Lab 3: Vision

Robot Autonomy

Prof. Oliver Kroemer

0 Prerequisites

In this lab, you will calibrate the Azure Kinect Camera and use the calibration to pick up objects on the table. Then you will use aruco tags in order to make a stack of blocks.

0.1 Update Lab Repo

1. Run `git pull` on the lab repo `robot-autonomy-labs` on your machine.
2. Install the requirements in `requirements.txt` by going into your virtualenv (or conda env) first, then running `pip install -r requirements.txt`

0.2 Starting the Robot

Refer to the first lab for detailed instructions.

1. Turn on the robot.
2. Unlock the joints via launching a remote Firefox session on the Control PC.

0.3 Note on Safety

When running a program that commands the robot, please:

1. Stay outside of the robot's workspace (defined by the edges of the table).
2. **Always keep a hand on the e-stop. Under no circumstances should the e-stop be out of reach.**

1 Calibrating the Azure Kinect Camera

1.1 Roslaunch Azure Kinect ROS node

1. Remote launch the Microsoft Azure Kinect Camera ROS node on the Control PC by:

   `cd` to where `robot-autonomy-labs` is on your machine.

   Run `bash lab3/bash_scripts/start_azure_kinect.sh -i iam-<name>`

2. Make sure that the Azure Kinect is working by using rostopic list and opening the RQT Image Viewer on your computer using the command:

   `rosrun rqt_image_viewer rqt_image_viewer`

   Then select the `/rgb/image_raw` topic.

3. Once you have verified that you can view the camera images on your laptop, you will need to run the following command:

   `rostopic echo -n 1 /rgb/camera_info`

4. Then open the `lab3/calib/azure_kinect_overhead/azure_kinect.intr` file.

5. Copy the first number in K: to after ‘‘`_fx`’’:.

6. Copy the third number in K: to after ‘‘`_cx`’’:.

7. Copy the fifth number in K: to after ‘‘`_fy`’’:.

8. Copy the sixth number in K: to after ‘‘`_cy`’’:.

9. Make sure the ‘‘`_height`’’: and ‘‘`_width`’’: fields match.

1.2 Installing CV Bridge for Python3

Due to incompatibility issues with CV Bridge and Python3, you will need to compile `cv_bridge` using the following instructions.

1. Create a new catkin-ws somewhere.

   ```
   mkdir -p cv_bridge_catkin_ws/src
   cd cv_bridge_catkin_ws/src
   git clone https://github.com/ros-perception/vision_opencv.git
   cd ..
   sudo apt-get install python3-tk python3-empy
   ```

2. Use a text editor and open the file: `cv_bridge_catkin_ws/src/vision_opencv/cv_bridge/CMakeLists.txt` and change line 11 from `find_package(Boost REQUIRED python37) to find_package(Boost REQUIRED python3)`

3. Next deactivate your virtual environment and run the following commands:

   ```
   pip3 install rospkg numpy
   ```

4. Then run the following command on one line:

   ```
   catkin_make -DPYTHON_EXECUTABLE=/usr/bin/python3
   -DPYTHON_INCLUDE_DIR=/usr/include/python3.6m
   -DPYTHON_LIBRARY=/usr/lib/x86_64-linux-gnu/libpython3.6m.so
   ```

5. Then add the following line to your `~/.bashrc` file.

   ```
   source /path/to/cv_bridge_catkin_ws/devel/setup.bash --extend
   ```

   ```
   source ~/.bashrc
   ```

6. Finally navigate to the following directory:

   ```
   cd /opt/ros/melodic/lib/python2.7/dist-packages
   ```

   ```
   sudo mv cv_bridge cv_bridge_2.7
   ```

1.3 Installing Perception Utils

Next you will need to download a specially modified package into your `ws` folder.

1. Clone the following repository:

   ```
   git clone https://github.com/iamlab-cmu/perception.git
   ```

   ```
   cd perception
   ```

2. Then run the following command in your franka virtual environment.

   ```
   pip install -e .
   ```

1.4 Position the Robot

1. On the hand of the robot is a checkerboard that you will need to position in order to be visible to the robot as shown in Fig. 1. You may have to move the robot a bit in order for the camera to better recognize the checkerboard.

2. Run `python lab1/run_guide_mode.py` to move the robot into a similar position so that the checkerboard is visible by the camera or just push down on the e-stop to move the robot in place.

3. If you pushed down on the e-stop to move the robot, you will need to release it and then run the `start_control_pc.sh` script again.

4. Change directory into the lab3 folder.

   ```
   cd lab3
   ```

5. Run `python lab3/scripts/register_camera.py`.

Figure 1: Robot Placement

6. The calculated camera intrinsics will be located in `lab3/calib/azure_kinect_overhead/azure_kinect_overhead_to_world.tf`.

7. Compare the file `lab3/calib/azure_kinect_overhead/azure_kinect_overhead_to_world.tf` with `lab3/calib/example_azure_kinect_overhead_to_world.tf` and make sure the rotation matrices and translations are similar. If they are vastly different try rerunning the `lab3/scripts/register_camera.py` script.

2 Testing the Camera Intrinsics and Extrinsics

Next we have written a very simple script that will allow you to select a place for the robot to pick up an item using the newly found camera parameters. Put a block on the white surface and then run the following script.

1. First move the robot into a good location using the `python lab1/run_guide_mode.py` script so that when the reset arm command is executed, it won't hit anything.

2. Next run the following script: `python lab3/scripts/run_pick_up_using_camera.py`

3. After the robot has finished resetting, an image from the camera will pop up and select the center of the block. Then press enter on your keyboard. The robot should start moving. **Make sure to hold the Emergency STOP because the camera calibration might be flipped.**

4. The robot should pick up the block and then put it down again.

3 Install AruCo Marker Library

1. Run the following command `sudo apt install ros-melodic-aruco-ros`.
2. Run `roslaunch lab3/launch/aruco.launch`.

   You should be able to see the markers of the blocks if you run `rosrun rqt_image_view rqt_image_view` and listen to the topic `/aruco_simple/result`.
3. The poses of the tags will be published in the topics `/aruco_simple/pose` and `/aruco_simple/pose2`.

4 Put it all together

1. First, write a script based on the previous script that allows you to click 3 times in order to pick up and stack all three blocks.
2. After you have finished the first script, use the AruCo Markers on 2 of the blocks and one click to again stack all the blocks on top of each other.

   Please keep a hand on the e-stop while the robot is moving!!!

5 Turning Off the Robot

Turn off the robot as instructed in the previous lab:

1. Reset joints
2. Press down on the e-stop to return robot to manual mode (white)
3. Click shutdown on the web interface.
4. Wait until the robot shuts down, the flip the switch on the FCI.