**Vietnam National University**

**University of Engineering and Technology**

**Challenge 2 Report**

**Group members:**

Phí Văn Minh - 16022411

Lê Trung Thông - 16020286

Nguyễn Duy Quang - 16020271

Hanoi, April - 2019

Table of Contents

[Introduction 3](#_Toc8513819)

[I. Requirements 4](#_Toc8513820)

[II. Installation 4](#_Toc8513821)

[1. Clone repository 4](#_Toc8513822)

[2. Active virtual environment 4](#_Toc8513823)

[3. Run code 4](#_Toc8513824)

[4. Tutorial 4](#_Toc8513825)

[III. Solution 5](#_Toc8513826)

[1. Get topview map of scene 5](#_Toc8513827)

[2. Localization 5](#_Toc8513828)

[3. Event processing 7](#_Toc8513829)

[IV. References 7](#_Toc8513830)

# Introduction

This report shows the solution to solve challange 2 in the Assignment 2. The challange 2 is: Drawing Trajectories on Top-view Map of the room.

April – 2019

Phí Văn Minh

Lê Trung Thông

Nguyễn Duy Quang

# I. Requirements

1. OS: Mac OSX 10.9+, Ubuntu 14.04+.

2. Graphics Card: DX9 (shader model 3.0) or DX11 with feature level 9.3 capabilities.

3. CPU: SSE2 instruction set support.

4. Python 3.6+

6. Linux: X server with GLX module enabled

7. Pip list necessary:

+ matplotlib (version: 3.0.3)

+ pillow (version: 6.0.0)

+ virtualenv (version: 16.5.0)

+ keyboard (version: 0.13.3)

+ ai2thor (version: 1.0.2)

8. ai2thor core file: thor-201904041253-… (491MB)

# II. Installation

## **1. Clone repository**

git clone <https://github.com/sugarless266/robot-uet-spring2019.git>

## **2. Active virtual environment**

cd robot-uet-spring2019

source bin/activate

## **3. Run code**

sudo python topview.py

## **4. Tutorial**

"*Left Arrow*": Move agent to the left

"*Right Arrow*": Move agent to the right

"*Up Arrow*": Move agent to straight ahead

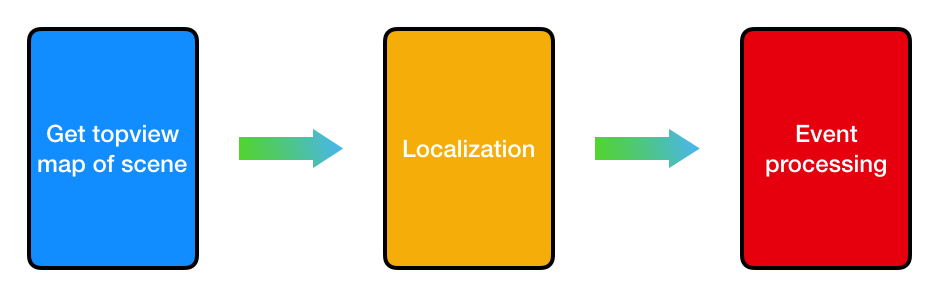
"*Down Arrow*": Move agent back

"*R*": Rotate agent 90 degree to the right

"*U*": Look Up

"*D*": Look Down

# III. Solution



*Figure 1. Flow to solve challenge*

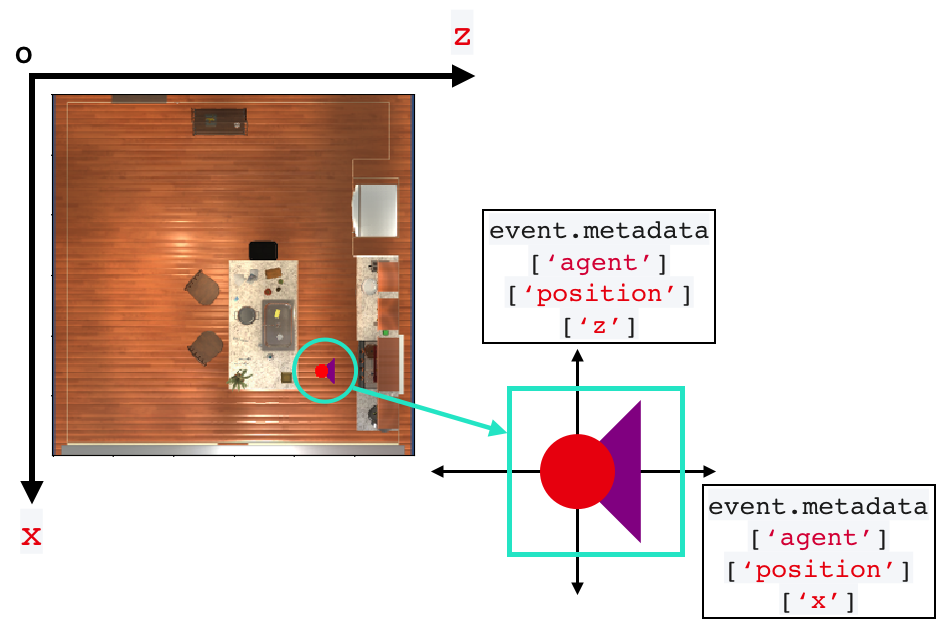
## **1. Get topview map of scene**

controller.step(dict(action = ‘ToggleMapView’))

Run code above once to get top view and again to get normal view back

## **2. Localization**

*2.1. Position*



*Figure 2. Agent’s position in global space*

Each agent have their own frame at every event. And the agent’s position in the frame is different from position in global space. So the mission here is have to convert agent’s position in frame to global space. There is a class to do this mission. That is “ThorPositionTo2DFrameTranslator”. It is written by ai2thor developer [1].

*2.2. Rotation*

We define a triangle that show the camera view. Each time initialize room or rotate agent, rotate the triangle follow that. The algorithm to rate triangle is base on rotation mathematics.

def rotatePoint(point, root):

x = point[0] - root[1]

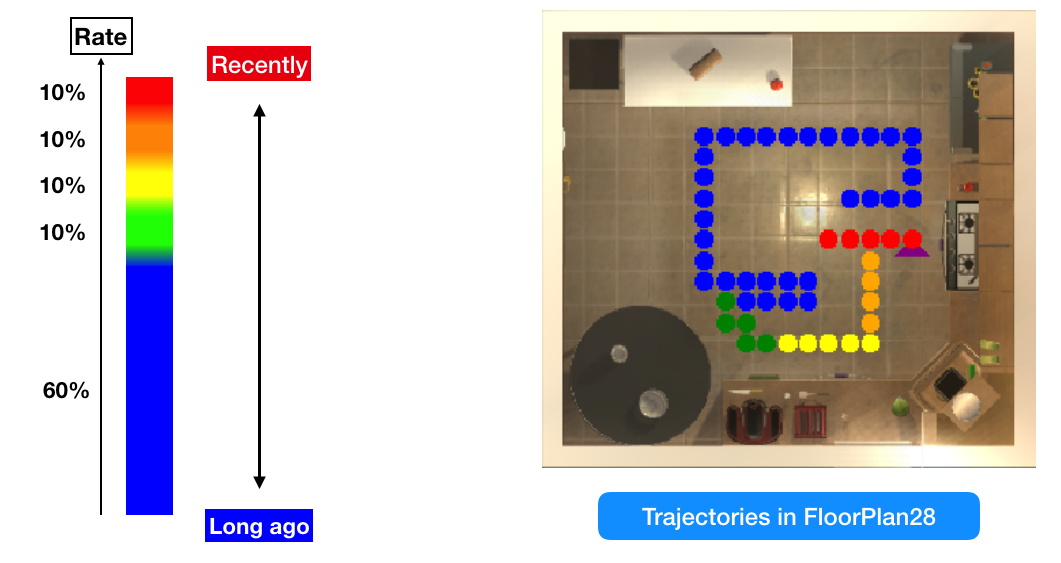
y = point[1] - root[0]

return (-y + root[1], x + root[0])

The code above show rotate point 90% with root point.

## **3. Event processing**

We define “listCircle” list to save position of the agent in global space at each step and rotate the triangle if rotate the agent. On the trajectories, fill color with rate: 10% red, 10% orange, 10% yellow, 10% green, 10% blue. Red color is recently visited, and decrese to Blue color is long ago visited.



*Figure 3. Event processing*

# IV. References

[1] <https://github.com/allenai/ai2thor/issues/124>