© August 2019 | IJIRT | Volume 6 Issue 3 | ISSN: 2349-6002

Video stabilization using Optical Flow

Kaushik Indranil Patil[1], Simran Sachin Sura[2]
[1]*Nashik District Maratha Vidya Prasarak Samaj's Karmaveer Baburao Thakare College of Engineering, Nashik*
[2]*Smt.Kashibai Navale College of Engineering, Vadgaon, Pune*

***Abstract*- - Video Stabilization is the technique to reduce jittery motion in a video. This paper discusses the steps involved in video stabilization using Optical Flow: Feature extraction, Optical Flow using Lucas-Kanade method, Image Affine transformation. In this paper, we will also discuss mathematical models involved in each step of video stabilization. In the later part of the paper, we have mentioned methods to improve the video stabilization results and make it more efficient while applying to high-resolution videos.**

Index terms- Video Stabilization, Optical Flow, Lucas-Kanade method, Taylor Series, Laplacian Pyramids

I. INTRODUCTION

Human mind is designed to get a satisfaction out of stable visual scenery and is disturbed by unstable and jittery visual scenery. Thus it becomes necessary to design a method to stabilize videos recorded by jittery support.

Video Stabilization method was first designed in 1991 through mechanical stabilization housing. Primitive mechanical stabilization used 2 axis gyroscope, each for sensing motion in motion in two perpendicular axes. The output signal of this gyroscope was analysed by a microcontroller which actuates two perpendicular motors to compensate the motion sensed by the gyros [1]. Today there are mechanical video stabilizers available having 5 stabilization axes. The major disadvantage of this method was large space and power requirement.

Other methods of video stabilization called Optical Image Stabilization(OIS) consists of a movable floating lens system. Similar to mechanical stabilization this method also uses gyroscope and accelerometer to sense motion change. The data from these sensors is analysed by a microcontroller which commands the floating lens assembly so as to compensate the unintentional motion in yaw and pitch direction.

The major disadvantage of this method is that it cannot compensate for high-frequency motion. Optical Image Stabilization(OIS) is widely used in smartphones in association with digital image stabilization to produce an excellent stabilization result.

Thus a method for stabilizing the video via software was developed. There are many techniques for digitally stabilizing videos. Here, we will be discussing corner matching method and apparent motion detection of these blocks across consecutive frames in a video stream via Optical Flow algorithm.

Section II describes the basic principle behind video stabilization, section III covers algorithm to detect good features to track, section IV shows the results of the experiment, section V covers the future scope and VI concludes the paper.

II. PRINCIPLE BEHIND VIDEO STABILIZATION

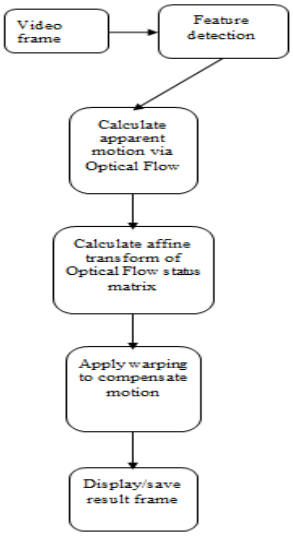

Fig 1. Steps involved in video stabilization using Optical Flow

Feature detection and minimizing the distance between identical feature across consecutive frames to produce steady video, having reduced jitter than the original video.

III.A FEATURE DETECTION

Features in images are the distinct regions in an image such as the pattern of colours, corners, edges and blobs which can be used to extract information from the image [4]. Good features should have the following properties [5]:

1. Repeatability: Irrespective of the angle of viewing the object of interest, good features are the one which is common to maximum viewing angles. Repeatability of features in an image also implies more robustness over uncertain image deformation.
2. Distinctiveness: A features should show maximum variation than the image region in its vicinity to call it as a good feature.
3. Locality: A feature should not be affected by problems of occlusion caused by viewing the object of interest from different viewing angles.
4. Quantity: Number of features in an image should be ample enough to be tracked without being affected by uncertain losses in an image.

In this paper, we will focus on tracking "good" corners in video frames.

III.B MODIFICATION TO HARRIS CORNER DETECTION

In 1994, J. Shi and C. Tomasi made a small modification to the Harris corner detection algorithm and simplified the algorithm. They modified the equation for Harris corner detection and simplified it to equation eq(1):

$$R = min(\lambda_1, \lambda_2) \quad \ldots(1)$$

Where λ_1 and λ_2 are the eigen values of *M*. M is defined as:

$$M = \begin{bmatrix} \overline{Ix} & \overline{Ix} \\ \overline{Ix} & \overline{Iy} \end{bmatrix} \Sigma w(x,y). \quad (2)$$

Where $\Sigma(x,y)$ is called as the window function, it gives weight to each pixel being scanned, x, y is the location of a pixel in an image and I is the intensity.

The equation is illustrated in figure Fig 2:

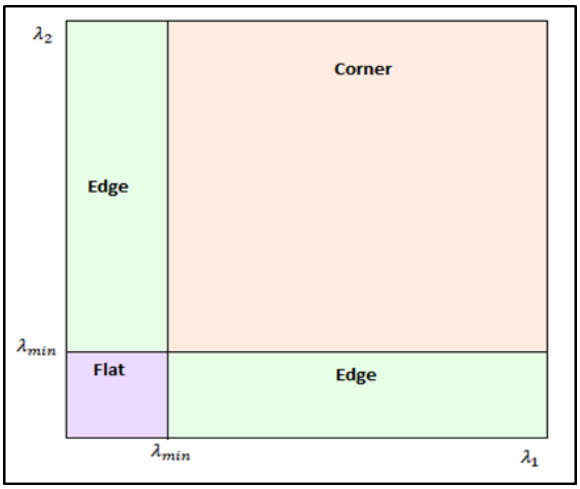

Fig 2. Good Features to Track output based on eigen values of M [8]

This algorithm is proven to have better results than its predecessor[7]. Thus in this paper, we will be using this algorithm to track features in a video frame.

OpenCV function of this method is given by:
cv::void goodFeaturesToTrack(InputArray image, OutputArray corners, int maxCorners, double qualityLevel, double minDistance, InputArray mask=noArray(), int blockSize=3, bool useHarrisDetector=false, double k=0.04)

III.C OPTICAL FLOW

Optical Flow assumes the following two conditions:

1. Pixel densities of an object do not change across consecutive video frames.
2. Neighbouring pixel of a moving object has similar motion as the moving object.

Equation eq(3) represents the first assumption[9] in mathematical form:

$$I(x,y,t) = I(x + dx, y + dy, t + dt) \quad \ldots(3)$$

where I represents the intensity of a pixel, x, y are spatial coordinates of the pixel, t is the temporal advancement of the pixel in video frames, dx, dy, dt represents a small change in x, y, t respectively.

On the application of Taylor series expansion to first assumption equation eq(3) then removing the common terms and dividing the whole equation with d_t we get equation eq(4)[9]:

$$f_x u + f_y v + f_t = 0 \quad \ldots(4)$$

Where,

$$f_x = \frac{\delta f}{\delta x}; \quad f_y = \frac{\delta f}{\delta y}$$

$$u = \frac{dx}{dt}; \quad v = \frac{dy}{dt}$$

Equation eq(4) is called the Optical Flow equation but this equation is under-constraint (one equation two unknown variables) and cannot be solved. There are several methods to solve this equation, one of the method is proposed by B. D. Lucas and T. Kanade.

The Lucas-Kanade method took into consideration the second assumption that is optical flow should be equal for all neighbouring pixel of a single-pixel so 9 equations were obtained from 3x3 pixel matrix(8 pixels surrounding one pixel). Lucas-Kanade system is over constraint system(9 equations, two variables), thus solvable.

After applying the least square fit method[9][10] we obtain equations eq(5) and eq(6):

$$\Sigma f_{xi}^2 u + \Sigma f_{xi} f_{yi} v = -\Sigma f_{xi} f_{ti} \quad \ldots(5)$$

$$\Sigma f_{xi} f_{yi} u + \Sigma f_{yi}^2 v = -\Sigma f_{yi} f_{ti} \quad \ldots(6)$$

For small motions, this algorithm works perfectly but fails when large motion occurs and gives arbitrary results because derivatives used in the method assume a small change in motion. This problem is solved by implementing the pyramid method to the Lucas-Kanade method. Pyramid method scales down the number of pixel motion for the derivatives to give correct results[10][11]

OpenCV function for Optical Flow:

void calcOpticalFlowPyrLK(InputArray prevImg, InputArray nextImg, InputArray prevPts, InputOutputArray nextPts, OutputArray status, OutputArray err, Size winSize=Size(21,21), int maxLevel=3, TermCriteria criteria = TermCriteria (TermCriteria::COUNT + TermCriteria::EPS, 30, 0.01), int flags=0, double minEigThreshold=1e-4)

III.D AFFINE TRANSFORMATION

Affine Transform is used to make a correction to distortions caused in an image. Fewer distortions in an image are favourable. Affine transform preserves points, parallel lines and planes during the correction. Affine transform by 2x3 matrix is denoted by equation eq(7)[12][8]:

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2\times 2} \quad B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2\times 1}$$

$$M = [A \quad B] = \begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}_{2\times 3} \quad \ldots(7)$$

Applying transform to a 2D vector using A and B we get equation eq(8):

$$T = M \cdot [x, y, 1]^T \quad \ldots(8)$$

IV. RESULTS

System configuration on which we tested the Video Stabilization program is CPU: i7 4750HQ, GPU: GTX 950M, RAM: 16GB, HDD: 7200RPM. We first used 1280 x 720 resolution and 30 FPS for video input. The average system usage scenario was CPU: 21% (single thread execution), RAM: 58.9 MB. The output video resolution was 1280x720 and the average FPS of output video was around 30 FPS. In the second scenario, we used 320 x 240 video resolution at 100 FPS. The average system usage: CPU: 10%, RAM: 15.8MB and the output video resolution = 320 x 240 at 63 FPS.

Feature detection using OpenCV function GoodFeaturestoTrack() result is shown in figure Fig 3:

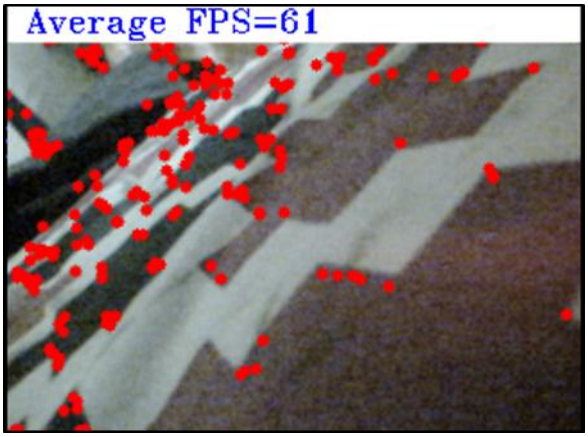

Fig 3. Feature detection result, red colour denotes detected features

V FUTURE SCOPE

Presently we have not used multithreading and parallel programming in our program. While parallelizing the video stabilization program will greatly improve the performance on high resolution and high FPS videos. One more method to improve the program performance is to transfer our program execution on a GPU with CUDA support. Introducing GPU will greatly boost the video stabilization performance by splitting large matrix mathematical calculation in small chunks that can be solved by hundreds of CUDA cores available in

GPU. Using a modern GPU boost the image processing performance by around 10 times[13].

VI CONCLUSION

This paper covers steps involved in video stabilization using Optical Flow with a mathematical representation of each step: Feature detection, Optical Flow using the Lucas-Kanade method and warp Affine transform. In the earlier part of the paper, we also discuss other video stabilization methods involving mechanical system and their disadvantages making digital video stabilization a better alternative. At the end of the paper, we have discussed our system usage while running video stabilization using Optical Flow.

REFERENCES

[1] Review of Motion Estimation and Video Stabilization techniques For hand held mobile video; Paresh Rawat, Jyoti Singhai.
[2] Flashpoint ZeroGrav 2-Axis Digital Gyro Stabilizer.
[3] (Thesis) Video stabilization: digital and mechanical approaches; Serhat Bayrak, November 2008.
[4] Recent Advances in Features Extraction and Description Algorithms: A Comprehensive Survey; Ehab Salahat, Member, IEEE, and Murad Qasaimeh, Member, IEEE.
[5] Local Invariant Feature Detectors: A Survey; Tinne Tuytelaars and Krystian Mikolajczyk.
[6] A Combined Corner And Edge Detector; Chris Harris & Mike Stephens, 1988.
[7] Good Features to Track; J. Shi and C. Tomasi, 1994.
[8] OpenCV 3.0.0-dev documentation.
[9] Optical Flow Estimation; David J. Fleet, Yair Weiss.
[10] UCF Computer Vision Video Lectures 2012, Subject: Optical Flow; Dr. Mubarak Shah.