Project 3 Report

Project 3 was an exercise in classifying objects using the varying tools related to binary images. We had to create a pipeline, where an image was thresholded, cleaned up using morphological filters, the regions were identified, the region of interest was selected, rotation/translation/scale-invariant data about that region was determined, and then that data was either saved to a database or compared to an existing database to classify the object as one of the already-known labels.

Application

Description automatically generated with medium confidenceA picture containing icon

Description automatically generated

Required images 1: two objects (mouse and mask) after simple thresholding. We used a process where we first blurred the image, then converted it to greyscale, then used a thresholding value where everything below that value was foreground while everything above that value was background.

A picture containing graphical user interface

Description automatically generatedIcon

Description automatically generated with medium confidence

Required Images 2: The same two objects after morphological filtering. We used a grassfire transform to count distance to the other classification, and then flipped the classification on everything less than the # of desired grows/shrinks. Specifically, we shrunk the foreground first twice in 4-connected space, then grew the foreground four times in 8-connected space, then shrunk the foreground twice again in 4-connected space, leaving the total number of grows and shrinks even, and alternating which connectivity was used for balance. We chose to shrink first because the background holes in some test images were small, then chose more sensitive connectivity for the foreground growing because we shrunk first so we wanted to be extra sure we removed foreground holes during the growing.

A picture containing logo

Description automatically generatedA picture containing icon

Description automatically generated

Required Images 3: Selected regions. We used OpenCV functionality for this part, instead of manual implementation, and we colored the three largest regions red, blue, and green respectively to show them off. If there were more than 3 regions, they’d be colored in shades of grey.

A picture containing graphical user interface

Description automatically generatedA picture containing text

Description automatically generated

Required Images 4: Selected region with bounding box, as well as two printed HuMoments to the side. We used the largest region that was “central” (i.e. its centroid was within the inner 50% of the image) for automatic region selection. The seven HuMoments we used are translation, scale, and rotation invariant moments that can be used to classify shapes.

There’s a paper I skimmed to get a basic understanding of HuMoments, but most importantly they’re rotation, translation, and scale invariant, and are calculated as follows:

Text, letter

Description automatically generated

(HuMoment calculations, Huang and Leng)

(The terms they’re based on are normalized central moments)

The training system operates by pressing “5” on the keypad, which saves the HuMoments of the current largest central region to the next line in a database file called data.csv. It then asks the user for a label to assign to that image, which is places next to the HuMoments. Later on, when classifying, it will refer back to this database.

A picture containing text

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface

Description automatically generatedA picture containing graphical user interface

Description automatically generatedA picture containing text, electronics

Description automatically generatedGraphical user interface, text

Description automatically generated

A picture containing text, electronics

Description automatically generatedText

Description automatically generatedGraphical user interface

Description automatically generatedA picture containing text

Description automatically generatedText

Description automatically generatedLogo

Description automatically generated with medium confidence

Required Images 5 (task 6): Classified images. The labels are clearly printed on the screen. One interesting quirk was that the sharpener was very finnicky and hard to identify- it had a high surface reflection and cast awkward shadows that with a limited lighting setup were hard to eliminate. Also, the object labeled “clip” is correctly identified by the system, but perhaps a better-named label could’ve been chosen, in hindsight. It’s not much of a clip but more of a hook.

For part 7, we chose KNN matching with K = 5, as reflected in our confusion matrix.

**//insert confusion matrix here once extensions have been added to it**

Required Image 8: Confusion matrix (accounting for extensions)

Part 9: demo video can be found at the following link: <https://drive.google.com/file/d/1fyJXKAPOUb0AAJ4a4BAsvOk1OB2K-uhr/view?usp=sharing>

Graphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface, text, whiteboard

Description automatically generatedExtensions: We included “unknown” classification by saying if the closest match is greater than some threshold in distance (we landed on 1.7), then the object is unknown. We also expanded our database to 20 images (as reflected earlier with the classified images and confusion matrix). Some examples of objects not in the database can be seen below:

A pair of scissors

Description automatically generated with medium confidence

Reflection: This project gave good practice with lots of different binary image techniques, through the whole pipeline from thresholding to region analysis. One issue that came up surprisingly often during the testing was reflection problems. Even dark objects can get annoying surface reflection, and getting a proper lighting setup to mitigate that is rather difficult. Using dark, typically black objects on a white background is great for most thresholding but it makes lighting-invariant spaces unhelpful for this type of application. Shadows have also been an issue, especially for objects with relatively large Z-dimensions compared to their X/Y area, such as the pencil sharpener. Additionally, this was the first project either of us worked in a group, which brought some of its own challenges with scheduling, assigning tasks, and alternating between windows and mac operating systems. In an ideal world with infinite time, implementing parts 3 and 4 from scratch would have been a very valuable exercise, however this assignment coincided with midterm season for both of us, and Dhruv was still catching up since he joined the class late, so we were under significant time crunch. Regardless, this assignment was still a very valuable learning experience and taught us both a lot about binary image manipulation, characterizing 2-d regions, and some of the more hands-on difficulties that can come alongside these endeavors such as imperfect lighting environments and non-ideal target objects.

References/acknowledgements:

Huang, Zhihu & Leng, J.. (2010). Analysis of Hu's moment invariants on image scaling and rotation. Proc. of 2nd International Conference on Computer Engineering and Technology (ICCET. 7. V7-476 . 10.1109/ICCET.2010.5485542.

OpenCV documentation for various OpenCv functions were referenced.

Bruce Maxwell’s class/lecture notes were frequently referenced.