Student Name: **Andy Xie** Student ID#:**1004102650**

Project Title:

Producing a 360 degree panorama from a series of omni-directional camera images with minimal effects of photo vignetting, distortion, and segmentation.

Project Problem Definition:

Problem to Address

The problem is almost identical to the original proposed. The goal of the project is to investigate whether if it's possible to create a 360^o panorama of the surrounding environment from a series of 5 or more omnidirectional rover camera images, while minimizing the effects of vignetting, distortion, and segmentation. I want to experiment with different feature detection algorithms such as SIFT and SURF (mainly SIFT), as well as experiment with different homography techniques, such as DLT homography and RANSAC point selection, to see if they can help me warp the photo features together neatly.

Changes to Original Problem

There hasn't been any drastic changes to the original problem proposed in the proposal. However, in light of the results, it may be appropriate to change the objective to producing a panorama that is at least 3/5 of 360 degrees (216 degrees) with minimal effects of vignetting, distortion, and segmentation. This is because image stitching and proper warping was a lot more complex than initially thought. The impurities within the image such a blurring, noise, and poor lighting made it difficult to smoothly stitch images together. Therefore, we might want to reduce the ambition of this project.

Relevance

The vast distance between Earth and Mars would mean any interplanetary rover would need to analyze its surroundings for any points of interest, detect obstacles, and plan a courses autonomously. The ability for the rover to take images and analyze them can help greatly with these particular functionalities. Creating a 360^o panorama with images from the omnidirectional cameras can help reduce memory size and computational complexity of terrain feature detection.

I have been very interested in the field of computer vision, and have been keeping up with the development of LiDAR, Curiosity, Spirit, and SpaceX and self-driving cars since high school. These topics have me wondering how vision and recognition technologies can be implemented in the next generation of space robotics.

Project Methodology:

Project Layout

The project involves a few key components to make it work. The first would be an algorithm that can detect common features between consecutive photos. This was done with the SIFT algorithm (Scale Invariant Feature Transform). We decided to try projecting the common features between two rover images using a homography transform as we were taught in the course. This would mean calculating a homography matrix that would allow us to transform between the coordinates of the two images. We have decided to use RANSAC and DLT homography for this purpose. RANSAC would allow the algorithm to eliminate the effect of outliers, which can reduce segmentation. The project would also need a way to warp the images together that avoids distortion towards the edges of the panorama and vignetting between the images. This is done through calculating the relative warping/projection homographies between other images not connected the center image and the center image of the panorama. Finally, the stitching of the warped images are done via OpenCV.

Student Name: **Andy Xie** Student ID#:**1004102650**

Algorithms Used

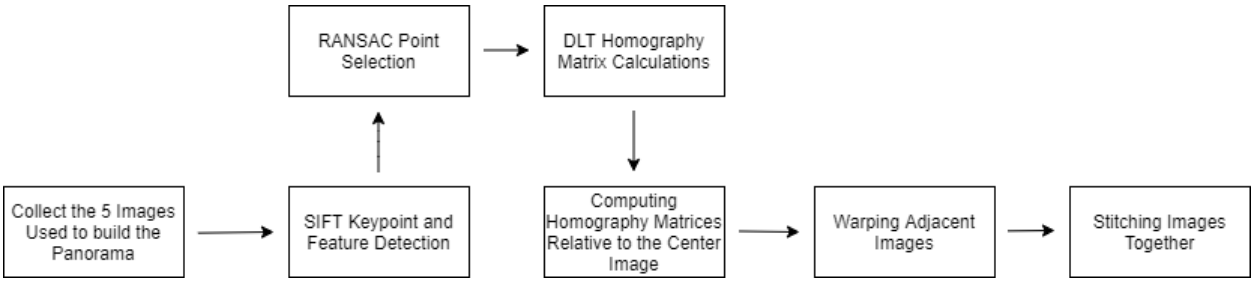
SIFT – SIFT is a feature detection algorithm that uses 5 main steps: Scale-space extrema selection, keypoint localization, orientation assignment, keypoint descriptor, and keypoint matching [1]. Scale-space extrema selection allows you to isolate key features in the images. This is done through Gaussian blurring of the input images to different scale spaces. These scale spaces are divided into octaves and Gaussian blurred before Difference of Gaussian (DOG) is calculated for each octave. DOG is used to find the optimal feature in the image [1]. Keypoint localization is done to reject flat and edge features. Orientation invariance makes the accepted keypoints rotation invariant as well scale invariant [1]. Keypoint descriptors are used to calculate gradients around the features to make them all unique [1]. Finally, keypoint matching matches features with similar gradients with each other between images [1]. This can be found in *feature_detection.py*.

RANSAC – Random Sample Consensus (RANSAC) is a general parameter estimation approach to accurately predict the fit of a model with a significant amount of outliers in the data. This is useful since SIFT often mismatches some of the more ambiguous features and you don't want them to be accounted for in the homography transforms. In this case, we take 4 random pairs of matching features and try to define a homography matrix with them. We would then determine how many of the matched feature pairs would be accommodated by the homography [2]. This is determined by an error threshold and a feature percentage threshold [2]. If enough matching features are accounted for within the error threshold, we can calculate the final homography matrix with those inlier features [2]. The feature percentage threshold is 75% in this case. This can be found in *homography.py*.

L2 Norm – The L2 norm is used to calculate the geometric distance between the actual feature points of the second image and the coordinates the homography matrix projected from the matching features of the first image. This is used as a benchmark for determining which feature points are outliers and which are inliers. We set this error threshold to 5. The feature percentage threshold is 75% in this case. This can be found in *homography.py*.

DLT Homography – This is the calculation that is specifically used to calculate the homography matrix for transferring between two images. The process is explained in the Dubrofsky paper [3]. The feature percentage threshold is 75% in this case. This can be found in *homography.py*.

Methodology



After collecting the 5 images we need to produce the panoramas, we would use SIFT on adjacent images to identify identical keypoints and features between the two images. Between each pair of adjacent images, we conduct RANSAC feature point selection where we randomly selection 4 feature points to compute the DLT homography matrix. After getting the final homography matrices between the images, we compute the homography matrices that transforms features from other images to the center image of the panorama to avoid distortion in the final result after warping. Finally, we use OpenCV's software to warp and stitch the images together.

We choose RANSAC since it was a computationally inexpensive way of ignoring outliers when computing accurate homography transformations. L2 Norm distance as a metric for inliers and outliers made sense since they can be easily computed on the images and they geometrically make sense. DLT algorithm was

an easy and inexpensive way to calculate the homography matrices. Lastly, SIFT was very convenient to use with OpenCV given the time frame and resources available, as does warping and stitching.

Libraries

Numpy – Scientific computing library. This library was used for making vector and matrix algebra computations, as well as providing a data type that is much faster to compute than the Python standard. This was used to calculate homography matrices in the DLT algorithm, inverse homography matrices, and the matrices used to project features between images that avoid distortion and vignetting. Version 1.19.2 is used for this project.

OpenCV – Computer Vision library. OpenCV was used to implement the SIFT algorithm for feature detection on the images, warping features from one image to another, and stitching the images together. Versions 4.4.0.46 and 3.4.2 were used for this assignment.

Data

For this project, we used images from the CPET dataset. The specific part of the data I am planning on using is the human-readable (base) data of Run 3 and Run 5 [4]. The rover run that we will demonstrate in the code is that of Run 3, stored in *run3_base_hr*. The specific folders that I am planning on using are:

omni_image1, *omni_image3, omni_image5, omni_image7, omni_image9* – Colored omnidirectional images from the rover cameras.

omni_stitched_image – Panoramic images created from the omnidirectional images by occam software.

The last image from the image folders *omni_image1*, *omni_image3*, *omni_image5*, *omni_image7*, and *omni_image9* were taken from both *run3_base_hr* and *run5_base_hr* to be used for the creation of the panorama. These experiments were conducted on OpenCV 3.4.2 and 4.4.0.46.

The sizes of Run 3 and Run 5 folders of the CPET dataset are both around 2 GBs in size.

Project Evaluation and Results:

Evaluation Methods

It is rather unclear how we can objectively measure the quality of the panorama we produced quantitatively. This is because there are no clear metrics on what makes a good panorama, and that ere are no high quality ground truths. After consulting Professor Kelly on this issue, he suggested the best way to compare the panoramas qualitatively. I have decided that I will take this approach and will take this strategy to evaluate my results.

I will be comparing the panorama results from Run 3 and Run 5 to their counterparts in the *omni_stitched* folders of each run. This will be done with the results from OpenCV 3.4.2 and OpenCV 4.4.0.46 for run 3, and OpenCV 3.4.2 for run 5. We will be exploring and commenting the key characteristics such as the effects of segmentation, distortion, and vignetting on both panoramas.

Evaluation of Results

We see that the results of the panorama produced by the algorithm have met the most if not all of the benchmarks set at the beginning of the project.

For Run 3 on OpenCV 3.4.2, the resultant image has virtually no vignetting. There is one visible segmentation between the 4th and 5th images on the right of the panorama. However, the features seem to line up very well, and in the context of a mars rover, this wouldn't be a major concern. There is also virtually no distortion at the ends of the panorama. The 1st and last component images are still straight in

perspective. In contrast, the *omni_stitched* reference has significant vignetting between all the image boundaries. We can safely say that the results produced have surpassed the quality of the reference.

For Run 5 on OpenCV 3.4.2, there was no vignetting, distortion, or visible signs of segmentation between the images. The resultant panorama can be said to be better than the *omni_stitched* reference, which has strong signs of vignetting. All that applies to Run 3 on OpenCV 3.4.2 can be applied here.

For Run 3 on OpenCV 4.4.0.46, 3 of the 5 images are visible in the resultant panorama. Although there is minimal vignetting, distortion, and segmentation, 2 of the images (1st and 2nd) do not seem to make an appearance. This is in contrast with the *omni_stitched* reference, which has all 5 photos. However, the *omni_stitched* reference has heavy photo vignetting and segmentation. We know that the most likely reason why the other 2 images didn't make it in is because of the OpenCV software on 4.4.0.46, as the features and homographies were still accurately calculated. The feature matching can be seen the zip folder in *feature_img_1 to 4*.

Results

Run 3 – Results and Comparison (OpenCV 3.4.2)

Figure 1. Resultant Panorama for Run 3

Figure 2. Comparison Image from omni_stitched for Run 3

Run 5 – Results and Comparison (OpenCV 3.4.2)

Figure 3. Resultant Panorama for Run 5

Figure 4. Comparison Image from omni_stitched for Run 5

Run 3 – Results and Comparison (OpenCV 4.4.0.46)

Figure 5. Resultant Panorama for Run 3

Figure 6. Comparison Image from omni_stitched for Run 3

Explanation

Some of the results produced were somewhat surprising. For all the panoramas produced by the proposed algorithm, they all have very little vignetting, segmentation, or distortion after the stitching. The panoramas seemingly blend the common sections between the component images perfectly, and there doesn't seem to be any bending/distortion towards the edges of the resultant images. This can be clearly seen with the panoramas produced for run 3 and run 5 of the dataset on OpenCV 3.4.2. We didn't expect the quality of the images to be this high.

The results produced on OpenCV 4.4.0.46 has produced more mixed results. The resultant panorama is still produced with minimal image vignetting, segmentation, and distortion, even in comparison with the reference panorama from the *omni_stitched* file. However, around 2 of the images out of the 5 that we want included in the resultant image were not accounted for, as seem above. The stitcher function cv2.Stitcher_create() from OpenCV 4.4.0.46 have left out these images. After experimenting with different image sets from all the rover runs available in the CPET dataset, we are confident that our feature detection and homography calculations were implemented correctly, and this issue only seems to appear with OpenCV 4.4.0.46.

We suspect that OpenCV 4.4.0.46 seems to be sensitive to certain imperfections of the images presented in the dataset. In the runs that we have experimented with, it appears the component images that didn't get

stitched together have matching features located in the regions where there is a lot of noise, have more blurring, or less lighting than usual, which would make the images harder to stitch.

Challenges and Limitations

I think the main challenges of this project was trying to produce a full 360^{o} image of standard to work with the version of OpenCV suggested in the project description. After experimenting with different sets of images, this issue persisted throughout the project. I think the requirement of minimal vignetting and frame segmentation has contributed to the issue to stitching together images that have lower lighting, more blurring, or more noise. Due to personal issues and the time constraint on this project, we will have to leave this issue at rest for now.

The clear solution to the challenges noted would be to create a manual uses translation-based image stitcher than used the one provided by OpenCV. This would ensure that noise, blurring, or lighting issues would not be as problematic to the stitching issue. However, this would have its disadvantages since vignetting, distortion, and segmentation will be much more likely to occur with a translation-based method.

References:

Insert bibliographic details for any references you intend to or expect to use in your project. You may use the IEEE standard bibliographic format.

[1] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, pp. 1–28, Jan. 2004.

[2] K. G. Derpanis, “Overview of the RANSAC Algorithm,” pp. 1–2, May 2010.

[3] E. Dubrofsky, “Homography Estimation,” pp. 1–32, 2009.

[4] O. Lamarre, O. Limoyo, F. Maric´, and J. Kelly, “The Canadian Planetary Emulation Terrain Energy-Aware Rover Navigation Dataset,” *The International Journal of Robotics Research*, vol. 39(6), pp. 641–650, 2020.