Final Lab

Image Contour Detection and Drawing

1 Important

Read the entire lab before starting and especially the “Grading” section so you are aware of all due dates and requirements associated with the lab. In this final lab, there is no singular correct solution, students are encouraged to research and explore their options and implement whatever solutions they feel fits best.

2 Objectives

This is the final lab of the semester and for this lab you are expected to come up with your own method of completing this lab. You will use code developed in previous labs along with research you do into OpenCV and other image processing techniques.

The objective is to take an input image and have the robot draw the image on paper using a sharpie.

References

- Open CV: [OpenCV: OpenCV modules]

4 Procedure

4.1 Lab Set Up

1. Create a new catkin workspace as you did in Lab 2.
2. Download the **finalandDriver.zip** from the course website and extract into your catkin workspace **/src** folder. Inside /src, you should have two folders **finalpkg_py** and **drivers.** Compile the workspace using **catkin_make**.

If you look inside **finalpkg_py/scripts**, you will see a number of Python files, and we should only be editing a few, but feel free to edit or add any other files you may find useful. A brief description of each file is provided here:

- **final_exec.py:** The main executable. It contains all the code to move the arm. You will edit the main function to complete the drawing, but feel free to edit other parts of the script for your own usage (etc. helper functions, variables …).
- **final_func.py:** This contains the forward and inverse kinematics code. You should edit to add your solutions from Lab 5.
- **final_header.py**: This is a standard header file with imports and constant definitions. There is no need to edit this.

Note: A new version of the UR3 driver is provided to give you the freedom to control the robot with both **MoveL** and **MoveJ**, which were introduced in the first lab. You will note that the ***move_arm( )*** function in **final_exec.py** has an additional argument ***move_type*** compared to that of the previous labs, which is used to specify the two different move types. 'J' and 'L' are for MoveJ and MoveL respectively. Think about how using different move types will affect your drawing.

4.2 Workspace Setup

Ensure that the robots are equipped with an extra end effector designed to hold a sharpie. If it seems like this attachment is missing let the TA know.

Each lab station will be provided with a sharpie and a foam board to enable drawing on a flat surface.

Paper and tape to secure the paper in place will be available on the desk at the front of the lab room.

4.3 Running the Program

To run the code, open a terminal window and in your catkin working directory:

*$ **source** devel/setup.bash*

$ roslaunch ur3_driver ur3_driver.launch

Then open a second terminal window and in your catkin working directory:

*$ **source** devel/setup.bash*

$ cd src/finalpkg_py/scripts

$ python3 final_exec.py

Note: To run the python script this time, **rosrun** is not supported due to some settings of the package.

4.4 Implement the Code

The next step is up to you! Research, test, and explore different ways to get the UR3 robot to draw an image based on an input image.

Our recommendations:

1) Copy your inverse kinematics code into **final_func.py**

2) Implement Find Keypoints

- An empty function **find_keypoints( )** is given to you in **final_exec.py**
- Begin by loading the provided image using OpenCV.
- Investigate ways to detect lines in the image using OpenCV
- Convert the lines into key points

3) Transform Coordinates

- Fill out the empty **IMG2W()** function
- Transform your image keypoints into x and y coordinates in the world frame.
- Make sure that the coordinates lay within the bounds of the sheet of paper, consider scaling or rotating your image or coordinates.

4) Draw Lines

- Complete the empty **draw_image()** function
- With the coordinates, design an algorithm that draws a line between points.
- Imagine when drawing, how you would trace these points to replicate

the figure. For example, should you divide those points into multiple groups and draw each group one at a time?
- You have two options for robot movement: **move_arm(..., 'J')** for joint space movement, which moves each joint directly to a target position, and **move_arm(..., 'L')** for linear movement, which moves the arm in a straight line between points. Choose the movement type based on the desired level of precision and the shape of the path.

5) Optimize

- Try different methods to make your contours and keypoints detection more precise. Think about ways to preprocess the image to filter out irrelevant details. *(Hint: RGB image to grayscale?)*
- As you prepare to connect the detected keypoints in the 3D space, consider how to plan the path logically to avoid unnecessary movements that may affect the quality of your drawing.

4.5 Test

To test your code we provided 3 test images. Each one represents a different level of difficulty with the levels being easy, medium, and difficult.

Also try testing your own images. To make sure your code works for a variety of images, testing on a wide variety of inputs is important.

4.5 Debug

This lab involves some steps that will require trial and error to fine-tune. Here are some suggestions to help you navigate the process more smoothly:

- Read the OpenCV manual carefully and learn how to use it to extract keypoints.
- Test and visualize each step. Use visualization functions, like **cv2.imshow( )**, at each stage to check the output.
- Print key values for verification. Print important values, such as the coordinates of detected keypoints, to confirm they match your expectations

5 Report

You should submit a lab report using the guidelines given in the ECE 470: How to Write a Lab Report document. Please be aware of the following:

- Lab reports will be submitted online at GradeScope.
- Reports are due on Wednesday 12/18 at midnight.
- **No late submissions.**

Your lab report should include the following:

- A motivation for how the tools used in this project might be practically used and an explanation about why you chose to draw the picture of your choice
- Your solution / method to solving the problem with details on how you extract key points and contours (note that you'll be strictly graded on how generalizable your approach was)
- A discussion on how you evaluated your approach and evaluated your own progress.
- Also include your thoughts on what you thought of the labs, whether you enjoyed them or not and what you might change about them for the next semester. Include thoughts on the final lab (this lab) as well.

6 Demo

Your TA will give you the demo image and you will get only **one chance** to draw it. Please test your program on other images before attempting to demo. Demo points will be scored based on how well the robot is able to draw the image. The drawing is not expected to be perfect, but it is expected to resemble the image clearly. The difficulty of the image will be harder than the medium test image given, but easier than the hardest image given.

Then have your robot draw your competition image. You will get as many chances as needed to draw your competition image and the one you find most satisfactory will be submitted for the competition. The submitted competition image does not need to be drawn in the presence of a TA, but if it is drawn unsupervised the student must record a video of the robot drawing the image and present it to the TA along with the original image and the robot drawn image.

7 Competition

There will be a competition at the end of this lab to see which group will be able to create the best robot drawing.

The competition will be voted on and decided by your peers. We will hold a voting session on **December 13th from 9 am to 1 pm**. If this time does not work for you, please schedule a time with your lab TA to cast your vote. Students who show up to vote will automatically be given one extra credit point for attending.

8 Grading

General Grading

- 50 points, demonstration.
- 45 points, individual report.
- 5 points, attendance.
- 1 extra credit point, attending competition voting.
- Up to 3 extra credit points, winning the competition.

(Extra credit points will go to exam grades)

Demonstration Grading:

For the given image, you'll be graded on a score of 1-5, where a score of 5 denotes a clearly representative drawing, a score of 3 denotes a drawing that has key features that are identifiable, and a score of 1 is some marks on the page.

For the student chosen image, you will be graded on a score of 1-5, where a score of 5 denotes an impressive, complex, and interesting picture, a score of 3 denotes an image with an equivalent of two complex shapes with reasonable representation, and a score of 1 denotes some lines on the page. A zero will be given to any inappropriate images.

After a score of 1-5 has been assigned, the scores will be multiplied by 5 to reach a cumulative score of up to 50 points for the demonstration.