Installation and Operating Manual - AutoPylot

Group: Autonomobile

June 2022

Contents

1 How to setup the software

It is recommended to have python 3.6.X installed, as this is the python version installed on the car. To avoid any packages conflicts with your existing python installation, we will use virtualenv install virtualenv using:

```
pip install virtualenv
```

Clone the repo, and install the package and it's dependencies:

```
git clone https://github.com/Autonomobile/AutoPylot.git
cd AutoPylot
```

Then, create a virtual env (you need to specify the path to your python3.6):

```
virtualenv –python your path to python.exe venv
```

Then, every time you will be working on the project, you will need to activate this environment, to do so:

```
\venv\Scripts\activate.
```

Now, to install autopylot and its requirements (including dev requirements):

```
pip install -e .[dev]
```

For the code formatting, we will use something called "pre-commit", that enables us to automate stuff as linting before committing. If the code is not well linted, it will throw an error before committing and will lint it, you will only have to commit again to apply the changes the linter did ! Here is how to setup pre-commit:

```
pip install -e .[dev]
```

You are now all setup to work on the project ! Don't forget to keep the setup.py and requirements.txt up to date.

To exit the virtualenv:

```
deactivate
```

Useful tools:

- Setup a python linter (we use flake8) : https://code.visualstudio.com/docs/python/linting
- Setup the test extension of VS-Code : https://code.visualstudio.com/docs/python/testing
- Use a docstring generator for example the VS-Code extension "Python Docstring Generator"

2 How to setup the Telemetry Server

2.1 Installation

Installation:

```
npm i
or
yarn
```

Start Development:

```
npm run dev
```

Start Production Build:

```
npm run build
npm run start
```

then open http://localhost:3000/

2.2 Usage

Before starting:

In your settings.json of your autopylot client, change the following line with the host and port you need.

```
"SERVER_ADDRESS": "ws://localhost:3000"
```

Don't remove the ws:// or change the protocol to http:// as it will not work.

2.3 User Manual

- start the server with npm run start.
- open the client in a browser at http://localhost:3000/ or your custom address.
- start the autopylot client.

  if everything goes well, you should see a popup with a message telling you that a new car is connected.

- select the car with the dropdown menu in the top right corner.
- start tracking !

3 How to collect, train and deploy your model

Here is the three main steps in the making of a model

3.1 Collect

First you need to collect some data.

You will need to select in the "settings.json" the "CAMERA_TYPE", "ACTUATOR_TYPE" and "CONTROLLER_TYPE". By default they are set for the car configuration eg "webcam", "serial" and "xbox" respectively, if you are collecting data on a PC, I suggest you using "sim", "sim" and "keyboard". This will use data coming from the simulator, inputs from the keyboard and output them (actuate) in the simulator.

Start the script with the following command (don't need to cd if you are in the right directory):

```
cd main_programs/examples
python3 drive_with_controller.py
```

To drive depending on the controller type you use:

- "xbox": steering: left joystick, throttle: left and right triggers
- "keyboard": steering: "q" and "d" keys, throttle: "z" and "s" keys.

To record data depending on the controller type you use:

- "xbox": hold button "a"
- "keyboard": hold key "r"

You should see the collected data your " /collect" folder. If unsure about the location of the folder, check the "COLLECT_PATH" settings in the settings.json

Note: by default a model will be loaded, if you don't touch anything, you will enter "autonomous" mode using the predictions from this default model.

3.2 Training

There are plenty settings for the training script:

- "MODEL_TYPE": you need to set the type of the model you want to build. For example, if you made a new method to create a model in architectures.py in the Models class called "steering_model". You can set the "model_type" field to steering_model and when training your model, it will be created using this function.
- "MODEL_NAME": the name of the model you want to train, the model will be saved under this name. If you wish to retrain it later, make sure to use the same name (if you wish you can copy a model, change its name and then train it again to avoid loosing the previously trained one).

- "TRAIN_LOAD_MODEL": whether you want to load the model or create one from scratch.
- "TRAIN_BATCH_SIZE": How much data you want in one batch.
- "TRAIN_EPOCHS": How many times you want to train the model on the dataset before saving it.
- "TRAIN_SPLITS": Proportion of data in the training set and testing set. If set to 0.9, this will result in having 90 percent of the data going to the training set and 10 percent going to the testing set.
- "TRAIN_AUGM_FREQ": How frequent we want data to be augmented using data augmentation functions.

you can now start this training script with:

```
cd main_programs/
python3 train.py
```

Once finished, your new model should be saved in the "models" folder at the root of your project.

3.3 Deploy

You will need to change in the settings.json the "MODEL_NAME" field to the name of the model you just trained. then same as for the "collect" part, start the drive_with_controller.py script and enjoy!

```
cd main_programs/examples python3 drive_with_controller.py
```