**iCompetition User Manual**



**Mor Baruch**

**Gal Lerman**

**Avi Sahar**

**Doron Shaharabani**

Table of Contents

[1 Introduction 3](#_Toc334296303)

[2 The Physical Robot 4](#_Toc334296304)

[2.1 iRobot create 4](#_Toc334296305)

[2.2 Charging the robot 4](#_Toc334296306)

[2.3 BAM 4](#_Toc334296307)

[2.4 Distance sensor 5](#_Toc334296308)

[2.4.1 EZ0 5](#_Toc334296309)

[2.4.2 The sensor’s switch 5](#_Toc334296310)

[2.4.3 How to connect the sensor 5](#_Toc334296311)

[2.5 Roomba 6](#_Toc334296312)

[3 Sensors Data 6](#_Toc334296313)

[4 GUI Overview 7](#_Toc334296314)

[4.1 The screen 8](#_Toc334296315)

[4.2 What can be shown on the screen 8](#_Toc334296316)

[4.3 PRM parameters 9](#_Toc334296317)

[4.4 Sensors parameters 9](#_Toc334296318)

[4.5 Game modes 9](#_Toc334296319)

[5 Creating a Scene file 10](#_Toc334296320)

[6 Running in Single Player Mode 11](#_Toc334296321)

[7 Running in Multi Player Mode (the competition) 12](#_Toc334296322)

# Introduction

Welcome to the new world of iCompetition where you have practical robots that help you test your brilliant algorithm (see the developer guide for information about how to add new motion plan algorithm).

iCompetition is a robots competition. The concept of the game is that your robot needs to reach a predefined target point before your competitors’ robot does.

This manual presumes that the reader understands all the concepts taught in Professor Dan Halperin’s [Planning High Quality Motion Paths for Robots](http://acg.cs.tau.ac.il/courses/workshop/fall-2012/high-quality-motion-planning-for-robots) Workshop.

The game includes a game field with a predefined starting point and a predefined target point for each robot. In addition, the field contains static obstacles that make the game more interesting. The field must be designed by the players before the game starts (for more explanation see the field creation chapter).

Each team receives the iCompetition kit that includes:

* An iRobot Create with a distance sensor.
* GUI software
* All software infrastructure of the game including the classes that the algorithm will be written in.
* This user guide.
* Developer guide

Both teams get a Roomba as a random dynamic obstacle in the game (The game is advisable for those who suffer from dust allergy).

# The Physical Robot

## iRobot create

iRobot Create is an affordable, preassembled mobile robot platform that provides an out-of-the-box opportunity for educators, students and developers to program behaviors, movements and add additional electronics.

## Charging the robot

The kit includes 2 chargers. Note that all of the iRobot Roomba chargers are also suitable for iRobot Create. When you connect the charger, the robot should be off. Once the charger is connected the robot’s indicator led should blink with an orange light and when the light turns green it indicates that the battery is fully charged. It is recommended to run the game when the batteries of the robots are fully charged.

## BAM



The BAM is a wireless serial cable replacement that uses the Serial Port Profile (SPP) to provide a means to send and receive serial packets to the iRobot Create mobile robot. The BAM allows the Create robot to be driven remotely with a PC hosted web server.

The BAM connects to the Create's cargo port to create a virtual serial port.

## Distance sensor

### EZ0



In addition to the built in sensors of the iRobot create, we connected the EZ0 sensor – a high performance ultrasonic rangefinder (for more information see <http://www.maxbotix.com/Ultrasonic_Sensors/MB1200.htm>).

### The sensor’s switch

We added a switch to the sensor. In order to save the batteries of the sensors it is important to make sure that the switch is off when you are not using the robot (even when the robot is off).

### How to connect the sensor



The sensor should sit on the top of the BAM straight up and it should be made sure that the sensor does not see the edge of the robot.

There are 3 wires connected to the sensor: GND and Vdd are connected from the batteries and the analog signal wire is connected between the BAM and the sensor.

Furthermore, you must connect to the BAM the same GND wire that is connected to the sensor (they must have the same GND reference).

## Roomba

The Roomba is a vacuum cleaner robot that moves randomly. Moreover, one can control its movements with a remote that is part of the kit. The remote can also control the iRobot create, thus if you want to control only the Roomba, the iRobot Create should work in full mode (this is the mode that the robot is set to when the competition begins).

# Sensors Data

The sensors that are read:

1. Angle - The angle in degrees that iRobot Create has turned since the angle was last requested is sent as a signed 16-bit value, high byte first. Counter-clockwise angles are positive and clockwise angles are negative.
2. Distance - The distance that Create has traveled in millimeters since the distance it was last requested is sent as a signed 16-bit value, high byte first.
3. Bumpers - The state[[1]](#footnote-1) of the left bumper and right bumper.
4. Battery charge - The current charge of Create’s battery in milliamp-hours (mAh). The charge value decreases as the battery is depleted during running and increases when the battery is charged.
5. Battery capacity - The estimated charge capacity of Create’s battery in milliamphours (mAh).
6. Wall Signal - The strength of the wall sensor’s signal is returned as an unsigned 16-bit value, high byte first. Note that after some experiments we decided that the information of this sensor is not informative enough for the game and thus we added the EZ0 distance sensor.
7. EZ0 distance sensor - The distance of Create from an obstacle in millimeters.



# GUI Overview



4.2

4.5

4.3

4.4

4.1

## The screen

After loading the scene file you will be able to see your robot positioned at the starting point with his starting azimuth, the target point and the obstacles. For example:



After figuring the path you will be able to see your solution (see chapter 4.2).

After the solution is computed, pressing the start button will show your virtual robot moving to the target point (for more information read chapters 6 and 7).

## What can be shown on the screen

There are six kinds of items that you can show or hide. In the flowing example you can see a screen that all options were chosen to be shown.



White

Red

Yellow

## PRM parameters



Above are all the parameters that are relevant to the PRM algorithm and are configurable.

## Sensors parameters



Distance – the distance that the robot passed in cm since the beginning of the game.

Angle- the angle that the robot turned in degrees since the beginning of the game.

Bumpers: left/right/front/none

Battery –the battery charge in percentage

EZ0 sensor – the distance from the sensor EZ0 that sits on the BAM to the closest obstacle (static or dynamic).

## Game modes

There are 3 modes:

The first is the Single player mode for a single player simulation.

The other two are the competition player and the competition referee (for more information read chapters 6 and 7).

# Creating a Scene file

Open a text file with the file extension scn.

This is the format of the file:

RotationSpeed 142

TranslationSpeed 200

BoundingBox (0,0) (409,850)

StartPosition1 (120,120) 1.57

EndPosition1 (375,500)

StartPosition2 (0,0) 1.57

EndPosition2 (800,800)

(0,83) (180,83) (180,0) (0,0)

(267,131) (322,131) (322,53) (258,53)

1.

2.

3.

4.

5.

6.

7.

8.

9.

Explanations:

1. The rotation speed of the wheels in millimeters/sec. This is the recommended rotation speed to minimize rotation errors.
2. The translation speed of the robot in millimeter/sec. This is the recommended translation speed to minimize translation errors.
3. Bounding Box – the field is rectangular and needs 2 points to be defined – the bottom left that is (0,0) and the top right that is according to the size of the field.
4. StartPosition1 – the starting point of the first robot and the direction of it. In the example the direction is 1.57 that is equal to$ \frac{π}{2}$.
5. EndPosition1 – the ending point of the first robot.
6. Same as section 4 for the second robot.
7. Same as section 5 for the second robot

8+9) You can add static obstacles. Each line contains an obstacle. An obstacle is a list of clockwise oriented points.

# Running in Single Player Mode

 

This mode is ideal for testing your robot’s algorithm.

In this mode only one robot is controlled.

If you want to simulate your algorithm, choose the *simulator* option and after loading your scene file, the scene will be displayed in the GUI.



Press on the solve button to run your static algorithm that creates a roadmap and finds a path to the target point. Press the start button when the algorithm finishes running to watch the virtual robot in the GUI performing the path.

If you want to try your algorithm with the physical robot, press on the *iRobot Create* button first and make sure that the COM is correct. Continue the same as explained with the simulator.

After pressing the start button, your robot will try to reach the target point according your algorithm.

# Running in Multi Player Mode (the competition)

In order to run a competition you need to run the application three times – twice for each competitor and the third is for the referee (it is recommended to run it from three different computers).

First you need to press the *Competition player* button in two of the applications (one for each competitor), then you press the *iRobot Create* button and make sure that the COM is correct. In the end you need to press the *Listen* button.



Make sure that each application is connected to a different robot.

After both applications are connected to the robots, in the third application you need to press the *competition referee* button, insert the IP and port for each competition player application and press the *start* button.



After pressing the start button the user will be asked to choose the scene file and finally the competition begins.

1. State means if it is pressed or not. [↑](#footnote-ref-1)