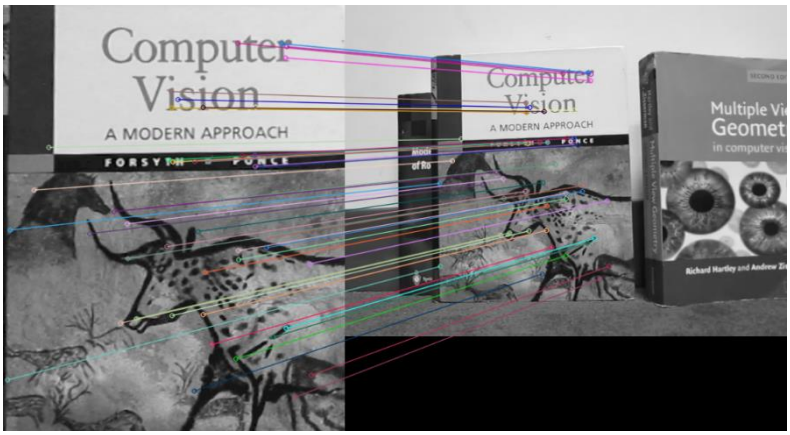
Part 1 : Augmented Reality Book Tracking and Movie Overlay

This project implements a simple augmented reality pipeline that overlays a movie onto a book detected in a video stream. Using feature detection, homography estimation, and image warping, the content of one video (a movie) is dynamically projected onto the cover of a book as it appears in another video.

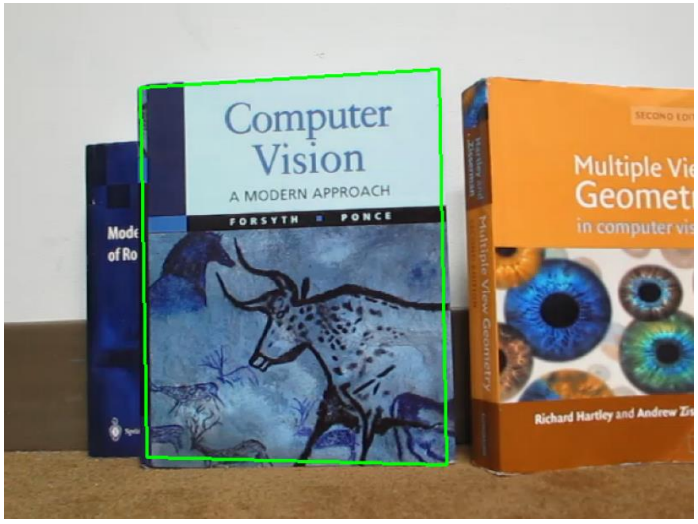
Algorithm Explanation

1. **Feature Matching with SIFT :** We use Scale-Invariant Feature Transform (SIFT) to detect and describe local keypoints on both the reference book image and the current video frame.
2. **Lowe's Ratio Test & Match Selection:** We apply Lowe's ratio test to filter good matches and select the top 50 most reliable ones.



3. **Homography Estimation** Using the matched keypoints, we estimate a homography matrix H that transforms points from the book image to the current frame.

4. **Book Localization:** With the homography matrix, we warp the corners of the reference book image to locate it in the frame.



5. **Movie Cropping and Overlaying:** To maintain aspect ratio and remove top/bottom artifacts from the movie, we crop it accordingly before projection. The cropped movie frame is warped to match the shape and perspective of the detected book using another homography, then blended into the video frame.

We then looped over both videos and performed these steps to produce the final overlayed video.

Part 2: Image Mosaics

The goal of Part 2 is to implement an image stitcher that automatically creates a mosaic by warping one image to align with another using homographies. This creates a combined, seamless view of the images.

Algorithm Explanation

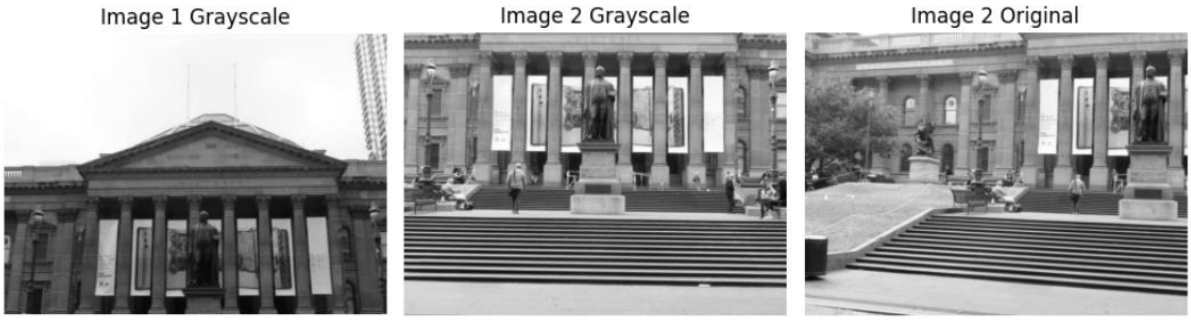
1. **Feature Matching with SIFT**
2. **Lowe's Ratio Test & Match Selection**

3. **Estimating optimal H parameters using RANSAC:** Similar to the first part we estimate the H function, however, here we use the RANSAC algorithm to get the best H that warps the points from one image to the other with the highest number of inliers.
4. **Normalize Intensities:** Using the key points and the descriptors, we iterate over key points, and get a patch of a set size around the key point, then we calculate the adjustment factor needed by dividing the intensities, and normalize one image to the other.
5. **Warping one image into other:** Using the estimated homography using RANSAC, fix one image into the middle of the output canvas, then using inverse warping and interpolation warp the second image onto the first image. We apply a mask to prevent overwriting of regions occupied by the first image.
6. **Cropping output canvas to least occupied rectangle**

Part 3: Mosaic of 3 images

This part isn't different from part 2. We just implement the previous steps 2 times. Stitching first images together, then stitching their output with the third image. The order of stitching makes a much larger difference here, resulting in much more different results.

Input pictures:



First Mosaic:

Final Mosaic of three: