EE 428: Computer Vision

Homework 2: Magic Wand

Instructor: Jonathan Ventura

In this assignment you will implement code to detect a colored ball, estimate its 3D position, and draw a 3D cube over the ball. You could imagine using this technique to implement a 3D mouse or a VR game controller.

The starter notebook provides the intrinsic parameters of the camera and the radius of the ball (3 cm). You need to figure out how to calculate the (X,Y,Z) coordinates of the ball and draw a cube around it using your knowledge of pinhole projection.

To install required dependency `imageio` with video support, try either `pip install "imageio[ffmpeg]"` or `pip install "imageio[pyav]"`.

Code requirements

Your code must fill in the empty methods in the starter notebook `magicwand.ipynb`.

Notes on the methods to fill in:

- `preprocess_image`: Convert the image to grayscale and return the result of Canny edge detection (`skimage.feature.canny`).
- `detect_circles`: Detect circles in the edge map using the Hough transform (see this page). You can play around with the parameter settings until you find something reasonable. (Be aware that the scikit-learn Hough transform is unfortunately quite slow. It is okay to set `total_num_peaks=1` except when processing `red_and_blue.mov` where you should set `total_num_peaks=2`.)
- `calculate_ball_position`: Calculate the 3D position of the ball.
- `project`: Project a 3D point to 2D.
- `draw_line_3d`: Draw a projected 3D line -- this should make use of `draw_line_2d`.
- `draw_bounding_cube`: Draw a 3D bounding box around the ball -- this should make use of `draw_line_3d`.

At the end of the notebook you should have the following demos, using the interactive video display code from HW0:

- Show the result of `preprocess_image`
- Show the result of `detect_circles` and `calculate_ball_position` (use `draw_ball` to draw the ball on the image.) You should draw all balls detected, even if there is more than one.
- Show the result of `draw_bounding_cube`

Finally, concatenate all 3D ball positions detected across all frames into an array and show the result as a 3D scatter plot.

Test out your code on videos `red.mov` , `blue.mov` , and `red_and_blue.mov` .

Report

Provide a short explanation of your solution. Be sure to document any sources you used in preparing your code, including websites and AI tools.

Discussion questions

Once you have the ball tracker implemented and working, inspect the results and answer the following qualitative evaluation questions:

1. Does the ball detector seem accurate? Explain when it fails to detect the ball and hypothesize why it fails.
2. Would it be possible to correctly rotate the 3D box according to the ball's orientation? Why or why not? If not, how could we physically modify the magic wand so that we could calculate the correct rotation of the box?

Submission instructions

Submit your Python notebook `magicwand.ipynb` and report (PDF or docx). Please do not put them in a zip file, just submit the files directly.