# 04_Convolutional Neural Network (CNN), Symbols detection

# 1. Summary

One of the goal of this project is to detect letters S = 'START', E = 'END' and digits from 1 to 5.

> For the moment we can't run a CNN on an AURIX™ (1G or 2G). That's why the CNN is implemented on a Raspberry Pi 3B+.

We are using :

- 1 Raspberry Pi 3B+ with the OS NOOBS v3.0.0
- 1 Camera Pi Module V2
- 1 Power supply of 5V-3A and 1 power bank 5V-2.1A / 10000mAh (this one will be used when the Raspberry will be on the robot) > You absolut have to respect it, even if the Raspberry will be under supply
- 1 Micro-SD Card of 64Gb with the FAT32 format

# 2. Raspberry Setup

## Firmware installation

(https://www.raspberrypi.org/help/noobs-setup/2/)

Download NOOBS from the Raspberry website Drag and drop all the content of the downloaded file on the micro-SD with de FAT32 format Plug your SD-card on the Raspberry and install the NOOBS

## Install python3 dependencies for the CNN

```
- Install Tensorflow 1.11 with pip3 (https://www.tensorflow.org/install/pip)
```

> Take care of which version you would like to use. The training is done with Tensorflow 1.11. You must have minimum Tensorflow 1.11 to run the CNN

```
- Install Opencv :
```

> pip3 install opencv-constrib-python opencv-contrib-python-headless
>
> sudo apt-get install libhdf5-dev libhdf5-serial-dev libjasper1

```
- Install dependencies :
```

> sudo apt-get install libatlas-base-dev
>
> pip3 install matplotlib pandas

```
- Enable UART pins and PiCamera
```

> sudo nano /boot/config.txt
>
> Add at the end of the file : "dtoverlay=pi3-disable-bt"
>
> sudo raspi-config
>
> Select option 5, Interfacing options, then option P6, Serial, and select No. Exit raspi-config.
>
> Got to the widget 'Raspberry Pi Configuration' and check 'Enabled' for the Camera and the Serial
>
> reboot

# 3. Training CNN

You have to follow this video for the training of the CNN : (https://www.youtube.com/watch?v=Rgpfk6eYxJA)

We are using the model : ssdlite_mobilenet_v2_coco (this model is dedicated for the Raspberry. We can't run a RCNN on the Raspberry because it requires a huge of processing). The dataset (about 7.000 photos) with the labeling of each pictures and the training are already done. Inside the

"CNN_folder" folder, 2 pythons files could help you for the datasheet: - rotate.py. To get twice more photos on you database, you can also rotate all of them ! - files_sorting.py. As described on the YouTube video, you will get 2 different folders (1 for the test and 1 for the training). This python program allows us to separate randomly all the pictures in 2 folders.

> All the CNN can be found on this github repository -> folder "CNN_folder"

# 4. CNN explanation

## RASPBERRY side

First, we have an UART link between the Raspberry and the AURIX™. When a number/letter is detected, the Raspberry sends this kind of packet to the AURIX™:

| **Header – Label** | | **Header – x coordinate** | | **Header – y coordinate** | | **Header – end of packet** |
|---|---|---|---|---|---|---|
| *'c'* | *'S', '1', '2', '3', '4', '5' or 'E'* | *'x'* | *value of its x coordinate* | *'y'* | *value of its y coordinate* | *'w'* |

Moreover, the function "Target_properties" of the file ../tensorflow2/models/research/object_detection/Oject_detection_picamera_V3.py allows us to find all the coordinates of each detected target. First, we should take a minimum probability of detection, here we got a threshold of 80%. If this condition is completed, the CNN gives the coordinates of the bounding box on the image. Then, trivially we can get the link between the image coordinates and the real coordinates, like this : - The camera has a field of view of 50cmx50cm (vertically x horizontally). Moreover, the camera is configured with this resolution 640x480 (width x height). So, 640 pixels represent 50cm and it's the same for 480 pixels.

As you can see on the video in the folder ../CNN_videos/CNN_raspberry.mkv of this repository, all the symbols are detected, and on the terminal each target information is displayed.

## AURIX™ side

On the AURIX™, more particularly on the file "cpu0_main.c", there is a state machine which takes all these information. The state machine decodes the packet sent by the Raspberry (shown above). Moreover, we configured a classic UART [Baudrate at 115200]. All the configuration is explained inside the (.c/.h) serial_Raspberry files.

All in all, when a complete packet is transmitted, the robot is able to achieve the target position. This is shown on the video in this repository in the folder ../CNN_videos/CNN_mode3.mov

# 5. Run the CNN on the Raspberry

If you have created a virtual environment before, launch this command :

> source ./CNN_venv/bin/activate

> cd Desktop/tensorflow2/models/research/object_detection

> python3 ./Object_detection_picamera_v3.py

# 6. Next steps

Port the CNN using Tensorflow on the next AURIX™ generation (the 3G). This one will include a CNN accelerator (the PPU). But, if you want, before that you can use 2 different boards with a CNN accelerator, which are : - sipeed : MAIXDUINO - sipeed : MAIX BIT