Setup Step 2.2.2

Camera Calibration

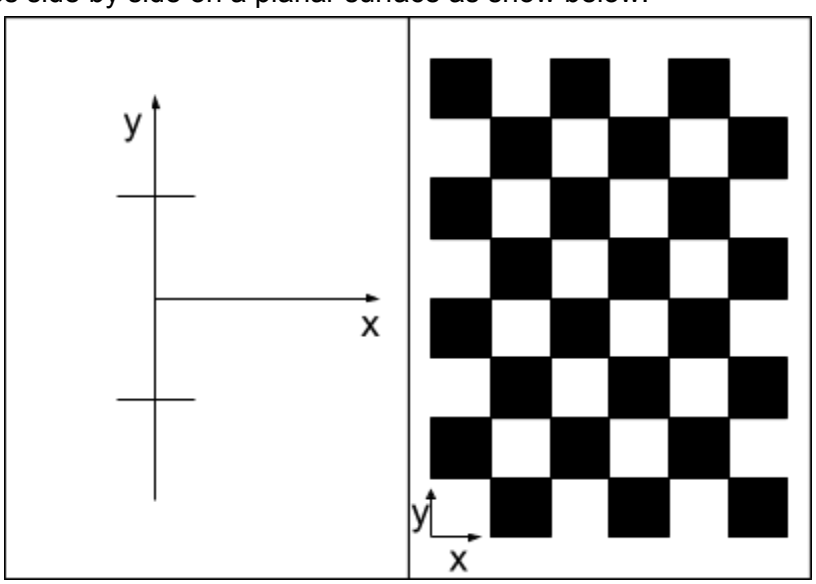
1. Setup

This document assumes that the car setup is done up to Step 2.1.

We will use a checkerboard to do both intrinsic and extrinsic calibrations.

Print calibration.pdf with US letter paper. When you print it, don't shrink or extend when print the pdf file. If it is properly printed, the side length of a black square should be 3.1 cm (about 1.22 in).

Put the two pages side by side on a planar surface as show below.



A table will be a good place. Fix them on the table top so that they will not move during calibration. Make sure that coordinate frames are conforming.

Install the `camera_calibration` node (http://wiki.ros.org/camera_calibration) on your external machine (e.g. laptop) if not installed

```
laptop $ sudo apt-get install ros-indigo-camera-calibration
```

2. Intrinsic Calibration

~~On your external machine, launch rosberrypi_cam node remotely~~

```
laptop $ roslaunch rosberrypi_cam remote_launch.launch veh:=veh_name
config:=baseline
```

~~If you list rostopics, you should see this~~

```
laptop $ rostopic list
/veh_name/rosberrypi_cam/camera_info
/veh_name/rosberrypi_cam/image_raw
...
```

~~With the previous running, to run camera calibration, run the following on your external machine~~

```
laptop $ rosrun camera_calibration cameracalibrator.py --size 7x5
--square 0.031 image:=/veh_name/rosberrypi_cam/image_raw
camera:=/veh_name/rosberrypi_cam
```

~~On **devel** branch:~~

Update your duckiebot:

```
duckiebot $ git fetch
duckiebot $ git checkout M04-PAI-<handle>
duckiebot $ cd ~/duckietown
Set environment
duckiebot $ source environment.sh
04
Re run make
duckiebot $ cd ~/duckietown/catkin_ws
duckiebot $ catkin_make
```

On your laptop:

```
laptop $ cd ~/duckietown/catkin_ws
laptop $ catkin_make
laptop $ roslaunch duckietown intrinsic_calibration.launch
veh:=${VEHICLE_NAME} raw:=true
```

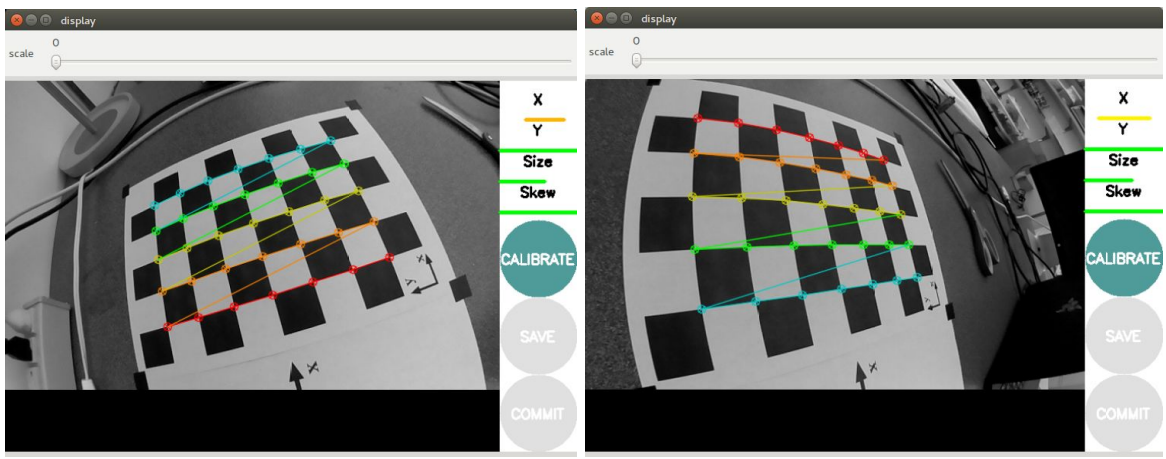
Note: do not run the above using byobu, because the calibrator needs to access the screen.

A window will pop up. Grab your Duckiebot and point the camera towards the checkerboard.

Move it around the checkerboard. If the camera sees all corners of the checkerboard, it automatically collects data. For good calibration, we need to capture diverse views.

As you move the car, you will see four bars on the upper right side increase. Each bar shows the observed range of the checkerboard in the camera's field of view.

- X bar: the observed horizontal range (left - right)
- Y bar: the observed vertical range (top - bottom)
- Size bar: the observed range in the checkerboard size (forward - backward from the camera direction)
- Skew bar: the relative tilt between the checkerboard and the camera direction



Once you collected enough images, all bars on the upper right side will be green and the 'CALIBRATE' button will be enabled. Press the 'CALIBRATE' button and it will do calibration. Depending on the number of images collected, it may take more or less a minute. During the calibration, the window might be greyed out; wait. After the calibration, it will show rectified video (i.e. undistorted video).

If you are satisfied with the current calibration, press the **'COMMIT'** button. It will automatically save to calibration file on your car (**not on your external machine**). The location of the file is: `~/duckietown/catkin_ws/src/duckietown/config/baseline/calibration/camera_intrinsic/${VEHICLE_NAME}.yaml`

Now let's push the`${VEHICLE_NAME}.yaml` file to git repository. Connect to your vehicle and push `${VEHICLE_NAME}.yaml` to git, i.e. do this

```
laptop $ ssh ubuntu@${VEHICLE_NAME}.local
duckiebot $ cd ~/duckietown
duckiebot $ git pull
```

```
vehicle $ git add
~/duckietown/catkin_ws/src/duckietown/config/baseline/calibration/cam
era_intrinsic/${VEHICLE_NAME}.yaml
duckiebot $ git commit -m "add intrinsic calibration file of
${VEHICLE_NAME}"
duckiebot $ git push
```

3. Extrinsic Calibration

Temporary step for Beta version:
The `ground_projection` node is still under development. It is developed in C++, and thus if there is change in the source code, you should re-compile the node. To compile catkin package, run `catkin_make` in the `catkin_ws` directory as:

```
$ cd ~/duckietown/catkin_ws
$ catkin_make
```

To run it, run Ctrl+C in the terminal launched `intrinsic_calibration.launch` and launch camera on your external machine (laptop):

```
laptop $ roslaunch duckietown camera.launch raw:=1
veh:=${VEHICLE_NAME}
```

And in another terminal, run `ground_projection` node with

```
laptop $ roslaunch ground_projection ground_projection.launch
veh:=${VEHICLE_NAME} local:=1
```

`$ rostopic list` should look like

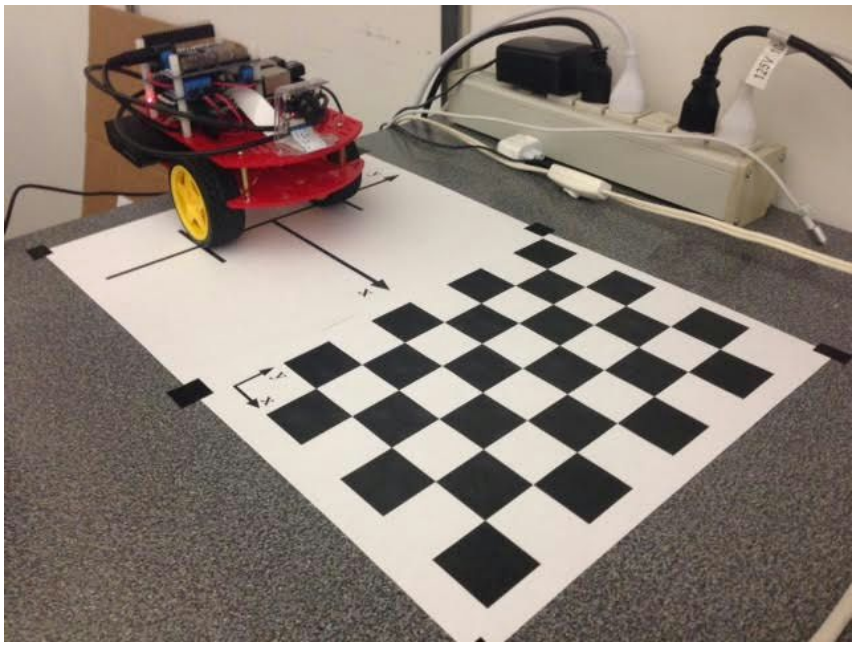
```
/<VEHICLE_NAME>/camera_node/camera_info
/<VEHICLE_NAME>/camera_node/image/compressed
/<VEHICLE_NAME>/camera_node/image/raw
/<VEHICLE_NAME>/ground_projection/lineseglist_out
/<VEHICLE_NAME>/line_detector_node/segment_list
/rosout
/rosout_agg
```

The `ground_projection` node has two services. They are not used during operation. They just provide a command line interface to trigger the extrinsic calibration (and for debugging).

```
laptop $ rosservice list
...
```

```
/veh_name/ground_projection/estimate_homography
/veh_name/ground_projection/get_ground_coordinate
...
```

To do extrinsic calibration, put your car as shown in the figure below



Note that the axis of the wheels is aligned with the y-axis.
IMPORTANT: put a uniform white wall behind the checkered board to block off any clutter.

The next step is to estimate a homography. Execute the following command:

```
laptop $ rosservice call /${VEHICLE_NAME}/ground_projection/estimate_homography
```

Once it estimates a homography, it automatically saves the estimated homography to `${VEHICLE_NAME}.yaml` file. The location of the yaml file should be **on your laptop** at

```
~/duckietown/catkin_ws/src/duckietown/config/baseline/calibration/camera_extrinsic/${VEHICLE_NAME}.yaml
```

Note that since we launch the ground_projection_node with local:=1, the node is running on your laptop. And the resulting **`${VEHICLE_NAME}.`**`yaml` file will be on your laptop too. Don't forget to get **`${VEHICLE_NAME}.`**`yaml` from your laptop to your vehicle since during operation we will run the ground projection node on the vehicle. You can by commit and push on laptop and pull on the vehicle.

To make sure if the estimated homography is correct, run the Testing in Part3 (method 3) in the Testing Plan document.