Precise Assembly of 3D Truss Structures Using EKF-based Error Prediction and Correction

Erik Komendera and Nikolaus Correll

Department of Computer Science, University of Colorado at Boulder,
430 UCB, Boulder, CO, USA
{erik.komendera,nikolaus.correll}@colorado.edu

Abstract. We describe a method to construct precise truss structures from non-precise commodity parts. Trusses with precision in the order of micrometers, such as the truss of a space telescope, can be accomplished with precisely machined truss connection systems. This approach is expensive, heavy, and prone to failure, e.g., when a single element is lost. In the past, we have proposed a novel concept in which non-precise commodity parts can be aligned using precise jigging robots and then welded in place. Even when using highly precise sensors and actuators, this approach can still lead to errors due to thermal expansion and structural deformation. In this paper, we describe and experimentally evaluate an EKF-based SLAM approach that allows a team of intelligent precision jigging robots (IPJR) to maintain a common estimate of the structure's pose, improve this estimate during loop closures in the construction process, and uses this estimate to correct for errors during construction. We also show that attaching a new node to the assembly site with the lowest uncertainty significantly increases accuracy.

1 Introduction

While high-precision assembly and welding have long been staples of factory automation, precise assembly in the field remains a difficult challenge. Industrial robots such as pick-and-place machines can achieve high precision with programmed motions, and are a critical component of modern industry, but do not function outside of their carefully controlled workspaces. Research into field assembly often focuses on self-correcting, interlocking components, freeing the robots from the task of ensuring accuracy, but this does not address the use of raw materials and on-line adjustments.

Large scale construction projects do not rely solely on self-correcting, interlocking components, and rigid bodies are not valid assumptions. Instead, it is common to cut or bend parts from stock materials as needed, often incorporating the state of the structure (including any position errors). Likewise, welding and other thermal processes induce stresses on structures, which change the outcome slightly. If robots assembling the structure were to assume rigid bodies, eventually error would accumulate and lead to geometrical issues that cannot be overcome without significant repairs, including inducing forces on the structure to "jam" in parts, or disassembling and trying again.

Assembling structures in space [1] has the potential to overcome payload limitations of earth-based missions, thereby enabling the manufacturing of scalable structures. Space telescopes that could be assembled on orbit are a high priority for NASA [2], with proposed diameters of tens of meters up to hundreds of meters; in contrast, the James Webb Space Telescope, at 6.5m, is at the upper limit for deployable telescopes [3]. Many of the benefits and techniques for assembling large space observatories are discussed in [4]. One recent robotic telescope assembly experiment [5] demonstrated the repeated assembly and disassembly of an 8 m telescope mirror, composed of 102 precisely machined truss members and 12 hexagonal panels, using an industrial manipulator arm and a rotating assembly platform. However, this approach for assembling large-scale telescope truss structures and systems in space have been perceived as very costly because they require high precision and custom components that rely on mechanical connections, increased launch mass, and the potential for critical failure if only one of the components is missing or defective [3].

This paper introduces a new prototype of the Intelligent Precision Jigging Robot (IPJR), for use with three dimensional truss structures such as shown in Figure 1, extending on our previous work on 2D IPJRs [6, 7]. IPJRs work in groups to assemble trusses one cell at a time: each IPJR rests on a single strut between the structure and the new node, and adjusts the length of the strut until attachment is ready to be performed.

Although the 2D IPJRs were able to assemble structures with accuracies that far exceeded that of the materials and processes used (glue gun and wooden dowels in [6] and spot welding of titanium rods facilitated by a large-scale robotic manipulator [7]), a fundamental problem in the IPJR approach are bias due to thermal expansion or structural deformation.

This paper also introduces an assembly algorithm based on the Extended Kalman Filter (EKF) and simultaneous localization and mapping (SLAM). Here, the EKF allows us to maintain and update a probabilistic state estimate (assuming a Gaussian distribution of the error), whereas SLAM allows us to take advantage of loop closures in the construction process to update the state of the entire structure. We can then use this improved estimate to change the build path during the construction process to minimize the expected variance as well as correct for errors by adjusting the length of future struts being placed.

To test the EKF-based algorithm, we perform several simulations and physical experiments on a telescope truss made of stock aluminum tubes, using IPJRs made from off-the-shelf and laser-cut parts. We show that the algorithm can detect and overcome hysteresis, bending and inconsistent IPJR robots by simulating the assembly process with an artificial bias. We compare the algorithm to a version without EKF in simulation, and assemble the structure with the EKF algorithm in a physical experiment. We show that the EKF assembly algorithm is a significant improvement over the assumption of zero-mean error, i.e., open-loop assembly of a structure.

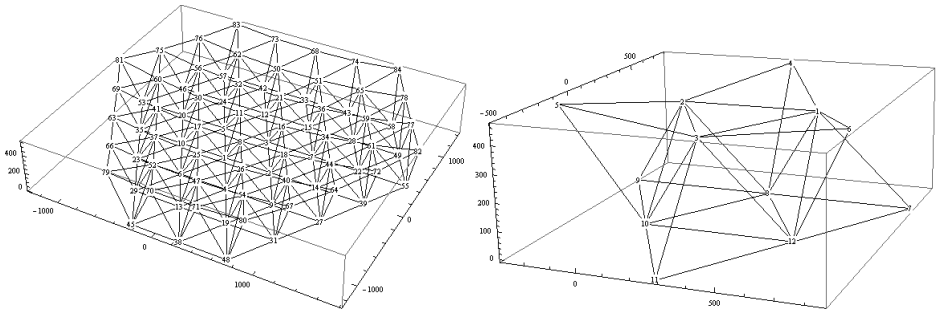

Fig. 1. Left: The full telescope assembled in [2]. Right: The innermost 12 nodes and 30 struts of the same telescope, with the top surface scaled by a factor of 4 to emphasize the curved surface. Both structures were considered in simulation, and the smaller one in physical experiments.

2 Related work

Algorithms for mechanical assembly planning (via disassembly sequences) for problems in well-known environments such as assembly lines, and with few assembly robots, were explored in the 80s and 90s, resulting in algorithms for finding fully-ordered sequences [8–11], or using opportunistic assembly planning [12] when necessary. All of these approaches attempted to find a suitable order for assembly that avoids construction deadlock, and assumed rigid bodies. In reality, structural forces and minor assembly errors often lead to situations where parts of a structure must be forced open to jam a new component in. Such algorithms do not consider these situations.

More recently, distributed field assembly by robots has been explored in large-scale assembly tasks. Three different robots are shown to successfully dock a part to an assembly [13]. An experiment performed at NASA's Jet Propulsion Laboratory demonstrated the precision assembly of beams by a pair of cooperative robots using highly rigid motions to ensure precision [14]. A robust algorithm is described in [15], which uses teams of robots to assemble structures while handling exceptions due to a wide range of failures, and relying only on a human operator when failures are beyond the scope of the assembly robots. Taking into account physical constraints such as structural stability and material properties into the build order has been shown in [16].

Recent robust and parallel assembly techniques include quadrotor teams that can assemble cubic truss structures [17], truss climbing and assembling robots [18], termite-inspired swarm assembly robots [19], and a robot team that can build IKEA furniture in cluttered environments [20]. Mobile assemblers [21] used visual feedback to estimate and correct errors in assembly. Many of these methods rely on self-correcting, interlocking mechanisms, which would add extra mass and expense due to machining requirements, and none consider welding or cutting.

3 Motivating Example and Experimental Setup

We are using the 84-node, 315-strut telescope truss described in [2] as the motivating example throughout the paper. We simulate its assembly, scaled by 1/6 and flattened on the bottom surface, as well as a 12-node, 30-strut subset of the structure with the top surface curvature scaled by a factor of 4, which we also assemble physically. Schematics are shown in Figure 1. Our performance metric is the average error of the node positions in the coordinate frame defined by an origin node, an X-axis node, and an XY-plane node.

This experiment considers the assembly of truss structures one node at a time, starting from the predefined origin node, and working incrementally until the entire structure is finished. For each additional node added, at least as many struts as there are nodal degrees of freedom must be added simultaneously. In three dimensions, a minimum of three IPJRs are required to guarantee a unique node position, with the exception of grounded nodes, which require only two when the z-position is fixed.

In order to forgo premade accurate building materials, we chose one-size-fits-all telescoping aluminum rods (30" long, 1/4" and 7/32" diameter) that can be locked in place with a shaft collar and that connect to the node balls via neodymium magnets. These design choices make assemblies temporary and allow us to reuse materials in different experiments. Each IPJR is designed to adjust the lengths of the telescoping struts by attaching to them during the assembly of a new part of the structure. In this paper, node balls are ball joints: the struts endpoints are free to slide around the surfaces of node balls.

The node-to-node nominal distances for the chosen structure range from 504.6 mm to 574.3 mm. A close-up image of a node ball with struts in the completed structure is shown in Figure 2. We acquired 18 node balls and 51 struts, and used 12 node balls and 30 struts for the physical experiments.

We built five IPJRs for use in this experiment. Each IPJR, shown in Figure 2, is an autonomous robot, consisting of a Raspberry Pi Model B for high level algorithmic control and communications, an Arduino for actuation and sensing, a motor driver board, and an Edimax WiFi dongle. The principal actuator is a Firgelli L-16 linear actuator with $140mm$ extension, with advertised $0.5mm$ accuracy, and potentiometer length feedback discretized to 140/1024 steps. To attach to both ends of a strut, each IPJR has two shaft collars. All components are fixed to a frame consisting of laser cut, 1/4" acrylic sheets. Each IPJR was powered by a power supply capable of 5V and 12V output for the electronics and motors, respectively.

While each IPJR is fully capable of running our implementation of the EKF assembly algorithm as-is, we chose to run the algorithm from a central PC to avoid communication challenges. The Raspberry Pis instead run an HTTP server, which receives commands in the form of GET requests, and returns the data in response. Commands used in this experiment include checking the length potentiometer voltage, commanding the IPJR to move until it stops near a commanded location (without further control), and checking to see if the IPJR has finished moving. The control PC executes the experiment using a version of the EKF

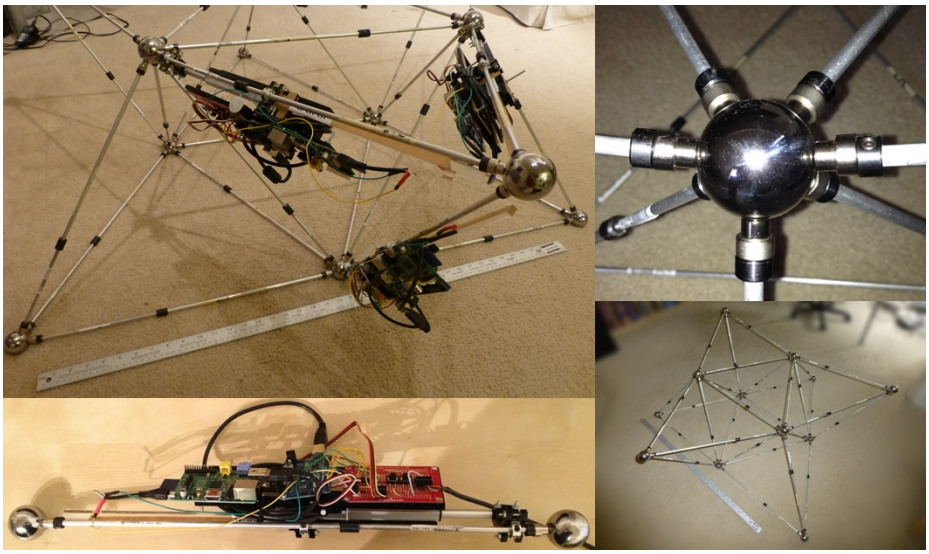

Fig. 2. Top left: three IPJRs attached to a small truss. Top right: close up of a node, showing the strut-node magnet interface. Bottom left: an IPJR prototype attached to a strut. Bottom right: a completed small truss.

assembly algorithm implemented in Mathematica, which controls and monitors the IPJRs through GET requests.

For this experiment, a human external manipulator is required to attach the IPJRs to struts and to place each new cell in a coarse configuration. Once placed, the IPJRs refine the cell by taking measurements of the distances between node balls. The EKF algorithm requires direct measurements of the structure, but the IPJRs lacked that capability for these experiments. Instead, a ruler was used in order to obtain an unbiased measurement, not subject to bending due to the weight of the IPJR or play in the shaft collar attachment mechanism. The ruler was also used for verification of the physical experiments using a maximum likelihood estimator.

4 EKF Assembly Algorithm

A truss structure consists of nodes and struts that can be mathematically abstracted to vertices and edges of a graph. The geometry of the truss is fully defined by the graph topology and the length of each individual edge. As the truss consists of triangles, this information is sufficient to unambiguously define all angles in the system. In order to construct a specific truss, we therefore need to precisely set the length of each strut. The assembly algorithm considers the assembly of the structure by attaching nodes that share an edge with nodes already attached.

Algorithm 1 EKF Assembly Algorithm.

Require: $\bar{\mathbf{X}}, S, \Delta, \sigma_l, \sigma_c, ChooseSite \in \{MinTrace, MaxTrace, Random\}$
  $\hat{\mathbf{X}} = \mathbf{0}_{3nx3n}, \mathbf{P} = \mathbf{0}_{3nx3n}, \mathbf{A} = \{0\}$
  while $||\mathbf{A}|| < n$ **do**
    $f = ChooseSite(\hat{\mathbf{X}}, \mathbf{P}, S, \mathbf{A})$
    while $||\bar{\mathbf{x}}_f - \hat{\mathbf{x}}|| > \Delta$ **do**
      $\bar{\mathbf{c}}_f = ||\bar{\mathbf{x}}_\mathbf{f} - \hat{\mathbf{x}}_\mathbf{i}|| \forall i \in \mathbf{J}_f$
      Command all IPJRs with lengths $\bar{\mathbf{c}}_f$
      Collect measurements $\mathbf{y}$
      $\hat{\mathbf{X}}, \mathbf{P} = EKF(\hat{\mathbf{X}}, \mathbf{P}, \mathbf{y}, \bar{\mathbf{c}}_f, \sigma_l, \sigma_c)$
    end while
    $\mathbf{A} = \mathbf{A} \cup \{f\}$
  end while
  return $\hat{\mathbf{X}}, \mathbf{P}$

Algorithm 2 Extended Kalman Filter.

Require: $\hat{\mathbf{X}}, \mathbf{P}, \mathbf{y}, \bar{\mathbf{c}}_f, \sigma_l, \sigma_c$
  $\hat{\mathbf{X}}^P = f(\hat{\mathbf{X}}, \mathbf{0})$
  $\mathbf{F} = \frac{\delta f(\mathbf{X}, \mathbf{w})}{\delta \mathbf{X}}\Big|_{\mathbf{X}=\hat{\mathbf{X}}, \mathbf{w}=\mathbf{0}}$
  $\mathbf{G} = \frac{\delta f(\mathbf{X}, \mathbf{w})}{\delta \mathbf{w}}\Big|_{\mathbf{X}=\hat{\mathbf{X}}, \mathbf{w}=\mathbf{0}}$
  $\mathbf{P}^P = \mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{G}(\mathbf{I}\sigma_c)\mathbf{G}^T$
  $\mathbf{H} = \frac{\delta g(\mathbf{X}, \mathbf{v})}{\delta \mathbf{X}}\Big|_{\mathbf{X}=\hat{\mathbf{X}}^P, \mathbf{v}=\mathbf{0}}$
  $\mathbf{U} = \frac{\delta g(\mathbf{X}, \mathbf{v})}{\delta \mathbf{v}}\Big|_{\mathbf{X}=\hat{\mathbf{X}}^P, \mathbf{v}=\mathbf{0}}$
  $\mathbf{K} = \mathbf{P}^P\mathbf{H}^T(\mathbf{H}\mathbf{P}^P\mathbf{H}^T + \mathbf{U}(\mathbf{I}\sigma_l)\mathbf{U}^T)^{-1}$
  $\hat{\mathbf{X}} = \hat{\mathbf{X}}^P + \mathbf{K}(\mathbf{y} - g(\hat{\mathbf{X}}^P, \mathbf{0}))$
  $\mathbf{P} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}^P$
  return $\hat{\mathbf{X}}, \mathbf{P}$

The assembly algorithm (Algorithm 1) takes as input the nominal structure $\bar{\mathbf{X}}$ and maintains its estimated state in vector $\hat{\mathbf{X}}$. Both $\bar{\mathbf{X}}$ and $\hat{\mathbf{X}}$ are $3n$-length vectors of node positions $\mathbf{x_i} = \{x_i, y_i, z_i\}$. The state vector does not have an external reference for the coordinate system; instead, the coordinate frame is a function of the first three nodes $\mathbf{x_1} = \{0, 0, 0\}$, $\mathbf{x_2} = \{x_2, 0, 0\}$, and $\mathbf{x_3} = \{x_3, y_3, 0\}$, defining the xy-plane to be the triangle formed by these nodes, with node 1 at the origin and node 2 on the x-axis. The origin node is placed first. With the exception of the first three nodes, the ordering of the nodes in $\bar{\mathbf{X}}$ and $\hat{\mathbf{X}}$ is independent of the assembly order. The set of strut edges is defined by S, and indexes into the ordering of $\bar{\mathbf{X}}$ and $\hat{\mathbf{X}}$. Δ is the position distance tolerance. σ_l is a vector of edge measurement variances for each IPJR. σ_c is a vector of edge length actuation variances for each IPJR.

The function *ChooseSite* which chooses the next node to attach using a heuristic and the EKF algorithm are described in Sections 4.1 and 4.2, respectively.

The algorithm initializes the structure state estimate $\hat{\mathbf{X}}$ and the $3n$-square covariance matrix $\mathbf{P}$ to zero, and the set of assembled nodes to be the origin node, $\mathbf{A} = \{0\}$. Then, until the size of $\mathbf{A}$ is equal to n:

- The algorithm chooses which node f to attach next, which is described in Section 4.1.
- The connecting nodes in A are mapped to the IPJR order in a vector $\mathbf{J}_f$; IPJR 1, for example, will connect f to the first node in $\mathbf{J}_f$. The commanded lengths $\bar{\mathbf{c}}_f$ are calculated to be the norm of the desired position for node f minus the estimated position of node i for all i in $\mathbf{J}_f$.
- The IPJRs are commanded to extend or contract to the lengths in $\bar{\mathbf{c}}_f$, with added (hidden) process noise $\mathbf{w} = N(0, \sigma_c)$
- Measurements of the IPJRs are collected in the same order, and assigned to measurement vector $\mathbf{y}$. Each measurement is of the form $\mathbf{y} = ||\mathbf{x}_i - \mathbf{x}_f|| \forall i \in \mathbf{J}_f$ with added measurement noise $\mathbf{v} = N(0, \sigma_l)$.
- The Extended Kalman Filter is executed to update the current state estimate $\bar{\mathbf{X}}$ and covariance $\mathbf{P}$, with measurements $\mathbf{y}$, commands $\bar{\mathbf{c}}_f$, process variance σ_c, and measurement variance σ_l.
- If the estimated position of the new node $\hat{\mathbf{x}}_f$ is within Δ from the desired state $\bar{\mathbf{x}}_f$, command the IPJRs again, else place f into $\mathbf{A}$ and move on to the next node.
- When the assembly is complete, return $\bar{\mathbf{X}}$ and $\mathbf{P}$.

4.1 Choosing the next site

The function *ChooseSite* can be one of three choices: the minimum trace heuristic (2), the maximum trace heuristic (2), or the random heuristic (3):

$$f = \underset{i}{\operatorname{argmin}}(Tr(\mathbf{P}_{\mathbf{J}_i})) \quad (1)$$
$$\forall i \in \{i | \exists a_1, a_2, a_3 \in \mathbf{A} \wedge (a_1, i) \in S \wedge (a_2, i) \in S \wedge (a_3, i) \in S\}$$
$$f = \underset{i}{\operatorname{argmax}}(Tr(\mathbf{P}_{\mathbf{J}_i})) \quad (2)$$
$$\forall i \in \{i | \exists a_1, a_2, a_3 \in \mathbf{A} \wedge (a_1, i) \in S \wedge (a_2, i) \in S \wedge (a_3, i) \in S\}$$
$$f = Random(\{i | \exists a_1, a_2, a_3 \in \mathbf{A} \wedge (a_1, i) \in S \wedge (a_2, i) \in S \wedge (a_3, i) \in S\}) \quad (3)$$

For each heuristic, the first step is to collect a list of all of the possible next nodes, which requires that there are at least three struts in S between nodes in A and each possible node (the exceptions are nodes 2 and 3, which only require the origin and the first two nodes). If the heuristic is random choice, one of the possible nodes is randomly chosen. Otherwise, for each possible node, the trace of the submatrix of $\mathbf{P}$ corresponding to the connecting nodes in A is found. For the minimum trace heuristic, the possible node that minimizes the trace is returned; for the maximum trace heuristic, the node that maximizes the trace.

4.2 Extended Kalman Filter

The Extended Kalman Filter (Algorithm 2) is a standard sequential estimation technique that linearizes the state and measurement models around the

estimated state so that the updated state is near-optimal given the new measurements. For brevity, we assume readership familiarity, and will define the additional terms without a tutorial. $\hat{\mathbf{X}}^P$ is the predicted update without noise. $\mathbf{F}$ and $\mathbf{G}$ are the Jacobians of the state function $f(\mathbf{X}, \mathbf{w})$ with respect to the state variables $\mathbf{X}$ and process noise variables $\mathbf{w}$, evaluated at the previous state $\hat{\mathbf{X}}$ without noise. The EKF considers the state transition function in the form $\mathbf{X}^{t+1} = f(\mathbf{X}^t, \mathbf{w})$ and $\mathbf{y} = g(\mathbf{X}^{t+1}, \mathbf{v})$. The state transition function is:

$$f(\mathbf{x}_i^{t+1}, \mathbf{w}) = \begin{cases} CalcNode(\bar{\mathbf{c}}_\mathbf{i}, \mathbf{X}_{\mathbf{J}_i}, \mathbf{w}) & \text{if } i \text{ is currently being placed} \\ \mathbf{x}_i^t & \text{if } i \text{ is built or yet to be built} \end{cases}$$

The function $CalcNode()$ is the solution for the attachment node position $\mathbf{x}_f$ to the nonlinear system of equations $\bar{c}_{if} + w_{if} = ||\mathbf{x}_i - \mathbf{x}_f||$ for all fixed nodes i and attachment node f: with desired lengths, known fixed nodes, and some noise, where does the attachment node go? With some algebraic manipulation, $\mathbf{x}_f$ can be found as a function of the states, lengths, and noises. The term $(\mathbf{I}\sigma_c)$ transforms the process variance vector into a diagonal matrix. The predicted covariance matrix estimate $\mathbf{P}^P$ is then updated with the sum of the state covariance propagation matrix and the process covariance matrix. Likewise, the measurement variance model $g(\mathbf{X}, \mathbf{v})$ is linearized with respect to the state variables $\mathbf{X}$ and measurement noise variables $\mathbf{v}$ to produce $\mathbf{H}$ and $\mathbf{U}$ at the predicted state $\hat{\mathbf{X}}^P$ without noise, and the term $(\mathbf{I}\sigma_l)$ transforms the measurement variance vector into a diagonal matrix. The Kalman gain $\mathbf{K}$ models the relative certainties of the state propagation and the measurement process. These terms are used to update $\hat{\mathbf{X}}$ and $\mathbf{P}$.

4.3 Verification

After each simulation was complete, the hidden state was directly compared to the final estimated states. For the physical experiments, we did not directly measure $\mathbf{x}_i$ and had to use a maximum likelihood estimator (MLE) on a set of ruler measurements. The MLE was used *independently* of the EKF algorithm. The MLE finds the structure that maximizes the joint probability distribution function of a structure, which is the product of the node position prior probabilities centered on $\bar{\mathbf{x}}$ and the measurement probabilities centered on the true distances $||\mathbf{x}_i - \mathbf{x}_j||$:

$$\tilde{\mathbf{X}} = \underset{\mathbf{X}}{\text{argmax}}\Big(\sum_{i=1}^{n} Log\left(\frac{\exp\left(\frac{1}{2}\left(-\frac{(x_i-\bar{x}_i)^2}{\sigma_p} - \frac{(y_i-\bar{y}_i)^2}{\sigma_p} - \frac{(z_i-\bar{z}_i)^2}{\sigma_p}\right)\right)}{2\sqrt{2}\pi^{3/2}\sqrt{{\sigma_p}^3}}\right) + \sum_{i,j\in S_m} Log\left(\frac{\exp\left(-\frac{\left(\hat{l}_{ij}-\sqrt{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2}\right)^2}{2{\sigma_l}^2}\right)}{\sqrt{2\pi}\sigma_l}\right)\Big) \tag{4}$$

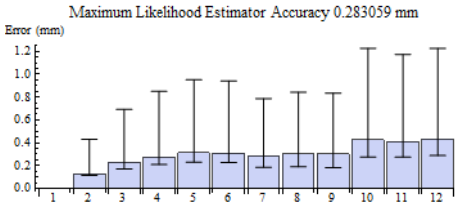

Fig. 3. The accuracy of the maximum likelihood estimator, used only for verification of the final physical assemblies, was simulated over 1000 random trials: the mean error of the maximum likelihood estimator with 5th and 95th percentiles is shown. The x-axis represents the node assembly order for the minimum trace case, and the further the node, the greater error in the MLE

To get the best estimate, we measured as many node-node distances as our $1m$ ruler would allow, which accounted for 57 pairs (S_m) of the possible 66 pairs in the 12-node structure, including 27 pairs not directly connected by struts. The ruler we used was $1m$ long with a variance of $\sigma_l = 0.25mm^2$. To determine the accuracy of the MLE itself, we simulated 1000 structures with each node coordinate offset by a variance of $25mm^2$, and set the node prior variance $\sigma_p = 1000mm^2$ to allow the ruler to dominate. MLE is accurate to within $0.3mm$ on average, and for the least certain node, 95 percent of the estimates are more accurate than $1.2mm$ (Figure 3), making this our confidence interval for the physical measurements.

5 Results

We first calibrated the IPJRs and found them have a minimum length of $\{496.5, 495.0, 500.7\}mm$, command variances $\sigma_c = \{1.15, 1.49, 0.16\}mm^2$, and step lengths $\{0.135, 0.138, 0.132\}mm$. The ruler used to calculate node distances was $\sigma_l = 0.25mm^2$. These results are used both for conducting simulations and during the EKF on the real robot experiment.

5.1 Simulation

To come up with a realistic set of simulations and to explore the effects of inaccurate calibration, the EKF algorithm used the calibrated values, but the hidden, simulated IPJRs *did not* match the calibration (with the exception of a control case). Each time an IPJR is attached to a new strut, the hidden offset and step size are reset. The reason is threefold: when each IPJR is attached to a new strut, there will be a different offset each time; the attachment quality (and strut quality) varies; and the orientation of the strut and IPJR can lead to variable bending. For both the large and small truss structure, we used two control cases: open loop with perfect calibration, and open loop with hidden biases. In these

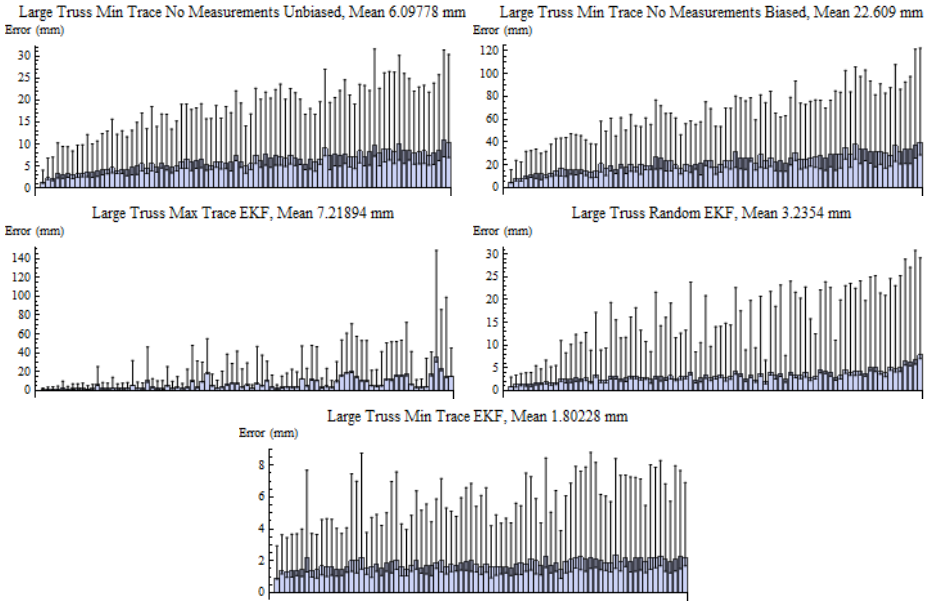

Fig. 4. Results of 100 simulated assemblies of the large truss. In reading order, starting at top left: the perfect calibration case without measurements, the hidden bias case without measurements, the maximum trace case with the EKF algorithm, the random case with the EKF algorithm, and the minimum trace case with the EKF algorithm. The x-axis indicates the node order placement. Error bars represent the 5th and 95th percentiles.

cases, the Extended Kalman Filter uses only the IPJR command variances, and uses the $MinTrace$ heuristic; thus, the control cases will attempt to place new nodes on what it thinks are the most certain sites. The control cases represent optimistic open loop results, the perfect calibration case more so than the hidden bias case.

The experimental cases test the EKF algorithm with measurements, using the minimum trace, random, and maximum trace heuristics. In the EKF assembly cases, the measurements were assumed to dominate the command variance, so we set the command variances to $\sigma_c = \{1000, 1000, 1000\} mm^2$ to treat the rulers as nearly-truth. To simulate hidden biases, each time an IPJR is placed on a strut, the hidden simulated IPJR offsets were randomly offset by up to $7.5mm$ in either direction, and the step sizes were randomly offset by up to $5\mu m$. These values were chosen empirically based on observations in physical experiments with the IPJRs and the telescope trusses.

We ran 100 simulations of the two control cases and the three heuristics for both structures. We present the large structure results first.

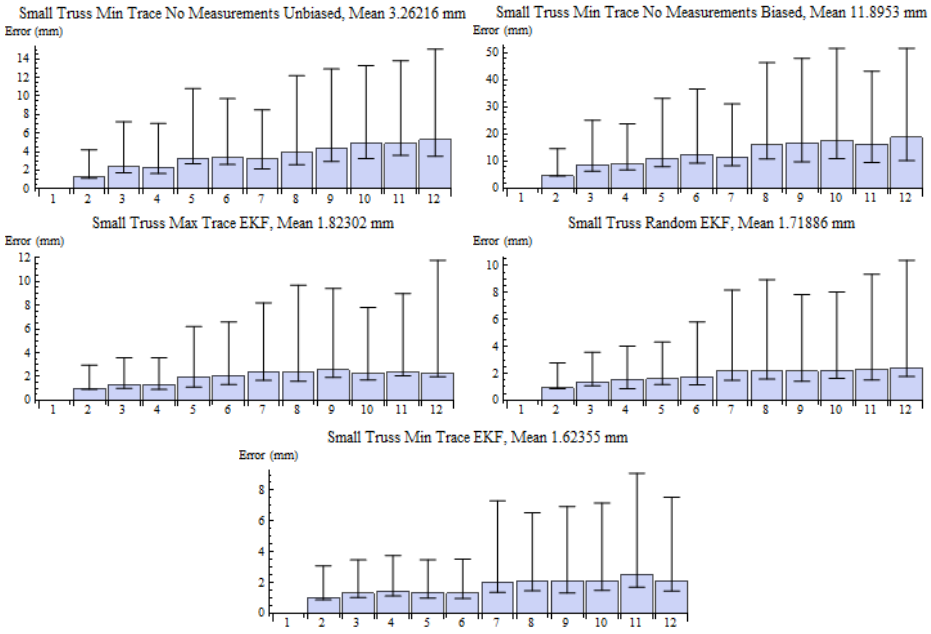

Fig. 5. Results of 100 simulated assemblies of the small truss. In reading order, starting at top left: the perfect calibration case without measurements, the hidden bias case without measurements, the maximum trace case with the EKF algorithm, the random case with the EKF algorithm, and the minimum trace case with the EKF algorithm. The x-axis indicates the node order placement. Error bars represent the 5th and 95th percentiles.

The results of simulations of the 84-node, 315-strut structure are shown in Figure 4. Each figure shows the errors of the nodes from the nominal in the order in which they were placed. The error bars represent the 5th and the 95th percentiles: 90 percent of all results were between the bounds. The optimistic, unbiased IPJR case without measurements can be expected to produce average errors of $6.1mm$. The more realistic case with hidden biases should have average deviations of $22.6mm$. In both control cases, the error grows over time. These simulations show that the EKF assembly algorithm with measurements and heuristics performed considerably better than the biased case, and the quality of performance ranks as expected: the mean error of the minimum trace assembly sequences was $1.8mm$, the random case $3.2mm$, and the maximum trace case $7.2mm$.

The results of the smaller truss structure are shown in Figure 5, following the same format as in Figure 4. We observe two trends: the control cases are worse than the EKF assembly algorithm cases, as expected, but the differences between the assembly heuristics is not significant when the wide variability of

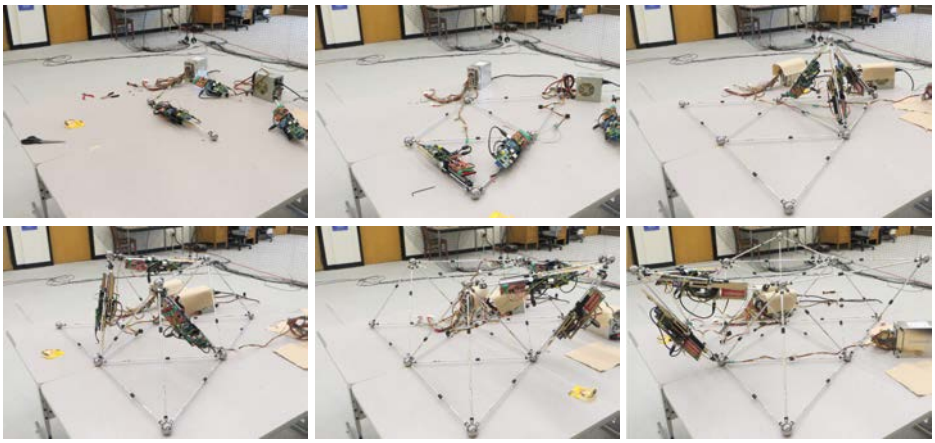

Fig. 6. In reading order, the first nodes placed requires only one IPJR, the other grounded nodes require 2 IPJRs, and the subsequent nodes require 3 IPJRs to be placed. The final frame shows the finished assembly, prior to the removal of the IPJRs.

simulation results is considered in the 5th and 95th percentiles. A perfectly calibrated IPJR had to contend with a growing error over time, and the mean node error is $3.3mm$. In the more realistic case with hidden biases, the mean node error is $11.9mm$, again with increasing errors. More importantly, when the 5th percentile errors of *each node* were averaged, the error was $4.0mm$: even when assuming low errors are correlated in such an ideal trial, at most 5 percent of trials will have a better mean error.

The EKF assembly algorithm mean node errors followed the predicted trends with 100 trials: minimum trace ($1.7mm$) is better than random choice ($1.8mm$), which in turn is better than maximum trace ($1.9mm$). No case violated the triangle inequality. However, all of these results fall within the 5th-95th percentile range, so these results are not significant. For a structure of only 12 struts and 30 nodes, with an insufficient number of opportunities to choose poorly, this is not surprising. Therefore, for the physical experiments, we chose to build only the minimum trace case.

5.2 Physical Experiments

To test the validity of the EKF algorithm on real hardware, we performed two assemblies of the small truss with the IPJRs and the aluminum-strut, steel-node structure (Figure 6), using the minimum trace heuristic. To collect measurements each step, in lieu of an on-board laser distance sensor planned for the future, the human operator measured the distance between node balls. Otherwise, the experiments progressed exactly in the same manner as the simulations.

Each assembled node required 1-3 attempts to converge to the desired tolerance. Each attempt partially corrected for the hidden biases due to the vari-

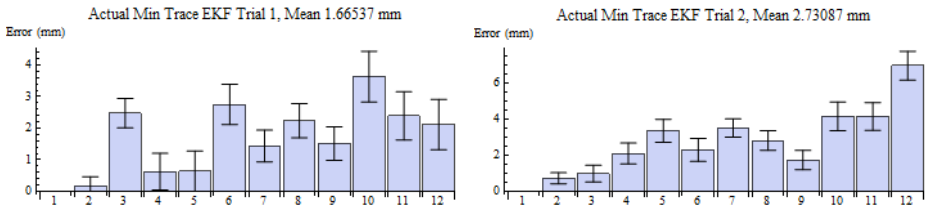

Fig. 7. The estimated errors of the two physical experiments, bounded by the MLE confidence interval.

ous factors affecting the parameters of the IPJRs: imprecise connections to the aluminum tubes, bent tubes, IPJR extension and contraction hysteresis, and deflection due to gravity.

The results of the two trials, bounded by the 95th percentile accuracy error of the maximum likelihood estimator, are shown in Figure 7. The mean errors are $1.7mm$ and $2.7mm$, with a confidence interval of $1.2mm$. This is a significant improvement over the realistic open loop assembly case with hidden biases. The second trial has the largest individual node error at $7mm$, which is better than the open loop case could achieve in less than 5 percent of the trials.

6 Discussion

The simulations show that the minimum trace assembly choice minimizes growing error in both truss structures, despite relying only on local measurements increasingly further from the origin. The data show that a suboptimal build sequence with EKF should do better on average than the open loop case: but only when the assembly trials finish. Half of the large truss maximum trace simulations failed, and three of the large truss random simulations failed. If the $CalcNode()$ function failed to find a solution to the attachment node, the triangle inequality was violated: the IPJRs could not connect with the desired lengths. This algorithm did not implement backtracking, so such cases terimated.

The two physical assembly experiments with the EKF assembly algorithm performed far better than the open-loop control cases, even with the conservative MLE confidence interval of $1.2mm$. Considering that the struts and IPJRs were not made equally, this shows that precise assembly can be made possible by lower quality components through error detection and correction. The second physical trial shows a trend of growing error: this is attributed to two unlocked struts not sliding as expected, deforming the bottom layer. We believe that the error growth would have stopped had there been more nodes to add. During the experiments, the IPJRs occasionally rebooted, and struts broke, but the trials never failed. One IPJR suffered a voltage error and was incapacitated. Since we had two spare IPJRs, the experiments were not hindered. The accuracy and ability to continue despite hardware failure justifies the use of simple robots when assembly requires precision and accuracy.

7 Conclusion

We proposed an EKF-based algorithm that measures errors, corrects future attachments based on the errors, and chooses an assembly sequence to minimize the mean error. We argued that robotic assembly without active state monitoring, while common in the literature, is inappropriate for meeting precision requirements on structures made of stock materials. An EKF-based algorithm can be used to prevent errors from growing with each new addition, and can overcome biases due to imperfect calibration, hysteresis in the IPJRs, deflections from stresses, and inconsistent connections between the IPJRs and the struts on the structure. Simulations showed that using the EKF-algorithm is a significant improvement over not measuring and adjusting for errors, and that choosing a build sequence that adds on to the most certain locations to the structure will perform better than other sequences. Finally, we validated the proposed algorithm using three IPJRs to assemble a truss structure, consisting of 12 steel nodes and 30 aluminum struts, and showed that the EKF algorithm produced structures with $1.6mm$ and $2.6mm$ average node error.

We intend to continue using these IPJR prototypes and truss prototypes. We will thoroughly analyze the comparisons between the EKF algorithm and the open loop assembly algorithm, determine when and why 50 maximum trace and 3 random order trials failed, find the limit of the applicability of the Extended Kalman Filter, and analyze the effects of loop closure. We grouped together sources of error such as gravity, IPJR attachment error, imprecise struts, and thermal expansion under the label of "hidden biases", but we intend to model each of these explicitly, incorporating work done in [16]. These experiments chose assembly sites greedily, but offline planning, ranging from looking ahead several plies to finding a full sequence, may do better. We plan to implement backtracking if a failure occurs, permitting all maximum trace and random assemblies to finish. For struts with interchangeable locked and free-sliding states, we will explore allowing IPJRs to visit struts repeatedly. The MLE worked well for verification, and we will test whether it outperforms EKF in the assembly algorithm. Finally, to demonstrate the concept that a group of cheap IPJRs working in parallel can build suitably accurate truss structures, we intend to build several more IPJRs, modify the algorithm to allow concurrent assembly, and incorporate robotic manipulators.

Acknowledgments

This work was supported by a NASA Office of the Chief Technologist's Space Technology Research Fellowship.

References

1. Zimpfer, D., Kachmar, P., Tuohy, S.: Autonomous rendezvous, capture and in-space assembly: past, present and future. In: 1st Space Exploration Conference: Continuing the Voyage of Discovery. Volume 1. (2005) 234–245

2. Watson, J.J., Collins, T.J., Bush, H.G.: Construction of large space structures at NASA Langley Research Center. In: IEEE Aerospace Conference. (2002)
3. Dorsey, J.T., Doggett, W., Komendera, E., Correll, N., Hafley, R., King, B.D.: An efficient and versatile means for assembling and manufacturing systems in space. In: Proceedings of the AIAA SPACE Conference. (2012)
4. Lillie, C.F.: On-orbit assembly and servicing of future space observatories. Proceedings of SPIE **6265**(62652D-1) (2006)
5. Doggett, W.: Robotic assembly of truss structures for space systems and future research plans. In: Aerospace Conference Proceedings, 2002. IEEE. Volume 7., IEEE (2002)
6. Komendera, E., Reishus, D., Dorsey, J.T., Doggett, W.R., Correll, N.: Precise truss assembly using commodity parts and low precision welding. Intelligent Service Robotics **7**(2) (2014) 93–102
7. Komendera, E., Dorsey, J.T., Doggett, W.R., Correll, N.: Truss assembly and welding by intelligent precision jigging robots. In: Proceedings of the 6th Annual IEEE Int. Conf. on Technologies for Practical Robot Applications (TEPRA). (2014)
8. Homem de Mello, L.S., Lee, S.: Computer Aided Mechanical Assembly Planning. Springer (1991)
9. DeFazio, T., Whitney, D.: Simplified generation of all mechanical assembly sequences. IEEE Journal of Robotics and Automation **RA-3**(6) (1987) 640–658
10. Röhrdanz, F., Mosemann, H., Wahl, F.M.: A high level system for generating, representing and evaluating assembly sequences. In: In Proceedings of the International Joint Symposia on Intelligence and Systems. (1996)
11. Wilson, R.H.: On Geometric Assembly Planning. PhD thesis, Stanford University (1992)
12. Fox, B.R., Kempf, K.G.: Opportunisitic scheduling for robotic assembly. In: Proceedings of the International Conference on Robotics and Automation. (1985)
13. Simmons, R., Singh, S., Hershberger, D., Ramos, J., Smith, T.: First results in the coordination of heterogeneous robots for large-scale assembly. In: Experimental Robotics VII. Springer (2001) 323–332
14. Stroupe, A., Huntsberger, T., Okon, A., Aghazarian, H.: Precision manipulation with cooperative robots. In Parker, L., Schneider, F., Schultz, A., eds.: Multi-Robot Systems: From Swarms to Intelligent Automata. (2005)
15. Heger, F.W.: Assembly Planning in Constrained Environments: Building Structures with Multiple Mobile Robots. PhD thesis, Carnegie Mellon University (2010)
16. McEvoy, M.A., Komendera, E., Correll, N.: Assembly path planning for stable robotic construction. In: Proceedings of the Sixth Annual IEEE International Conference on Technologies for Practical Robot Applications. (2014)
17. Lindsey, Q., Kumar, V.: Distributed construction of truss structures. In: Algorithmic Foundations of Robotics X. Springer (2013) 209–225
18. Detweiler, C., Vona, M., Yoon, Y., Yun, S., Rus, D.: Self-assembling mobile linkages. Robotics & Automation Magazine, IEEE **14**(4) (2007)
19. Werfel, J., Petersen, K., Nagpal, R.: Designing collective behavior in a termite-inspired robot construction team. Science **343**(6172) (2014) 754–758
20. Knepper, R.A., Layton, T., Romanishin, J., Rus, D.: Ikeabot: An autonomous multi-robot coordinated furniture assembly system. In: Robotics and Automation (ICRA), 2013 IEEE International Conference on, IEEE (2013) 855–862
21. Worcester, J., Lakaemper, R., Hsieh, M.y.A.: 3-dimensional tiling for distributed assembly by robot teams. In: Proceedings of the 13th International Symposium on Experimental Robotics (ISER2012), Springer (2012) 143–154