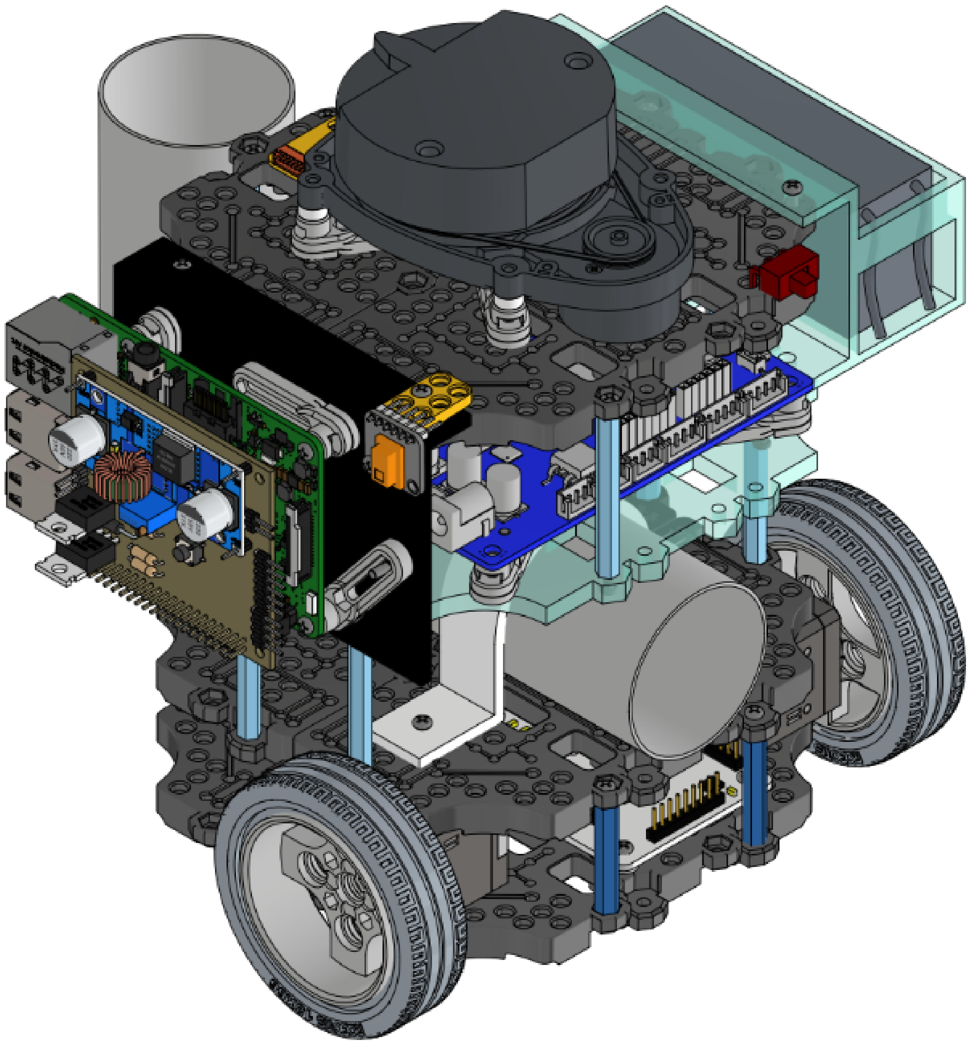
EG2310 AY 21/22

Studio 2 Group 2

Tánkyu 2310i, Software Setup Manual



Ang Kai Jun	A0235198W
David Chong Joon Wei	A0236533H
Ha Zhe Li	A0236541J
Rayner Lim Fang Yuh	A0238285W
Wang Yanxiao Austin	A0240524R

Contents Page

1.0 Description

The Tánkyu 2310i, developed by NUS IDP students from EG2310, is a modified Robotis Co. Turtlebot 3. It is augmented with LIDAR, NFC-detection, IR-detection and a flywheel firing system to accomplish its set objectives - Autonomous navigation within a closed and connected maze, locating a loading bay demarcated by NFC tags, and firing a ping-pong ball at a target with high IR-signature.

This Software Setup Manual contains detailed information on the laptop and RPi setup process. It aims to be a step-by-step guide on how to set up and operate the Tánkyu 2310i from scratch for beginners and thus contains overlapping information from our GitHub repository. For experienced users, refer to our GitHub repository for just the required packages and code.

In this guide, we will be using Ubuntu 20.04.4, ROS2 Foxy and Python3.6. This guide is correct as of 22/04/2022.

2.0 Setup Guide

The following will be one time setup instructions if it is your first time operating the Tankyu 2310i. Otherwise, navigate directly to Section 3.0: Operating Instructions.

2.1 On your Linux Laptop

1. Install Ubuntu 20.04 and ROS 2 Foxy using the instructions in Section 3.1 of the following webpage (make sure you click on the “Foxy” tab at the top of the webpage):
   https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/#pc-setup

2. Add the following to the .bashrc on your laptop:
   ```
   $ export TURTLEBOT3_MODEL=burger
   $ alias rteleop ='ros2 run turtlebot3_teleop teleop_keyboard'
   $ alias rslam='ros2 launch turtlebot3_cartographer
   cartographer.launch.py'
   ```

3. Reload the .bashrc:
   ```
   $ source ~/.bashrc
   ```

4. We will first create a ROS2 package
   ```
   $ cd ~/colcon_ws/src
   ```

```
$ ros2 pkg create --build-type ament_python auto_nav
$ cd auto_nav/auto_nav
```

5. Move the file in the directory temporarily to the parent directory:
   ```
   $ mv __init__.py ..
   ```

6. We are now ready to clone the GitHub repository.
   (NOTE: If you would like to edit the code based on your application, you may first fork the repository on Github. After that, you may clone your Github repository by replacing the user id ‘hahaha2002’ in the code below with your user id.)
   ```
   $ git clone git@github.com:hahaha2002/r2auto_nav.git
   ```

7. Move the file init.py back
   ```
   $ mv ../__init__.py
   ```

8. (Optional) If you wish to commit your changes to the GitHub repository, eg. you forked the repository, configure your git environment.
   ```
   $ git config --global user.email "<your email>"
   $ git config --global user.name "<your user name>"
   $ git config --global push.default simple
   ```

9. Next, build the auto_nav package
   ```
   $ cd ~/colcon_ws
   $ colcon build --packages-select auto_nav
   ```

10. Install the scipy Python package
    ```
    $ pip3 install scipy --user
    ```

11. Install numpy and matplotlib packages
    ```
    $ pip3 install numpy matplotlib
    ```

2.2 On your Raspberry Pi

1. Using Ubuntu, burn the ROS2 Foxy image to the SD card of the R-Pi's using the instructions starting from Section 3.2.3 at (make sure you click on the "Foxy" tab at the top of the webpage):
   https://emanual.robotis.com/docs/en/platform/turtlebot3/sbc_setup/#sbc-setup

2. Complete the OpenCR setup in Section 3.3
   https://emanual.robotis.com/docs/en/platform/turtlebot3/opencr_setup/#opencr-setup

3. Add the following to the .bashrc on both R-Pi's:
   ```
   $ export TURTLEBOT3_MODEL=burger
   $ alias rosbu='ros2 launch turtlebot3_bringup robot.launch.py'
   ```

4. Reload the .bashrc:
   ```
   $ source ~/.bashrc
   ```

5. Install gpio packages
   ```
   $ sudo apt-get install rpi.gpio
   ```

6. Install i2c-tools
   ```
   $ sudo apt install -y i2c-tools python3-pip
   ```

7. Change permissions of /dev/i2c-1
   ```
   $ sudo chmod a+rw /dev/i2c-1
   ```

8. Install i2c-tools
   ```
   $ pip3 install smbus2
   ```

9. Install spidev module
   ```
   $ pip3 install spidev
   ```

10. Create a ROS2 package
    ```
    $ cd ~/turtlebot_ws/src
    $ ros2 pkg create --build-type ament_python auto_nav
    $ cd auto_nav/auto_nav
    ```

11. Copy mission.py and the RPi_files folder from your laptop to the RPi
(** Code run on laptop with the cloned Github repository)

```
$ scp <path to r2auto_nav directory>/mission.py ubuntu@<RPi IP
address>:~/turtlebot_ws/src/auto_nav/auto_nav
$ scp -r <path to r2auto_nav directory>/RPi_files ubuntu@<RPi IP
address>:~/turtlebot_ws/src/auto_nav/auto_nav
```

12. Build the auto_nav package

```
$ cd ~/turtlebot_ws
$ colcon build --packages-select auto_nav
```

2.3 Optional Setup

This section includes optional setups to make it easier to work on the Tankyu 2310i for multiple runs.

2.3.1 Keeping track of RPI IP address

You can use a virtual machine on Amazon Web Services (AWS) to keep track of the dynamic IP address assigned to the Raspberry Pi (R-Pi) on your TurtleBot. You may do the following in Linux.

a. Sign up for a free-tier AWS account at (click the Create a new AWS account button): https://console.aws.amazon.com/console/home?nc2=h_ct&src=header-signin

b. When asked to choose the Account type, select “Personal”.

c. Select “Singapore” as the Country/Region.

d. Enter a credit/debit card in order to create the account. If you follow the instructions, you will not incur any charges.

e. Enter your mobile number in order to get an access code.

f. Select the “Basic Plan” when asked to select a Support Plan.

g. Click the “Sign in to the Console” button to sign into your AWS account.

h. Click “EC2” under “AWS services”.

i. Click the “Launch Instance” button.

j. Make sure the “64-bit (x86)” option under “Amazon Linux 2 AMI (HVM), SSD Volume Type” is selected, and then click the “Select” button.

k. Make sure the “t2.micro” instance type is selected, and then click the “Review and Launch” button. The free-tier account allows you to have a t2.micro EC2 instance running 24 hours a day without any charges.

l. Click the “Launch” button.

m. When the key pair window comes up, select the “Create a new key pair” under the first drop-down menu.

n. Enter a name for your key pair (e.g. MyKeyPair), and then click the “Download Key Pair” button. A .pem file will be downloaded, which you can use to access your AWS account in the future.

o. Click the “Launch Instances” button.

p. While waiting for the instance to launch, you can click the “Services” button in the top-left of the window and select “EC2”. This will take you to the EC2 Dashboard.

q. Click on the “Instances (running)” button, which will show you information about your running instances. There should only be one entry at this point, so you can click on the instance under the “Instance ID” column. This should show you information about your instance, including the public IPv4 address. Note down the address as you will be using it for the rest of the semester.

r. You will need to change the permission settings on the key pair file, so first open up the Terminal application and check your file permissions:

```
ls -l ~/Downloads/MyKeyPair.pem
```

This should return the following:

```
-rw-r--r--@ 1 syen staff 1700 Dec 13 22:18
/Users/syen/Downloads/MyKeyPair.pem
```

indicating that the owner (i.e. you) has read and write permissions and other users only have read permissions. AWS will reject these permissions as not being secure enough, so you will need to change the permissions by doing:

```
chmod 400 ~/Downloads/MyKeyPair.pem
```

If you check your permissions again:

```
ls -l ~/Downloads/MyKeyPair.pem
```

you should see:

```
-r--------@ 1 syen staff 1700 Dec 13 22:18
/Users/syen/Downloads/MyKeyPair.pem
```

which means that only you have read permissions, and even the write permissions have been removed so you will not accidentally modify the file.

s. Now you can try accessing your EC2 instance (replace IP-address below with the IP address of your EC2 instance):

```
ssh -i ~/Downloads/MyKeyPair.pem ec2-user@IP-address
```

You might get a warning that says:

```
The authenticity of host '52.221.221.183 (52.221.221.183)' can't be established.
ECDSA key fingerprint is
SHA256:XRs+sZbEFRleJY8H41EH4nxRG1BYj4XO5+tKWCREzlk.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

You should enter “yes”. This should show you the following login screen:

```
       __|  __|_  )
```

```
   _|  (     /   Amazon Linux 2 AMI
  ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
7 package(s) needed for security, out of 19 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-33-20 ~]$
```

This means that you have successfully accessed your EC2 instance. You can type "exit" to logout.

t. In order to simplify the login process, create a SSH public-private key pair on your laptop by doing the following in your Terminal window:

```
ssh-keygen
```

Hit Return three times to use the default file name, and to accept a blank password. This will create two files in the .ssh directory in your home directory: id_rsa and id_rsa.pub

u. You can now use the following command to copy the second file to your EC2 instance (you can hit the up arrow to access your command history, and then use the left and right button to edit the command):

```
scp -i ~/Downloads/MyKeyPair.pem ~/.ssh/id_rsa.pub ec2-user@IP-address:~/
```

v. Login again to your EC2 instance (using the up arrow):

```
ssh -i ~/Downloads/MyKeyPair.pem ec2-user@IP-address
```

w. Run the ssh-keygen command to create keys on your EC2 instance:

```
ssh-keygen
```

x. Move the id_rsa.pub file from your laptop to the .ssh directory:

```
mv id_rsa.pub ~/.ssh/authorized_keys
```

y. Logout again by entering "exit".

z. You should now be able to login without using the .pem file (use the up arrow to edit a previous command):

```
ssh ec2-user@IP-address
```

You can logout after verifying that you can login successfully.

aa. You can create an alias on your laptop using the nano editor to make logging in even easier:

```
nano ~/.ssh/config
```

bb. Enter the following (replace IP-address below with the IP address of your EC2 instance):

```
Host aws
        HostName IP-address
        User ec2-user
```

cc. Type Ctrl-x to exit the editor, type “y” to save the document, and hit Return to use the current file name.

dd. You should now be able to login by doing:

```
ssh aws
```

ee. Be sure that you do not shut down or restart your AWS instance as it will change the IP address of the AWS instance, and you will then need to update the address in your scripts.

2.3.2 Bypassing RPi password requirement

You can now optimise your setup so you will find it easier to work with the TurtleBot. First, we will remove the need to plug a monitor into your R-Pi in order to find out its IP address. We will do this by creating a script that will be run during startup to send the R-Pi’s IP address to your AWS instance.

a. Open a terminal window on your laptop and ssh to your R-Pi. We will then create a SSH public-private key pair on the R-Pi that will allow it to ssh to AWS to update the IP address:

```
ssh-keygen
```

Hit Return three times to use the default file name, and to accept a blank password. This will create two files in the .ssh directory in the home directory: id_rsa and id_rsa.pub

b. Display the public key on the R-Pi:

```
cat ~/.ssh/id_rsa.pub
```

c. In another Terminal window on your laptop, login to your group’s AWS account:

```
ssh aws
```

d. Edit the authorized_keys file using the nano editor:

```
nano ~/.ssh/authorized_keys
```

and paste the public key from the R-Pi into the file.

e. Exit and save the file.

f. On the R-Pi, use the nano text editor to create a file with information to connect to your AWS instance:

```
nano ~/.ssh/config
```

g. Enter the following into the file (replace IP-address below with the IP address of your EC2 instance):

```
host aws
        HostName <IP-address>
        User ec2-user
```

h. Exit and save the file.

i. You should now be able to do the following from your R-Pi to login to your AWS instance:

```
ssh aws
```

j. We now need to create a system service that will run whenever the R-Pi starts up:

```
sudo nano /etc/systemd/system/ip2aws.service
```

k. Enter the following into the file, which will run the ip2aws.bash script in the home directory when the R-Pi boots up:

```
[Unit]
After=network.service

[Service]
ExecStart=/home/ubuntu/ip2aws.bash

[Install]
WantedBy=default.target
```

l. We will now create the script that the service will run:

```
nano ~/ip2aws.bash
```

m. Enter the following into the file for the R-Pi, which will get the IP address, remove the space character at the end, and use the ubuntu account to save the IP address on AWS in a file named “rpi.txt”:

```
#!/bin/bash

myIP=`hostname -I | tr -d " "`
echo $myIP | runuser -l ubuntu -c 'ssh aws "cat - > rpi.txt"'
```

n. Change the permissions of the files:

```
chmod 744 ~/ip2aws.bash
sudo chmod 664 /etc/systemd/system/ip2aws.service
```

o. Register the service:

```
sudo systemctl daemon-reload
sudo systemctl enable ip2aws.service
```

p. Reboot the R-Pi’s:

```
sudo reboot
```

q. Once the R-Pi’s have booted up, you should be able to check that the IP addresses are correctly saved into the rpi.txt file by doing the following from your laptop:

```
ssh aws cat rpi.txt
```

r. Edit the .bashrc file on your laptop to add the following at the end:

```
alias sshrp='ssh ubuntu@`ssh aws cat rpi.txt`'

alias sshrp2='ssh ubuntu@`ssh aws cat rpi2.txt`'
```

s. Reload the bashrc file:

```
$ source ~/.bashrc
```

t. Copy the ssh public key from your laptop to the R-Pi’s:

```
ssh-copy-id ubuntu@`ssh aws cat rpi.txt`
```

u. You should now be able to ssh to the TurtleBot R-Pi by doing:

```
$ sshrp
```

3.0 Operating Instructions:

1. Turn on the Tánkyu 2310i
2. SSH into the RPi on the turtlebot.

   ```
   $ sshrp
   ```

   Refer to section 2.3 if you have not set up an alias for sshrp. Alternatively, you may manually ssh into your RPI by using

   ```
   $ ssh ubuntu@<RPI IP address>
   ```

3. Run the bring up on the RPi

   ```
   $ rosbu
   ```

4. Start the mission.py code on the RPi

   ```
   $ python3 mission.py
   ```

5. The mission.py file should automatically run the pigpio daemon, if you encounter an error with the pigpio daemon not running automatically, enter this command and repeat step 4.

   ```
   $ sudo pigpiod
   ```

6. Start the navigation.py code from your linux system

   ```
   $ python3 navigation.py
   ```

7. The navigation.py code should open a RViz window automatically, if you encounter an error with the RViz window not opening automatically, you can manually open a RViz window by entering this command (refer to section 2.3 if you have not set up an alias for RViz).

   ```
   $ rslam
   ```

8. To terminate the program, press ‘ctrl + C’.
9. You should see three new files in the directory:
   a. mazemapfinally.png
   b. lidar.txt
   c. map.txt

3.1 Factory Acceptance Test / Pre-mission Checks

The Factory Acceptance Test or Pre-mission Checks aim to ensure each individual system of the Tánkyu 2310i is operational before the mission. It also acts as a way to isolate faults or for modular testing if the need arises. It is recommended that the Factory Acceptance Tests be run before every mission to ensure a successful mission.

1. Turn on the Tánkyu 2310i.
2. SSH into the RPi on the turtlebot.

```
$ sshrp
```

3. Navigate to the fac_test directory.

```
$ cd RPi_Files/fac_test
```

4. Run the code for each individual system to test.

```
$ python3 <code name>
```

code name	How to trigger the test	Expected outcome	Remarks
servo_test	Enter an integer from 0 to 180	Servo moves to specified angle	Ensure no obstructions before actuating servo, else the servo may fry.
motor_test	Enter either 0 or 1. 0: Stop motor 1: Start motor	Both motors run on command. Left motor spins anti-clockwise Right motor spins clockwise	Ensure no obstructions between motors before running code. Motors do not have enough torque to eject a wedged ping-pong ball.
nfc_test	Place an nfc tag below PN532	Terminal reads “NFC Detected”.	To check i2c connectivity, use `$ sudo i2cdetect -y 1`
ir_publisher	Nil	Terminal starts publishing messages to ir_data topic	To check i2c connectivity, use `$ sudo i2cdetect -y 1`
ir_test	Place a hot object in front of the AMG8833	Live feed of the temperature array appears. Terminal reads out the hottest temperature.	Run this code **on laptop** after running ir_publisher on RPi

3.2 Calibration Instructions

3.2.1 Navigation Code

Under 'Adjustable variables to calibrate wall follower', you may experiment with different parameters to calibrate the navigation algorithm to suit your needs.

Variable name	**Description**	**Recommended value**
d	Distance from wall	0.35
fd	Front distance when approaching a merging wall	d + 0.1
reverse_d	Distance to reverse when too close to front wall	0.20
speedchange	Linear speed	0.17
turning_speed_wf_fast	Fast rotate speed, used when avoiding obstacle in front	0.75
turning_speed_wf_slow	Slow rotate speed, used when reversing, finding wall or too close to wall	0.40
snaking_radius	Distance from wall before correcting drift	d - 0.07
cornering_speed_constant	Coefficient of speedchange during cornering to prevent over/under steer	0.5

3.2.2 Mission Code

Under 'Adjustable variables to calibrate targeting' you may experiment with different parameters to calibrate the targeting algorithm to suit your needs.

Variable name	Description	Recommended value
rotatechange	Angular speed	0.35
speedchange	Linear speed	0.20
min_temp_threshold	IR live feed minimum value reference (Appears blue)	30.0
max_temp_threshold	IR live feed maximum value reference (Appears red)	35.0
detecting_threshold	Minimum temperature to identify target as a "Hot target"	32.0
firing_threshold	Acts as a second check after centering target	35.0
ir_offset	Offset angle for each IR sensor orientation (0°→0, 45°→31, 90°→69)	31.0

4.0 References

Part of the Software Setup Manual were adapted from the lab handouts produced by Prof Yen Shih Cheng, Director of NUS, Innovation and Design Programme, as part of the EG2310 - Fundamentals of Systems Design module.