Windows

If Anaconda3 is already installed (like on the TU Delft lab computers), you can proceed to step 3 directly.

1. Download the Anaconda3 Python 3.7 installer from this link
2. Install Anaconda3 Python 3.7. Choose the option to install for the current user only, in order to avoid needing admin permissions.
3. Download the practicum archive file from brightspace. Save the file in, for example `H:\Desktop\mpiv`.
4. You can extract the tar file by opening the `Start menu -> Command Prompt` and executing `tar -xvzf practicum1-<hash>.tar.gz`.
5. Download `data.tar.gz` from brightspace to your practicum directory (`H:\Desktop\mpiv`) and extract the file using `tar -xvzf data.tar.gz`.
6. Open **Anaconda Prompt (Anaconda3)** from the Start menu
7. In the terminal use `cd` and navigate to the practica materials directory.
8. Type `conda env create -f environment.yml`create the conda environment `mp-iv` with the packages defined in `environment.yml`
9. Type `01-activate.bat`. This will select the conda environment `mp-iv` and set the needed environment variables (`PYTHONPATH`)
10. Type `jupyter notebook`
11. Now open the notebook (`ipynb`) file of the practicum by selecting the file in the browser.
12. Happy coding!

Every time that we want to start to work again we have to: 1. Open **Anaconda Prompt (Anaconda3)** in the Start menu 2. Go to the practica materials directory (for example `cd Desktop/mpiv`) 3. Type `01-activate.bat` (this will also activate the environment mp-iv, that you see in parenthesis before the directory and set needed environment variables (`PYTHONPATH`) 4. Type `jupyter notebook` (this will open the notebook server from where you can open the practica and assignment notebook files)

Linux

The instructions were tested on Ubuntu 18.04.

Be aware that if conda is already installed on your system, the following steps may result in conflicts. Proceed to step 4 if conda is already installed.

1. Download **anaconda** 64-Bit (x86) Installer for Linux from their website. A file named `Anaconda3-2021.05-Linux-x86_64.sh` (or similar) should be downloaded.

2. Move the downloaded file to your folder and execute the file.

   mv ../Downloads/Anaconda3-2021.05-Linux-x86_64.sh ./ bash Anaconda3-2021.05-Linux-x86_64.sh

Accept the license terms by typing yes as in the following:

```
Do you accept the license terms? [yes|no]
[no] >>>  yes
```

Anaconda3 will now be installed into this location:

```
/home/$user$/anaconda3

Press ENTER to confirm the location
Press CTRL-C to abort the installation
Or specify a different location below

[/home/$user$/anaconda3] >>>

Press Enter and wait until conda is successfully installed.

Now initialize conda by typing yes as in the following:

Executing transaction: done
installation finished.
Do you wish the installer to initialize Anaconda3
by running conda init? [yes|no]
[no] >>> yes

==> For changes to take effect, close and re-open your current shell. <==
```

If you'd prefer that conda's base environment not be activated on startup, set the auto_activate_base parameter to false:

```
conda config --set auto_activate_base false

Thank you for installing Anaconda3!
```

Type the following and press Enter:

```
source ~/.bashrc
```

Now run the following command to update the conda packaging tool to the latest version:

```
conda update -n base -c defaults conda
```

The following packages will be UPDATED: conda 4.8.5-py37_0 –> 4.10.3-py37_0

```
Proceed ([y]/n)?
```

Press **Enter** and wait for the update process to finish.

Try the following commands to see if the installation worked for you.

```
conda --version
```

You should be able to see something like this: “conda 4.10.3”

3. Create the directory `~/mpiv` with `mkdir ~/mp-iv`.
4. Download the practicum archive and save it in `~/mpiv`. Navigate to the directory with:

   cd `~/mp-iv`
5. Extract the archive with `tar -xvzf practicum1-<hash>.tar.gz`.
6. Download `data.tar.gz` from brightspace to `~/mpiv` and extract the archive with `tar -xvzf data.tar.gz`.
7. Create the mp-iv Environment

   conda env create -f environment.yml

Next, activate the new environment:

```
source 01-activate.sh
```

Also try the following:

```
which python
```

You should be able to see an output similar to the following:

```
/home/$user$/anaconda3/envs/mp-iv/bin/python
```

8. Write the following command to open a jupyter notebook in your browser:

   bash 02-start-notebook.sh

You now should be able to run the first practicum

Please make sure, that you will have to activate the environment (mp-iv) you just created every time you restart your terminal window (or computer) by executing `source 01-activate.sh`. So do not forget to do this before you start working on your practica with `02-start-notebook.sh`.

Docker (on Linux)

You could also run the notebook in a docker container.

1. Create the directory `~/mpiv` with `mkdir ~/mp-iv`.
2. Download the practicum archive and save it in `~/mpiv`. Navigate to the directory with:

   cd `~/mp-iv`
3. Extract the archive with `tar -xvzf practicum1-<hash>.tar.gz`.
4. Download `data.tar.gz` from brightspace to `~/mpiv` and extract the archive with `tar -xvzf data.tar.gz`.

5. Build the `mp-iv` image with:

   docker build -t mp-iv .

6. Start the container using:

   docker run –rm -it -p 8888:8888 -v ~/mp-iv:/mp-iv mp-iv bash

7. Source the environment variables used in the practicum with:

   source 01-activate.sh

8. Start the jupyter notebook with `--allow-root`, since inside the container you are running it as the root user:

   jupyter notebook –ip 0.0.0.0 –no-browser –allow-root

9. Start a browser on the host (*not* inside the container) and navigate to the URL as shown in the terminal output. It will be something like:

   http://127.0.0.1:8888/?token=

10. You should now be able to open your practicum notebook file.

Target file structure

After unzipping the files of all practica and assignments, the directory tree should look like this (`data` directory is not shown)

```
.
+-- 01-activate.bat
+-- 01-activate.sh
+-- 02-start-notebook.sh
+-- 82-assemble-data-for-students.filelist
+-- 84-check-data-folder-structure.sh
+-- 90-docker-build-image.bat
+-- 90-docker-build-image.sh
+-- 92-docker-start-notebook.sh
+-- Dockerfile
+-- environment.yml
+-- git-hash
+-- installation_instructions.md
+-- installation_instructions.pdf
+-- pyproject.toml
+-- release
    +-- assignment
    |   +-- fa_00_overview.ipynb
    |   +-- fa_01a_data_visualization.ipynb
    |   +-- fa_01b_trajectory_and_ground_planes.ipynb
    |   +-- fa_02a_3d_pedestrian_detection_single_camera.ipynb
```

```
|   +-- fa_02b_mp_only_3d_pedestrian_detection_multiple_sensors.ipynb
|   +-- fa_03_3d_pedestrian_tracking.ipynb
|   +-- fa_04_iv_only_motion_planning.ipynb
|   +-- git-hash
|   +-- interfaces.py
|   +-- iv_only_ground_planes.json
|   +-- iv_only_ts_newworld_cam.json
|   +-- linear.png
|   +-- piecewise.png
|   +-- planning_visualization.py
|   +-- solution_helpers.py
|   +-- validation_metrics.py
+-- common
|   +-- git-hash
|   +-- k3d_helpers.py
|   +-- sequence_loader.py
|   +-- visualization.py
+-- practicum1
|   +-- activation_fns.py
|   +-- BoundingBox.py
|   +-- evaluation.py
|   +-- git-hash
|   +-- helpers.py
|   +-- ImagePatchClassifier.py
|   +-- ImagePatch.py
|   +-- image_processing.py
|   +-- load_data.py
|   +-- media
|   |   +-- animation_for_features_and_classifier.mp4
|   |   +-- coord_systems_diagram.svg
|   |   +-- iou_equation.png
|   |   +-- MLP_classifier.drawio.svg
|   |   +-- MobileNetv2_architecture.svg
|   |   +-- modern_view_of_ML.png
|   |   +-- overlapping_bboxes.png
|   |   +-- proposal_box_example.mp4
|   +-- metrics.py
|   +-- NeuralNetClassifier.py
|   +-- pedestrian_classifier
|   |   +-- saved_model.pb
|   |   +-- variables
|   |       +-- variables.data-00000-of-00001
|   |       +-- variables.index
|   +-- practicum1.ipynb
|   +-- preprocessing_fns.py
+-- practicum2
```

```
|   +-- check_kf.py
|   +-- compare_measurements.py
|   +-- copy_sensor.py
|   +-- define_occupancy_map.py
|   +-- evaluate_tracker.py
|   +-- funcs.py
|   +-- git-hash
|   +-- hidden_test_25.pkl
|   +-- images
|   |   +-- assignment_mot.pdf
|   |   +-- assignment_mot.png
|   |   +-- assignment_self-localization.pdf
|   |   +-- assignment_self-localization.png
|   |   +-- vehicle-pf.pdf
|   |   +-- vehicle-pf.png
|   +-- Measure.py
|   +-- Object.py
|   +-- pf_init_around_state.py
|   +-- pf_init_freespace.py
|   +-- practicum2.ipynb
|   +-- run_multi_object_tracker.py
|   +-- run_multiple_kfs.py
|   +-- run_pf.py
|   +-- run_single_kf_gating.py
|   +-- run_single_kf.py
|   +-- selfloc_scenario.py
|   +-- Sensor.py
|   +-- trackeval
|   |   +-- datasets
|   |   |   +-- _base_dataset.py
|   |   |   +-- __init__.py
|   |   |   +-- mot_challenge_2d_box.py
|   |   +-- eval.py
|   |   +-- __init__.py
|   |   +-- metrics
|   |   |   +-- _base_metric.py
|   |   |   +-- count.py
|   |   |   +-- hota.py
|   |   |   +-- __init__.py
|   |   +-- _timing.py
|   |   +-- utils.py
|   +-- visualizations.py
+-- practicum3_iv
|   +-- animate_parking_manoeuvre.py
|   +-- animate_steering_profile_plot.py
|   +-- determine_occupied_cells.py
```

```
|   +-- git-hash
|   +-- make_discrete_space.py
|   +-- motion_planning_types.py
|   +-- parking_scenario.py
|   +-- planning_problems.py
|   +-- plot_functionality.py
|   +-- plot_setup_groundplane_2d.py
|   +-- plot_setup_trajectory_planning.py
|   +-- plot_setup_vehicle_configuration_space_3d.py
|   +-- practicum3_iv.ipynb
|   +-- reconstruct_via_backtracking.py
|   +-- resources
|   |   +-- a_star_pseudo_code.png
|   |   +-- best_first_pseudo_code.png
|   |   +-- candidate_trajectories.png
|   |   +-- spline_steering_profile.png
|   +-- setup_trajectory_scenario.py
|   +-- trajplanning_obstacle_scenario.py
|   +-- vertices_reachable_in_n_steps.py
+-- practicum3_mp
    +-- git-hash
    +-- practicum3_mp.ipynb
```