Matthew Borkowski 11/15/2023
201588010

CP467: Assignment 4 Report Content

Algorithms Used:

Face and Car Recognition: The functions faceRecognition() and carRecognition() apply Haar Cascade classifiers for detecting facial features and cars. These classifiers are object detection methods that rely on machine learning and use positive and negative image samples during training. Additionally, both functions use the Scale-Invariant Feature Transform (SIFT) algorithm for identifying keypoints in images, enabling feature matching and object recognition.

Image Manipulation: The imageManipulation() function exemplifies image blending through Poisson blending. It reduces the saturation of a figure image and smoothly blends it into a background image using the cv2.seamlessClone() function. Saturation reduction involves converting the image to the HSV color space and scaling down the saturation channel to better fit the background image.

Image Stitching: The imageStitching() function employs SIFT feature matching and homography estimation to stitch two images into a panorama. It involves extracting SIFT keypoints and descriptors, matching them through FLANN-based matching, filtering good matches with a ratio test, and estimating the homography matrix.

Problems Encountered:

For the vehicle and face identification methods, I initially struggled with the implementation of the detectMultiScale parameters. The default values were far too sensitive and required from tweaking. I then encountered a very similar issue with the SIFT algorithm. Far too many key points were being identified in the image. My desired output was just to have the outlines of the people (included facial features) for the first image. For my second image I wanted the vehicles and the road barriers to be outlined. I tried to minimize the clothing and foliage being identified in both images the best I could with the contrastThreshold and edgeThreshold parameters; if I were to continue working on this in the future, further adjustment will be required.

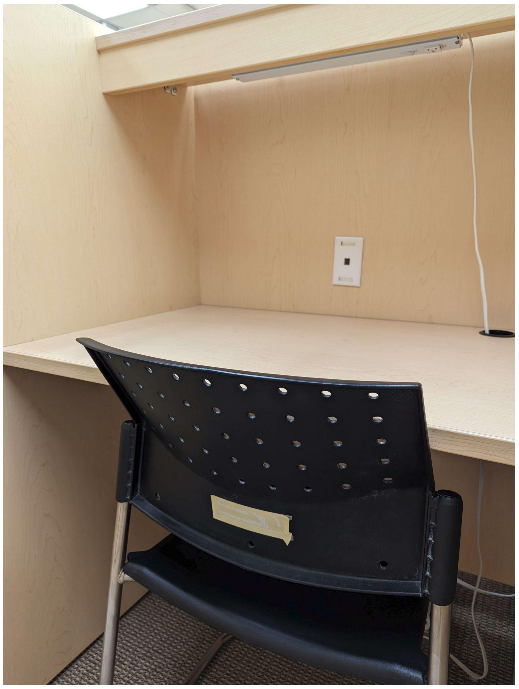
As for the second question, I did not struggle with much of the implementation until it came to the color saturation of the final image. Using MIXED_CLONE for the blending parameter was producing the best output however, The LEGO figure was far more saturated than the background. My best solution was to convert the image to HSV and downscale the saturation factor, then do the blending.

Matthew Borkowski 11/15/2023
201588010

CP467: Assignment 4 Report Content

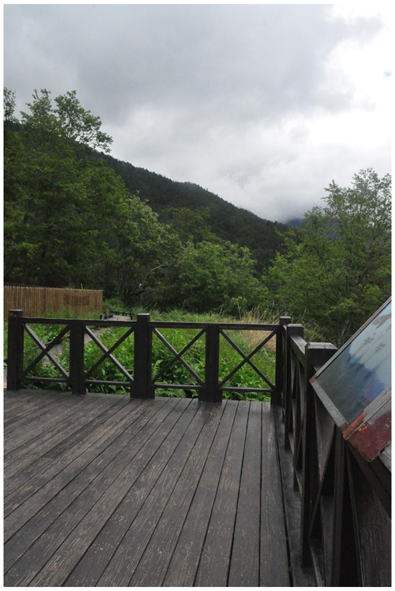
In dealing with the SIFT algorithm and homography estimation for image stitching, I faced some challenges. The SIFT algorithm generated too many keypoints, making it tough to get the desired level of feature matching. This surplus of keypoints resulted in inaccuracies during homography estimation, messing up the alignment of the images in the panorama. Despite trying to adjust parameters like contrastThreshold and edgeThreshold, the algorithm's sensitivity stayed high, needing more tweaks. Creating a seamless panorama turned out to be a tricky task, and refining the balance in feature detection and matching will be an ongoing process for future work.

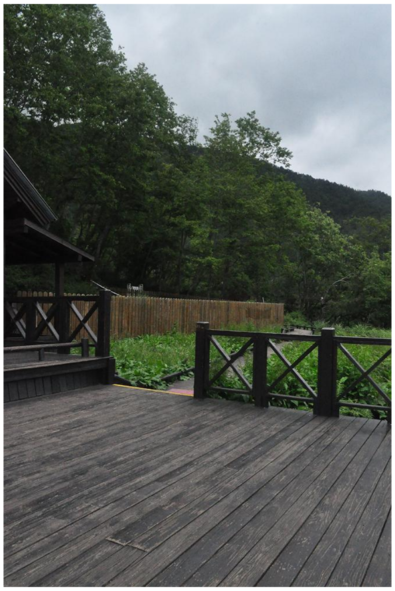
Input Images:



Matthew Borkowski 11/15/2023
201588010

CP467: Assignment 4 Report Content





Output Images: