi, Yiwen Zhao. "Global path planning for mobile robot based on improved artificial potential function", 2009 IEEE International Conference on Automation and Logistics, 2009

12. Wang Siming, Zhao Tiantian, Li Weijie. "Mobile Robot Path Planning Based on Improved Artificial Potential Field Method", 2018 IEEE International Conference of Intelligent Robotic and Control Engineering (IRCE), 2018.

13. Van Bay Hoang, Van Hung Nguyen, Lan Anh Nguyen, Truong Dang Quang, Xuan Tung Truong. "Social constraints-based socially aware navigation framework for mobile service robots", 2020 7th NAFOSTED Conference on Information and Computer Science (NICS), 2020.

14. Lianyu Zhao, Yanqiang Wang, Jutao Wang. "Mobile Blasting Robot Obstacle Avoidance Planning", 2019 IEEE International Conference on Mechatronics and Automation (ICMA), 2019.

15. Ankit A. Ravankar, Abhijeet Ravankar, Takanori Emaru, Yukinori Kobayashi. "HPPRM: Hybrid Potential Based Probabilistic Roadmap Algorithm for Improved Dynamic Path Planning of Mobile Robots", IEEE Access, 2020.

16. C. L. Cheng, D. Q. Zhu, B. Sun et al., “Path planning for autonomous underwater vehicle based on artificial potential field and velocity synthesis,” in Proceedings of the Canadian Conference on Electrical and Computer Engineering, Halifax, Canada, May 2015.

17. O. Montiel, R. Sepúlveda, and U. Orozco-Rosas, “Optimal path planning generation for mobile robots using parallel evolutionary artificial potential field,” Journal of Intelligent & Robotic Systems, 2015.

18. X. X. Liang, C. Y. Liu, X. L. Song, and Y. K. Zhang, “Research on improved artificial potential field approach in local path planning for mobile robot,” Computer Simulation, 2018.

19. J. B. Sun, G. L. Liu, G. H. Tian, and J. H. Zhang, “Smart obstacle avoidance using a danger index for a dynamic environment,” Applied Sciences, Apr. 2019.

20. F. Matoui, B. Boussaid, and M. N. Abdelkrim, “Distributed path planning of a multi-robot system based on the neighborhood artificial potential field approach,” Simulation, 2019.

21. G. Li, Y. Tamura, A. Yamashita and H. Asama, "Effective improved artificial potential field-based regressin search method for autonomous mobile robot path planning," Internation Journal of Mechatronic and Automation, 2007.

22. Lee, D., Jeong, J., Kim, Y. H. and Park, J. B. An improved artificial potential field method with a new point of attractive force for a mobile robot. 2017 2nd International Conference on Robotics and Automation Engineering (ICRAE), Shanghai, 2017.

23. Xu, D., Zhang, X., Zhu, Z., Chen, C., and Yang, P. (2014). Behavior-Based Formation Control of Swarm Robots, 2014.

24. Guo, J, Gao, Y, Cui, G. Path planning of mobile robot based on improved potential field. Inf Technol J 2013.

25. J. Fink, A. Ribeiro, and V. Kumar, Robust control of mobility and communications in autonomous robot teams, IEEE Access1(2013).

26. Patle BK, Anish P, Jagadeesh A, Parhi DR Path planning in uncertain environment by using firefly algorithm, 2018

27. Hui Miao, “Robot Path Planning in Dynamic Environments Using a Simulated Annealing Based Approach”, M.Sc. Thesis, Queensland University of Technology, March 2009.

28. Ioannis Arvanitakis, Anthony Tzes, "Trajectory optimization satisfying the robot's kinodynamic constraints for obstacle avoidance", Control & Automation, 2012.

29. A. Sgorbissa and R. Zaccaria, “Planning and obstacle avoidance in mobile robotics,” Rob. Auton. Syst, 2012.

30. Mohanty,P.K.;Parhi,D.R.,“Controlling the motion of an autonomous mobile robot using various techniques: a review, ” Journal of Advance Mechanical Engineering, 2013.

31. Buniyamin, N.; Wan N. W. A. J.; Sariff, N.; Mohamad, Z., “A simple local path planning algorithm for autonomous mobile robots, ”International journal of systems applications, Engineering & development, 2011.

32. J. Han and Y. Seo, “Mobile robot path planning with surrounding point set and path improvement,” Applied Soft Computing, 2017.

33. www.jmprime.co.uk - Internet Source

APPENDIX: MATLAB CODE

```
%Dynamic environment with both static and dynamic obstacles
clc;
clear all;
close all;
%Defining the intial conditions
target=[46.5 20]; %Final point
pos=[2 20]; %Initial point
%Defining the position of Obstacle 5
origin_x5 = [40 41 41 40]; %// initial coordinates of vertices
origin_y5 = [5 5 15 15];
%Defining the position of Obstacle 6
origin_x6 = [40 41 41 40]; %// initial coordinates of vertices
origin_y6 = [25 25 35 35];
%Defining the position of Obstacle 7
origin_x7 = [30 31 31 30]; %// initial coordinates of vertices
origin_y7 = [5 5 15 15];
%Defining the position of Obstacle 8
origin_x8 = [30 31 31 30]; %// initial coordinates of vertices
origin_y8 = [25 25 35 35];
%Defining the position of Obstacle 9
origin_x9 = [25 26 26 25]; %// initial coordinates of vertices
origin_y9 = [5 5 15 15];
%Defining the position of Obstacle 10
origin_x10 = [25 26 26 25]; %// initial coordinates of vertices
origin_y10 = [25 25 35 35];
%Defining the position of Obstacle 11
origin_x11 = [10 11 11 10]; %// initial coordinates of vertices
origin_y11 = [5 5 15 15];
%Defining the position of Obstacle 12
origin_x12 = [10 11 11 10]; %// initial coordinates of vertices
origin_y12 = [25 25 35 35];
%Defining the position of Obstacle 13
origin_x13 = [45 46 46 45]; %// initial coordinates of vertices
origin_y13 = [5 5 15 15];
%Defining the position of Obstacle 14
origin_x14 = [45 46 46 45]; %// initial coordinates of vertices
origin_y14 = [25 25 35 35];
%Defining the position of Obstacle 15
origin_x15 = [5 6 6 5]; %// initial coordinates of vertices
origin_y15 = [5 5 15 15];
%Defining the position of Obstacle 16
origin_x16 = [5 6 6 5]; %// initial coordinates of vertices
origin_y16 = [25 25 35 35];
%Defining the position of Obstacle 17
origin_x17 = [47 55 55 47]; %// initial coordinates of vertices
```

```
origin_y17 = [18.75 18.75 18.9 18.9];
%Defining the position of Obstacle 18
origin_x18 = [47 58.5 58.5 47]; %// initial coordinates of vertices
origin_y18 = [22.5 22.5 22.75 22.75];
%Defining the position of Obstacle 3
origin_x3 = [35 37 37 35]; %// initial coordinates of vertices
origin_y3 = [28 28 29 29];
destination_x3 = origin_x3+0; %// final coordinates of vertices
destination_y3 = origin_y3-12;
%Defining the position obstacle 1
origin_x1 = [20 22 22 20]; %// initial coordinates of vertices
origin_y1 = [0 0 1 1];
destination_x1 = origin_x1 +0; %// final coordinates of vertices
destination_y1 = origin_y1 +36;
%Defining the position of Obstacle 2
origin_x2 = [15 17 17 15]; %// initial coordinates of vertices
origin_y2 = [0 0 1 1];
destination_x2 = origin_x2 +0; %// final coordinates of vertices
destination_y2 = origin_y2 +14;
n_steps = 100; %// number of "frames"
t_pause = 0.25; %// seconds between frames
%Properties of the obstacles
h5 = fill(origin_x5, origin_y5, 'b'); %// create object at initial
position
hold on
h6 = fill(origin_x6, origin_y6, 'b'); %// create object at initial
position
hold on
h7 = fill(origin_x7, origin_y7, 'b'); %// create object at initial
position
hold on
h8 = fill(origin_x8, origin_y8, 'b'); %// create object at initial
position
hold on
h9 = fill(origin_x9, origin_y9, 'b'); %// create object at initial
position
hold on
h10 = fill(origin_x10, origin_y10, 'b'); %// create object at
initial position
hold on
h11 = fill(origin_x11, origin_y11, 'b'); %// create object at
initial position
hold on
h12 = fill(origin_x12, origin_y12, 'b'); %// create object at
initial position
hold on
h13 = fill(origin_x13, origin_y13, 'b'); %// create object at
initial position
hold on
```

```
h14 = fill(origin_x14, origin_y14, 'b'); %// create object at
initial position
hold on
h15 = fill(origin_x15, origin_y15, 'b'); %// create object at
initial position
hold on
h16 = fill(origin_x16, origin_y16, 'b'); %// create object at
initial position
hold on
h17 = fill(origin_x17, origin_y17, 'b'); %// create object at
initial position
hold on
h18 = fill(origin_x18, origin_y18, 'b'); %// create object at
initial position
hold on
h3 = fill(origin_x3, origin_y3, 'r'); %// create object at initial
position
hold on
h1 = fill(origin_x1, origin_y1, 'r'); %// create object at initial
position
hold on
h2 = fill(origin_x2, origin_y2, 'r'); %// create object at initial
position
%Declaring the Area
A=rectangle('position',[0 0 60 40]);
axis equal %// same scale in both axes
axis manual %// prevent axes from auto-scaling
%Empty Matrices for Initialisation and storing the values
X1=[];
da1=[];
X2=[];
da2=[];
X3=[];
da3=[];
axis equal %// same scale in both axes
axis manual %// prevent axes from auto-scaling
%Initial Constants
AFC=0.008;
d=0.6;
RFC=[0.1 0.3 1.2];
 for t = linspace(0,1,n_steps)
    pre_pos=pos;
    calc=target-pos;
%Defining the initial position
    initial_position=rectangle('position',[pos(1) pos(2) 1
1],'curvature',[1 1],'facecolor','y');
% Defining the target and the position
    target_location=rectangle('position',[target(1) target(2) 2
2],'curvature',[1 1],'facecolor','g');
```

```
%To make Obs 3 to move along y axis
    x3 = destination_x3;  %// update x
    y3 = (1-t)*origin_y3 + t*destination_y3; %// update y
    set(h3, 'Vertices', [x3(:) y3(:)]); %// change object's position
    X3=[x3;y3]';
%To make Obs 1 to move along y axis
    x1 = destination_x1; %// update x
    y1 = (1-t)*origin_y1 + t*destination_y1; %// update y
    set(h1, 'Vertices', [x1(:) y1(:)]); %// change object's position
    X1=[x1;y1]';
%To make Obs 2 to move along y axis
    x2 = destination_x2;  %// update x
    y2 =(1-t)*origin_y2 + t*destination_y2; %// update y
    set(h2, 'Vertices', [x2(:) y2(:)]); %// change object's position
    X2=[x2;y2]';
%Conditions for Repulsive force
%Obs 1
for i1=1:length(X1)
     da1=[da1;(([X1(i1,1) X1(i1,2)])-pos)];
    Fa1=-(RFC(1)*(([X1(i1,1) X1(i1,2)])-pos)/norm(([X1(i1,1)
X1(i1,2)]-pos))^3);
        Fr1=Fa1;
end
%Obs 2
for i2=1:length(X2)
     da2=[da2;(([X2(i2,1) X2(i2,2)])-pos)];
    Fa2=-(RFC(2)*(([X2(i2,1) X2(i2,2)])-pos)/norm(([X2(i2,1)
X2(i2,2)]-pos))^3);
        Fr2=Fa2;
end
%Obs 3
for i3=1:length(X3)
     da3=[da3;(([X3(i3,1) X3(i3,2)])-pos)];
    Fa3=-(RFC(3)*(([X3(i3,1) X3(i3,2)])-pos)/norm(([X3(i3,1)
X3(i3,2)]-pos))^3);
        Fr3=Fa3;
end
%Attractive Force
FA=AFC*(calc);
Fr=Fr1+Fr2+Fr3;
Ft=FA+Fr;
angle = atan(Ft(2)/Ft(1));
pos(1)=pre_pos(1)+(d*cos(angle));
pos(2)=pre_pos(2)+(d*sin(angle));
if pos (1) >=target(1)-0.5
      pos=pre_pos;
      Fr=[0 0];
      Ft=[0 0];
end
```

```
pause(t_pause) %// a pause is needed to make movement slower
 end


%Defining the intial conditions
target=[58.5 20]; %Final point
pos=[46.5 20]; %Initial point
%Defining the position of Obstacle 4
origin_x4 = [56.5 57.5 57.5 56.5]; %// initial coordinates of
vertices
origin_y4 = [20 20 21 21];
destination_x4 = origin_x4+0; %// final coordinates of vertices
destination_y4 = origin_y4-3;
n_steps = 18; %// number of "frames"
t_pause = 0.25; %// seconds between frames
h4 = fill(origin_x4, origin_y4, 'y'); %// create object at initial
position
hold on
%Declaring the Area
A=rectangle('position',[0 0 60 40]);
axis equal %// same scale in both axes
axis manual %// prevent axes from auto-scaling
%Empty Matrices for Initialisation and storing the values
X4=[];
da4=[];
axis equal %// same scale in both axes
axis manual %// prevent axes from auto-scaling
%Initial Constants
AFC=0.008;
d=0.6;
 for t = linspace(0,1,n_steps)
    pre_pos=pos;
    calc=target-pos;
% Defining the initial position
    initial_position=rectangle('position',[pos(1) pos(2) 1
1],'curvature',[1 1],'facecolor','y');
% Defining the target and the position
    target_location=rectangle('position',[target(1) target(2) 2
2],'curvature',[1 1],'facecolor','g');
%To make Obs 4 to move along y axis
    x4 = destination_x4;  %// update x
    y4 = (1-t)*origin_y4 + t*destination_y4; %// update y
    set(h4, 'Vertices', [x4(:) y4(:)]); %// change object's position
    X4=[x4;y4]';
%Attractive Force
FA=AFC*(calc);
Ft=FA;
angle = atan(Ft(2)/Ft(1));
pos(1)=pre_pos(1)+(d*cos(angle));
```

```
pos(2)=pre_pos(2)+(d*sin(angle));
if pos (1) >=target(1)-0.5
      pos=pre_pos;
      Ft=[0 0];
end
pause(t_pause) %// a pause is needed to make movement slower
      end


% plotting a circle in 2d given the coordinates of its centre x and
% the radius r and desirable number n
% of points on the circle boundary
function circleBlack(x,y,r,n)
theta = linspace(0, 2*pi,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'k-')
h=fill(x1,y1,'k');
axis equal

% plotting a circle in 2d given the coordinates of its centre x and
% the radius r and desirable number n
% of points on the circle boundary
function circleBlue(x,y,r,n)
theta = linspace(0, 2*pi,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
h=fill(x1,y1,'b');
axis equal

function circleGreen(x,y,r,n)
theta = linspace(0, 2*pi,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
h=fill(x1,y1,'g');
axis equal

function circleRed(x,y,r,n)
theta = linspace(0, 2*pi,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
h=fill(x1,y1,'r');
```

```
axis equal

% plotting a circle in 2d given the coordinates of its centre x and

% the radius r and desirable number n
% of points on the circle boundary
function circleBluerot1(x,y,r,n)
close all %close all figures
%axis([-2*r,2*r,-2*r,2*r])
theta = linspace(pi/4, 2*pi +pi/4,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
%h=fill(x1,y1,'b');
axis equal % axis equal in order to have a correct scalling
axis([-2*r,2*r,-2*r,2*r]) % this is in order to be able to see the
plot inside the figure properly
%axis off

% plotting a circle in 2d given the coordinates of its centre x and

% the radius r and desirable number n
% of points on the circle boundary
function circleBluerot2(x,y,r,n)
close all %close all figures
theta = linspace(pi/4,pi/2 ,n); % to drow a segment of a circle
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
h=fill(x1,y1,'b');
axis equal

% plotting a circle in 2d given the coordinates of its centre x and

% the radius r and desirable number n
% of points on the circle boundary
function circleBluerotarc(x,y,r,n)
close all %close all figures
theta = linspace(pi/4, pi/2,n);
x1 = x + r*sin(theta);
y1 = y + r*cos(theta);
plot(x,y, 'o', 'MarkerFaceColor', 'w', 'Markersize', 6), hold on
plot(x1,y1, 'b-')
%h=fill(x1,y1,'b');
axis equal

% creating an environment one rectangular and one circular
```

```
% obstacles
% drawing rectangle without axes
clf % clear content of the Figure
close all % close all current figures
figure1 = figure('Color', 'w')
axis ([0 13 0 10])
rectangle('Position',[0,0,12,8],'Curvature', [0,0], ...
    'LineWidth', 2), hold on
rectangle ('Position',[8,4,1,3],'Curvature', [0,0], 'LineWidth', 4),
hold on
rectangle ('Position', [3, 2, 1,1],'Curvature', [1,1], 'LineWidth',
3, 'EdgeColor', 'b'), hold on
axis off

% creating an environment one rectangular and one circular
% obstacles
% drawing rectangle without axes
clf % clear content of the Figure
close all % close all current figures
figure1 = figure('Color', 'w')
axis ([0 130 0 100])
rectangle('Position',[0,0,120,80],'Curvature', [0,0], ...
    'LineWidth', 2), hold on
circleBlack(5,5,2,100)
circleRed(90,70,5,9)
circleGreen(40,20,5,100)
Kobs =5;
Kgoal =1;
Obs1 = [40,20];
Goal = [90,70];
Robot = [5,5];
%Create Repulsive and attractive potential field forces
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Matrix X x Y of  100by100
i=1:120;
j=1:80;
[I,J] = meshgrid(i,j);
Z = []; %empty matrix for z-axis plot

for i=1:120
    for j=1:80
        pos = [j i]; % scaled to fit 100by100 matrix
        z=0;    %accumulator

        %Obstacles with exponential curve while goal is by vector
        %z=z+exp(-norm(pos-Obs1)/Kobs)+exp(-norm(pos-Obs2)/K2)+exp(-
norm(pos-Obs3)/K3)+exp(-norm(pos-Obs4))/K4+Kg*(norm(Goal-pos));
```

```
    % The potential functions chosen are similar to the function
proposed in
    % the document 'apf.pdf'
    z=z+exp(-norm(pos-Obs1)/Kobs) +Kgoal*(norm(Goal-pos));
        Z(i,j) = z; %store in matrix

    end
end
t = norm(Robot-Goal);
count=0;
 while t>1 %check if the robot had reached the specified target;
stop when reach target

dx =[0 0];
 dx= dx + (Robot-Obs1)*exp((-norm(Robot-Obs1))/Kobs)+Kgoal*(Goal -
Robot)/(norm(Goal-pos));
  Robot=Robot+dx;

   %Calculate the value difference between the robot and the goal
   t = norm(Robot-Goal);
    count=count+1;
   %update display
   %calculate distance between robot and goal
   plot(Robot(1,1), Robot(1,2), 'K.','MarkerSize',20);
   title(sprintf('Iteration: %d',count));
   refresh;
  % pause(1)
   drawnow;
end
%{
rectangle ('Position',[8,4,1,3],'Curvature', [0,0], 'LineWidth', 4),
hold on
rectangle ('Position', [3, 2, 1,1],'Curvature', [1,1], 'LineWidth',
3, 'EdgeColor', 'b'), hold on
%}
%axis off

% This code visualise a potential function with one  goal and a
robot
clc                    %clear the Command Window
close all              %close all figures
% clear all            %this is an option, will clear the workspace
from all
                       % previously generated variables, If you want
to use
                       % this option, delete '%' before the statement
'clear
                       % all'
```

```
x=0:0.50:30;          % This creates an array x with values 0, 0.50,
1.00 ...
                      % 30.  The ';' at the end suppresses the
                      % screen display of the result. You can chose
different
                      % boundaries values as well as a different
increment.

y=0:0.50:30;              %Similar statement for the y array.
Robot = [2,2];          % Robot position
Goal = [18.2,18.2];     % Goal position

[xx,yy]=meshgrid(x,y);     %meshgrid creates an x-y grid with
coordinates of
                           % the points given by the vectors x
and y; the
                           % matrices xx and yy contain the x-
values at the
                           % grid points and the y-values,
respectively.


figure1 = figure('Color', 'w')
% drawing the environment
axis ([0 32 0 32])
rectangle('Position',[0,0,30,30],'Curvature', [0,0], ...
    'LineWidth', 2), hold on
circleBlack(Robot(1),Robot(2),0.5,100);  %drawing the roboot as a
black circle
circleBlue(Goal(1),Goal(2),2,8);         % drawing the goal as a
octagon

axis off
negd2=0.05*((xx-18.2).^2 +(yy-18.2).^2)    %Distance-squared from
the negative charge (goal)

neg=negd2;       % Magnitude of the goal  potential.
                 % The goal is 'sensed' at any point in the
environment
                 % Due to the nature of the paraboloid function, we
do not
                 % need to change the sign of negd2
zz=neg;     %global potential
figure(2) % Superpose the two potentials.
contour(zz,21)        % Make a contour plot of the potential with 21
contours.
                     % Using an odd number of contours lets us see
the
                    % zero-potential contour.
```

```
pause(1)         % pause before the appearance of the next figure
                 %pause(n) pauses execution for n seconds before
continuing,
                %where n can be any nonnegative real number
figure(2)
mesh(zz)
pause(1)
figure(3)   % A 3-D mesh plot of the potential. MESH produces
wireframe surfaces
             % that color only the lines connecting the defining
points.
surf(zz)
                % A surface plot. SURF displays both the connecting
lines
                % and the faces of the surface in color
pause(1)
figure(4)
[px,py]=gradient(zz,1, 1)%We use the gradient function to calculate
the velocity (force)
                      % field from the potential.  The  parameters 1
and 1 are the
                      % appropriate dx and dy values.
quiver(x,y,-px,-py,1,'r'), hold on          % This gives a vector
plot of the field; the pre-final
                      %parameter gives a scaling factor, 'r' -
colour
       %quiver(x(22),y(22), -px(22,22), -py(22,22), 20,'k') %
increasing the magnitude of
                                                          %the velocity
vector
                                                          %at a
specified
                                                          %location

% creating an environment with o one rectangular
% obstacle
clf; close all;
figure(1) %= figure('Color', 'w')
axis ([0 12 0 10])
rectangle('Position',[0,0,12,10],'Curvature', [0,0], 'LineWidth', 2,
'FaceColor','w'), hold on
% the statement above draws the rectangular environment(floor),
%the boundaries of which are indicated with a black rectangle, and
space
%inside is coloured white
rectangle ('Position',[5,3,2,2],'Curvature', [0,0], 'LineWidth', 4,
'FaceColor', 'g'), hold on
% the sttaement above draws a rectangular obstacle coloured with
green
```

```
x1 = 5; y1 =3; x2=5; y2 =5; x3 = 7; y3 = 5; x4 =7; y4 =3; % these
values are x-coordinates and y-coordinates of corner vertices of the
obstacle

line1y = 3;
line2y = 5;
line1x =5;
line2x =7;
% four lines provides the equations  of the bounding lines  of the
obstacle
%rectangle ('Position', [3, 2, 1,1],'Curvature', [1,1], 'LineWidth',
3, 'EdgeColor', 'b'), hold on
%plot(3.5,2.5,'ro', 'MarkerSize', 6, 'MarkerFaceColor', 'r')
axis off  % The rectangular environment will be displayed but
without axes
pause(1)
% creating the potential function of the
x = 0:0.2:12;
y = 0:0.2:10;
[xx,yy] = meshgrid(x,y); % creating the plaid (xy-grid)
Urep =[]; % 'Urep' is the repulsive potential, we have to determine
its values for each pair of (x,y) coordinates.
          %Initially Urep is an empty matrix, which at the end of
          %computations will be the same size as size(xx) (or
size(yy)).
          % The following statements define the values of Urep;
          % for each pair x,y) the shortest distance to the obstacle
is
          % found. The obstacle influences a robot only if it is
situated
          % sufficiently close to the obstacle.
          % In this case the distance q when the robot starts react
to the
          % obstacle, is 2 (units). The repulsive potential fuction
used in
          % this example is the FIRAS function, i.e. 'Force
Involving and Artificial Repulsion from the
          % Surface Function' introduced by Khatib, Urep = (1/2) *
krep*
          % *(1/dist(rob,obs) -1/q)^2. krep is chosen to be '1', and
          % dist(rob,obs) is the shortest distance between the robot
and the
          % obstacle.
          % As the repulsive potential function grows unboundlessly
when
          % the distance between the robot and the obstacle
decreases, the
          % distance dmin = 0.2, have been chosen, so for each
distances
```

```
          % that are smaller than dmin, the values of the potential
finction
          % will be the same and equal to Urep = (1/2) * krep*
          % *(1/dmin -1/q)^2. which in this case is 20.25/2. The
same values will be assigned to each
          % (xy)-pair that iis 'inside' the obstacle.

for i = 1:numel(y) % i refers ro the rows in Urep, which corresponds
to values along OY-axis
    for j = 1:numel(x) %refers to the columns in Urep, which
corresponds to values along OX-axis
        if yy(i,j) < y1;
          if ((xx(i,j) >= x1) && (xx(i,j) <= x4))
      dist1 = sqrt((yy(i,j)-line1y)^2);
   if (dist1 <= 2)
       if dist1 >=0.2
   Urep(i,j) = (1/2)*(1/dist1 - 1/2)^2;
       else
         Urep(i,j) = 20.25/2;
       end
   else
       Urep(i,j) =0;
   end
          elseif  xx(i,j) < x1

              dist2 = sqrt((xx(i,j)-x1)^2 + (yy(i,j)-y1)^2);

   if (dist2 <= 2)
       if dist2 >=0.2
   Urep(i,j) = (1/2)*(1/dist2 - 1/2)^2;
       else Urep(i,j) = 20.25/2;
       end
   else
       Urep(i,j) =0;
   end
          elseif xx(i,j) >= x4
              dist3 = sqrt((xx(i,j)-x4)^2 + (yy(i,j)-y4)^2);
              if (dist3 <= 2)
                  if dist3 >=0.2
   Urep(i,j) = (1/2)*(1/dist3 - 1/2)^2;
                  else Urep(i,j) = 20.25/2;
                  end
   else
       Urep(i,j) =0;
              end
          end
      elseif yy(i,j) > y3
     if (xx(i,j) >= x2) && (xx(i,j) <= x3)
       dist4 = sqrt((yy(i,j)-line2y)^2);
```

```
if (dist4 <= 2)
    if dist4 >=0.2
Urep(i,j) = (1/2)*(1/dist4 - 1/2)^2;
    else Urep(i,j) =20.25/2;
    end
else
    Urep(i,j) =0;
end
       elseif  xx(i,j) < x2

           dist5 = sqrt((xx(i,j)-x2)^2 + (yy(i,j)-y2)^2);
if (dist5 <= 2)
     if dist5 >=0.2
Urep(i,j) = (1/2)*(1/dist5 - 1/2)^2;
     else Urep(i,j) =20.25/2;
     end
else
    Urep(i,j) =0;
end
       elseif xx(i,j) >= x3
           dist6 = sqrt((xx(i,j)-x3)^2 + (yy(i,j)-y3)^2);
if (dist6 <= 2)
    if dist6 >=0.2
Urep(i,j) = (1/2)*(1/dist6 - 1/2)^2;
    else Urep(i,j) =20.25/2;
    end
else
    Urep(i,j)=0;
end
 end
else
   if xx(i,j)< x2
       dist7 = sqrt((xx(i,j)-line1x)^2);
       if dist7 <= 2
            if dist7 >=0.2
            Urep(i,j) = (1/2)*(1/dist7 - 1/2)^2;
            else Urep(i,j) =20.25/2;
            end
       else Urep(i,j) = 0;
       end
   elseif xx(i,j) > x3
       dist8 = sqrt((xx(i,j)-line2x)^2);
       if dist8 <= 2
            if dist8 >=0.2
            Urep(i,j) = (1/2)*(1/dist8 - 1/2)^2;
            else Urep(i,j) =20.25/2;
            end
       else Urep(i,j) = 0;
       end
```

```
      end
  end
  if ((xx(i,j) >= x1) && (xx(i,j) <= x4)&&(yy(i,j)>=y1)
&&(yy(i,j)<=y2))
      Urep(i,j) =20.25/2;
  end
    end
end
  %Urep = Urep1 + Urep2 +Urep3 +Urep4 + Urep5 +Urep6 +Urep7 +Urep8
+Urep9;
    zz=Urep;

  figure(2)
axis ([0 12 0 10 0 25])
surf(zz)
pause(1)
figure(3)
contour(zz,5)
pause(1)
figure(4)
[px,py]=gradient(zz,0.2,0.2);    %We use the gradient function to
calculate the velocity (force)
                      % field from the potential.  The  parameters
0.5 and 0.5 are the
                      % appropriate dx and dy values.
quiver(x,y,-px,-py,0.2, 'r'), hold on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Position of obstacles and goals
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
close all;
clf;
%Obstacle 1
K1=0.4;
Obs1=[6 7];

%Obstacle 2
K2=0.3;
Obs2=[6 11];

close all;
clf;
%Obstacle 3
K3=0.5;
Obs3=[9 17];

%Obstacle 4
K4=0.6;
Obs4=[10 13];
```

```
%Goal
Kg=0.11;
Goal=[13 18];

%Robot
Robot=[1 1];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Original Robot Environment
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
circleBlue(Robot(1)*10,Robot(2)*10,2,100); %robot 1
circleBlue(Goal(1)*10,Goal(2)*10,6,8);%goal right

circleBlue(Obs1(1)*10,Obs1(2)*10,5,100);%obstacle
circleBlue(Obs2(1)*10,Obs2(2)*10,4,100);%obstacle
circleBlue(Obs3(1)*10,Obs3(2)*10,6,100);%obstacle
circleBlue(Obs4(1)*10,Obs4(2)*10,9,100);%obstacle

figure(1)
title('Environment')
hold on
axis([0 100 0 100]);
grid on;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Create Repulsive and attractive potential field forces
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Matrix X x Y of  100by100
i=1:100;
j=1:100;
[I,J] = meshgrid(i,j);
Z = []; %empty matrix for z-axis plot

for i=1:100
    for j=1:100
        pos = [j/10 i/10]; % scaled to fit 100by100 matrix
        z=0;    %accumulator

        %Obstacles with exponential curve while goal is by vector
        z=z+exp(-norm(pos-Obs1)/K1)+exp(-norm(pos-Obs2)/K2)+exp(-
norm(pos-Obs3)/K3)+exp(-norm(pos-Obs4))/K4+Kg*(norm(Goal-pos));
    % The potential functions chosen are similar to the function
proposed in
    % the document 'apf.pdf'
        Z(i,j) = z; %store in matrix
```

```
    end
end
 t=(abs(Robot(1)-Goal(1))+abs(Robot(2)-Goal(2)));
count=0;
 while t>0.1 %check if the robot had reached the specified target;
stop when reach target

     dx=[0 0];  %accumulation matrix

    %Obstacles Accumulate vector sum  %Goal Accumulate
    dx= dx + (Robot-Obs1)*exp((-norm(Robot-Obs1))/K1)+ (Robot-
Obs2)*exp(-norm(Robot-Obs2)/K2)+ (Robot-Obs3)*exp((-norm(Robot-
Obs3))/K3)+ (Robot-Obs4)*exp((-norm(Robot-Obs4))/K4)+Kg*(Goal-
Robot)/norm(Goal-Robot);

    %Change in Robot Position over time (differenttiation)
    Robot=Robot+dx;

   %Calculate the value difference between the robot and the goal
   t=(abs(Robot(1)-Goal(1))+abs(Robot(2)-Goal(2)));
   count=count+1;
   %update display
   %calculate distance between robot and goal
   plot(Robot(1,1)*10, Robot(1,2)*10, 'K.','MarkerSize',20);
   title(sprintf('Iteration: %d',count));
   refresh;
   drawnow;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plot Figures
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(2)   %Contour figure
title('Contour')
hold on
contour(Z,22) %plot figure with 22 contours
pause(2)
hold on;

figure(3)   %combine figures
title('Potential Force Field on Contour')
hold on
contour(Z,22) %plot figure with 22 contours
hold on
[px,py]=gradient(Z); %calculate gradient
quiver(I,J,-px,-py,'r'), hold on % plot velocity vectors
pause(3)
```

```
figure(4)
title('Potential Function Landscape')
hold on; grid on
surfc(Z)        %plot surface and contour plot
view([100,35,30])
pause(1)
hold on;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Path planning for Robot
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Robot=[1 1];
%Plot current robot motion
 t=(abs(Robot(1)-Goal(1))+abs(Robot(2)-Goal(2)));
 % The above is one of possibilities to determine the distance
between the
 % robot and the goal, which is not strictly Euclidean
 %
 count=0;
 while t>0.1 %check if the robot had reached the specified target;
stop when reach target

     dx=[0 0];  %accumulation matrix

    %Obstacles Accumulate vector sum  %Goal Accumulate
    dx= dx + (Robot-Obs1)*exp((-norm(Robot-Obs1))/K1)+ (Robot-
Obs2)*exp(-norm(Robot-Obs2)/K2)+ (Robot-Obs3)*exp((-norm(Robot-
Obs3))/K3)+ (Robot-Obs4)*exp((-norm(Robot-Obs4))/K4)+Kg*(Goal-
Robot)/norm(Goal-Robot);

    %Change in Robot Position over time (differenttiation)
    Robot=Robot+dx;

   %Calculate the value difference between the robot and the goal
   t=(abs(Robot(1)-Goal(1))+abs(Robot(2)-Goal(2)));
   count=count+1;
   %update display
   %calculate distance between robot and goal
   plot3(Robot(1,1)*10, Robot(1,2)*10, Z(round(Robot(1,2)*10),
round(Robot(1,1)*10)),'K.','MarkerSize',20);
   title(sprintf('Iteration: %d',count));
   refresh;
   drawnow;
end
```