Comparing Position- and Image-Based Visual Servoing for Robotic Assembly of Large Structures

Yuan-Chih Peng*, Devavrat Jivani*, Richard J. Radke*, John Wen*

***Abstract*—This paper considers image-guided assembly for large composite panels. By using fiducial markers on the panels and robot gripper mounted cameras, we are able to use an industrial robot to align the panels to sub-millimeter accuracy. We considered two commonly used visual servoing schemes: position-based visual servoing (PBVS) and image-based visual servoing (IBVS). It has been noted that IBVS possesses superior robustness with respect to the camera calibration accuracy. However, we have found that in our case, PBVS is both faster and slightly more accurate than IBVS. This result is due to the fact that the visual servoing target in the image plane is derived from a reference target, which depends on the accuracy of the camera model. This additional dependency essentially nullifies the robustness advantage of IBVS. We also implemented a simple scheme to combine inputs from multiple cameras to improve the visual servoing accuracy. Both simulation and experimental results are included to show the effectiveness of visual servoing in an industrial setting.**

***Index Terms*—Visual servoing, robot assembly, industrial robot, computer vision for manufacturing, assembly automation**

I. INTRODUCTION

Visual servoing, i.e., controlling a robot's motion using computer vision in the servo loop, has been studied since the early 1980's [1], allowing robots to operate accurately in versatile work environments. The two major visual servo approaches are position-based visual servoing (PBVS) and image-based visual servoing (IBVS). In the case of PBVS, the camera is essentially used as a 3D sensor. It first transforms the image measurement to an error in the 3D Euclidean space and adjusts the robot motion to reduce this error [2]. In IBVS, the camera is used as a 2D sensor and the robot motion is determined to directly reduce the errors between current and desired features in the 2D image plane [3].

Several researchers have noted their preference for IBVS over PBVS due to its simplicity and robustness to calibration and image noise [4]–[6]. In this paper, we report a practical industrial scenario in which we surprisingly found PBVS shows a comparable level of accuracy and outperforms IBVS in terms of speed. In our case, a camera is used to guide an industrial robot to assemble a large structure in an eye-in-hand configuration. The physical testbed in our lab is shown in Figure 1. The robot is programmed to pick up and place large structures in locations relative to fiducial markers fixed in the testbed.

A key aspect in our implementation is that the robot does not pick up the panel at exactly the same location each time. This results in additional errors, particularly in terms of the orientation of the transform between the gripper and panel frames. Furthermore, the image target is derived from a fixed target on the testbed, which uses this transform. The coupling of these two factors essentially nullifies the robustness advantage of IBVS.

The key contributions of this paper are:

1) We perform error analysis for both PBVS and IBVS. Our analysis shows that if the target image is known *a priori* then IBVS is more robust with respect to the camera calibration parameters. If the target image is inferred from a target pose, as in our case, then PBVS and IBVS have comparable robustness properties.
2) We demonstrate the comparable placement accuracy of PBVS and IBVS in both simulation and physical experiments. PBVS has additional advantages in terms of the computational speed and convergence rate, as PBVS avoids the image Jacobian computation and the tuning of the corresponding update rate in IBVS.
3) We describe a simple approach of combining multiple cameras mounted at different locations on the gripper, and demonstrate its efficacy in our testbed.

The remainder of the paper is structured as follows. Sec. II states the problem formulation. We introduce the algorithms used for computer vision in calibration, PBVS, and IBVS, and analyze their robustness in Sec. III. Sec. IV describes the overall system setup. Results from simulation and the experimental testbed are reported in Sec. V and Sec. VI.

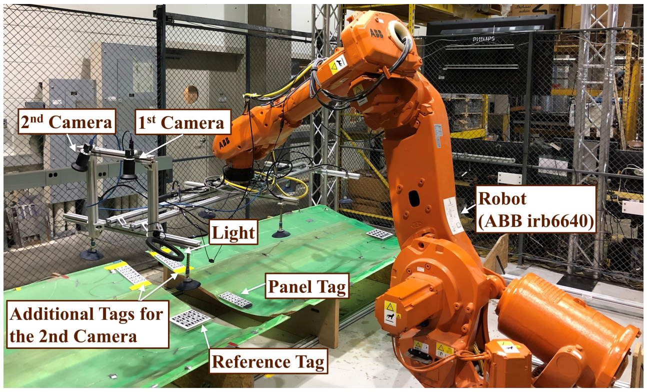

Fig. 1: The testbed setup. The main components of the placement task include an ABB IRB6640 6 DOF robot, two panels with various reference tags, and a robot gripper mounted with two cameras and an illuminating light.

*: Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY

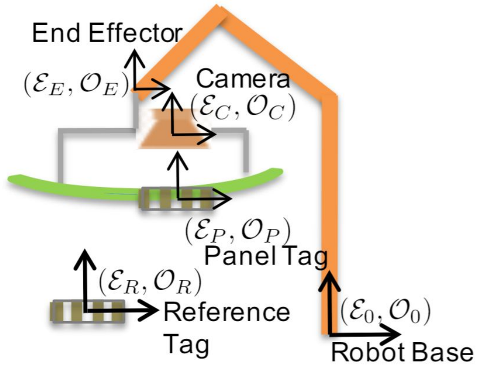
End Effector
$(\mathcal{E}_E, \mathcal{O}_E)$
Camera
$(\mathcal{E}_C, \mathcal{O}_C)$
$(\mathcal{E}_P, \mathcal{O}_P)$
Panel Tag
$(\mathcal{E}_R, \mathcal{O}_R)$
Reference Tag
$(\mathcal{E}_0, \mathcal{O}_0)$
Robot Base

Fig. 2: The Euclidean frames of the robot base, robot end effector, camera, panel tag, and reference tag.

II. PROBLEM STATEMENT

We consider the task of accurate placement of a large composite panel into a wooden nest using an industrial robot. The robot holds a suction gripper upon which are mounted downward-pointed cameras, as shown in Figure 1. Various tags on the panels provide the feature input for visual servoing. In particular, the moving panel has a *panel tag* on one end that must be aligned with a *reference tag* either on the floor of the nest or on the end of the previously placed panel.

A Euclidean frame consists of an orthonormal frame $\mathscr{E}$ and an origin $\mathscr{O}$. Denote the homogeneous transform (rotation and translation) of the end effector, camera, panel, and reference frames with respect to the robot base frame as T_{OE}, T_{OC}, T_{OP}, and T_{OR} as shown in Figure 2. Since the camera is rigidly attached to the robot, $T_{EC} = T_{OE}^T T_{OC}$ is a fixed transform. The panel is rigidly grasped by the robot end effector, so $T_{EP} = T_{OE}^T T_{OP}$ is also fixed. The setup is meant to mimic a large structure assembly, so the pick up and placement operation will be repeated multiple times. Each time the robot grasps the panel, the location is different. Thus, T_{EP} varies for each run (while T_{EC} is always fixed). The panel pose relative to the reference frame is $T_{PR} = T_{OP}^T T_{OR} = T_{EP}^T T_{OE}^T T_{OR}$.

The camera projects the reference tags to the image frame. Let $Y = c(T_{CR}) = c(T_{CE} T_{EP} T_{PR})$ be the vector of image space features (e.g., stacked image coordinates of corner points of the reference tags). If the set of features is sufficiently rich, we can uniquely determine $T_{CR} = c^{-1}(Y)$ (in practice, this will be solved using camera 3D pose estimation, e.g., the perspective-n-point algorithm [7]). This formulation includes the case of multiple cameras. In our specific setup, there are two cameras mounted on the gripper: the first camera at the midpoint aiming directly downward at the panel tag, and the second camera at the edge of the gripper aiming at an edge tag.

During the training stage, the panels are carefully aligned by hand and the relative pose T_{PR}^* between the panel and reference tags is recorded as the ideal pose. Note that the recorded manual alignment is the best that visual servoing can achieve, and the accuracy may improved by using high accuracy gauges during the manual alignment process. The corresponding image output is $Y^* = c(T_{CR}^*) = c(T_{CE} T_{EP} T_{PR}^*)$. Note that T_{CE}, T_{EP}, and T_{OR} are fixed, so the robot motion objective is to adjust the end effector frame T_{OE} to drive T_{PR} to T_{PR}^*. Visual servoing uses the measurement of Y to determine the update of T_{OE}. For PBVS, the measured Y is first converted to the desired relative pose T_{PR}^*. The robot motion updates the end effector pose T_{OE}^* to reduce the error in T_{PR}. For IBVS, the robot end effector T_{OE} is updated to directly reduce the difference between Y and Y^*. Both schemes employ a dynamic look-and-move approach.

The goal of this paper is to present and compare simulated and experimental results of PBVS and IBVS applied to the panel assembly task. Camera calibration establishes the reference tag location T_{OR} and the camera location in the end effector frame T_{EC}. As explained earlier, T_{EP} changes in each repeated run (as the grasp location may change slightly). It may be estimated from the panel tag, but the accuracy is not as good as using the full calibration. We will evaluate the impact of this imprecision on the visual servoing accuracy. As our testbed has two cameras, we will also discuss a simple method of combining the camera measurements.

III. METHODOLOGY

A. Camera Calibration

Camera calibration finds the intrinsic parameters (focal lengths, optical center location, skew, distortion) and extrinsic parameters (pose of the camera relative to the end effector, T_{EC}) of the camera. Many tools exist for intrinsic camera parameter calibration; we use the MATLAB Camera Calibrator [8]. For the extrinsic parameters, we use the hand-eye calibration technique [9], [10]. A stationary calibration target is placed in front of the camera, and the robot moves together with the camera at different poses while the target images and joint angles are recorded. Once the camera is calibrated, it may be used to estimate the reference tag location T_{OR} from camera images of the reference tag. To reduce the estimation error, multiple images from different camera poses are used. Similarly, we can estimate the panel location with respect to the camera T_{CP} once the panel is grasped by the robot. However, we no longer have the luxury of using images from multiple camera perspectives since the panel is in a fixed relative pose with respect to the camera. Hence, T_{CP} is less accurate than T_{OR}.

B. Visual Servoing

As stated in Section II, the goal is to move the robot end effector T_{OE} based on the image feature vector $Y = c(T_{CR})$ to achieve the desired panel location relative to the reference T_{PR}^*. With camera calibration, we assume T_{EC}, T_{OR}, and T_{CP} are known, though T_{CP} may be imprecise.

To simplify analysis, let Z be a 6×1 parameterization of T_{PR} (orientation and position) and X be a 6×1 parameterization of T_{OE}. Write the image output equation as

$$Y = c(T_{CR}) = c(T_{EC}^T T_{EP} T_{PR}) := h(Z). \qquad (1)$$

Express the mapping $T_{PR} = T_{CP}^T T_{EC}^T T_{OE}^T T_{OR}$ as

$$Z = g(X). \tag{2}$$

Denote the target pose as Z^* corresponding to T_{PR}^*. Given the discussion in Section III-A, g is precisely known, but h is less precise because of T_{EP}. Denote the identified, but imprecise, mapping as h'.

The visual servoing problem is then to use X to drive Z to Z^* based on the measurement Y.

C. PBVS

In PBVS, we first find Z from the output Y using the imprecise h':

$$\hat{Z} = h'^{-1}(Y). \tag{3}$$

For incremental robot motion, the change of the end effector pose is related to the change in Z through

$$\delta Z = \underbrace{(\nabla_X g)}_{:=J_g} \delta X. \tag{4}$$

The robot motion may be chosen as a simple proportional update by solving the following least squares problem:

$$\min_{\delta X} \left\| J_g \delta X - (-K_{PBVS}(\hat{Z} - Z^*)) \right\| \tag{5}$$

where K_{PBVS} is a positive definite proportional feedback gain. Additional linear constraints may be added, e.g., limiting the size of end effector motion δX, and then solved as a quadratic programming problem.

D. IBVS

In IBVS, we directly form the error in the image space. In this case, using the imprecise h', we have

$$\hat{Y}^* = h'(Z^*). \tag{6}$$

Again consider incremental robot motion; the change in the image plane is related to the change in Z through

$$\delta Y = \underbrace{(\nabla_Z h)}_{:=J_h} J_g \delta X. \tag{7}$$

Note that only h' is known, so we can only use $J_{h'} := \nabla_Z h'$ in the solution. The robot motion may be chosen as a simple proportional update by solving a quadratic programming problem (with linear constraints included):

$$\min_{\delta X} \left\| J_{h'} J_g \delta X - (-K_{IBVS}(Y - \hat{Y}^*)) \right\| \tag{8}$$

with K_{IBVS} a positive definite gain.

E. Effect of Imprecise Parameters

Assume the only imprecision is in h. In the PBVS case, with the gain chosen sufficiently small, $\hat{Z}$ will converge to Z^*. This means the steady state error in Z is

$$\tilde{Z}_{PBVS} = Z^* - h^{-1}(h'(Z^*)). \tag{9}$$

In the IBVS case, if $J_{h'}$ still maintains the descent direction, e.g., $J_h J_{h'}^{-1} K_{IBVS}$ is positive definite, then Y will converge to $\hat{Y}^*$, i.e., $Y = h(Z) = \hat{Y}^* = h'(Z^*)$ at the steady state. In this case, the steady state error in Z is

$$\tilde{Z}_{IBVS} = h^{-1}(h'(Z^*)) - Z^*, \tag{10}$$

which is the same as the PBVS case.

In the literature, it has been reported that IBVS possesses superior robustness properties [4]–[6]. Indeed, if Y^* is known, then PBVS would still have the same error, but IBVS would have zero steady state error (since $Y \to Y^*$, and therefore $Z \to Z^*$). This is not the case here, as our target image itself contains error. We shall see that this conclusion is indeed observed in our experiments.

F. Combining Measurements from Multiple Cameras

When there are multiple cameras, with corresponding feature vectors $Y_1, \ldots, Y_N$, we can generalize the IBVS update in (8) as a convex combination of the individual IBVS updates:

$$\min_{\delta X} \sum_{i=1}^{N} \alpha_i \left\| J_{h'} J_g \delta X - (-K_{IBVS}(Y_i - \hat{Y}_i^*)) \right\|^2 \tag{11}$$

where $\alpha_1 + \alpha_2 + \ldots + \alpha_N = 1$ and $\hat{Y}_i^* = h_i'(Z^*)$. Suppose the measurement of the i^{th} camera is corrupted by an additive zero-mean Gaussian noise with variance σ_i^2. The individual term in the objective function should be weighted by $1/\sigma^2$ to normalize the noise. In that case,

$$\alpha_i = \frac{1}{\sigma_i^2 \left(\frac{1}{\sigma_1^2} + \ldots + \frac{1}{\sigma_N^2} \right)}. \tag{12}$$

The optimum δX is then determined by (11). Then the optimum δZ follows from $\delta Z = \nabla g(X) \delta X$.

We have two cameras, so $\alpha_1 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}$, $\alpha_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$.

IV. SYSTEM DESCRIPTION

The experimental testbed is built to emulate a real-world smart factory environment for large structure assembly. The system software implementation uses the open-source Robot Operating System (ROS) [11] and is programmed in Python.

A. ABB IRB 6640/180

To handle the bulky and heavy load of the panel, an industrial robot from ABB, IRB6640/180, is bolted to the ground at the center of the testbed. The robot has 6 degrees of freedom (DOF) and spans the working area with a farthest reach of 2.55m. The robot payload capacity of 180 kg and positioning repeatability of 0.07 mm position satisfies our goal of placing a 35 kg panel with less than 1 mm positioning accuracy on all sides. A 6-suction-cup vacuum gripper is customized to the robot end effector, and is equipped with two cameras and an illumination light as shown in Figure 1.

The motion command from our algorithm is communicated directly to the robot controller through the external command interface for ABB robots, External Guided Motion (EGM), at 4 ms rate. The high fidelity simulation using ABB RobotStudio may be visualized in Gazebo or Rviz for the user to preview the operation.

B. Vision System

Two identical FLIR Vision cameras (Blackfly S Color 5MP 2448×2048 USB3), are mounted at the left end of the robot gripper so that the field of view includes both the edge of the moving panel and the ground/environment. Each camera is instrumented with a Tamron 1.1" 8mm f/1.8 12MP fixed focal lens, and has a field of view of 0.88 m × 0.66 m from a working distance of roughly 0.5 m, giving a corresponding resolution of around 0.36 mm/pixel × 0.32 mm/pixel. We use the Python wrapper to create local camera nodes, which interface and stream video from the cameras with the ROS master at a rate of about 75 fps. A flat and stiff composite checkerboard with 30 mm × 30 mm grids from Calib.io is used for camera calibration. The re-projection error from the calibrated extrinsic parameters with the checkerboard is about 0.13 pixels.

To enable robust detection, localization, and identification of image-space features, we chose the ArUco markers [12] and the open source Augmented Reality (AR) library in OpenCV. The (40 mm)^2 markers are printed at 600 dpi resolution on flat acrylic boards to reduce distortion. The library enables marker IDs, 2D image coordinates, and relative 3D poses to be acquired from a single camera. Each of the panel and reference tags is composed of multiple markers for robustness and accuracy. The panel tag is composed of 24 markers (8×3) and the reference tag has 16 markers (4×4). In practical settings, other features such as the edge and existing patterns on the panel may be used as visual servoing markers.

A dimmable LED light is also mounted on the gripper to make sure that the imaging process has sufficient and consistent lighting. Since the working area is large, we used 10 m active USB3 extension cables (Newnex FireNEXAMAF10M) to ensure that the camera data is reliably transmitted without attenuation.

V. SIMULATION

In this section, we report results from simulations of single-camera PBVS and IBVS that we implemented in ROS Rviz. All the components, including the robot, gripper, tags, and panels, are represented by their true dimensions and locations. A virtual gripper camera is also simulated using its extrinsic and intrinsic calibration data. With the virtual robot and virtual images, we implement the single-camera PBVS and IBVS algorithms presented in Sections III-C and III-D. The simulated camera view is shown in Figure 3, in which the poses of the markers are denoted with red-green-blue axes. The detected corners of the markers are used as the feature vector Y in (1). The ideal reference tag corners, $\hat{Y}^*$ in (6), are generated based on the panel tags through the 3D transformation followed by projection to the image plane as the red circles. The robot is first positioned above the placement nest. The goal is to drive the robot to move the red circles to match the real blue circles on the reference tag as shown in Figure 3. Both the scaling factors (K_{pbvs}, K_{ibvs}) for each step are set to 0.25, which is about a quarter of the maximum desired speed. The process stops when the measured (x,y) error norm is smaller than 1 mm. Note that the error we are concerned about is only in the x-y plane because the panel is already in the assembly nest and the z-axis motion is through force control.

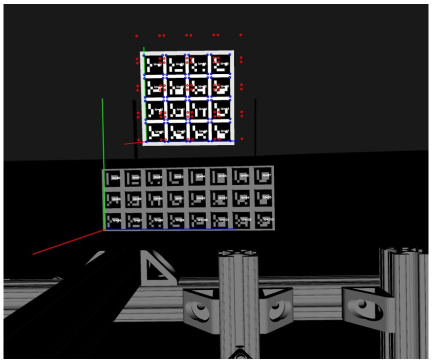

Fig. 3: The images from the virtual gripper camera. In the simulation, the robot is driven by both PBVS and IBVS to move the panel from the initial pose to the desired pose.

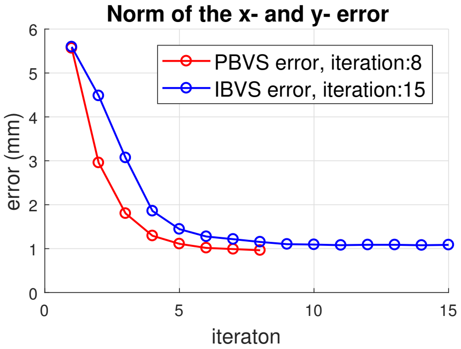

Fig. 4: The simulation results of panel placement using PBVS and IBVS. We see that PBVS converges to the target pose with fewer iterations.

The results from the simulation under known parameters are shown in Figure 4. Both methods converge asymptotically. PBVS converges faster in 8 steps while IBVS converges in 15 steps. This is due to the gain selection in (5) and (8). The idealized simulation environment ignores the image noise and calibration error, and both methods work as expected.

VI. EXPERIMENTS

The real-world experimental setup for panel placement is shown in Figure 1. Under this setup, the panel is first moved to a position about 0.3 m above the assembly nest using direct joint command, keeping both the panel and reference tags in the camera view. Then the robot starts visual servoing to place the panel. During the placement, the compliance force control is also running in the z-axis perpendicular to the ground to maintain a safe and gentle placement.

We use a motion capture system from PhaseSpace to establish the reference for accuracy verification. A total of eight motion capture cameras surround the testbed and detect the positions of 12 LEDs mounted on the panel. The resolution from this motion capture system is approximately 0.1 mm with 220 Hz sampling rate [13]. This is sufficient to verify

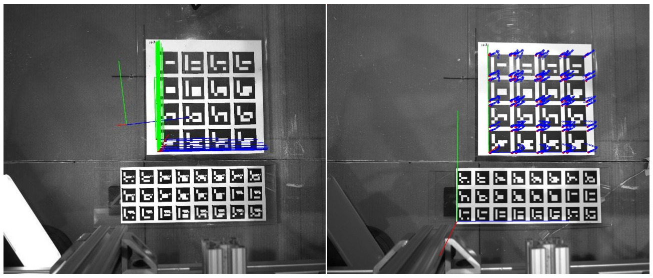

Fig. 5: The image view from the gripper camera during PBVS (left) and IBVS (right). In PBVS, the robot is trying to match the observed reference tag frame to the desired one. In IBVS, the robot is trying to match the observed features (the corners of the reference tag/blue circles in the image), to the desired positions (red circles) that are created by the transformation from the panel tag.

the panel alignment accuracy requirement of 1 mm on all sides.

A. *Visual Servoing Results*

In this section, we report the performance of the PBVS and IBVS algorithms in our testbed. The ideal pose between the panel tag and the reference tag T^*_{PR} is first recorded. The robot then executes PBVS or IBVS until the observed placement error converges to 1 mm or below. We use the same algorithmic parameters as in the simulation.

1) PBVS Placement: We executed the PBVS placement process illustrated in Figure 5 12 times and report the placement error measured from the motion capture system in Figure 6. Both the x and y errors are smaller than 1 mm with an average of $(0.13, 0.39)$ mm and a standard deviation of $(0.27, 0.58)$ mm. The execution times for the 12 trials are shown in Figure 7, which indicates an average placement time of 33.88 sec and a standard deviation of 5.52 sec.

2) IBVS Placement: As discussed in Section III-D, 64 corners from the 4×4 ArUco makers are used as the feature vector, as illustrated in Figure 5. A predefined reference tag location relative to the static panel tag (8×3 markerboard) is set as the desired target for the reference tag (red circles in Figure 5).

As before, we executed 12 trials and report the accuracy and timing results in Figure 6 and Figure 7. The average placement accuracy in the x-y-plane is $(0.3, 0.23)$ mm with a standard deviation of $(0.28, 0.54)$ mm, but we note that one of the placements was outside the 1 mm accuracy requirement in the y-axis, which could be attributed to the calibration error between the gripper camera and the motion capture system. The average operating time is 45.85 sec with a standard deviation of 7.55 sec, which is about 35% more than PBVS. Since the features are acquired from the ArUco markers through a similar process as PBVS, the longer execution time comes from the additional 2D-3D transformations required for IBVS and an additional Jacobian calculation, instead of the direct 3D error as in PBVS. As discussed in Section III-E, here IBVS does not show better robustness than PBVS, compared to the past literature.

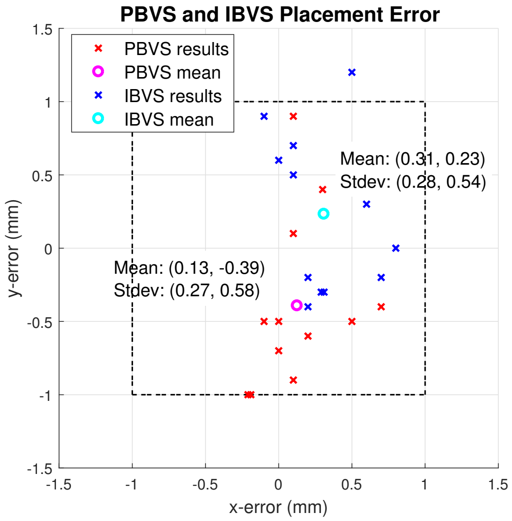

Fig. 6: The average placement accuracy for PBVS in the xy-plane is $(0.13, 0.39)$ with a standard deviation of $(0.27, 0.58)$; the average placement accuracy for IBVS in the xy-plane is $(0.31, 0.23)$ with a standard deviation of $(0.28, 0.54)$.

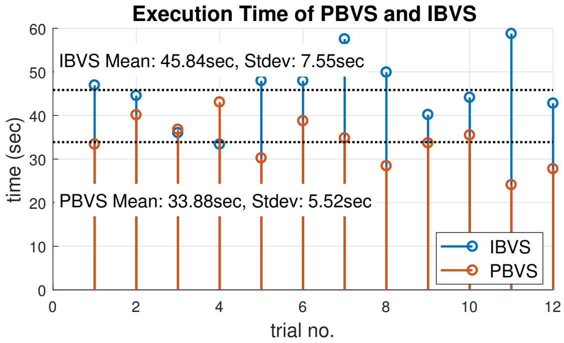

Fig. 7: The placement times of PBVS and IBVS. The average operating time of IBVS is about 35% more than PBVS (45.8 sec vs. 33.8 sec).

B. *Multi-camera IBVS Placement*

We also evaluated IBVS performance using both gripper cameras. We mounted additional tags as shown in Figure 1 as the target for the second camera, which is tilted at about 50 degrees from vertical and is about 0.5 m further along the gripper from the first camera. The distances between the two cameras and their targeting tags are illustrated in Figure 8. We assume that the noise standard deviation from each camera is proportional to its physical distance to the tag. Therefore, we estimate $\frac{\sigma_1}{\sigma_2} = \frac{67}{117}$. From (12), the calculated optimal weights to combine the measurements from the two cameras are $\alpha_1 = 0.75, \alpha_2 = 0.25$.

We ran our algorithm with the weights in Eq. (11) set to $\alpha_1 = \alpha = \{1, 0.75, 0.5\}$ for the first camera and $\alpha_2 = 1 - \alpha = \{0, 0.25, 0.5\}$ for the second camera. When $\alpha = 1$, the process only uses the main camera that is perpendicular to the tag. When $\alpha = 0.5$, the process uses equal weights on both of

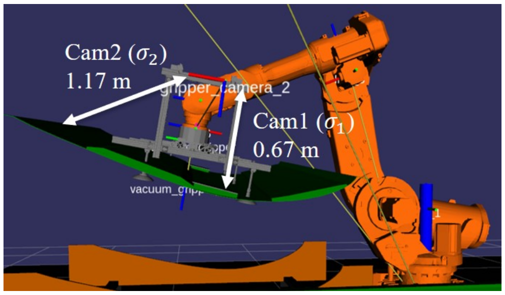

Fig. 8: A high fidelity simulation environment can be visualized in ROS RViz. The distances from the 1st and 2nd cameras to their target tags are 0.67 m and 1.17 m respectively.

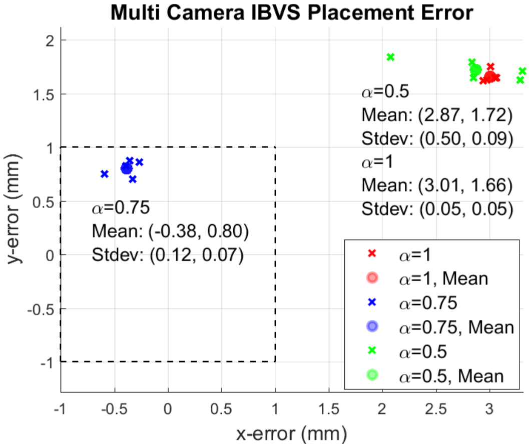

Fig. 9: The multi-camera placement errors with different weights. The placement errors using different weights are indicated with red ($\alpha = 1$), blue ($\alpha = 0.75$), and green ($\alpha = 0.5$) colors. The optimal weights when $\alpha = 0.75$ result in better accuracy based on the motion capture system observation.

the two cameras. The resulting errors for 5 repeated trials measured by the motion capture system with different weights are shown in Figure 9.

The result indicates that when using only the main camera at this setting, the x-y error can be bigger than 1 mm. Using the optimal weighting $\alpha_1 = 0.75$ significantly improves the result. When too much information from the second camera is used with $\alpha_1 = 0.5$, the accuracy decreases again, due to the higher uncertainty from the second camera.

VII. CONCLUSIONS

We compared the performance of PBVS and IBVS in a specific industrial application in both simulation and experimentation. In the proposed application, large structures are assembled to sub-millimeter accuracy based on aligning reference tags detected using visual servo control. Both in simulation and experiment, we found PBVS and IBVS to have comparable convergence performance, with PBVS showing faster execution and convergence rate. Past literature has observed the superior robustness property of IBVS. We showed that this advantage is lost when the target image itself contains modeling error, as in our case. We also combine measurements from multiple cameras based on their uncertainties and experimentally verified the performance.

VIII. ACKNOWLEDGEMENTS

The authors would like to thank Glenn Saunders for the testbed construction, John Wason and William Lawler for the software implementation, Shuyang Chen for the EGM controller implementation, and Jeff Trinkle for inputs and feedback. This work was supported in part by Subaward No. ARM-TEC-17-QS-F-01 from the Advanced Robotics for Manufacturing ("ARM") Institute under Agreement Number W911NF-17-3-0004 sponsored by the Office of the Secretary of Defense. ARM Project Management was provided by Matt Fischer. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of either ARM or the Office of the Secretary of Defense of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

REFERENCES

[1] S. Hutchinson, G. D. Hager, and P. I. Corke, “A tutorial on visual servo control,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Oct 1996.

[2] F. Chaumette and S. Hutchinson, “Visual servo control. i. basic approaches,” *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90, Dec 2006.

[3] P. I. Corke and S. A. Hutchinson, “Real-time vision, tracking and control,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, April 2000, pp. 622–629 vol.1.

[4] G. Palmieri, M. Palpacelli, M. Battistelli, and M. Callegari, “A comparison between position-based and image-based dynamic visual servoings in the control of a translating parallel manipulator,” *Journal of Robotics*, vol. 2012, p. 11, 2012.

[5] L. Deng, “Comparison of image-based and position-based robot visual servoing methods and improvements,” Ph.D. dissertation, University of Waterloo, CAN, 2004, aAINQ91991.

[6] G. L. L. Muñoz and J. M. S. T. da Motta, “Comparative performance analysis of image-based and position-based visual servoing in a 6 dof manupulator,” *ABCM Symposium Series in Mechatronics - Part I - International Congress Section IV - Robotics*, vol. 6, 2014.

[7] L. Quan and Z. Lan, “Linear n-point camera pose determination,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 8, pp. 774–780, 1999.

[8] The MathWorks, Inc. MATLAB Camera Calibrator. [Online]. Available: https://www.mathworks.com/help/vision/ref/cameracalibrator-app.html

[9] R.-h. Liang and J.-f. Mao, “Hand-eye calibration with a new linear decomposition algorithm,” *Journal of Zhejiang University-SCIENCE A*, vol. 9, no. 10, pp. 1363–1368, Oct 2008. [Online]. Available: https://doi.org/10.1631/jzus.A0820318

[10] N. Andreff, R. Horaud, and B. Espiau, “On-line hand-eye calibration,” in *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, Oct 1999, pp. 430–436.

[11] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA Workshop on Open Source Software*, 2009.

[12] S. Garrido. Aruco marker detection (aruco module). [Online]. Available: https://docs.opencv.org/3.4/d9/d6d/tutorial_table_of_content_aruco.html

[13] PhaseSpace, Inc., “PhaseSpace Impulse X2E: Data Sheet,” PhaseSpace, Inc., Tech. Rep., 2017.