ME 735: Project Report

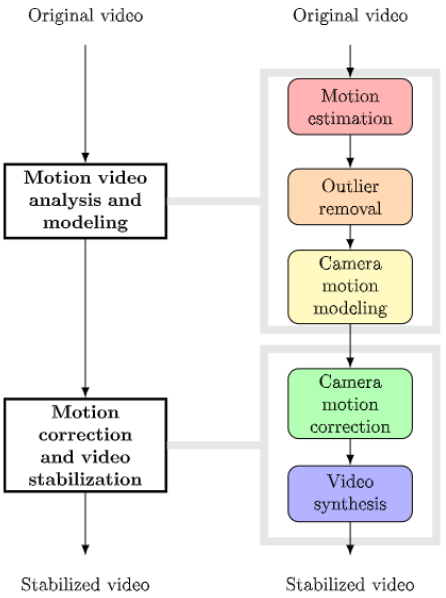
Group - 2
Aniruddh Radhakrishnan - 190260007
Vidushi Verma - 190100135
Prasham Shah - 19D100018
Dhruv Sah - 190110020

1. Abstract:

While shooting videos, if the camera is not stable, i.e. there is some rapid camera motion, we get shaky videos as the overall frame captured in these videos is not constant and changes in an inconsistent manner. Through this project idea, we used different graphic algorithms to stabilize the frame of the video and facilitate a smooth transition from one frame to the next without having to crop the content. We tried and tested different standalone as well combination of methods for video stabilization using python programming language.

2. Introduction:

The term "video stabilisation" describes a group of techniques used to lessen the impact of camera movement on the video. The camera may move in translation (i.e., in the x, y, or z directions) or rotation (yaw, pitch, roll)
Videos must be stabilized in medical diagnostic applications like endoscopy and colonoscopy to pinpoint the precise location and extent of the issue. Similar to this, films taken by aerial vehicles during a reconnaissance flight that are used for military purposes must be stabilised for localization, navigation, target tracking, etc. Robotic applications follow the same rules.
The video stabilization technique has a number of steps included in it. I'll explain them using the below figure:



We can divide the process into 2 main buckets: Motion video analysis and modelling followed by Motion correction and video stabilization.

A. Motion video analysis and modelling
   1. Motion estimation - It estimates the camera movements from the video. It has 2 methods namely Pixel/Feature Based Matching and Block Matching.
   2. Outlier removal - It focuses on camera driven movement by motion outlier detection/ removal. It has 2 methods namely Frame to Frame analysis and Video Stream Analysis.
   3. Camera motion modeling - This process corrects and smoothens the estimated camera movements. It has 3 methods which are 2D Model, 3D Model and Perceptual.

B. Motion correction and video stabilization
   1. Camera motion correction - This process corrects the camera motion by applying a low-pass filter in order to smoothen the camera movements. It includes low pass filtering and camera filtering.
   2. Video synthesis - It processes the video by using the softened camera movements. It has dense reconstruction as well as sparse reconstruction methods.

3.Methodology

To analyse which Video stabalization method provides the best results, we have used 3 algorithms (Explained below). These algorithms have been used in different comibnations to find the output stabalized videos.

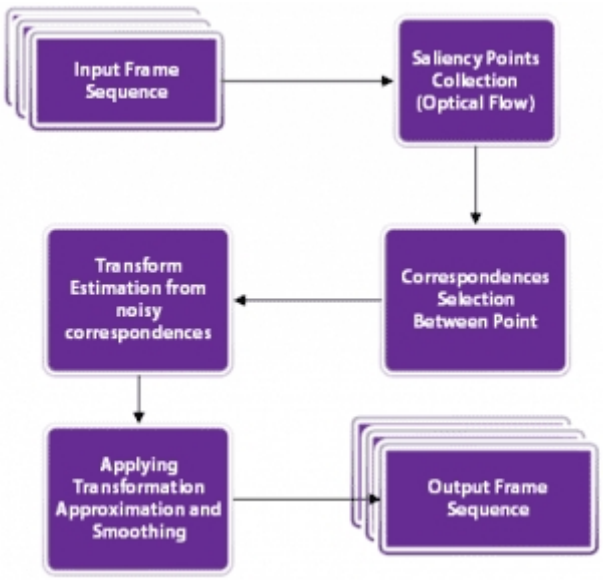
Two kinds of videos have been used to check the stability

1. Static videos: The objects are stationary, vibrations are caused anly due to the cameraman
2. Dynamic videos: The objects in the video are moving as well. In the test case, an oscillating pendulum is used

The quality of stabalization is anaysed using Algoruim 4 (explained in section 4.4) is used. The algorithm wit the lowest/highets stability value is the one providing the best result

4. Algorithm Used

4.1 Algorithm-1: Point Feature Matching



1. **Step 1 - Set Input and Output Videos:** The first step includes completing the setup for reading the input video and writing the output video.
2. **Step 2 - Read the first frame and convert it to grayscale**
3. **Step 3 - Find motion between frame:** Iterate over all the frames and find the motion between the current frame and the previous frame. We generally find the motion of 50-100 points and then use them to robustly estimate the model.

   3.1 Good Features to Track : We use the OpenCV function goodFeaturesToTrack to detect features ideal for tracking which are generally textured regions with lots of corners.

   3.2 Lucas - Kanade Optical Flow : After getting good features, we track them in the next frame using the Lucas-Kanade Optical Flow algorithm which can implemented using the function calcOpticalFlowPyrLK in OpenCV. Do note, we use status flag in the function to filter out the occluded values.

   3.3 Estimate Motion: We use the function estimateRigidTransform in OpenCV to find rigid transformation mapping previous frame to the current frame.
4. **Step 4 - Calculate smooth motion between frames:** This step gives the trajectory of the motion by cumulatively adding the differential motion.

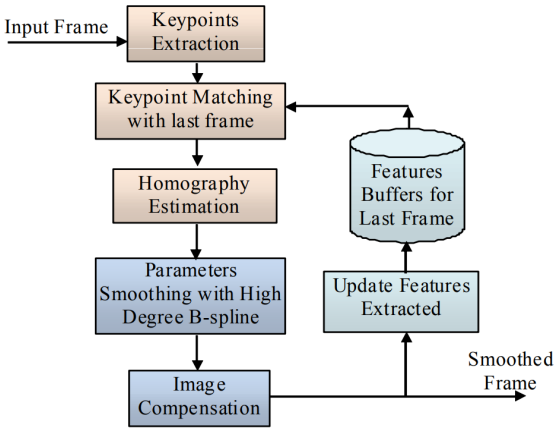
   4. 1 Calculate trajectory: We use the cumsum function in numpy to calculate the trajectory.

   4.2 Calculate smooth trajectory: Moving average filter is used to smoothen the curve.

   4. 3 Calculate smooth transforms: This is done to obtain smooth transforms that can be applied to the frames of video to stabilize it which is done by finding the difference between the smooth and original trajectory.
5. **Step 5 - Apply smoothed camera motion to frames:** This step loops over the frames and applies the transforms that we calculated.

   5.1 Fix border artifacts: We mitigate this problem by scaling the video about it's center using the function fixBorder and getRotationMatrix2D.

4.2 Algorithm-2: Point Spline Matching



1. **Step 1:** Extract corner based features using Features from Accelerated Segment Test(FAST) corner detector followed by matching the pairs.
2. **Step 2:** Estimate motion between two consecutive frames based on a projective transform model.
3. **Step 3:** Estimated motion parameters are accumulated and smoothed by a B-spline model(Degree=3 in our case).
4. **Step 4:** Frames compensated based on smoothened parameters to reproduce a stable video.

4.3 Algorithm-3: L1 path optimisation

This algorithm can be divided into 6 steps which are explained in the following steps:

1. **Step-1:** The algorithm tracks features across the frames in this step.
2. **Step-2:** Movements between frames in identified and the movement is given a motion vector.
3. **Step-3:** Motion across all the frames is obtained.
4. **Step-4:** A Linear programming problem is formulated along with certain constraints.
5. **Step-5:** Motion across all the frames is minimized through the Linear Programming problem.
6. **Step-6:** The obtained minimal motion is used to smoothen the camera trajectory.

The intuition briefly explained below:

$$P_t = C_t B_t,$$

Here $B_t = C_t^{-1} P_t$ is the update transform that when applied to the original camera path C_t, yields the optimal path P which is stabilized video path.

D is basically a discrete differentiator which gives the difference between 2 consecutive frames which is what we want to minimize using the linear programming method as below:

$DP(t) = 0$, $D^2P(t) = 0$, $D^3P(t) = 0$.
Inclusion and Proximity constraint

$$\mathcal{O}(P) = w_1|D(P)|_1 + w_2|D^2(P)|_1 + w_3|D^3(P)|_1$$

1. Minimizing $|D(P)|_1$:

$$|D(P)| = \sum_t |P_{t+1} - P_t|$$

$$|D(P)| = \sum_t |C_t F_{t+1} B_{t+1} - C_t B_t|$$

$$\leq \sum_t |C_t||F_{t+1} B_{t+1} - B_t|.$$

4.4 Algorithm-4:Stabality quality analyser

Step 1: Capture a video with a distinct bright red light visible in the background
Step 2: Render the final videos after using permutations of the chosen algorithms for video stabilization
Step 3: Use the image processing software to track the motion of the red light in the stabilized software. The algorithm trackes the X and Y coordinate of the red bright spot. The mean location of the Red spot is calculated.
Step 4: R(X,Y) for the red light is plotted against time.
Step 5: The area under the R curve and time is calculated (Maxr- in table). The inverse of the area under the contour is used to compare the stability of different videos.

5. Results

The following algorithms are used

1. L-1 Stabilization
2. Point feature matching
3. Point spline matching

The results for static video stabilization are as follows:

Static Stabilization		
file	**maxr**	**stability**
Static_(l1)	204.67	0.0048
Static_(l1+pointspline)	224.17	0.0044
Static_(pointspline+l1+point-feature)	231.98	0.0043
Static_(pointspline+l1)	232.65	0.0042
Static_(point-feature+pointspline+l1)	239.57	0.0041
Static_(l1+point-feature)	255.98	0.0039
Static_(l1+point-feature+pointspline)	409.94	0.002439
Static_(pointspline+point-feature+l1)	410.97	0.002433
Static_(point-feature+l1)	412.37	0.002425
Static_(point-feature+pointspline)	492.73	0.002029
Static_(point-feature+l1+pointspline)	496.76	0.002013
Static_(l1+pointspline+point-feature)	500.65	0.001997
Static_(point-feature)	529.51	0.001888
Static_(pointspline+point-feature)	533.10	0.001875
Static_(pointspline)	544.89	0.001835
Static	545.27	0.001833

The results for the dynamic video stabilization are as follows:

Dynamic Stabilization		
file	**maxr**	**stability**
Dynamic_(pointspline+l1)	221.575	0.0045
Dynamic_(point-feature+pointspline+l1)	261.771	0.0038
Dynamic_(l1)	313.761	0.00318
Dynamic_(pointspline)	390.630	0.00255
Dynamic_(l1+point-feature)	432.225	0.00231
Dynamic_(point-feature+pointspline)	451.783	0.00221
Dynamic_(l1+point-feature+pointspline)	480.492	0.00208
Dynamic_(pointspline+point-feature)	485.416	0.00206
Dynamic_(l1+pointspline)	515.235	0.00194
Dynamic_(l1+pointspline+point-feature)	528.914	0.00189
Dynamic_(point-feature+l1)	540.528	0.00185
Dynamic_(pointspline+l1+point-feature)	540.659	0.00184
Dynamic_(pointspline+point-feature+l1)	545.488	0.00183
Dynamic_(point-feature)	571.399	0.00175
Dynamic	596.163	0.00167
Dynamic_(point-feature+l1+pointspline)	599.441	0.00166

6. Conclusions

The comibinaiton of algorithms XYZ gives the best results for static video stability
The combination of ALogorithms XYZ gives the best result for dynamic video stability
WIth the use of more algorithms, the computational time increases exponentially.
The use of more algorithms doesnot necessarily mean more stability.For dynamic stability, the XYZ algorithm provided better results in comparison with UGHS algorithms. The stabilization in the stable video is more than the dynamic video. This is due to the presence of a moving feature

7. References

a. Guilluy, Wilko & Oudre, Laurent & Beghdadi, Azeddine. (2021). Video stabilization: Overview, challenges and perspectives. Signal Processing: Image Communication. 90. 116015. 10.1016/j.image.2020.116015.
b. L. M. Abdullah, N. Md Tahir and M. Samad, "Video stabilization based on point feature matching technique," 2012 IEEE Control and System Graduate Research Colloquium, 2012, pp. 303-307, doi: 10.1109/ICSGRC.2012.6287181.
c. Y. Wang, R. Chang, T. W. Chua, K. Leman and N. T. Pham, "Video stabilization based on high degree B-spline smoothing," Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), 2012, pp. 3152-3155.
d. M. Grundmann, V. Kwatra and I. Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," CVPR 2011, 2011, pp. 225-232, doi: 10.1109/CVPR.2011.5995525.