



Build Your Own Virtual Robot

By: Marina Rizk

Mentors: Dr. Ákos Lédeczi
Gordon Stein



Tel (615) 343-7472 | Fax (615) 343-7440
1025 16th Avenue South Nashville, TN 37212
www.isis.vanderbilt.edu



VANDERBILT UNIVERSITY

Overview

- Motivation:
 - Build an educational tool to facilitate the teaching of computational thinking especially for beginners in robotics
- Restrictions:
 - Apply realistic constraints to the designed robot such as the motion mechanism and different sensors
- Goals:
 - Develop problem-solving methods through NetsBlox
 - Appeal to a wider audience

Building the Project

- Used Platforms:

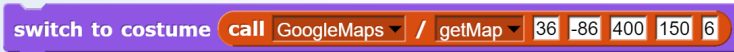
- 1) Unity:

- A cross-platform engine with a built-in IDE that can also give realistic simulations when provided the suitable restrictions. It also has built in physics properties such as gravity
- The robot components and the testing environments were designed in Unity
- Provided the suitable script, the robot in Unity could be controlled through Netsblox

Building the Project

2) Netsblox:

- NetsBlox uses a visual programming language (blocks-based) which allows people to develop networked programs such as getting maps and controlling robots

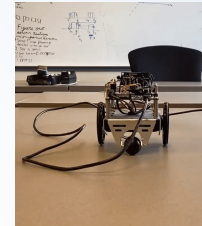
A screenshot of a NetsBlox code block. It is a purple block with a 'switch to costume' button on the left. The main body of the block contains a 'call' block with a dropdown menu set to 'GoogleMaps', followed by a '/' separator, another dropdown menu set to 'getMap', and a series of input fields containing the coordinates '36', '-86', '400', '150', and '6'.

- NetsBlox will be used to configure the robot built in Unity by writing the code to solve the given challenges

Restrictions

Motion:

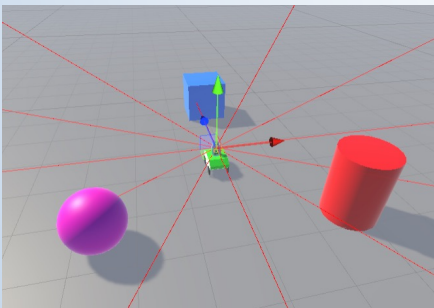
- Robot motion was restricted to be like that of the Activitybot such that code was written in Netsblox to test the motion of both real and simulated robots



```
when key up-arrow pressed?
repeat until not key up-arrow pressed?
if key up-arrow pressed? and key left-arrow pressed?
run RoboScape / send Robot ID set-speed=8-75
else
if key up-arrow pressed? and key right-arrow pressed?
run RoboScape / send Robot ID set-speed=75-8
else
if key up-arrow pressed?
run RoboScape / send Robot ID set-speed=75-75
if not key up-arrow pressed?
run RoboScape / send Robot ID set-speed=0-0
when key down-arrow pressed?
repeat until not key down-arrow pressed?
if key down-arrow pressed? and key left-arrow pressed?
run RoboScape / send Robot ID set-speed=5-75
else
```

Sensors:

- Sensors were also programmed to function as they would in real life i.e., laser mechanism in LIDAR

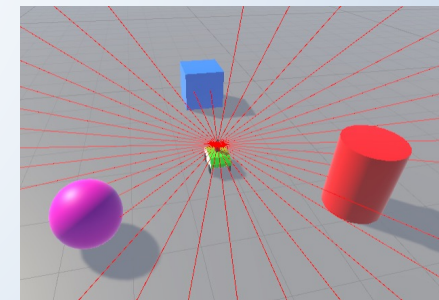


Angle range: 360°
Resolution: 30°

```
call LIDAR / measure call LIDAR / getDevices
```

Calling the Lidar Function and getting the results in Netsblox

1	100
2	100
3	2.209885
4	100
5	100
6	100
7	2.406087
8	100
9	100
10	100
11	2.281992
12	100
length: 12	



Angle range: 360°
Resolution: 10°

Demo

Conclusion

- Lessons learned:
 - Make the program as much efficient as possible before inserting the graphics
 - When facing an issue with the code, look at a resource that is directly related to the project
- Challenges:
 - Limit the design to the given restrictions
 - Build challenges to teach robotics configuration in Netsblox while making it appealing to users
- What went well
 - Feedbacks to assure that the project was developing in the desired direction
 - Gain a lot of experience in Unity, C#, and Netsblox