Stabilising Handheld Videos with Key Feature Tracking Algorithms

Tristan Butcher and Hugo Smith

November 2023

Contents

Abstract

With the development of handheld video cameras in the last few decades, video stabilisation has become an important area of active research. One such area of research are post-processing algorithms which serve to stabilise shaky handheld footage. Although video stabilisation algorithms are generally a solved problem, we aimed to create a simple stabilisation algorithm which would work for several types of camera motion, particularly X, Y translational and angular shake. The algorithm was based on the technique of feature matching which is known to work well for low amplitude shaking. The algorithm successfully stabilised slow rotational and translational shaking, however it failed to give the desired results when the video contained underlying movement (such as panning) and large amplitude movements. The impact of three different smoothing filters and their smoothing radius on the level of image stabilisation was also investigated. It was concluded that a gaussian smoothing filter with a smoothing radius of 50 was most aesthetically pleasing for slow handheld camera shake.

1 Introduction

Nowadays, video cameras produce high quality images and are more available than ever. Consequently, they are used in a wide range of industries ranging from police body cameras to medical diagnostic devices. Furthermore, the popularisation of small handheld devices, such as GoPro's and smartphones means more videos than ever are being produced. Despite the improvements in camera availability, many videos still suffer from distortions due to camera motion during acquisition. These videos are unstable, jittery and ultimately result in a loss of information. Therefore, it is important to be able to stabilise these videos.

The two main stabilising techniques are digital and optical image stabilisation[1]. Optical image stabilisation is a mechanical technique which occurs within the camera body while digital image stabilisation uses algorithms in post processing to remove unwanted camera shake. Video stabilisation algorithms work through three key steps: motion estimation, motion smoothing and finally image recomposition. The first step involves identifying key features in consecutive frames of the video and calculating the difference between them. This is done for all frames of the video such that the trajectory of the key features is found. In the second step, this trajectory is smoothed using a filter to prevent any jolted movements. Finally, each frame is transformed using an affine transformation matrix based on the smoothed trajectory.

Video stabilisation algorithms are by and large a solved problem. With the improvements in high performance computing, algorithms which successfully stabilise $3D$ camera motion using content-preserving warps[2] have recently been developed. This report aims to explain how each step in a $2D$ video stabilisation algorithm works, and its limitations. Firstly, a simple algorithm that stabilised a video of a moving marker on a whiteboard using color thresholding and affine transformations is discussed. This simple algorithm only corrected for slow X and Y translational shaking. Next, the algorithm was improved

to stabilise videos without prior knowledge of its contents. This algorithm automatically detected key features using Shi-Tomasi corner detection and used Lucas-Kanade optical flow to track these points through each frame. The motion of the key features was smoothed using three smoothing filters.

2 Basic Algorithm

First a basic algorithm is formed that performs the two main steps of image stabilisation, key feature detection and frame transformation. A simple video of a marker moving on a whiteboard is used to show how the algorithm works. The algorithm aims to stabilise the video about the marker such that it appears stationary. It identifies the position of the marker (key feature) in every frame and calculates the pixel displacement from the initial marker position. These displacement values are then used to translate the entire frame such that the marker returns to the initial position.

2.1 Theory

The basic algorithm draws on two main concepts, colour thresholding and affine transformations.

Colour thresholding is one of the simplest ways to identify key features in video stabilisation. It involves the separation of pixels based on their colour characteristics[3]. Each pixel in an image has RGB values. They represent how much of each primary colour (red, green and blue) is contained in a pixel's colour on a scale of 0-255. Upper and lower limits are set on each RGB component. Pixels whose RGB values fall within the set limits are kept whilst those that don't are ignored. Thus features with a specific colour range in an image are identified.

Affine transformations are used to translate the frames in the basic algorithm. An affine transformation is one that conserves co linearity, meaning all points lying on a line initially remain on a line after the transformation, and ratios of distances, meaning the midpoint of a line segment remains the midpoint after a transformation[4]. Distances and angles are not necessarily conserved.

In affine transformations, new pixel coordinates (x', y') are found through a linear combination of the original coordinates (x, y),

$$x' = a_0x + a_1y + a_2 \quad \{1\}$$
$$y' = b_0x + b_1y + b_2 \quad \{2\}$$

where the a and b coefficients define how the coordinates are transformed. In Matrix form,

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad \{3\}$$

$$T = \begin{pmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{pmatrix} \quad \{4\}$$

where T is the transformation matrix. Homogeneous coordinates $(x, y, 1)$ are used to ensure the constants a_2 and b_2 are contained in T.

2.2 Method

The first step in stabilisation is to identify key features that occur in consecutive frames. Key features can be detected automatically using edge/corner detection techniques, however these are difficult to implement, thus the basic algorithm uses the simplicity of the video to its advantage. The obvious key feature in the simple video is the marker, which is present in every frame and has a colour distinct from the background. Thus, a colour threshold is applied to find the position of the marker in every frame. The RGB (red, green, blue) values of the pixels containing the marker in the initial frame are manually extracted. Upper and lower bounds of these RGB values are then defined and used as the threshold conditions. The centre of mass position of the marker is found by taking the mean of all the pixel positions that fall within the threshold. Applying the threshold conditions to a frame returns the pixel positions of the marker, the x and y coordinates of each one of these pixels are averaged over to calculate the centre of mass position.

To isolate the marker the RGB threshold limits were set at (10-100) for both the red and green components and (50-200) for the blue component. The ranges are large because the marker is the only object present in the foreground and the background is all white. The displacement of the marker's centre of mass coordinates between the initial frame and each corresponding frame is calculated $(\Delta x, \Delta y)$, uncovering the trajectory of the marker. The displacement values are used as the coefficients in the transformation matrix 4. Since there is no information about angular displacement, the affine transformation matrix is purely translational.

$$T = \begin{pmatrix} 1 & 0 & -\Delta x \\ 0 & 1 & -\Delta y \\ 0 & 0 & 1 \end{pmatrix} \quad \{5\}$$

This matrix is applied frame by frame to every pixel. The displacements are negative such that the matrix acts to counter act the marker displacement in each frame, returning the marker to its original position.

2.3 Results

The application of a colour threshold on the first frame of the video is shown in 1. The threshold completely isolates the marker allowing its centre of mass pixel position to be

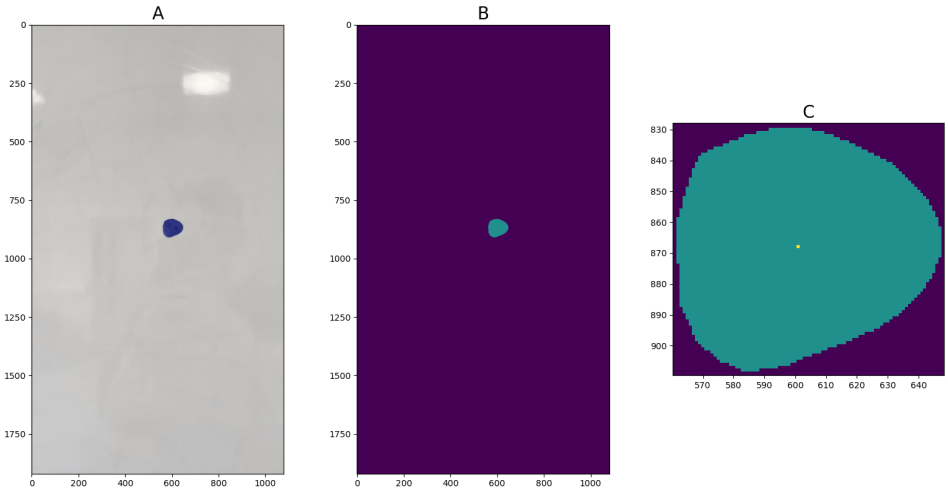

Figure 1: A) The first frame of the marker on a whiteboard video. B) A colour threshold applied to the first frame to isolate the marker. C) The centre of mass pixel of the marker is identified in yellow.

found. This pixel is shown in yellow in C.

In each frame the key feature (the marker) was successfully identified and isolated. These marker points are shown in turquoise in 2. The centre of mass coordinates of the marker in each frame were successfully found and their difference with the initial marker position (shown in yellow in 2) were found.

The differences were inserted into the transformation matrix which was applied to every pixel in each frame. Consequently, each frame was transformed such that the marker remains stationary, this can be seen in the video 'Stabilised Marker Video.mp4' in the zip file. Additionally 6 stabilised frames of interest are shown in 3.

The trajectory of the frames, shown in 4, is jagged and unsmooth.

2.4 Discussion

A simple video of a marker moving on a whiteboard was successfully stabilised about the initial position of the marker using the basic algorithm. The marker appears stationary, as expected.

The basic algorithm successfully portrays the main principles of video stabilisation. The key features were successfully identified using colour thresholds and frames were translated using an affine transformation matrix.

The stabilised video is not perfect. Black border artifacts limit the aesthetics of the video. These can be minimised by rescaling the video about its centre so that less of the border is seen. This type of function is included in the complex algorithm.

Furthermore, though the video stabilises around the marker well the transitions between

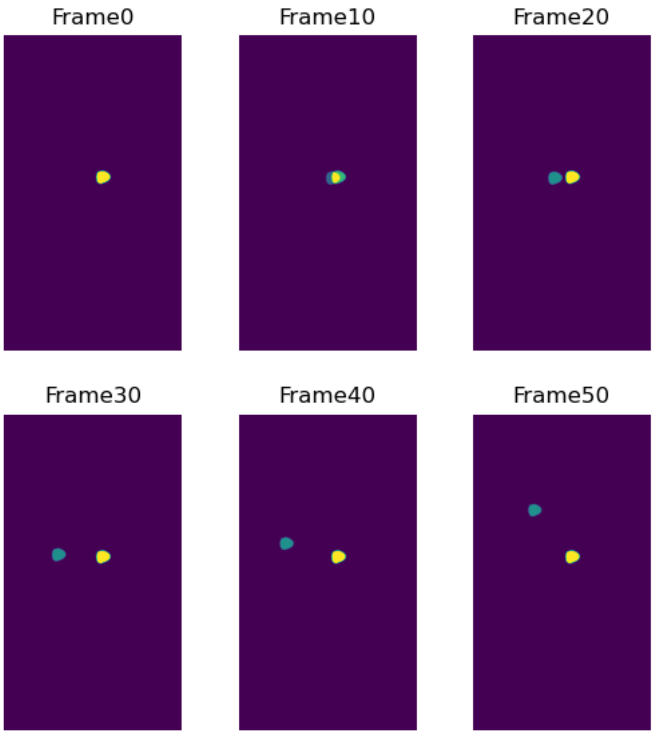

Figure 2: 6 frames from the marker on a whiteboard video. The position of the marker is identified in each frame in turquoise, whilst the initial marker position is plotted in yellow in each frame.

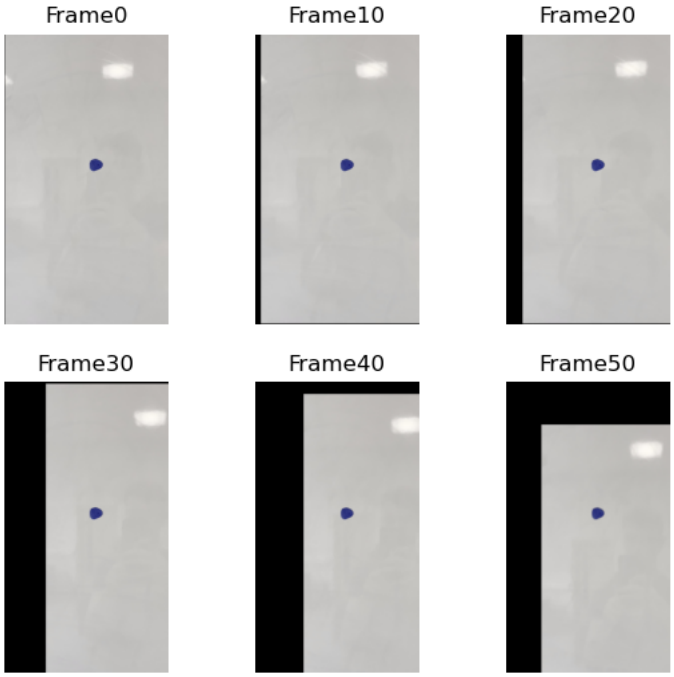

Figure 3: The same 6 frames from 2. Each frame has been translated such that the marker moves to the initial position in Frame0 and thus appears stationary.

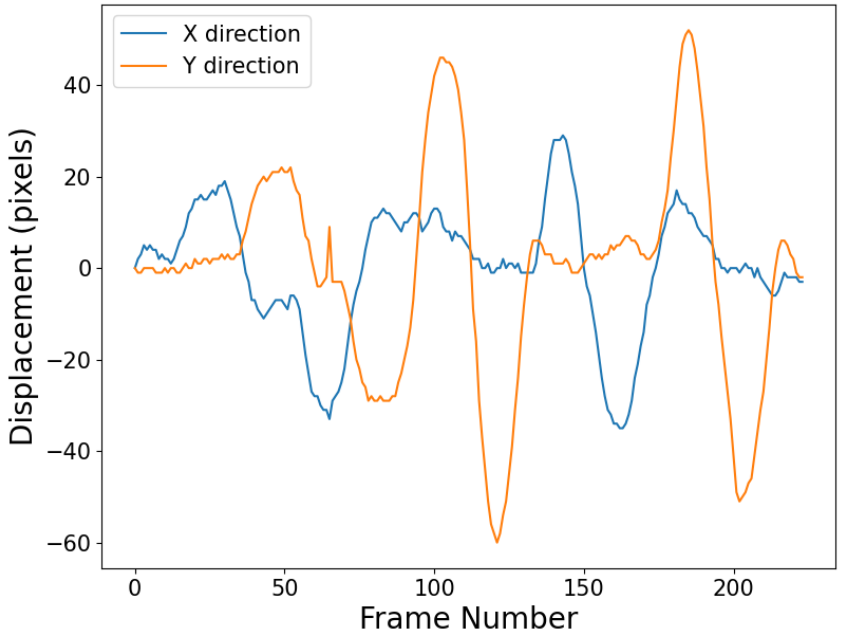

Figure 4: The trajectory of the frames in the x and y directions.

frames are unsmooth and not very stable. This is because the frame trajectory 4 is jagged. To resolve this, a smoothing algorithm can be applied to the trajectory of the frames such that the frames move continuously and don't jump around. Smoothing can be achieved using a filter that averages over the displacements of neighbouring frames. This is discussed in 3.1.3.

3 Complex Algorithm

The complex algorithm builds on the concepts of the basic algorithm but aims to stabilise a video without pre-requisite knowledge of its contents, as well as stabilising angular shake. This requires a method of automatically detecting features of interest within each frame and accurately tracking their motion[5]. This was achieved using Shi-Tomasi corner detection and Lucas-Kanade optical flow to track the corners' positions through the video. The position and rotation data from the tracked points was then smoothed using three different filters: a moving average filter, gaussian filter and a Locally Weighted Scatterplott Smoothing (LOWESS) filter. The smoothed data was then used to translate the frames accordingly so that the video appears stable. A simplified visualisation of what the algorithm does is shown in 5. Once the video had been stabilised, the black borders were corrected for by scaling the whole image slightly.

Three videos were investigated. The main video shows an unstable recording captured in a lecture theatre. The aim of the algorithm was to remove unwanted translational shake. The second video shows a human smiling, where the camera motion is rotational and the aim of the algorithm is to remove the rotation motion such that the smiling human apppears stationary. Furthermore, a video of a human running whilst the camera is also moving is investigated to understand how high frequency camera motion is affected by

the algorithm.

3.1 Theory

3.1.1 Shi-Tomasi Corner Detection

In order to identify key features within a frame, Shi-Tomasi corner detection was used[6]. This corner detection method identified key feature points within each frame by scanning over small areas of the image and finding areas where there is a large change in pixel intensity when moving in X or Y.

Let the centre of a window have intensity $I(x, y)$. If the window shifts by distance (u, v) then the intensity of the central pixel is now $I(x + u, y + v)$. The difference in intensity as the window shifts is $I(x + u, y + v) - I(x, y)$. When this difference in intensity is very large, a corner is identified. The weighted sum of square differences is,

$$S(u, v) = \sum_{pixels} w(x, y)(I(x + u, y + v) - I(x, y))^2 \quad \{6\}$$

where $w(x, y)$ is a weighting factor.

Assuming the window shifts by a small distance, $S(u, v)$ can be further simplified using a Taylor expansion of $I(x + u, y + v)$,

$$I(x + u, y + v) \approx I(x, y) + I_x u + I_y v \quad \{7\}$$

where $I_x = \frac{\partial I}{\partial x}$ and $I_y = \frac{\partial I}{\partial y}$. Equation 7 can be substituted into equation 6 to give,

$$S(u, v) = \sum_{pixels} w(x, y)(I_x^2 u^2 + I_y^2 v^2 + 2 I_x I_y uv) \quad \{8\}$$

which is written in matrix form as,

$$S(u, v) = \begin{pmatrix} u & v \end{pmatrix} M \begin{pmatrix} u \\ v \end{pmatrix} \quad \{9\}$$

where

$$M = \begin{pmatrix} \sum_{pixels} I_x^2 & \sum_{pixels} I_x I_y \\ \sum_{pixels} I_x I_y & \sum_{pixels} I_y^2 \end{pmatrix} \quad \{10\}$$

If the eigenvalues of M (λ_1 and λ_2) are both large and positive, then a corner is found. λ_1 and λ_2 also define the threshold for which points are defined as corners,

$$R = min(\lambda_1, \lambda_2) \quad \{11\}$$

If R is greater than a threshold, it's identified as a corner. This can be seen in Figure 6 and allows the quality of the key feature tracking points to be controlled (i.e. make sure a point is actually a corner).

Figure 5: Frame A (top left), Frame B (top right), combination of frames A & B (bottom left), frame B translated to position of frame A (bottom right). This figure shows a simplified visualisation of how the algorithm works between two consecutive frames. The green crosses represent the key feature points which have been identified in both frames using Shi-Tomasi corner detection.

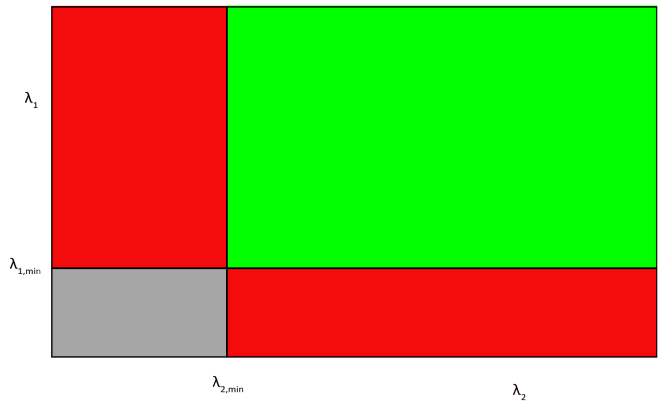

Figure 6: The classification of points based on eigenvalues of M. The grey region represents flat regions, red regions represent edges and the green regions identify corners.

3.1.2 Lucas-Kanade Method

The Lucas-Kanade technique is a method of estimating the velocity of pixels between consecutive frames using the apparent movement of objects (optical flow) between frames taken at time t and $t + \Delta t$.

Consider a pixel at position (x, y, t) with intensity $I(x, y, t)$ in a given frame. In the next frame the same pixel has intensity $I(x + \Delta x, y + \Delta y, t + \Delta t)$. In order to track this point between the two frames, it is assumed the pixel has constant brightness,

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \quad \{12\}$$

If the movement of the pixel between frames is assumed to be small, a Taylor expansion can be performed,

$$I(x + \Delta x, y + \Delta y, t + \Delta t) \approx I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial x}\Delta y + \frac{\partial I}{\partial t}\Delta t \quad \{13\}$$

Substituting equation 13 into equation 12, it follows that,

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial x}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0 \quad \{14\}$$

and dividing both sides by Δt,

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial x}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t} = 0 \quad \{15\}$$

$$I_x V_x + I_y V_y = -I_t \quad \{16\}$$

where V_x and V_y are the velocities (i.e. optical flow) of the pixel $I(x, y, t)$ in the x and y directions and I_x, I_y and I_t are the derivatives of the image evaluated at (x, y, t). Equation 16 is known as the optical flow equation.

In order to solve the optical flow equation, the Lucas-Kanade method assumes that all pixels within a window centred at p have constant motion in the same direction. This results in n optical flow equations where the local velocity vector (V_x, V_y) satisfies,

$$\begin{aligned} I_x(q_1)V_x + I_y(q_1)V_y &= -I_t(q_1) \\ I_x(q_2)V_x + I_y(q_2)V_y &= -I_t(q_2) \\ &... \\ I_x(q_n)V_x + I_y(q_n)V_y &= -I_t(q_n) \end{aligned} \quad \{17\}$$

where q_1,q_2,...q_n are the pixels within the window and I_x, I_y and I_t are defined as before but evaluated at the position q_i.

This equation can be written in matrix form as $Av = b$ where,

$$A = \begin{pmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ ... & ... \\ I_x(q_n) & I_y(q_n) \end{pmatrix} v = \begin{pmatrix} V_x \\ V_y \end{pmatrix} b = \begin{pmatrix} -I_t(q_1) \\ -I_t(q_2) \\ ... \\ -I_t(q_n) \end{pmatrix} \quad \{18\}$$

This system of equations has no solution, so the Lucas-Kanade method uses the principle of least squares to find a solution. This means it solves,

$$A^T Av = A^T b \quad \{19\}$$

$$v = (A^T A)^{-1} A^T b \quad \{20\}$$

or similarly,

$$\begin{pmatrix} V_x \\ V_y \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} I_x(q_i)^2 & \sum_{i=1}^{n} I_y(q_i) I_x(q_i) \\ \sum_{i=1}^{n} I_y(q_i) I_x(q_i) & \sum_{i=1}^{n} I_y(q_i)^2 \end{pmatrix}^{-1} \begin{pmatrix} -\sum_{i=1}^{n} I_x(q_i) I_t(q_i) \\ -\sum_{i=1}^{n} I_y(q_i) I_t(q_i) \end{pmatrix} \quad \{21\}$$

Therefore, the Lucas-Kanade method calculates the velocity (V_x and V_y) of the small window of pixels between consecutive frames (i.e. the optical flow). This allows the position of points to be predicted between frames. This method assumes that the motion of pixels is small between frames, so isn't suitable for motion tracking in videos where there are large movements.

3.1.3 Motion Smoothing

In the basic algorithm discussion it was suggested that the jagged nature of the trajectory of the frames resulted in a less stable video. Smoothing this trajectory accounts for this. There are many smoothing techniques, this report discusses three of them.

The simplest smoothing technique is a moving average filter, which takes the average of neighbouring data points for each point. Suppose we have an unsmooth curve C which contains n data points. To get a smooth curve S a moving average filter with a width (smoothing radius) of 5 is applied. The k^{th} element of the smooth curve is calculated via the following equation,

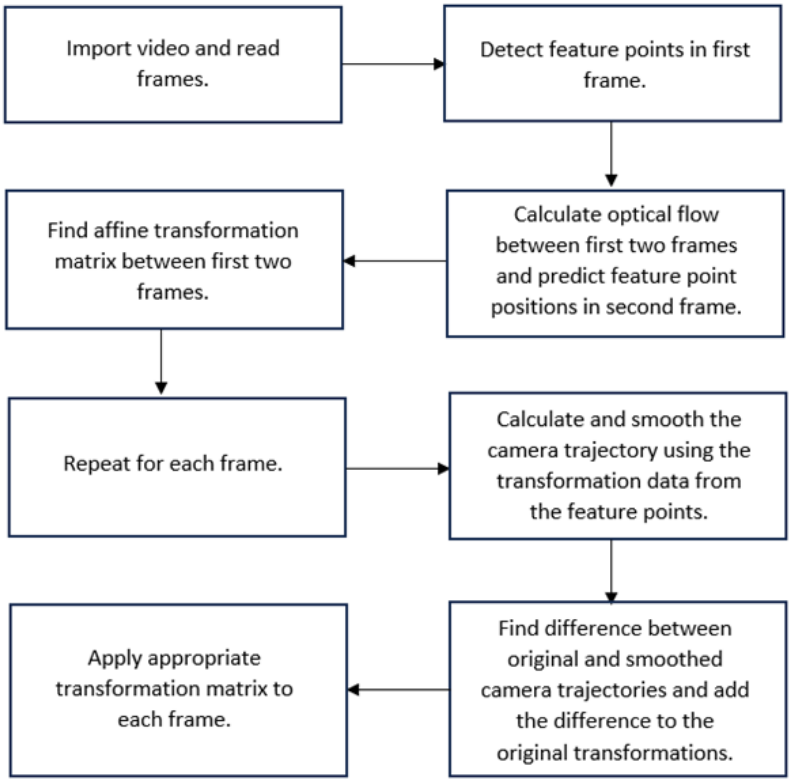

Figure 7: Flowchart of algorithm.

$$S[k] = \frac{C[k-2] + C[k-1] + C[k] + C[k+1] + C[k+2]]}{5} \quad \{22\}$$

An extension to this is the gaussian filter, which again considers neighbouring data points, however instead of averaging them it applies a gaussian weighting such that the neighbours closer to the point of interest hold more significance than those further away.

A more advanced technique is Locally Weighted Scatterplot Smoothing (LOWESS). Again neighbouring data points that are weighted dependent on their distance away from the point of interest are considered. A weighted least-squares fit line or parabola is calculated and used to define a smoother data point. Computing this for every point creates a smoother curve, however outliers are considered as equal to any other point. To reduce the influence of outliers an additional weight is included that depends on how far the original point is from the new point. Old and new points that are close together get a higher weighting than those that are far apart. The whole process is then repeated, however now there are two separate weighting parameters. The result is a smooth curve whereby the influence of outliers has been diminished.

3.2 Method

A flowchart of the algorithm is shown in 7. The first step of the algorithm was to automatically identify key features within a given frame using Shi-Tomasi corner detection.

The number of key feature points to track was chosen at 200 to ensure there were enough high-quality points being tracked. This also meant that if a few points lost their tracking through the animation, they would not have a large impact on the final affine transformation matrix (which considered all 200 points).

Once the key feature points (corners) were identified for a given frame, the position of the same feature points were estimated in the next frame using the optical flow between consecutive frames (the Lucas-Kanade method). The affine transformation matrix, containing both translational (δx and δy) and rotational data ($\delta\theta$), was calculated for the two sets of key feature points between frames. This allowed frames to later be translated in a way that stabilised them (i.e. the key feature points remained at similar positions between frames). This was repeated for each frame of the video and resulted in a set of affine transformations going between every consecutive frame in the video.

The next step was to use the translational and rotational data to reconstruct the X, Y and angular trajectory of the camera within the scene by making an array of the cumulative sum of the translations and rotations. Once a trajectory was calculated, the algorithm smoothed it using one of three filters: moving average filter, gaussian filter and a LOWESS filter. These filters all removed any high frequency, jittery motion in the camera trajectory but in slightly different ways. The intensity of smoothing was controlled by the smoothing radius of the filters.

The effect of the three filters on the frame trajectory was investigated to select the ideal filter. Then the effect of the smoothing radius of the chosen filter was investigated to find the value that best stabilises the trajectory whilst keeping the underlying camera motion. Smoothing radius values of 5 to 500 were considered.

The difference between the smoothed and original trajectory was calculated at each frame, and that difference was added onto the original translation and rotation data. Once the X, Y and angular data had been smoothed, the data was compiled into an affine transformation matrix of the form,

$$M = \begin{pmatrix} \cos\delta\theta & -\sin\delta\theta & \delta x \\ \sin\delta\theta & \cos\delta\theta & \delta y \end{pmatrix} \qquad \{23\}$$

where δx, δy and $\delta\theta$ are the translation and rotation changes between consecutive frames. The algorithm then applied the appropriate M matrix to each frame of the video which resulted in a stabilised final video.

Due to the nature of the stabilising process, the final video contained black border artefacts. These were corrected for by the algorithm with a simple scaling technique. For relatively stable original videos, a scaling of 8% was usually sufficient to remove the border artefacts, however trial and error was often required to ensure no black borders were present.

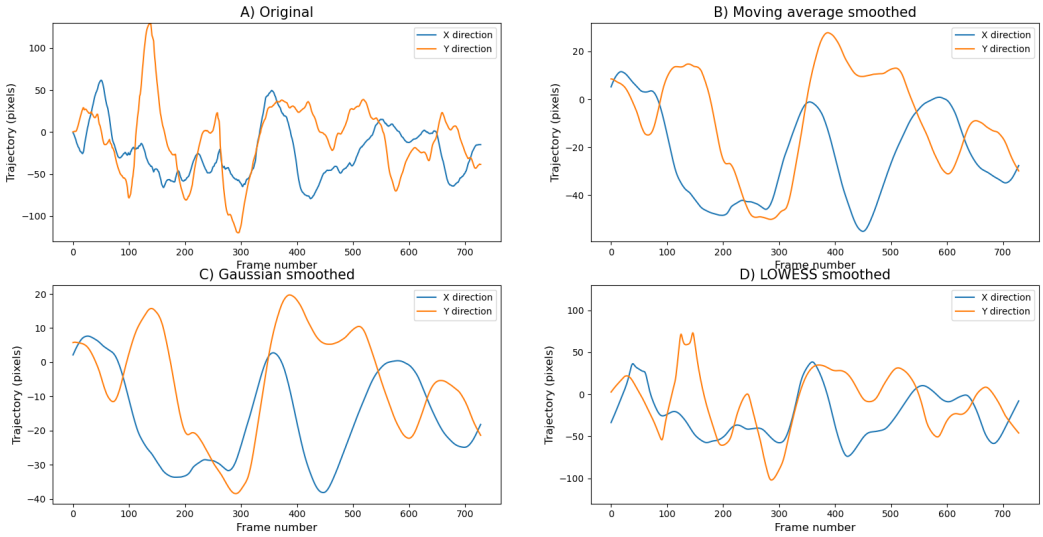

Figure 8: A) The original trajectory in the x and y directions of the lecture video before smoothing is applied. Smoothing filters with a smoothing radius of 50 were applied. Note the difference in the size of the Trajectory axes. B) The trajectory smoothed with a moving average filter. C) The trajectory smoothed with a gaussian filter. D) The trajectory smoothed with LOWESS filter.

3.3 Results

The initial trajectory of the lecture video is jagged and unsmooth, this is shown in 8A. The three smoothing filters, each with a smoothing radius of 50, were applied resulting in three slightly different trajectories. All three filters significantly reduce the range of displacement values and all remove the majority of the jagged points making the trajectory smoother. However, the moving average and gausian filters reduce the trajectory range much more dramatically than the LOWESS filter does. Consequently, when the displacement values are recalculated for each filter to include smoothing, as seen in Figure 9, the gaussian and moving average filters produce larger transformations and the curve is much smoother. Smoother transformations are idealistic for a stable video, furthermore the larger on average transformation values mean that more of the camera motion was removed during stabilisation. This is the ideal result, because for the lecture video all but very subtle motion was assumed to be the undesired, thus there is little underlying motion that was required to be preserved. Therefore, the gaussian filter was selected as the best filter to use for the lecture video.

Next the effect of the smoothing radius on gaussian stabilisation was investigated. The smoothed trajectories for different radius values is shown in Figure 10. Low values of around 5 produce very subtley smoothed curves, however high values around 500 produce flat lines very close to 0, thus both produce very minimal stabilisation affects. This implies that the ideal smoothing radius lies between these values, more likely in the lower region as to preserve more of the original trajectory. As the smoothing radius is increased the trajectory curves lose more an d more of their jagged edges and begin to flatten out. The ideal curve has a small amount of flattening to produce a larger differences with the initial trajectory, however the general shape of the curve should be maintained. It was decided that a smoothing radius of 50 best fits this criteria. The final displacement graph for a gaussian filter of smoothing radius 50 applied to the unstable lecture video

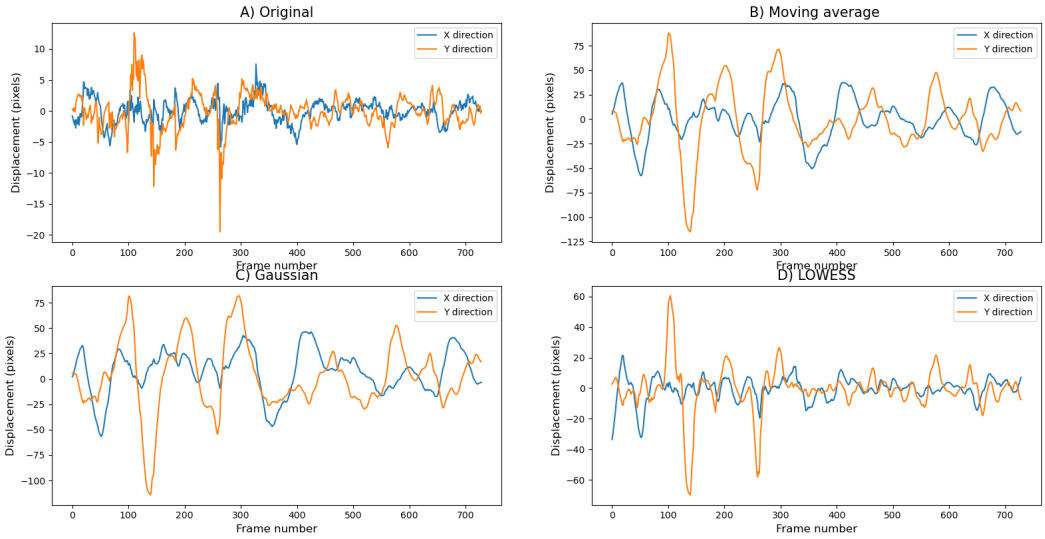

Figure 9: A) The original x and y transform values for each frame calculated from the optical flow algorithm. The difference between the original trajectory and smooth trajectories were added to the orginal transforms to form smoothed transforms. B) The transform smoothed with a moving average filter. C) The transform smoothed with a gaussian filter. D) The transform smoothed with LOWESS filter.

is shown in Figure 11.

The final stabilised video can be found in the zip folder as 'Stabilised Lecture Video.mov' and stills are shown in 12. The left sided videos show the original video, which has lots of camera motion. The right sided videos show the stabilised version with a gaussian filter of smoothing radius 50 applied. The lecturer and desks stay in roughly the same position as expected. There is a slight black border artefact in Frame 6.

The algorithm can successfully stabilise rotational camera motion such that the object of interest appears to not rotate, this is shown in 13 and the video 'Stabilised Rotation Video.mp4' in the zip folder.

The algorithm fails to stabilise videos with high frequency camera motion. This is shown in 14 and the video 'Stabilised Running Video.mp4' in the zip file. The stabilise frames move chaotically around the screen and are much less stable than the original video.

3.4 Discussion

The lecture video was successfully stabilised using the complex algorithm with a range of filters and smoothing radius values, however the optimal stabilisation was achieved with a gaussian filter of smoothing radius 50. This stabilisation removed the jittery motion and the unwanted camera panning resulting in a video fixed on the lecture taking place.

The complex algorithm stabilises videos more generally than its basic counterpart as it smooths trajectories and removes unwanted camera motion whereas the basic algorithm seeks to stabilise the video around a specific feature. The complex algorithm detects key features automatically using Shi-Tomasi corner detection and thus can be applied to

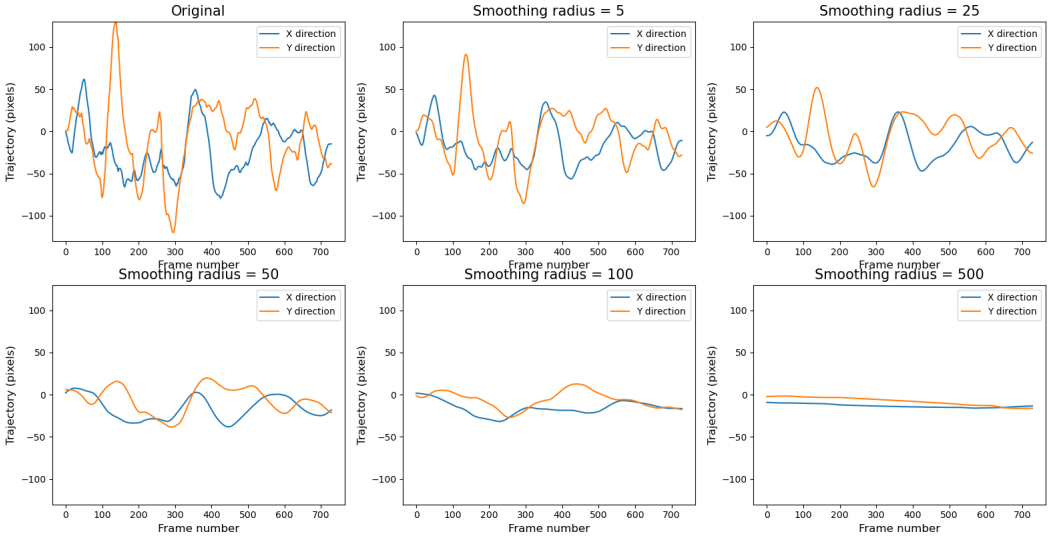

Figure 10: Plots of the original x and y trajectories smoothed by a gaussian filter with different values for the smoothing radius.

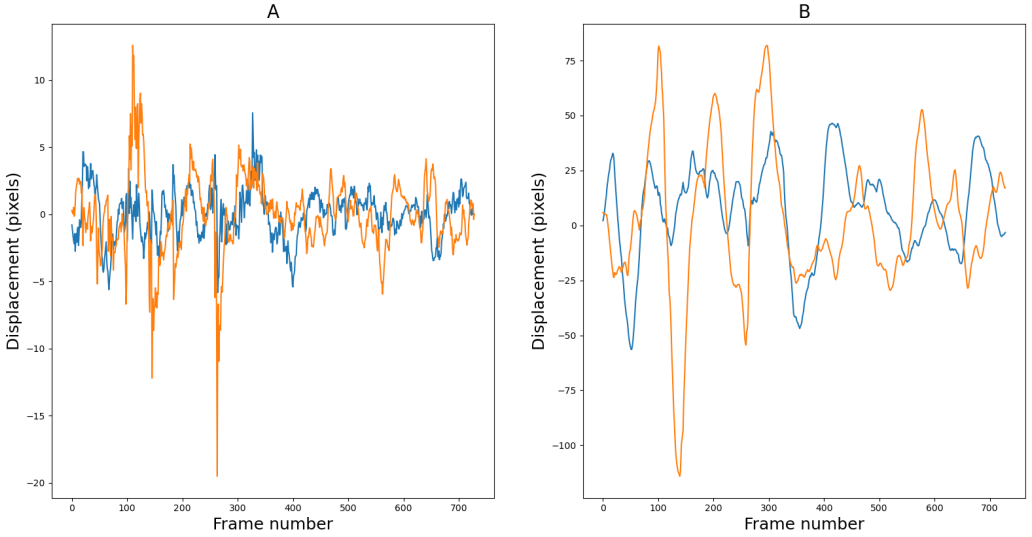

Figure 11: The x and y transforms for the optimal gaussian filter of smoothing radius 50 for the lecture video compared with the initial transform.

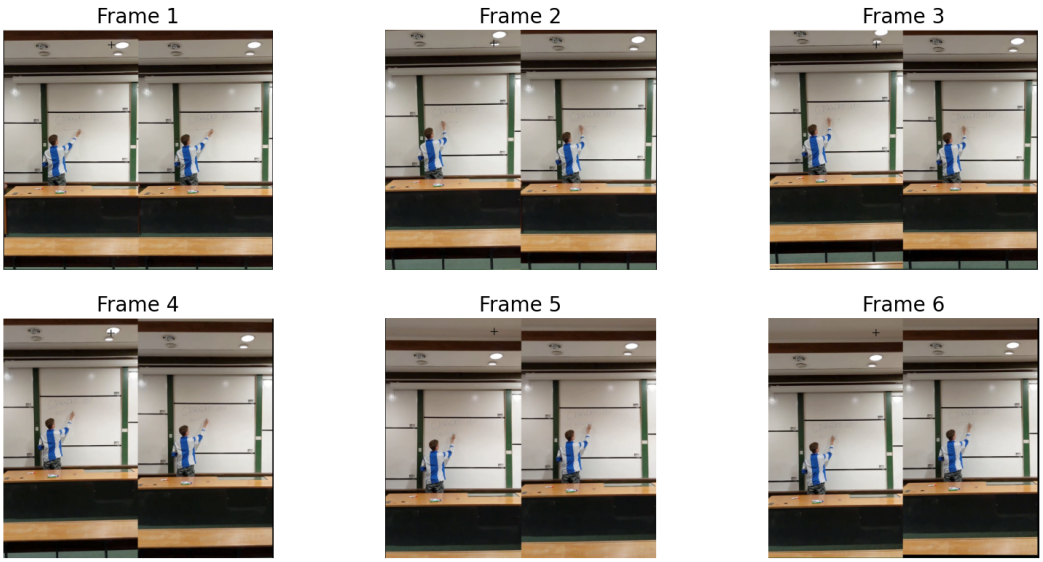

Figure 12: Six frames from the original lecture video (left) and the stabilised lecture video (right).

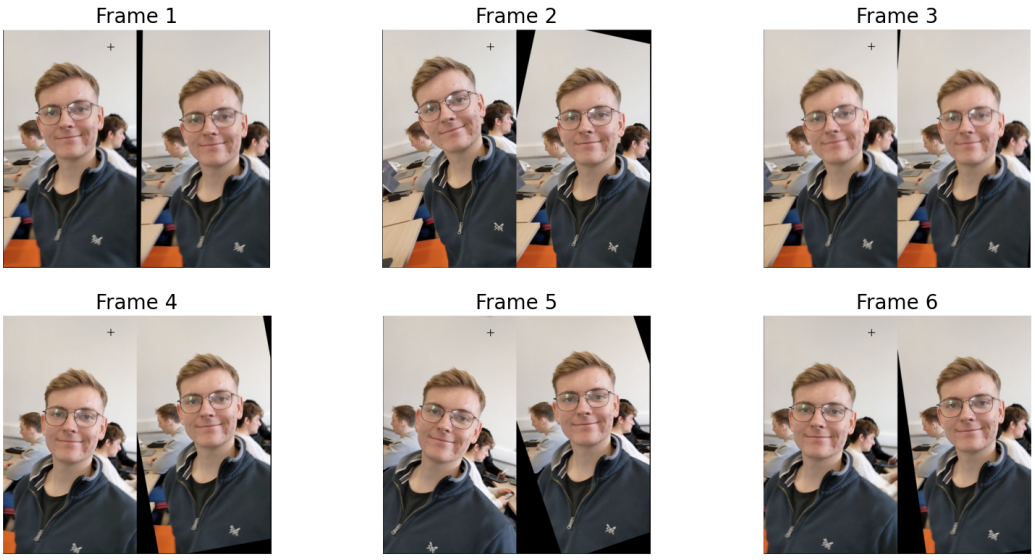

Figure 13: Six frames from the original rotation video (left) and the stabilised rotation video (right).

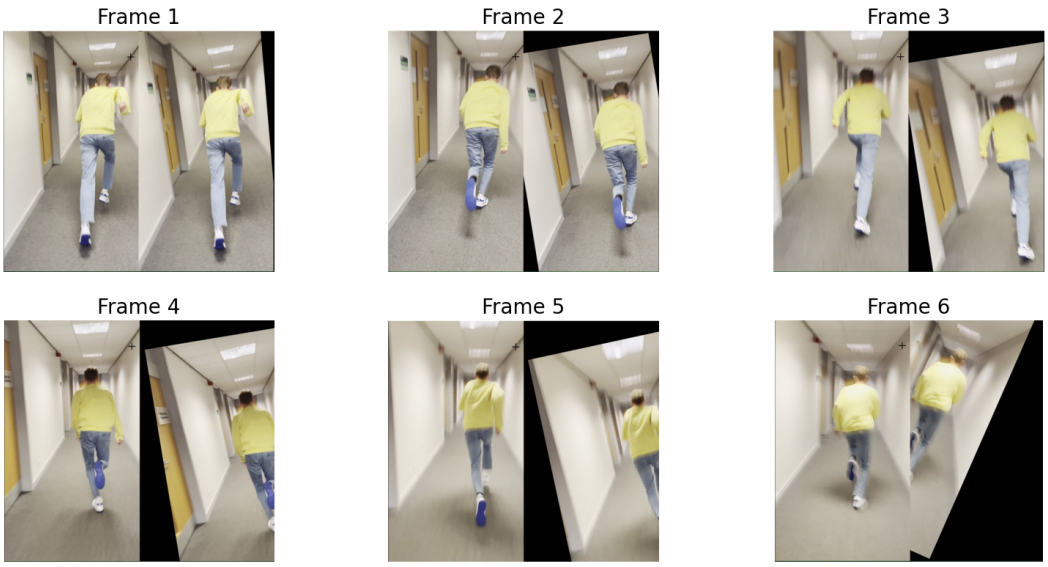

Figure 14: Six frames from the high frequency camera motion video of someone running. The original frames are on the left and the stabilised frames are on the right.

more complex videos than the basic algorithm, which needs manual input to define the colour thresholds used to identify the object of interest and relies on the fact that the object is the only thing in the frame that falls into the threshold. The Lucas-Kanade method successfully tracked the optical flow of the key features.

Additionally, the complex algorithm accounts for rotational instability by calculating angular transformations. This allows frames to be rotated in order to stabilise more effectively. This feature of the algorithm is highlighted in the rotation video 13, which stabilises angular movements around the human, making him appear stationary as expected.

Furthermore, since the complex algorithm does not fixate solely on one initial point and instead considers multiple points on a frame by frame basis, black border artefacts are significantly reduced. These were also minimised with the border fixing function which scales the frames by 8% without moving the centre of the frames. However, at some points in the stabilised video these artefacts were still present. Therefore to improve the algorithm a cutoff function could be included that dampens any transformation that exceeds 8% of the frame dimensions.

Moreover, the complex algorithm strongly minimises low frequency motion, however it struggles with higher frequencies. This is highlighted in the running video 14 where the frames move chaotically making the final video less stable than the original. This is because key features in these videos become blurred and distorted making them difficult to detect and track, thus the algorithm breaks down. One method to remove high frequency motion is to enter the fourier domain by applying a discrete fourier transform to the camera motion and applying a low pass filter[7].

4 Conclusion

This report has explained two algorithms that stabilise different types of videos. The basic algorithm stabilises videos around a specific object by identifying key features using a colour threshold and applying an affine transformation to translate each frame to the original object position. This is successfully applied to a video of a moving marker. However, black border artefacts and discontinuous motion limit the aesthetics.

A shaky video of a lecture was successfully stabilised using the complex algorithm. Key features in frames were detected using Shi-Tomasi corner detection and tracked using the Lucas-Kanade method. This detection and tracking method is automated and more general. Smoothing filters applied to the motion trajectory prevent jittery discontinuous motion.

For the lecture video the gaussian filter with a smoothing radius of 50 was determined to have the best stabilisation effect.

References

[1] Wilko Guilluy, Laurent Oudre, and Azeddine Beghdadi. Video stabilization: Overview, challenges and perspectives. *Signal Processing: Image Communication*, 90:116015, 2021.

[2] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3d video stabilization. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 631–639. 2023.

[3] CD Fermin and S Degraw. Colour thresholding in video imaging. *Journal of anatomy*, 186(Pt 3):469, 1995.

[4] Eric W Weisstein. Affine transformation. *https://mathworld. wolfram. com/*, 2004.

[5] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[6] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, volume 2, pages 674–679, 1981.

[7] SARP Erturk and TJ Dennis. Image sequence stabilisation based on dft filtering. *IEE Proceedings-Vision Image and Signal Processing*, 147(2), 2000.