AIAA SciTech Forum
11–15 &amp; 19–21 January 2021, VIRTUAL EVENT
AIAA Scitech 2021 Forum

10.2514/6.2021-0017

# Usability Studies of a Predictive Heterogeneous Vision System in Mitigating the Effects of Visual Display Delay

Nazmus Sakib*, James Gresham† and Craig A. Woolsey‡
*Virginia Tech, Blacksburg, VA, 24061*

**This paper investigates the use of a heterogeneous stereo vision system in mitigating the effects of time delays present in the display that is presented to a human operator. Time delays in the display system of a telerobotic operation refer to the time difference between the operator's input action and the corresponding visible outcome. In man-machine interfaces, time delays can arise due to computation, telecommunication, and mechanical limitations. The delay needs to be overcome to bridge the gap between humans and machines as future workplaces may require more interactions between the two. A heterogeneous stereo vision predictive algorithm is presented that can reduce the negative effects of time delays present in the operator's display. The heterogeneous stereo vision system consists of an omnidirectional camera and a pan-tilt-zoom (PTZ) camera. The human operator controls the PTZ camera in the presence of time delay wearing first-person-view goggles. Delays present in a first-person-view environment can deteriorate the operator's performance and increase their workload resulting in an aversion to using man-machine systems. As a solution, a simple predictive display algorithm is developed that takes the delayed video imagery from the omnidirectional camera and stitches it to the delayed video imagery of the PTZ camera to provide an almost immediate visual response to the operator's control actions. The usability of the system is determined by doing human performance testing with and without the predictive algorithm. It has been found that the system is more efficient, more user-friendly, and more accurate in completing tasks if the predictive algorithm is implemented compared to a non-predictive case.**

## I. Introduction

To bridge the gap between humans and machines, it is necessary to enhance man-machine interfaces as future workplaces will see a synergy between human workers and semi-autonomous robotic systems. With that goal in mind, it has been proposed to investigate the use of aerial robots with augmented reality first-person view (AR/FPV) interfaces to enhance worker performance and shared situational awareness in infrastructure inspection tasks. The promise of increased efficiency, reduced workload and greater capability is attractive. However, the technology has to overcome some hurdles before it can be user friendly. One such hurdle is the presence of time delay due to video augmentation, georectification and wireless telecommunication. According to de Vries [1], some of the sources of delays include: signal transport, data encryption, data compression, error correction, and computation. Throughout this paper, the aerial robot will be referred to as a drone or unmanned air vehicle (UAV). The drone is equipped with a heterogeneous stereo vision system consisting of a pan-tilt-zoom (PTZ) camera as its "central vision" and an omnidirectional camera (i.e. with a fisheye lens) as its "peripheral vision".

A heterogeneous stereo vision system was used by Kang et al. [2] to develop a peripheral-central vision system for small UAV tracking. The system is capable of providing a threat aircraft's class and pose which other sensors like RADAR or LiDAR cannot. The system is also able to determine the range to the threat aircraft by analyzing video from two different cameras. Proper calibration of the heterogeneous camera system is required, however, before the system can be used for this or some other purpose. Rathnayaka et al. [3] presented two simple calibration techniques for a stereo camera setup with heterogeneous lenses. Instead of a conventional black-white checkerboard pattern they developed an embedded checkerboard pattern by combining two differently colored patterns. The outer pattern consists of red-blue checkers while the small inner pattern consists of red-blue-yellow-cyan checkers with an aspect ratio of 2:1 between the two patterns. In the first method they use left and right transformation matrices to calibrate the cameras. In the second method they use planar homography relationships between the two cameras for calibration. Their results

*Graduate Student, Aerospace and Ocean Engineering, Virginia Tech, Email: nazmus.sakib@vt.edu.
†Graduate Student, Aerospace and Ocean Engineering, Virginia Tech, Email: jamesgresham@vt.edu.
‡Professor, Aerospace and Ocean Engineering, Virginia Tech, AIAA Member, Email: cwoolsey@vt.edu.

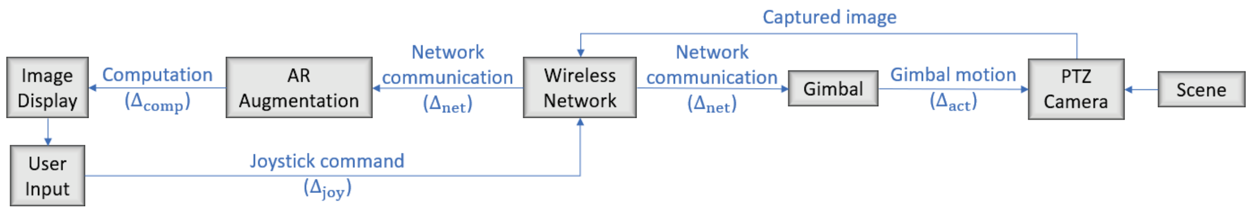


**Fig. 1 Delays involved in each step**

show that the second method is slightly more accurate than the first. However, their setup assumes fixed stereo vision cameras, that is, no relative motion between the two camera frames, which is different than the case presented in this paper. Using proper camera calibration, at first the omnidirectional camera image is undistorted. Then Zhang's pinhole camera model [4] is used for both the PTZ camera image and the undistorted omnidirectional camera image to obtain a real-time relationship between the two camera frames when there is relative motion.

Chen et al. [5] summarized human performance issues associated with time delays in teleoperated robots. By time delay the authors mean the time difference between an input action and the corresponding visible response. Having more than $170\,\mathrm{ms}$ of time delay degrades a robot operator's driving performance, tracking, telemanipulation and telepresence. It also causes over-actuation, field-of-view (FOV) reduction and motion sickness. Sheridan and Ferrell [6] show that for considerably longer delays ($1\,\mathrm{s}$ and above), telemanipulation operators employ a "move and wait" strategy where they move slowly and wait to see the outcome of their action. Then they provide a corrective input that moves the end effector closer to the goal. By repeating this discontinuous process as long as necessary to achieve a desired accuracy, the operators complete a task. Even though this method can be effective in accomplishing the task, it requires considerable time and patience from the operator. The respective authors of [5] and [6] suggest the use of "predictive displays" as a solution to counter the delay-induced effects. According to Sheridan [7], in a predictive display, a visual indicator generated by a computer is used to depict the motion of a telerobotic system. The indicator is then extrapolated forward in time to help the operator understand instantly what might happen with the given initial conditions of the system and the input.

Brudnak [8] developed an algorithm for delay-mitigation in teleoperated ground vehicles using a single camera predictive display. His approach consists of two parts: state estimation and predictive display (PD). The state estimator (SE) works in feedforward and feedback modes simultaneously. In feedforward mode, the SE accepts driver commands in the form of throttle, brake and steer and predicts an immediate response. The feedforward mode is based on a very simplified vehicle dynamic model. If only the feedforward mode is utilized the states tend to drift from actual values. Hence a feedback correction term is required to eliminate the drift in the state estimate error. The PD then takes information and error calculation from the SE and manipulates the displayed video. Combining the PD and the SE, Brudnak was able to provide the operator immediate visual feedback, which otherwise would have been delayed, regarding his/her control actions. Brudnak's results show that the assistance from the predictive display helps the operator achieve a higher average speed compared to a no PD case. However, some complexities are associated with Brudnak's method. First, the use of a state estimator requires a dynamic model of the vehicle. Second, many times the predicted frame falls outside the frame of the currently displayed image. This makes parts of the predicted image appear black or empty as information from those regions has not reached the display system yet.

The main goal of this paper is to investigate the use of a heterogeneous stereo vision system in mitigating the effects of time delays present in the display that is presented to a human operator. To meet this objective, a predictive display algorithm is developed with the capability of taking some portions of the delayed omnidirectional camera image and stitching it to the delayed PTZ camera image. This creates a composite image that represents closely what a user will see after the delay time has passed. To test the effectiveness of the algorithm, a usability study is conducted to compare a user's performance with variable time delays when the predictive algorithm is employed versus when it is not.

## II. The Delay Effect

In infrastructure inspection, an inspector is typically the operator looking at the live video feed coming from the PTZ camera through a first-person view (FPV) display. The video imagery that is being displayed contains augmented reality (AR) cues that identify structural elements (e.g. bars, beams and gusset plates), cracks and defects to assist

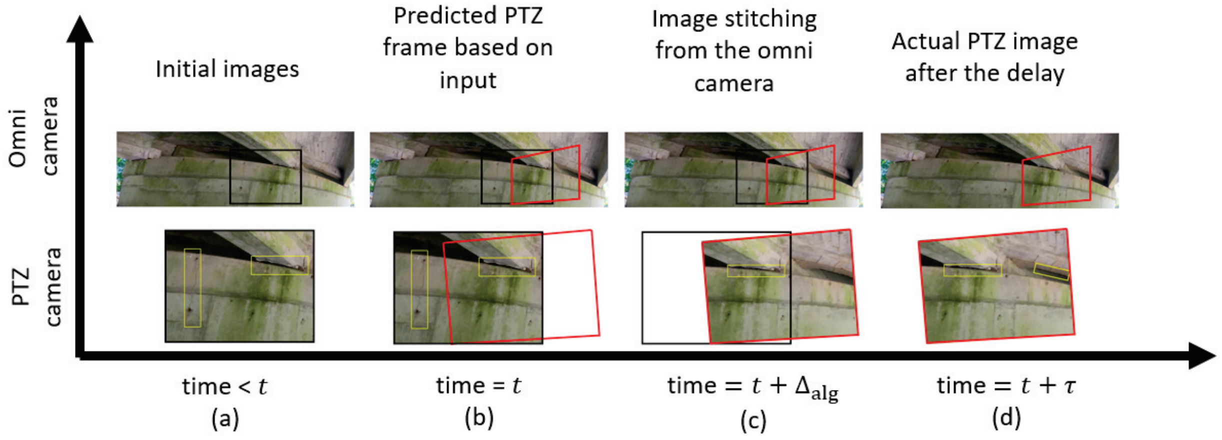

Initial images
Predicted PTZ frame based on input
Image stitching from the omni camera
Actual PTZ image after the delay
Omni camera
PTZ camera
time < $t$
(a)
time = $t$
(b)
time = $t + \Delta_{\text{alg}}$
(c)
time = $t + \tau$
(d)

**Fig. 2 Prediction timeline**

in the inspection process [9–11]. The time of acquiring an image and the time at which that image gets displayed on the screen is different due to the presence of several delays as shown in Fig. 1. These delays can hamper real time structural inspections as the camera control becomes difficult due to large overshoots or undershoots. The aerial robot technology then, instead of alleviating, increases the workload of the inspectors which can create an aversion to using the proposed technology. That is why it is important to predict the camera movement ahead of the delay time based on user input so that an infrastructure inspection can be made without having to face the delay-induced effects. Referring to Fig. 1, suppose that an image is displayed on a screen and at time $t$ a user gives a control input to the PTZ camera, that is, a reference command is sent to the gimbal that is controlling the PTZ camera's line of sight. The necessary input to the gimbal is provided through a joystick containing internal sensors with inherent delay ($\Delta_{\text{joy}}$). This input measurement is then sent to the PTZ camera gimbal through a wireless network with communication delay ($\Delta_{\text{net}}$). The gimbal receives the joystick input and its actuators take some time ($\Delta_{\text{act}}$) to reach the commanded orientation. The PTZ camera then captures the imagery along this new line of sight. This new image is then sent to a computer through the wireless network with another network delay ($\Delta_{\text{net}}$). The computer augments the image with cues, such as pointers to possible defects or georectified annotations from earlier inspections, adding some computation delay ($\Delta_{\text{comp}}$) and sends the augmented image to the user. Thus a commanded input at time $t$ gets displayed on the screen at time $t + \Delta_{\text{joy}} + 2\Delta_{\text{net}} + \Delta_{\text{act}} + \Delta_{\text{comp}}$. The total delay is defined as: $\tau = \Delta_{\text{joy}} + 2\Delta_{\text{net}} + \Delta_{\text{act}} + \Delta_{\text{comp}}$. In practice, one can expect that $\Delta_{\text{joy}}$ and $\Delta_{\text{act}}$ are much smaller than $\Delta_{\text{comp}}$ and $\Delta_{\text{net}}$ so that $\tau \approx \Delta_{\text{comp}} + 2\Delta_{\text{net}}$. The delays $\Delta_{\text{comp}}$ and $\Delta_{\text{net}}$ vary with time which makes the overall delay $\tau$ a variable that must be taken into account.

To further explain the concept, referring to Fig. 2, two different displays are shown depicting the omnidirectional camera image and the PTZ camera image versus time running along the horizontal axis. The images in the column at the far left, labeled (a), denote the view obtained from the omnidirectional and PTZ cameras mounted onboard a UAV that is hovering in place relative to a scene. The black rectangle in the omnidirectional image frame indicates the PTZ image frame that is currently displayed. The yellow rectangles indicate the augmentations – cues – identifying defects and introducing computational delay, $\Delta_{\text{comp}}$. Figure 2 does not contain representative imagery; it serves only as a sketch of the concept. The omnidirectional camera image remains the same as long as the UAV does not move. When a user input is obtained at time $t$, the current PTZ frame is calculated based on the current gimbal inertial measurement unit (IMU) angles. Then the PTZ frame which would be viewed after the delay time $\tau$ is calculated. This prediction is represented by the red quadrilateral in both the PTZ image frame and the omnidirectional image frame shown in Fig. 2b. The portion of the predicted PTZ frame that is beyond the PTZ camera's current view, as indicated by the white space in Fig. 2b, would be missing from the predicted image and is therefore replaced by imagery obtained from the omnidirectional camera. As the omnidirectional camera and the PTZ camera are located at a fixed distance from each other onboard the UAV, the image from the omnidirectional camera needs to undergo perspective transformation before it is stitched to the image from the PTZ camera. Let the time required for the algorithm to perform these calculations and transformations be $\Delta_{\text{alg}}$ which is very small compared to the delay present in the system. Thus the stitched image that predicts the estimated PTZ camera frame at time $t + \tau$ is visible almost instantly at time $t + \Delta_{\text{alg}}$. The augmented cues, however, do not get updated with this predictive algorithm and the predicted image displays parts of the old augmentations from

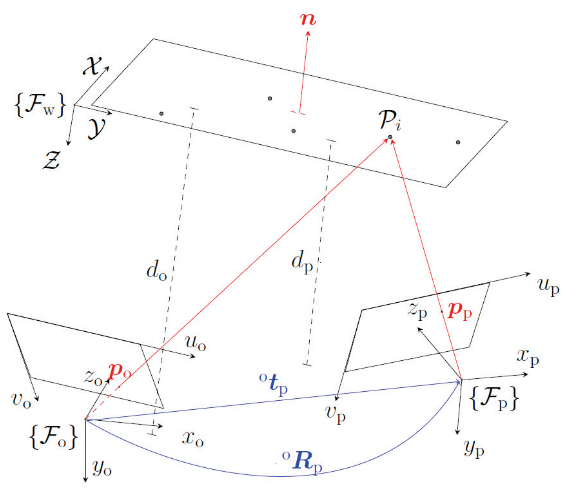


**Fig. 3 Omnidirectional and PTZ camera frames with involved notations**

time $t$ as shown in Fig. 2c. The cues get updated when the information from the actual PTZ camera image at time $t + \tau$, as shown in Fig 2d, is received for the next prediction step. As information is available from the omnidirectional camera, no portion of the predicted image remains empty as would occur with a single camera, as in [8].

## III. Perspective Projection

The theory and the notation are similar to [12] where two different camera frames are considered: omnidirectional camera frame, $\{\mathcal{F}_\mathrm{o}\}$, and PTZ camera frame, $\{\mathcal{F}_\mathrm{p}\}$, as shown in Fig. 3. Additionally $\{\mathcal{F}_\mathrm{w}\}$ is the the world coordinate frame attached to a plane. Let $\boldsymbol{\mathcal{P}}$ denote a point of interest, with Cartesian world coordinates $\boldsymbol{\mathcal{P}} = [\mathcal{X}, \mathcal{Y}, \mathcal{Z}]^\mathrm{T}$. Suppose that both cameras observe a common planar object, consisting of $i$ 3D feature points. These world coordinate points can be represented in terms of the coordinates of both the frames $\{\mathcal{F}_\mathrm{o}\}$ and $\{\mathcal{F}_\mathrm{p}\}$ and are denoted as ${}^\mathrm{o}\boldsymbol{\mathcal{P}}$ and ${}^\mathrm{p}\boldsymbol{\mathcal{P}}$, respectively. The symbols ${}^\mathrm{p}\boldsymbol{t}_\mathrm{o}$ and ${}^\mathrm{p}\boldsymbol{R}_\mathrm{o}$, respectively, indicate the translation and rotation of the omnidirectional camera frame with respect to the PTZ camera frame. The perpendicular distances from the object plane to the corresponding camera frames are $d_\mathrm{o}$ and $d_\mathrm{p}$, respectively. The normal to the object plane, $\boldsymbol{n}$, can also be expressed in the $\{\mathcal{F}_\mathrm{o}\}$ and $\{\mathcal{F}_\mathrm{p}\}$ frame coordinates as ${}^\mathrm{o}\boldsymbol{n}$ and ${}^\mathrm{p}\boldsymbol{n}$, respectively.

### *1. PTZ Camera Model and Calibration.*

For the PTZ camera, a simple pinhole model described in [4] is used. A point in the object plane in world coordinates can be represented in the homogeneous form by augmenting the coordinates with a "1" as: $\tilde{\boldsymbol{\mathcal{P}}} = [\mathcal{X}, \mathcal{Y}, \mathcal{Z}, 1]^\mathrm{T}$. An image of an object is grabbed by the PTZ camera in a way that its 3D feature point in world coordinates, $\boldsymbol{\mathcal{P}}$, is projected on an image plane having the same depth from the PTZ camera origin. If $\boldsymbol{p}_\mathrm{p} = [u_\mathrm{p}, v_\mathrm{p}]^\mathrm{T}$ is the projection of $\boldsymbol{\mathcal{P}}$ on the image plane, in pixels, and $\tilde{\boldsymbol{p}}_\mathrm{p} = [u_\mathrm{p}, v_\mathrm{p}, 1]^\mathrm{T}$ is the homogeneous transformation of $\boldsymbol{p}_\mathrm{p}$ then the relationship between the two is given by:

$$\lambda \tilde{\boldsymbol{p}}_\mathrm{p} = \boldsymbol{K}_p [{}^\mathrm{p}\boldsymbol{R},\ {}^\mathrm{p}\boldsymbol{t}] \tilde{\boldsymbol{\mathcal{P}}}_i \quad \text{with } \boldsymbol{K}_p = \begin{bmatrix} \alpha f_p & \gamma & u_0 \\ 0 & \beta f_p & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where $\lambda$ is an arbitrary scale factor that captures the depth information of the object plane with respect to the camera frame $\{\mathcal{F}_\mathrm{p}\}$. The matrix $[{}^\mathrm{p}\boldsymbol{R},\ {}^\mathrm{p}\boldsymbol{t}]$ is called the extrinsic parameter matrix consisting of the rotation and the translation of the object plane in the world coordinate system with respect to the PTZ camera coordinates. The matrix $\boldsymbol{K}_p$ is the camera intrinsic matrix with $(u_0, v_0)$ as the pixel coordinate of the principal point. By definition, the principal point is the point on the image plane onto which the perspective center is projected and from Fig. 3 the coordinates of the principal point is: $(u_0, v_0) = (u_\mathrm{p}/2, v_\mathrm{p}/2)$. The symbols $\alpha$ and $\beta$ are the scale factors in the $u_\mathrm{p}$ and $v_\mathrm{p}$ axes, $f_\mathrm{p}$ is the focal length of the PTZ camera, and $\gamma$ is the parameter that describes the skew of the two image axes. All of the quantities of the matrix $\boldsymbol{K}_\mathrm{p}$ can be obtained by following the calibration technique outlined in [4].

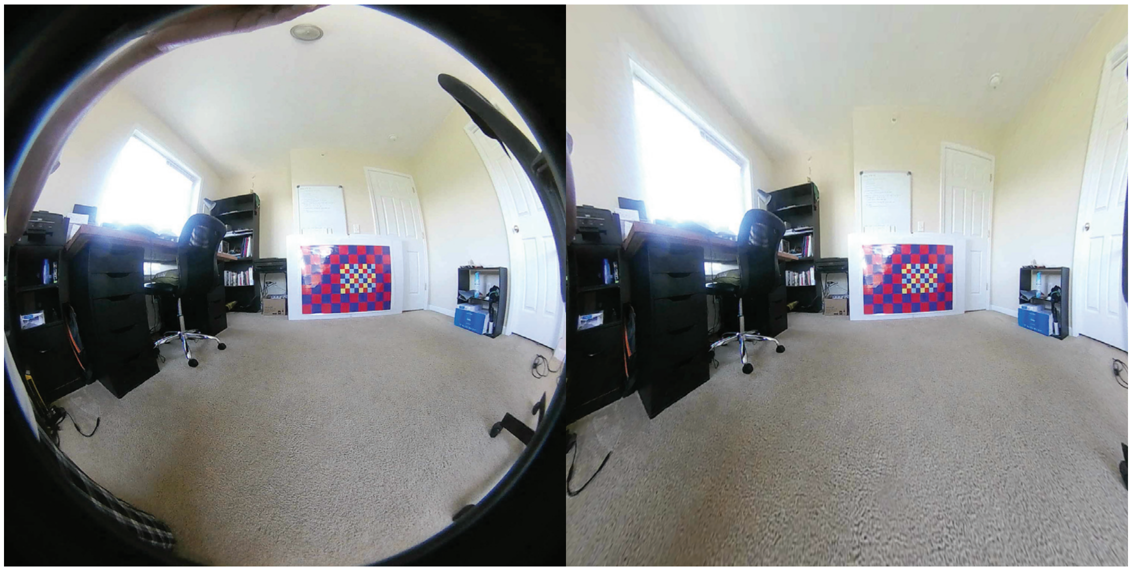


**Fig. 4 Original omnidirectional camera image (left) and undistorted omnidirectional camera image (right)**

## A. Omnidirectional Camera Model and Calibration.

Omnidirectional cameras are special in a sense that they have much larger FOV and significant distortion because of the fisheye lenses they use. The images thus obtained are not suitable for use unless they are first undistorted into a simple pinhole projection model. The omnidirectional camera calibration and undistortion can be done using common `OpenCV` functions that implement the algorithm described in [13], which is based on the following equation:

$$^{\mathrm{o}}\boldsymbol{\mathcal{P}} = [^{\mathrm{o}}\boldsymbol{R},\ ^{\mathrm{o}}\boldsymbol{t}]\tilde{\boldsymbol{\mathcal{P}}}_i \tag{2}$$

The matrix $[^{\mathrm{o}}\boldsymbol{R},\ ^{\mathrm{o}}\boldsymbol{t}]$ is called the extrinsic parameter matrix consisting of the rotation and the translation of the object plane in the world coordinate system with respect to the omnidirectional camera coordinates. The pinhole projection coordinates, $[a_{\mathrm{o}}, b_{\mathrm{o}}]^{\mathrm{T}}$, are then converted to the distorted coordinates using the following procedure:

If $^{\mathrm{o}}\boldsymbol{\mathcal{P}} = [x_{\mathrm{o}}, y_{\mathrm{o}}, z_{\mathrm{o}}]^{\mathrm{T}}$ then:

$$\begin{aligned}
&a_{\mathrm{o}} = x_{\mathrm{o}}/z_{\mathrm{o}} \text{ and } b_{\mathrm{o}} = y_{\mathrm{o}}/z_{\mathrm{o}}\\
&\qquad\Rightarrow \quad r^2 = a_{\mathrm{o}}^2 + b_{\mathrm{o}}^2\\
&\qquad\Rightarrow \quad \theta = \tan^{-1}(r)\\
&\qquad\Rightarrow \quad \theta_{\mathrm{d}} = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \quad [\theta_{\mathrm{d}} = \text{Fisheye lens distortion}]
\end{aligned}$$

Then the distorted point coordinates are:

$$x_{\mathrm{d}} = \frac{\theta_{\mathrm{d}} a_{\mathrm{o}}}{r} \qquad \text{and} \qquad y_{\mathrm{d}} = \frac{\theta_{\mathrm{d}} b_{\mathrm{o}}}{r} \tag{3}$$

Finally, on converts to pixel coordinates as follows:

$$\eta\tilde{\boldsymbol{p}}_{\mathrm{o}} = \boldsymbol{K}_{\mathrm{o}}[x_{\mathrm{d}}, y_{\mathrm{d}}, 1]^{\mathrm{T}} \tag{4}$$

The matrix $K_{\mathrm{o}}$ takes the same form as $K_{\mathrm{p}}$ and can be obtained during the omnidirectional camera calibration along with the distortion coefficients $k_1, k_2, k_3$, and $k_4$. Using the equations above, the omnidirectional camera frame can be undistorted as shown in Fig. 4. The arbitrary scale factor $\eta$ captures the depth information of the object plane with respect to the camera frame $\{\mathcal{F}_{\mathrm{o}}\}$. In Fig. 4 the edges of the undistorted image have been cropped to avoid division by $z_{\mathrm{o}} = 0$ for the pixel points near the edges; otherwise, one would have $a_{\mathrm{o}}, b_{\mathrm{o}} \to \infty$.

### B. Transformation Between the Frames $\{\mathcal{F}_\mathrm{o}\}$ and $\{\mathcal{F}_\mathrm{p}\}$.

According to [12], two types of homographies relate one camera frame to another. The euclidean homography matrix, $\boldsymbol{H}$, transforms 3D points in the euclidean space from one frame to the other whereas the projective homography matrix, $\boldsymbol{G}$, transforms pixel coordinates from one camera frame to the other one. The euclidean homography matrix can be obtained from a known camera displacements using:

$$\boldsymbol{H} = {}^\mathrm{o}\boldsymbol{R}_\mathrm{p} + \frac{{}^\mathrm{o}\boldsymbol{t}_\mathrm{p} \cdot {}^\mathrm{o}\boldsymbol{n}^\mathrm{T}}{d_\mathrm{o}} \tag{5}$$

Then the projective homography, $\boldsymbol{G}$, can be obtained from the euclidean homography, $\boldsymbol{H}$ using:

$$\boldsymbol{G} = \boldsymbol{K}_\mathrm{p}\boldsymbol{H}\boldsymbol{K}_\mathrm{o}^{-1} \tag{6}$$

Equation (6) takes vectors of pixel points in the undistorted omnidirectional camera image and gives the corresponding vectors of pixel points in the PTZ camera image. Let, $\boldsymbol{G}_t$ and $\boldsymbol{G}_{t+\tau}$ be the projective homographies that exist between the omnidirectional camera frame and the PTZ camera frame at times $t$ and $t + \tau$, respectively. Then the projective homography that exists between the same camera frames but at two different times is expressed as:

$$\boldsymbol{G}_{t\tau} = \boldsymbol{G}_{t+\tau}\boldsymbol{G}_t^{-1} \tag{7}$$

Equation (7) converts a vector of pixel coordinates of the camera frame at time $t$ to a vector of corresponding pixel coordinates of the same camera frame at time $t + \tau$.

## IV. Human Performance Evaluation

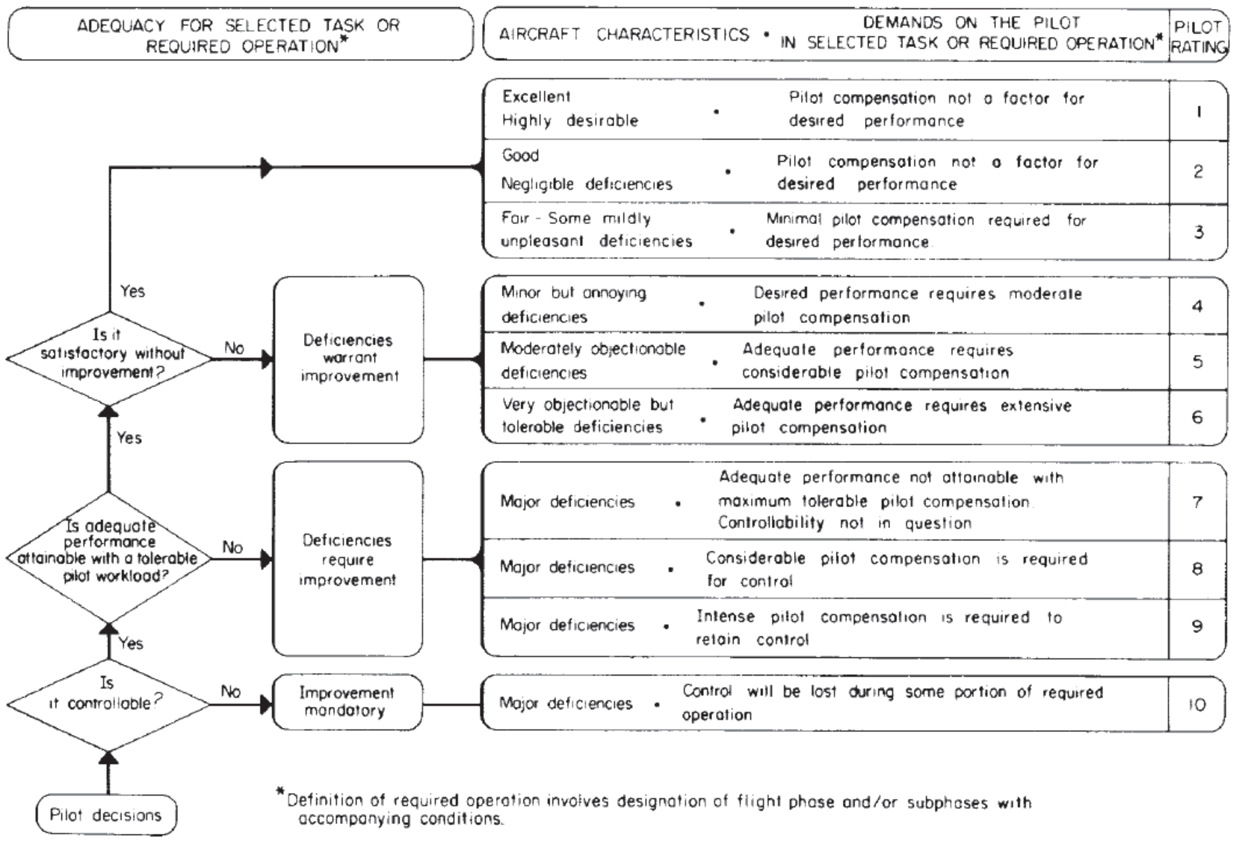

ADEQUACY FOR SELECTED TASK OR REQUIRED OPERATION*
AIRCRAFT CHARACTERISTICS • DEMANDS ON THE PILOT IN SELECTED TASK OR REQUIRED OPERATION*
PILOT RATING
Excellent Highly desirable • Pilot compensation not a factor for desired performance 1
Good Negligible deficiencies • Pilot compensation not a factor for desired performance 2
Fair - Some mildly unpleasant deficiencies • Minimal pilot compensation required for desired performance 3
Minor but annoying deficiencies • Desired performance requires moderate pilot compensation 4
Moderately objectionable deficiencies • Adequate performance requires considerable pilot compensation 5
Very objectionable but tolerable deficiencies • Adequate performance requires extensive pilot compensation 6
Major deficiencies • Adequate performance not attainable with maximum tolerable pilot compensation. Controllability not in question 7
Major deficiencies • Considerable pilot compensation is required for control 8
Major deficiencies • Intense pilot compensation is required to retain control 9
Major deficiencies • Control will be lost during some portion of required operation 10
Yes
Is it satisfactory without improvement?
No
Deficiencies warrant improvement
Yes
Is adequate performance attainable with a tolerable pilot workload?
No
Deficiencies require improvement
Yes
Is it controllable?
No
Improvement mandatory
Pilot decisions
*Definition of required operation involves designation of flight phase and/or subphases with accompanying conditions.

**Fig. 5 Cooper-Harper Handling Qualities Rating (HQR) scale [14]**

Another goal of this paper is to study performance of the algorithm through human subject testing. Several subjective evaluation methods are followed in different scientific communities. The human-computer interaction (HCI) society, for example, uses System Usabilty Scale (SUS). NASA, on the other hand, developed their own Task Load Index (TLI) for subjective assessments. In this paper, however, the human performance assessments will be done through the Cooper-Harper Handling Qualities Rating (HQR) scale [14]. The Cooper-Harper HQR scale is widely used by aircraft test pilots and flight test engineers to evaluate the handling qualities of an aircraft in performing a specific task. Although the scale is used widely in aircraft handling qualities assessment by pilots, the scale is generalized enough to be applicable to other systems. The scale is also able to capture the effects of time delays present in a system by comparing the performance deterioration due to the delay with a similar non-delayed task. In fact, Jennings et al. [15] determined the delay tolerance limit of a pilot in a helicopter simulation using the Cooper-Harper HQR scale. Their results showed an increased variability of position error with as little as $134$ ms of delay. Handling qualities deteriorated as the delay was increased. The justification for using the HQR scale is that it is widely known in the aerospace/aeronautical communities. Even though all the tests for this paper were performed while keeping the UAV on the ground, the algorithm's performance will ultimately be evaluated in flight tests.

According to [16], there are two types pilots in aircraft testing: a high-gain pilot, and a low-gain pilot. Typically, the low-gain pilot injects inputs that are smoother and smaller in amplitude than the inputs by a high-gain pilot. High-gain pilots are expected to encounter the non-linear responses in handling qualities more than low-gain pilots, and such pilots are great for exposing potential limitations in the aircraft handling qualities. Similar to the test pilots, the participants taking part in this research may be thought of as low-gain and high-gain participants. A good experimental design is one where both the low-gain and high-gain participants rate the system similarly. To enable assessment using the Cooper-Harper HQR scale, bounds for *desired* and *adequate* performance must be established; see Figure 5. Two different sets of bounds were chosen for this experiment, one in the spatial domain and the other in the temporal domain.

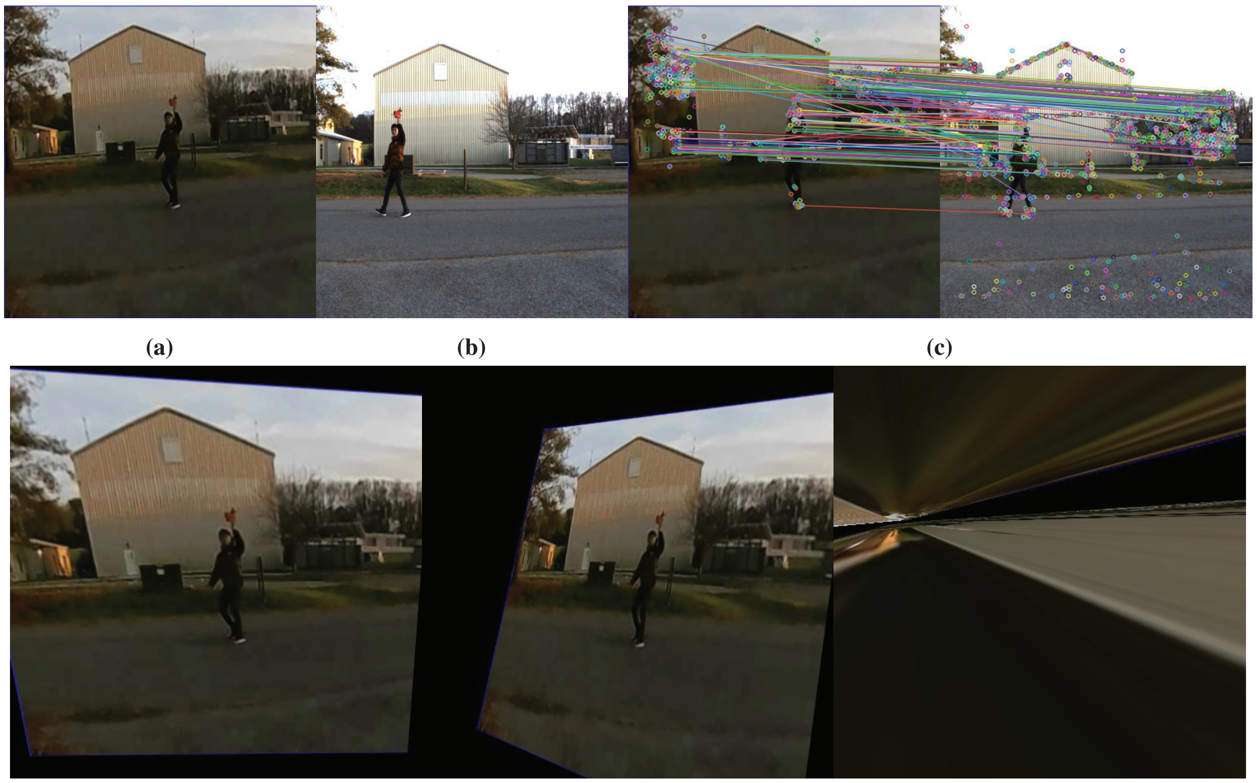


**Fig. 6 Two reference images: (a) Omnidirectional camera image and (b) PTZ camera image; (c) feature matching with ORB algorithm; (d) warped image after applying the homography matrix with a maximum of 10,000 feature points. Two incorrectly warped images obtained using a maximum of (e) 3000 feature points and (f) 50 feature points.**

In the spatial domain, participants were required to track a line with a cursor with as much precision as possible. This task assignment forced the high-gain operators to slow down a little to achieve the desired path. The temporal bounds, involving minimum completion times for desired and adequate performance, forced the low-gain operators to speed up their response a bit to complete the task within the time boundaries. To claim the system exhibits desired performance, for the rating purpose, a participant must have achieved desired performance with respect to both the temporal and spatial specifications.

## V. Estimating the Homography Matrix

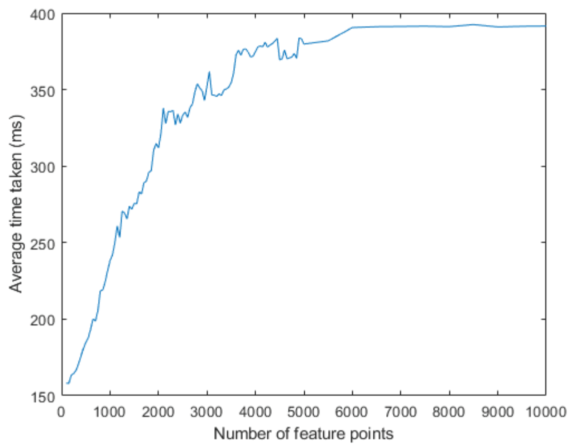


**Fig. 7 Representative illustration of homography computation time versus maximum number of feature points**

In computer vision applications, the homography matrix is estimated by finding good feature points that are visible in both of the images of interest and then applying the `RANSAC` algorithm to obtain the best estimate. Of course, if the heterogeneous camera system is fixed and the two cameras do not rotate with respect to each other then the homography matrix remains constant and can be pre-computed before running real-time image stitching. The case considered in this paper is different where the PTZ camera rotates with respect to the omnidirectional camera, thus the homography matrix keeps changing as the camera rotates and zooms in or out. Figure 6 shows different homography matrices computed using `ORB` and `RANSAC` algorithms in `OpenCV`. Figure 6a represents the delayed PTZ camera frame view, cropped from the low-resolution, undistorted omnidirectional camera image. Figure 6b represents the actual delayed PTZ camera image. (The two images are merely illustrative.) To perform proper image stitching, it is necessary to warp Fig. 6a to make it look similar to Fig. 6b, that is, to do a perspective warp. Figure 6c shows the matched feature points using `ORB` and, at most, $10,000$ feature points. The top $15\%$ of all the detected features are kept as the best matched feature points which are then used to compute the homography matrix using the `RANSAC` algorithm. The obtained homography matrix is then used to warp the omnidirectional camera image in Fig. 6a to obtain the omnidirectional camera image viewed from the PTZ camera perspective; see Fig. 6d. However, the estimation is not robust if a lower number of detected feature points are used. Figure 6e is obtained by warping Fig. 6a using a homography matrix computed using $3000$ matched feature points. The error in the estimation is very large. For the case of $50$ feature points only, Fig. 6f, the estimation is wrong and the resultant warped image is incorrect.

From Fig. 7 it can be seen that the average computation time (obtained by averaging $100$ calculations at each of the maximum number of feature points: $100$, $150$, $200$ ... etc) increases as the number of feature points increase but after reaching $6500$ feature points the computation time remains the same, reflecting the fact that `ORB` cannot detect more than $6500$ feature points in between the two reference images. Regardless, for the given hardware and software, using only $100$ feature points requires about $160$ ms. It is impossible to use feature detection techniques to compute real-time homography matrices at each instant of the video imagery using the hardware and software considered here. To overcome this problem, homography matrices are calculated using Eqn. (6).

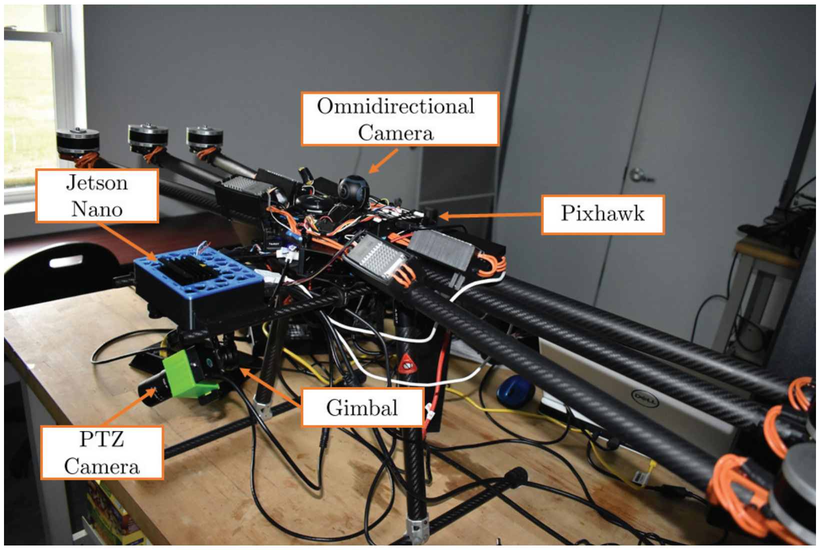


**Fig. 8 System setup: Two cameras mounted on a UAV**

## VI. Heterogeneous Stereo Vision Algorithm

### A. System Setup

Different camera models and parameters used in this experiment are shown in Table 1 and a picture of the setup mounted on a UAV (stowed for transport) is shown in Fig. 8. The radius vector from the omnidirectional camera frame to the PTZ camera frame was calculated to be: ${}^{\mathrm{o}}\boldsymbol{t}_{\mathrm{p}} = [12.7, 152, 267]^{\mathrm{T}}\ \mathrm{mm}$. The gimbal was calibrated beforehand to align the PTZ camera axes with the omnidirectional camera axes. The heterogeneous stereo vision algorithm is implemented in real-time using ROS (Robot Operating System) which is a collection of tools, libraries and conventions that aim to simplify robotic hardware and software integration. An NVIDIA Jetson Nano acted as the on-board computer. The Nano was responsible for running the necessary ROS drivers for the two different cameras and the gimbal. The messages generated by the cameras and the gimbal were then sent to a ground station through a wired local area network (LAN) connection. The average delay in the wired connection path, $\Delta_{\mathrm{net}}$, was found to be $4.3\,\mathrm{ms}$ and was ignored since it was negligible compared to the total delay, $\tau$. The ground station, a Linux machine, received the ROS messages containing camera images and the gimbal's IMU angles. It then ran the image stitching and predicting algorithm and displayed the output image. The gimbal was controlled through a Pixhawk 4 avionics unit using the Mission Planner software interface. These elements required the use of another laptop, running the Windows operating system, which took the joystick inputs and sent them to the gimbal. The amount of delay in this message transmission could not be calculated, but was observed to be negligible. The joystick used for this experiment was a generic Microsoft Xbox controller. The Windows machine also ran a simple python code to measure the time that elapses while depressing a particular button on the Xbox controller. Table 2 contains some details about the different hardware used.

To test the validity of the image stitching and prediction algorithm the entire experiment was carried out on the ground where wired LAN connections could be established and the disturbances and noises were minimal. The wired connection was important because it made the network delay, $\Delta_{\mathrm{net}}$, minimal which meant that the computation delay $\Delta_{\mathrm{comp}}$ was the dominant delay term in the total delay of the system, $\tau$. This would not be the case for a wireless LAN connection and its effects will be studied in the future. The total delay of the system, $\tau \approx \Delta_{\mathrm{comp}}$, was specified through the implementation of the `message_filters` package in ROS. The `TimeSequencer()` function in that package guaranteed that ROS messages would be called in the temporal order in which they were received. All ROS messages contain timestamps indicating when the messages were acquired. If a specific delay amount has been specified the `TimeSequencer` filters out and stores the messages based on their timestamp until the messages are out of date by at least the specified delay amount. This enables the creation of a simulated environment where the images displayed on the screen are delayed by at least the specified amount. In this way, various amounts of delay were injected to the incoming images and the IMU angles. It had been observed that the ROS `image_transport` package, responsible for publishing or subscribing to image messages, published the PTZ images at $16\,\mathrm{Hz}$ and the omnidirectional camera

**Table 1 Camera specifications**

| Parameters | Omnidirectional Camera | PTZ Camera |
|---|---|---|
| Model | Insta 360 Air | Webcamera USB |
| Focal Length (mm) | $1.0$ | $5 - 50$ |
| Sensor Size (mm) | $3.3 \times 3.3$ | $4.8 \times 3.6$ |
| Pixel Size ($\mu$m) | $2.19 \times 2.19$ | $3 \times 3$ |
| Resolution (px) | $1504 \times 1504$ | $1920 \times 1080$ |
| Mass (g) | $26.5$ | $330$ |

images at $7\,\mathrm{Hz}$. Image messages are large because each message contains pixel intensities of three different color channels (red, green and blue) and one alpha channel (a component that represents the degree of transparency or opacity of a color). The PTZ camera with resolution $1920 \times 1080$ sends $1920 \times 1080 \times 4 = 8.294\,\mathrm{Megapixels}$ per frame. As each of the pixels take $32\,\mathrm{bits}$ of data approximately $31.64\,\mathrm{Mbits}$ of data per frame is required for the PTZ camera. Similarly, the omnidirectional camera with resolution $3008 \times 1504$ sends $3008 \times 1504 \times 4 \times 32 \approx 69.04\,\mathrm{Mbits}$ of data per frame. In this case, the camera captures images faster than they can be published as ROS messages and some of frames are thus discarded. So, in one second, only 16 PTZ and 7 omnidirectional camera frames can be published before using all the available bandwidth ($1000\,\mathrm{Mbit/s}$) of a gigabit Ethernet LAN cable. Although the stitching and prediction algorithm could run as fast as $50\,\mathrm{Hz}$, it was slowed to down to run only at $15\,\mathrm{Hz}$. Running the algorithm faster would not improve performance as the PTZ camera frame rate was maxed out at $16\,\mathrm{Hz}$.

**Table 2 Hardware specifications**

|  | Jetson Nano | Linux Machine | Windows Machine |
|---|---|---|---|
| CPU | Quad core ARM A57 | Intel Core i7-8750H | Intel Core i7-8550U |
| GPU | NVIDIA maxwell 128 core | GeForce RTX 2060 | Intel UHD Graphics 620 |
| Memory (GB) | 4 | 16 | 16 |
| Operating System | Ubuntu 18.04 | Ubuntu 16.04 | Windows 10 |

## B. Algorithm Steps

The real time implementation of the heterogeneous stereo vision image stitching algorithm can be described in the following eight steps. The algorithm execution time $\Delta_{\mathrm{alg}} \approx 20\,\mathrm{ms}$, which means that it can be run as fast as $50\,\mathrm{Hz}$.

Step 1. The gimbal IMU angles are obtained in the form of quaternions. They are converted to Euler angles and stored in a vector along with their respective timestamps. The most recent of these IMU angles are taken as the pitch and yaw angles ($\theta_{t+\tau}$ and $\psi_{t+\tau}$) at the predicted frame $\{\mathcal{F}\}_{t+\tau}$. The logic behind doing this is that the joystick input at time $t$ moves the gimbal almost instantly, as $\Delta_{\mathrm{net}}$ is assumed to be negligible, and the movement gets displayed at time $t + \tau$.

Step 2. The algorithm then looks back at time $t - \tau$ to find the gimbal angles ($\theta_t$ and $\psi_t$) corresponding to the currently displayed image frame $\{\mathcal{F}\}_t$. The time value $t - \tau$ is compared against the vector of timestamps and the angles corresponding to the closest time are chosen as the gimbal angles for the current frame $\{\mathcal{F}\}_t$. The storage vector should be sufficiently large so that the time value $t - \tau$ is always bigger than the timestamp of the first element. For this paper 100 elements were stored in the storage vector before the oldest value got replaced by the newest value. As the algorithm was running at $15\,\mathrm{Hz}$, 100 values meant that the vector stored the gimbal angles for the past $6.67\,\mathrm{s}$.

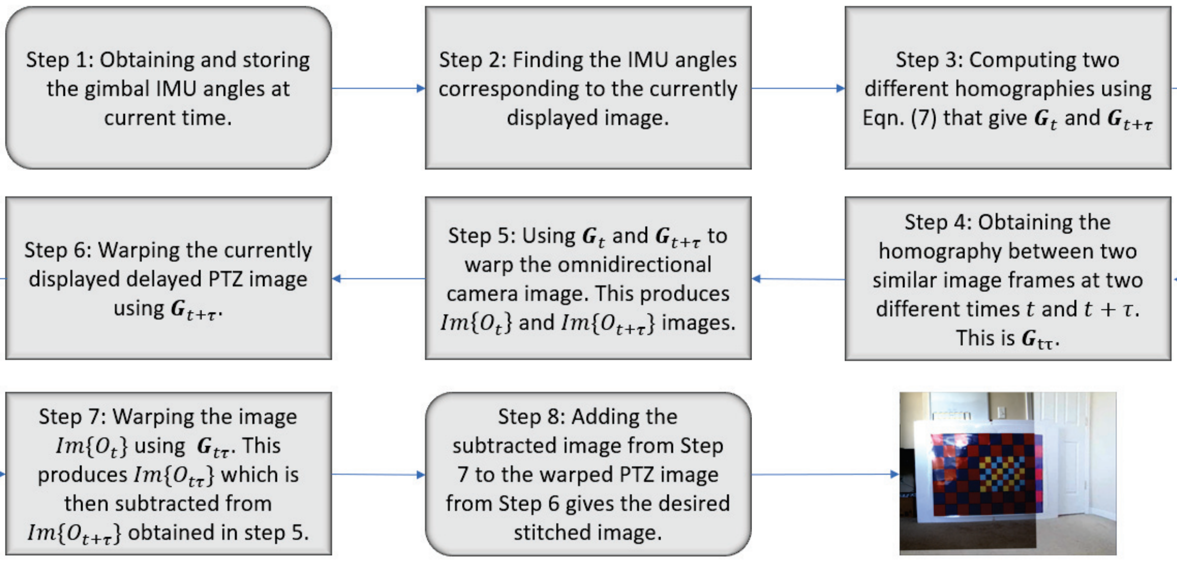

Step 1: Obtaining and storing the gimbal IMU angles at current time.
Step 2: Finding the IMU angles corresponding to the currently displayed image.
Step 3: Computing two different homographies using Eqn. (7) that give $G_t$ and $G_{t+\tau}$
Step 4: Obtaining the homography between two similar image frames at two different times $t$ and $t+\tau$. This is $G_{t\tau}$.
Step 5: Using $G_t$ and $G_{t+\tau}$ to warp the omnidirectional camera image. This produces $Im\{O_t\}$ and $Im\{O_{t+\tau}\}$ images.
Step 6: Warping the currently displayed delayed PTZ image using $G_{t+\tau}$.
Step 7: Warping the image $Im\{O_t\}$ using $G_{t\tau}$. This produces $Im\{O_{t\tau}\}$ which is then subtracted from $Im\{O_{t+\tau}\}$ obtained in step 5.
Step 8: Adding the subtracted image from Step 7 to the warped PTZ image from Step 6 gives the desired stitched image.

**Fig. 9 Flowchart of the proposed algorithm with the final output in the end**

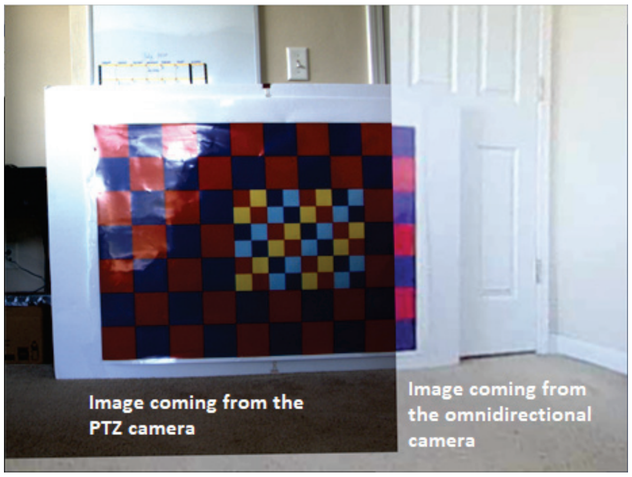


Image coming from the PTZ camera
Image coming from the omnidirectional camera

**Fig. 10 Stitched image using heterogeneous cameras. The checkerboard pattern is obtained from [3].**

Step 3. After obtaining $\theta_{t+\tau}$, $\psi_{t+\tau}$, $\theta_t$ and $\psi_t$ two different projective homographies are calculated – $\boldsymbol{G}_{t+\tau}$ and $\boldsymbol{G}_t$ – using Eqn. (6).

Step 4. Using Eqn. (7) the homography between two similar image frames at two different times $t+\tau$ and $t$ is obtained. This is $\boldsymbol{G}_{t\tau}$ for both the omnidirectional and the PTZ camera frames. The main difference between $\boldsymbol{G}_{t\tau}$ and $\boldsymbol{G}_t$ or $\boldsymbol{G}_{t+\tau}$ is that the former is a transformation between the same camera frames at two different times whereas the latter two are transformations at some particular time between two different camera frames.

Step 5. The homographies, $\boldsymbol{G}_{t+\tau}$ and $\boldsymbol{G}_t$, obtained in Step 3 are used to warp the delayed omnidirectional camera image; the `OpenCV` function `cv::warpPerspective()` is used here. The resulting images $Im\{O_{t+\tau}\}$ and $Im\{O_t\}$ are (undistorted) omnidirectional camera images, but from the perspective of the PTZ camera at times $t+\tau$ and $t$, respectively.

Step 6. Next, the currently displayed PTZ image, $Im\{PTZ_t\}$, is warped using $\boldsymbol{G}_{t\tau}$, obtained in Step 4, to obtain $Im\{PTZ_{t\tau}\}$. The image $Im\{PTZ_{t\tau}\}$ actually contains the currently displayed PTZ image but from the perspective of the predicted frame at $t+\tau$ due to a joystick input at time $t$.

Step 7. Similarly, $Im\{O_t\}$ is warped to $Im\{O_{t\tau}\}$ using $\boldsymbol{G}_{t\tau}$. The image $Im\{O_{t\tau}\}$ is then subtracted from $Im\{O_{t+\tau}\}$ using the function `cv::subtract()`. The subtracted image contains the portion of the image in $Im\{O_{t+\tau}\}$ that

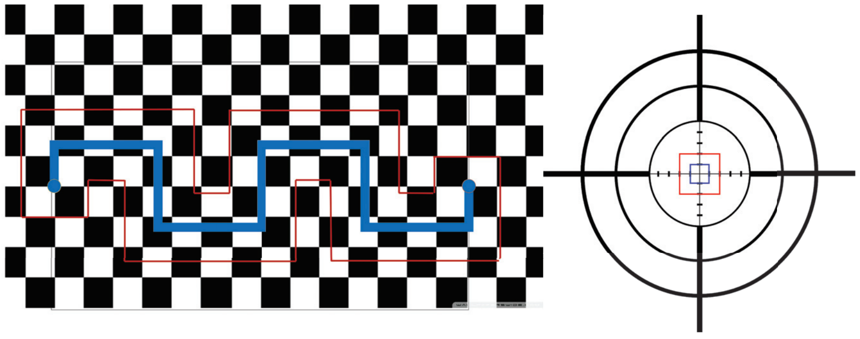

(a) (b)

**Fig. 11 Images used for the (a) tracking and (b) target acquisition tasks.**

is not present in $Im\{O_{t\tau}\}$.

Step 8. Finally, the subtracted image in Step 7 is added to $Im\{PTZ_{t\tau}\}$, obtained from Step 6, using the function `cv::add()`. The resulting image is the composite image consisting of delayed omnidirectional and PTZ camera images. The whole process takes about $20\,\mathrm{ms}$ and is then displayed on the screen. Thus, an image which was supposed to be displayed at time $t+\tau$ gets displayed at time $t+\Delta_{\mathrm{alg}} \ll t+\tau$. An example of the composite stitched image is shown in Fig. 10.

## VII. System Usability Determination

### A. Study Design

A human performance study was designed to test the effectiveness of the proposed predictive display system in mitigating display delays in the FPV environment. The study was designed to capture the common maneuvers that a typical bridge inspector might perform while using the system. Participants wore an Eachine EV800D FPV headset and completed several tasks under three different test cases:

(i) delayed case 1: $500\,\mathrm{ms}$ delay
(ii) delayed case 2: $1000\,\mathrm{ms}$ delay
(iii) delayed case 3: $1500\,\mathrm{ms}$ delay

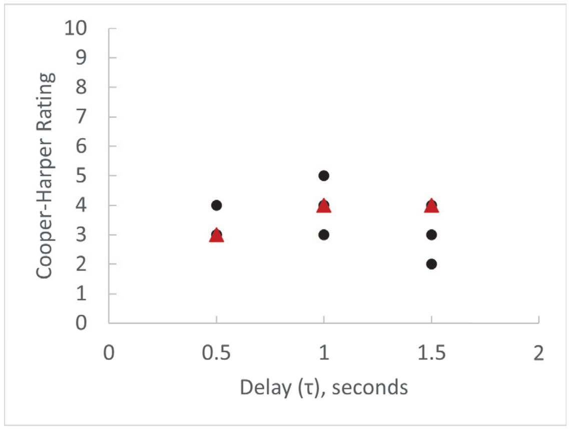
In each of the cases (i)-(iii), a precision tracking task (a) was performed along the $xy$-plane followed by a target acquisition task (b). The PTZ camera images for the tracking tasks were saved for later analysis of the cursor paths generated by each participant. The total error in a tracking task was obtained as the total (unsigned) area of the regions that fell outside the desired bounds. The Cooper-Harper rating scale was used primarily for the subjective evaluation of the system. In the precision tracking task the participants were asked to track a stepped line from left to right, as shown in Fig. 11a, as closely and as accurately as possible. The thick blue line was the desired bound whereas the red line was the adequate bound in the spatial domain. The center of the displayed image was augmented with a red dot, indicating a cursor location, which the operator controlled in order to track the line as closely as possible. To achieve the desired performance, participants had to ensure that the red dot at least touched the thick blue line while completing the tracking task. The temporal bounds were specified as $45\,\mathrm{s}$ for the desired performance and $90\,\mathrm{s}$ for the adequate performance. The spatial and temporal bounds for desired and adequate performance were used for all the test cases, with or without prediction.

In the target acquisition task the participants were asked to acquire a target, by placing the camera red dot in the center of a bullseye from an initial orientation as fast as possible (Fig. 11b). The blue and the red squares in the image indicate desired and adequate performance in the spatial domain, respectively. The temporal bounds were set to $15\,\mathrm{s}$ for desired performance and $20\,\mathrm{s}$ for adequate performance. Again, just like in the tracking task, the spatial and the temporal bounds remained unchanged for all the different cases with or without prediction.

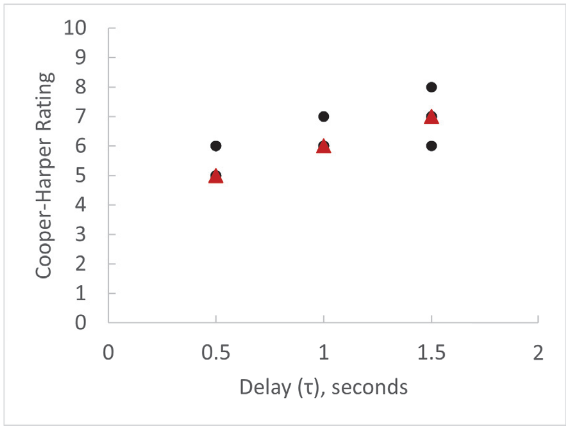
To remove any potential bias, the participants were not told of the amount of delay being applied. Each participant had to perform 6 tracking tasks (a), one for each of the three cases (i)-(iii) with and without the predictive display algorithm. They then performed 12 target acquisition tasks (b), two for each of the three cases (i)-(iii) with and without the predictive display algorithm. In the target acquisition task (b), each of the cases was run twice: once with the cursor initially located at bottom left corner of the screen and once with it located at the bottom right corner. The data obtained from each of the participants were the handling qualities ratings and the time required to complete each task. The experimental test plan also allowed for the assessment of motion sickness symptoms of the participants due to wearing an FPV headset. However, none of the participants reported feeling any motion-sickness like effects.

## B. Study Population

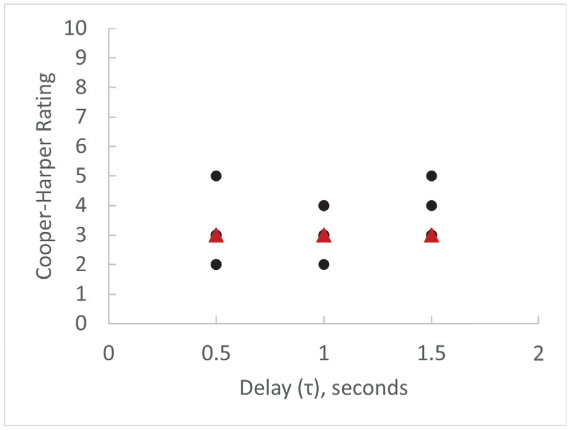
Nine participants took part in the study. HQR data from participants A, D and E were used to establish a baseline rating for the system and to set the desired and the adequate bounds. Participant A was a professional air force test pilot with prior experience with the HQR scale and identified himself as a moderate-high gain participant. Participant D was found to be of low-gain whereas participant E was of high-gain. The baseline ratings for several different example cases were chosen by participant A. The spatial and temporal bounds were chosen after analyzing the performances of participants A, D and E.



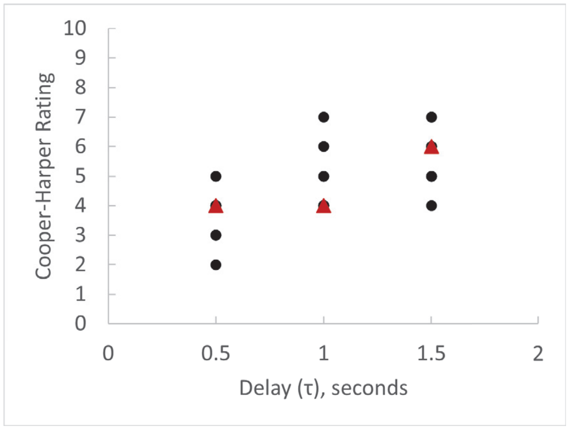
**(a)**



**(b)**



**(c)**



**(d)**

**Fig. 12 Cooper-Harper ratings for cases (i)-(iii): (a) Tracking with predictive display (PD); (b) Tracking without PD; (c) Target acquisition with PD; (d) Target acquisition without PD. Black dots indicate individual operator ratings; red triangles represent the mode of all ratings. Note that a lower number corresponds to a more favorable rating; see Figure 5.**

Participants B, C, F, G, H and I were calibrated according to the baseline ratings. Each participant was given sufficient time at first to get familiarized with the system and the controller. Then they were asked to perform some tracking and targeting tasks for several different example cases. The baseline rating for each of those cases was then revealed to the participants so that they had an idea of what a system with a good, bad or an average handling quality rating felt like. After the participants are shown five cases of good, bad, and average handing quality ratings, they were

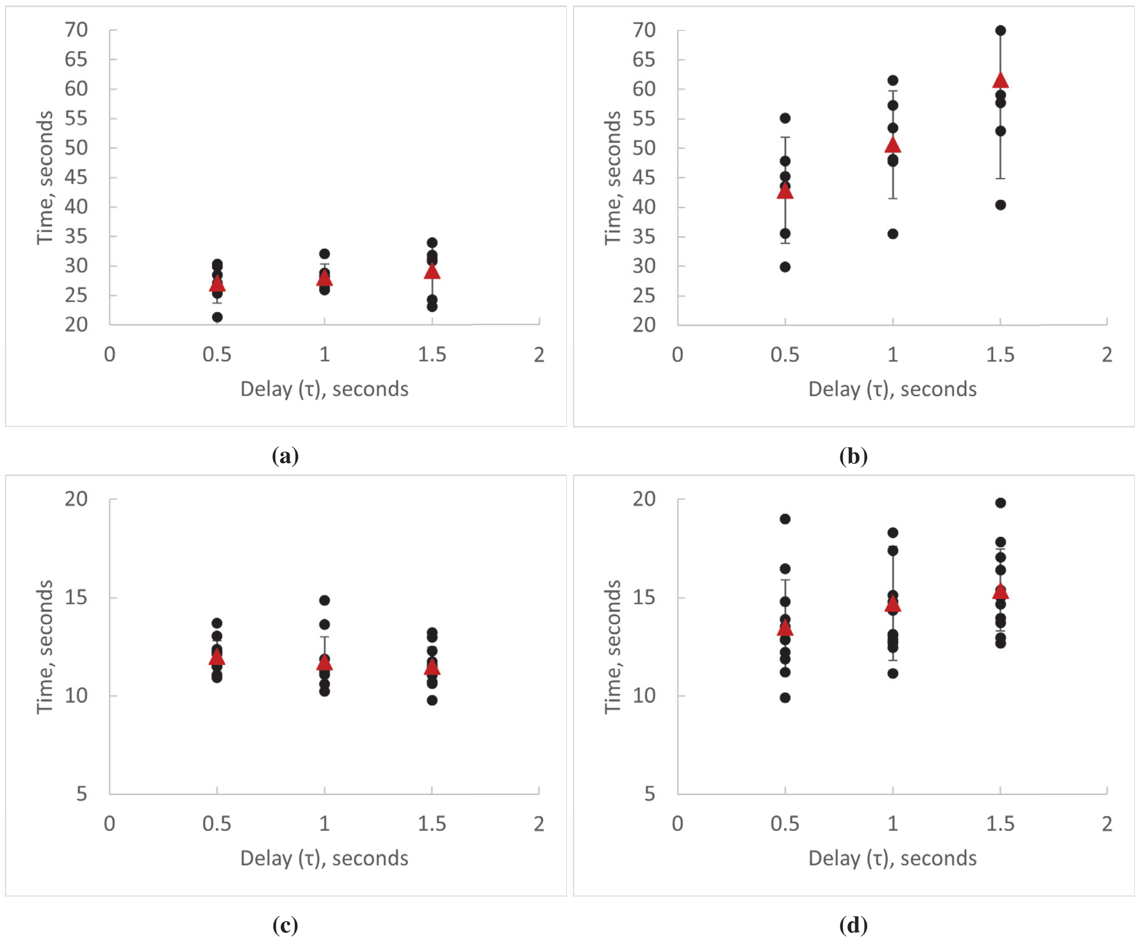


**Fig. 13 Time to complete the cases (i)-(iii): (a) Tracking with predictive display (PD); (b) Tracking without PD; (c) Target acquisition with PD; (d) Target acquisition without PD. Black dots indicate results for individual operators; red triangles represent the mean of all ratings.**

tested on three different example cases where they were asked to rate the system after completing a task. Their ratings were then matched with the baseline rating obtained. If the participant's ratings fell within one unit of the baseline rating, they were considered proficient in rating the system. Otherwise, three more example cases were presented and the participants were tested again. These ten calibration cases proved to be sufficient to help each participant apply the HQR scale in a manner consistent with the other participants.

## VIII. Results

Figures 12a - 12d contain the the Cooper-Harper rating of the system with and without the predictive algorithm. In all cases, the same image in Fig. 11 is used. The red triangle markers are the mode of the ratings obtained from the six participants. Since a non-integer Cooper-Harper rating has no meaning, the mode of the ratings, that is, the rating that was selected the most often, was taken instead of the mean. It is seen that in both the tracking and targeting tasks the prediction algorithm yields better ratings. For the tracking task, the ratings are better with the predictive display in operation (Fig. 12a) than without (12b). The performance deterioration with increasing delay is also lower with active prediction. Overall at lower delays, the prediction achieves a rating of 3, which means minimal compensation is required to achieve the desired performance. At higher delays (1 s and 1.5 s) the performance deteriorates a little as the rating increases to 4, which means a moderate amount of compensation is required to obtain desired results. Without prediction, on the other hand, the rating is 5 at 0.5 s delay dropping to 7 at a 1.5 s delay. A rating of 5 means the operator cannot perform as desired even with considerable compensation. At the highest tested delay, even adequate

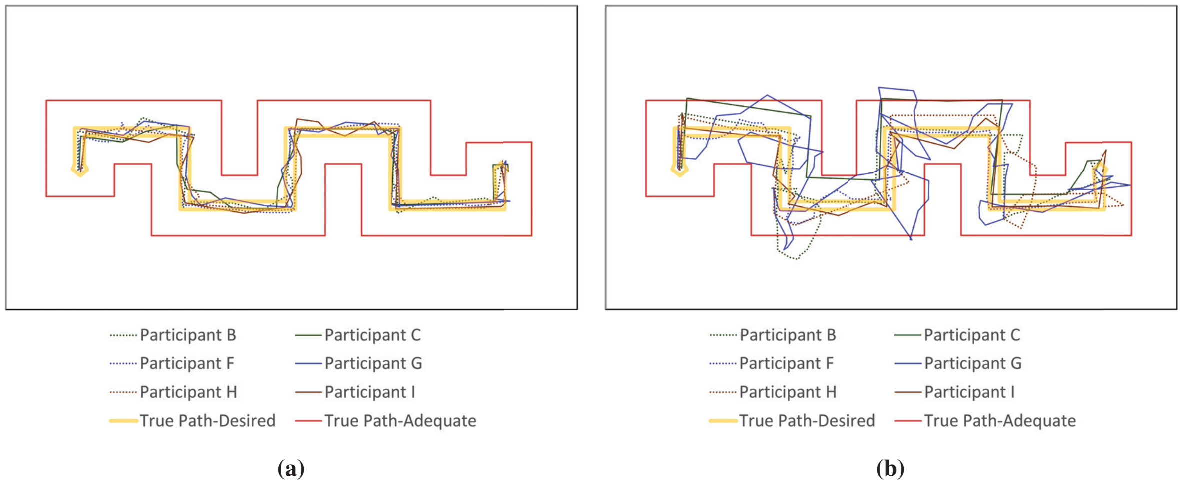


**Fig. 14 Participant's path at $1.5$ s delay with a) predictive algorithm and b) no predictive algorithm**

performance is not obtainable.

Similar conclusions can be drawn from Figs. 12c and 12d. For the targeting task, assessments of the system's performance were not greatly affected by the delay when the prediction was enabled, as seen by the consistent rating of 3. Without prediction, however, operators failed attain the desired performance at the highest tested delay. Even at low delay, moderate compensation was required for desired performance.

The time data collected for each of the tasks are also sorted according to the three different delay cases and are presented in Figs. 13a-13d. Comparing Figs. 13a and 13c with Figs. 13b and 13d, it can be seen that the average time required to complete a task with the prediction algorithm in operation is much lower and the difference between the mean execution times across different delay cases is smaller than without prediction. The wide variation in task completion time for the case where there is no prediction merits special attention. Different participants reacted differently when they were presented with delay in the system. Some participants sought to achieve adequate results and tried to finish the task as quickly as possible without pausing to correct the path tracking errors. Others employed the "move and wait" strategy mentioned earlier to finish the tracking or targeting tasks accurately. This took considerably more time as the participants had to pause a lot to see what corrective action would be needed. This can be seen in Fig. 14b where participants B, C, and G tried to finish the task in a short time and produced paths which had large deviations from the desired path. Participants F, H and I used "move and wait" strategy and produced paths with minimal deviation. All the participants were able to track the desired path for the same amount of delay much more accurately with the prediction enabled. Figure 14 is a good indication that the prediction algorithm helps the participants track more quickly and accurately. Moreover, the participants do not need to know any special strategy or sacrifice accuracy to counter the effects of the delay. The same conclusion can be obtained from the participant paths involving $0.5$ s and $1$ s of delays. Those figures are included in the appendix.

After obtaining all the participants' path data, like the ones shown in Fig. 14, the error in the tracking task is computed by summing the area under the curve that fell outside of the region containing the desired path. The total error thus has the unit of square pixels. For each participant, all the errors are normalized by dividing with the error value obtained at $0.5$ s of delay with the predictive display enabled. This data is shown in Table 3. As different participants followed different approaches while facing a delay, it is not logical to compare the error data of one participant with another. Instead, the normalized error data for each of the participants with predictive algorithm is compared with the normalized error data of the same participant without the predictive algorithm. It seems that participants who followed the move and wait strategy had less error in their path tracking than the ones who chose to sacrifice accuracy for speed. Except for participants F and I, all the other participants show an increased error when the delay time is increased and prediction is not enabled. For the case of $1.5$ s delay with no prediction enabled, the error data for participant G are not calculated as they lost control while performing the task. The HQR scale provides a subjective assessment of the system under different delays whereas the task completion time and the error in the tracking task provide objective assessment of the system. In this study it has been found that both the objective and the subjective assessments indicate that the predictive display helps to mitigate the negative effects of delay.

**Table 3 Error in the tracking task (normalized)**

| | .5 s delay with PD | 1 s delay with PD | 1.5 s delay with PD | .5 s delay no PD | 1 s delay no PD | 1.5 s delay no PD |
|---|---|---|---|---|---|---|
| Participant B | 1 | 0.27 | 0.40 | 3.64 | 6.65 | 16.25 |
| Participant C | 1 | 0.33 | 1.35 | 6.89 | 5.81 | 13.76 |
| Participant F | 1 | 1.81 | 1.71 | 4.09 | 5.56 | 4.13 |
| Participant G | 1 | 0.40 | 0.44 | 1.59 | 4.04 | - |
| Participant H | 1 | 0.63 | 0.06 | 1.35 | 1.33 | 2.23 |
| Participant I | 1 | 0.21 | 0.95 | 1.73 | 0.53 | 1.08 |

For the experiments described here, the tasks presented to participants involved only PTZ imagery. While the images included stitched views from the omnidirectional camera, the focal area for tasks did not extend into these regions. The image stitching algorithm does not currently perform as well as expected because the undistortion process in `OpenCV` that corrects for the highly distorted fisheye lens effect crops the edges of the image. As a result, if the yaw and pitch angles are beyond $\pm 25^o$ the `warpPerspective()` functions warps the points incorrectly. The error in the warping is higher the further away a pixel point is from the optical center. Figure 15 shows an example of poor stitching. On closer inspection, it is seen that the edges, where the images from the two different cameras are stitched together, are not properly aligned. In fact, the image coming from the omnidirectional camera does not seem to be warped and scaled correctly which is the case of having a faulty homography matrix due to the fisheye lens distortion. In future studies, different fisheye camera models will be used rather than the generic `OpenCV` fisheye camera model in the hope that they will provide better image stitching results.

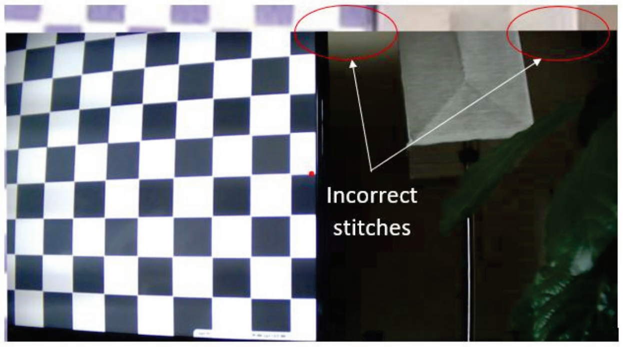


**Fig. 15 Incorrectly stitched image.**

## IX. Conclusion

In this paper a heterogeneous stereo vision system is considered which can assist bridge inspectors in performing their duties via teleoperation in real time. Like in all teleoperation cases, the delays present in the system can cause usability problems that would discourage inspectors from adopting this technology. Making use of a heterogeneous vision system, a solution has been presented where the delayed imagery coming from the omnidirectional camera is stitched to the delayed imagery of the PTZ camera. The stitched image predicts the actual scenario seen by the PTZ camera ahead of the delayed time. The usability studies showed that the prediction algorithm improves the users' experience of the system by decreasing the workload in performing a task, reducing the time of completing a task and increasing the accuracy of performing a task. Ongoing work involves using better omnidirectional camera models than the ones provided by `OpenCV` to perform better image stitching. Also, instead of letting the PTZ camera move with respect to the fixed omnidirectional camera both the cameras can be mounted on a larger gimbal. This would make the homography present in between the two camera frames constant, which may also improve stitching performance. Finally, the work will be expanded from a ground based setup to an airborne setup where the participants would perform the same tasks as the ones described in this paper to understand the effectiveness of the algorithm in an actual bridge inspection scenario.

## Acknowledgements

The authors gratefully acknowledge the participation of human operators who participated in these experiments, as well as the helpful assistance of Changkoo Kang, Haseeb Chaudhry, Naveed Nekoo and Kyle Rearwin. The work in this paper was supported by the National Science Foundation (NSF) under Grant No. IIS-1840044.

## References

[1] de Vries, S. C., "UAVs and Control Delays," *TNO Defence, Security and Safety*, 2005.

[2] Kang, C., Chaudhry, H., Woolsey, C., and Kochersberger, K., "Development of a Peripheral-Central Vision System for Small UAS Tracking," *AIAA SciTech 2019 Forum*, 2019. https://doi.org/10.2514/6.2019-2074.

[3] Rathnayaka, P., Baek, S.-H., and Park, S.-Y., "An Efficient Calibration Method for a Stereo Camera System with Heterogeneous Lenses Using an Embedded Checkerboard Pattern," *Journal of Sensors*, Vol. 2017, 2017, pp. 1–12. https://doi.org/10.1155/2017/6742615.

[4] Zhang, Z., "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 11, 2000, pp. 1330–1334. https://doi.org/10.1109/34.888718.

[5] Chen, J. Y. C., Haas, E. C., and Barnes, M. J., "Human Performance Issues and User Interface Design for Teleoperated Robots," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, Vol. 37, No. 6, 2007, p. 1231–1245. https://doi.org/10.1109/tsmcc.2007.905819.

[6] Sheridan, T. B., and Ferrell, W. R., "Remote Manipulative Control with Transmission Delay," *IEEE Transactions on Human Factors in Electronics*, Vol. HFE-4, No. 1, 1963, pp. 25–29.

[7] Sheridan, T. B., "Space Teleoperation Through Time Delay: Review and Prognosis," *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 5, 1993, pp. 592–606. https://doi.org/10.1109/70.258052.

[8] Brudnak, M. J., "Predictive Displays for High Latency Teleoperation," *NDIA Ground Vehicle Systems Engineering Technology Symposium*, 2016.

[9] Van Dam, J., Krasner, A., and Gabbard, J. L., "Drone-based Augmented Reality Platform for Bridge Inspection: Effect of AR Cue Design on Visual Search Tasks," *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, IEEE, 2020, pp. 201–204.

[10] Van Dam, J., Krasner, A., and Gabbard, J. L., "Augmented Reality for Infrastructure Inspection with Semi-autonomous Aerial Systems: An Examination of User Performance, Workload, and System Trust," *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, IEEE, 2020, pp. 743–744.

[11] Bianchi, E. L., "COCO-Bridge: Common Objects in Context Dataset and Benchmark for Structural Detail Detection of Bridges," Master's thesis, Virginia Tech, 2019.

[12] Malis, E., and Vargas, M., "Deeper understanding of the homography decomposition for vision-based control," Tech. Rep. INRIA-00174036, 2007.

[13] Kannala, J., and Brandt, S., "A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, 2006, pp. 1335–40. https://doi.org/10.1109/TPAMI.2006.153.

[14] Cooper, G. E., and Harper, R., "The Use of Pilot Rating in the Evaluation of Aircraft Handling Qualities," Tech. rep., NASA, 1969.

[15] Jennings, S., Craig, G., Reid, L., and Kruk, R., "The effect of visual system time delay on helicopter control," Vol. 3, No. 1992, 2000, pp. 69–72. https://doi.org/10.1177/154193120004401318.

[16] Mitchell, D. G., and Klyde, D. H., "This is pilot gain," *AIAA Scitech 2019 Forum*, 2019, pp. 1–17. https://doi.org/10.2514/6.2019-0562.

## Appendix

The paths taken by the participants for cases (i) and (ii) for both the predictive and the non-predictive setups are given below.

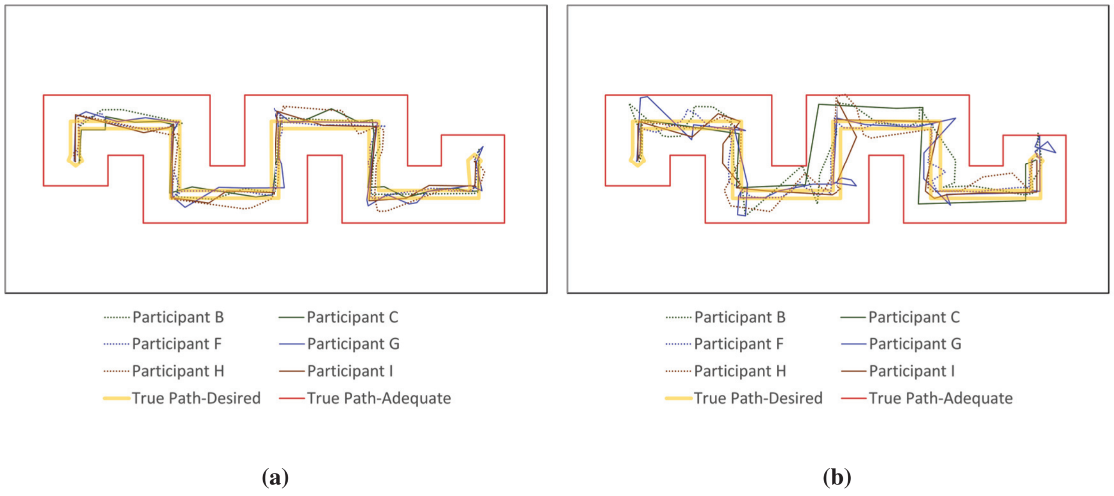


**Fig. 16 Participant's path at $0.5$ s delay with a) predictive algorithm and b) no predictive algorithm**

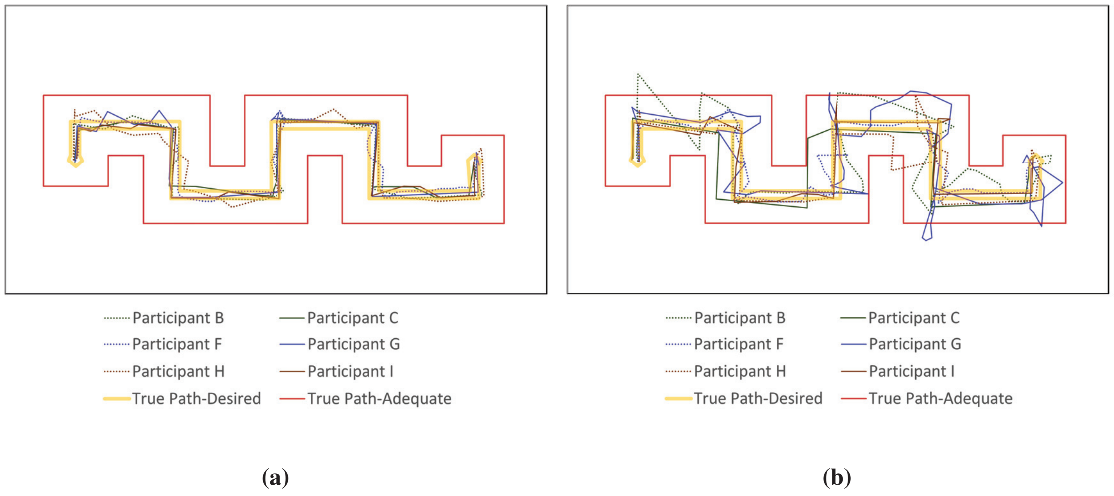


**Fig. 17 Participant's path at $1$ s delay with a) predictive algorithm and b) no predictive algorithm**