Component pose reconstruction using a single robotic total station for panelized building envelopes

Mengjia Tang[1], Nolan W. Hayes[1], Bryan P. Maldonado[1], and Diana Hun[1]

[1]Buildings and Transportation Science Division, Oak Ridge National Laboratory, United States of America *

tangm@ornl.gov, hayesnw@ornl.gov, maldonadopbp@ornl.gov, hunde@ornl.gov

Abstract -

The construction industry can benefit greatly from increased automation in construction processes in terms of precision, time, and labor costs. This is especially true for prefabricated construction for either new buildings or envelope retrofits. Here, prefabricated components are manufactured offsite according to design specifications and installed onsite. Accurate information on the component's position and orientation (pose) is needed to achieve this. A tool named "Real-Time Evaluator" (RTE), designed to autonomously track prefabricated components as they are being installed and provide real-time installation instructions, is currently under development using a single robotic total station. This is challenging since at least three points are needed to determine the pose of an object in space. To achieve this goal with a single robotic total station, two key algorithms were developed: (1) the "resection" algorithm aligns the digital twin with the physical twin regardless of the location of the total station, and (2) the "transformation" algorithm gives instructions of translations and rotations to installers to achieve the desired installation pose. The algorithms were evaluated experimentally on a lab-scale demonstration. Results show that the resection algorithm achieved an average error of < 3.1 mm, while the transformation algorithm predicted the rotation angle along a single axis with an error of < 0.5°.

Keywords -

Prefabricated construction; Automation; Resection; Rotation; Accuracy

1 Introduction

Building construction requires a high degree of accuracy between as-designed and as-built structures, which is challenging given the long project duration, heavy reliance on workers' skills and experience, and complicated onsite environments (e.g., crane operators not being able to see the load all the time). For prefabricated construction, not meeting the required tolerances during the installation of prefabricated components often requires expensive corrections. Therefore, a tool is needed to actively check the quality of construction during the installation of prefabricated components for real-time installation instructions.

Rapid advances in surveying technologies present a great potential for automating construction processes. Robotic total stations can autonomously turn their telescopes and aim at targets by following commands from software, enabling fast and precise measurement and tracking with little user input. Three-dimensional (3D) laser scanners can produce point clouds to provide reality capture, including as-built information during or after construction [1]. While commercially available, these instruments need a streamlined procedure to interact with (1) data in the digital space, (2) the building structure and prefabricated components in the physical space, and (3) the users or construction workers.

The Real-Time Evaluator (RTE) is a tool that uses a single robotic total station to gather the data needed to compare the actual position of a prefabricated component against the as-designed position in a digital twin during installation in order to provide real-time feedback to installers [2]. Two major tasks were identified during the development of the RTE. The first task is to align the 3D model of a building (i.e., digital twin) with the physical twin. When the total station is set up to obtain an initial scan of the building, track the installation process of prefabricated components, or obtain an as-built scan of the building, it will most likely have different positions and orientations in each session. Moreover, the digital twin may be obtained from a 3D laser scanner, building information modeling (BIM) software, or computer-aided design (CAD) software foreign to the total station [3]. Thus, discrepancies can exist between the coordinate system of the digital twin and the coordinate system of the total station in its current setup, which prevents the RTE from giving correct real-time installation instructions. A resection function is available in software such as Trimble® Access™ and Leica® Captivate™ to determine the coordinates of the total station based on the Helmert trans-

formation [4]. However, the available third-party software does not use the digital twin as input to adjust the coordinate system of the total station accordingly. Therefore, it is necessary to develop a resection algorithm that uses the digital twin in an arbitrary coordinate system as input to achieve accurate RTE installation commands.

The second task involves generating optimal installation instructions to move a prefabricated component from its current pose to the intended (goal) pose. While prior research has focused on pose estimation of the crane and load [5] or deviations of the current pose from the as-designed pose [6], less attention has been directed to reconstructing a component pose by tracking targets attached to it. The installation time of prefabricated components could be notably reduced by explicitly specifying the steps to move them directly to the goal pose, eliminating the need for trial-and-error placement. Such steps need to consider six degrees of freedom in the movement of a rigid body, i.e., the translations along the three Cartesian coordinate axes and the rotations around such axes. The trajectory is not unique and needs to be tested and optimized based on various factors including the initial and final poses of the component, the center of rotation, rotation axes, and the connectors on the building structure.

While the conversion between coordinate systems and the transformation of rigid bodies are not new mathematical problems, they are new tasks for automating the prefabricated construction because the procedures, requirements, and available tools need to be considered as constraints. To address the above two tasks, this paper develops a resection algorithm to align the digital twin with the physical building within measurement error. This paper also develops a transformation algorithm that applies a rigid-body transformation to a component in order to achieve a goal pose with a series of translations and rotations around three axes. Accuracy of the algorithms are obtained from lab-scale experiments.

2 Methodology

This section first outlines the algorithms for resection and rigid-body transformation. Subsequently, it describes the lab-scale experimental setup used to demonstrate and evaluate these two algorithms.

The two algorithms, in essence, solve the same problem. The resection algorithm computes the rotation and translation of points from one coordinate system to another. The transformation algorithm computes the rotation and translation of points when they move in the same coordinate system. In both algorithms, the method used to calculate the rotation matrix between two sets of at least three points is the Kabsch or the Kabsch-Umeyama algorithm [7]. It uses the singular value decomposition (SVD) to find the rotation matrix that best aligns the two paired sets of points.

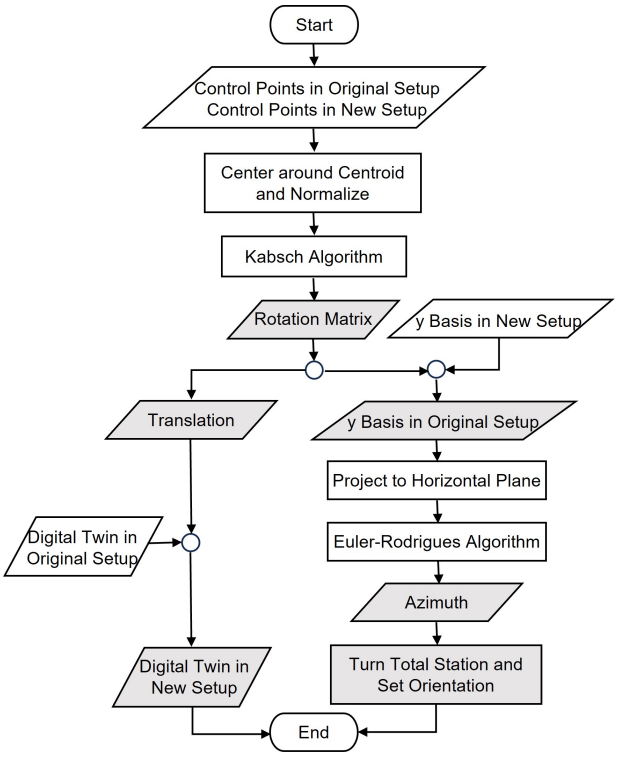

Figure 1. Workflow chart for Algorithm 1: resection.

It was implemented through the "`align_vectors`" function in the SciPy library in Python [8]. When the number of points is two, the Euler–Rodrigues formula was used to align the vector formed by the two points [9] and was fulfilled in this paper with a user-defined function.

2.1 Resection

Let CP_{old} be the set of three or more control points measured in the original (old) coordinate system. Let CP_{new} be the corresponding set of control points measured in the new coordinate system. Finally, let DT_{old} be the set of all the points in the digital twin. The algorithm that adjusts DT_{old} and total station orientation to align the digital twin with the physical twin using CP_{old} and CP_{new} is described in Algorithm 1 and shown in Figure 1. It has the following four steps.

First, the translation vector $\boldsymbol{t}$ and rotation matrix R are calculated such that, for any point in the new coordinate system P_{new}, the corresponding point in the original coordinate system is calculated as $P_{\text{old}} = \boldsymbol{t} + RP_{\text{new}}$. This determination is based on the measurements of control points by the total station at both the original and new setups. The measured coordinates of the control points are re-centered around their centroid in the original ($\boldsymbol{G}_{\text{old}}$) and new ($\boldsymbol{G}_{\text{new}}$) coordinate system, respectively. The re-centered control

Algorithm 1: `resection`: align digital twin with physical twin

Data:
CP_{old}, CP_{new}, DT_{old}

Result:
$\boldsymbol{t}$, Az, DT_{adjusted}

// Calculate rotation matrix and translation vector
$\boldsymbol{G}_{\text{old}} \leftarrow$ `centroid`(CP_{old})
$\boldsymbol{G}_{\text{new}} \leftarrow$ `centroid`(CP_{new})
$cp_{\text{old}} \leftarrow$ `normalize`$(CP_{\text{old}} - \boldsymbol{G}_{\text{old}})$
$cp_{\text{new}} \leftarrow$ `normalize`$(CP_{\text{new}} - \boldsymbol{G}_{\text{new}})$
$R \leftarrow$ `Kabsch`(align cp_{new} to cp_{old})
$\boldsymbol{t} \leftarrow \boldsymbol{G}_{\text{old}} - R\boldsymbol{G}_{\text{new}}$

// Calculate azimuth
$\boldsymbol{j}_{\text{old}} \leftarrow R\boldsymbol{j}_{\text{new}}$
$\text{proj}_{xy}(\boldsymbol{j}_{\text{old}}) \leftarrow$ `project_xy`$(\boldsymbol{j}_{\text{old}})$
$R_{xy} \leftarrow$ `Euler-Rodrigues`(align $\boldsymbol{j}_{\text{new}}$ to $\text{proj}_{xy}(\boldsymbol{j}_{\text{old}})$)
$Az \leftarrow z$ component of `factor`(R_{xy}) in degrees
if $Az < 0$ **then**
  $Az \leftarrow 360 + Az$
end

// Total station commands
`total_station.turn_to_angle`$(Az \times \frac{\pi}{180})$
`total_station.set_orientation`()

// Adjust digital twin
$DT_{\text{adjusted}} = DT_{\text{old}} - \boldsymbol{t}$

return DT_{adjusted}

points are normalized to unit vectors (cp_{old} and cp_{new}). Such vectors are then used as inputs to the Kabsch algorithm to obtain the rotation matrix R. The translation vector is calculated as the difference between the centroid in the original coordinate system and the one measured in the new coordinate system and rotated by R as follows: $\boldsymbol{t} = \boldsymbol{G}_{\text{old}} - R\boldsymbol{G}_{\text{new}}$.

Then, the rotation matrix is used to calculate the azimuthal angle (Az) for the total station to turn to the same north (y-axis) as in the original coordinate system. Note that only the horizontal angle is needed since the total station is set to level before each use, which results in the x- and y-axes always forming a horizontal plane within machine tolerance. Let $\boldsymbol{j}_{\text{new}} = \begin{bmatrix}0 & 1 & 0\end{bmatrix}$ be the y basis vector with respect to the new coordinate system. Then, the y basis vector of the original coordinate system is obtained as follows: $\boldsymbol{j}_{\text{old}} = R\boldsymbol{j}_{\text{new}}$. The resulting vector is then projected to the horizontal xy plane: $\text{proj}_{xy}(\boldsymbol{j}_{\text{old}})$, which is used to compute the rotation matrix of y-axis on the xy plane using the Euler–Rodrigues formula. The resulting rotation matrix (R_{xy}) is factorized into a sequence of Euler angles around the y-, x-, and z-axes using the user-defined function [10], which always leads to zero angles around the y- and x-axes.

The angle around the z-axis (Az) is sent to the total station to command it to turn to Az and reset north. Note that this angle is the counterclockwise rotation of the original coordinate system to the new one, so the new frame should be turned by the same angle clockwise to align with the original north.

Finally, the original digital twin points are adjusted by the translation. Now the points in the digital space (DT_{adjusted}) have the same coordinates as measured by the total station in its current position and orientation.

2.2 Rigid-body transformation

The movement of solid prefabricated components is considered a rigid-body transformation. By rigidly attaching prisms as reflectors at a known distance from the external corners of the component, the pose of the component can be determined with a total station by measuring the location of at least three prisms. Let $P^n_{\text{curr}}(t)$ be the set of current coordinates of n prisms at time t and P^n_{goal} be the set of corresponding prisms at the goal pose, for $n \geq 3$. An automated workflow was developed to achieve the targeted component pose by transforming $P^n_{\text{curr}}(t)$ into P^n_{goal}, as described by Algorithm 2 and shown in Figure 2.

The center of rotation is important for determining the correct translation. The center of rotation can be any point that moves in the same way as the component, e.g., the center, one corner, or the bearing point of the component. In the local coordinate system of the component, and in the case where there are $n = 3$ prisms, the basis $\mathcal{B}(P^n)$ can be created using the three prisms, i.e., two vectors formed by three prisms and one vector normal to the plane formed by the three prisms. The center of rotation is then calculated as a linear combination of the three basis vectors as follows: $\boldsymbol{G} = \mathcal{B}(P^n)\boldsymbol{g}$. By doing this, the coefficients $\boldsymbol{g}$ of the linear combination in the component coordinate system are independent of the component pose and the global coordinate system set by the total station, in other words, $\boldsymbol{g}$ is constant and independent of time. The coefficients can be determined by referring to the component design drawing or conducting an initial measurement of the three prisms and the center of rotation using the total station. Then the coefficients $\boldsymbol{g}$ can be used to compute the coordinates of the center of rotation using $P^n_{\text{curr}}(t)$ and P^n_{goal}, which is the first step in Algorithm 2.

Next, the rotation matrix $R(t)$ is computed with the Kabsch algorithm after the measured prisms at the current and goal poses are re-centered around their respective center of rotation and then normalized. The translation vector

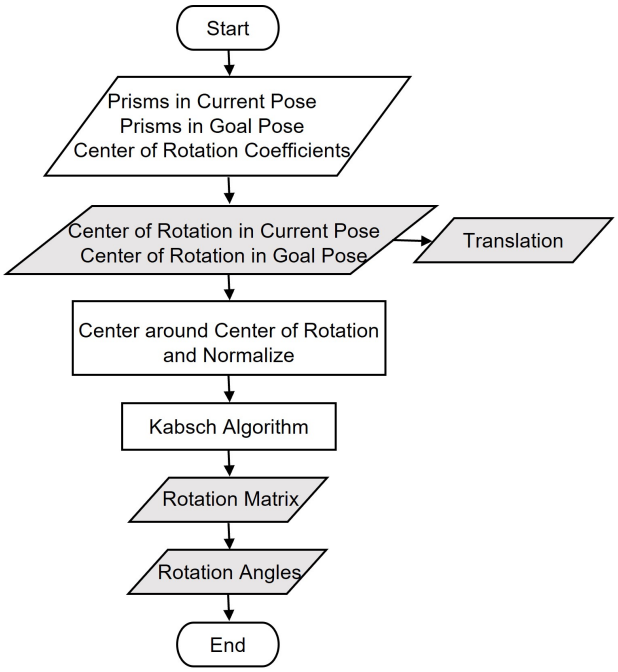

Figure 2. Workflow chart for Algorithm 2: transformation.

Algorithm 2: `transformation`: convert current pose to goal pose

Data:
$P^n_{\text{curr}}(t)$, P^n_{goal}, $\boldsymbol{g}$

Result:
$\boldsymbol{T}(t)$, $\alpha(t)$, $\beta(t)$, $\gamma(t)$

// Calculate the center of rotation at current and goal poses
$\boldsymbol{G}_{\text{curr}}(t) \leftarrow \mathcal{B}(P^n_{\text{curr}}(t))\boldsymbol{g}$
$\boldsymbol{G}_{\text{goal}} \leftarrow \mathcal{B}(P^n_{\text{goal}})\boldsymbol{g}$

// Calculate rotation matrix and translation vector
$p_{\text{curr}}(t) \leftarrow \texttt{normalize}(P^n_{\text{curr}}(t) - \boldsymbol{G}_{\text{curr}}(t))$
$p_{\text{goal}} \leftarrow \texttt{normalize}(P^n_{\text{goal}} - \boldsymbol{G}_{\text{goal}})$
$R(t) \leftarrow \texttt{Kabsch}(\text{align } p_{\text{curr}}(t) \text{ to } p_{\text{goal}})$
$\boldsymbol{T}(t) \leftarrow \boldsymbol{G}_{\text{goal}} - \boldsymbol{G}_{\text{curr}}(t)$

// Find rotation angles
$\alpha(t), \beta(t), \gamma(t) \leftarrow \texttt{factor}(R(t))$

return $\boldsymbol{T}(t)$, $\alpha(t)$, $\beta(t)$, $\gamma(t)$

$\boldsymbol{T}(t)$ is calculated as the difference between the centers of rotation of the goal and current poses.

Finally, the Euler angles around the x, y, and z- axes in a Cartesian coordinate system set by the total station are computed from the rotation matrix $R(t)$. The order of the rotation affects the Euler angles, so it needs to be specified. In Algorithm 2, the rotation matrix is factorized into three Euler angles in the order of y-axis, x-axis, and z-axis, similar to Algorithm 1 [10]. Therefore, the rotation is expressed as a composition of sequential extrinsic rotations about axes y, x, and z with angles $\alpha(t)$, $\beta(t)$, and $\gamma(t)$, respectively.

2.3 Lab-scale experiments

The algorithms for resection (Algorithm 1) and transformation (Algorithm 2) were tested using the Leica Nova MS60 Multi-Station on a lab-scale setup. The shortest measuring distance between the multi-station and reflector target is 1.5 m, while the longest distance varies from 200 m to 10000 m depending on the reflector type and atmospheric conditions. The measuring accuracy can be decreased by haze, sunlight, heat shimmer, beam interruptions, and moving objects within the beam path. The multi-station should be set up in such a way that the beam can reach the reflective part of the target.

The algorithms were written in Python and interfaced with the multi-station using a Bluetooth connection and Leica GeoCOM commands. While the multi-station was used instead of a robotic total station, the algorithms should be able to be applied on robotic total stations as long as the communication between the software and total station can be established because the scanning capability of a multi-station is not needed for executing the algorithms. The following two sections describe the test procedures and evaluation metrics for the resection and transformation algorithms, respectively.

2.3.1 Experiment on resection

The resection algorithm (Algorithm 1) was tested using a rectangular mock-up wall (1.8 m wide by 3.1 m high), a pole with one Leica 360° full-size prism (Model GRZ122, 2.0 mm pointing accuracy) installed on the top, along with the multi-station (Figure 3). The mock-up wall was kept stationary during all tests to simulate a physical building. Four Leica reflective tapes (Model GZM31) were placed close to the four corners of the mock-up wall as control points (*CP*). Two positions were used to set up the multi-station: Position A was at a distance of ~8 m away from the mock-up wall on the right, and Position B was at a distance of ~5 m from the mock-up wall on the left. The pole with the prism was used to determine the exact translation vector ($\boldsymbol{t}$) when the multi-station was moved from one position to the other and was set to a 1.5 m target height for all tests.

A total of four tests were conducted. One baseline test was performed without moving the multi-station to establish the baseline error of the resection algorithm due

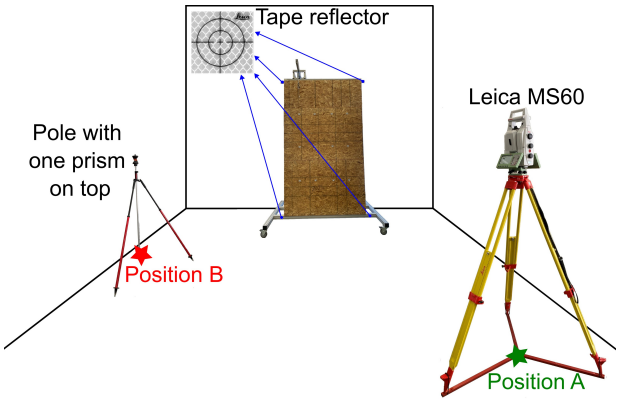

Figure 3. Experimental setup for testing resection with Algorithm 1.

to instrument accuracy. In the remaining three tests, the multi-station was moved from Position A to Position B or vice versa. During each test, the following steps were performed. For ease of explanation, the initial setup in Figure 3 is used as an example.

1. Set up the multi-station at one position (e.g., Position A in Figure 3).
2. Set up the pole at the other position (e.g., Position B in Figure 3).
3. Measure the coordinates of four reflective tapes to obtain the control points (CP_{old}).
4. Measure the coordinates of the prism on the pole.
5. Move the multi-station to Position B (except in the baseline test).
6. Reset the orientation of the multi-station randomly (except in the baseline test).
7. Measure the height of the multi-station.
8. Remeasure the coordinates of the four reflective tapes to obtain the control points in the new coordinate system (CP_{new}).
9. Run Algorithm 1 to determine translation $\boldsymbol{t}$ and change in orientation Az. Adjust the data stored in the original coordinate system to obtain CP_{adjusted} and reset the orientation of the multi-station to Az.
10. Remeasure the coordinates of the four reflective tapes (CP_{measured}).

Two different measurement modes of the multi-station — "Auto" and "Manual" — were used in the tests to measure the coordinates of the reflective tapes. In the "Auto" mode, the laser of the multi-station was pointed close to the reflective tapes, then the multi-station searched for it and measured automatically. In the "Manual" mode, a human manually aimed the laser at the target and commanded the multi-station to measure and record the coordinates of the target. By varying the measurement mode, the error for each measurement mode can be determined and compared.

The true translation of the multi-station was determined in Steps 4 and 7 above. By measuring the prism on the pole, the translations in the x- and y-axes (i.e., the two axes in the horizontal plane) relative to the original setup of the multi-station were directly obtained. To calculate the true translation in the z-axis (i.e., the vertical axis), the z-coordinate of the prism on the pole measured in Step 4 was first subtracted from the height of the prism (1.5 m), which was then subtracted from the height of the multi-station measured in Step 7 after the multi-station was moved.

To assess the accuracy of the resection algorithm, two metrics were used. The first one is the absolute difference between the true and calculated translation along three axes. The second one is the Euclidean distance between CP_{measured} in Step 10 and CP_{adjusted} in Step 9 for each control point on the mock-up wall, which reflects the difference between the true position of control points relative to the new setup of the multi-station and the calculated position based on the resection algorithm.

2.3.2 Experiment on rotation

Experiments were conducted to test the accuracy of the algorithm for rigid-body transformation (Algorithm 2) in computing the rotation angle around a single axis. These simplified experiments isolated the rotation around one axis from translation and rotation around the other two axes, which serve as a good starting point for evaluating the algorithm.

The experimental setup is shown in Figure 4. A 0.76 m high by 0.76 m wide prefabricated wall panel with a mass of 14 kg was installed on a heavy-duty desk monitor arm using a VESA mount. The monitor arm allows for rotating the panel around multiple joints within a range of at least 75°. Three Leica 360° mini prisms (Model GRZ101, 1.5 mm pointing accuracy) were installed on the top right, top left, and bottom left corners of the panel (Figure 4a). One reflective tape was placed at the center of the panel's front face. The table that the monitor arm was attached to was level and provided a flat surface to place one leg of a digital protractor (Bosch GAM 220 MF, ±0.1° accuracy) to measure angles (Figure 4).

In each test, the table remained stationary, and the panel was only rotated around the joint on the VESA mount at the back face of the panel, which formed a rotational movement around the axis (hereafter referred to as *rotation axis*$_{\text{arm}}$) that was perpendicular to the panel face and went through the center of the panel's front face as marked by the reflective tape. The north or y-axis of the multi-station needs to be parallel to *rotation axis*$_{\text{arm}}$ to make sure that the angle around the y-axis obtained from Algorithm 2 is around the same axis as the actual rotation

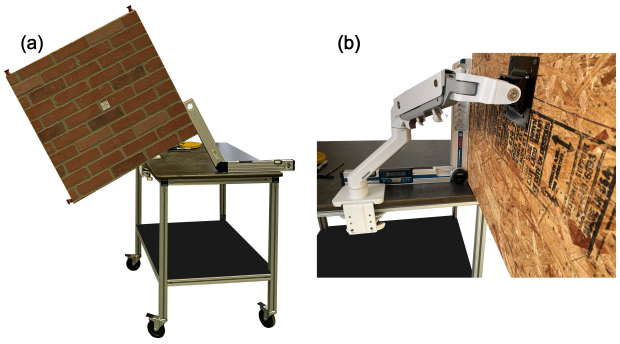

Figure 4. Experimental setup for testing rotation around a single axis with Algorithm 2.

to compare angles. To achieve this, the panel was set to plumb before all tests (a 90° between the table and the panel's back face shown in Figure 4b), and the north of the multi-station was set to perpendicular to the plane formed by the three prisms on the panel at the beginning of each test. The rotation angle around *rotation axis*$_{\text{arm}}$ was determined by measuring the angle between the edge on the right side of the panel and the table before and after the rotation with the digital protractor and then calculating the difference in the two angles (Figure 4a).

The center of rotation is essential in Algorithm 2 to determine translation accurately. While the true center of rotation was the monitor arm's joint at the center of the panel's back face, the reflective tape was used as the center of rotation when running Algorithm 2 because the center of rotation affects translation but not rotation angles. The tape and prisms were measured before all tests to determine the coefficients $\boldsymbol{g}$ of the linear combination that can express the coordinates of the panel center using the coordinates of the three prisms. These coefficients were used to calculate the panel center. A total of seven tests were performed following the following procedure:

1. Set the panel at a position as the goal pose.
2. Measure the angle between the right edge of the panel and the table with the protractor (α_{goal}).
3. Measure the coordinates of the three prisms with the multi-station.
4. Set the north of the multi-station perpendicular to the plane formed by three prisms.
5. Measure the coordinates of the three prisms (P^n_{goal}) and the tape ($\boldsymbol{G}_{\text{goal,measured}}$) with the multi-station.
6. Rotate the panel around the *rotation axis*$_{\text{arm}}$ to the current pose.
7. Measure the angle between the right edge of the panel and the table with the protractor (α_{curr}).
8. Measure the coordinates of the three prisms ($P^n_{\text{curr}}(t)$) and the tape ($\boldsymbol{G}_{\text{curr,measured}}$) with the multi-station.
9. Run Algorithm 2 to obtain the angles between the current panel pose and the goal pose.

The accuracy of the algorithm was evaluated by the difference between the rotation angles that were measured with the protractor and calculated with the algorithm. The measured rotation angle was obtained by subtracting α_{goal} in Step 2 from α_{curr} in Step 7. Additionally, the translation from the current panel position to the goal position was compared between the algorithm output and measurement. The measured translation was obtained by subtracting $\boldsymbol{G}_{\text{curr,measured}}$ in Step 8 from $\boldsymbol{G}_{\text{goal,measured}}$ in Step 5.

3 Results

This section presents the performance of the algorithm on resection (Algorithm 1) first, followed by the performance of the algorithm on rigid-body transformation (Algorithm 2).

3.1 Experiment on resection

Four tests where the multi-station measured four control points fixed on a mock-up wall were conducted, and the translation and the rotation angle of the north ("azimuth") of the multi-station are shown in Table 1.

In Test No. 1 where the multi-station stayed stationary, the maximum absolute error in translation was 1.0 mm and that the rotation angle in the north was 0.004°. These non-zero values were caused by the measurement error of the multi-station.

In Tests No. 2—4, the multi-station was translated in all three axes and rotated around the vertical axis (z-axis) by an angle in a wide range. The results of Tests No. 2—4 show that the translation calculated by Algorithm 1 was close to the true translation as determined by the full-size prism on a pole with a maximum absolute error of 6.5 mm on one axis. It should be noted that the true error in translation was likely smaller than 6.5 mm because the multi-station may have not been set up at the exact location of the pole due to human error. Therefore, the accuracy of Algorithm 1 is better reflected by the Euclidean difference in the measured and adjusted coordinates of the four control points on the mock-up wall as shown in Table 2.

Consistent with Table 1, Table 2 shows a small baseline error (an average of 0.6 mm point difference) in the coordinates of control points in Test No.1 due to instrument error. After the multi-station was translated and rotated (Tests No. 2—4), Algorithm 1 was able to predict the coordinates of control points in the current setup with an average error between 2.3 and 3.1 mm, which is lower than a threshold of 3.2 mm as recommended by industry partners for measuring panel positions [2]. The error varied among different control points and tests, but the variation was small. There was no noticeable difference between

Table 1. Performance of Algorithm 1: translation and azimuth.

Test No.			1	2	3	4
	Measurement mode		Auto	Manual	Auto	Auto
Translation	True (m)	x	0	4.4909	-3.1040	-2.9615
		y	0	-1.8064	3.2602	3.8264
		z	0	0.0037	-0.0076	-0.3000
	Calculated (m)	x	0.0010	4.4898	-3.1105	-2.9658
		y	0.0001	-1.8122	3.2657	3.8299
		z	-0.0005	0.0046	-0.0066	-0.3017
	Absolute error (mm)	x	1.0	1.1	6.5	4.3
		y	0.1	5.8	5.5	3.5
		z	0.5	0.9	1.0	1.7
Azimuth	Calculated (°)		0.004	58.3	83.7	267.9

Table 2. Performance of Algorithm 1: Error in the coordinates of four control points (*CP*, mm).

Test No.	1	2	3	4
CP_1	0.7	2.5	1.9	2.8
CP_2	0.6	3.1	1.5	3.1
CP_3	0.6	3.9	4.1	3.8
CP_4	0.6	3.1	1.5	2.6
Average	0.6	3.1	2.3	3.1
Maximum	0.7	3.9	4.1	3.8

Test No.2 and the other two tests with a different measurement mode, which indicates that the auto mode can achieve an accuracy as good as the manual aiming mode.

While on average, the error in the predicted coordinates was acceptable for prefabricated construction, the maximum error was about 1 mm above the threshold of 3.2 mm (Table 2). Therefore, this error needs to be reduced further. One approach is to investigate the reason for the maximum error appearing at CP_3 which was the control point at the lower left corner of the mock-up wall in all three tests. It was likely that the tape of this control point was not measured as accurately as the other three tapes. Another approach is to study if the error can be reduced by using more control points as the input to Algorithm 1.

3.2 Experiment on rotation

The algorithm of rigid-body transformation (Algorithm 2) was tested by rotating a panel around a single y-axis, and the results are displayed in Table 3.

The test panel was rotated around the y-axis as set by the multi-station by an angle from $\sim 5°$ to $\sim 26°$ at an interval of $\sim 5°$ both in the clockwise and counterclockwise direction. An angular error smaller than 0.5° was achieved when comparing the rotation angle measured by the digital protractor and the angle around the y-axis calculated by Algorithm 2. The error in the angle tends to be larger when the angle is larger, which was likely caused by a larger human error in rotating the panel. In addition to the angle around the y-axis, angles less than 0.57° in absolute value were obtained for the x and z- axes. The non-zero angles around these two axes were caused by the measurement error of the multi-station and the possibility that the panel was slightly rotated around these two axes when it was rotated around the monitor arm's joint on the VESA mount due to human error. The experiment setup needs to be improved further to eliminate the human error.

The true translation of the panel is zero because the panel was only rotated, and the translation measured by the tape at the panel center and that calculated by using the coefficients of the linear combination of three prisms are both nearly zero. A maximum Euclidean distance error of 5.5 mm was observed in the measured and calculated translation. This error can be reduced further by increasing the accuracy of the calculated coefficients of the linear combination by taking the average of multiple measurements.

4 Conclusions and next steps

Two algorithms were developed to aid the real-time tracking and positioning of prefabricated components in construction. The resection algorithm enables the alignment of the digital twin with physical twin regardless of the position and orientation of the total station. Lab-scale tests show that an average error of less than 3.1 mm was achieved in point coordinates, which satisfies the industry recommendation of less than 3.2 mm. The second algo-

Table 3. Performance of Algorithm 2

Test No.	Measured angle (°)			Rotation angle from Algorithm 2 (°)			Error	
	Current	Goal	Rotation angle	y-axis	x-axis	z-axis	Rotation angle (°)	Translation (mm)
1	95.4	90.2	5.2	5.13	0.03	0.29	0.07	1.0
2	100.9	91	9.9	9.91	0.02	0.12	0.01	5.5
3	105.3	90.8	14.5	14.42	-0.01	-0.29	0.08	4.0
4	111.7	91	20.7	20.45	0.01	-0.33	0.25	4.8
5	115.7	89.4	26.3	26.13	-0.24	-0.57	0.17	1.9
6	90.8	111.6	-20.8	-20.45	-0.23	0.43	0.35	4.4
7	91.1	117.9	-26.8	-26.37	-0.22	0.32	0.43	3.0

rithm transforms the component from the current pose to the intended goal pose. The tests of rotating a component around a single axis show that the error in the rotation angle was less than 0.5°, which is acceptable given the large rotation angles of the component.

Several limitations in this paper require future research. First, the errors observed in the performance of the two algorithms need to be reduced. Potential approaches include reducing instrument error (e.g., selecting different measurement settings, using reflectors with higher accuracy), revising the algorithms to decrease their sensitivity to instrument error, and eliminating human error by designing better experiments. Second, experiments that can test the transformation algorithm for rotations around all three axes need to be conducted. Third, the Euler angles around the x, y, and z- axes in a Cartesian coordinate system set by the total station are currently used in the transformation algorithm. The choice of using the prefabricated component as the local coordinate system should be included in the algorithm. Ultimately, these two algorithms will be implemented on the Real-Time Evaluator (RTE) to optimize and automate the process of installing prefabricated components with high accuracy.

5 Acknowledgements

This research was supported by the DOE Office of Energy Efficiency and Renewable Energy (EERE), Building Technologies Office, under the guidance of Sven Mumme, and used resources at the Building Technologies Research and Integration Center (BTRIC), a DOE-EERE User Facility at Oak Ridge National Laboratory.

References

[1] Tarek Omar and Moncef L Nehdi. Data acquisition technologies for construction progress tracking. *Automation in Construction*, 70:143–155, 2016.

[2] Nolan W. Hayes, Bryan P. Maldonado, Diana Hun, and Peter Wang. Automated tracking of prefabricated components for a real-time evaluator to optimize and automate installation. In *Proceedings of the 40th International Symposium on Automation and Robotics in Construction*, pages 192–199, Chennai, India, 2023.

[3] Bryan P. Maldonado, Nolan W. Hayes, and Diana Hun. Automatic point Cloud Building Envelope Segmentation (Auto-CuBES) using Machine Learning. In *Proceedings of the 40th International Symposium on Automation and Robotics in Construction*, pages 48–55, Chennai, India, 2023.

[4] Guobin Chang. On least-squares solution to 3d similarity transformation problem under gauss–helmert model. *Journal of Geodesy*, 89(6):573–576, 2015.

[5] Leon C Price, Jingdao Chen, Jisoo Park, and Yong K Cho. Multisensor-driven real-time crane monitoring system for blind lift operations: Lessons learned from a case study. *Automation in Construction*, 124:103552, 2021.

[6] Yan Xu, Yi Luo, and Jian Zhang. Laser-scan based pose monitoring for guiding erection of precast concrete bridge piers. *Automation in Construction*, 140:104347, 2022.

[7] Jim Lawrence, Javier Bernal, and Christoph Witzgall. A purely algebraic justification of the kabsch-umeyama algorithm. *Journal of research of the National Institute of Standards and Technology*, 124:1–6, 2019.

[8] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[9] Jian S Dai. Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections. *Mechanism and Machine Theory*, 92:144–152, 2015.

[10] David Eberly. Euler angle formulas. Online: `https://www.geometrictools.com/Documentation/Documentation.html`. Accessed: 15/12/2023.