P4: Navigating the Unknown (Using 2 late days)

Mayank Bansal
Robotics Engineering
Worcester Polytechnic Institute
Email: mbansal1@wpi.edu

Siyuan 'Oliver' Huang
Robotics Engineering
Worcester Polytechnic Institute
Email: shuang4@wpi.edu

Miheer Diwan
Robotics Engineering
Worcester Polytechnic Institute
Email: msdiwan@wpi.edu

***Abstract*—This project aims to develop an algorithm to detect an unknown arbitrary-shaped window in the world and fly through it using the DJI TelloEDU drone and the NVIDIA Jetson Orin Nano. We use SPyNet to detect the gap using optical flow and after some post-processing, the gap's location is found.**

I. Environment

The environment has an arbitrarily shaped window(s) that can be 'seen' from the origin location. The window is made of foam core with texture stuck on it. The texture is not known prior to the flight but the windows are not textureless. Also, the colors on the window and the background could be similar but the patterns were not exactly the same. The board was nearly planar and had multiple holes/gaps. In this case, the goal was to fly through the largest gap. Furthermore, there were offsets of about +20 and -20 degrees in the yaw angle and a similar offset in the X-axis.

Fig. 1. Environment image

II. Gap Detection

A. Optical Flow

Optical flow is a technique used to understand image motion. It is usually applied to a series of images that have a small time step between them, for example, video frames. There are two types of optical flow - Sparse and Dense. Sparse optical flow follows the movement of only a few pixels while dense optical flow follows the movement of all the pixels in the image. To find the location of the gap, dense optical flow is more useful than sparse optical flow. The idea is that the pixels belonging to the background texture moves slower compared to the foreground texture pixels when drone is moved. This disparity in pixel flow (optical flow) will enable us to find the gap.

Initially, we tried using classical methods like Farneback algorithm to find the optical flow vector, but this method is pretty slow and the resultant vector field is not of a high resolution which makes it difficult to segment out the gap effectively. We then shifted to deep learning methods which are usually more accurate than classical methods.

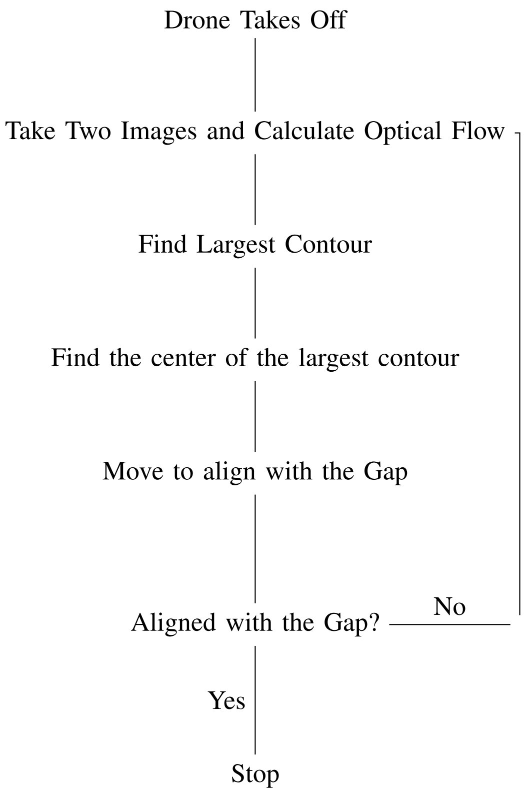
We used the PyTorch implementation of Anurag Ranjan's SPyNet (Spatial Pyramid Network for Optical Flow). It is a light-weight network with fast inference speeds (We were getting each inference in around 0.5 seconds).

Link to Github Repository: pytorch-spynet

B. Methodology

The drone takes off and then moves a bit forward so that the full board is in view. We take two images in 0.5 seconds succession and then we then calculate the optical flow using SPyNet from these two images. Figure 2 shows the colored optical flow from the two images. The color(hue) is based on the angle of the optical flow vectors and the saturation is based on the magnitude of these vectors. Figure 3 shows the normalized magnitude of these vectors. Now we need to detect the largest contour and segment that out to find the largest gap. Initially we used Canny edge detection and thresholding on the normalized optical flow image to find the largest contour but this method was not very robust. We ended up using otsu + binary thresholding which decides the threshold value dynamically for the magnitude image and segments the darker parts based on this threshold value. Then, the contours are detected on this thresholded image and the largest contour is found along with its centroid. Figure 4 illustrates this step.

The next step is visual servoing. The primary logic here was very simple. If the centroid of the largest contour was away from the image center, we commanded the drone to fly in that direction. For example, if the centroid is to the right of the image center, we command the drone to move right for a very small distance. This was repeated till the centroid was within a set range of the image center and helped us align the drone with the window. After centering, we fly through the window.



III. Result

Fig. 2. Colorized Optical Flow

Video Submission Link

IV. Conclusion

Our approach using Optical flow works well for this task, however, the thresholding is not very robust and requires very fine parameter tuning to detect the contours. Similarly, using visual servoing may not always be very reliable. One more approach which we could not implement was to get depth using Stereo. While the drone does not have a stereo camera setup, we can use odometry and the existing monocular camera on the drone to get depth. We can command the drone to capture an image, move a certain distance, and capture another image. The distance moved by the drone will be treated as the baseline for our pseudo-stereo setup. We can estimate the relative pose of the drone and use stereo photogrammetry and disparity maps to get depth. This will help us to find the gap along with its depth from the drone.

Fig. 3. Normalized Optical Flow Magnitude

Fig. 4. Largest Contour with its Centre