EPFL tandem race 2022 - Loomo pipeline

In this file, we will explain how you can deploy your models to the robot. Please follow the steps carefully for the deployment.

Figure 1: Loomo race at EPFL in 2019

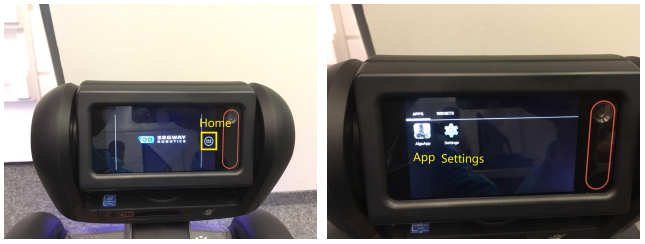
1 Deployment

Once a model is well-trained, you can now deploy it to Loomo, a Segway robot, for the person tracking task. To ease the model deployment, we will use a communication protocol between a robot and a cloud computer, *e.g.*, your laptop or a server. With this protocol, you can run a trained model on the cloud that receives image streams from the robot and returns results to it. We suggest you to first use your laptop and then switch to the server (login details will be provided later).

1.1 Robot

1.1.1 Turn on your robot

To turn on a Loomo, press the button on its body and the button on its head in turn. Figure 2 shows the main pages after you turn on the robot.

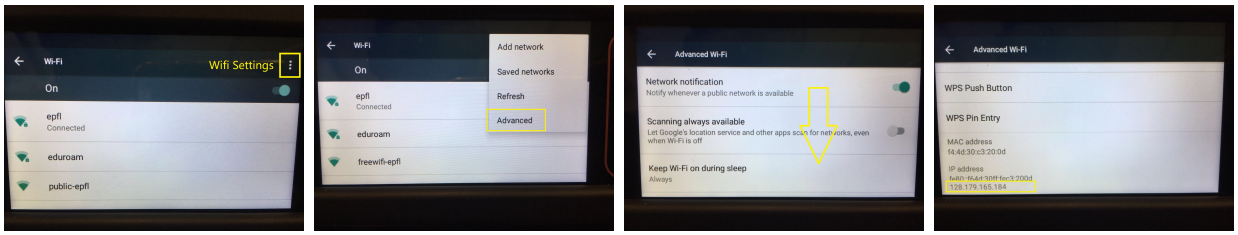


(a) Home **(b)** Apps

Figure 2: Turn on robots

1.1.2 Find dynamic IP address

To establish the communication between the robot and the cloud via Wi-Fi, you need to find the IP address, as shown in Figure 3.

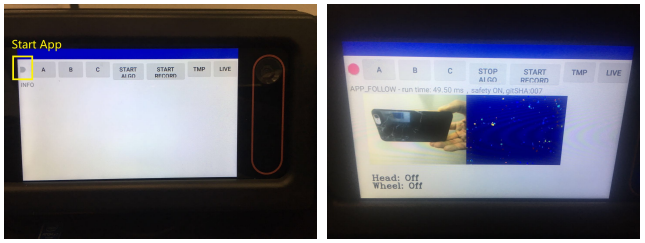


(a) Step 1 **(b)** Step 2 **(c)** Step 3 **(d)** Step 4

Figure 3: Find IP address

1.1.3 Start robot app

You can now get back to the main page by tapping the robot's ear. Figure 4 shows how to start the robot app for person tracking.



(a) Before Start **(b)** After Start

Figure 4: Start App

1.1.4 Turn on controllers

Once the app is initialized, you can switch on the controller of the robot's head and wheel with the button

- A: head control
- B: wheel control

1.2 Cloud

Once the robot is settled, you can now start your model on the cloud (your laptop or the server). The codes can be found here.

client.py file receives images from robot and passes them to *detector.py*. Then, *detector.py* file receives the images and sends back the position of the person to *client.py* which also sends back the data to the robot. **Note: You should not change *client.py*. Thus, note the changes in *detector.py* to be consistent with the *client.py* code.**

1. Test the IP address and the video stream (inside python folder)

```
python test.py --ip-address <robot ip>
```

2. Run your model

```
python client.py --ip-address <robot ip> --checkpoint
↪  <location of trained model>
```

1.3 Tuning

To improve the performance of your tracking robot, you may want to fine-tune the resolution of your image stream as well as the parameters of your controllers. A eash way to change these parameters is to push a configuration file into the robot.

1. Connect your laptop to a robot through WiFi

```
add connect <robot ip>
```

2. Push configuration file

```
adb push follow.cfg /sdcard/
```

The default parameters are as follows:

```
1.50    % k_p, head controller
0.10    % k_i, head controller
0.01    % k_d, head controller
0.50    % k_p, vehicle controller
0.01    % k_i, vehicle controller
```

```
0.01    % k_d, vehicle controller
8.00    % image downscale rate, the original image is 640 x
↪  480
```

2 Appendix: Environment Setup on Robots

The robots have environments set up. In case applications are not working, you can follow these steps to install them. **Definitely check with a TA before doing these steps.**

Prior to the model deployment, some preparations on robots are required to setup the hardware and software environments. First, the robot needs to be switched to the *developer* mode, as shown in Figure 5.

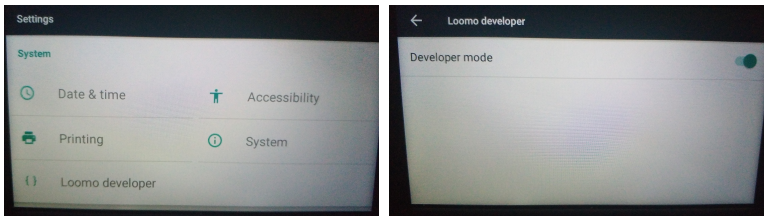


Figure 5: Loomo Developer Mode

Furthermore, an android app designed for this project needs to be installed. The source code can be found at https://github.com/segway-robotics/loomo-algodev. It is recommended to use Android Studio 3.1.4. to build and install the app.

Before compiling the source code, make sure that the local paths to ndk and sdk are added in the *local.properties* file under *gradle*, as shown in Figure 6.

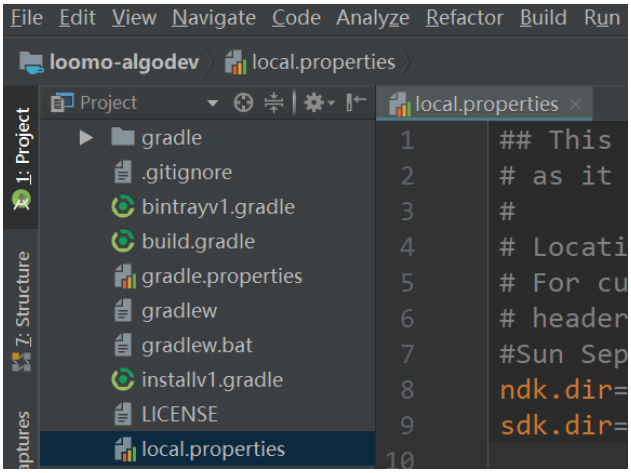


Figure 6: Gradle settings

To build the source code, open the file *AlgoFollow.cpp*, and build the module as shown in Figure 7.

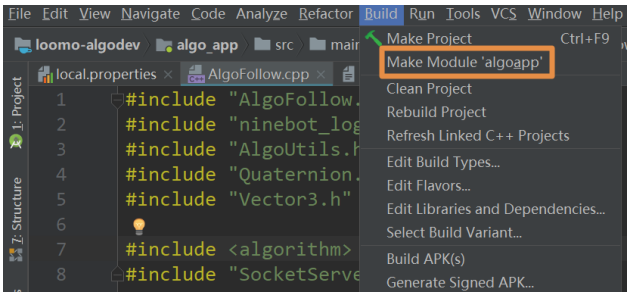


Figure 7: Build project

Once the source code is built successfully, we can now install it to the robot, as shown in Figure 8. Note that our laptop needs to be connected to the robot through WiFi, as described in Section 1.3.

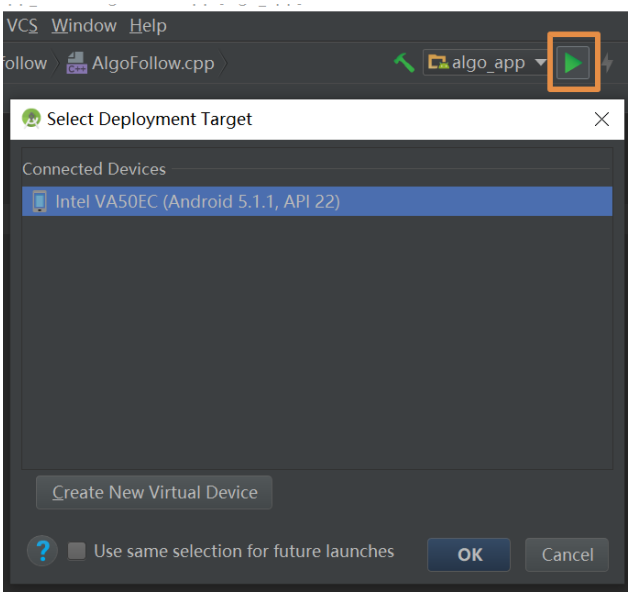


Figure 8: Install app