GoAvoid User’s Manual

Public domain version v1.0 for Microsoft Windows
Release date: 06/12/2016

Minnetoglu Okan, Conkur Erdinc Sahin

Pamukkale University Engineering Faculty Mechanical Engineering Department Kinikli Campus 20070 Denizli, Turkey

ominnetoglu@pau.edu.tr, sconkur@pau.edu.tr

GoAvoid® software is a path planner application for mobile robots that take advantage of GDI+ graphical user interface, which is a class-based API for C/C++ programmers in Microsoft Windows®. It is developed as an open-source framework for researchers who work on the mobile robot path planning research field. The software includes a simple working algorithm for obstacle avoidance in the workspace cluttered with both stationary and moving obstacles. The researchers can use the program as a starting point to develop their own algorithms.

Contents

Introduction 1

GoAvoid Fundamentals 2

1. Terminology 3

    1.1 Getting and Installing GoAvoid 4

2. User Interface 5

    3.1 Windows Functions 5

    3.2 GoAvoid Document Windows 6

    3.3 Function Selection and Feedback 7

    3.4 Menus 7

    3.5 Toolbar 7

    3.6 Mouse Buttons 7

3. Workspace Design 8

4. Environment Creation 10

    5.1 Setting Start-Goal Point Position 10

    4.2 Toolbar Menu Functions 11

    4.3 Draw Menu Functions 12

    4.4 Drawing Obstacle 12

    4.5 Obstacle Draw Modes 13

    4.6 Selecting Obstacles 13

5. View Tools 15

    5.1 Changing Workspace Color 15

    5.2 Panning on the Workspace 15

    5.3 Zooming on the Workspace 16

5.4 Showing Original View 16

5.5 Setting Default View 16

5.6 Showing Default View 17

5.7 Showing the Grid in the Workspace 17

5.8 Snapping to the Grid 18

5.9 Drawing Bitmap Field 18

5.10 Adding Random Obstacles 19

5.11 Deleting Selected Obstacle 20

6. Parameters Settings 21

6.1 Filtering path 21

6.2 Drawing Major Tick 21

6.3 Setting Distance between Path Points 22

6.4 Setting Reference Speed 22

6.5 Setting Safety Margin 22

6.6 Drawing Obstacle Path 23

7. Animation 24

7.1 Modes panel 24

7.2 Delay panel 24

7.3 Go panel 25

7.4 Information panel 25

8. Software Classes 27

8.1 Frameform Class 27

8.2 Obstacle Class 28

8.3 Obstacle Base Class 29

8.4 Potential Field Class 33

8.5 Mobile Robot Class 36

8.6 Mainform Class 36

List of Figures

Figure 1.1: Balloon tip for a control 2

Figure 1.2: Terms appear throughout the software and documentation 4

Figure 1.3: Software Setup Wizard 4

Figure 2.1: File and Windows menu 5

Figure 2.2: Panels in the software 6

Figure 2.3: Current mouse coordinates 7

Figure 2.4: Main menu 7

Figure 2.5: Toolbar menu 7

Figure 3.1: Workspace panel 8

Figure 3.2: Changing workspace dimension problem 8

Figure 3.3: Workspace borders 9

Figure 3.4: Current dimensions information 9

Figure 4.1: Setting start-goal point 10

Figure 4.2: Setting start-goal via toolbar menu 10

Figure 4.3: Setting start-goal via the context shortcut menu 10

Figure 4.4: Start point position problem 11

Figure 4.5: Toolbar menu functions 11

Figure 4.6: Draw menu functions 12

Figure 5.1: Changing workspace color 15

Figure 5.2: Panning via main menu 15

Figure 5.3: Panning via the context shortcut menu 15

Figure 5.4: Enabled panning mode 15

Figure 5.5: Going back to the original view 16

Figure 5.6: Setting default view 16

Figure 5.7: Showing default view 17

Figure 5.8: Showing grids in the workspace 17

Figure 5.9: Snapping to the grid in the workspace 18

Figure 5.10: Drawing bitmap field on the workspace 18

Figure 5.11: Drawing bitmap field on the workspace 19

Figure 5.12: Adding random obstacle via draw menu 19

Figure 5.13: Adding random obstacle via the context shortcut menu 20

Figure 5.14: Deleting selected obstacle 20

Figure 6.1: Set panel parameters 21

Figure 6.2: Overview of the path points 22

Figure 6.3: Drawing obstacle path points 23

Figure 7.1: Modes panel 24

Figure 7.2: Delay panel 24

Figure 7.3: Go panel 25

Figure 7.4: Information panel 25

Figure 8.1: The classes of the framework 27

Figure 8.2: A snapshot of the MDI interface 28

Figure 8.3: The relationship between an obstacle and the path 29

Figure 8.4: Choosing valid path point 30

Figure 8.5: Mobile robot speed calculation 31

Figure 8.6: 3D potential field as bitmap 34

Figure 8.7: Point p on the grid system 35

Figure 8.8: The largest descent 36

Figure 8.9: Tweak tool for each obstacle 37

Introduction

GoAvoid Software

GoAvoid® software is a path planner application for mobile robots that take advantage of GDI+ graphical user interface, which is a class-based API for C/C++ programmers in Microsoft Windows®. Using GDI+, users can draw rectangle shaped obstacles, set their speeds and directions. The workspace consists of the robot, start-goal points and obstacles. The user can change the position of the start-goal points by using mouse movements on the workspace. The obstacles can be drawn, moved and resized with mouse movements. The direction and speed of the obstacles may be set individually at any time using a simple tool appearing near the obstacle with a mouse click on the obstacle. The reference speed of the robot can also be set.

The software first calculates the path from the start point to goal point using the artificial potential field. Then calculates critical speeds of the mobile robot along the path, which is defined as the robot position in coincidence with each moving obstacle. Then safe robot speeds are determined based on these critical speeds. Finally, an animation is performed moving the robot along the path while avoding obstacles.

GoAvoid Fundamentals discuss the basic terminology and useful functions of the software. It guides the user how the software works via illustrations.

GoAvoid Fundamentals

GoAvoid Fundamentals introduce you to the following areas:

- **Terminology:** Lists the common GoAvoid terms used in the software.
- **User interface:** Describes the graphical user interface.
- **Workspace design:** Sets the dimensions of the workspace.
- **Environment creation:** Creates a motion scenario consisting of the start-goal and obstacles.
- **View tools:** Changes the workspace view.
- **Parameters setting:** Sets the parameters such as mobile robot reference speed and path filtering coefficient etc.
- **Animation:** Animates the mobile robot movement by taking into account the motion scenario.

Note: Balloon Tips are included in the interface to get informaton about all of the controllers by hovering the mouse on them. A balloon tip for the control is shown as in Figure 1.1. The user may disable Help->Show Balloon Tips option when he or she gets used to the software interface.

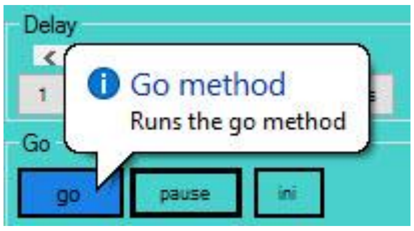


Figure 1.1: Balloon tip for a control

1. Terminology

The following terms appear throughout GoAvoid software and the documentation (Figure 1.2):

- **Workspace**: The area where the mobile robot can move.
- **Origin:** The (0, 0) coordinate of the workspace. Marked with a letter “O” and two green arrows showing “x” and “y” directions.
- **Start:** The start position of the mobile robot on the workspace.
- **Goal:** The final position that mobile robot reaches.
- **Stationary obstacle:** The obstacle whose speed value is set to zero.
- **Fixed size moving obstacle:** The fixed size obstacle whose speed is different from zero.

Note: The algorithm that is available in the software does not take into account the obstacle size of the moving obstacle. However, the user is free to change the algorithm in any way he or she wishes.

Path: The way that the mobile robot has to follow to avoid the obstacles and reach the goal. The path consists of points determined by means of the potential field and following each other almost uniformly spaced.

- **Intersected path point**: The path point intersected with the obstacle’s geometric center point.
- **Obstacle distance**: The distance from the moving obstacle to the path in the direction of the moving obstacle.
- **MajorTick**: The perpendicular line to the line made by the two respective path points, which is used to help the user see how the algorithm works.

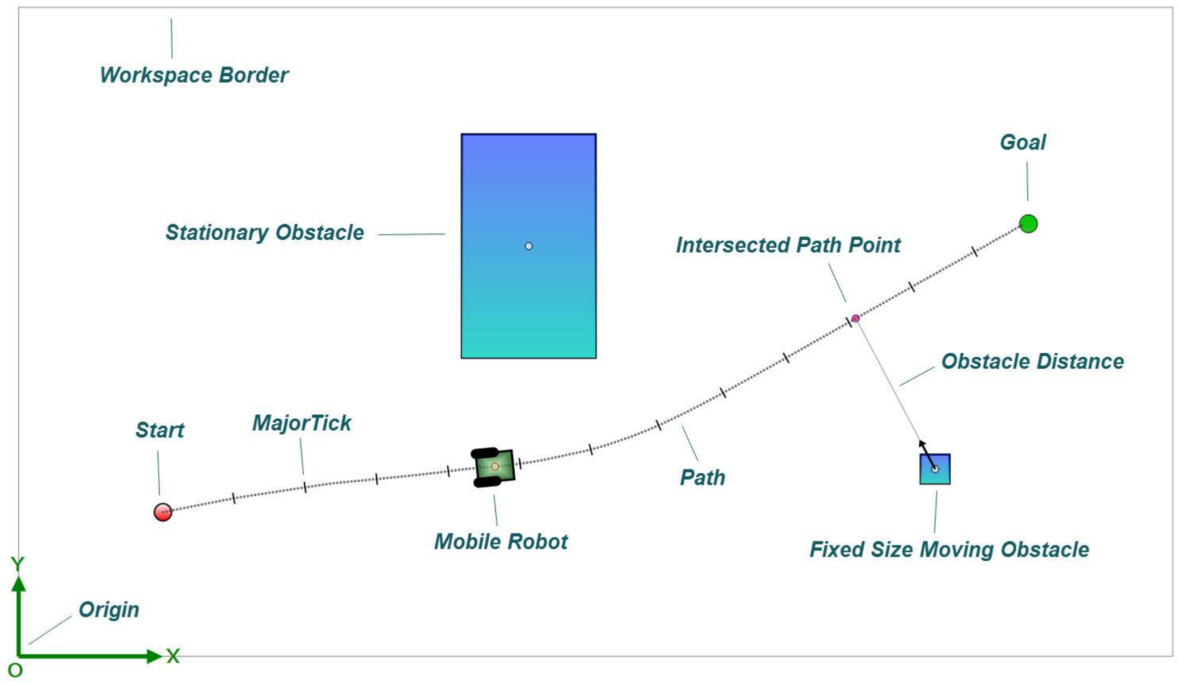
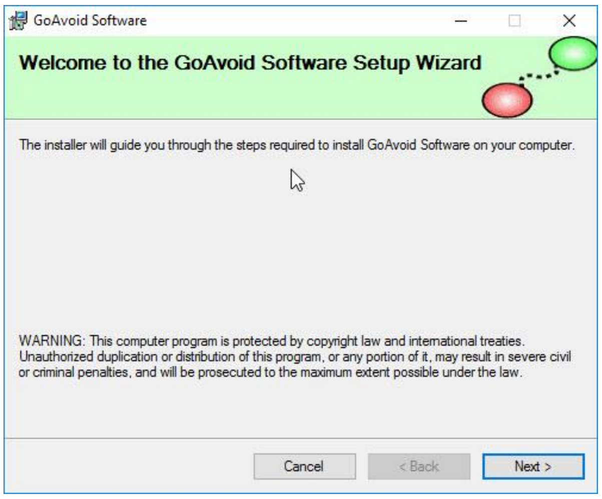


Figure 1.2: Terms appear throughout the software and documentation

1.1 Getting and Installing GoAvoid

GoAvoid software website is at https://github.com/ominnetoglu/GoAvoid. To run GoAvoid, it is required that (1) Windows XP SP3 operating system or above version; (2) Microsoft .NET Framework 4 or above to be installed. Microsoft .NET Framework 4 (as a pre-requirement file) and the software can be downloaded separately from https://github.com/ominnetoglu/GoAvoid/releases/tag/v1.0. The software can be installed by using software setup wizard, as shown in Figure 1.3.



Welcome to the GoAvoid Software Setup Wizard

The installer will guide you through the steps required to install GoAvoid Software on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 1.3: Software Setup Wizard

2. User Interface

GoAvoid software includes a variety of user interface tools and capabilities to help you create and edit a workspace efficiently. These tools and capabilities include the following:

- Windows functions
- GoAvoid document window
- Function selection and feedback

3.1 Windows Functions

GoAvoid includes familiar Windows functions such as resizing, open and save windows.

Some common Windows-related functions include:

- **Create new document:** Creates new blank document
- **Open new document:** Opens saved document
- **Save document as a different name:** Saves the document with a different name with GoAvoid file extension "*.goav".
- **Save document:** Overwrites the saved document.
- **Tile the opened documents:** Tiles all of the opened documents vertically, horizontally. The user can also cascade the documents.
- **Close the document:** Closes the selected document.

The user can reach these functions via File and Windows menus (Figure 2.1).

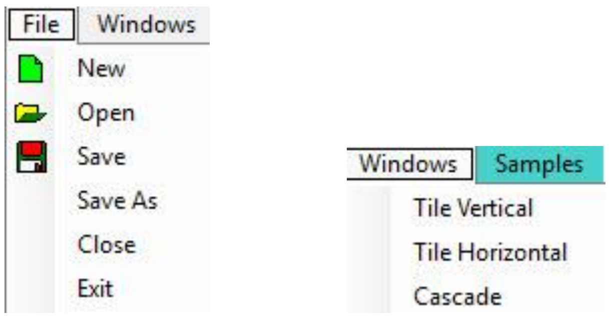


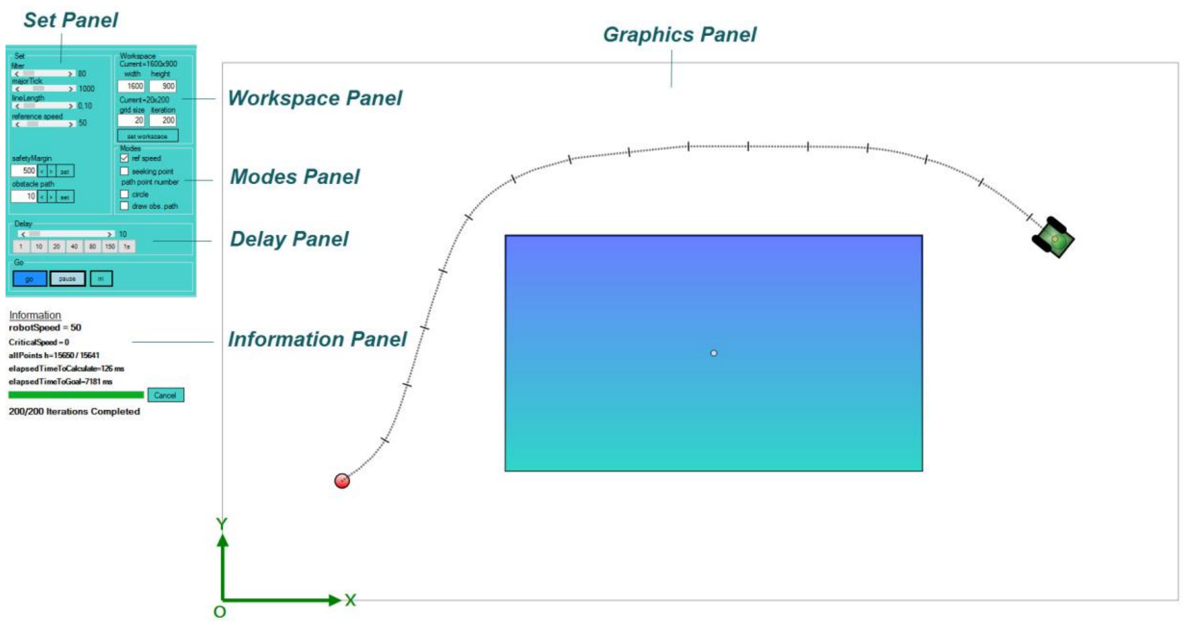
Figure 2.1: File and Windows menu

3.2 GoAvoid Document Windows

GoAvoid document Windows have various panels as shown in Figure 2.2. These panels are listed below;

- Set panel
- Workspace panel
- Modes panel
- Delay panel
- Go panel
- Information panel
- Graphics panel

The right panel is the graphics area where you create and manipulate the items on the workspace.

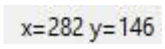


filter 80
majorTick 1000
lineLength 0.10
reference speed 50
safetyMargin 500
obstacle path 10
Current=1600x900
width 1600 height 900
Current=20x200
grid size 20 iteration 200
[✓] ref speed
[ ] seeking point
path point number
[ ] circle
[ ] draw obs. path
Delay 10

Information
robotSpeed = 50
CriticalSpeed = 0
allPoints h=15650 / 15641
elapsedTimeToCalculate=126 ms
elapsedTimeToGoal=7181 ms
200/200 Iterations Completed

Figure 2.2: Panels in the software

In addition, the user can see current mouse coordinates with respect to the origin at the right bottom of the screen (Figure 2.3).



x=282 y=146

Figure 2.3: Current mouse coordinates

3.3 Function Selection and Feedback

GoAvoid allows the user to perform tasks in different ways. It also provides feedback as the user performs a task such as setting safety margin parameter of the mobile robot or activating pan. Feedback includes hand and arrow pointers and selected buttons highlighted.

3.4 Menus

You can access all GoAvoid commands using menus. GoAvoid menus use Windows conventions, including sub-menus and check marks to indicate if an item is active and so on. You can also use the context shortcut menu to perform common tasks. Main menu is shown in Figure 2.4.

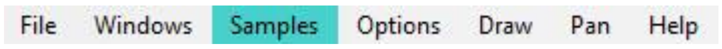


Figure 2.4: Main menu

3.5 Toolbar

You can access GoAvoid functions using its toolbar as well. Toolbar is consisted of individual icons that represent functions (Figure 2.5).

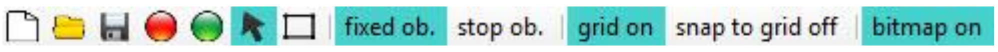


Figure 2.5: Toolbar menu

3.6 Mouse Buttons

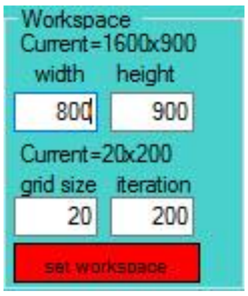
Mouse buttons operate in the following ways;

- **Left:** Selects menu items and entities in the graphics area and objects on the left panel.
- **Right:** Displays the context shortcut menu.
- **Middle:** Activates/deactivates the panning function on the graphic area.
- **Middle Double Click:** Sets the default view.

Note: Any view can be set as the default view by using "Set as Default View" function on the context shortcut menu. This issue will be explained in detail in the Section 5.

3. Workspace Design

GoAvoid software makes it possible to resize the workspace. The user can set the dimensions of the workspace by using the workspace panel. Workspace panel components are shown in Figure 3.1.


Workspace
Current=1600x900

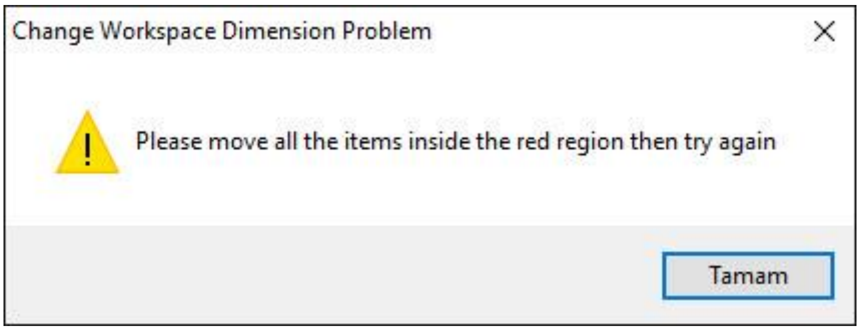
width	height
800	900

Current=20x200

grid size	iteration
20	200

Figure 3.1: Workspace panel

When the current dimension values on the panel change, the set button becomes red to warn the user that there is an attempt to change the workspace dimensions. After the set button is clicked, the software checks whether all of the items in the current workspace are in the user-defined workspace. If there is an item out of the user-defined workspace, a messagebox appears on the screen to inform the user that all of the items on the current workspace must be moved into the user-defined workspace (Figure 3.2).


Change Workspace Dimension Problem

Please move all the items inside the red region then try again

Figure 3.2: Changing workspace dimension problem

A temporary red user-defined workspace is drawn on the workspace to point out the user-defined workspace location after closing the messagebox as follows (Figure 3.3).

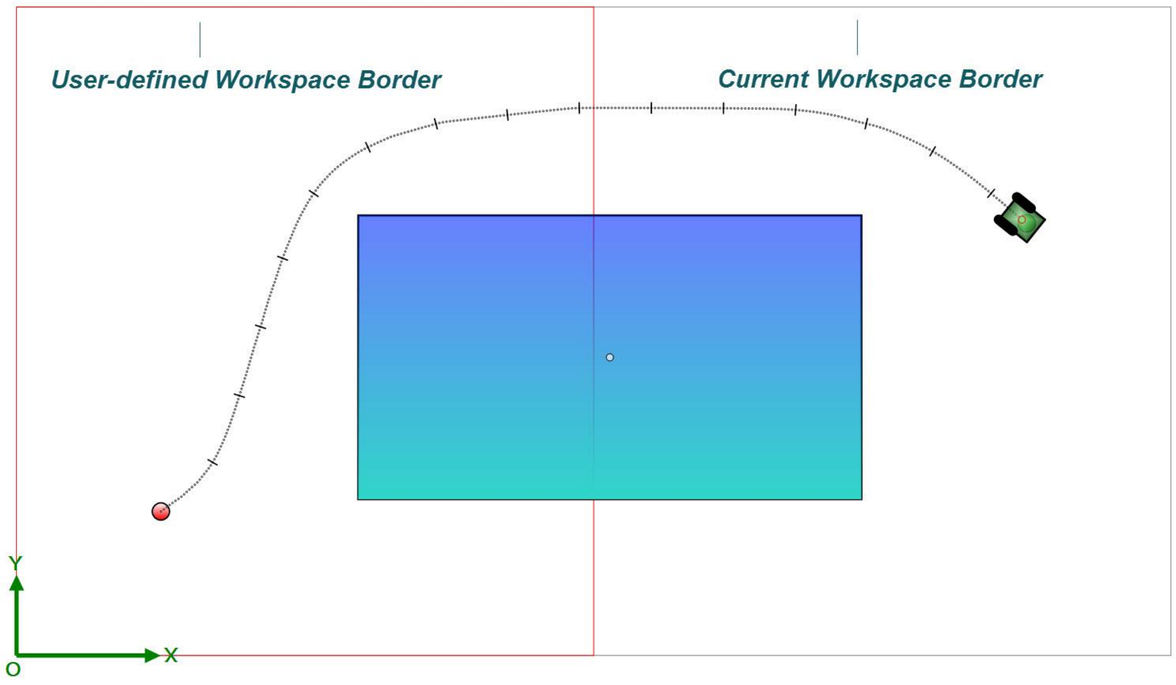


Figure 3.3: Workspace borders

If this button's color becomes white, the dimensions of the workspace are changed. In addition, the current dimension information will be updated as seen in Figure 3.4.

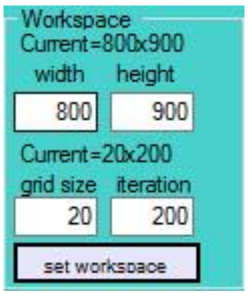


Figure 3.4: Current dimensions information

Once the dimensions of the workspace are changed, the software clears the temporary red workspace from the screen.

4. Environment Creation

The user can easily create a motion scenario consisting of the start-goal points and obstacles using the software. “Start” represents the start point of the motion and “Goal” represents the point where mobile robot aims to reach.

5.1 Setting Start-Goal Point Position

The user can invoke the setting process of the start or goal positions by means of the menus as shown in Figure 4.1, Figure 4.2 and Figure 4.3:

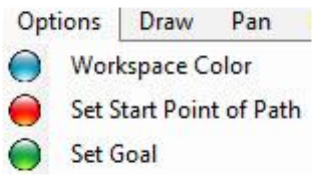


Figure 4.1: Setting start-goal point

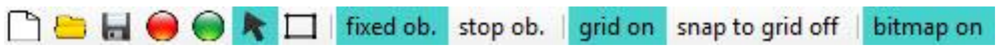


Figure 4.2: Setting start-goal via toolbar menu

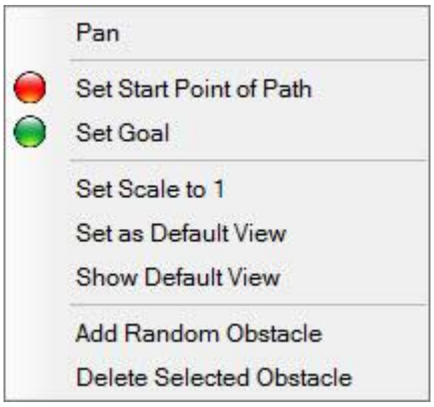
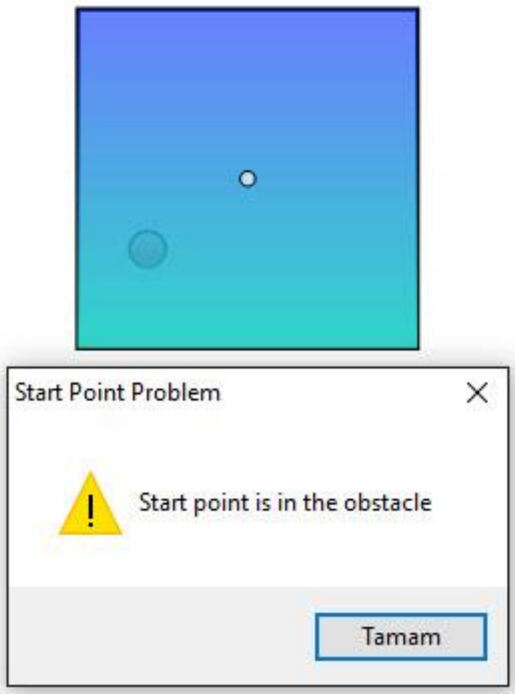


Figure 4.3: Setting start-goal via the context shortcut menu

After invoking the process, the user can change the start or goal position by pressing the left button while moving the mouse on the workspace.

Note: The software checks whether the start or goal point is inside an obstacle when the go button is clicked. If one of them is left inside the obstacle, a messagebox appears on the screen to inform the user as shown in Figure 4.4.


Start Point Problem

Start point is in the obstacle

Figure 4.4: Start point position problem

GoAvoid allows the user to draw rectangle shaped obstacles in the workspace and set their speeds and directions.

4.2 Toolbar Menu Functions

The user can rapidly reach frequently used functions using this menu. All of the functions on this menu are shown in Figure 4.5.

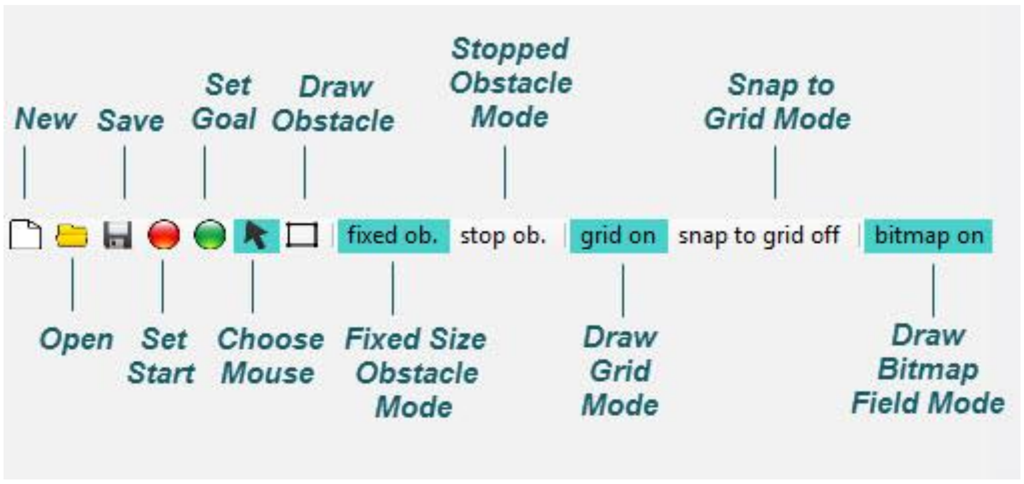


Figure 4.5: Toolbar menu functions

4.3 Draw Menu Functions

The draw menu functions are described below in Figure 4.6.

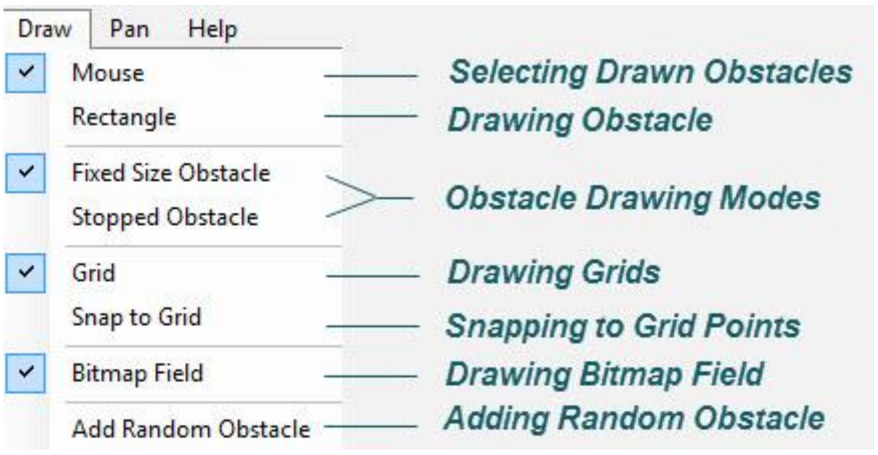


Figure 4.6: Draw menu functions

4.4 Drawing Obstacle

The user can initiate the drawing mode via followings:

- Associated toolbar icon

  Rectangle shaped toolbar icon can be used (Figure 4.5).

- Draw menu

  Rectangle option can be used in the Draw menu (Figure 4.6).

After initiating the drawing mode, the user can draw a rectangle shaped obstacle on the workspace with the mouse. The edge of the obstacle is specified by pressing the left button. Then, the mouse is dragged and the rectangle is completed by releasing the left button.

Note: The drawn obstacles must be in the region of the workspace. If the user attempts to draw an obstacle outside of the workspace, the software does not allow it and switches to selecting mode automatically.

4.5 Obstacle Draw Modes

The user can sets two obstacle draw modes.

- Fixed size obstacle

  This mode allows user that they can draw a fixed size obstacle by only clicking the left button.

- Stop Obstacle

  This mode determines whether the drawn obstacles must be stopped when it reaches the border of the workspace.

The user can start the obstacle drawing modes via followings:

- Associated toolbar icon

  Toolbar icons can be used captioned as “fixed ob. “and “stop ob.” (Figure 4.5).

- Draw menu

  Fixed size and stopped obstacle options can be used in the Draw menu (Figure 4.6).

4.6 Selecting Obstacles

The user can activate the selecting mode via followings:

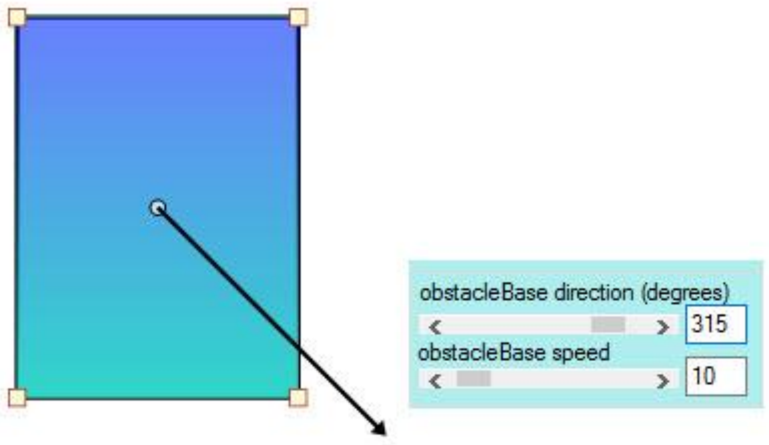
- Associated toolbar icon

  Arrow shaped tolbar icon can be used (Figure 4.5).

- Draw menu

  Mouse option can be used in the Draw menu (Figure 4.6).

After activating the selecting mode, the user can easily select an obstacle by clicking on them. Once the obstacle is selected, the placeholders and obstacle information panel appear on the screen as seen in Figure 4.7. The user can resize the selected obstacle by dragging the placeholders and set the motion direction and speed of the obstacle via controllers on the panel.


obstacleBase direction (degrees) 315

obstacleBase speed 10

Figure 4.7: Placeholders and obstacle information panel

5. View Tools

5.1 Changing Workspace Color

The user can easily change the workspace color by using options menu as in Figure 5.1.

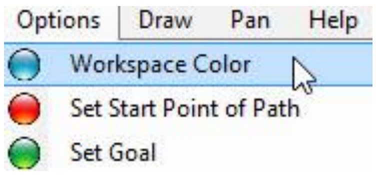


Figure 5.1: Changing workspace color

5.2 Panning on the Workspace

Panning mode can be enabled by means of the followings:

- Main menu (Figure 5.2)

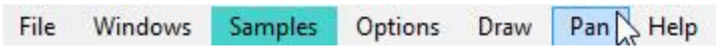


Figure 5.2: Panning via main menu

- Context shortcut menu (Figure 5.3)

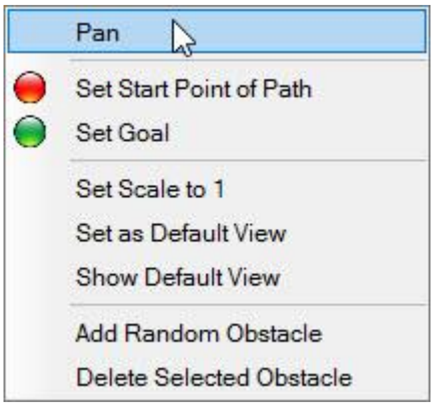


Figure 5.3: Panning via the context shortcut menu

- Mouse middle button clicking

The user can enable/disable pan mode by clicking the middle button of the mouse. The pan menu becomes turquoise when the panning mode is on as shown below in Figure 5.4.

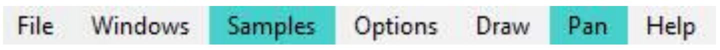


Figure 5.4: Enabled panning mode

5.3 Zooming on the Workspace

The user can zoom in or zoom out in the workspace via mouse wheel.

5.4 Showing Original View

After zooming in or out, the user can go back to the original view using the context shortcut menu as below in Figure 5.5.

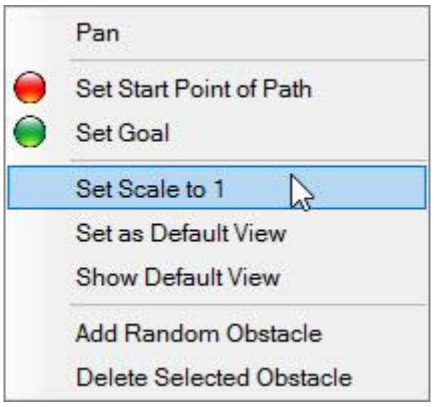


Figure 5.5: Going back to the original view

5.5 Setting Default View

For each document, different views can be set as a default view. This view is saved with the document files. The default menu can be set via context shortcut menu as in Figure 5.6.

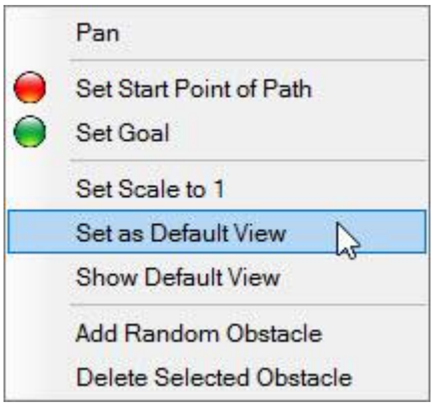


Figure 5.6: Setting default view

5.6 Showing Default View

The user can view the default view by using the context shortcut menu as below in Figure 5.7. If the default view is saved with the document, this view will be shown. Otherwise, the original view will be shown on the screen.

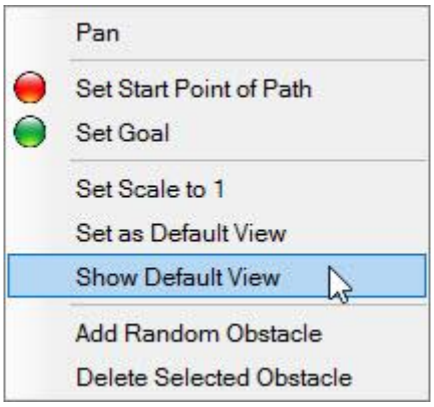


Figure 5.7: Showing default view

5.7 Showing the Grid in the Workspace

The grid is not shown when a new document is created since it slows down the processes in the workspace. It can be enabled by using draw menu in Figure 5.8.

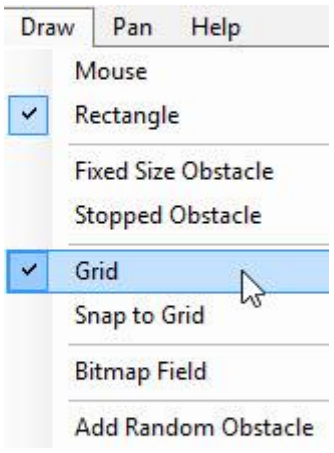


Figure 5.8: Showing grids in the workspace

5.8 Snapping to the Grid

When you draw, resize or move an obstacle, this property will align the nearest intersection of lines in the grid even if the grid is not visible. The user can enable this property by using draw menu as shown in Figure 5.9.

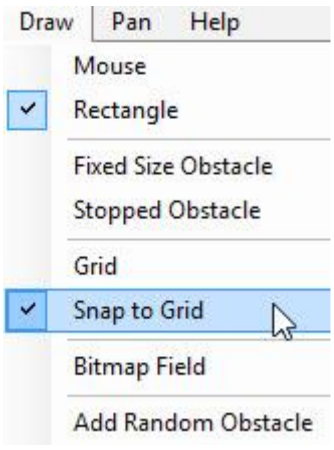


Figure 5.9: Snapping to the grid in the workspace

5.9 Drawing Bitmap Field

Similar to the grid, not only drawing the bitmap field slows down the animation but also it increases the software response time. It can be enabled via Main menu-> Draw menu as in Figure 5.10.

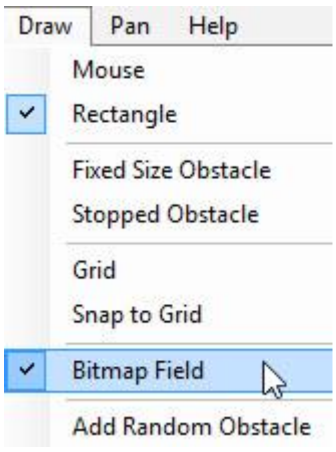


Figure 5.10: Drawing bitmap field on the workspace

This command draws the workspace calculated by the potential field as bitmap. Using this option the user can observe the slope from the start point to the goal. A workspace is shown below in Figure 5.11.

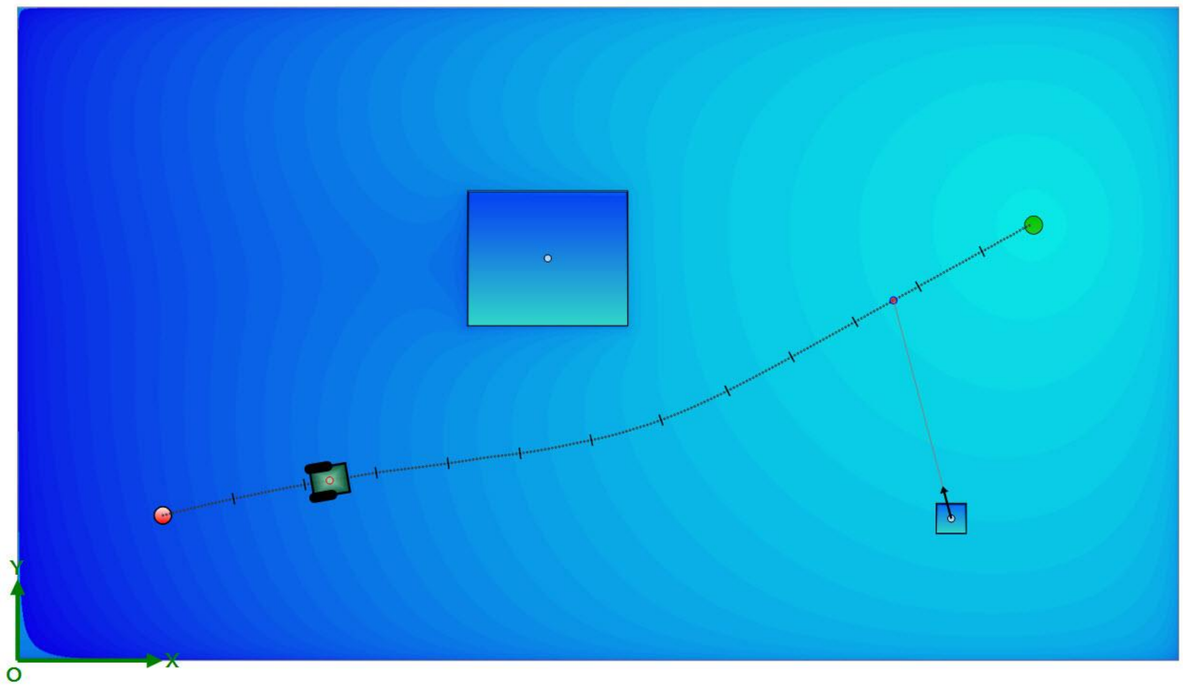


Figure 5.11: Drawing bitmap field on the workspace

5.10 Adding Random Obstacles

The user can add random obstacles when the animation does not run via followings:

- Draw menu (Figure 5.12)

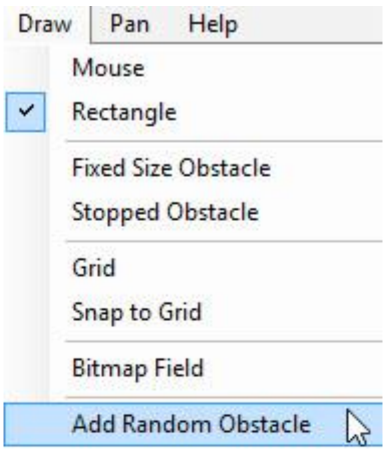


Figure 5.12: Adding random obstacle via draw menu

- Context shortcut menu (Figure 5.13)

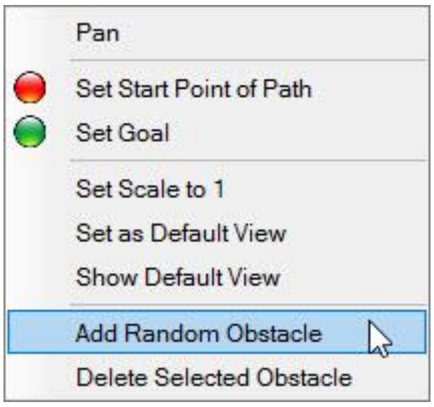


Figure 5.13: Adding random obstacle via the context shortcut menu

Note: This function adds non-fixed size random obstacles and sets its motion direction and speed automatically. The user can move, resize and set its paramaters.

5.11 Deleting Selected Obstacle

The user can delete the selected obstacle via the context shortcut menu as below in Figure 5.14.

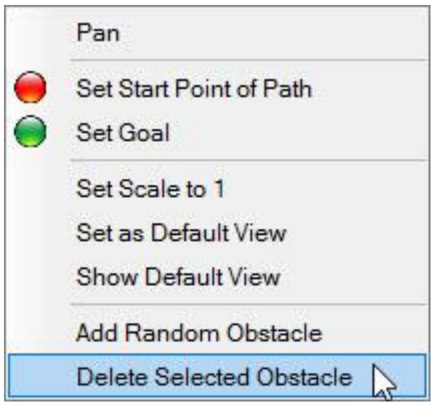


Figure 5.14: Deleting selected obstacle

Note: Selected obstacle can be deleted by using the “delete” button on the keyboard as well.

6. Parameters Settings

The user can use the set panel to assign values to the parameters in GoAvoid software (Figure 6.1). There are seven parameters on the set panel shown as follows.

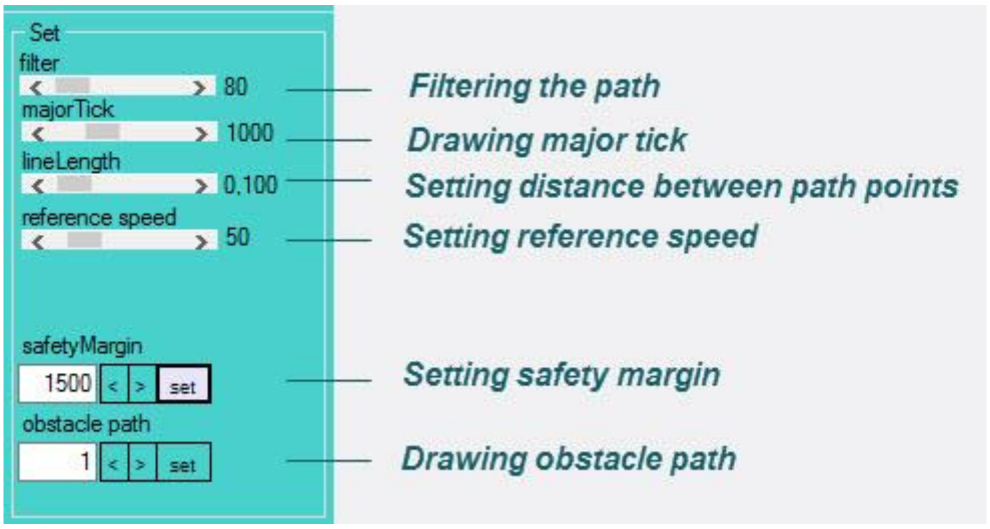


Figure 6.1: Set panel parameters

All of the parameters are described below respectively.

6.1 Filtering path

This parameter sets the number of points including in the windows averaging calculations that smooth the path calculated. It cannot be used while the animation is running. If its value is set too high, the path is diverted from its original location and it is no more represent the way through obstacles. It is recommended to set this parameter before the animation. The default value of this parameter is 80.

6.2 Drawing Major Tick

As mentioned in the section of terminology, the major tick is a mark drawn at a certain frequency to make it easy for the user to follow the robot and the path. The frequency of the majorTick can be set by the user. The default value of this parameter is 1000.

6.3 Setting Distance between Path Points

As mentioned before, the path generated by the potential field consists of successive points. This parameter sets the distance between two path points. It cannot be used while the animation is running. The default value of this parameter 0.1 (Figure 6.2).

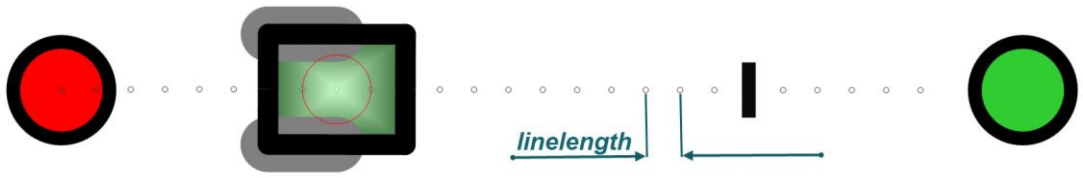


Figure 6.2: Overview of the path points

6.4 Setting Reference Speed

Reference speed of the mobile robot represents the desired speed value. This mode uses the reference speed value instead of the calculated speed considering the safety margin parameter value. The default value of this parameter is 50.

6.5 Setting Safety Margin

The safety margin parameter represents the difference between the robot distance on the path and the current obstacle position when the geometric center of the current obstacle intersects with the path points. It is clear that the path point representing the robot's position must be different from the obstacles intersection path point. Otherwise, the robot collides with the obstacle. The safety margin ensures that the robots's position is far away from the current obstacle position.

It cannot be used while the animation is running as mentioned above. If the safety margin value is equal to zero, the robot and obstacles' centers are coincided.

The aim of the algoritm included in the software is to show that how the software works. It is not intended to be a perfect obstacle-avoiding algorithm. The main limitations of the algorithm is listed below:

- The direction angle of an obstacle to the path must be windin the range +-45 degrees.
- The safety margin cannot be higher than the distance between two obstacles' intersection points.

Considering this limitations, the robot can avoid all of the obstacles. On the information panel the critical speed is the speed that safety margin value equals to zero, the "robotspeed" is the speed value according to the safety margin value set by the user. The default value of this parameter is zero.

6.6 Drawing Obstacle Path

The obstacle path parameter represents the frequency of drawing the obstacle path points while obstacles are moving if draw obstacle path mode is activated on the modes panel. If this parameter value is set to 5, it means that the software draws one point after 5 points. Using this property the user can calculate the distance that obstacle is gone via obstacle speed value. The default value of this parameter is 10.

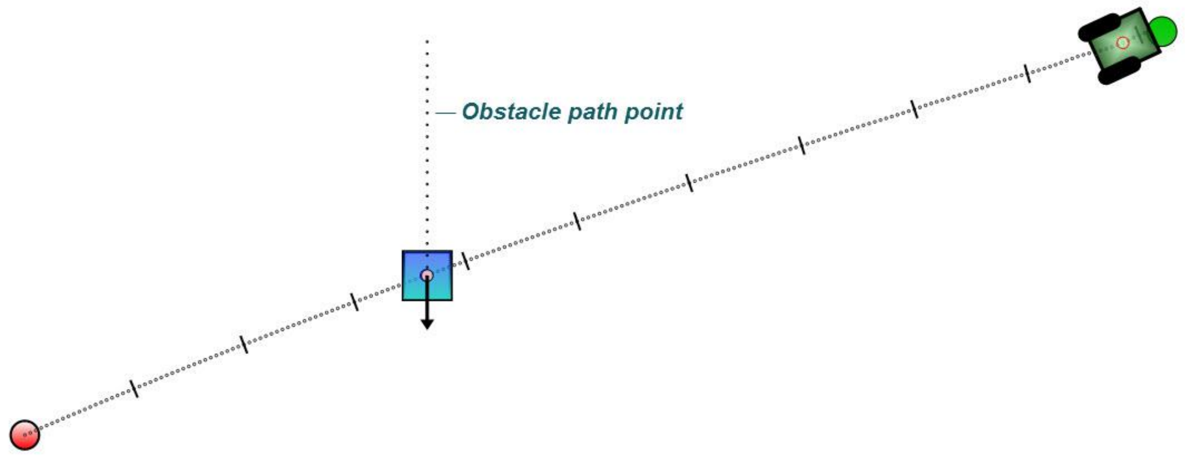


Figure 6.3: Drawing obstacle path points

7. Animation

7.1 Modes panel

There are four modes that the user can enable or disable as shown in Figure 7.1. These modes are described respectively below.

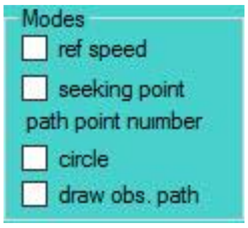


Figure 7.1: Modes panel

- **Reference speed mode:** This mode uses the reference speed value instead of the calculated speed considering the safety margin parameter value.
- **Seeking path point mode:** This mode allows the user to seek for the path point number by moving the left button pressed mouse on the path when it is on. The path point number will appear on the label below this mode’s checkbox per mouse movement.
- **Draw enclosing circle mode:** This mode draws red enclosing circles to all moving obstacles when it is on.
- **Draw obstacle path mode:** This mode draws moving obstacles path on the screen considering the obstacle path parameter as mentioned in the section 6.7 when it is on.

7.2 Delay panel

The user can slow down the animation to observe the robot motion details to check whether the algorithm proposed works as intended. The user can set the delay value not only by using hscrollbar controller but also by using the buttons as shown below (Figure 7.2). If the user clicks the button with the caption 150, it means the software will delay the animation 150 milliseconds.

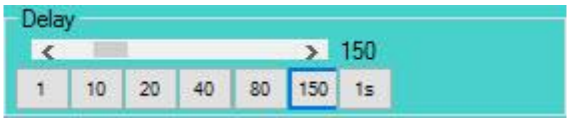


Figure 7.2: Delay panel

7.3 Go panel

There are three animation buttons including “go”, “pause” and “ini” on this panel (Figure 7.3). These buttons are described respectively as follows.

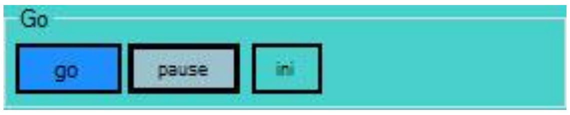
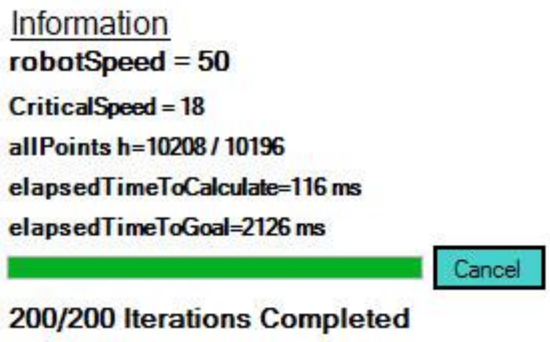


Figure 7.3: Go panel

- **Go button:** It runs the main method. In other words, it performs all of the calculations and starts the animation.
- **Pause button:** It pauses the animation. When paused, the caption of the button changes from “pause” to “continue”. The user can restart the animation by clicking this button again.
- **Ini button:** It sets the mobile robot and obstacle values to their initial positions.

7.4 Information panel

There is information about the robot on this panel as shown in the Figure 7.4. The information is described respectively below.



robotSpeed = 50
CriticalSpeed = 18
allPoints h=10208 / 10196
elapsedTimeToCalculate=116 ms
elapsedTimeToGoal=2126 ms
200/200 Iterations Completed

Figure 7.4: Information panel

- **Robot speed:** The current speed value of the mobile robot when it moves on the calculated path.
- **Critical speed:** Mobile robot speed value that coincides with the obstacles. If overall value is set to zero, the critical speed equals to the robot speed.
- **Path points:** It informs the user about progress of the movement.
- **Calculation time:** It is the artificial potential field calculation time for the created environment in milliseconds.

- **Elapsed time:** It is the time the robot takes from the start to the goal point in milliseconds.
- **Progressbar:** It informs the user about progress of the artificial potential field calculation.

8. Software Classes

The framework consists of classes related to mobile path planning and a test application. The classes perform their operations through their functions and variables. All of the available classes are shown in Figure 8.1. In this section, some of the classes will be discussed.

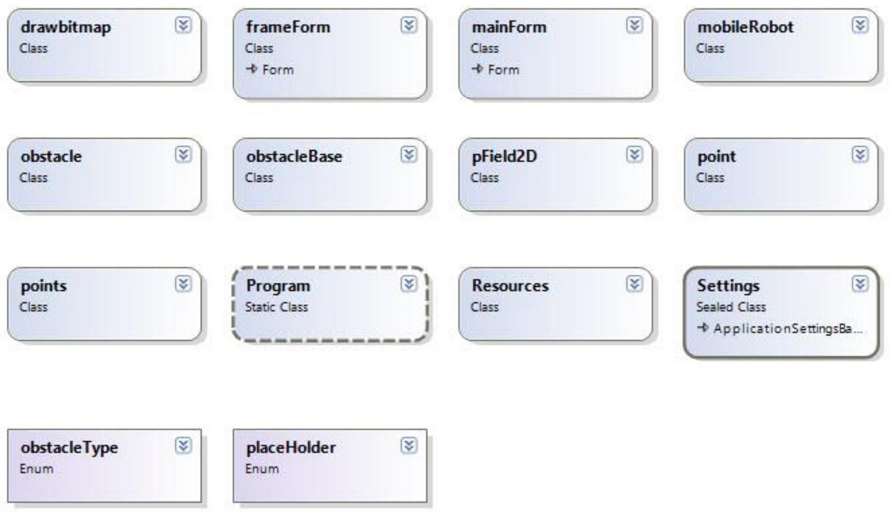
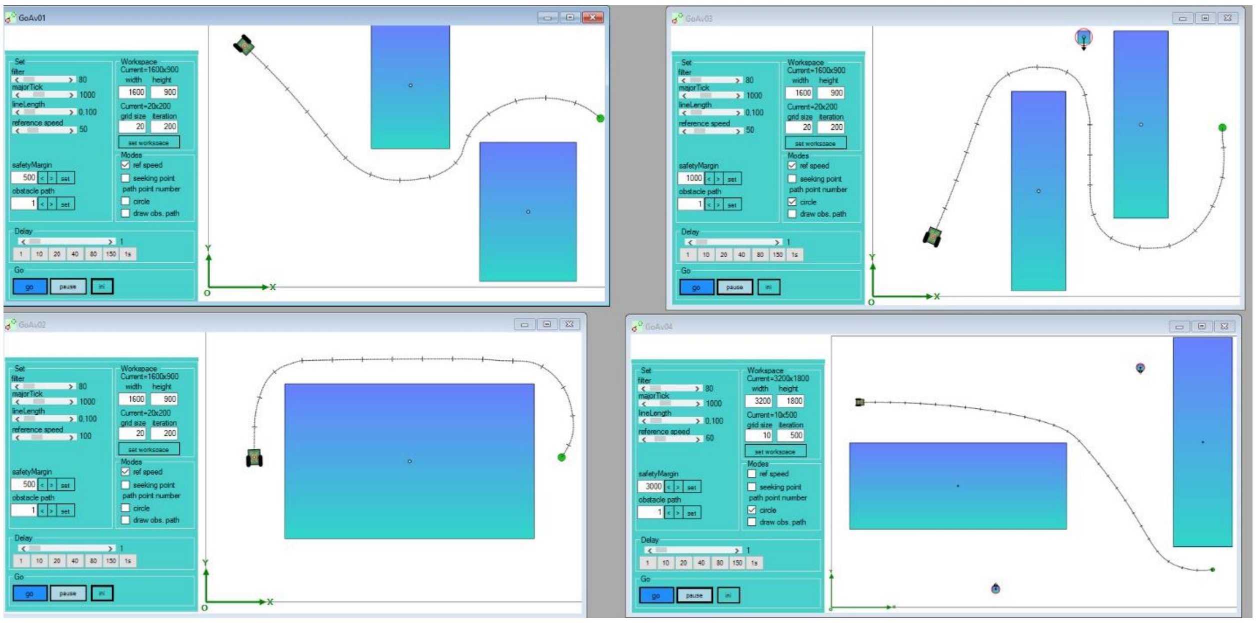


Figure 8.1: The classes of the framework

8.1 Frameform Class

The software has a MDI interface, which allows the user to create or open more than one document in a single opened software window. The user can tile vertically, horizontally and cascade all of the created or opened documents and can work on any document. The MDI interface is implemented using a class called “frameForm”. An instance of this class is the entry point to the program and manages all “mainForm” class instances. The user can select the desired document and modify its parameters using this class instance. A snapshot of the four different documents is shown in Figure 8.2.


GoAv01

Set
filter 80
majorTick 1000
lineLength 0.100
reference speed 50
safetyMargin 500 < > set
obstacle path 1 < > set

Workspace
Current=1600x900
width 1600 height 900
Current=20x200
grid size 20 iteration 200
set workspace

Modes
[✓] ref speed
[ ] seeking point
path point number
[ ] circle
[ ] drew obs. path

Delay 1
1 10 20 40 80 150 1s

Go
go pause ini

GoAv03

Set
filter 80
majorTick 1000
lineLength 0.100
reference speed 50
safetyMargin 1000 < > set
obstacle path 1 < > set

Workspace
Current=1600x900
width 1600 height 900
Current=20x200
grid size 20 iteration 200
set workspace

Modes
[✓] ref speed
[ ] seeking point
path point number
[✓] circle
[ ] draw obs. path

Delay 1
1 10 20 40 80 150 1s

Go
go pause ini

GoAv02

Set
filter 80
majorTick 1000
lineLength 0.100
reference speed 100
safetyMargin 500 < > set
obstacle path 1 < > set

Workspace
Current=1600x900
width 1600 height 900
Current=20x200
grid size 20 iteration 200
set workspace

Modes
[✓] ref speed
[ ] seeking point
path point number
[ ] circle
[ ] draw obs. path

Delay 1
1 10 20 40 80 150 1s

Go
go pause ini

GoAv04

Set
filter 80
majorTick 1000
lineLength 0.100
reference speed 60
safetyMargin 3000 < > set
obstacle path 1 < > set

Workspace
Current=3200x1800
width 3200 height 1800
Current=10x500
grid size 10 iteration 500
set workspace

Modes
[ ] ref speed
[ ] seeking point
path point number
[✓] circle
[ ] draw obs. path

Delay 1
1 10 20 40 80 150 1s

Go
go pause ini

Figure 8.2: A snapshot of the MDI interface

8.2 Obstacle Class

The user can draw rectangle shaped obstacles in any size via mouse. However, moving obstacles dimensions are fixed, which is considered as point-sized obstacles. A useful mode called “fixed size obstacle mode” is included in the user interface. This mode enables drawing fixed size obstacles. The user can enable/disable this mode by using the button captioned as “fixed ob.” on the toolbar menu as shown in Figure 4.5 above. “Obstacle class” contains the functions related to the obstacle drawing. This class also has functions to perceive the corner of the obstacle selected at the beginning of the drawing process to resize the obstacle. The relationship between an obstacle and the path is shown in Figure 8.3.

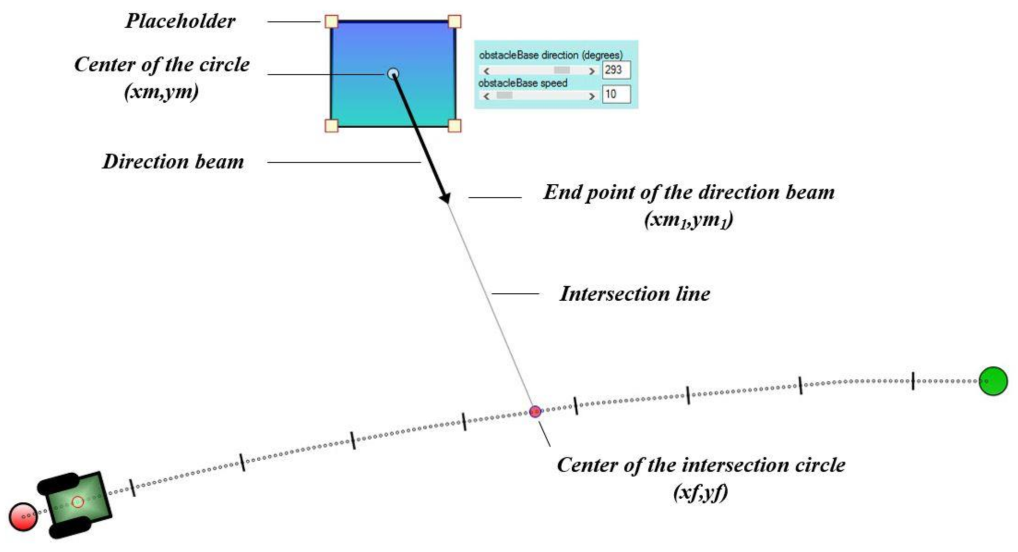

Placeholder
Center of the circle (xm,ym)
Direction beam
obstacleBase direction (degrees) 293
obstacleBase speed 10
End point of the direction beam (xm_1,ym_1)
Intersection line
Center of the intersection circle (xf,yf)

Figure 8.3: The relationship between an obstacle and the path

The obstacle center coordinates (xm, ym), the end of the beam coordinates (xm_1, ym_1), the width and the height of the obstacles are updated with respect to the new position when the user moves and/or resize them. The direction angle and speed of the obstacle are set by the user. The software calculates the new point coordinates where the obstacle intersects with the path and the time to reach this point.

8.3 Obstacle Base Class

The workspace is divided into equal sized square grids. In this class, grid values in free space are set to false while grid values in the space obstacles occupy are set to true. These values are used in calculating potential function, which will be described later in "Potential Field" class section.

In this class, the coordinates of the point of each moving obstacle that intersects with the path are obtained by using the parametric equation of the line that represents moving obstacle length and direction. The line length is arbitrarily determined, long enough to intersect with the path. Nevertheless, this line may intersect with both more than one point and stationary obstacles. In such situations, one of the proper path point must be chosen. The software saves all of the intersected points coordinates to an array and it chooses the proper one using the parameter t of the line parametric equation. The path point coordinates intersected with the smallest parameter t value are considered as a valid path point. Considering the motion scenario that is shown in Figure 8.4, the parametric equation of the line that belongs to a moving obstacle is represented by the following equations.

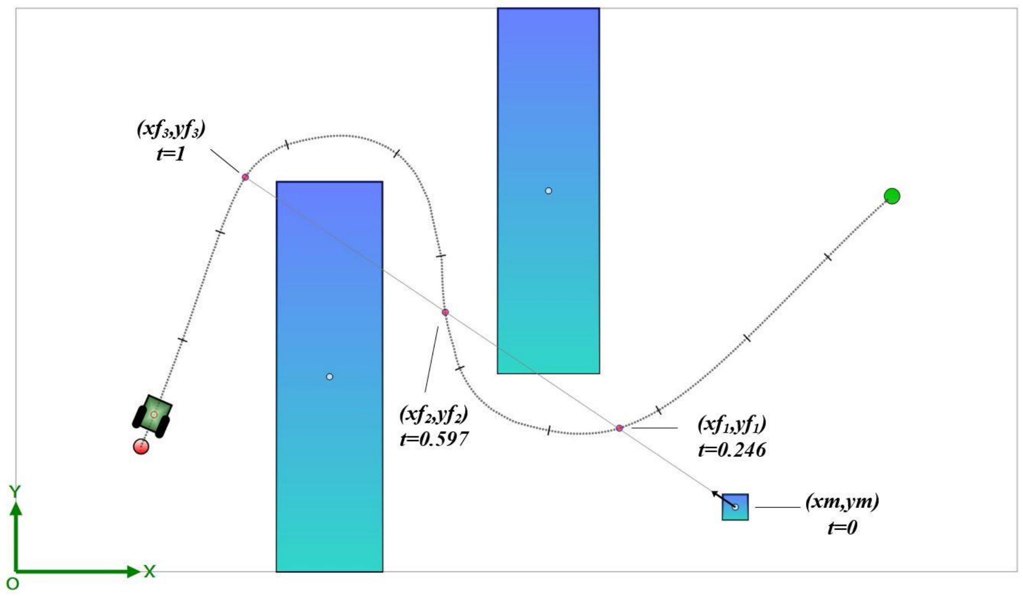


Figure 8.4: Choosing valid path point

$$xf_1 = xm + t(xf_3 - xm) \tag{1}$$

$$yf_1 = ym + t(yf_3 - ym) \tag{2}$$

$$xf_2 = xm + t(xf_3 - xm) \tag{3}$$

$$yf_2 = ym + t(yf_3 - ym) \tag{4}$$

There are three intersection path points. The valid point is the point with the shortest distance. In this case, the point (xf_1, yf_1).

The software does not use timers since they are slow in Windows. Instead, it uses “for” loop in a new thread. A single “for” loop corresponds to smallest execution time. For example, if the animation lasts for 200 loops and a single loop’s execution time is 0.1 seconds, the total execution time becomes 200*0.1=20 seconds. The higher the computer speed is, the faster “for” loop execution speed is.

The mobile robot motion is analyzed for all of the valid speed values whether or not all of the obstacles can be avoided with a single speed value. If this cannot be achieved, the software calculates the critical speed that is defined as the speed where the mobile robot meets the moving obstacle. If this speed is maintained, it is obvious that the robot will collide with the obstacle. If this speed is increased with a safety margin, the robot avoids the obstacle. Therefore, a simple obstacle-avoidance algorithm is achieved. A workspace with one moving obstacle is shown in Figure 8.5. Let us do it mathematically.

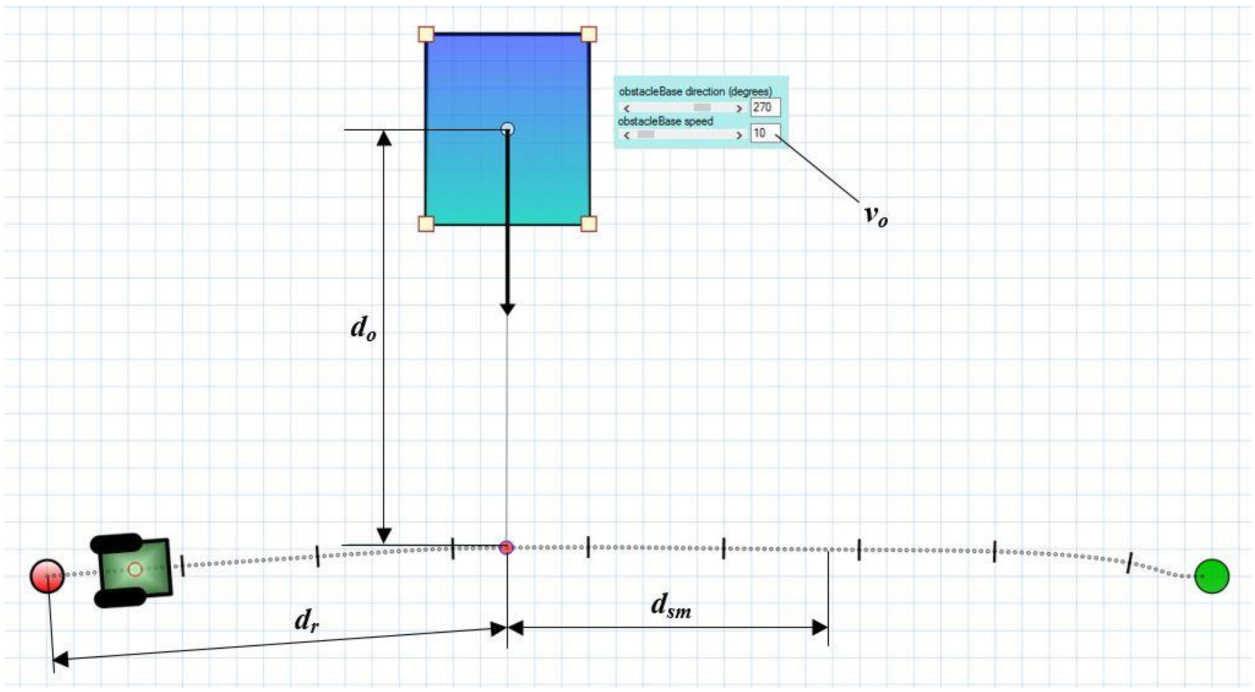

obstacleBase direction (degrees) 270
obstacleBase speed 10

Figure 8.5: Mobile robot speed calculation

In order to calculate the critical speed of the mobile robot, we first calculate the obstacle distance in steps d_{os} as follows.

$$d_{os} = \frac{d_o}{s} \tag{5}$$

where d_{os} is the obstacle distance in steps, d_o is the obstacle distance in mm and s is the step size in mm.

Then we determine how many loops are necessary for the obstacle to reach the intersection path point.

$$n = \frac{d_{os}}{v_o} \tag{6}$$

where n is the loop number and v_o is the obstacle speed in steps/loop.

After obtaining the loop number n, the robot distance in steps d_{rs} is calculated by considering the robot distance value d_r in mm.

$$d_{rs} = \frac{d_r}{s} \tag{7}$$

Since the loop number n must also satisfy the robot loop equation below to maintain the coincidence with the obstacle, the critical robot speed v_c in steps/loop which is unknown is easily calculated.

$$n = \frac{d_{rs}}{v_c} \tag{8}$$

Having the critical robot speed v_c, the safety margin speed value v_s in steps/loop is calculated considering the safety margin value d_{sm} in mm as follows.

$$d_{sms} = \frac{d_{sm}}{s} \tag{9}$$

$$n = \frac{d_{sms}}{v_s} \tag{10}$$

where d_{sms} is the safety margin distance in steps.

Finally, the robot speed v_r is calculated.

$$v_r = v_c + v_s \tag{11}$$

Let us give an example. Assume that the step size is equal to 2 mm . The workspace has one moving obstacle whose speed is set to 10 $steps / loop$. After clicking the go button, using the object locations in the workspace, obstacle distance d_o is calculated as 100 mm and the robot distance d_r is calculated as 200 mm. The safety margin parameter d_{sm} is also set to 30 mm. The critical robot speed v_c and the safe robot speed v_s are calculated as follows.

$$d_{os} = \frac{d_o}{s} = \frac{100}{2} = 50\, steps$$

$$n = \frac{d_{os}}{v_o} = \frac{50}{10} = 5$$

$$d_{rs} = \frac{d_r}{s} = \frac{200}{2} = 100\,steps$$

$$n = \frac{d_{rs}}{v_c} \Rightarrow 5 = \frac{100}{v_c} \Rightarrow v_c = 20\,steps / loop$$

$$d_{sms} = \frac{d_{sm}}{s} = \frac{30}{2} = 15\,steps$$

$$n = \frac{d_{sms}}{v_s} \Rightarrow 5 = \frac{15}{v_s} \Rightarrow v_s = 3\,steps / loop$$

$$v_r = v_c + v_s = 20 + 3 = 23\,steps / loop$$

8.4 Potential Field Class

This software includes all functions to calculate a potential field defined over a grid on “mainForm”. An artificial potential field function, U_m, is defined by the Laplace equation $\nabla^2 U = 0$ in a closed region, Z, which is continuous and equally connected. The boundary of Z is defined by δZ and there is no connection required. It contains the surfaces of all obstacles and the start-goal point. There are no local minima on U_m. However, the exponential decay of the field from any point leads to areas where the magnitude of the gradient on U, $|\nabla U|$, is very small while the range of $|\nabla U|$ may be very large. As will be seen in Figure 8.6, near the goal the field decays rapidly, but far from the goal there is only a small change in the field.

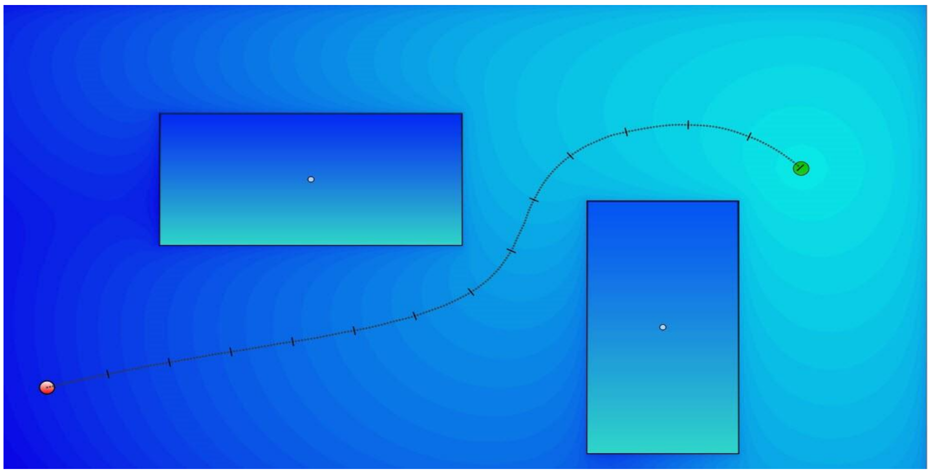


Figure 8.6: 3D potential field as bitmap

The Laplace equation in two dimensions is represented on equally spaced and connected grid under Dirichlet boundary conditions by the following partial difference equation (Eq. (12)),

$$U_{(i,j)} = \frac{U_{(i+1,j)} + U_{(i-1,j)} + U_{(i,j+1)} + U_{(i,j-1)}}{4} \tag{12}$$

where i = position on the grid in the x direction and j = position on the grid in the y direction. Position of a point p on the grid system is shown in Figure 8.7;

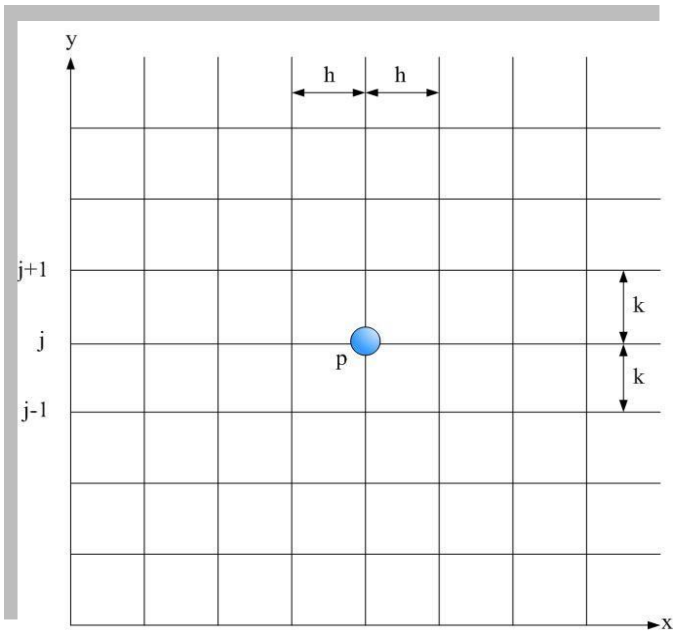


Figure 8.7: Point p on the grid system

The software detects the grid coordinates of the start and goal point that the user specifies in the workspace. The potential field over the grid is managed using two arrays. The first array is a bool type two-dimensional array called "savepoint", which is used to exclude obstacle grid points in grid value calculations. The second array is a double type two-dimensional array called "gridvalue", which keeps the numerical values of the grid points in double precision. In artificial potential field calculations, the goal grid value is set to a very small value, that is, -2^{124} and the boundary points are set to zero. Then using the potential field formula all grid values are calculated using Eq. (12). Then a number of iterations are performed over the field to have smooth grid point values.

After the implementation of the potential field in the workspace, using the grid values, the path-determining algorithm finds a smooth path consisting of points close enough to each other. The robot follows this path to reach the goal safely by avoiding obstacles. To draw a path, starting from the center of the mobile robot, the direction of the largest descent α is determined using the field values of surrounding grid points shown in Eq. (13) below;

$$\alpha = a\tan 2\left(\frac{U_{(i,j-1)} - U_{(i,j+1)}}{U_{(i-1,j)} - U_{(i+1,j)}}\right) \tag{13}$$

Then, using α, the x and y components of L_k are calculated using Eq. (14);

$$L_{kx} = \cos(\alpha)\left|L_k\right| \tag{14}$$

$$L_{ky} = \sin(\alpha)\left|L_k\right|$$

Vector L_k is drawn in the direction of the largest descent starting from N_k as shown in Figure 8.8. From the end point of this line, the direction of the largest descent is determined again. The current point is not necessarily located at exactly on a grid point. Interpolated values between grid points are calculated in such cases. Another vector is drawn in the new direction from N_{k+1}. The procedure is carried on until the goal point is reached. An array that keeps all of the path points coordinates is employed. Vector length $\left|L_k\right|$ drawn at each increment determines the closeness of the points to each other.

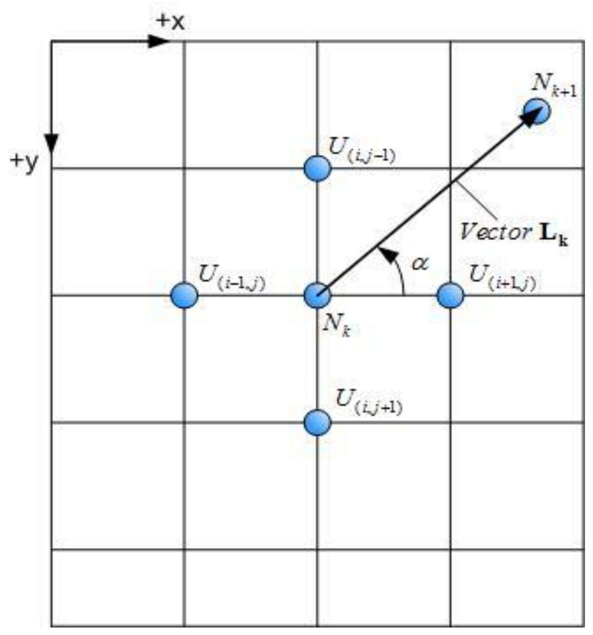


Figure 8.8: The largest descent

The path points determined as explained above may not be considered smooth enough due to local irregularities in the path. To have a smooth path, an operation called “windowing” is carried out. In this process, each path point’s value is recalculated by averaging certain point values that come before and after from that point.

8.5 Mobile Robot Class

It contains the three simple functions related to mobile robot drawing in the workspace. One function sets the mobile robot position on the calculated path. Another function determines the direction angle of mobile robot depending on each its movement of the robot between two path points on the path. The last function draws the mobile robot using values just mentioned whenever a redraw command is received.

8.6 Mainform Class

“Mainform” class implements other class instances and maintains program flow. Major functions and function groups will be mentioned in this section.

All paint operations in the client area of the user form are carried out in this class via class-based API for C/C++ programmers in Microsoft Windows called GDI+ (Graphical Drawing Interface) drawing interface. The panels mentioned in the previous section are included in this class. Parts of the interface are designed using “.NET Framework” objects having a graphical shape such as buttons, textboxes, scrollbars, etc.

Using GDI+, the user can draw rectangle shaped obstacles, set their speeds and directions. The workspace consists of the robot, start-goal points and obstacles. The user can set the position of the start and goal points with the mouse. The obstacles can be drawn, moved and resized with the mouse. Once the user clicks on an obstacle, placeholders appear on the corners of the obstacle. The user can move the obstacle to the desired place by dragging the mouse.

“Save” function group saves all the variables and arrays to the harddisk in a binary form with the file extension “*.goav”.

There is a tweak tool for the obstacle that allows the user to set the obstacle speed and direction as shown in Figure 8.9. The user can reach that tool by clicking the obstacle while animation is not running.

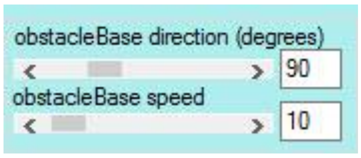

obstacleBase direction (degrees) 90
obstacleBase speed 10

Figure 8.9: Tweak tool for each obstacle

In the software, we use two different lists that keep the obstacle information such as coordinates, speed, directions etc. One of these lists keeps temporary values and the other one keeps the updated values. When the user changes the values in the tweak tool, the data is transferred from the temporary list to the updated list and it preserves the updated obstacle data as a new motion scenario.

“User input” function group includes mouse functions such as mouse down, mouse move, mouse wheel, etc. Pan and zoom features are implemented using mouse wheel function. The pan feature may be activated by clicking the related button on the menubar, right clicking the context menu or simply clicking the middle button of the mouse. The workspace can be zoomed in or out using the mouse wheel. The workspace view can be set to a default view using the right click context menu or simply double clicking the middle button of the mouse. Any workspace view may be set as default view for each document via right click context menu as well.

The software has error checking algorithms that prevent the user to perform invalid operations. It checks whether the calculated path is obstructed or not. If there is an obstruction, it warns the user with a messagebox. This checking algorithm mainly uses grid values. If all of the vertical, horizontal or both grid values are set to false or the start and/or goal point is drawn within an obstacle grid area, the software doesn’t attempt to calculate the potential field to avoid any exceptions in the software.

Index

A

algorithm, 1, 3, 22, 24, 31, 35, 37
angle, 22, 29, 36
animation, 2, 17, 18, 21, 22, 24, 25
API, 1, 36
application, 1, 27
array, 29, 35
artificial potential field, 1, 25
avoid, 3, 22

B

balloon tip, 2
beam, 29
bitmap, 18, 19
border, 13
buttons, 7, 24, 25

C

cascade, 5
check marks, 7
classes, 5, 27
commands, 7
components, 8
context menu, 37
context shortcut menu, 7, 10, 15, 16, 17, 20
continue, 25
coordinate, 3
coordinates, 4, 29, 35, 37
critical speed, 1, 22, 25, 30

D

default view, 7, 17, 20
delay panel, 6, 24
delay value, 24
dimensions, 4, 2, 8, 9, 28, 34
directions, 1, 11
disable, 15, 24
distance, 3, 22, 23, 30, 31, 32
document, 5, 6, 16, 17, 27, 37
draw, 5, 1, 11, 12, 13, 17, 18, 19, 23, 28, 35, 37
draw enclosing circle mode, 24
draw menu, 12, 13, 17, 18
draw obstacle path mode, 23, 24
draw rectangle shaped obstacles, 1
drawing mode, 12

E

enable, 2, 15, 18, 24
enclosing circles, 24
environment, 2, 10
execution, 30

F

feedback, 5, 7
field, 1, 5, 18, 19, 33, 35
file extension, 5
filtering coefficient, 2
fixed size moving obstacle, 3
fixed size obstacle, 3, 13
fixed size obstacle mode, 28
fixed size obstacles, 28
frameForm, 27
framework, 1, 5, 27
frequency, 21, 23
function, 7, 20, 33, 36, 37
functions, 4, 1, 5, 7, 11, 12, 27, 28, 33, 36, 37
fundamentals, 1, 2

G

GDI+, 1, 36, 37
go button, 10, 32
go panel, 6, 25
goal, 1, 2, 3, 10, 18

gradient, 33
graphics area, 6, 7
grid, 5, 6, 17, 18, 29, 33, 34, 35, 37
grid system, 6, 34, 35
grids, 5, 29

H

help, 5
horizontally, 5

i

icon, 7, 12, 13
implementation, 35
information panel, 6, 25
initial positions, 25
interface, 1, 5, 1, 2, 5, 27, 28, 36
intersect, 29
intersected path point, 3
iterations, 35

L

Laplace equation, 33, 34
length, 29, 35
line, 3
local minima, 33
loop, 30, 31, 32

M

magnitude, 33
mainform, 36
mainForm, 27, 33
mainform class, 36
major tick, 3, 21
MDI, 5, 27, 28
menu, 5, 7, 10, 11, 12, 13, 15, 18, 19
menu items, 7
messagebox, 8, 10
mobile, 1, 2, 3, 7, 10, 22, 25, 27, 30, 31, 35, 36
mobile robot, 1, 2, 3, 7, 10, 22, 24, 25, 26
mobile robot class, 36
mode, 5, 12, 13, 15, 22, 24, 28
modes panel, 6, 24
motion, 2, 10
motion scenario, 2, 10
mouse, 4, 1, 2, 6, 7, 10, 12, 15, 16, 24, 28, 37
mouse buttons, 7
mouse coordinates, 6, 7
mouse down, 37
mouse middle button, 15
mouse move, 37
mouse movements, 1
mouse wheel, 16, 37
move, 1, 8, 18, 20
moving obstacle, 3
moving obstacles, 28

O

obstacle, 1, 5, 6, 1, 3, 10, 12, 13, 14, 18, 20, 22, 23, 24, 25, 28, 29, 31, 32, 35, 37
obstacle base, 3
obstacle class, 28
obstacle distance, 3
obstacle information panel, 13, 14
obstacle-avoidance, 31
obstacles, 1, 2, 3, 10, 12, 13, 19, 21, 22, 23, 24, 25, 28, 29, 30, 33, 35, 37
open, 5
origin, 3, 6
original view, 16, 17
overall, 25

P

pan, 7, 15, 37
panels, 6
panning mode, 15
parameters, 5, 2, 21, 22, 23, 24, 27
parametric equation, 29
path, 1, 2, 3, 21, 22, 23, 25
path planning, 1, 27
path point number, 24
pause, 25
perpendicular, 3
placeholders, 13, 14
pointers, 7
position, 1, 3, 10, 11
potential field, 6, 3, 22, 26, 33, 34, 35, 37
potential function, 29
property, 18, 23

R

random obstacle, 19, 20
reach, 3, 5, 10, 11
rectangle shaped obstacle, 11, 12
reference, 1, 2
reference speed, 22
reference speed mode, 24
resize, 1, 5, 8, 13, 18, 20, 28, 29
response time, 18
robot speed, 25

S

safety margin, 7, 22, 24, 31, 32
save, 5, 37
screen, 6, 8, 9, 10, 13, 16, 17, 24
seeking path point mode, 24
select obstacle, 13
set, 1, 3, 7, 8, 11, 13, 16, 20, 21, 24, 25
set panel, 6, 21
setup, 4
snap to grids, 18
software, 1, 4, 1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 18, 21, 22, 23, 24, 27, 29, 30, 33, 35, 37
speed, 6, 1, 2, 22, 24, 25, 29, 30, 31, 32, 37
speeds, 1, 3, 11, 13, 20, 22, 23
start, 1, 2, 10, 13, 18
start-goal point, 4, 10, 33
start-goal points, 1
stationary obstacle, 3
step size, 31, 32
stopped obstacle, 13
sub-menus, 7

T

task, 7
terminology, 1, 21
terms, 2, 3
the dimensions of the workspace, 8
thread., 30
tile, 5, 27
time, 1, 25, 26, 29, 30
timers, 30
tool, 6, 1, 37
toolbar, 4, 7, 10, 12, 13, 18, 28
tools, 2, 5
tweak, 37

U

user-defined workspace, 8

V

value, 2, 3, 8, 21, 22, 25
vertically, 5
view, 2, 7, 16, 17, 20

W

website, 4
windowing, 36
windows functions, 5
wizard, 4
workspace, 1, 4, 5, 1, 2, 3, 5, 6, 8, 9, 10, 11, 12, 13, 15, 16, 17, 18, 19, 29, 31, 32, 35, 36, 37
workspace panel, 6, 8

Z

zoom, 16, 37
zoom in, 16, 37
zoom out, 16, 37