ECEn 631 Motion Field and 3D Reconstruction

Objectives:

- Learn optical flow and its limitations.
- Use template matching to calculate motion field.
- ~~Learn to track features across multiple frames to get correspondences for long baseline applications.~~
- Learn to compute the fundamental matrix from two images taken by one camera.
- Learn to reconstruct 3D up to an unknown scale factor.

Instructions:

- Generate a PDF file that includes your video links for this assignment.
- Submit your PDF file and source code file(s) in one zip file without the folder or directory.
- Use your first name and last name (e.g., justinsmith.zip) as the file name.
- Login to myBYU and submit your work through BYU Learning Suite online submission.
- Use your own camera to capture videos for processing.
- Convert the image frame to 8-bit single channel using cvtConvert() with the CV_RGB2GRAY flag before processing.
-

Task 1: Optical Flow 25 points

- Move your camera in a 3D scene that has rich features for tracking and record a 10-second video that includes at least five different motion fields (rotation, focus of expansion, focus of contraction, and translation in at least two directions) for this assignment.
- Use goodFeaturesToTrack() to detect enough good features that can be used to calculate a motion field of at least 300 motion vectors. Enter these good features as the previous points to the calcOpticalFlowPyrLK() function to find the next points in the next frame.
- Use the OpenCV calcOpticalFlowPyrLK() function to obtain a **spares** motion field of at least 300 motion vectors between Frames *i-m* and *i* (*i* = *m*, *m* +1, *m* +2,…*N-1*). *N* is the total number of frames in the video and *m* is the number of frames skipped.
- Depending on your camera framerate and how fast you move the camera, select an adequate optical flow window size and find the maximum skipped frames *m* that the calcOpticalFlowPyrLK() function could still find 300 good motion vectors even with the highest pyramid level. The window size can be adjusted depending on the baseline (how many frames you skip). They should be small enough that affine transformation would not noticeably affect the result and large enough to find good motion vectors.
- Show motion vectors on each NEXT frame (*i* = *m*, *m* +1, *m* +2, …*N*-1) by drawing a red line that connects each next point to its corresponding previous point (*m* frames back) and a green dot at the location of each next point.
- Submit the YouTube link to the resulting video with the motion vectors from Frame *m* to Frame *N*-1.
- Report the number of skipped frames (*m*), the pyramid level, and optical flow window size, your observation, and what you learn from this task.
- Submit your code and your discussion.

Task 2: Template Matching 25 points

- Use the same recorded video for Task 1.
- Use a template matching method (SSD or NCC) to obtain a sparse motion field of at least 300 good features to track.
- Choose your own template size and search window size. These window sizes can be adjusted depending on the baseline (how many frames you skip). They should be small enough that affine transformation would not noticeably affect the result and large enough to find good motion vectors.
- Repeat the same experiments in Task 1 (without pyramid) but use template matching to find motion vectors.
- Show motion vectors on each NEXT frame (*i* = *m*, *m* +1, *m* +2, …*N*-1) by drawing a red line that connects each next point to its corresponding previous point (*m* frames back) and a green dot at the location of each next point.
- Submit the YouTube link to the resulting video with the motion vectors from Frame *m* to Frame *N*-1.
- Report the number of skipped frames (*m*), the template and search window sizes, your observation, and what you learn from this task.
- Submit your code and your discussion.

Task 3: Camera Pose (Motion) between Two Views 25 points

Calibrate the camera you used to record the video for Tasks 1 and 2 and use its intrinsic parameters and distortion coefficients for this task. You can use the intrinsic parameters and distortion coefficients of your own camera from the Camera Calibration assignment if it is the same camera. With a calibrated camera but unknown camera motion, the object structure can be reconstructed up to an unknown scale factor.

- Include the intrinsic parameters and distortion coefficients of your camera in your PDF file.
- Use the matching data points between two frames from either Task 1 or Task 2 for this task.
- You will need an adequately long baseline (an *m* that is equivalent to a few inches) to obtain accurate result.

- You may have to track features across multiple frames (more than *m*) to obtain a long baseline.
- Undistort the data points using the undistortPoints() function before calculating a good fundamental matrix. Remember DO NOT to pass R1(2) and P1(2) (We don't have them anyway) to the undistortPoints() function because we just want to correct the distortion not rectification.
- Use the undistorted matching data points and findFundamentalMat() function to calculate the fundamental matrix F.
- Calculate the essential matrix E from F using $E = (M_r)^T F\, M_l$.
- Since we do not know the exact camera motion, E should be normalized so that we don't get confused (see 3D Reconstruction lecture slides 31 and 32). Remember $E = \hat{T}R$, where $\hat{\mathrm{T}}$ is a 3×3 skew-symmetric matrix.
- You can use OpenCV recoverPose() functions to recover R and T from E.
- Exam the R and T matrices you select and explain what your estimated camera pose is (up to an unknown scale factor).
- Submit your explanation and the R, T, E, and F matrices in your PDF file. Submit your code.

Task 4: Simple Structure from Motion 25 points

Structure from Motion uses multiple views to computer the rotation and translation between the camera and the 3D object as well as the structure of the object. With a calibrated camera but unknown camera motion (scale factor), the object structure can only be reconstructed up to an unknown scale factor. We will use only two views to perform simple structure from motion.

- Measure the distance in inches of two points in your 3D scene. Find the unknow scale factor that will result in the correct distance between those two points and provide accurate estimate of the R and T between the camera and the 3D scene.
- Report your R (3×3), T, and the estimated scale factor in your PDF file.
- You could perform stereo rectification using the stereoRectify() function to obtain the reprojection matrix Q to calculate the 3D measurements of those two points (similar to the previous assignment) if your camera motion is not too different from simple translation (canonical configuration).
- You can implement one of the triangulation methods (3D Reconstruction lecture) to calculate the two 3D points if the camera motion is more than a simple translation.