CoSpace Rescue 2015
Robotics Platform Training
Notes

RoboCup Junior Official Platform
- Virtual Environment
- Manual Control of Virtual Robot
- Introducing to Virtual Programming
- Your First Program
- Working with Ultrasonic Sensor
- Working with Color Sensor
- Object Detection by Two Methods
- Object Deposition
- Teleportation
- Using Variables
- Coordinate System (Secondary)
- Real Robot (Secondary)
Visual Environment

- Launch the CoSpace Robot Simulator (Rescue) and select the “Primary Age Group”

<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Q”, “Page Up”</td>
<td>Move upwards</td>
</tr>
<tr>
<td>“E”, “Page Down”</td>
<td>Move downwards</td>
</tr>
<tr>
<td>“Home”</td>
<td>Return to initial position</td>
</tr>
</tbody>
</table>

- Shift + Keys provided:
  - Move much faster

- You can also use mouse and dragging around
You can now use the dashboard (for manual control) to move the robots around in the field. Try controlling one, and then both virtual robots.
AI Programming Panel

• To program a robot you create statements which have 3 properties associated with them:
  ❖ Condition
  ❖ Action
  ❖ Priority – set by the order of the statements
There is a Deposit area

There is a Super object

There is a Red object

There is an obstacle in 20 cm

Priority Flow chart

Begin Scan

Deposit

Pick up

Pick up

Turn

Forward

N

Y

N

Y

N

Y

N

6
## Your First Program - Program your robot spin

<table>
<thead>
<tr>
<th>Statement type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition</td>
<td>No changes to the default setting as the action is independent to all sensors values.</td>
</tr>
<tr>
<td>Action</td>
<td>We will make the two wheels turning in opposite directions in order to let the robot to spin on a spot. E.g.</td>
</tr>
<tr>
<td>Duration</td>
<td>60 ms (millisecond)</td>
</tr>
</tbody>
</table>

This means the action will be executed continuously for 60 milliseconds before it scans all sensor readings again.

### Practice 1:
Program your robot moving straight forward

Save it, Build it, Load it and Run it
Working With Ultrasonic Sensor

• **A ROBOT STOPS WHEN IT IS CLOSER TO A WALL**
  • Program a robot to move forward and stop when it approaches a wall ($D \leq 20$ cm).

Practice 2:
Program your robot turns when it is closer to a wall
1. Launch the CoSpace Robot Simulator (Rescue) and select Primary platform.

2. Click on the “Robot” tab on the “Control Panel” and use the dashboard to move the robot over a RED object on the field.

3. Read the colour sensor value:

   ![Colour Sensors](https://example.com/colour-sensors.png)

This means that when the colour sensor senses the red object, the RGB reading is:

- Red: > 240
- Green: < 50
- Blue: < 50

4. Manually move the virtual robot over the GREEN, BLACK, BLUE, YELLOW, and ORANGE objects. Record the colour sensor RGB reading.

<table>
<thead>
<tr>
<th>Colour</th>
<th>RED</th>
<th>GREEN</th>
<th>BLACK</th>
<th>BLUE</th>
<th>YELLOW</th>
<th>ORANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You will need the values later, so keep these values safe!
Pick up object by Two methods

Program a robot to pick up a RED object using either left or right colour sensor by **Advanced Action**. Once the RED object is found, the robot should

- Stop for 3 seconds
- Flash LED
- Pick up the object

<table>
<thead>
<tr>
<th>Statement</th>
<th>RED object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>RED object</td>
</tr>
<tr>
<td>Type</td>
<td>Default</td>
</tr>
<tr>
<td>Duration</td>
<td>3000 ms</td>
</tr>
<tr>
<td>Condition</td>
<td>There are two methods can be used:</td>
</tr>
</tbody>
</table>

**Method 1**: use GUI to specify RGB range for the Left sensor, and the Right sensor RGB range can be defined in the advanced condition editor.

**Method 2**: use advanced condition editor.
Object Deposition

<table>
<thead>
<tr>
<th>If an action is executed for</th>
<th>It means that this statement is executed for</th>
</tr>
</thead>
<tbody>
<tr>
<td>60 milliseconds</td>
<td>1 duration</td>
</tr>
<tr>
<td>120 ms</td>
<td>2 durations</td>
</tr>
<tr>
<td>3 seconds or 3000 ms</td>
<td>50 duration</td>
</tr>
<tr>
<td>3600 ms</td>
<td>60 duration</td>
</tr>
</tbody>
</table>

A robot is considered to deposit the objects in the collection box (orange area) successfully, if -

- Both colour sensors must detect the orange zone at the same time.
- Deposit objects
- The robot must stop and turn on the LED for 3 seconds (with a steady light)
- The robot must be able to exit the deposit area automatically.
Object Deposition

In deposit box?

Set duration = 60 (3600 ms)

Robot stops, LED is on

Duration - 1
main loop scans in very 60 ms
(1 duration)

Duration < 10*

Robot moves away, LED is off

Search and Navigation

* When the remaining time is 10 duration (600 ms), the robot moves away.
Object Deposition

**Statement**

<table>
<thead>
<tr>
<th>Name</th>
<th>Deposit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Default</td>
</tr>
<tr>
<td>Duration</td>
<td>3600 ms</td>
</tr>
</tbody>
</table>

(3600ms/60ms) = 60 durations.

In the CoSpace simulator, the duration variable counts from 0. Therefore, 60 durations means 0, 1, 2, 3, ..., 59.

**Condition**

**Action**

WheelLeft: 0

WheelRight: 0

**Advanced Action**

```c
if (Duration<10)
    WheelLeft = -5;
    WheelRight = -3;
```

In the simulator, the duration is counted down by 1 for each scan. If the duration = 60, it means it counts down from 59 to 0. If duration = 10, it means there are still 11x60ms = 660ms remains.

**LED**

<table>
<thead>
<tr>
<th>LED 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

**Key Action**

Key Action: Deposit
A CoSpace Rescue game lasts 8 minutes, with two teams competing in one game. A game consists of WORLD_1 and WORLD_2. The robot can spend between 3-5 min in WORLD_1 and then should be teleported to the WORLD_2 for the remaining of the time to complete the mission.
Variable – LoadedObject

Can you change your code so that it allows the robot to collect 6 blocks and then does not stop to collect anymore until it has deposited the 6 it is carrying.

Can you change your code so that it allows the robot across Trap if there is no object on your car?
Variable – Coordinate system

If(PositionX < 5) || (PositionX > 355)
If(PositionY < 5) || (PositionY > 265)
The CoSpace server will send the coordinates (X,Y) of the SUPER or SUPER+ objects to the respective team upon SUPER or SUPER+ objects’ creation.

SuperObj_Num will be changed, if there is some SUPER or SUPER+ objects.
And you will get SuperObj_X, SuperObj_Y. But this value only can last 2 second, after that these values will be reset.

Think about it.
Variable – New Variables, Save data

Save Data:

\[ S_X = \text{SuperObj}_X \]
How to program a robot moving from point A to point B.

Step 1: Robot turns towards 270°

Step 2: Robot moves 150 units (while moving, make sure the robot should still facing 270°, or say the Y coordinates of the robot (PositionY) shouldn’t have much offset from 80.

Step 3: Robot turns towards 0°

Step 4: Robot moves 150 units (while moving, make sure the robot should still facing 0°, or say the X coordinates of the robot (PositionX) shouldn’t have much offset from 200.
Trap Avoidance

• Make your robot to avoid trap
Using Variables

Can you use a variable to ignore all green blocks once your robot has collected one green block?

Robot stops to collect first green block it finds

Robot carries on moving around the World

Robot ignores all other green blocks

Challenge

○ Can you change your code so that it allows the robot to collect 6 blocks and then does not stop to collect anymore until it has deposited the 6 it is carrying.

Program it

CoSpace comes with some predefined variables that you can use in your code. You can see these when you click on the Variable icon in the file menu at the top of the screen.

One of the useful variables is LoadedObjects, this keeps track of the number of blocks that your robot has collected. This is initially set to zero, incremented when a block is successfully collected and returns to zero when you deposit your blocks in the orange area.

For this challenge you need to write a little bit of C# programming. To do this click on the edit icon next to Advanced Condition. This will then open up the Code Editor.

You now want to tell your robot to stop and pick up a block only if LoadedObjects does not equal 1, that is if the robot has already collected a block, do not stop and collect anymore.

```
LoadedObjects != 1
```

“!=" means “does not equal” in C#.

Now save your code and your C# appears in the Advanced Condition box.

As before you now need to set the duration the robot waits to pick up the object, flash the set the Key Action to FindObject. This Action will only be executed when LoadedObject = 0.

Save it, Build it, Load it, Run it
How to program a robot moving from point A to point B.

Step 1: Robot turns towards 90°

Step 2: Robot moves 150 units (while moving, make sure the robot should still facing 90°, or say the Y coordinates of the robot (PositionY) shouldn’t have much offset from 80.

Step 3: Robot turns towards 0°

Step 4: Robot moves 150 units (while moving, make sure the robot should still facing 0°, or say the X coordinates of the robot (PositionX) shouldn’t have much offset from 200.