ENPM673 Perception for Autonomous Robots (Spring 2024)

Project 2

Rohan Maan, Jay Prajapati, Samer Charifa, Tommy Chang

Submission guidelines:

- This homework is to be done and submitted individually.
- Your submission on ELMS/Canvas must be two files:
    a. Google Colab (.ipynb) file
    b. Google Colab (.pdf) file (Convert the same .ipynb file to a .pdf file)
    c. Video(.mp4) file

  following the naming convention ***YourDirectoryID_project2***.
- If your email ID is **abc@umd.edu** or **abc@terpmail.umd.edu**, then your Directory ID is **abc**.
- Submit both the files in one attempt.
- Provide detailed explanations for each step using text cells in Google Colab.
- Ensure the code is well-formatted for readability.
- Comment on each section (preferably every 2 lines) of the code to explain its purpose and functionality.
- Include relevant output images within the text cells where required.
- Include explanations to illustrate the results effectively.

Problem 1: [50]

Link to input video: https://drive.google.com/file/d/1wV4QV3OzrNAsK4ObwwsJTZAD2TzmSsRr/view?usp=sharing

Design a video processing pipeline to find the 4 corners of the paper using Edge Detection, Hough Transform and Corner Detection. Overlay the boundary(edges) of the paper and highlight the four corners of the paper. The boundary(edges) of the paper and highlight the four corners of the paper. (Note: that you must remove any frames which are blurry by using Variance of Laplacian and report the number of blurry frames removed).

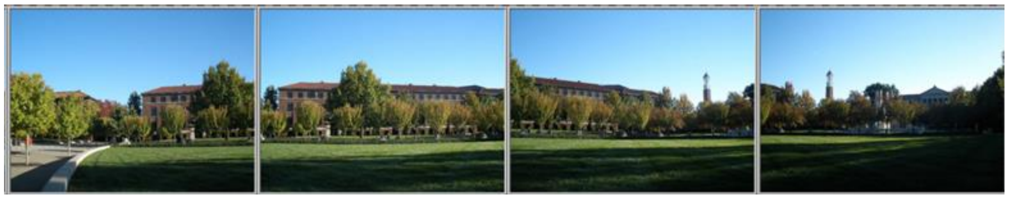
Example Pipeline:

1) Read the video and extract individual frames using OpenCV.
2) Skip blurry frames (use **Variance of the Laplacian** and decide a suitable threshold)
   Note: Any value below 150 for the Variance of the Laplacian, suggests that it's a blurry image.
3) Segment out the unwanted background area (example: convert to gray scale and keep white regions)
4) **Detect edges pixels** in each frame (you can use any edge detector)
5) Use the detected edge pixels to extract straight lines in each frame (hint: use Hough Transform)
6) Filter out "short" lines and only **keep a few dominant lines**.
7) Compute the **intersections** of the Hough Lines – these are the putative corners of the paper.
8) **Verify** the existence of those corners with **Harris corner detector**. (use OpenCV built-in function)
9) Filter out remaining extraneous corners that are not the 4 corners of the paper.
10) Generate the output video in which you have to overlay the **4 blue boundary lines** and 4 red corners of the paper in each frame (excluding the blurry frames).

Problem 2: [50]

Link to the images: https://drive.google.com/drive/folders/11iuYdIaysz6fyV0gIO-v6VST5daBm9Sn?usp=drive_link

A) Given four overlapping images of a far-way building taken from the same camera (images may be taken with both rotation and slight translation). Design an image processing pipeline to stitch these images to create a panoramic image. (Note: You can consider far away objects/features to be approximately on the same plane.)


(Note: These are the given four images in sequence. It is not the output image.)

B) In general, why does panoramic mosaicing work better when the camera is only allowed to rotate at its camera center?

Example pipeline:

1) Extract features from each frame (You can use any feature extractor and justify).
2) Match the features between each consecutive image and visualize them. (hint: Use RANSAC)
3) Compute the homographies between the pairs of images.
4) Combine these frames together using the computed homographies.

Guidelines:

1) You are not allowed to use cv2.createStitcher.
2) Any function that can solve the stitching problem directly is not allowed.
3) You are not allowed to use the 'imutils' package.
4) You can use OpenCV and NumPy inbuilt functionality.
5) You can use any feature extractor of your choice and be sure to provide justification for your selection.

****