Full frame video stabilization using motion inpainting

May 27, 2015

Submitted by: M.V.Sandeep

Under the Guidance of Prof. Manish Okade

Declaration

I declare that all the information presented in this thesis has been acquired and presented according to the academic rules and regulations and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work and also that this work is not submitted to any other university or institute for the award of any degree or diploma.

Name : MANTHI VENKAT SANDEEP

Roll Number: 710EC4032

Date :

National Institute of Technology
Rourkela

Department of Electronics and Communication Engineering

Certificate

This is to certify that the thesis entitled, **"Full Frame Video Stabilization using Motion Inpainting"** submitted by **Mr. Manthi Venkat Sandeep** bearing roll no. **710EC4032** in partial fulfillment of the requirements for the award of **Master of Technology Degree (Dual Degree)** in Electronics and Communication Engineering with specialization in "Communication and Networks" during section of 2010-2015 at the National Institute of Technology, Rourkela is an authentic work carried out by him/her under my/our supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/ Institute for the award of any degree or diploma.

Date:

Place:

Prof. Manish Okade
Dept. of Electronics and Comm. Engg.
National Institute of Technology
Rourkela - 769008

Acknowledgements

I am deeply grateful to Professor Manish Okade for being an excellent mentor and advisor. I am extremely thankful not only for his insightful guidance and support throughout my studies, but also for his integrity, discipline and attention to detail which will continue to inspire me for the rest of my life. His interest and dedication towards research and teaching is exceptional and knows no bounds. I feel privileged to be his student.

I would like to thank my thesis panel members, Professor Ajit Kumar Sahoo and Professor Siddharth Deshmukh for their thoughtful comments and suggestions that have helped me to improve this thesis. I am also grateful to Professor Shrishail Hiremath for kindling the interest towards research in me. I also would like to thank Professor Santos Kumar Das for his continuous guidance in various matters throughout my graduate studies.

Finally, this thesis would not have been possible without help and support from my friends and lab mates. I express my gratitude to all of them.

Abstract

The amount of video data has increased dramatically with the advent of digital imaging. Most of the video captured these days originates from a mobile phones and handheld video cameras. Such videos are shaky compared to videos that are shot with a tripod mounted camera. Stabilizing this video to remove the shaky effect using software is called Digital video stabilization which results in a stable and visually pleasant video. In order digitally stabilize the image, we need to (1) Estimate the motion of camera, (2) Regenerate the motion of camera without the undesirable artifacts and (3) Synthesize new video frames. This dissertation is targeted at improving the last two steps of stabilizing the video.

Most of the previous techniques of video stabilization produce a lower resolution stabilized video output and clip portions of frames to remove the empty area formed by transformation of the video frames. We use a Gaussian averaging filter to smoother the global motion in the video. Then the frames are transformed using the new transformation matrices obtained by subtracting the original transformation chain from the modified transformation chain. For the last step of synthesizing new video frames, we introduce an improved completion technique which can produce full frame video by using the pixel information from nearby frames to estimate the intensity of the missing pixels. This technique uses **motion inpainting** to ensure that the video frames are filled in both the static image area and dynamic image area with the same consistency. Additionally, the quality of the video is improved by using a deblurring algorithm which further improves the smoothness of video by eliminating undesirable motion blur. We do not estimate the PSF, in its place, we transfer and interpolate the sharper pixels from nearby frames to improve the sharpness and deblur current frame. Completing the video with motion inpainting and deblurring technique allow us to construct a full frame video stabilization system with good image quality. This is verified by implementing the technique on different video sequences.

Contents

List of Figures

Chapter 1

Introduction

As the digital imaging technology evolved, cameras have continued to become smaller and more mobile. Majority of the videos these days are usually captured by hand-held devices like smart phones and digital camcorders. These videos are often shaky and appear to have an undirected motion. Expensive professional equipment like tripod and camera rigs are used with a camera to capture a stabilized video with no shake. Stabilizing a video digitally post capture is an important video enhancement technology which improves upon the video quality by digitally processing the videos captured from consumer devices. Such devices are illustrated in Figure 1 on left.

Stabilizing video digitally mainly consists of three vital steps: (1) Estimate the motion of camera to obtain the trajectory of the original shaky camera path, (2) Regenerate the motion of camera trajectory by removing or smoothing the shaky component. (3) Complete the video by synthesizing new video frames using smoothed trajectory and other improvements like inpainting to fill in the unknown or empty image areas. For the first step, motion can be estimated in either 2D or 3D. According to the adopted motion model, the video stabilization technique can be identified as 3D-based, 2D-based or 2.5D-based techniques. 3D-based techniques use Structure from Motion (SFM) algorithms to reconstruct and recover the 3D camera poses. 2D-based techniques use affine or homography model models to estimate motion transformations among consecutive frames. Camera path is constructed by estimating and accumulating the rigid transforms obtained by these linear transformations. 2.5D-based techniques adopt a partial 3D reconstruction by using information such as epipolar geometry[3]. A comprehensive literature review is presented in Chapter 2.

Figure 1: Left: Digital video stabilization improves the quality of the video captured by hand-held devices. Right: Professional videos are often captured by expensive camera rigs and external stabilizers like tripods.

2D-based techniques are much better and stronger and also work quicker as they just estimate a lineaar trnsformation model among nearby frames. But this technique is very poor to basically handle the paralax which take place due to the non-trvial depth changes. The 3D stabilization techniques, on other hand can can easily deal with parallax and produce better results. However, 3D techniques are less robust to changes such as motion blur, rolling shutter, camera zoom and feature tracking failures. In the following section, the challenging issues of video stabilization and common artifacts that form when stabilization fails are presented.

1.1 Challenges of the Video Stabilization problem

There have been great improvements in video stabilization algorithms. Some of these algorithms have also been implemented commercially e.g., YouTube stabilizer developed from homography model mixture model[2] and Warp Stabilizer in Adobe Premiere Pro developed from the technique of subspace technique[4]. They produce good results on a wide variety of videos and also can handle certain difficult examples. However, there are some challenging scenarios that still need to be solved and cannot be properly handled by these software tools. In this section, we first discuss some of the major challenges that are present in casually captured videos from hand-held devices. We also demonstrate the type of failure that these challenges cause in a video stabilization algorithm. This discussion motivates our design of new video stabilization algorithms.

Quick motion of camera: Certain motion of cameras like quick rotation and zooming present a challenge for video stabilization. 3D and 2.5D techniques make use of long feature trajectories to stabilize a video. However, when a quick motion occurs, the length of these trajectories drop very quickly and even approach zero in some extreme scenarios. This degrades the performance of trajectory-based stabilization techniques. Figure 2 shows two examples of quick camera zooming(top) and quick camera rotation(bottom). These videos when stabilized contain large empty regions.

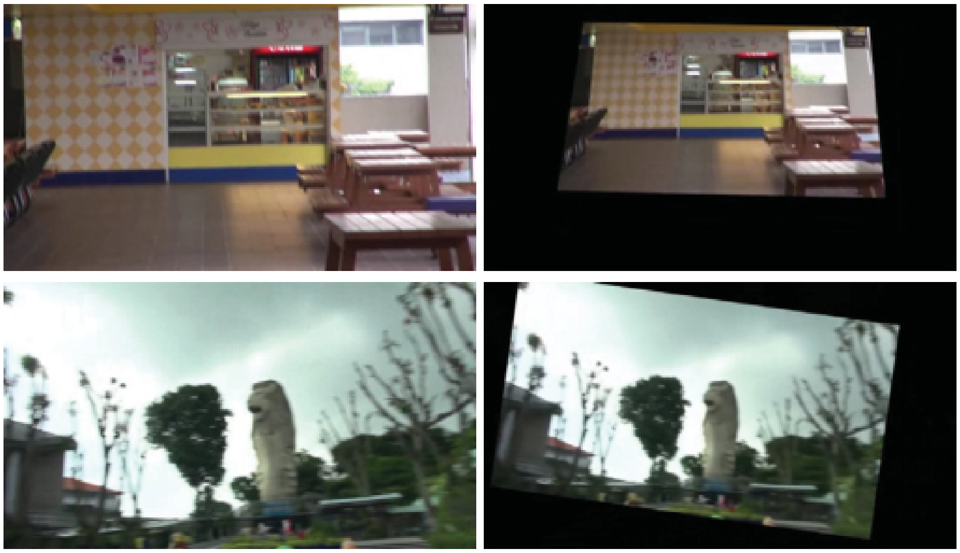


Figure 2: Quick camera movement results in missing image areas

Large moving foreground: Large dynamic objects in a video can easily mislead the video stabilizer during the motion of camera estimation. If size of the dynamic object is not big, RANSAC can be used for both 2D and 3D techniques, to eliminate moving objects. However, it is extremely difficult to distinguish the foreground and background when a large moving object is present in the video. In most of such scenarios, the video stabilizer assumes the foreground motion to be the background motion of camera which would lead to jitter and unstable results. There are a few user-assisted techniques[6] to address this issue by allowing the user to select the background features during motion estimation. The problem of motion segmentation however, still remains as an elusive challenge for automatic systems. Figure 3 shows the kind of video sequences where a large moving foreground could cause problems in video stabilization.

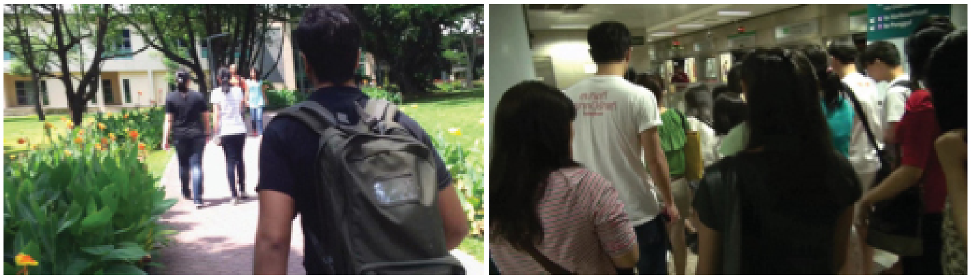


Figure 3: Failure of motion of camera estimation in presence of large moving foreground

Motion blur: When there is extreme camera shake, it can lead to significant blurring of information in video frames. Figure 4 shows one such example where camera shake has lead to intense motion blur in the video. Many video stabilization algorithms can still process this footage and produce stabilized results. However, the motion blur which occurs in the original motion of camera is left untouched. This also leads to additional motion blur in the final stabilized video that appears unnatural. Stabilization algorithms usually depend on feature tracking to estimate the motion of camera. However, feature tracking over blurred frames is not reliable as there are no sharp image features in the frame. Therefore, handling motion blur is a very important task for a good video stabilization system.

Rolling shutter effects: The rolling shutter effect is caused because of the way the data is read out from a CMOS sensor inside the camera. Pixels that are in a row are read simultaneously, but the vertical pixel read out is shifted row by row. This results in the bending of straight lines in the captured video frame as shown in figure 5. This effect is not very visible when the motion of camera is slow. However, during quick motion of cameras, this effect becomes much more noticeable. Since a large number of consumer captured videos that are shaky have quick motion of cameras, they are also more likely to have this rolling shutter effect. Most of the smart phones and consumer capture devices use CMOS sensors because of their lower power consumption. A good stabilization system should also address the rolling shutter effect.

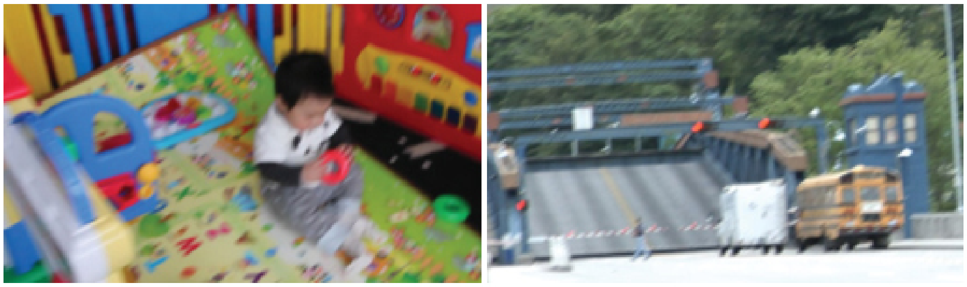


Figure 4: Motion Blur in Video Frames

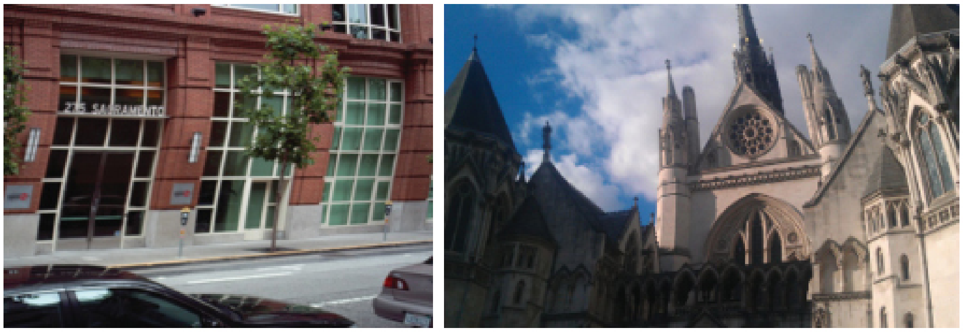


Figure 5: Rolling shutter effects

Large depth variation: Most of the videos have varying depth in them. In 2D video stabilization, affines or homographies are used to model the motion among nearby frames. A single homography model technique is only valid for a planar scene or camera under pure rotation. No single homography model can best fit all the motions of the scene. This results in wobble artifacts. For small depths, these stabilization algorithms can produce relatively good result. However, when the depth variation is large, multiple homographies are needed for motion estimation[7][8] and this can present a challenge to the problem of video stabilization. In figure 6, two such video sequences are shown which get wobble distortions in the final stabilized video.

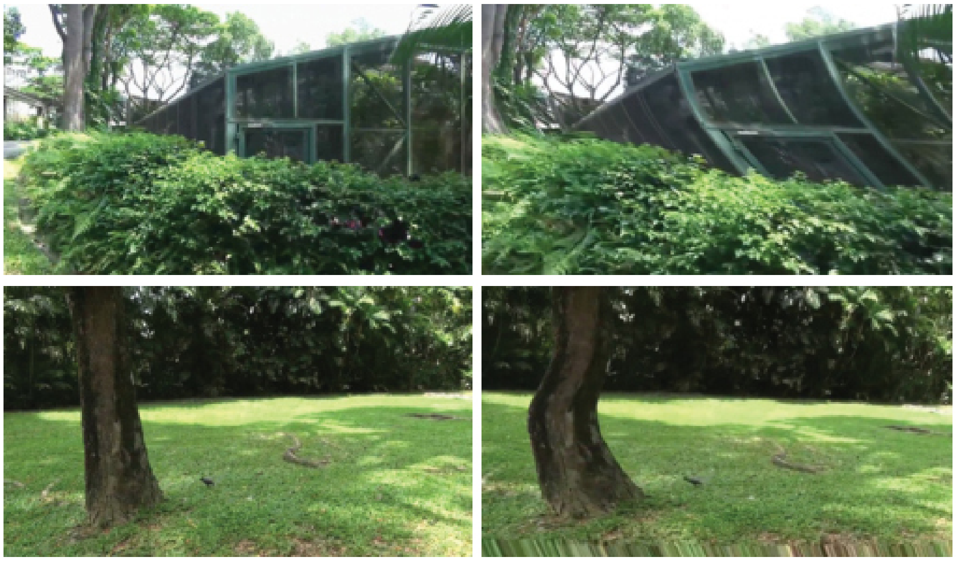


Figure 6: Large Depth Variation and Wobble distortions

1.2 Objective

Each of the challenge presented above is a research problem in itself. A good video stabilization system should try to overcome as many challenges as possible. Nonetheless, the problem becomes more pronounced and difficult when multiple challenges are presented in a single video. e.g., rolling shutter effect with large moving foreground or large depth variations with motion blur. It is very likely that such challenges are also linked together in real world scenarios. In fact, when the 3D reconstruction technique is feasible, 3D techniques often produce excellent results for a sequence with large depth variations. Yet they lack the ability to handle other types of challenges. The 2D based stabilization techniques however, are robust to quick motion of cameras, but are limited in their capacity to handle large depth variations. A video with large dynamic objects or moving foreground often require motion segmentation techniques to discover the motion of camera. If the size of the foreground is dominant, then it presents a very difficult challenge for stabilization.

Other prominent problem is the issue of stability. Some stabilization techniques successfully eliminate the high-frequency motion of camera, but fail to correct the low-frequency camera shake that is unintentional. To develop a high quality video stabilizer, we also need to identify the low-frequency jitters that were unintentional and remove them. Another thing that could be done is to give the user control over the frequencies of motion that need to be eliminated. This could help the user to produce better results if the first pass does not work. Another important issue is of the cropping of video because of empty areas obtained from video stabilization. To address this problem, we can introduce techniques such as motion inpainting and mosaicing to locally warp the pixels from nearby frames to populate the empty pixels in the current frame. Artifacts like wobble would be introduced if the camera path is not smoothed properly. Practically, a reduction in stability suppresses the wobble in videos. However, we need to achieve a fine balance among stability and reducing the wobble to produce a high quality output.

In summary, a good video stabilization system should produce a good stable full frame video output with no geometrical distortions and wobbles. It should also correct the rolling shutter effects and also handle motion blur in a better way and eliminate the additional blur that is caused in the final output because of non association of blur in the input video to that of the digitally stabilized video. Existing techniques cannot satisfy all the goals. However, it is worth exploring towards the direction of a perfect video stabilizer with all the above mentioned features.

1.3 Contributions

This section provides a brief introduction to all the problems we have studied in this work: Stabilizing videos using 2D Affine transformation model to estimate the motion among frames, A technique to complete the video using motion inpainting and local pixel warping to obtain full frame stabilized videos and a technique for deblurring the motion blur in the final result using an interpolation based technique.

Motion Estimation and Smoothing: We estimate the Global motion which represents the frame to frame image transform using a homography model model to detail the geometric transformation among the two frames. The hierarchial motion estmation framwork that is introduced by Bergen et al.[5] is used. Local motion is estimated separately using the Lucas-Kanade pyramidal model optical flow computation[9]. The high frequency part present in global motion chain is removed by applying a Gaussian averaging filter.

Completing the Video with Motion inpainting: We initially locally adjust the image mosaics from nearby frames utilizing the motion field obtained from local motion estimation. Next, a motion inpainting technique is used propogate the local motion field into the empty areas of image at which the local motion cannot be computed directly. After the motion field in the area of image with empty pixels is obtained, we locally warp pixels from the nearby frames using the local motion information using Fast Marching technique[10]. Additional missing pixels are filled by using a blur filter.

Image Deblurring: In this technique we first evaluate how much higher frquency component is eliminated from the frame compared to nearby frames. We calculate the blurriness measure and determine which frames are relatively blurry. We then use this information to transfer sharper pixels from nearby frames[11][12] to respective blurred image areas in the current frame.

1.4 Thesis Organization

The remainder of this thesis is organized as following: Chapter 2 is literature review and the previous works on video stabilization, motion inpainting and image deblurring are studied. Chapter 3 presents the work of video stabilization using a homography model to describe the geometric transformation among the two frames. Chapter 4 discusses the Video Completion technique and motion inpainting. Chapter 5 details the work on Image deblurring. Chapter 6 summarizes the thesis with a discussion of limitations and the scope for future research in this direction.

Chapter 2

Literature Review

Depending on the motion model used, video stabilization can be categorized in to 3D, 2D and 2.5D techniques. Also, the problem of video completion to yield full frame stabilized result and that of image deblurring are also important when building a good video stabilization system.

3D techniques: These techniques need explicitly defined 3D structures for video stabilization, including 3D camera poses and scene depth. These structures can be defined from SFM algorithms or by using depth sensors. The 3D camera path is determined from these structures and smoothing is applied to it to remove the shaky component. The stabilized video is obtained by rendering the original sequence with the modified path as if it is taken from a new path. This rendering process is referred to as novel view synthesis. When 3D reconstruction is feasible, it often produces the highest quality stabilized video because of its theoretical and practical correctness.

2D techniques: 2D techniques estimate a linear transformation among the nearby video frames. By accumulating the rigid transforms from these linear transforms, the camera path in 2D space is obtained. The stabilized video is obtained by smoothing this 2D camera path using a filter. Affines and homographies are the most common 2D transformations used in this technique. A homography model is only valid for planar and pure rotational motions and is invalid for scenes with large depth variations. Using 2D stabilization with such videos will result in content distortion. Nevertheless, 2D techniques are more robust and only requires that features math among the nearby frames. This research is focused on motion estimation and smoothing using 2D techniques of video stabilization.

2.5D techniques: These techniques relax the requirement of full 3D reconstruction to some partial 3D information like epipolar geometry[3]. The 3D information is embedded in the feature trajectories. 2.5D techniques can produce results comparable to that of full 3D techniques at a lower computational cost. Nonetheless, the requirement of long feature tracking is still a barrier for the robustness of this technique.

Video completion techniques: The quality of a stabilized video is highly dependent on the technique used for video stabilization. Most of the time, this technique is simply cropping the video to remove the empty regions in the image and up scaling the resulting image to match the original resolution. In this section we explore the possibility of other completion techniques which allow the creation of a video stabilizer which does not need clipping and can produce full frame videos. This is done by locally adjusting the image pixels from nearby frames to populate the empty region in the frame. One such technique called motion inpainting[1] is deeply explored in this work.

Image deblurring techniques: After the process of stabilization, the motion blur which is unassociated with the new video sequence becomes very noise like and distracting. This needs to be eliminated to improve the quality of the image. Previously, this was achieved by obtaining Point Spread Functions in order to sharpen the frames and reduce the motion blur. This technique is difficult and is not very reliable in producing good results. In this work, we explore other types of Image deblurring techniques that are not based on PSFs.

In the following sections, we briefly review the prior works based on the categories. We highlight one representative work for each category.

2.1 3D Video Stabilization

3D stabilizing techniques estimate 3D motion of camera for stablization. Beuhler er al.[13] introduced a 3D video stablization technique that is built on projective reconstruction of the scene with an uncalibrated camera. Whenever Euclidean reconstruction is feasible, Zhang et al.[14] introduced trajectory smoothing of the camera to decrease the acceleration in rotation, translation and zooming. Liu et al.[15] introduced a full 3D stabilization technique by introducing content-preserving warps(IPW) for the rendering of the 3D motion of camera along the new path. Zhou et al.[16] further extended the content-preserving warps with plan-based constraints. These techniques are generally limited by their adopted 3D reconstruction algorithms. Although there is a good amount of progress in 3D reconstruction, reconstruction of a general video is still hard. We briefly review the technique of content-preserving warp next.

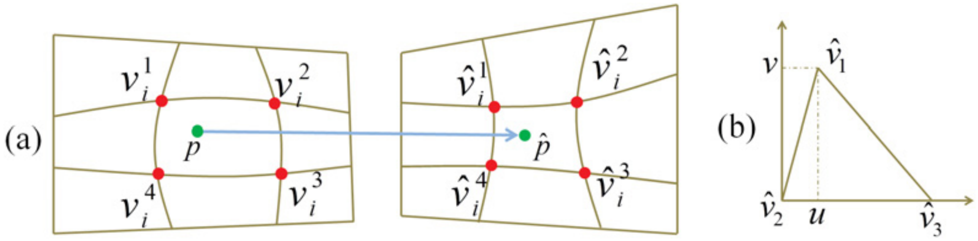


Figure 7: (a) A pair of matched features $(p, \hat{p})$ should be represented by the same set of bi-linear interpolation weights of their four enclosing vertices. (b) The smooth term requires each triangle $\hat{\nu}_1, \hat{\nu}_2, \hat{\nu}_3$ to follow a similarity transformation.

Content-Preserving Warp:

Liu et al.[15] introduced the content-preserving warp for the novel view synthesis. This technique was inspired by as-rigid-as-possible shape manipulation[17]. Given the input video frame $\hat{I}_t$, the corresponding output video frame I_t is generated by a warp from $\hat{I}_t$. 3D reconstruction provides a sparse set of 3D points. They can be projected onto both the input and output cameras, giving two sets of respective 2D points. $\hat{P}$ on the input frame and P on the output frame.

data term: Assume $\{p, \hat{p}\}$ is the p-th matchd feature pair from input and output frame respectively. Then p can be indicated by a 2D bi-linear intrpolation of the four vertices $V_P = [\nu_p^1, \nu_p^2, \nu_p^3, \nu_p^4]$ of the enclosing grid cell; $p = V_p w_p$, where $w_p = [w_p^1, w_p^2, w_p^3, w_p^4]^T$ are intrpolation weights that add up to I. The coresponding feature $\hat{P}$ can be indicated by the equal weights of the warped grid vertices $V_P = [\hat{\nu}_p^1, \hat{\nu}_p^2, \hat{\nu}_p^3, \hat{\nu}_p^4]$. Figure 7(a) shows the relationship. So, the data term is defined as

$$E_d(\hat{V}) = \sum_P ||\hat{V}_p w_p - \hat{p}||^2 \tag{1}$$

Here $\hat{V}$ consists all the warped grid vertices.

similarity transformation term As illustrated in Figure 7(b), the similarity term is described as

$$E_s(\hat{V}) = \sum_{\hat{v}} ||\hat{v} - \hat{v}_1 - sR_{90}(\hat{v}_0 - \hat{v}_1)||^2, R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{2}$$

where $s = ||v - v_1||/||v_0 - v_1||$ is a already recognised scalar computed from the primary mesh. This similarity transformation term needs the triangle of nearby vertices v, v_0, v_1 undergoes a similarity transformation.

The final energy $E(\hat{V})$ is obtained by combining two terms.

$$E(\hat{V}) = E_d(\hat{V}) + \alpha E_s(\hat{V}), \tag{3}$$

where α is a weight to direct the quantity of regularization. This energy equation is quadratic and can be minimized by solving a sparse linear system. Content preserving warp is applied to warp a frame to its novel view point. It shows greater advantage over traditional image based rendering techniques.

2.2 2D Video Stabilization

2D video stabilization techniques mostly use homography model or affine transformations to estimate the motion of camera. Then these transformations are smoothed using a filter to stabilize the shaky video. If the transformation among two consecutive frames can be described by homography model, then the relationship among the two images $I(P)$ and $I^/(P^/)$ can be described by $p \sim Tp^/$. $p = (x, y, 1)^T$ and $p^/ = (x^/, y^/, 1)^T$ are local of the pixels in projective cordinates, and ˜ describes equality up to scale as the 3x3 matrix T is not effected by scaling.

Hierarchial motion estmation framework:

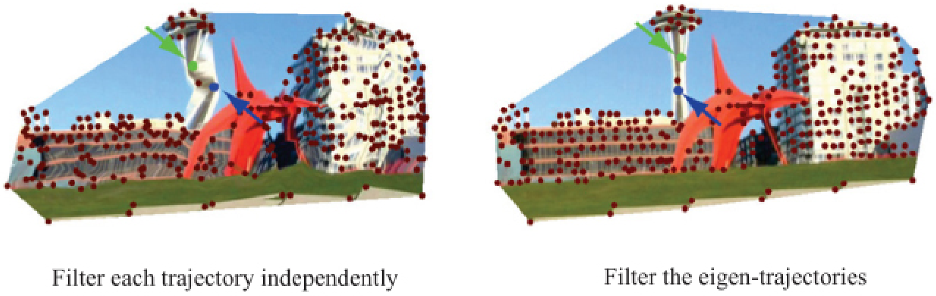
This is introduced by Bergen er al.[5]. By implementing the parameter estimation for each pair of side by side frames, a global trnsformation chain is extracted. This transformation chain contains both the high frequency motion and low frequency motion of camera. As assumed in [18], we define the intentional motion of camera in the video as pleasent, lengthy and smooth. As this corresponds to the low frequency component, we remove the high frequency part from the global motion chain as unintended motion. In other techniques, as smoothing is implemented to the original transformation chain T_0^1.....T_{i-1}^i, the eased transformation chain $\hat{T}_0^1$.....$\hat{T}_{i-1}^i$is calculated. In this scenario, a motion compnsated frame $I_i^{'}$ is computed by transforming I_i with $\prod_{n=0}^{i} T_{n+1}^{n} \bar{T}_n^{n+1}$. This flow of original and smoothed trnsformation chain usually creates accumulation error. But our technique is free from such error as it locally smooths displacement from the present frame to the nearby frames.

2.3 2.5D Video Stabilization

This technique uses partial 3D information to smooth the trajectory of the feature points that are tracked. There are several developments in this area.

Goldstein and Fattal[19] used an “epipolar transfer” method to prevent the difficult process of 3D reconstruction. Wang er al.[20] represented each trajectory of the feature point as a Bezier curve and smoothed with a spatial-temporal optimization. Liu et al.[8] smoothed some basis trajectories of the subspace[4] extracted from the feature tracks(longer than 60 frames). This technique produces a result that is comparable to the full 3D techniques, while decreasing the requirement from 3D reconstruction to long feature trajectories. This technique is also used as “Warp stabilizer” in the commercial software Adobe After Effects. Recently, this Liu et al.[8] extended the subspace technique to also deal with stereoscopic videos. The basic ideas of subspace[4] is one of the representative work in 2.5D techniques.

Subspace Video Stabilization:



Filter each trajectory independently

Filter the eigen-trajectories

Figure 8: Subspace Low-path filtering. Left: filter each trajectory independently introduce artifacts as ignoring of 3D information. Right: filter eigen-trajecotries in the subspace. The figures are borrowed from [4]

We intend to find the proper positions for a given set of 2D point trajectories, at the output frame to produce a stabilized video. The trajectories can be expressed together as a trajectory matrix M:

$$M_{2N*F} = \begin{bmatrix} x_1^1 & x_2^1 & ... & x_F^1 \\ y_1^1 & y_2^1 & ... & y_F^1 \\ & & & \\ x_1^N & x_2^N & ... & x_F^N \\ y_1^N & y_2^N & ... & y_F^N \end{bmatrix} \tag{4}$$

with N features per frames and F frames in total. If we directly apply a low pass filter to this matrix, there would be distortion since smoothing feature trajectories independently will breakdown the relationship among the points. Figure 8 shows such an example. To maintain this relationship during the process of smoothing the trajectories, a subspace constraint is introduced. Usually, trajectories of motion from a perspective camera will lie on a non-linear manifold. We can calculate the probable manifold locally with a linear subspace. Irani [21] showed that the trajectory matrix should have a rank of at most 9. Such a low rank constraint implied that the trajectory matrix M can be factored into the product of two lower rank matrices:

$$M_{2n*k} {\approx} W.*(C_{2n*r}E_{r*k}) \tag{5}$$

where W is a binary mask matrix that indicates missing or unknown data, and .* indicates component wise multiplication. E is the eigen trajectories and C contains the coefficient for the linear combination. Output frames can be obtained by content preserving warp guided by the control points in M and $\hat{M}$. Figure 8 right shows an example. With the subspace constraint, the relationship among the features are appropriately preserved.

2.4 Video Completion

This technique usually completes or finalizes the video by either trimming the edges to remove the empty regions caused by video stabilization or by using more advanced techniques such as image inpainting[22] to fill the missing area. The first technique results in the reduction of resolution of the video. Video completion techniques based on image inpainting do not produce consistent results and so, are not reliable. These image inpainting techniques fail when there is a large movement in the video or when there are too many moving foreground elements. They also require huge computational resources and take a long time for inpainting the entire video. To address this problem, A.Telea[23] introduced a new image inpainting algorithm based on fast marching technique. FMM presents as a solution to the Eikonal problem.

Mathematical model of Fast Marching technique:

From figure 9, Consider that we need to inpaint the point p which is situated on the boundary $\partial\Omega$ of the region Ω. Taking a small neighborhood $B_{\varepsilon}(p)$ of size ε of the known image around p. As described in [22], the inpainting of p needs to be done by examining the intensity values of the image points that are known and are close to p which is the neighborhood $B_{\varepsilon}(p)$. For small enough ε, we can assume a first order approximation $\mathrm{I}_q(p)$ of the point p in image, given the image $I(q)$ and gradient $\nabla I(q)$ values of point q.

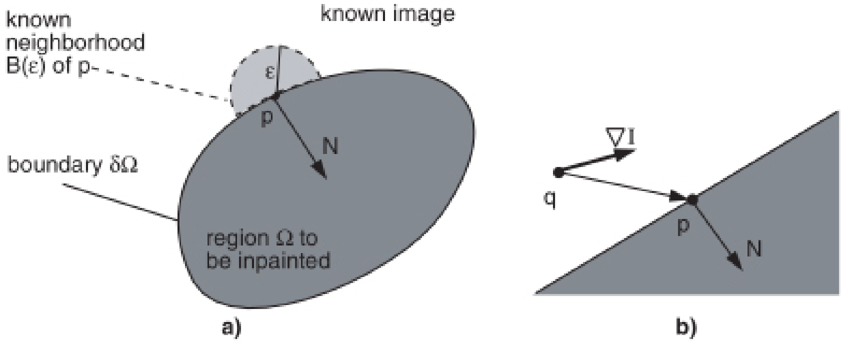


Figure 9: The inpainting technique. Figure is borrowed from[23]

$$Iq(p) = I(q) + \nabla I(q)(p{-}q) \tag{6}$$

In further step, we use inpainting on point p as a function of points q in the neighborhood $B_\varepsilon(p)$ by the summation of the estimates of all points q, that are weighed by a normaliezed weighting function $w(p, q)$.

$$I(p) = \frac{\sum_{q \in B\varepsilon(p)} w(p, q)[I(q) + \nabla I(q)(p{-}q)]}{\sum_{q \in B\varepsilon(p)} w(p, q)} \tag{7}$$

The weighing function w(p,q) is selected such that the inpainting of p propagates the intensity value along with the sharp details of image over $B_\varepsilon(p)$. The inpainting and extension to color images are further discussed in Chapter 4.

2.5 Image Deblurring

In the video stabilization process, motion blur that is not related with the new modified sequence of video frames arises due to the inherent motion blur present in the input video sequence. In the final output, this becomes very noticeable and unnatural. So, removing the motion blur is an important step in improving the quality of a stabilized video. To do this, we need to sharpen frames where the motion blur is present. This boils down to a problem of Image deblurring which mostly uses a Point Spread Function to estimate the blurriness in the image and deconvolutes the image accordingly. But estimating the PSFs from the blurred image is a difficult process. Tanaka [24] developed a technique to accurately estimate the PSFs in the scenario of linear motion blur. Jeong Ho Lee[25] improved upon this work to introduce a technique for PSF parameters estimation by using the periodicity of motion blurred images in the frequency domain. This technique is effective for both Noise less and Noisy images. In the following, we briefly describe the Motion Blur Parameter Estmation in Noise less Images. However, estimating PSFs accurately for an unknown type of motion blur is still highly difficult and time consuming.

Motion Blur Parameter Estimation in Noise Less Images:

If G(u,v), F(u,v), and H(u,v) are the frequency responses of the observed image, original image and the degradation function respectively, then when the noise is absent, [25] concludes that

$$G(u,v) = F(u,v)\cdot H(u,v) \tag{8}$$

The parameters of the motion blur can be determined as follows:

Motion Direction Estimation:

The paralel dark lines that occur in the Fourier spectrum showed in Figure 10 are used. From [26], it is clear that the direction of motion blur (φ) is equal to the angel (ϑ) among any of these paralel dark lines and the vertical axis. So, to find the direction of the motion, it is sufficient to find the parallel lines direction. We can use Radon transform in either of the following forms to fit the line and determine the direction.

$$R(\rho,\theta) = \int_{\infty}^{-\infty}\int_{\infty}^{-\infty} g(x,y)\delta(\rho - xcos\theta - ysin\theta)dxdy \tag{9}$$

$$R(\rho,\theta) = \int_{\infty}^{-\infty} g(\rho cos\theta - ssin\theta, \rho sin\theta + scos\theta)ds \tag{10}$$

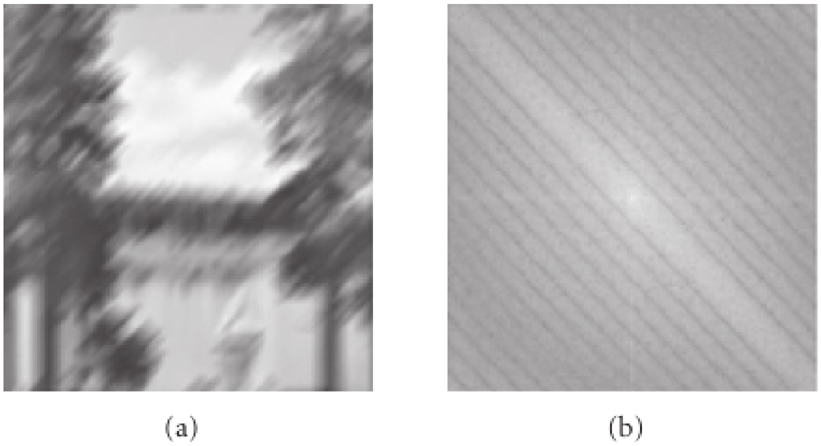


Figure 10: (a) The lake image that is distorted by linear motion blur using L = 20 pixels, $\varphi = 45 \circ$, (b) Fourier spectrum of (a). Figure borrowed from [25]

Motion Length Estimation:

After obtaining the direction of motion, the cordinate system of log | G(u,v) | is rotated, instead of rotating the perceived image, to line it up with the direction of motion. This solves the problems of intrpolation and out of range pixels. Due to the effect of rotation, some components of the Fourier spectrum will appear in the areas out of the cordinate system support, as a consequence the same amount of valid data will not be available in all columns in the new coordinate system. Most of correct data is present in the column that is passing through the centre of frequency. The method presented is built to operate on the central peaks and valleys in the Fourier spectrum, therefore this rotation has no affect on preciscion and robustnes of the algorithm.

In this scenario, uniform motion blur is one dimensional.

$$h(i) = \begin{cases} \frac{1}{L} & if - \frac{L}{2} \leq i \leq \frac{L}{2} \\ 0 & Otherwise \end{cases} \tag{11}$$

The continuos Fourier transform of h is a SINC function.

$$Hc(u) = \frac{2Sin(u\pi L/2)}{u\pi L} \tag{12}$$

The discretetized variety of H in horizontal direction is:

$$H(u) = \frac{Sin(Lu\pi/N)}{LSin(u\pi/N)}, \qquad 0{\leq}u{\leq}N{-}1, \tag{13}$$

Where N is the image size. To find L, the equation H(u) = 0 is solved.

$$Sin(\frac{Lu\pi}{N}) = 0 \tag{14}$$

$$u = \frac{k\pi}{LW} \qquad such\ that\ W = \frac{\pi}{N}, K > 0. \tag{15}$$

If u_0 and u_1 are two respective zero points in a way that $H(u_0) = H(u_1) = 0$, then

$$u1{-}u0 = \frac{N}{L} \tag{16}$$

which results in

$$L = \frac{N}{d} \tag{17}$$

where d is the distance among the two successive dark lines in $log(|G(u,v)|)$.To compute d,we should use the first group of u.

Chapter 3

2D Video Stabilization

3.1 Introduction

In 2D techniques of video stabilization, a geometric transformation model is used to estimate the motion of the camera and create the transformation chain. Then these transformations are concatenated to obtain the camera path in 2D space. Then this path is smoothed using a low pass filter to remove the unnecessary shake from the video. Using the modified path, a new transformation chain is created by subtracting the new path from the existing transformation chain. These new transformations are applied to the video sequence to the respective frames to obtain the final stabilized video. However, this video contains empty regions and has motion blur not associated with the motion of the frames.

In this section we detail the technique used for stabilizing the videos before sending them to the video completion system. We adopt a affine transformation model and estimate the motion among the two nearby frames or images $I(P)$ and $I^/(P^/)$. This relationship can be described as $p \sim Tp^/$. $p = (x, y, 1)^T$ and $p^/ = (x^/, y^/, 1)^T$ are locations of pixels in projective coordinates, and $\sim$ denotes equality up to scale since the 3x3 matrix T is not effected by scaling. The next section describes in detail the technique used for Global motion estimation.

3.2 Global Motion Estimation

Global motion is obtained by aligning two consecutive frames at a time presuming a geometric transformation model. In the current technique, we use affine model to describe relationship among the two frames. We utilize the hierarchial motion estmation framework that is introduced by Bergen et al. [5]. We apply this framework and estimate the parameters for all the pairs of nearby frames to establish a global transformation chain.

We indicate the location of the pixel in the image coordinateI_t as p_t . The subscript t denotes the index of the frame. We also indicate the global transformation T_i^j to represent the coordinate transformation from frame i to j. And so, the transformation of image I_t to the I_{t-1} coordinate can be described as $I_t(T_t^{t-1}p_t)$. Note that transformation T only denotes the cordinate transform, hence $I_{t-1}(T_t^{t-1}p_t)$ has the values of pixel from frame $t-1$ in the cordinates of frame t.

3.3 Local Motion Estimation

Local motion is that component in video that departs from the global motion. e.g., large depth variations or moving foreground objects. This motion is obtained by calculating optical flow. This is done among the frames after application of global transformation using only the common area that is covered and is present among both the frames. To do this, we use Lucas-Kanade's pyramidal optical flow computation[9] to compute the optical flow field $F_t^{t'}(p_t) = [u(\mathbf{p}_t)v(\mathbf{p}_t)]^t$. $F_t^{t'}(p_t)$. This indicates an optical flow from frame $I_t(p_t)$ to $I_t(T_{t'}^{t}p'_t)$, and u and v indicate the flow vector along the x- and y-direction respectively in $\mathbf{p}_t$ cordinates. Local motion is not used in the process of video stabilization but needs to be processed at this stage for the simplicity. But this plays a very important role in the video completion stage.

3.4 Motion Smoothing

To obtain a stabilized motion path, the undesired motion fluctuations in the video needs to be removed. From [2] we know that the intentional motion in the video sequence is generally pleasent, lengthy and smooth. So the high frequency part in the global motion chain is defined as the unintended shaky motion. Smoothing techniques in past works directly smoothed out the transformation chain or cumulative transform chain by using a base frame. In our technique, we smooth the local displacement instead to obtain a smooth motion.

If the technique of smoothing the original transformation chain $T_0^1.....T_{i-1}^i$ is followed, the smoothed transformation chain $\hat{T}_0^1.....\hat{T}_{i-1}^i$ is calculated. In this scenario, a motion compensated frame $I_i^{'}$ is computed by transforming I_i with $\prod_{n=0}^{i} T_{n+1}^{n}\bar{T}_{n}^{n+1}$. This flow of original and eased transformation chain usually creates an accumulation error. But our technique is free from such error as it locally smooths displacement from the present frame to the nearby frames.

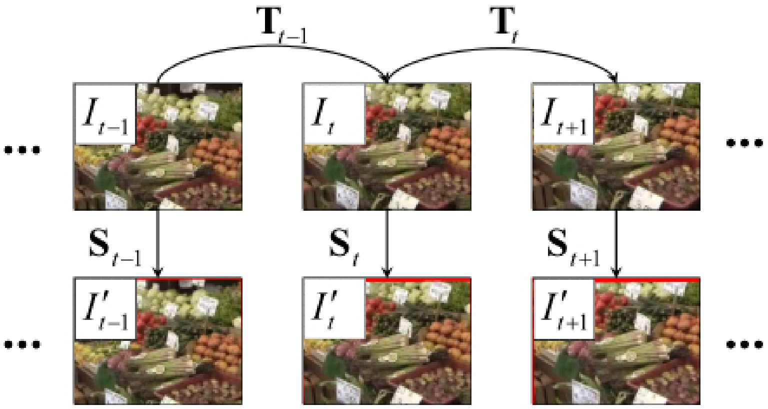


Figure 11: Global transformation chain and the transformation from original trajectory to the smoothed trajectory. Figure borrowed from [1]

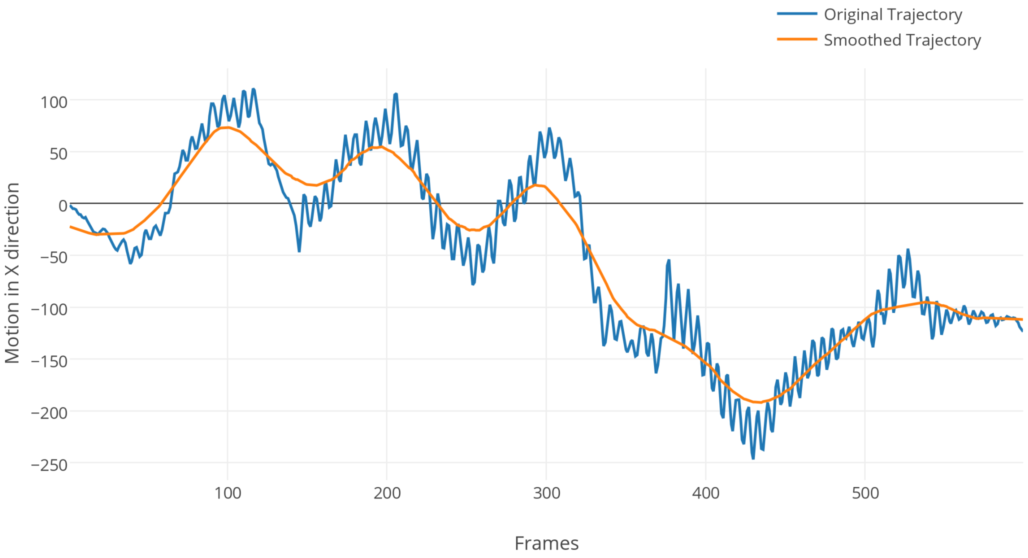


Figure 12: Original and smoothed trajectories of a sample video

Without applying the smoothing on the original transform chain, we calculate the transform $\mathbf{S}$ from one frame to the respective motion compensated frame utilizing just the nearby transform matrices. The indices of nearby frames are denoted as $N_t = \{j|t-k \leq j \leq t+k\}$. If the frame I_t is present at the origin, and is lined up with the major axes, We can compute the position of each of the nearby frame I_s , in relation to frame I_t , by the local displacement $\mathbf{T_t^s}$. We can find the modifying transform $\mathbf{S}$ from the original video frame I_t to the motion compensated video frame $I_t^{'}$ according to the following

$$\mathbf{S_t} = \sum_{\mathbf{i} \in \mathbf{N_t}} \mathbf{T_t^i} \bigstar G(k), \tag{18}$$

where $G(k) = \frac{1}{\sqrt{2\pi\sigma}} e^{-k^2/2\sigma^2}$ is a Gaussian kernel, and the $\bigstar$ operator represents convolution and $\sigma = \sqrt{k}$ is used. Using the computed matrices

$S_0, ..., S_t$, the input frames can be warped to the motion compensated stabilized frames by

$$I_t^{'}(\mathbf{p}_t^{'}) \leftarrow I_t(\mathbf{S}_t\mathbf{p}_t) \tag{19}$$

In, Figure 12 we can observe the result of the motion smoothing technique that we applied. In the results, both x- and y-translation elements and rotation elements of the motion of camera paths are displayed. As it can be observed from the figure, sudden changes in the motion of camera which are considered unwanted are reduced by the smoothing process. This smoothness can be changed by varying k and with increasing k, a smoother stabilization result can be expected. k=6 corresponds to about 0.5 sec interval in NTSC format videos. If a smoother video is preferred, the value of k can be increased.

3.5 Experimental Results

To evaluate our technique, we have implemented our 2D video stabilization algorithm on a number of shaky videos. Results of two samples are presented here.

3.5.1 Sample 1 : inter_iit.avi

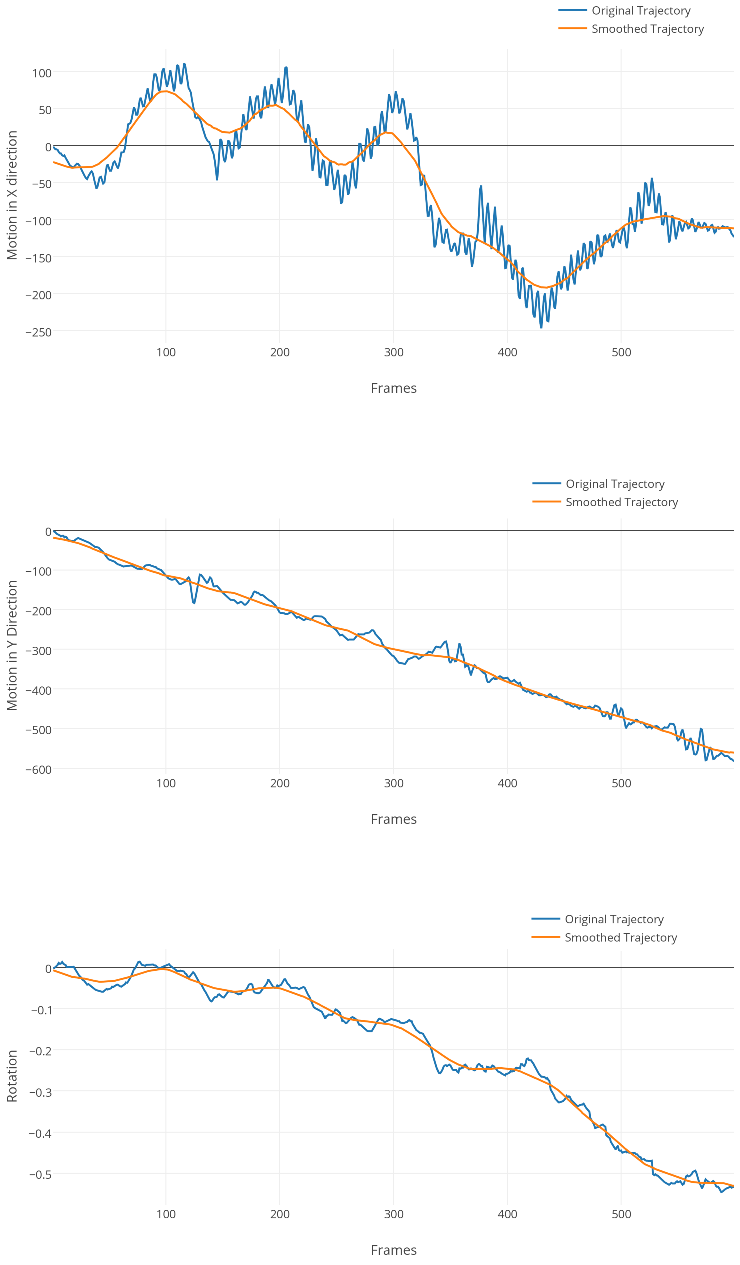


Figure 13: Original and Smoothed Trajectories of Sample video sequence 1

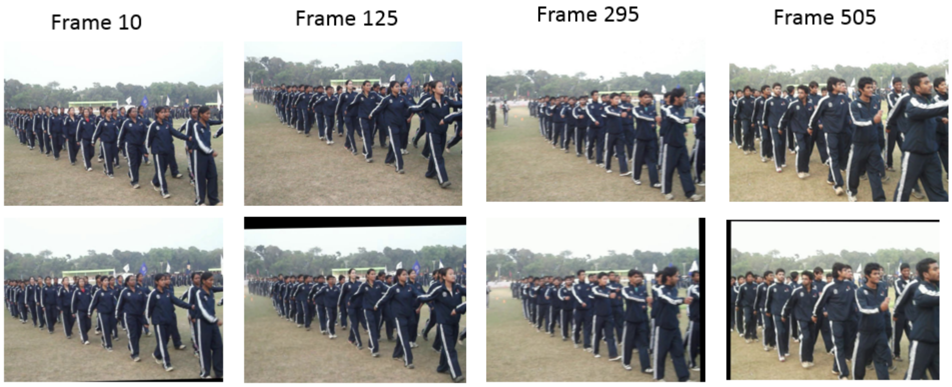


Figure 14: Comparision of frames from original sequence and 2D stabilized sequence

3.5.2 Sample 2: SANY0016.avi

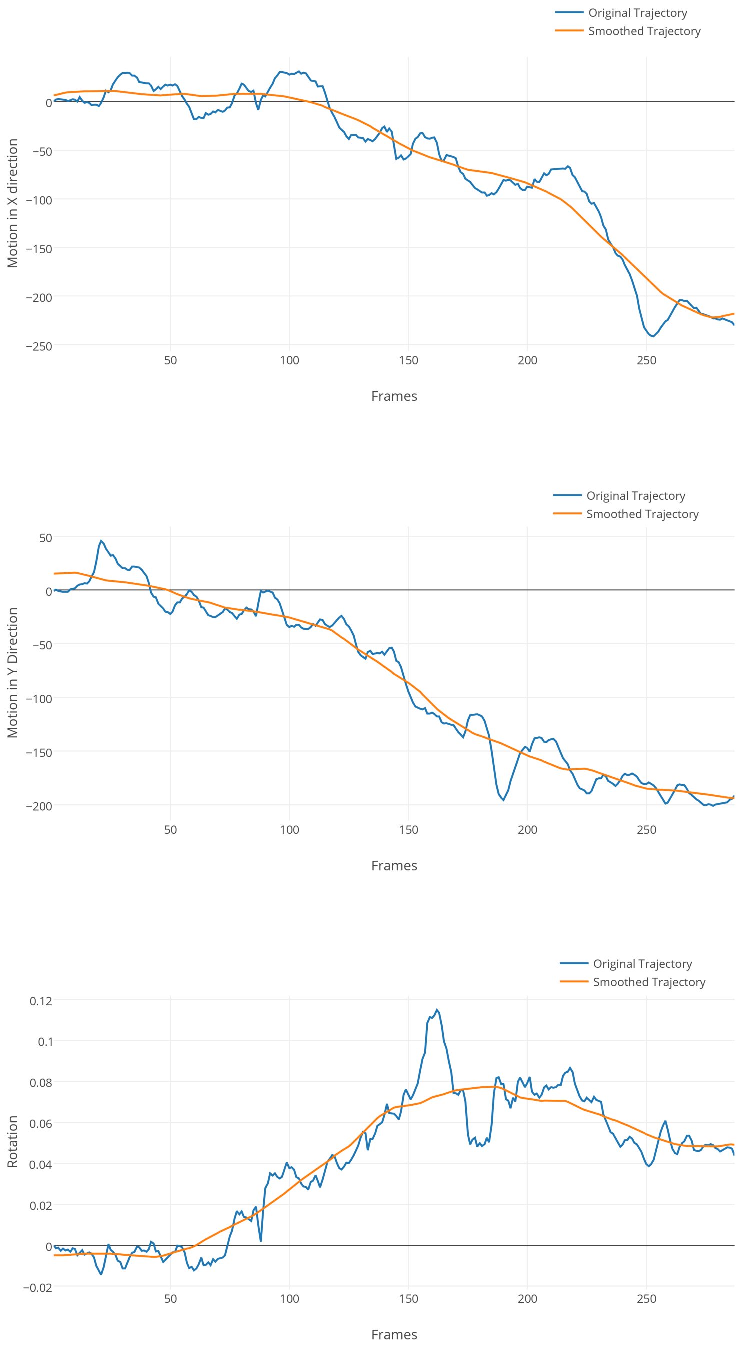


Figure 15: Original and Smoothed Trajectories of Sample video sequence 2

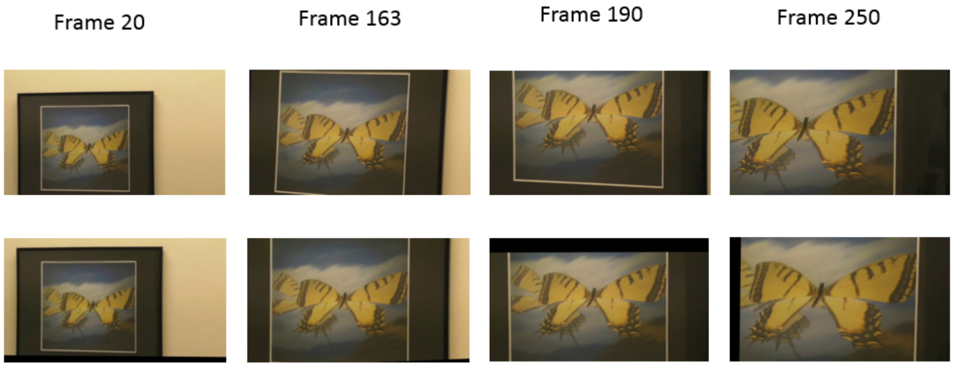


Figure 16: Comparision of frames from original sequence and 2D stabilized sequence

3.6 Conclusion

We used affine transformation model to describe motion among frames and we applied parametric estimation of transformations among each set of nearby frames to smooth the shaky motion. We were able to stabilize the sequence albeit the empty spaces due to the transformation of the frame. This problem will be rectified in the next section.

Chapter 4

Video Completion

4.1 Introduction

A video completion technique based on the method of motion inpainting[1] is implemented. The main idea behind this technique is to propagate the local motion, substituting color or intensity as commonly seen in image inpainting[22], in to the areas with missing pixels. The propogated motion is then used to fill in the area of the image with empty pixels naturally. This technique also works for scene which have non-planar or dynamic content. Using this motion information as a guide, pixel data from the nearby frames are warped locally to keep the spatial and temporal consistencies in the modified frame the same. Shum and Szeliski[27] introduced a de-ghosting algorithm in which a panorama image is constructed by warping image based on local motion. This technique is different in the sense that the local motion is propogated in to an area where the local motion field cannot be computed directly.

4.2 Completing the Video with Motion Inpainting

In this technique, we locally adjust the image mosaics by utilizing the local motion field so that we can get seamless stitching of the mosaics in the image areas with missing pixels. At the core of this technique, we use motion inpainting to compute the local motion of the missing pixels using the local motion data available in the nearby frames. We assume that the local motion of the missing pixels is similar to that of pixels in adjoining image areas. The flowchart of this algorithm is presented in Figure 13.

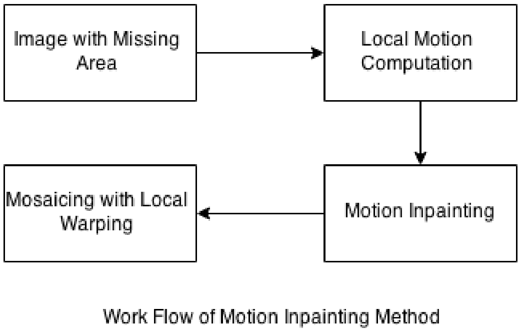


Image with Missing Area
Local Motion Computation
Motion Inpainting
Mosaicing with Local Warping
Work Flow of Motion Inpainting Method

Figure 17: Flow Chart of motion inpainting

Initially, the local motion of the nearby frame is calculated over the common image area that is covered in both the frames. The local motion field is then propogated into image areas with missing pixels. Unlike the previous works in image inpainting, we only propagate the motion field instead of intensity or color. Finally, this local motion is used as a guide to locally warp the image mosaics to get smooth stitching of the mosaics.

If total missing image area in a particular frame is represented by M_t, we wish to completely fill this area for every frame t with good quality. The following steps describe procedure in which we fill the missing image pixels.

4.2.1 Mosaicing with consistency constraint

This is the starting step in completing the video. In this technique, we initially try to fill the missing image pixels that belong to the non dynamic and static regions of the frame. A mosaic of the pixel with respect to the pixels in same location in the nearby frames is taken with an evaluation of its validity. When the 2D video completion step has been completed successfully and if the missing pixel belongs to the static image area, the mosaic that we have got should be consistent with the image area with missing pixels. The mosaic obtained can be evaluated by examining the consistency of the different mosaics which cover the same area in the neighboring frames. The variance of the mosaic pixels is taken as a measure of consistency. With an increase in variance, the reliability of the obtained mosaic decreases. For each pixel $\mathbf{p}_t$ in the image area with missing pixels M_t, the variance of the mosaic pixel values is calculated as follows:

$$\upsilon_t(\mathbf{p_t}) = \frac{1}{n-1} \sum_{t^! \in N_t} [I_{t^!}(\mathbf{T_t^{t^!}} \mathbf{p_t}) - \bar{I}_{t^!}(\mathbf{T_t^{t^!}} \mathbf{p_t})]^2 \tag{20}$$

where

$$\bar{I}_{t'}(\mathbf{T_t^{t'}} \mathbf{p_t}) = \frac{1}{n} \sum_{t' \in N_t} I_{t'}(\mathbf{T_t^{t'}} \mathbf{p_t}) \tag{21}$$

and n is the total number of nearby frames that we taken in to consideration. For colored images, we utilize the intensity values of the image pixel which is calculated by 0.30R+0.59G+0.11B[17]. A pixel $\mathbf{p}_t$ is filled in by the median value of all the warped pixels only if calculated variance is lesser than a preset threshold value T:

$$I_t(p_t) = \begin{cases} median_{t'}(I_{t'}(\mathbf{T_t^{t'}} \mathbf{p}_t)) & if\ \upsilon_t < T \\ missing & otherwise \end{cases} \tag{22}$$

If all of the pixels in M_t are filled with this step, then we can leave the next steps and directly proceed to the succesive frame.

4.2.2 Local motion computation

In this step, every nearby frame $I_{t'}$ is given a priority score based on the alignment error. Generally, it is observed that the most nearby frame has a smaller alignment error, and so it is given a higher priority for processing. The alignment error is calculated by using the common area that is covered between $I_t(p_t)$ and $I_t(T_t^{t'} p_t)$ by the following.

$$e^t_{t'} = \sum_{\mathbf{p_t}} |I_t(p_t) - I(T^{t'}_t p_t)| \tag{23}$$

The process of estimating local motion is already described in the section 3.3

4.2.3 Motion Inpainting

The local motion present in the known image areas is propogated in to the empty image areas. This process starts at the pixels present at the boundary of area containing all the empty pixels in the image. Utilizing the motion values of the nearby pixels obtained from local motion estimation, the motion values for the missing pixels on the boundary are defined. This boundary advances gradually in to the empty image area M till it gets fully filled.

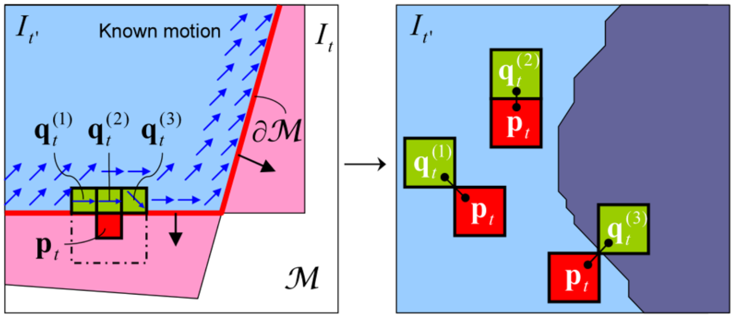


Figure 18: Motion inpainting. The motion field is gradually propogated in to the missing image area until it is completely filled. Image is borrowed from [1]

From figure 14, if $\mathbf{p_t}$ is a pixel belonging to the empty image area M, and $H(\mathbf{p_t})$ is collection of pixels in around the point $\mathbf{p_t}$, which earlier have a motion value that is defined directly by the local motion computation or by extrapolating through motion inpainting. The values for motion for the pixel $\mathbf{p_t}$ is calculated by a weighted average of the motion vectors of the pixels $H(\mathbf{p_t})$.

$$\mathbf{F}_t^{t'}(\mathbf{p}_t) = \frac{\sum_{q_t \in H(\mathbf{p_t})} \omega(\mathbf{p_t}, \mathbf{q_t})\mathbf{F}_t^{t'}(\mathbf{q}_t)}{\sum_{q_t \in H(\mathbf{p_t})} \omega(\mathbf{p_t}, \mathbf{q_t})} \tag{24}$$

In the above equation $\omega(\mathbf{p_t}, \mathbf{q_t})$ indicates the amount of contribution of the value of motion of $q_t \in H(\mathbf{p_t})$ to pixel $\mathbf{p_t}$. We make use of the color similarity for colored images as a measure for motion similarity presuming that the nearby pixels with almost same colors represent to the same object in the video and so, it is assumed that they will move in a common fashion. As the colour of the pixel $\mathbf{p_t}$ is not known in the frame I_t we use the nearby frame $I_{t'}$ to estimate the $\omega(\mathbf{p_t}, \mathbf{q_t})$. As shown in figure 14, $\mathbf{q_{t'}}$are initially located in the nearby image $I_{t'}$ utilizing $\mathbf{q_t}$ and their local motion. Utilizing geometry among $\mathbf{q_t}$ and $\mathbf{p_t}$, $\mathbf{p_{t'}}$ are obtained in $I_{t'}$. Utilizing $\mathbf{p_{t'}}$and $\mathbf{q_{t'}}$, the color similarity is measured by $\omega(\mathbf{p_t}, \mathbf{q_t}) = 1/\{ColorDistance(I_{t'}(\mathbf{p}_{t'}), I_{t'}(\mathbf{q}_{t'})+\epsilon\}$, where ϵ is a some little value to avoid the case of dividing by zero. In this way, the weight factor is calculated utilizing the color similarity, and the value of the motion that is calculated is propogated to $\mathbf{p_t}$. In this method, we calculate the color distance by using the l^2-norm in RGB to save some computing resources. But a better measure could also be used.

Motion inpainting is carried out in our technique by using the Fast Marching technique(FMM)[18] which is detailed in the case of image inpainting by A.Telea[19]. Using FMM, we can visit each undefined pixel, just one time and advance the boundary inside the empty area M till all the unknown pixels are propogated with the values of motion. The pixels are handled in the increasing distance order from the principal boundary, in a way that pixelswhich are closer to the area with known pixels are filled initially. The consequence of using FMM is a smooth extrapolation of the local motion flow to the area with empty pixels in a way that retains the object boundaries with color similarity measure. FMM is described in detail in 4.3.

4.2.4 Mosaicing with local warping

After we obtain the motion filed in the area with the missing pixels M_t, we utilize it as a guide to locally warp the image frame $I_{t'}$ in order to create a smooth mosaic even taking in to consideration the dynamic objects.

$$I_t(p_t) \leftarrow I_{t'}(\mathbf{F}_t^{t'}(\mathbf{p}_t)) \tag{25}$$

If a few pixels still are left empty in the frame I_t, then we go back to step 4.2.2 and use the next nearby frame.

After a loop of steps from 4.2.2 to 4.2.4, all the unknown or missing pixels are generally filled. Nonetheless, in the scenario that there are still a few missing pixels that have not been covered by warped mosaics, we just apply a blurring filter to fill up these areas. Such areas are usually small and better techniques can be used to fill them up at the expense of computing cost.

4.3 Fast Marching technique for Motion Inpainting

Using FMM[23], each pixel of the empty image area M_t can be visited only once and be propogated with motion information from the nearby pixels. This saves computation time and also helps to extrapolate the motion field in a smooth way while preserving the spatial and temporal consistency. In section 2.4, we describe the mathematical model of Fast Marching technique. In the following, we describe the process of using motion inpainting in combination with the FMM algorithm.

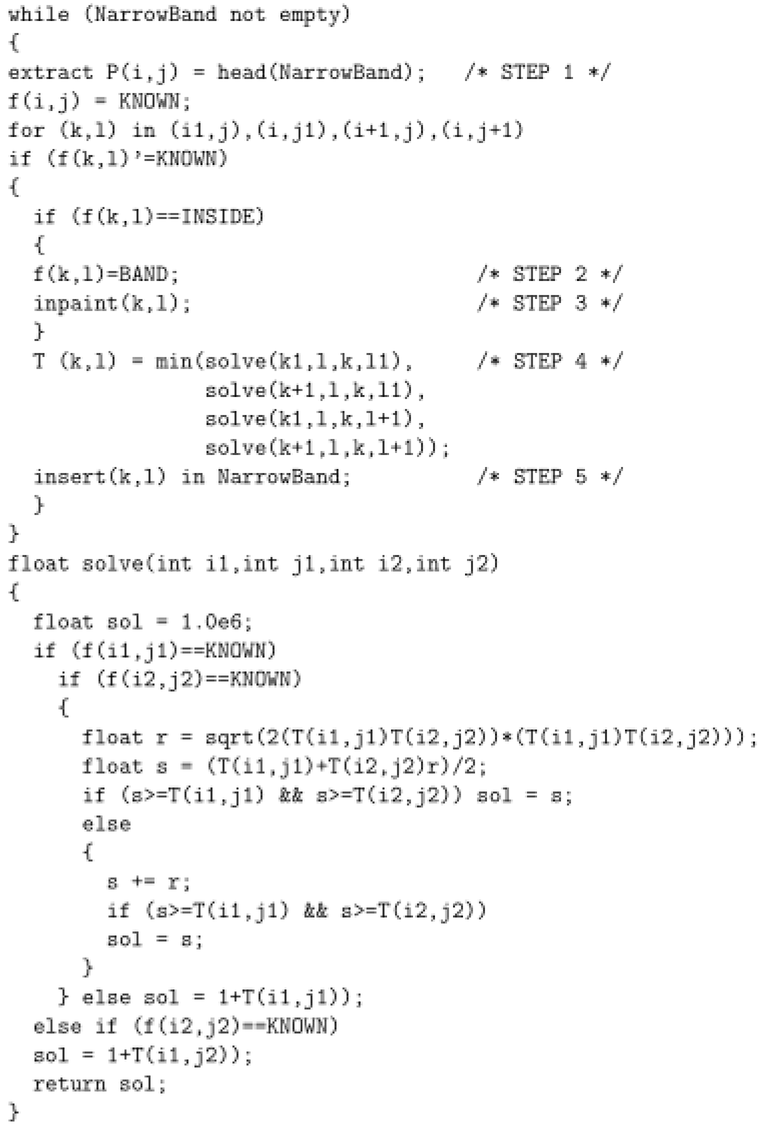
From section 2.4, we know that inpainting points in ascending distance order from $\partial\Omega_i$ make sure that the areas that are nearer to known image pixels are first filled, similar to that of manual inpainting techniques. Implementing the model described in section 2.4 requires an algorithm which would solve the Eikonal equation:

$$|\nabla T| = 1 \quad on\ \Omega, \qquad with\ T = 0\ on\ \partial\Omega. \tag{26}$$

The solution of the above equation is the distance map of Ω pixels to the boundary $\partial\Omega$. The isolines of T are the exact successive boundaries $\partial\Omega$ of the reducing space Ω that we have to inpaint. The normal N to $\partial\Omega$, which is also required for inpainting, is ∇T. The FMM technique ensures that the pixels belonging to boundary are always processed in ascending order of their distance-to-boundary T [23].

FMM technique is better compared to other Distance Transform (DT) techniques which calculate the distance map T to a boundary $\partial\Omega$. The main benefit of FMM is that it seperately stores the narrow band which separates the known pixels from the area of image with empty pixels and specifies which pixel to inpaint next. Other DT techniques do not store this narrow band as it would complicate their implementation. A complete pseudocode of FMM is detailed in figure 15. For every pixel in the image, we store its value T, its intensity value I, and a flag f that can contain one of the below three values:

- BAND: The pixel belongs to narrow band. Its T value is is updated.
- KNOWN: The pixel lies out of the boundary $\partial\Omega$, in the image area with known pixels. Its T and I values are already known.
- INSIDE: The pixel is contained inside $\partial\Omega$, in the area to inpaint. Its T and I values are not yet known.



```
while (NarrowBand not empty)
{
extract P(i,j) = head(NarrowBand);   /* STEP 1 */
f(i,j) = KNOWN;
for (k,l) in (i1,j),(i,j1),(i+1,j),(i,j+1)
if (f(k,l)'=KNOWN)
{
  if (f(k,l)==INSIDE)
  {
  f(k,l)=BAND;                        /* STEP 2 */
  inpaint(k,l);                       /* STEP 3 */
  }
  T (k,l) = min(solve(k1,l,k,l1),     /* STEP 4 */
                solve(k+1,l,k,l1),
                solve(k1,l,k,l+1),
                solve(k+1,l,k,l+1));
  insert(k,l) in NarrowBand;          /* STEP 5 */
  }
}
float solve(int i1,int j1,int i2,int j2)
{
  float sol = 1.0e6;
  if (f(i1,j1)==KNOWN)
    if (f(i2,j2)==KNOWN)
    {
      float r = sqrt(2(T(i1,j1)T(i2,j2))*(T(i1,j1)T(i2,j2)));
      float s = (T(i1,j1)+T(i2,j2)r)/2;
      if (s>=T(i1,j1) && s>=T(i2,j2)) sol = s;
      else
      {
        s += r;
        if (s>=T(i1,j1) && s>=T(i2,j2))
        sol = s;
      }
    } else sol = 1+T(i1,j1));
  else if (f(i2,j2)==KNOWN)
  sol = 1+T(i1,j2));
  return sol;
}
```

Figure 19: Pseudo code for Fast Marching technique

Initially, we set the T value to 0 on and outside the boundary $\partial\Omega$ of the area to inpaint and to a higher value (e.g., 10^6) inside, and we start propogating f over the entire image. All the pixels belonging to narrow band called BAND pixels are inserted in to a Priority Queue in an increasing order of their T values. Next, we propagate the T, f and I values utilizing the code that is shown in Figure 15. Step 4 in the code propagates the value T of point (i, j), to its neighbors (k, l) by solving the finite difference discretization of the Eikonal equation which is given by:

$$max(D^{-x}T, -D^{+x}T, 0)^2 + max(D^{-y}T, -D^{+y}T, 0)^2 = 1 \tag{27}$$

where $D^{-x}T(i, j) = T(i, j) - T(i-1, j)$ and $D^{+x}T(i, j) = T(i+1, j) - T(i, j)$ and in a similar fashion for y. Following [23], we solve the above equation for (k, l)'s four quadrants and return the solution with the smallest value. In the end, Step 5 inserts (k, l) again with its new T in the heap.

4.4 Experimental Results

4.4.1 Sample 1: inter_iit.avi

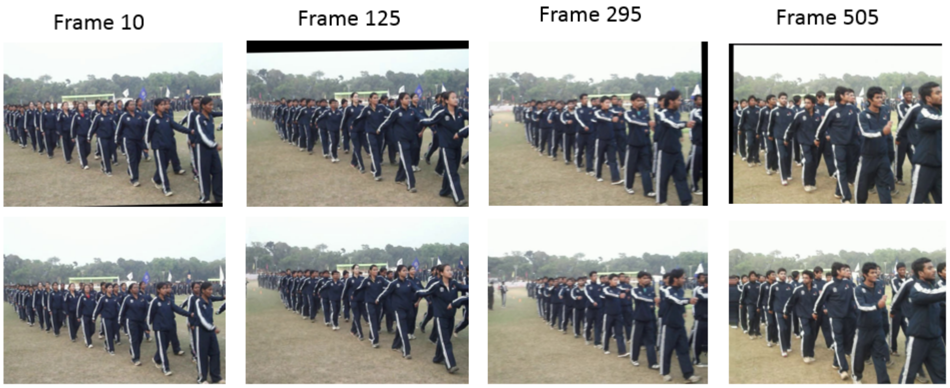


Figure 20: Comparison of 2D stabilized frames with motion inpainted frames for sample 1

4.4.2 Sample 2: SANY0016.avi

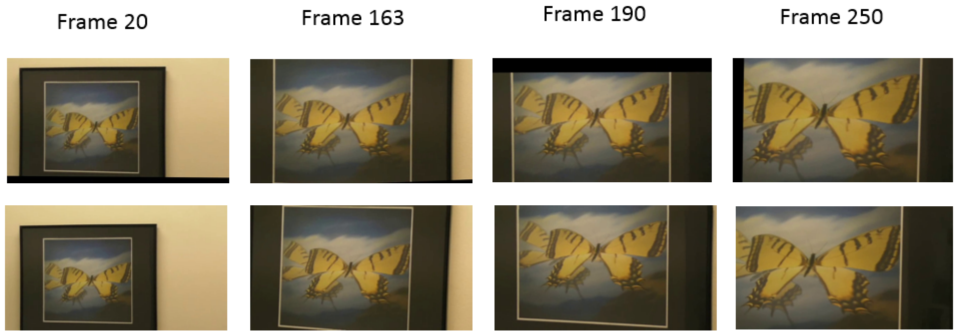


Figure 21: Comparison of 2D stabilized frames with motion inpainted frames for sample 2

4.5 Conclusion

We used a different approach for video stabilization by implementing a motion inpainting based completion technique. In this technique, we warped mosaics from nearby frames using the local motion obtained from optical flow as the guide. A set of steps are followed until we fill all the missing pixels in the video frames after 2D stabilization. It is found the results are quite satisfactory for videos which do not have significant foreground motion.

Chapter 5

Image Deblurring

5.1 Introduction

After the process of video stabilization, the motion blur which is not related to the motion of the new video becomes very noticeable and acts like a noise. As already discussed in section 2.5, it is difficult to accurately estimate Point Spread Functions for a blurred image from a camera that can move freely. So, deblurring the image by using deconvolution and PSF is not possible in this scenario. To sharpen the blurred frames without the help of PSFs, we use an interpolation-based deblurring technique described in [1]. In this technique, we mainly transfer sharper pixels from the nearby frames to the respective pixel locations in blurred frame.

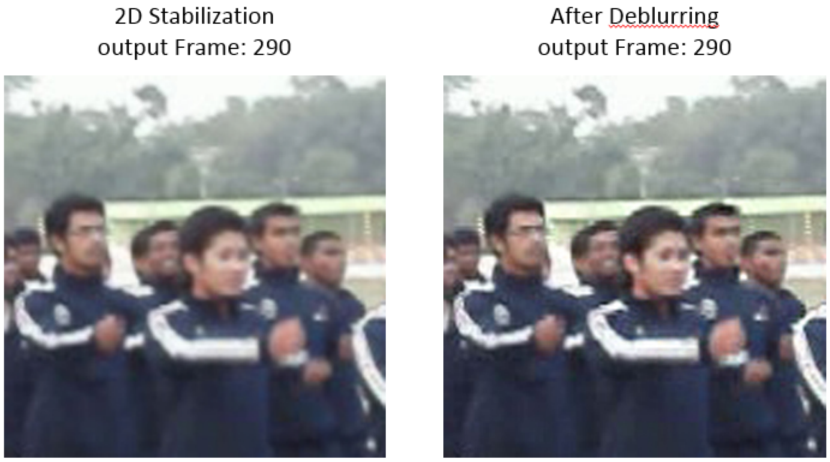
5.2 Determining Relative Blurriness

Relative Blurriness represents the amount of high frequency part that has been eliminated from the frame compared to the nearby frames. Image Sharpness, which is inverse to blurriness, is already studied properly in microscopic imaging where a tight focus is important[11][12]. To evaluate the relative blurriness, we utilize the inverse of *sum of squared gradient measure* due to its robustness to image alignment error and computing speed. By taking two derivative filters along the $x-$ and $y-$directions by f_x and f_y respectively, the measure of blurriness can be defined as

$$b_t = \frac{1}{\sum_{\mathbf{p}_t}\{((f_x \bigstar I_t)(\mathbf{p_t}))^2 + (f_y \bigstar I_t)(\mathbf{p_t}))^2\}} \tag{28}$$

However, this measure of blurriness does not provide the absolute evaluation of image blurriness, but it gives a relative image blurriness among similar images when compared to other blurred images. So, we only use this technique in a limited number of nearby frames where prominent changes in the scene are not observed. Also, the blurriness is calculated using a common coverage area that is observed in all the nearby frames. Relatively blurry frames can be obtained by caclulating the ratio $b_t/b_{t'}, t^{'} \in N_t$. When $b_t/b_{t'}$ is greater than 1, frame $I_{t'}$ is considered as sharper than frame I_t.

5.3 Frame Sharpening



2D Stabilization
output Frame: 290

After Deblurring
output Frame: 290

Figure 22: Image deblurring

Once the relative blurriness is found out, the blurred frames are sharpened by transferring and interpolating the respective pixels from sharper frames. To lower relying on pixels related to dynamic objects, a weight factor that is calculated by pixel-wise alignment error $E^{t}_{t!}$from $I_{t!}$ to I_t is utilized:

$$E^{t}_{t'}(\mathbf{p}_t) = |I_{t'}(\mathbf{T}^{t'}_{t}\,\mathbf{p}_t) - I_t(\mathbf{p}_t)|. \tag{29}$$

Greater alignment error is generally caused either by dynamic objects or error computing of global transformation. Utilizing the inverse of pixel-wise alignment error E as a weight factor for the interpolation, blurred pixels are replaced by interpolating sharper pixels. This process of deblurring can be described by

$$\hat{I}_t(\mathbf{p}_t) = \frac{I_t(\mathbf{p}_t) + \sum_{t' \in N} w^t_{t'}(\mathbf{p}_t) I_{t'}(\mathbf{T}^{t'}_t \mathbf{p}_t)}{1 + \sum_{t' \in N} w^t_{t'}(\mathbf{p}_t)} \tag{30}$$

where w is the weight factor which contains the pixel-wise alignment error $E^t_{t'}$ and relative blurriness $b_t/b_{t'}$, expressed as

$$w^t_{t'}(\mathbf{p}_t) = \begin{cases} 0 & if\ \frac{b_t}{b_{t'}} < 1 \\ \frac{b_t}{b_{t'}} \frac{\alpha}{E^t_{t'}(\mathbf{p}_t) + \alpha} & Otherwise \end{cases} \tag{31}$$

$\alpha \in [0, \infty]$ controls the sensitivity of the alignment error, e.g., by increasing α the alignment error contributes less to the weight. As it is seen in the weighting factor defined above, the interpolation uses only frames that are sharper than the current frame. Figure 16 shows the result of the deblurring algorithm we used.

5.4 Experimental Results

5.4.1 Sample 1: inter_iit.avi

The result is already shown in figure 22 for a particular frame.

5.4.2 Sample 2: SANY0016.avi

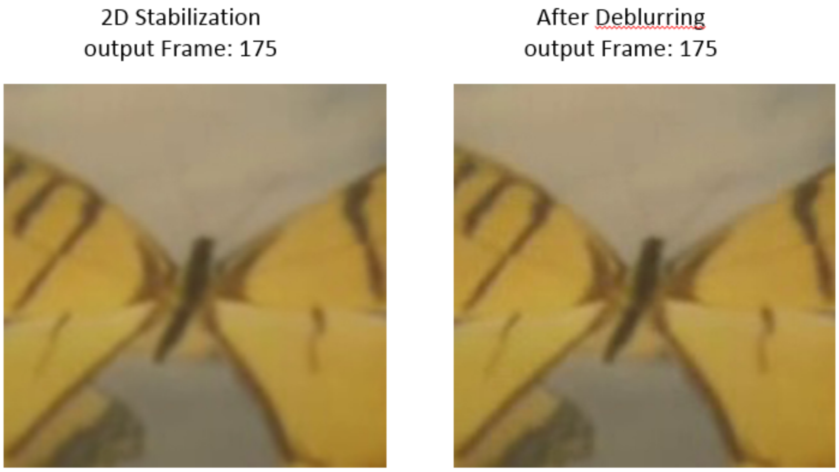


Figure 23: Image deblurring observed in sample 2 frame

5.5 Conclusion

We implemented a deblurring algorithm to get rid of the annoying motion blur and sharpen the blurred frames. It produced satisfactory results.

Chapter 6

Conclusions

6.1 Chapter Summaries

In this thesis, a robust technique for full frame video stabilization with video completion is detailed. In chapter 1, the problem of video stabilization is introduced and the challenges related to the topic were discussed. We described how quick motion of cameras and large moving foreground can pose a significant challenge to the problem of video stabilization. We also introduced the problem of rolling shutter effects and motion blur which are commonly found in videos that require stabilization. We also demonstrated the artifacts that are caused by these challenges on various video stabilization techniques.

According to the adopted motion models, video stabilization can be categorized into 2D, 3D and 2.5D. In chapter 2, we described the common video stabilization approaches in all the three categories and where they are more suitable for use. Additionally, we also described the common video completion techniques used and the problem of Image deblurring and how it can be rectified. In this thesis, a 2D video stabilization technique is implemented along with a video completion technique using motion inpainting that are described in chapter 3 and 4. To sharpen the video, we used a different approach of image deblurring without using PSFs which is described in chapter 5.

Chapter 3 presented a technique for video stabilization in 2D by using a geometric transformation to describe the relationship among the frames. Instead of smoothing the original transformation chain, we smoothed local displacement to avoid the problem of accumulation error. Then we calculate the new transformations to stabilize the video and apply them to the corresponding frames. The resulting stabilized video is passed on to the video completion process then.

Chapter 4 presented a different video completion technique which used motion inpainting to locally warp the image mosaics from nearby frames to fill up the empty regions in the current frame. Motion is propogated by using Fast Marching technique to ensure the smooth extrapolation and the preservation of spatial and temporal consistencies. By following a series of steps, the entire frame is filled with the corresponding intensity value.

Chapter 5 describes an image deblurring technique that is not based on deconvolution using PSFs. Since videos usually come from a free motion camera, it is very difficult to determine the kind of motions that may be present in the video. This makes it very difficult to estimate an accurate PSF that can model the blur. So, instead, we calculate the relative blurriness and then transfer and interpolate sharper pixels from nearby frames to the blurred pixel in the current frame.

6.2 Future Work

There are several directions for work presented in this thesis. The problem of large foreground movement is one, for which a solution could be developed by allowing the user to define locked features which serve as key points in the video, and based on which the stabilization algorithm could be tweaked so that the video is stabilized according to the movement of the background. Advanced motion segmentation could also be used in conjunction with video stabilization to allow only certain features to be in the tracking scheme.

Stabilizing videos using hardware can also be improved. Nowadays, micro controllers and mini PCs are available cheaply and these could be used along with gyroscopes and accelerometers to design a self stabilizing platform or gimbal on which a video capture device could be mounted.

On this note, we would like to conclude this thesis.

References

[1] Y. Matsushita, E. Ofek, X. Tang, and H.-Y. Shum. Full-frame video stabilization. In IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2005. 3, 12, 69

[2] M. Grundmann. Computational video: Post-processing techniques for stabilization,retargeting and segmentation. Doctoral Thesis. Georgia Institute of Technology,2013. 3, 16

[3] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. ACM Transactions on Graphics(TOG), pages 1–10, 2012. xi, xii, 1, 2, 12, 17, 60, 65, 69, 71, 85,89

[4] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. ACM Transactions on Graphics(TOG), 30, 2011. x, xii, 1, 3, 17, 22, 24, 25, 34, 35, 38, 40, 60, 62, 65, 69, 75, 79, 85, 86, 88, 89

[5] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in Proc. of 2nd European Conf. on Computer Vision, 1992, pp. 237–252.

[6] J. Bai, A. Agarwala, M. Agrawala, and R. Ramamoorthi. User-assisted video stabilization. In Computer Graphics Forum(CGF), volume 33, pages 61–70, 2014. 5, 96

[7] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In IEEE International Conference on Computational Photography(ICCP), 2012. x, xi, xii, 3, 19, 20, 55, 56, 57, 58, 61, 69, 87, 88, 90

[8] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled camera paths for video stabilization. ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH), 32(4), 2013. xii, 3, 9, 12, 13, 69, 82, 85, 86, 87, 89, 93, 97

[9] J.Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker : description of the algorithm," OpenCV Document, Intel, Microprocessor Research Labs, 2000.

[10] J.A. Sethian, Level Set techniques: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences., Cambridge Univ. Press, 1996.

[11] E.P. Krotkov, "Focusing," Int'l Journal of Computer Vision, 1(3):223–237, 1987.

[12] N.F. Zhang, M.T. Postek, R.D. Larrabee, A.E. Vladar, W.J.Keery, and S.N. Jones, "Image sharpness measurement in the scanning electron microscope," The Journal of Scanning Microscopies, vol. 21, pp. 246–252, 1999.

[13] C. Buehler, M. Bosse, and L. McMillan. Non-metric image-based rendering for video stabilization. In IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2001. 13, 15, 25

[14] B. M. Smith, L. Zhang, H. Jin, and A. Agarwala. Light field video stabilization. In IEEE International Conference on Computer Vision(ICCV), 2009. 11, 13

[15] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH), 28, 2009. 1, 13, 18, 22, 25, 31, 33, 34, 40, 42, 43, 44, 46, 60, 65, 79, 86

[16] Z. Zhou, H. Jin, and Y. Ma. Plane-based content-preserving warps for video stabilization. In IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2013. 13

[17] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH), 24(3):1134–1141, 2005. 13, 40, 42, 43

[18] A. Litvin, J. Konrad, and W.C. Karl, “Probabilistic video stabilization using Kalman filtering and mosaicking,” in Proc. of IS&T/SPIE Symposium on ElectronicImaging, Image and Video Communications., 2003, pp. 663–674.

[19] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. ACM Transactions on Graphics(TOG), pages 1–10, 2012. xi, xii, 1, 2, 12, 17, 60, 65, 69, 71, 85,89

[20] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. ACM Transactions on Graphics(TOG), 30, 2011. x, xii, 1, 3, 17, 22, 24, 25, 34, 35, 38, 40, 60, 62, 65, 69, 75, 79, 85, 86, 88, 89

[21] Y. Wexler, E. Shechtman and M. Irani, "Space-Time Video Completion," Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 1, pp. 120-127, 2004.

[22] M. Bertalmio, G. Sapiro, V. scenariolles and C. Ballester, "Image Inpainting," Proc. SIGGRAPH Conf., pp. 417-424, 2000.

[23] A. Telea, "An Image Inpainting Technique Based on the Fast Marching technique," J. Graphics Tools, vol. 9, no. 1, pp. 23-34, 2004.

[24] Tanaka, M.; Yoneji, K.; Okutomi, M., "Motion Blur Parameter Identification from a Linearly Blurred Image," Consumer Electronics, 2007. ICCE 2007. Digest of Technical Papers. International Conference on , vol., no., pp.1,2, 10-14 Jan. 2007 [25] Jeong Ho Lee; Ki Tae Park; Young Shik Moon, "Image deblurring by using the estimation of PSF parameters for image devices," Consumer Electronics (ICCE), 2010 Digest of

Technical Papers International Conference on , vol., no., pp.387,388, 9-13 Jan. 2010

[25] M. E. Moghaddam and M. Jamzad, “Finding point spread function of motion blur using radon transform and modeling the motion length,” in Proceedings of the 4th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT ’04), pp. 314–317, Roma, Italy, December 2004.

[26] H.-Y. Shum and R. Szeliski, “Construction of panoramic mosaics with global and local alignment,” Int’l Journal of Computer Vision, 36(2):101–130, 2000.