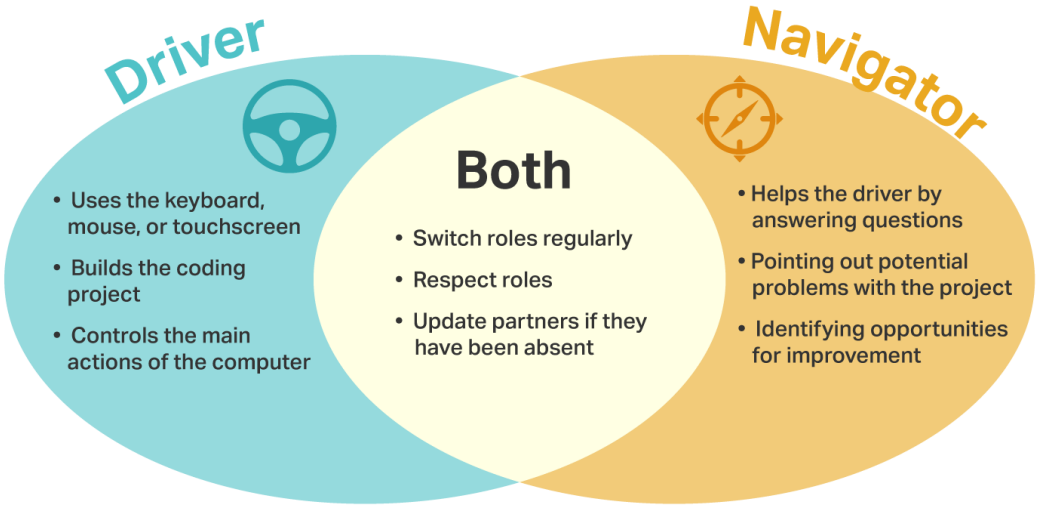
Assignment 5 - Chamfer Distance

Start Assignment

Due Oct 5 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip
Available Sep 28 at 2pm - Oct 8 at 11:59pm 10 days

This assignment will be done in pairs of two students using a pair-programming approach

Pairs of students should alternate between two roles: a 'driver' and a 'navigator' and switch roles for each task in the assignment. The 'driver' uses the keyboard, mouse, or touchscreen and controls the main actions of the computer. The 'navigator' helps the driver by guiding them on how to implement the solution at an algorithmic level, answering questions, pointing out potential problems with the code, and identifying opportunities for improvement.



The pairs have been generated at random and can be found under People > Assignment 5 - Groups. Student emails can be found in **Student_emails.txt** (https://canvas.txstate.edu/courses/1798576/files/203624183/download?download_frd=1)

It is important that partners arrange a time when they are both available to work together either in-person or remotely using screen sharing so that both users can see the screen at the same time.

A cloud file sharing service like Matlab Drive can optionally be used for sharing the files of the project between the partners.

Only one of the group members needs to submit the assignment. Both members will receive the same grade unless one of the members reports that the other member did not equally contribute to the solution.

Tasks:

-

-

Figure 1. Examples of color image (top), corresponding edge image (middle), and template (bottom), on which `chamfer_search` and `skin_chamfer_search` can operate. Note that the color image is not used for `chamfer_search`, which only uses edges, no color information.

Task 1 [50 pts]: Implement an algorithmically efficient version of object detection using the chamfer distance. In particular, implement a function

```
[scores, result_image] = chamfer_search(edge_image, template, scale, number_of_results),
```

where:

- `edge_image` is the image (2D matrix) you want to search,
- `template` is the pattern (2D matrix) you are searching for,
- `scale` A scale s means that the template size must be multiplied by s in order to match the occurrence of the object in the image.

- `number_of_results` specifies the number of results that will be displayed on `result_image` (note: `result_image` is an output argument).
- `scores` is a matrix of size equal to the size of the image, and scores(i,j) is the directed chamfer distance from the template to a window centered at (i, j).
- `result_image` is a copy of `edge_image`, with white bounding rectangles drawn for the best matches found during the search. The number of bounding rectangles to be drawn is specified by input argument `number_of_results`. The centers of these bounding rectangles are simply the pixel locations in `scores` (which is the first output argument). Note: two functions for drawing rectangles on an image can be found in "Code\00_common\00_utilities".

Your function will be graded based on the correctness and efficiency of the implementation. As usual, feel free to use any functions that Matlab already defines (such as `bw_dist`), or code posted on the course web page.

Task 2 [50 pts]: Implement a function called `[scores, result_image] = skin_chamfer_search(color_image, edge_image, template, scale, number_of_results)` that, in addition to your solution to Task 1, also uses skin detection to improve results, for cases where we are interested in detecting hands. An example of a test image, corresponding edge image, and template, that can be passed to this function, is seen in Figure 1. Feel free to pass additional input arguments to the function, if you need to.
Your solution will be graded based on correctness and efficiency. In particular, think carefully about how to combine skin detection and the chamfer distance so as to get a performance that is significantly faster than your solution for Task 1.

How to submit

Submissions are only accepted via Canvas. The submission should include, as an attachment, a zip file containing your Matlab code and a README.txt file. The zip file should be named `NetId1_NetId2_Assignment5.zip`, with no spaces or other extra characters. Your solution should definitely contain:

- A Matlab file called chamfer_search.m that implements the solution for task 1.
- A Matlab file called skin_chamfer_search.m that implements the solution for task 2.
- In addition, your zip file should contain ALL files needed to run the code. If you are using files that are posted on the course web page, you still have to include them in your zip file. At least 5 points will be taken off otherwise.
- The README.txt file should contain the names and Net IDs of the students and specify who was the driver and who was the navigator for each task, in addition to any additional comments/instructions

useful for running the code and understanding the underlying ideas. Also, the README file should explicitly discuss the running time for Task 1, and the running time for Task 2, on the test case of Figure 1.

Additional resources for assignment:

clutter1.bmp (https://canvas.txstate.edu/courses/1798576/files/203623179/download?download_frd=1)

clutter1_edges.gif (https://canvas.txstate.edu/courses/1798576/files/203623183/download?download_frd=1)

template.gif (https://canvas.txstate.edu/courses/1798576/files/203623177/download?download_frd=1)

positives.bin (https://canvas.txstate.edu/courses/1798576/files/203623187/download?download_frd=1)

negatives.bin (https://canvas.txstate.edu/courses/1798576/files/203623181/download?download_frd=1)

sample_result-top5-with_scale_1.jpg (https://canvas.txstate.edu/courses/1798576/files/203623185/download?download_frd=1)