# Stabilization of Crane-Mounted Cameras

Kerry He

February 2021

## 1 Introduction

### 1.1 Background

Construction with cranes is a dangerous activity, where being struck by a crane load is the most frequently recorded accident type (23%) in the USA between 1997 and 2003 [1]. Lack of crane vision is a predominant factor to these injuries and fatalities, with maneuvers such as blind lifts demonstrating the difficulties crane operators face in this regard.

As a result, some research has been done on using cameras on cranes to provide crane operators with visual feedback. Overhead cameras have been a popular position to mount these cameras [2], [3], and provide crane operators with a birds-eye-view of the construction site. A camera mounted to the hook block, which seems to be less commonly investigated, would provide a first-person view of the crane load, and may be more intuitive for a crane operator to use.

However, crane loads are susceptible to swinging motions, which would result in a dizzying video feed. Therefore, video stabilization techniques should be employed to maximise the usability of such an approach.

### 1.2 Existing Work

The method used to stabilize crane images is predominantly adapted from existing stabilization methods. This typically consists of the following steps:

- **Global motion estimation**: Two main methods used to describe this include optical flow methods [4] or geometric methods. Within geometric methods, this is typically done by calculating the image transform between two consecutive frames using a pose estimation algorithm such as feature point matching techniques [5], then accumulating the relative transforms together to find a global transform. Image transform models used to estimate this transform are usually the 6-parameter affine model [6] or the 8-parameter homography model [7].

- **Intentional motion estimation**: The intended noise-free trajectory of the camera is estimated from its measured trajectory. Online applications of this are typically done using a filtering algorithm such as Kalman Filtering [6] or IIR high-pass filtering [8]. Other offline methods typically involve some kind of curve fitting method [5], [9], including the L1 optimal method presented by Grundmann [10].

- **Compensation for unwanted motion**: The image is warped to compensate for the difference between the global and intentional motion estimates. During this step, information loss is typical due to the image shifting outside of the frame. Techniques to compensate for this involve mosaicking [6] or adjusting the motion compensation to minimise the amount of information loss [11].

The problem with these existing techniques is predominantly in the intentional motion estimation methods. The online methods presented all use a form of low-pass filter to remove high-frequency noise. However, crane swinging is characterised as a low-frequency disturbance. Furthermore, methods that are able to remove low-frequency swaying or swinging motion are limited to the offline methods presented. Thus, the main contribution of this project is to develop a low-frequency online video stabilization algorithm.

# 2 Methodology

## 2.1 Global Motion Estimation

Due to the structured environment for which this algorithm is being developed, it is assumed that a marker-based pose estimation approach could be used. Not only does this eliminate issues with drift that markerless approaches face, it also allows for the displacement between the camera and marker to be calculated, which can be used for autonomous control of the crane.

As opposed to existing work which use affine models, as the magnitudes of motion compensation the algorithm is expected to account for are much larger than that faced by low-amplitude, high-frequency vibrations, the homography model is used for this work. Moreover, homography transforms are more commonly used in literature for Euclidian pose estimation purposes.

An off-the-shelf marker, AprilTag [12], which is commonly used for pose estimation applications, is used as the known marker for its proven detection accuracy. Once the Apriltags are detected, the four corners of each marker are used to estimate a homography transform $H_w^c \in SO(3)$ which describes the image transformation from the marker to the camera. Four point-correspondences are required to solve for a homography matrix using a non-iterative closed form method [13]. However, this makes the estimated homography matrix extremely sensitive to noise [14]. Hence, an iterative method Random Sample Consensus (RANSAC) [15], which excels at dealing with outliers, is used alongside multiple

AprilTag markers. This homography estimation method is implemented using OpenCV [16].

The homography $H_w^c$ is then be decomposed to obtain a Euclidian transformation using the method described by Simon [13]:

$$\begin{bmatrix} \boldsymbol{r}_1^c & \boldsymbol{r}_2^c & \boldsymbol{t}_c \end{bmatrix} = K^{-1} H_w^c \tag{1a}$$

$$\boldsymbol{r}_3^c = \boldsymbol{r}_1^c \times \boldsymbol{r}_2^c \tag{1b}$$

Where $K$ is the intrinsic camera matrix, $\boldsymbol{r}_n^c$ represents column vector components of a matrix and $\tilde{R}_c = \begin{bmatrix} \boldsymbol{r}_1^c & \boldsymbol{r}_2^c & \boldsymbol{r}_3^c \end{bmatrix}$ and $\boldsymbol{t}_c$ represent the rotations and translations from the marker to the camera. Equation (1a) arises from the definition of homography matrices, and Equation (1b) arises to enforce orthogonality of the rotation matrix. However, as $H_w^c$ is an estimated value, $\tilde{R}_c$ is not guaranteed to be orthogonal. Hence, $\tilde{R}_c$ is normalised to find the closest valid rotation using an SVD method:

$$USV^T = \tilde{R}_c \tag{2a}$$

$$R_c = UV^T \tag{2b}$$

## 2.2 Intended Motion Estimation

A model-based technique inspired by [7] was implemented, where we remove undesired changes in certain state variables. We define the intended motion of the camera as the translation of the boom head, as this is the component of the crane the crane operator is directly in control of, and unwanted motion as swinging of the rope.

We assume that the crane can be modelled using a single-pendulum model using a rotation and translation $x = \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix}^T$, as this allows us to fully define the crane state from the camera state alone.

Using this model, we can derive the relationship between the measured camera state and desired camera state with the transformation $T = \left[\begin{array}{c|c} R_T & \boldsymbol{t}_T \end{array}\right]$:

$$R_T = R_c^{-1} \tag{3a}$$

$$\boldsymbol{t}_T = (I_3 - R_c)\hat{\boldsymbol{g}}l \tag{3b}$$

## 2.3 Motion Compensation

Now that we have a Euclidian transformation that we want to transform the camera pose by, we need to convert this to a corresponding image transform that warps the image to mimic this camera movement. This can be done using the following relationship [17]:

$$H_E = R - \frac{\boldsymbol{t}\hat{\boldsymbol{n}}^T}{d} \tag{4}$$

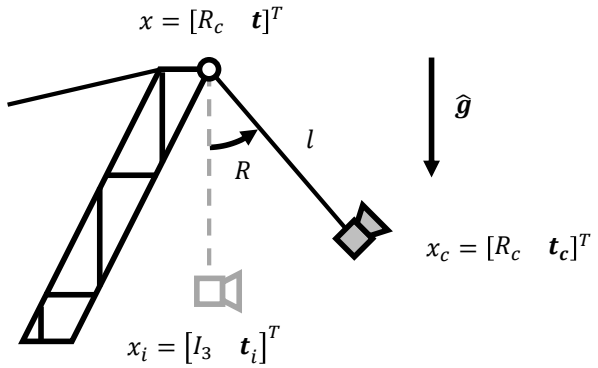


Figure 1: Single pendulum diagram of the simplified crane model. The crane state is given by, $x$, the actual camera state is given by $x_c$ and the intended camera state is given by $x_i$. Each state can be related to each other given a known rope length $l$.

Where $H_E$ is the Euclidian homography corresponding to the transformation given by $R$ and $\boldsymbol{t}$, $\hat{\boldsymbol{n}}$ is the unit normal vector of the plane the homography is transforming, and $d$ is the depth measured in the direction of $\hat{\boldsymbol{n}}$ between the camera and the plane. Note that we can derive the depth as $d = \boldsymbol{t}_c \cdot \hat{\boldsymbol{n}}$, and $\hat{\boldsymbol{n}} = \hat{\boldsymbol{z}}$. The projective homography matrix $H$ which is needed to transform the image is found from:

$$H = KH_EK^{-1} \tag{5}$$

In order to properly transform the image, we note that to flatten the marker, a homography transform of $H = KRK^{-1}$ is needed. Any additional translational component $\boldsymbol{t} \neq \boldsymbol{0}$ to the homography will affect how much the image is skewed, resulting in a plane that isn't completely flat nor proportioned correctly.

Hence, we are required to isolate the rotational and translational components into two separate homographies to achieve the desired affect. Important to note is that after the inital rotational homography, the implied depth of the plane has changed, which needs to be accounted for in the second translational homography.

$$H_E^R = R_T \tag{6a}$$

$$H_E^t = I_3 - \frac{\boldsymbol{t}_T \hat{\boldsymbol{n}}^T}{(R\hat{\boldsymbol{n}}^T d) \cdot \hat{\boldsymbol{n}}} \tag{6b}$$

$$H_c^i = KH_E^t H_E^R K^{-1} \tag{6c}$$

## 2.4 High Frequency Filtering

Although the main goal is to filter out low frequency swinging, high frequency vibrations are still present, and pose an unpleasant disturbance to the video feed. Hence, we should still aim to remove this type of noise. A method using Kalman Filtering similar to that described by Litvin [6] is implemented.

When using a Kalman Filter to stabilize images, the state variable is usually defined with respect to a global frame, so that we are smoothing the trajectory of the camera. For our problem, we want to stabilize the resultant image after applying the homography $H_c^i$. Hence. the variable we want to filter is the global homography corresponding to intended camera pose relative to the world frame $H_w^i = H_c^i H_w^c$.

We use a first order Kalman Filter formulation as described by Labbe [18]. To formulate the Kalman Filter, we define the state vector as:

$$x = \begin{bmatrix} \text{vec}(H_w^i) & \text{vec}(\dot{H}_w^i) \end{bmatrix}^T \tag{7}$$

The update matrix is defined as:

$$A = \begin{bmatrix} I_{9\times 9} & \Delta t_{9\times 9} \\ 0_{9\times 9} & I_{9\times 9} \end{bmatrix} \tag{8}$$

The measurement matrix is defined as:

$$H = \begin{bmatrix} I_{9\times 9} & 0_{9\times 9} \end{bmatrix} \tag{9}$$

The Kalman Filter algorithm is as follows:

$$x_{n|n-1} = Ax_{n-1} \tag{10a}$$
$$P_{n|n-1} = AP_{n-1}A^T + Q \tag{10b}$$
$$K_{n|n-1} = P_{n|n-1}H^T(HP_{n|n-1}H^T + W)^{-1} \tag{10c}$$
$$x_n = x_{n|n-1} + K(z_n - Hx_{n|n-1}) \tag{10d}$$
$$P_n = P_{n|n-1} - KHP_{n|n-1} \tag{10e}$$

Where $Q$ is the process error covariance, $W$ is the sensor measurement covariance, and $z_n$ is the measurement vector defined as $z_n = \text{vec}(H_{w,n}^i)$. The filtered global homography $\tilde{H_w^i}$ can then be extracted from the state vector, from which our filtered image transform homography can be calculated:

$$\tilde{H}_c^i = \tilde{H}_w^i (H_w^c)^{-1} \tag{11}$$

## 2.5 Mosaicking

Due to the high-amplitude swinging that the algorithm has to remove, we expect there to be large amounts of information loss after warping the image. Hence, mosaicking is required to retain as much information of the scene as possible. An

iterative approach is used in which the previous mosaic, which already contains information about all previous images, is used to stitch together with the new image, as opposed to stitching all images together individually. This minimises the amount of memory and complexity required from the algorithm. However, this means parts of the mosaic which exit the frame will lose information, so the size of the canvas represents the amount of spatial memory the mosaicking algorithm has.

To achieve the mosaicking effect, the previous mosaic needs to be transformed into the same camera frame as the new image after it has been warped by $\tilde{H}_c^i$. This can be done by chaining together the following set of transforms:

$$H_{i,n-1}^{i,n} = \tilde{H}_{c,n}^{i} H_{w,n}^{c} (\tilde{H}_{c,n-1}^{i} H_{w,n-1}^{c})^{-1} \tag{12}$$

However, if the mosaic has a different size to the raw image, a translational transform needs to be included as well, firstly to account for the different image principal points, and secondly to shift the new image onto the larger mosaic canvas. Let us define this transform as:

$$H_i^f = \begin{bmatrix} 1 & 0 & c_{x,mosaic} - c_{x,img} \\ 0 & 1 & c_{y,mosaic} - c_{x,img} \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

Hence, this gives us the final forms of the homography matrices we use to warp both the new image and previous mosaic:

$$H_{mosaic} = H_i^f H_{i,n-1}^{i,n} (H_i^f)^{-1} \tag{14a}$$

$$H_{img} = H_i^f \tilde{H}_{c,n}^{i} \tag{14b}$$

After the mosaic has been constructed, a smaller section approximately the same size as the original image is cropped out to form the output of the algorithm.

## 2.6 Alternative Intended Motion Definitions

### 2.6.1 Known Boom Pose

The crane dynamics are more closely estimated using a double pendulum or more complex model, meaning the single pendulum model assumption made prior may not estimate the swinging motion accurately. Using additional sensors, we may instead be able to have full knowledge of the boom head state, which defines the intended motion of the camera.

In this scenario, we can calibrate the camera and boom head at some initial pose by measuring the boom head position $\boldsymbol{t_0}$ and homography matrix $H_{w,0}^{c}$.

The following transforms then consist of a transform to return the camera to this origin position, then displacing it by the amount the boom head is measured to have moved. The full motion compensation expression can be found as:

$$H_c^i = H_{t_{boom}} H_{w,0}^c (H_{w,n}^c)^{-1} \tag{15}$$

Where $H_{t_{boom}}$ is equal to the projective homography corresponding to a pure translation $\boldsymbol{t}_T = \boldsymbol{t}_n - \boldsymbol{t}_0$ as calculated in Equation (6).

As part of future work, other ways to model the motion of the crane and load swinging could be investigated. For example, in a similar vein to how a single pendulum has a fixed frequency of oscillation, it may be possible to calculate the frequency of oscillation of the crane, and use a band-pass filter to remove these frequencies, in which case a more traditional stabilization method can be used.

#### 2.6.2 Minimum Depth

This formulation of treating intended motion estimation as any Euclidian transform is extremely general, and lends itself to tasks beyond stabilization. For example, one additional feature we may want to add to the algorithm is a function that limits the depth at which the camera is from the wall plane, as a video taken too close to the wall would lack useful information.

We can implement this simply by redefining the intended motion of the camera as $d = \boldsymbol{t}_c \cdot \hat{\boldsymbol{n}} > d_{min}$. This can be implemented as:

$$\boldsymbol{t}_T = \begin{bmatrix} 0 & 0 & \min(0, d - d_{min}) \end{bmatrix}^T \tag{16}$$

# 3 Results and Discussion

## 3.1 Experimental Setup

Experiments were conducted using the ABB IRB 14000 YuMi robot to simulate the motion of a crane load. A Basler ace acA2040-55uc USB 3.0 camera was used with a 6mm lens. The camera was calibrated using images of a checkerboard to obtain instrinsic camera parameters and distortion parameters. A custom 3D-printed adapted was used to attach the camera to the YuMi gripper. A marker array consisting of eight 36h11 AprilTags was used for pose estimation.

For the simulated paths the YuMi will follow for the experiments, there are two main types of motion:

- **Single Pendulum Kinematics**: Single pendulum motion assuming a rigid rope is simulated, as described by Figure 1. Only kinematic constraints of this system are considered as the method described should work for any feasible state. This type of motion was mainly used to verify that the methodology was implemented correctly for a proof of concept.

- **Double Pendulum Dynamics**: A path is pre-generated using the simulation of a double pendulum dynamic model of a tower crane [19]. This is represented as a series of timestamped transformation matrices which

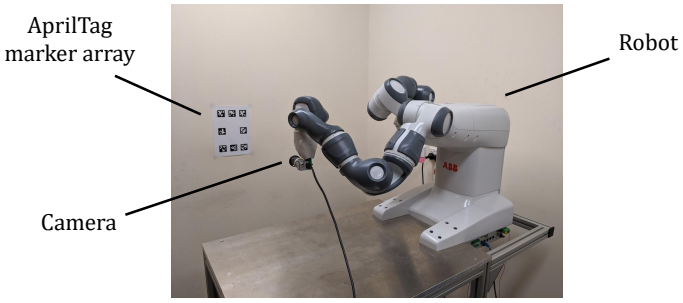


Figure 2: Experimental setup. The YuMi robot is used to simulate crane motion to simulate a video feed captured from a swinging hook block.

the YuMi follows in succession. As this motion model is much more representative of the crane's actual motion, this path is used to evaluate the performance of the algorithm.

Waypoints derived using both methods are pre-processed to be aligned to the correct coordinate frame, and to account for the offset between the YuMi end-effector coordinate frame and the camera origin coordinate frame.

## 3.2 Pose Estimation

To analyse the accuracy of the pose estimation algorithm, the measured pose of the camera from the AprilTag detection homography is compared against the actual pose as measured from the YuMi internal sensors. To compare the two results, the measurements had to be transformed into the same frame. Pose estimation from homography decomposition is in the camera frame, so had to be transformed into the global marker frame. The YuMi-derived pose is in the YuMi reference frame, so had to be similarly transformed into the global marker frame. Note that this global marker coordinate frame is equivalent to the coordinate frame of an un-rotated camera frame with its origin at the defined center of the marker. The measured and actual poses are then zeroed relative to the first measurements so that relative displacements can be measured.

Overall, the pose estimation is quite accurate. Major noise in the positional measurements are predominantly caused by noise in the orientation measurement, which seems to be the more unstable measurement between position and orientation. As expected, the use of additional AprilTag markers in the known marker improves the pose estimation accuracy. Other sources of error may include errors in the offset between gripper and camera origins, constant offsets between the YuMi orientation and the wall, vibrations in the table caused by movement of the YuMi arm, and errors in the intrinsic camera parameters.

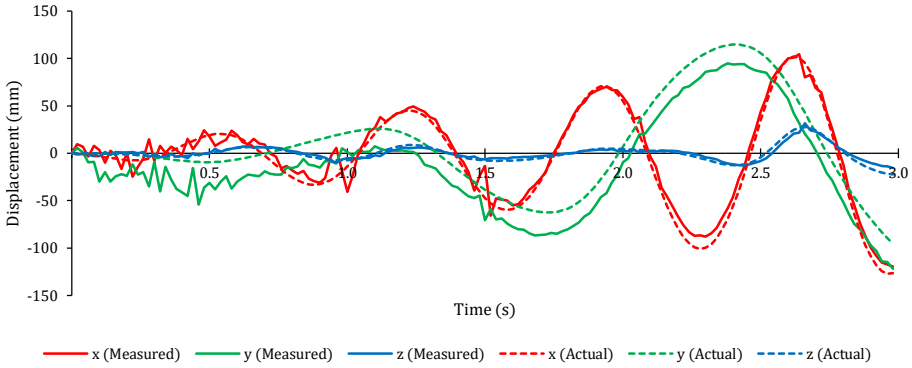


Figure 3: Comparison of measured position of camera based on AprilTag detections and the actual position as measured from the YuMi for a double pendulum dynamic simulation. The noise at the beginning is caused by the noise in the measured rotation seen in Figure 4. Steady state error in the $y$-axis measurement is similarly caused by an initial offset from noise.

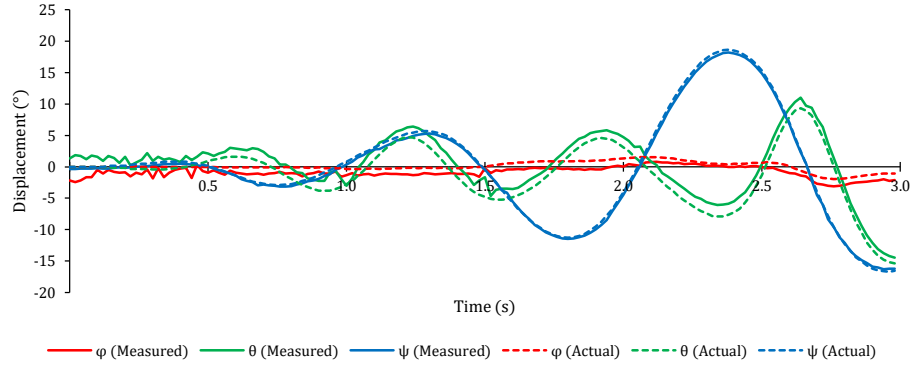


Figure 4: Comparison of measured orientation of camera based on AprilTag detections and the actual orientation as measured from the YuMi for a double pendulum dynamic simulation. Some constant offset erorr is observed, which may be accounted to the YuMi not being perfectly positioned parallel to the wall. There is a significant amount of noise observed at the beginning of the measurement.

### 3.3 Intended Motion Estimation

The performance of intended motion estimation using the single pendulum system as described in Section 2.2 is evaluated against both the single and double pendulum simulations. The single pendulum simulation was used as a proof of concept, and was performed on the real robot to see how sensitive errors in pose estimation would be to the filtering quality. The double pendulum simulation

was evaluated on the ground truth camera pose data and compared against the ground truth boom head pose to evaluate how much of an impact the simpler single pendulum model assumption would have on the more complex double pendulum model.

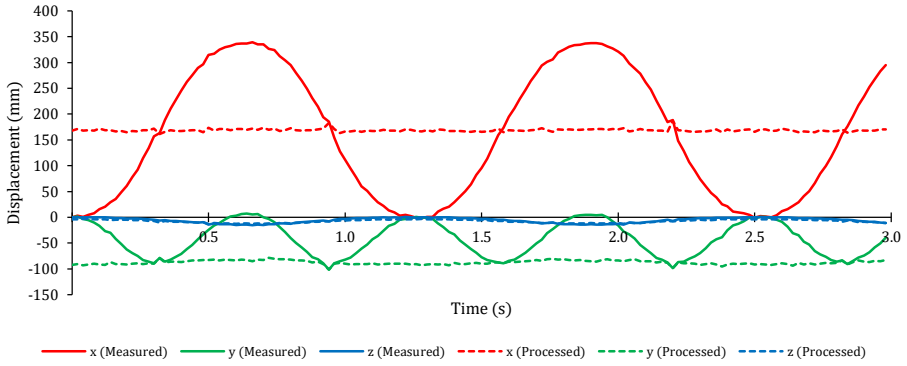


Figure 5: Comparison of measured position of camera based on AprilTag detections and the processed position following the method described in Section 2.2, based on a single pendulum kinematic simulation following a pure swinging motion. As evident, the method works extremely well to remove most unwanted swinging movement.

As expected, the single pendulum assumption used for intended motion estimation works very well for a single pendulum motion model as seen in Figure 5. However, the method breaks down with the double pendulum assumption as seen in Figure 6. This is as the measured angle of the camera is the sum of both the first and second pendulums, which exaggerates the amount the single pendulum model thinks it is swinging. This effect is more significant the more intense the significant the swinging motion is.

Hence, the single pendulum method of intended motion estimation is deemed to be too unreliable. Following results are therefore based on the method assuming a known boom position as described in Section 2.6.1, which is more robust albeit reliant on additional sensors.

## 3.4 Video Stabilization

Video stabilization results are very good. Original footage that consists of swinging motions would be filtered out. High-frequency noise from vibrations of the physical environment and errors in pose estimation are effectively removed using Kalman Filtering, resulting in a satisfyingly smooth video feed.

Mosaicking also helps tremendously with limiting information loss. However, it is sometimes reliant on the camera to have some amount of initial movement to have information to build the mosaic. Another realisation is that mosaicking is one of the most computationally expensive components of the algorithm,

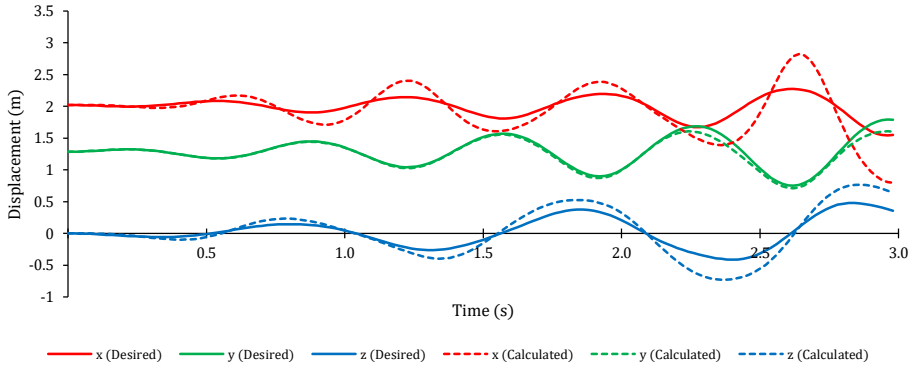


Figure 6: Comparison of the ground truth desired movement of the camera and the calculated position derived using the method described in Section 2.2, based on a double pendulum dynamic simulation. Although able to capture the general shape of motion that is desired, the calculated motion often overshoots the amount of correction it need to do due to the difference between the single and double pendulum models. This error is amplified the higher the swinging amplitude.

and has a processing time that scales quickly with image size. Hence, this limitation often acts as the upper bound for the size the mosaic can realistically be. Downscaling the raw image is a way to bypass this issue. Moreover, image blending technique such as those described by Chen [5] would help in removing artifacts caused by lighting differences.

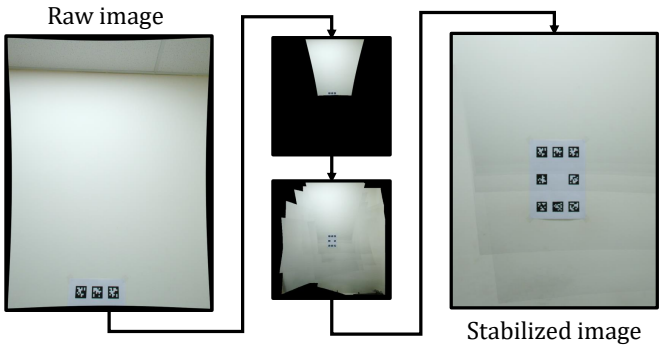


Figure 7: The process of mosaicking an image. The raw image is first warped to correspond to the desired camera position. This new warped image is then added to the previous mosaic, from which a small portion of this mosaic is cropped to form the output image.

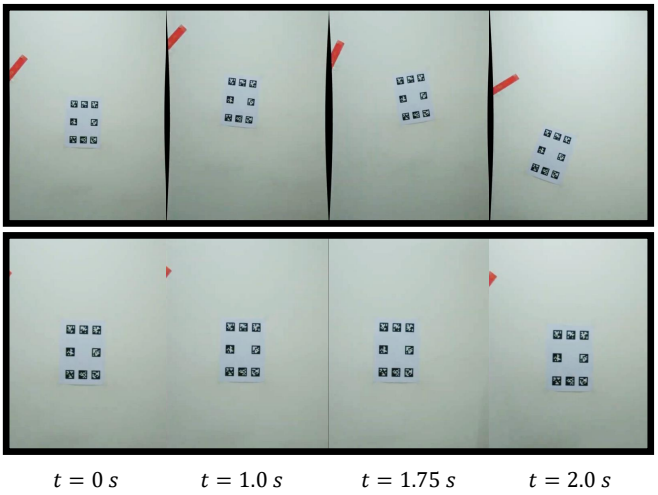


Figure 8: Image stabilization results, comparing the raw frames (top) to the processed frames (bottom). The simulated motion is that of a double pendulum dynamic model, and the intended motion estimation method assumes a known boom pose as described in Section 2.6.1.

# 4 Conclusion

The proposed algorithm is an effective way of stabilizing a video captured from a camera mounted to a swinging hook block. By relating Euclidian and perspective transforms together, we achieve a high degree of freedom in the stabilization effect we want to achieve.

The main bottleneck at the moment is an effective way of calculating the intended motion of the camera. As shown, a single pendulum model is insufficient to approximate the crane kinematics, which makes approximating the intended motion of the boom head difficult using information about the camera pose alone. Additional sensors would allow us to measure the boom head directly, and use this to define the intended motion instead. However, additional work to allow us to estimate the boom head pose directly from information provided by the camera would allow the entire algorithm to be self-sufficient, and is an interesting route for future work.

Conducting a user study using this camera feed to aid with crane operator visual feedback would also be beneficial, and if defining intentional motion of the camera as the boom head displacement is intuitive for them to understand and use to control the crane.

# References


[1] E. Gharaie, H. Lingard, and T. Cooke, "Causes of fatal accidents involving cranes in the australian construction industry," *Construction Economics and Building*, vol. 15, no. 2, pp. 1–12, 2015.

[2] J. Chen, Y. Fang, and Y. K. Cho, "Mobile Asset Tracking for Dynamic 3D Crane Workspace Generation in Real Time," in *Computing in Civil Engineering 2017*. Reston, VA: American Society of Civil Engineers, jun 2017, pp. 122–129. [Online]. Available: http://ascelibrary.org/doi/10.1061/9780784479247.083 http://ascelibrary.org/doi/10.1061/9780784480830.016

[3] H. J. Yoon, Y. C. Hwang, and E. Y. Cha, "Real-time container position estimation method using stereo vision for container auto-landing system," *ICCAS 2010 - International Conference on Control, Automation and Systems*, pp. 872–876, 2010.

[4] S. Battiato, G. Gallo, G. Puglisi, and S. Scellato, "SIFT features tracking for video stabilization," *Proceedings - 14th International conference on Image Analysis and Processing, ICIAP 2007*, no. Iciap, pp. 825–830, 2007.

[5] B. Y. Chen, K. Y. Lee, W. T. Huang, and J. S. Lin, "Capturing intention-based full-frame video stabilization," *Computer Graphics Forum*, vol. 27, no. 7, pp. 1805–1814, oct 2008. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2008.01326.x

[6] A. Litvin, J. Konrad, and W. C. Karl, "Probabilistic video stabilization using Kalman filtering and mosaicing," *Image and Video Communications and Processing 2003*, vol. 5022, no. May 2003, p. 663, 2003.

[7] W. G. Aguilar and C. Angulo, "Real-Time Model-Based Video Stabilization for Microaerial Vehicles," *Neural Processing Letters*, vol. 43, no. 2, pp. 459–477, 2016.

[8] S. Raut, K. Shimasaki, S. Singh, T. Takaki, and I. Ishii, "Real-time high-resolution video stabilization using high-frame-rate jitter sensing," *ROBOMECH Journal*, vol. 6, no. 1, 2019. [Online]. Available: https://doi.org/10.1186/s40648-019-0144-z

[9] S. Liu, H. Zhao, L. Wang, and Y. Mai, "Electronic image stabilization algorithms based on flight characteristics of the small UAV," *Airborne Intelligence, Surveillance, Reconnaissance (ISR) Systems and Applications X*, vol. 8713, no. May 2013, p. 87130W, 2013.

[10] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, no. 1, pp. 225–232, 2011.

[11] L. Wang, H. Zhao, S. Guo, Y. Mai, and S. Liu, "The adaptive compensation algorithm for small UAV image stabilization," *International Geoscience and Remote Sensing Symposium (IGARSS)*, no. 1, pp. 4391–4394, 2012.

[12] S. M. Abbas, S. Aslam, K. Berns, and A. Muhammad, "Analysis and improvements in apriltag based state estimation," *Sensors (Switzerland)*, vol. 19, no. 24, pp. 1–32, 2019.

[13] G. Simon, A. W. Fitzgibbon, and A. Zisserman, "Markerless tracking using planar structures in the scene," *Proceedings - IEEE and ACM International Symposium on Augmented Reality, ISAR 2000*, pp. 120–128, 2000.

[14] C. P. Lu, G. D. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, 2000.

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[16] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[17] E. Malis and M. Vargas, "Deeper understanding of the homography decomposition for vision-based control," Ph.D. dissertation, INRIA, 2007.

[18] R. Labbe, "Kalman and bayesian filters in python," *Chap*, vol. 7, p. 246, 2014.

[19] B. Johns, E. Abdi, and M. Arashpour, "Dynamics of luffing tower cranes: Modelling in construction and infrastructure," under review.