Thymio

Lishun Su - fc56375, Ayla Stehling - fc63327

May 23, 2024

1 Introduction

Thymio is a robot designed for a simple introduction to programming. The robot can drive and turn on command. It reacts to sensors on its sides and bottom, as well as to touch, buttons, and sound. These actions trigger responses such as driving, stopping, turning, and emitting light or sound. In the following, how to program the robot to follow your desired instructions will be explained.

Before learning how to program the robot, you need to understand how the robot works.
The robot has two motors, one on the left and one on the right, which allow it to drive, stop, and turn. The robot is equipped with buttons located on top, as well as five sensors in the front and two in the back. Additionally, the robot can be maneuvered by tapping the top or speaking into the microphone.

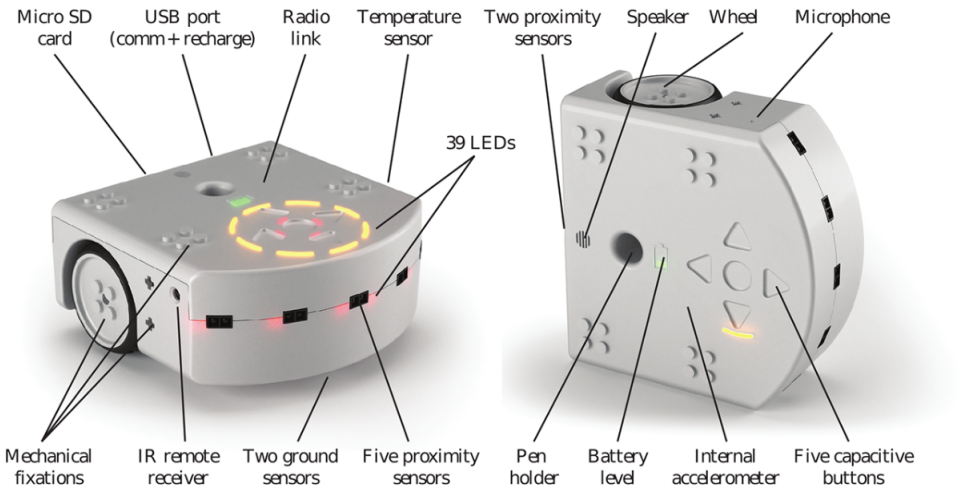

Figure 1: Thymio Robot [W J]

2 Getting Started with Thymio Programming

2.1 Code Sample

Here is a simple sample code for the robot sensing an object on the left and turning right:

```
Condition:
    -Sensor LeftSensor sensorPosition 1 detectObstacle
    OBJECT

Action:
    -MotorAction TurnRight motorLeft 200 motorRight -200

Event ObjectOnLeft
    conditions: LeftSensor actions: TurnRight
```

Start by declaring the conditions and actions, later used to build the event. The name of the event is "ObjectOnLeft".
The code will be further described in the following section.

2.2 Code Syntax

Everything marked with "" is the name of an event, action, or condition, which can be changed to any name you want but should be descriptive of the content. All numbers are not fixed and can be changed; ranges of numbers are mentioned in the description of each action or condition.

Declaring an event:

```
Event "ObjectOnLeft"
    conditions: "LeftSensor" actions: "TurnRight"
```

All events start with a name. Afterwards, an event can have conditions and actions. If an event has more than one condition, they are separated by 'and'. If it has more than one action, they are separated by a ','.
Before creating an event, we need to declare the conditions and actions.

Conditions:

```
Condition "ConditionName"
```

A condition starts with how the condition is called upon: Button, Sensor, Sound, or Tap.
The name of the condition should be descriptive of what it does. The following parts depend on which type of condition it is.

Button:

```
Button "ButtonBackward" BACKWARD
```

For the button, the name is followed by the ButtonType. ButtonType can be: BACKWARD, LEFT, RIGHT, CENTER, or FORWARD.

Sensor:

```
Sensor "LeftSensor" sensorPosition 1 detectObstacle OBSTACLE
```

In the sensor condition, the name is followed by 'sensorPosition' and then a number between 1 and 7 for the corresponding sensor. There are two more sensors on the bottom of the robot, sensors 8 (bottom right) and 9 (bottom left). Then the 'detectObstacle' is defined to decide wether a object has triggered the sensor or no object has been detected, either: OBSTACLE or NOOBSTACLE.

The corresponding sensors are:

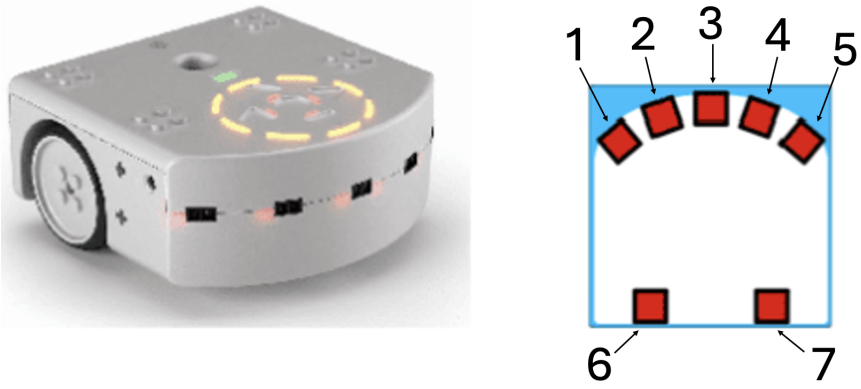

Figure 2: Sensors 1 to 5 in the front, sensors 6 and 7 in the back [Jiw14]
(not shown: sesnor 8 and 9 on the bottom)

Sound:

```
Sound "SoundName"
```

The sound condition only has a name of the condition.

Tap:

```
Tap "TapName"
```

Tap also only has a name the name of the condition.

Actions:
An Action is one of three types: MotorAction, SoundAction or LightAction.

MotorAction:

```
MotorAction "TurnRight" motorLeft 200 motorRight -200
```

Starting with 'MotorAction' and then followed by a name, here TurnRight. Then it is defined how the robot should turn, drive, or stop. The keywords are 'motorLeft' and 'motorRight', followed by a number between 500 and -500.
Some guide values:

- 300, 300 for driving forward
- 300, -300 for turning right
- -300, 300 for turning left
- 0, 0 for stopping

SoundAction:

```
SoundAction "Play" set: (1, MEDIUM, 1)
```

After the name of the action, a 'set:' is defined in parentheses. This begins with a note, which is a number between 1 and 5, followed by a duration of NONE, MEDIUM, or LONG, and a pos between 1 and 6.

LightAction:

```
LightAction "TurnOnLight" pos: TOP red: 32 green: 0 blue: 0
```

After the name, the position of the light is defined. After 'pos:' either TOP or BOT is chosen. Then, the color of the light is designated. The colors are based on RGB values, which means 'red:', 'blue:', and 'green:' are all values between 0 and 32. Whichever color is set to 32 is the color of the light. In this example, the color is red.

Actions and conditions are listed with '-' and follow this pattern:

```
Action:
    -MotorAction "TurnRight" motorLeft 200 motorRight -200

Condition:
    -Sensor "LeftSensor" sensorPos 0 distance CLOSE
```

Actions and conditions are listed with '-' and follow the same pattern as defined above.

Arithmetic functions +, -, * and / are applicable to all numbers. For example, you can add sensor positions 1 and 4 to get sensor position 5.

3 First Program: Follow a Black Line

Here is an example code where the Thymio robot follows a black line drawn on the ground.
Firstly, always implement a function to stop the robot!
Here we will implement a robot that will stop on tap.

```
Condition:
    -Tap TapSensor
Action:
    -MotorAction StopMotors motorLeft 0 motorRight 0

Event TapEvent
    conditions: TapSensor actions: StopMotors
```

Implement that the robot starts to drive forward if a black line is detected with a sensor on the bottom (8, 9). Detecting a black line with a sensor means detecting an OBJECT.

```
Condition:
    -Sensor LineBottomRight sensorPosition 8 detectObstacle OBSTACLE
    -Sensor LineBottomLeft sensorPosition 9 detectObstacle OBSTACLE
Action:
    -MotorAction Drive motorLeft 300 motorRight 300

Event FollowLine
    conditions: LineBottomRight and LineBottomLeft actions: Drive
```

Now the robot can drive along a straight line. If the line has a curve or a turn, we have to add two events: FollowLineRight and FollowLineLeft.
When following a line to the right, the robot will not detect a line anymore with the bottom left sensor but should still detect a line with the bottom right sensor.
For following the line to the left, the case is reversed.

```
Condition:
    -Sensor LineBottomRight sensorPosition 4+4 detectObstacle OBSTACLE
    -Sensor LineBottomLeft sensorPosition 18/2 detectObstacle OBSTACLE
    -Sensor NoLineBottomRight sensorPosition 8 detectObstacle NOOBSTACLE
```

```
    -Sensor NoLineBottomLeft sensorPostition 9 detectObstacle NOOBSTACLE
Action:
    -MotorAction Drive motorLeft 200+100 motorRight 400-100
    -MotorAction TurnRight motorLeft 150*2 motorRight -300
    -MotorAction TurnLeft motorLeft -300 motorRight 300

Event FollowLineRight
    conditions: LineBottomRight and NoLineBottomLeft actions: TurnRight
Event FollowLineLeft
    conditions: NoLineBottomRight and LineBottomLeft actions: TurnLeft
```

If an object is on the line turn the robot.

```
Condition:
    -Sensor FrontSensorLeftMiddle sensorPosition 2 detectObstacle OBSTACLE
    -Sensor FrontSensorMiddle sensorPosition 3 detectObstacle OBSTACLE
    -Sensor FrontSensorRightMiddle sensorPosition 4 detectObstacle OBSTACLE
Action:
    -MotorAction TurnAround motorLeft -300 motorRight 300
Event AvoidObject
    conditions: FrontSensorLeftMiddle and FrontSensorRightMiddle and FrontSensorMiddle actions: T
```

The robot will now follow a black line on the ground.

When all put together:

```
Condition:
    -Tap "TapSensor"
    -Sensor LineBottomRight sensorPosition 4+4 detectObstacle OBSTACLE
    -Sensor LineBottomLeft sensorPosition 18/2 detectObstacle OBSTACLE
    -Sensor NoLineBottomRight sensorPosition 8 detectObstacle NOOBSTACLE
    -Sensor NoLineBottomLeft sensorPostition 9 detectObstacle NOOBSTACLE
    -Sensor FrontSensorLeftMiddle sensorPosition 2 detectObstacle OBSTACLE
    -Sensor FrontSensorMiddle sensorPosition 3 detectObstacle OBSTACLE
    -Sensor FrontSensorRightMiddle sensorPosition 4 detectObstacle OBSTACLE
Action:
    -MotorAction "StopMotors" motorLeft 0 motorRight 0
    -MotorAction Drive motorLeft 200+100 motorRight 400-100
    -MotorAction TurnRight motorLeft 150*2 motorRight -300
    -MotorAction TurnLeft motorLeft -300 motorRight 300

Event Stop
    conditions: TapSensor actions: StopMotors

Event FollowLine
    conditions: LineBottomRight and LineBottomLeft actions: Drive

Event FollowLineRight
    conditions: LineBottomRight and NoLineBottomLeft actions: TurnRight

Event FollowLineLeft
    conditions: NoLineBottomRight and LineBottomLeft actions: TurnLeft

Event ObjectInFront
    conditions: FrontSensorLeftMiddle and FrontSensorRightMiddle and FrontSensorMiddle actions: T
```

Now you can run the program and test it out!

4 Adding More Functionalities

Now you might want to add effects to the robot.
Start by implementing a sound by pressing the center button:

```
Condition:
    -Button CenterButton button CENTER
Action:
    -SoundAction PlaySound set (1, MEDIUM, 1)

Event ButtonPressSound
    conditions: CenterButton actions: PlaySound
```

If you now also want to add lights:

```
Condition:
    -Button LeftButton button LEFT
Action:
    -LightAction ChangeLightColor pos TOP red 32

Event ButtonPressLight
    conditions: LeftButton actions: ChangeLightColor
```

As a last example, use the sound condition of the Thymio robot:

```
Condition:
    -Sound Clap
Action:
    -MotorAction TurnInCircle motorLeft 200+50 motorRight -200-50

Event SoundDetectedTurnInCircle
    conditions: Clap actions: TurnInCircle
```

If you want to add this to the robot we implemented in Section 3, add each condition,action and event to the RobotModel.

5 Good luck!

Now that you know how to program the robot and have implemented a simple code, you can start programming on your own.
If you're having trouble starting, look at the code snippets given above. You can copy a lot of lines and create new programs with a mix-and-match pattern.
If you make a mistake, an error will be shown, and quick fixes will be proposed. Just hover over the word or number which is underlined in red, and select one of the proposed fixes.

If you're interested in learning more about Thymio:https://www.thymio.org/

References

[W J] A. Dame W. Johal S. Magnenat F. Mondada O. Robu. *Augmented Robotics for Learners: A Case Study on Optics*. sensors ith numbers. URL: https://gtc.inf.ethz.ch/publications/augmented-robotics-for-learners--a-case-study-on-optics.html (visited on 05/22/2023).

[Jiw14] Stéphane Magnenat Jiwon Shin R. Siegwart. *Visual Programming Language for Thymio II Robot*. sensors ith numbers. 2014. URL: https://www.semanticscholar.org/paper/Visual-Programming-Language-for-Thymio-II-Robot-Shin-Siegwart/754041308dcac5657dd703bb89d55a8citing-papers (visited on 05/22/2023).