DGMD S-17 Final Project
JetBot AI kit based on NVIDIA Jetson Nano

Wen Zhe Xian

Goal of this project

- Collision avoidance
- Line following
- Line following + stop when obstruct present, go when obstruct remove
- Object following + stopping when close to the object

Level of Autonomy

- Should be level 3 since it only has one camera in the front to detect the Jetbot’s surroundings and it can make informed decisions for itself, such as turning, starting and stopping. But it still requires human supervision. The person must remain alert and ready to take control if the system is unable to execute the task.

Material list

1. Waveshare Jetbot AI kit

2. NVIDIA Jetson Nano 4GB (for better performance)

Material list

3. 18650 battery X 3 packs

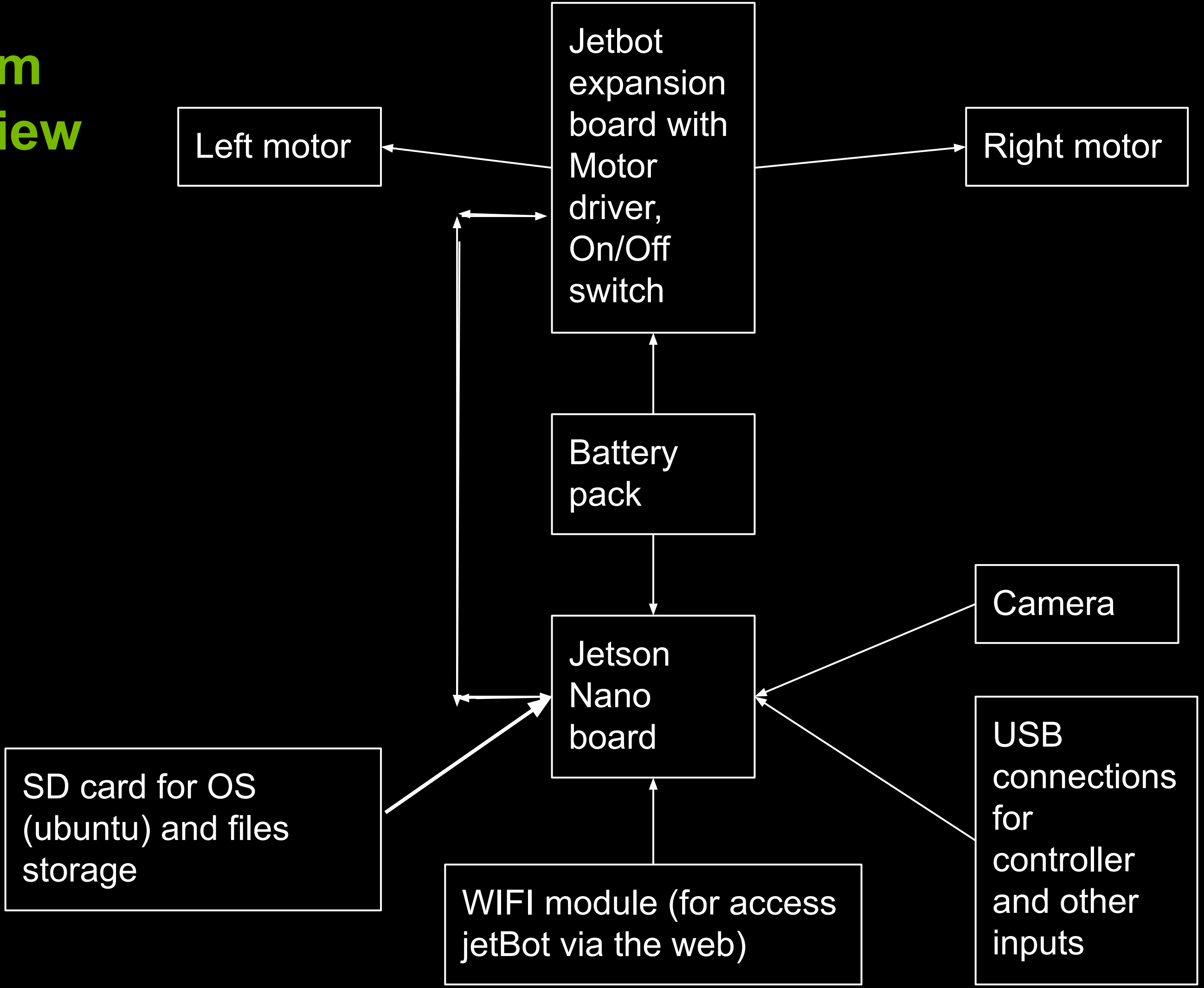
Jetson nano has 4 processing cores.

Jetson nano could runs in 5W (50% performance, 2 processing cores will being shut down)

and 10W power mode (100 % performance)

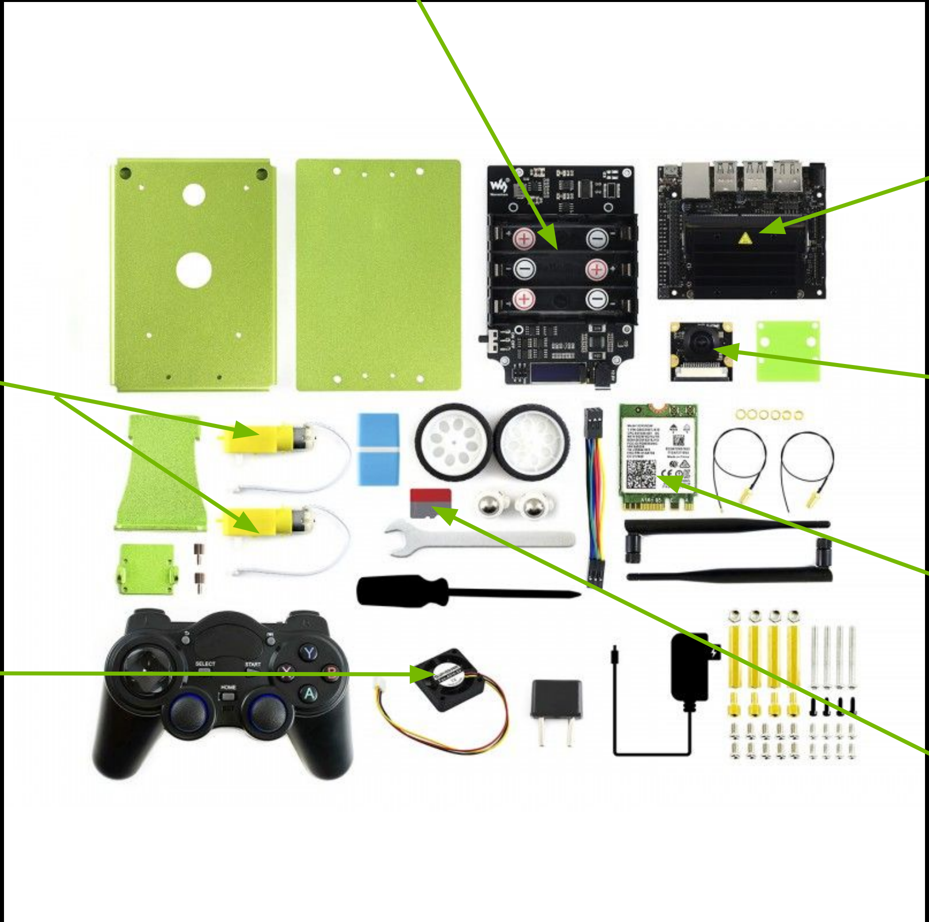
Output: 3.6 V each, 3 packs could be around 10.8V to support 10W max performance

System overview



Jetbot expansion board with Motor driver, On/Off switch
Left motor
Right motor
Battery pack
Jetson Nano board
Camera
USB connections for controller and other inputs
SD card for OS (ubuntu) and files storage
WIFI module (for access jetBot via the web)

Major Components



Jetbot expansion board with Motor driver, On/Off switch

NVIDIA Jetson Nano board

Camera module

Wifi module for web based access to the jetbot

SD memory card for the operating system (Ubuntu)

Two Motors to control the left wheel and right wheel individually

Optional fan for better thermo

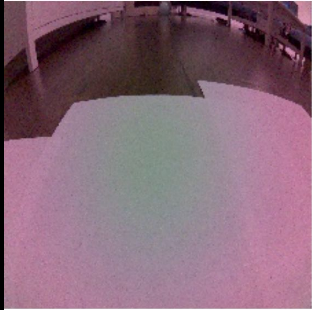
Software and developing tools

- Python 3.0 Jupyter notebook
- Opencv
- Codes are all base on Github NVIDIA-AI-IOT / jetbot wiki, with modifications to fit my waveshare third party jetbot, because the scale of the speed is different than the first party kit
- Jetson OS named JetPack 4.3 based on Ubuntu from NVIDIA
- TensorRT, is a library developed by NVIDIA for faster inference on NVIDIA graphics processing units (GPUs)
- PyTorch, is open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing
- COCO dataset for objects tracking neural network training
- 420 pictures for line following neural network training
- 120 pictures for collision avoidance neural network training

Hardware assembling and Software installation

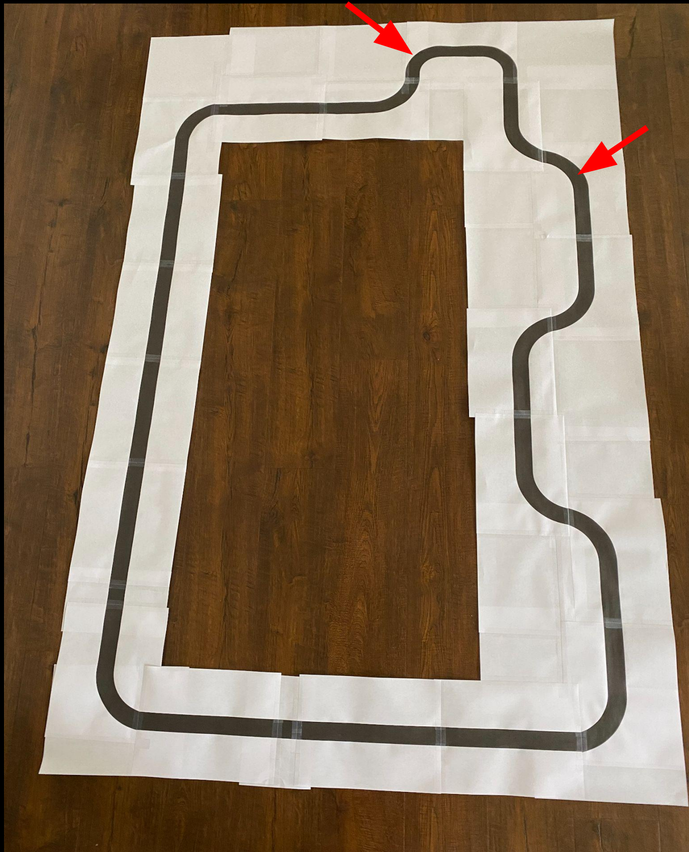
- Hardware assembling by following
  - https://www.waveshare.com/wiki/JetBot_AI_Kit_Assemble_Manual
- Software installation by following
  - https://www.waveshare.com/wiki/JetBot_AI_Kit
  - https://jetbot.org/master/
- Configured to 5W power mode by following the waveshare software installation instructions.
  - It turned out to be a big mistake
  - Most features not perform properly and laggy by using 5W power mode.

Line following: Trial #1



- Racing track # 1
- 5W power mode
- 120 pictures were taken for neural network training
- Jetbot lost control with default speed = 0.4 and steering = 0.9, steering KD (turing rate) = 0
  - System fail due to hardware limitation
- Jetbot is not able to see the line and lose the target when making the turns on the places labeled with arrow, then lost control
  - Sharp turn and not enough data for training are the main reasons for the failure

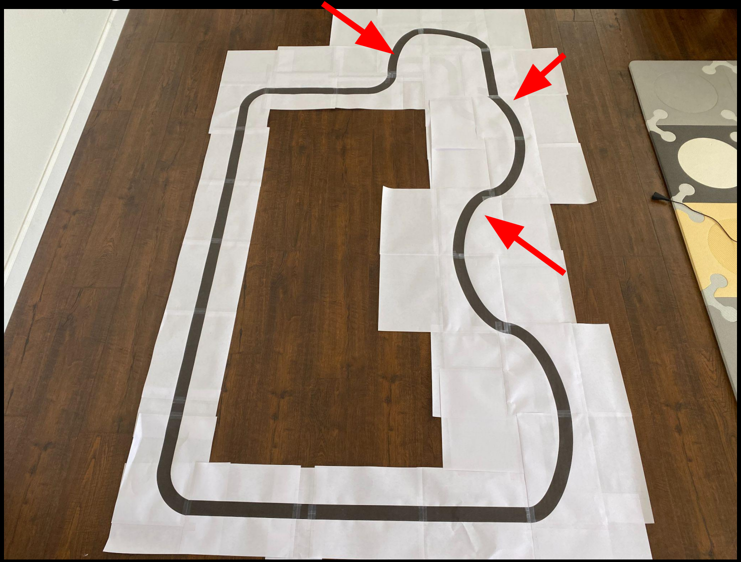
Racing track #1



Line following: Trial #2

- Racing track #2 (improved and successful)
    - Camera is able to see the line during the turn now
- 420 pictures were taken for neural network training
- 10W power mode, Speed = 0.09, steering = 0.02, steering KD (turing rate) = 0.07

Racing track #2



Collecting data for training

Collision avoidance

- 10W power mode
- Pre-train model from Gibhub
- Speed = 0.09, steering = 0.02
- Success

Line following + stop when obstruct present + go when obstruct remove

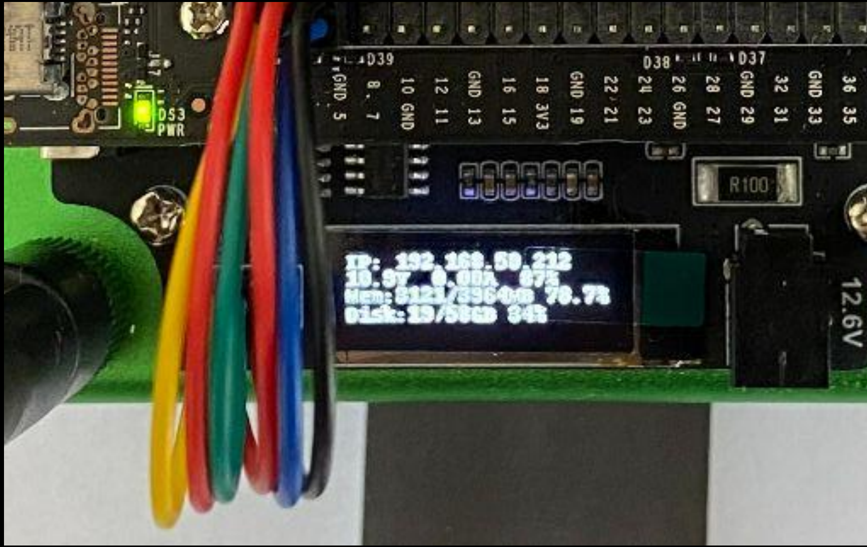
- 420 pictures taken by myself for line following neural network training.
- 120 pictures taken by myself for collision avoidance neural network training.
- Speed = 0.09, steering = 0.02
- Success

Object following + Stop when getting close trial #1 Failed

- Jetbot could only go round and round no matter what speed and steering setting.
- I spend most of the time to solve this issue

Object following + Stop when getting close trial #2 succeed

- A lot of people encounter the issues in trial #1, they are not able to make object following works on the jetbot
- Most people claim its the hardware limitation, the jetson’s processing power is not fast enough handle the data.
- After a lot of child handling, testing, tweaking on the variables and algorithm of the code, I verified hardware limitation is the reason behind this issue. Because the memory usage are always above 78 % and easily reach 100% while trying to locate the target amount other objects



Object following + Stop when close to the object trial #2

Solution:

1. Need 10W power mode to get the max performance
2. Following bigger object such as human for easier detection
3. Speed = 0.09, steering = 0.10

MOST IMPORTANT

4. Camera frame rate needs to be ≤ 10 fps
5. Limited the interference around the jetbot, otherwise, the jetbot will be busy on detecting objects and lost control for the locomotion
6. Jetbot will not able to handle the data and lost control without the settings above due to hardware processing limitation

Conclusion

- Computing power is very important for AI / autonomous driving
- Need enough / more than one cameras and other detecting input devices to cover the suddenings to achieve safety goal on autonomous driving.
- Need massive data to train the neural network to achieve more precise outputs, more data, more accurate.
- AI / autonomous driving need more regulation and rules from the government for public use.

Q & A

- Questions?