NIST Technical Note
NIST TN 2258

Design and Implementation of a Closed-Loop Mobile Manipulator Control System

Omar Aboul-Enein
Ya-Shian Li-Baboud
Roger Bostelman
Ann Virts

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2258

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

NIST Technical Note
NIST TN 2258

Design and Implementation of a Closed-Loop Mobile Manipulator Control System

Omar Aboul-Enein
Intelligent Systems Division
Engineering Laboratory

Ya-Shian Li-Baboud
Software and Systems Division
Information Technology Laboratory

Roger Bostelman
Smart HLPR, LLC
Frederick, MD

Ann Virts
Intelligent Systems Division
Engineering Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2258

July 2023

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Under Secretary of Commerce for Standards and Technology

NIST Technical Series Policies
Copyright, Fair Use, and Licensing Statements
NIST Technical Series Publication Identifier Syntax

Publication History
Approved by the NIST Editorial Review Board on 2023-07-27

How to cite this NIST Technical Series Publication:
Omar Aboul-Enein, Ya-Shian Li-Baboud, Roger Bostelman, Ann Virts (2023) Design and Implementation of a Closed-Loop Mobile Manipulator Control System. (National Institute of Standards and Technology, Gaithersburg, MD), NIST TN 2258. https://doi.org/10.6028/NIST.TN.2258

NIST Author ORCID iDs
Omar Aboul-Enein: 0000-0002-5361-4589
Ya-Shian Li-Baboud: 0000-0003-3234-4345
Roger Bostelman: 0000-0002-8605-7758
Ann Virts: 0009-0003-3420-4888

Contact Information
omar.aboul-enein@nist.gov, 100 Bureau Dr. MS 8230, Gaithersburg, MD 20899, 301-975-2703

Abstract

Mobile manipulators, which integrate a robotic manipulator with an automatic–autonomous mobile base, have the potential to augment automation by combining the capability of navigation with complex manipulation to support unstructured and dynamic environments. Targeted applications include manufacturing large-scale parts commonly encountered in the aerospace, energy, shipbuilding, and transportation sectors. While autonomous mobility no longer restricts the robotic manipulator to working at a single, rigidly fixtured workstation or work-piece, the increased flexibility introduces new sources of position and orientation uncertainty, and manufacturing processes of large-scale parts with complex, curved surfaces require high repeatability and accuracy. A standardized measurement methodology, including a configurable measurement artifact to simulate dynamic manufacturing operations, is being developed to identify and evaluate these sources of performance uncertainty. As part of this methodology and associated test methods development, this work details the design and prototype implementation of a closed-loop mobile manipulator control system that integrates feedback from an optical tracking system. Development of the control system presents a new test implementation to demonstrate the performance evaluation of mobile manipulation performance in application scenarios where the workstation or work-piece is physically disturbed during operation or where rapid registration between distant locations along the same work-piece is required. This work will promote advances in control scheme development by providing a standardized, reproducible test method for evaluation in a variety of simulated application spaces.

Keywords

Closed-loop control system; configurable mobile manipulator apparatus; coordinate registration; ground truth; large-scale manufacturing; mobile manipulators; mobile robot; optical tracking system; re-configurable mobile manipulator artifact; robotics in manufacturing; systems integration.

Table of Contents

List of Tables

List of Figures

Acknowledgments

The authors thank the following individuals for their contributions to this project. Dr. Jeremy Marvel and Dr. Kamel Saidi of the Intelligent Systems Division at NIST, for their technical consultation on the manipulator controller hardware and sharing of ASTM E3124-17 test artifacts, documentation, and code, respectively. Dr. Vinh Nguyen, Department of Mechanical Engineering-Engineering Mechanics at Michigan Tech for his sharing of sample velocity-based manipulator control code. Dr. Jennifer Case, Fermi National Accelerator Laboratory managed by Fermi Research Alliance, LLC for her consultation on PID controller design and tuning. Samuel Reed-Weidner, Omron Robotics and Safety Technologies, for his technical consultation on the cart transporter hardware and software, which included information on the application layer cycle time of the controller. Dr. Soocheol Yoon of Georgetown University and the Intelligent Systems Division at NIST for his consultation on mobile manipulator capabilities and assistance with the data collection for the updated ASTM E3064-16 evaluation of the OTS measurement uncertainty.

1. Introduction

Mobile manipulators are an integrated system consisting of a robotic arm mounted to an autonomous mobile base. These systems have the potential to navigate and perform a variety of automated tasks across an industrial environment (e.g., composite task of material delivery and machine tool tending) [1–3]. An important subset of such applications also includes large-scale manufacturing. Examples of large-scale components are common in the aerospace, energy, shipbuilding, and transportation sectors and can include, but are not limited to, wind turbine blades, aircraft wings, high speed train bodies, ship bows, and towers [1, 4–8]. Tasks may include prototyping, assembling, surfacing, or otherwise processing (i.e., drilling, boring, polishing, sanding, deburring, painting and coating removal) such work-pieces [1, 5, 7, 8]. Furthermore, utilization of mobile manipulators to permit expanded build volumes for large-scale conformal Additive Manufacturing (AM) has been suggested and implemented for experimentation with varying robot planning, control, and localization approaches [9–15]. Note that these implementations have included both instances where the mobile base stops while printing and where continuous/simultaneous robot movement and printing was implemented [10–15].

In order to leverage the agility and flexibility of mobile manipulators for such tasks, the key limitation of increased performance uncertainty must be overcome. In the case of tending Computer Numerical Control (CNC) machines and conformal AM, localization performance in unstructured environments and insufficient pose accuracy/repeatability in the mobility component have been cited as challenges preventing the adoption of mobile manipulators for such tasks [2, 9–13, 15]. Furthermore, the complex, curved shapes associated with many of the aforementioned large scale components also require a high degree of manufacturing accuracy [4–6].

To overcome these challenges, the National Institute of Standards and Technology (NIST) is developing performance test methods to help identify and quantify sources of mobile manipulator performance uncertainty, and to aid in the development of robot hardware and software designs, control algorithms, planning algorithms, and localization methods. Additionally, these test methods can be used to verify the suitability of a mobile manipulator platform to their specific manufacturing application domain and environment. Towards this goal, NIST has developed a set of re-configurable, cost-efficient test artifacts, called the Configurable Mobile Manipulator Apparatus (CMMA) (previously referred to as the Re-configurable Mobile Manipulator Artifact (RMMA))[1] [16–19]. The CMMA, outfitted with machined configurations of aluminum surfaces and tapped holes, can be used as a mock workspace or work piece to simulate a variety of potential mobile manipulator manufacturing capabilities.

Mobile manipulator capabilities can be broadly classified under non-continuous task performance and continuous task performance. During non-continuous tasks, the mobile robot

[1] From Sec. 3.3 onward, this acronym refers specifically to the Type B CMMA unless otherwise noted.

transports the manipulator between one or more task locations, but the two are not in simultaneous motion while physically connected. Continuous task performance consists of both mobile robot and manipulator moving continuously and simultaneously while the two are physically connected. Non-continuous tasks can be further sub-divided into static and indexed tasks. In static tasks, the mobile base repeatedly docks the manipulator at a single intended pose. However, for indexed tasks, the mobile base docks the manipulator in sequence amongst a finite set of intended poses. In past work, two variants of the CMMA were developed and used for evaluations: The Type A CMMA (formerly called the static RMMA) represented non-curved and smaller work-pieces, while the Type B CMMA served as a more direct analogue to large-scale components with complex shapes [19, 20].

The Type A CMMA has been used to measure the static and indexed performance of an Automatic Guided Vehicle (AGV) mobile manipulator [20–24], the indexed performance of an Autonomous Mobile Robot (AMR) mobile manipulator [25], and the indexed performance of a mobile manipulator-on-a-cart, in which an Autonomous Mobile Robot Cart Transporter (AMR-CT) coordinated with a manipulator mounted on a detachable cart [26]. While the AGV mobile manipulator was limited to following pre-programmed paths, the AMR and AMR-CT mobile manipulator systems were capable of performing their own path-planning, including re-computing paths around unanticipated obstacles. Furthermore, the AMR-CT mobile manipulator could allow for increased hardware utilization (e.g., allowing the AMR-CT to tend other carts and/or payloads) and concurrent workflows. In addition to evaluating mobile robot docking uncertainty and manipulator position accuracy/repeatability, the CMMA can also be used to compare the performance of coordinate registration methods and algorithms, as demonstrated with various laser-based and visual marker-based coordinate registration test implementations [23, 26].

The continuous performance scenario, which was evaluated using the Type B CMMA had been previously conducted in simulation environments and evaluated on a prototype mobile manipulator platform [27, 28]. The most recent developments in measuring continuous mobile manipulator performance featured integration of the previously developed algorithms and simulations with the mobile manipulator-on-a-cart, which used a commercially oriented mobile platform and manipulator [29]. A more comprehensive summary of previous development milestones and evaluation activities can be found in Ref. [19].

A major component towards past development and evaluation of the CMMA as a test artifact was to independently validate the measurements obtained solely using the CMMA and mobile manipulator controller logs against ground-truth reference measurements. For this purpose, the rigid body tracking of a motion capture Optical Tracking System (OTS) was used as the ground truth measurement system. The measurement uncertainty of the OTS was previously determined to be 0.022 mm (positional) and 0.00040143 rad (angular) for tracking stationary rigid bodies [20, 30]. Likewise, the tracking uncertainty for rigid bodies in motion was 0.26 mm (positional) and 0.0017453 rad (angular) [20, 30]. Here, the OTS was manually-controlled by an operator to passively record the position and orientation (pose) of rigid bodies that were placed on the manipulator End-of-Arm Tool (EOAT), the

manipulator base, or the frame of the mobile base, for comparison to either the pose of the CMMA itself or individual assembly target locations on the CMMA.

However, some mobile manipulator capabilities have remained previously untested using the CMMA. Specifically, the capabilities in Table 1 lie within the gap between non-continuous tasks and continuous tasks. The capabilities are focused on scenarios in which the work-piece may be moved or disturbed. Note that perturbations may be small (i.e., 1 m or less) or large distances (e.g., greater than 1 m). For example, small work-piece perturbations that could occur in a real manufacturing environment might include "bumps" or "nudges" accidentally induced by contact forces exerted by humans or other robots, as well as floor vibrations induced by passing heavy equipment, such as manually controlled forklifts or AGVs. In this scenario, the mobile base of the mobile manipulator should correct for this error in its navigation goal if the mobile manipulator has not yet reached the work-piece. Alternatively, if already docked at the work-piece, the manipulator should rapidly correct for this small error in its manufacturing task target location. The CMMA evaluation of the laser-based edge rapid coordinate registration method from Ref. [26] only partially represented this scenario because the registration method took between 15 s and 23 s to run. Therefore, multiple or continuous small perturbations after initial registration could not be tested using this implementation.

Rapid mobile manipulator registration across large distances is typical when dealing with large-scale components or when more significant perturbations to the CMMA pose occur, such as needing to temporarily move the work-piece to a different location due to limited space. In this scenario, again the mobile base should correct for this error if it occurs before docking. However, if the perturbation occurs after docking, then the manipulator should stow and the mobile base should transport the manipulator to the new docking location near the work-piece. To create a test implementation satisfying the capabilities in Table 1, a faster closed-loop control system is required to allow for rapid, continuous mobile manipulator pose feedback. Furthermore, in past work, the CMMA was not tested at a tilt, even though it supports this configuration. Therefore, full 6 Degrees of Freedom (DoF) registration capabilities should be tested alongside the previous 3 DoF registration capabilities, as shown in Table 1.

To demonstrate performance evaluation of these previously untested mobile manipulator capabilities, this work focuses on the design and prototyping of a new NIST mobile manipulator test implementation that integrates closed-loop control using pose feedback from the OTS. This control system, when used after docking to register the mobile manipulator to the CMMA, can serve as a reference of comparison with or for the possible development of alternative coordinate registration methods that use on-board sensing in the future. For example, use of the OTS is expected to be faster and more accurate than the prior laser-based coordinate registration methods (i.e., spiral search, bisection, edge detection, and stochastic Kalman filters) or provide a ground-truth reference to compare with closed-loop feedback

from an on-board vision system[2].

Table 1. CMMA task scenarios for mobile manipulator/OTS closed-loop control testing.

Scenario	Registration DoF[a]	CMMA Perturbation	Capability
1	3	Small ($<$ 1m)[b]	AMR-CT corrects for small CMMA perturbation.
2	3	Large ($\geq$ 1m)[b]	AMR-CT corrects for large CMMA perturbation.
3	3	Small ($\leq$manipulator reach)[c]	manipulator corrects for small CMMA perturbation.
4	3	Large ($>$manipulator reach)[c]	manipulator stows, AMR-CT transports manipulator to new CMMA docking location.
5	6	Small ($\leq$manipulator reach)[c]	manipulator corrects for small CMMA perturbation.
6	6	Large ($>$manipulator reach)[c]	manipulator stows, AMR-CT transports manipulator to new CMMA docking location.

[a] Degrees of Freedom [b] Occurring before mobile manipulator is docked with CMMA.
[c] Occurring after mobile manipulator is docked with CMMA.

The remainder of this paper is organized as follows. In Sec. 2, a summary of recent literature is presented to foster understanding of past examples, including capabilities, limitations, and potential challenges in using an OTS or other measurement instruments in pose update feedback for closed-loop mobile manipulator control. In Sec. 3 the hardware and software specifications of the prototype test implementation, including the cart transporter, manipulator-on-a-cart, CMMA, OTS, and network architecture are documented. Section 4 outlines the coordinate systems, transformations, and registration procedures required for the closed-loop control system, and Sec. 5 describes the design of the closed loop controllers for the manipulator and the AMR-CT, respectively. An evaluation of the time synchronization delay variability is also provided. The project resulted in the implementation of the closed-loop mobile manipulator controllers as a prototype to satisfy the capabilities in Table 1, though experimental design and performance evaluation of the final test implementation is left to future work.

[2]For example, a deep-learning based model for use with an on-board camera was developed as part of NIST grant 70NANB18H259: "Performance Measurement of Mobile Manipulators using Coarse-to-Fine Deep Learning Methods"

2. Background

A review of existing literature conducted prior to this work uncovered a number of examples where OTS feedback was integrated for mobile manipulator control. Additionally, methods of control system integration using alternative external measurement systems were reviewed. In summary, the following observations were derived from the literature review. First, the need to ensure that the NIST mobile manipulator performance measurement framework could accommodate mobile manipulators that integrate external metrology systems for registration was established, as their use in academia was found to be prominent when testing new control and planning strategies. Second, was that the limitations, challenges, and successes of the reviewed works informed and provided basis for the development of the NIST closed-loop controller as a test implementation. Finally, the intended application domains of these works further demonstrated the gap in standardized evaluation of mobile manipulator performance capabilities that lie between non-continuous and continuous (i.e., work-piece disturbances), as many of the reviewed works either focused on developing control or planning advances for either the former or the latter.

Reference [5, 6] identified Indoor Global Positioning System (iGPS) (see also Refs. [31–34]), laser trackers, and laser tracers as systems that could be applied to mobile manipulators for large-scale manufacturing scenarios to improve pose accuracy. Laser occlusion, linear error (in the case of laser trackers), and limited measurement DoF were identified as potential challenges for such solutions [5, 6]. Furthermore, Refs. [1, 35–39] provide a specific example where feedback from a 6 DoF iGPS was applied to the end-effector of an AGV and industrial robot arm mobile manipulator to achieve closed loop control for laser paint removal of aircraft wings. However, the accuracy requirement of this application was 254 mm and it has been observed that, in comparing previous experiments with the CMMA, mobile manipulators that use AMR bases can have more docking uncertainty than those that use AGV bases [25, 26, 39].

Focus now turns more specifically to the use of OTSs for mobile manipulator control. Reference [40], which evaluated the ability of an aerial manipulator to pick and place an 0.11 m long, 0.02 m diameter cylinder, used an OTS as a substitute for Global Positioning System (GPS) localization when operating indoors. However, the manipulator was operator controlled and the application, more reflective of scientific and emergency response tasks, required a less stringent performance tolerances than would be needed for precision manufacturing.

In Ref. [41], an OTS was utilized to provide localization as part of a reactive closed-loop control scheme for legged mobile manipulator motion planning. Here, a dedicated manipulator was not used since the work focused on allowing the robot to use its default appendages (also used for mobility) to rearrange stools. Additionally, since the planning problem was the main focus of the work, the application domain did not appear to reflect the high-accuracy and precision manufacturing processes that are the focus of this report.

In Ref. [42], a sensor fusion approach combining measurements from an inertial measurement unit (IMU) (as the "fast" sensor operating at 500 Hz) and a 12-camera OTS (as the "slower" sensor operating between 30 Hz and 240 Hz) using an Extended Kalman filter was evaluated to determine if they could replace the robot forward kinematics (FK) algorithms for aerospace manufacturing. Since the work was focused on evaluating the tracking solution, the experiment relied primarily on manual jogging of the manipulator, and an offset of 15 mm between robot FK and the sensor fusion pose estimate was observed. In the paper, this offset was attributed to several possible errors in the hand-eye calibration between the IMU, OTS, and world coordinate frames, including reflections/occlusions impacting the OTS, insufficient mounting of the OTS cameras, and in misalignment in the IMU mounting, among others.

In Ref. [43], case studies were conducted in simulation and with physical robots to develop and evaluate distributed software architectures for multi-robot cooperation between a mobile manipulator and a mobile robot. For the physical robots, OTS rigid body tracking was used to feed robot and work piece positions to facilitate transferring the part from one robot to another (with the mobile robot being controlled via teleoperation). In this work, the lack of real-time communication over a wireless network was cited as a major obstacle in combining the two robots into one real-time context. Therefore, each robot was instead treated as its own system with separate real-time contexts. The case study was extended in Ref. [44] to encompass the multi-robot cooperation of two mobile manipulators to transfer a baton, and the OTS was used primarily to assist in navigating to the handover region and motion synchronization between the two robots. Since the software architecture was the main focus, evaluation of the system task performance was limited to whether or not the handover was accomplished, whereas the CMMA, as a test artifact, can provide a standardized method to evaluate quantitative assembly performance in addition to pass/fail metrics [19].

Reference [45] presented a "hierarchical inverse dynamics control scheme" with pre-defined time convergence and model uncertainty robustness guarantees. The control scheme was tested on a similar multi-robot pick-and-place task using two torque-controlled mobile manipulators. In this work, OTS feedback was used to provide global coordinates of the manipulator end-effector, mobile base, work piece, and obstacle locations, as well as to compute errors accounted for in the closed-loop control system. In the work, it was noted that the assumption of a horizontal, flat surface with no wheel slippage (among other assumptions) was made for the model of the mobile manipulator representing the relationship between the actuator torques and generalized forces and torques. Prior experiments with the mobile manipulator-on-a-cart demonstrated an environment that may not meet such assumptions, which can introduce additional performance uncertainty [26]. Therefore, usage of the CMMA could bolster further advances in control scheme development by allowing for a standardized, reproducible test method for evaluation in a variety of environmental conditions. Additionally, it should be noted that, in Refs [43–45], an educational, research based mobile manipulator platform was tested [46], whereas the mobile manipulator to be

used in this work represents both a different type of mobile manipulator system (a mobile manipulator-on-a-cart) and one that uses a commercially-targeted mobile base (see Sec. 3.1).

In Refs. [10, 15], an OTS was integrated into a closed-loop mobile manipulator controller as part of research developing path planning algorithms and a model predictive controller (MPC) for print-in-motion AM. The OTS, which operated at a rate of 120 Hz, was initially used as a placeholder to provide localization input to a synchronous drive controller for the mobile base and to the manipulator trajectory planner for motion compensation/reachability detection. This integration was intended to isolate the controller evaluation from localization performance. However, the OTS print extruder was also tracked in order to determine the print trajectory error. As print-in-motion was tested, the application can be more closely classified under continuous mobile manipulation capabilities, though the experiments with this system also simulated mobile base disturbance via a sinusoidal motion of consistent amplitude, but varying frequency [10]. It was shown that motion compensation performance was impacted by disturbance oscillation frequency (and by extension the mobile base velocity), however comparisons of several print-in-motion trajectories against a static spiral trajectory revealed the same error bound of at most 15 mm. Later experimentation with the developed Task-Consistent Path Planning (TCPP) algorithm also saw the printing performance compared when using the OTS versus laser-base simultaneous localization and mapping (SLAM) algorithms for localization [15]. The extruder accuracy was found to be nominally between 6 mm and 8 mm when using the OTS.

In Ref. [47], a bi-level motion optimization algorithm, using Sequential Quadratic Programming (SQP) and Quadratic Programming (QP) for real-time trajectory generation and a learning-based controller for high-speed motion tracking was applied to a mobile manipulator to dynamically catch a ball using contionuous mobile manipulator capabilities. Such a demonstration has stringent mobile manipulator timing and coordination requirements, but may not translate as apparently to a manufacturing use case. For this, an OTS was used to track only the position of a retro-reflective ball, which was then fed into a Kalman filter for trajectory prediction. The trajectory prediction ultimately operated at a frequency of 100 Hz, the planning operated at 30 Hz, and the Deep Neural Network (DNN) operated at 20 Hz to serve as control input for the mobile base controller (which could operate at a max of 100 Hz) and manipulator (which could operate at a max of 125 Hz). In experiments of the complete bi-level optimization scheme, the mobile manipulator was able to catch the ball with 85.33% accuracy. For the failures unrelated to planning that occurred, insufficient ball trajectory prediction and tracking errors due to aggressive maneuvers were cited.

In Ref. [48], an OTS was integrated into a closed-loop control system designed for a nonholonomic mobile parallel manipulator intended for automated production line transport and object positioning tasks. In this case, the OTS, which had a max frame rate of 240 Hz and system latency of 2 ms, was used to track each of the two mobile carts, as well as the platform positioned via a closed-kinematic chain manipulator. The OTS data was transmitted wirelessly between the OTS, a remote computer, and the carts. Again, Kalman filtering

was used to reduce noise and improve tracking accuracy and commands could be generated and forwarded to the mobile cart mini computers, running an Proportional/Integral (P/I) controller, at a rate of 10 Hz. According to the paper, the average component-wise pose errors, at 95% confidence, observed with the designed mobile parallel manipulator were no greater than 30 mm in position and 0.35 rad in rotation. Note that the pose error for the Z position component and Y rotation component for the fourth trial exhibited the largest error, while the rest of the positional and rotational errors were no greater than 15 mm and 0.2 rad, respectively.

Additionally, Refs. [3, 7, 49] mentioned use of an OTS for mobile base, manipulator and/or work-piece localization, however, since use of the OTS was incidental to the main focus of each respective work, details on the implementation were limited. In particular, Ref. [49] focused on mobile manipulator control strategy refinements for large-scale work-pieces, including a wind-turbine blade as an example, however, the mobile manipulator capabilities were limited to the indexed scenario. In Ref. [3], OTS feedback was used to localize a manipulator for a construction-oriented drilling task, but again evaluation was limited to non-continuous performance and the mobile manipulator featured a research-oriented, custom-designed mobile base[3].

3. Hardware Platform and Software Environment Design

The same mobile manipulator-on-a-cart[4] featured in Ref. [26, 29] and shown in Fig. 1 was used for this work. A more complete description of the specifications for the mobile manipulator-on-a-cart can be found in Ref. [26]. However, in Sec. 3.1 and 3.2, a summary of key changes made to the system since the previous project is provided. One key change was that the control code for the AMR-CT and manipulator-on-a-cart was planned to be run on an embedded System on Module (SoM) compute module with an Input/Output (I/O) board, which was mounted to a cart payload structure in place of the previously used personal computer (PC) [50, 51]. The compute module had a four-core, 64-bit processor running at a clock rate of 1.5 GHz and was configured with 8 GB of RAM, 8 GB of eMMC storage, and Wi-Fi support [50]. The Input/Output (I/O) module provided an additional wired Gigabit Ethernet port with support for Power over Ethernet (POE), as well as a real-time clock that was used to assist with synchronization and timing [51]. A 64-bit Linux-based operating system with a real-time kernel patch (see Sec. 3.2) was used. A summary of the specifications for the CMMA and OTS is provided in Sec. 3.3 and 3.4, respectively. Additionally, the marker configuration for the tracked OTS rigid bodies is provided in Sec. 3.5. Finally, the design of the network topology and time synchronization

[3]Note also the PID controller presented in [3] was for control of the mobile base, which appeared to primarily use robot odometry, with OTS feedback listed as an alternative.

[4]Certain commercial equipment, instruments, or materials are identified in this paper in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

implementation is detailed in Sec. 3.6 and depicted in Fig. 2. Note that Fig. 2 also provides a summary of the software libraries used to integrate closed-loop OTS feedback with the SoM.

3.1. Cart Transporter

The full manufacturer specifications for the AMR-CT can be found in Ref. [52], and the software and firmware versions[5] from previous work were not changed [26]. Additionally, it should be noted that the docking repeatability of the AMR-CT was ± 100 mm in position and ± 0.035 rad in orientation according to the manufacturer [52]. In the past work, the AMR-CT was selected as an extension of tests with the AGV and AMR mobile manipulators. In contrast to the AGV, which relied on pre-programmed paths, the AMR-CT had a higher level of autonomous capability for dynamic path planning and obstacle avoidance [20, 26]. Additionally, in contrast to the AMR, the AMR-CT possessed a higher level of task concurrency and hardware utilization, since the AMR-CT could detach from the cart with the on-board manipulator and be used to perform other tasks while the manipulator was occupied [26]. As in previous experiments, the lower front sensor was disabled to prevent erroneous detection of the floor as an obstacle [26]. Additionally, the rear sensor, which was also not needed, was disabled and the maximum height at which the left and right side scanners could detect obstacles was kept at a reduced value (lowered to 250 mm from 1000 mm) [26, 29]. The latter was done previously to prevent the CMMA from erroneously being detected as an obstacle [29]. Under the path planning settings, the free space between the robot base and any obstacles along a planned path was increased from 300 mm to 700 mm to avoid collisions between the AMR-CT side laser scanners and the CMMA.

Since the AMR-CT used laser-based mapping and localization as its primary means of navigation, an environmental map of the lab could be generated in the same fashion as prior work, which followed manufacturer-recommended procedures (i.e., manually driving the AMR-CT through the lab to create scans of static room features) [53]. It should be noted that the coordinate frame origin of the AMR-CT map, which was 2D and used a right-handed coordinate system, was determined upon map creation. Since the location of the AMR-CT map coordinate frame origin was not physically accessible for measurement and no option to change the origin in software was available, a calibration procedure was used to determine the pose between the map and other coordinate systems (further addressed in Sec. 4.4). Additionally, the AMR-CT laser map scans were assumed to contain static room features, and the manufacturer notes that dynamic objects not present in the map can impact localization performance [53]. Therefore, the CMMA, as well as objects previously placed underneath it to assist the AMR-CT with localization, were excluded from the new map scans since the CMMA pose was no longer static.

[5]The AMR-CT software included controller firmware version 1.9.0d, manufacturer software operating system version 4.8.0, automation management software version 4.9.9, and map generation and configuration software version 4.7.7

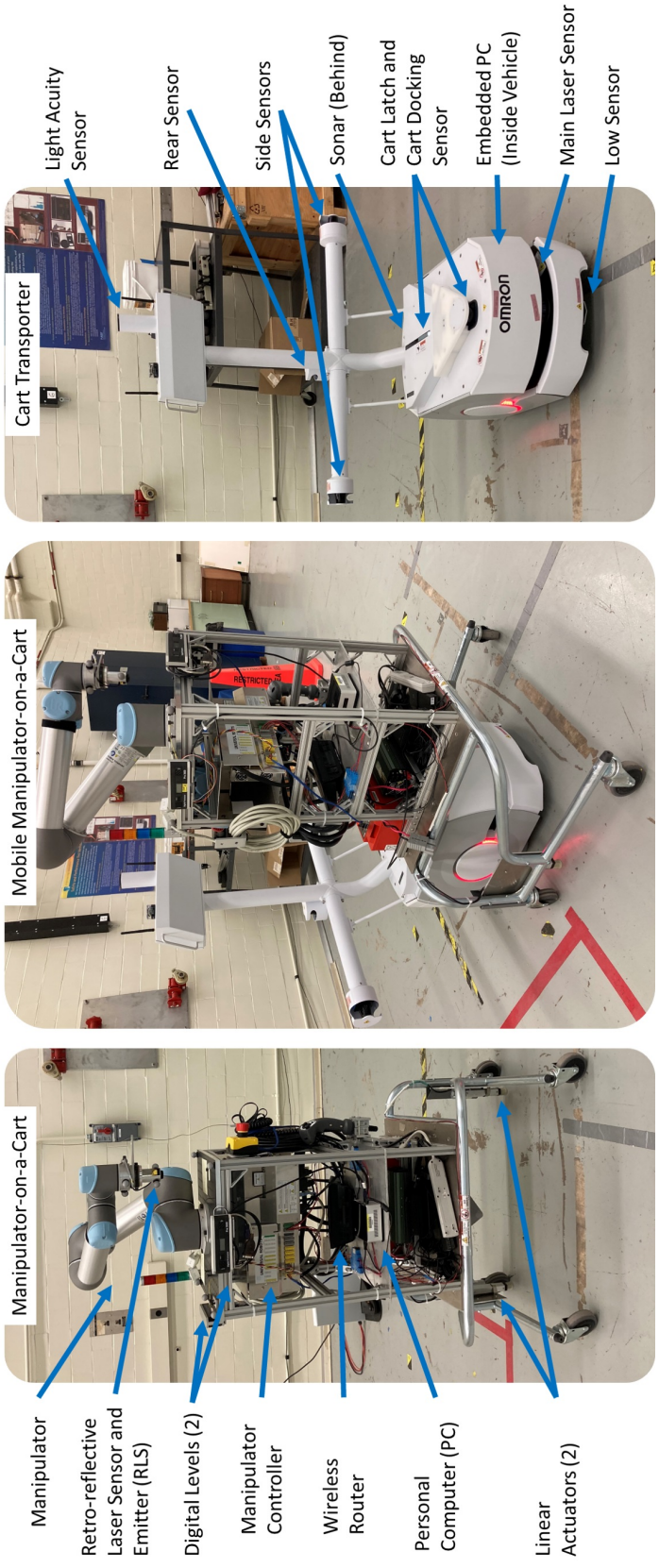

Fig. 1. The manipulator-on-a-cart and hardware components (left). The AMR-CT docking with the manipulator-on-a-cart (center). The AMR-CT (right) [26]

.

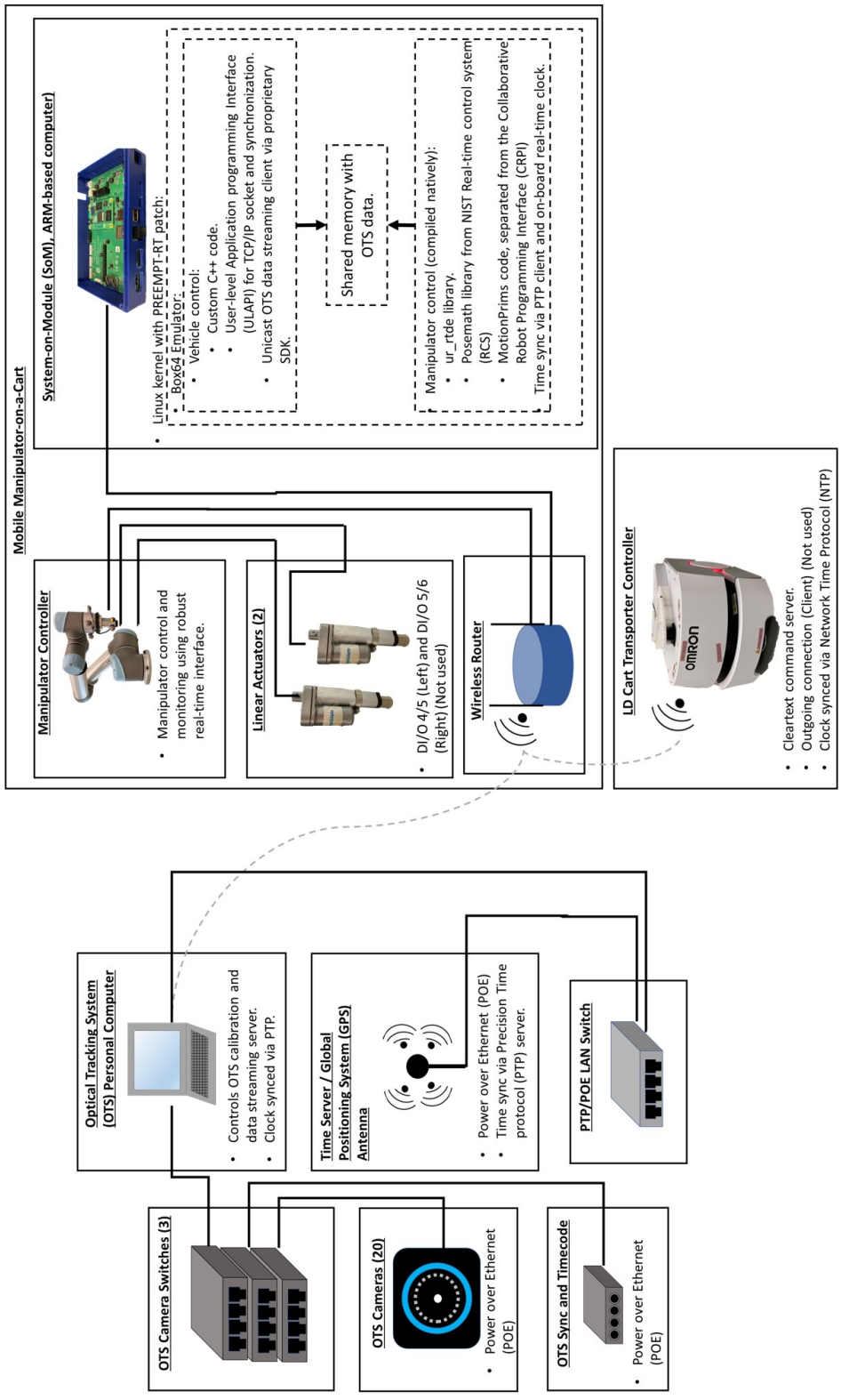

Fig. 2. Network diagram showing OTS integration with the mobile manipulator-on-a-cart and labeled with software libraries needed to implement closed-loop mobile manipulator control. Note the PTP time synchronization with GPS as the primary time reference enabling UTC timestamping of manipulator and AMR-CT position data from OTS for closed-loop control of the mobile manipulator.

Table 2. Dimensions used to refine Light Analysis performance.

Dimension	Measured Value
Light Drawing Height[a]	6089.3 mm
Min Light Height	5175.9 mm
Max Light Height	7002.7 mm
Light Length[b]	1219.2 mm

[a] Only affects map display and not robot operation [54]. [b] Note a multiple of this value may need to be used depending on how many adjacent lights are turned on.

To mitigate the impact of having less static room features on localization performance, an additional type of localization supported by the AMR-CT was used on the prototype test implementation. Specifically, a map of the ceiling lights (using the light acuity sensor depicted in Fig. 1, left) was also created following manufacturer-recommended procedure, since the feature was, according to the manufacturer, intended to improve localization performance in dynamic environments [53, 54]. The steps to setup and use this feature included 1) installation of a firmware update to enable the light acuity sensor features, 2) re-measuring the camera position and tilt to verify and/or set to the correct values, 3) verifying/setting the light analysis parameters to the correct values, and 4) creating the light map. For step 2, the default camera position values were verified to be correct, however the x and y tilt of the camera was re-measured to be 0.014 rad and 0.035 rad, respectively, using the manufacturer-recommended procedure. For step 3, the dimensions listed in Table 2 were used to tweak related key parameters in the robot configuration intended to improve light analysis performance. These parameters were determined as follows: First, nine sample light heights across the lab were measured using a laser range finder, as shown in Table 3. Then, the mean light height was computed and used as the Light Drawing Height (Note: this parameter, according to the manufacturer does not impact robot operation) [54]. The minimum and maximum light height was then derived according to manufacturer recommendation (i.e., $\pm 15\%$ of the mean height) [54]. Finally, the light length was known to use a common length given in Table 2. An additional step was required to improve acuity performance as it was found that multiple adjacent lights in sequence could not be differentiated by the acuity system [54]. Specifically, the fluorescent bulbs of certain lights were removed as shown in Fig. 3. The lights that were selected to be turned off were chosen in an asymmetric pattern.

Table 3. Light Measurements used to refine Light Analysis performance.

	Left	Center	Right
Front	5926.1 mm	6417.5 mm	5927.7 mm
Middle	5922.2 mm	6411.1 mm	5915.0 mm
Back	5915.0 mm	6415.1 mm	5953.9 mm

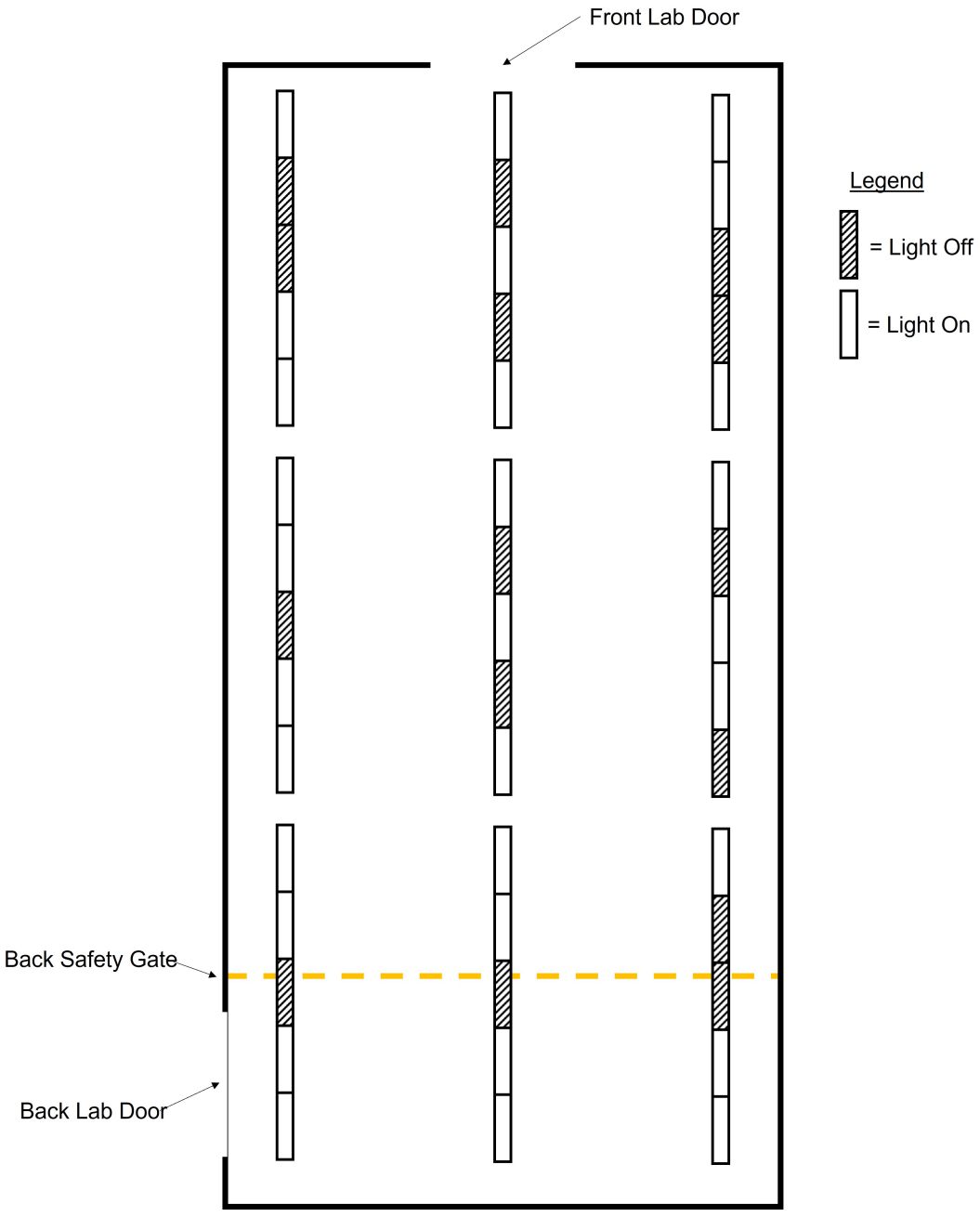

Fig. 3. Diagram of lights disabled in the lab.

The primary means to send commands to and receive feedback from the AMR-CT was a clear-text language interface. As documented in Ref. [26, 53, 55], the interface was exposed through Transmission Control Protocol / Internet Protocol (TCP/IP) network sockets and had two modes of communication. The so-called "Outgoing" connection, previously used only to obtain feedback from the AMR-CT, was not needed since the OTS was solely

used instead for feedback. However, the "command-response mode", in which a client application connected to a server running on the AMR-CT controller, was used for sending pre-defined movement commands and infrequent, individual status queries to the AMR-CT. Upon receiving these commands, the AMR-CT would then execute the given command and forward back the response. In this case, the `goToPoint`, `move`, `deltaHeading`, and `stop` commands were used as the primary movement commands needed to navigate and dock the AMR-CT to the CMMA.

The `goToPoint` command was needed to allow the AMR-CT to use its autonomous path planning to navigate to a point 1 m in front of the CMMA. Then, to allow the AMR-CT to dock close to the CMMA without detecting it as an obstacle, the dead-reckoning `move` command with input parameters that set the distance to 1000 mm, the front clearance to 25 mm, and the speed to 150 mm/s was used [26]. Finally, the `deltaHeading` and `stop` commands were used to allow for heading corrections in a manner similar to that implemented in Ref. [29].

Despite the maximum application layer cycle time of the AMR-CT controller being 10 Hz, and that newly issued commands would preempt existing commands when received by the AMR-CT controller, there did not appear to be a practical way to implement continuous goal updates using the `goToPoint` command and `move` commands. Informal tests were done in which `goToPoint` and `move` commands were issued rapidly to the AMR-CT. However, the AMR-CT was unable to respond fast enough to commands issued at a rate of 5 Hz or faster. This was speculated to be possibly due to the limitations imposed by the time needed to compute path planning (in the case of the `goToPoint` command) and to actuate the wheel motors. While the AMR-CT could move when the commands were implemented at a rate of 2 Hz or slower, movement speed was slow and inconsistently maintained. Therefore, the control strategy for the AMR-CT instead used the OTS to continuously monitor the pose of the CMMA in the OTS coordinate system and issue an updated movement command only if a change in the CMMA pose was detected; a so-called "event-based" update strategy.

As a final note, custom C++ code to interface with the AMR-CT using the clear-text language had been written as part of previous work and was used as a starting point to implement the new AMR-CT control strategy [26]. The custom software used the User-Level Application Programming Interface (ULAPI) for making system-calls, such as those needed for TCP/IP sockets and mutual exclusions (mutexes) [56]. Since ULAPI could not be readily re-compiled for the SoM's central processing unit (CPU) architecture and only event-based goal updates were implemented, the AMR-CT control code was run within an emulator along with the OTS data streaming code (see also Sec. 3.4) [57–59].

3.2. Manipulator-on-a-Cart

The same manipulator from past experiments was used for this work [25, 26]. The manipulator was fixtured to the detachable cart using a custom payload structure as depicted

in Fig. 1 (right). According to the manufacturer, the manipulator had a maximum reach and repeatability of 850 mm and 0.1 mm, respectively [60]. Additionally, the same Retro-reflective Laser Sensor and Emitter (RLS) from prior work was used to digitally detect the retro-reflective mock-assembly targets mounted to the CMMA, which represented peg-in-hole assembly performance (see also Sec. 3.3) [19, 21, 61]. The custom mount for fixturing the RLS to the EOAT was modified with additional threaded M4 screw holes to rigidly attach OTS markers. The technical drawing for the updated mount design is provided in Fig. 22 of Appendix B. Since the 3D orientation of the manipulator cart base was measured using the OTS, the two digital levels, each of which had an accuracy of ± 0.00087266 rad when the level was between 0 rad and 0.17453 rad and a repeatability of ± 0.00087266 rad, were no longer needed during operation [62]. However, the digital levels were utilized as part of an offline coordinate system calibration procedure discussed in Sec. 4.4.

The two linear actuators, which were attached to the cart for the purpose of keeping the cart from rolling after docked with the CMMA, were unchanged from prior work. The full specifications for the actuators and the electronics used to connect the actuators to the manipulator controller are documented in Appendix C of Ref. [26]. All items on the cart were still powered by a 900 W, 24 V DC to AC power inverter connected to two 12 V, 28 Ah batteries [63, 64]. However, the batteries were replaced and an additional 2C40DL circuit breaker, with 250 V DC voltage rating and 10 kA short-circuit rating, was introduced to completely disconnect the batteries from the inverter when the manipulator-on-a-cart was not in use [65].

The manipulator controller supported several different TCP/IP client interfaces, which allowed feedback monitoring and control via a scripting language at different update rates [66, 67]. Of particular relevance for this work were the non-robust real-time interface and the robust real-time interface. Both interfaces operated at a max frequency of 125 Hz [67]. According to the manufacturer, the frequency of the non-robust real-time interface should vary depending on the compilation time of the script commands received by the controller. In contrast, the robust real-time interface is supposed to allow external and controller-run processes to be synchronized with the real-time control loop of the controller over TCP/IP while maintaining the real-time properties of the controller [67, 68]. However, overall performance with external applications may still vary with network latency [67, 68]. Due to these stated properties, the robust real-time interface was used instead of the non-robust real-time interface. As part of prior work, the firmware of the manipulator controller was upgraded to version 3.7.2.40245, which enables use of the robust real-time interface [29].

Custom C++ code leveraging the NIST Collaborative Robot Programming Interface (CRPI) was written as part of previous work and was referenced as a starting point [69]. However, several considerations and changes were required to allow the manipulator control code to compile and run natively on the SoM. First, any calls to ULAPI (previously used to implement TCP/IP socket communications, mutexes, and semaphores) needed to be replaced with direct Linux system calls. Second, the NIST posemath library, which is part of the NIST Real-time Control System (RCS) library was re-compiled to natively run on the SoM

[70–72]. Finally, the code controlling the manipulator was re-written to use the C++ based ur_rtde library instead of CRPI[6], the former of which will allow closed-loop OTS integration via the robust real-time interface [73].

The ur_rtde library allowed for utilization of either Cartesian pose commands, linear servo commands, or linear velocity commands, with the Cartesian pose command able to be either blocking[7] or non-blocking and the latter two of which are non-blocking only according to the documentation [74]. The ur_rtde library documentation also claimed to posses real-time capabilities, however, these capabilities required installation of a real-time operating system (OS) kernel patch on the SoM [75]. The real-time kernel patch, which ultimately used branch 5.15.y (i.e., version 5.15.y) of the SoM Linux kernel (downloaded from Ref. [76]) with PREEMPT-RT patch 5.15.65-rt-49 (downloaded from Ref. [77]), was installed using guides provided by the ur_rtde library documentation and similar SoM documentation [78, 79]. Furthermore, a real-time control loop example, also part of the ur_rtde library, was used as a reference to implement the control loops presented in Sec. 5 [78].

3.3. Type B CMMA

Among the two variants of the CMMA mentioned in Sec. 1, the Type B CMMA was used for this work since it provided a more direct analog to large scale part manufacturing [19]. The Type B CMMA, depicted in Fig. 4, consisted of a machined, anodized aluminum build with a general tolerance of ± 0.254 mm. The height and top surface rotation were reconfigurable, with the latter being adjusted using two linear actuators. Both linear actuators had an input voltage of 12 VDC, a max load of 900 N, a 76.2 mm stroke length, and a travel speed of 10 mm/s [80]. Unlike the Type A CMMA, the Type B CMMA featured a top surface constructed of multiple semi-circle shaped arcs, each with a general fabrication tolerance of 0.254 mm (i.e., the same tolerance as the Type A CMMA [81]). The Type B CMMA was configured such that each side had two arcs that could be reached by the manipulator. Full specifications and design information for both the Type A and Type B CMMA[8] can be found in Ref. [19]. Throughout the remainder of this report, the acronym, "CMMA", will be assumed to refer to the Type B CMMA unless otherwise noted.

To emulate manufacturing on a large-scale part, the mobile manipulator-on-a-cart was configured to dock at two arcs on opposite ends of the CMMA (see Fig. 5), which are about 3048 mm apart in length. Two, 3 mm diameter retro-reflective mock assembly targets, or AFs, were mounted per arc (as depicted in Fig. 6). Again, the detection of the AFs by the manipulator using the RLS (depicted in Fig. 7) corresponded to the task of typical peg-in-hole assembly [19, 21]. The configuration of the CMMA with OTS rigid body markers is

[6]For convenience, the AssemblyPrims source code, which is part of the MotionPrims library within CRPI, was extracted and re-written to function independently from CRPI, which allowed for easier porting of the spiral search routine [69].

[7]Meaning that the calling program waits until the function completes.

[8]Again note that, in Ref. [19], these were previously referred to as the "Static" and "Continuous" RMMAs, respectively.

Fig. 4. The Type B CMMA used to represent curved, complex parts for mobile manipulator performance measurement.

discussed further in Sec. 3.5.

3.4. Optical Tracking System

The same 20 camera array OTS (shown in Fig. 8) used for previous experiments was utilized again for the testbed [25, 26]. Each camera had 4.1 Megapixel (MP) resolution and was capable of outputting a full image at a native frame rate of up to 180 Frames-per-Second (FPS) (i.e., a latency of 5.5 ms) [83]. Although the manufacturer software had the option to output images at a rate exceeding the native frame rate (i.e., up to 250 FPS), this level of performance is achieved by reducing the processed image size [84]. Therefore, since this setting could adversely impact the field-of-view (FOV) and increase the presence of occlusions, the use of this option was avoided. To provide initial estimates of the different sources of OTS latencies, the outputs of the manufacturer software status pane when

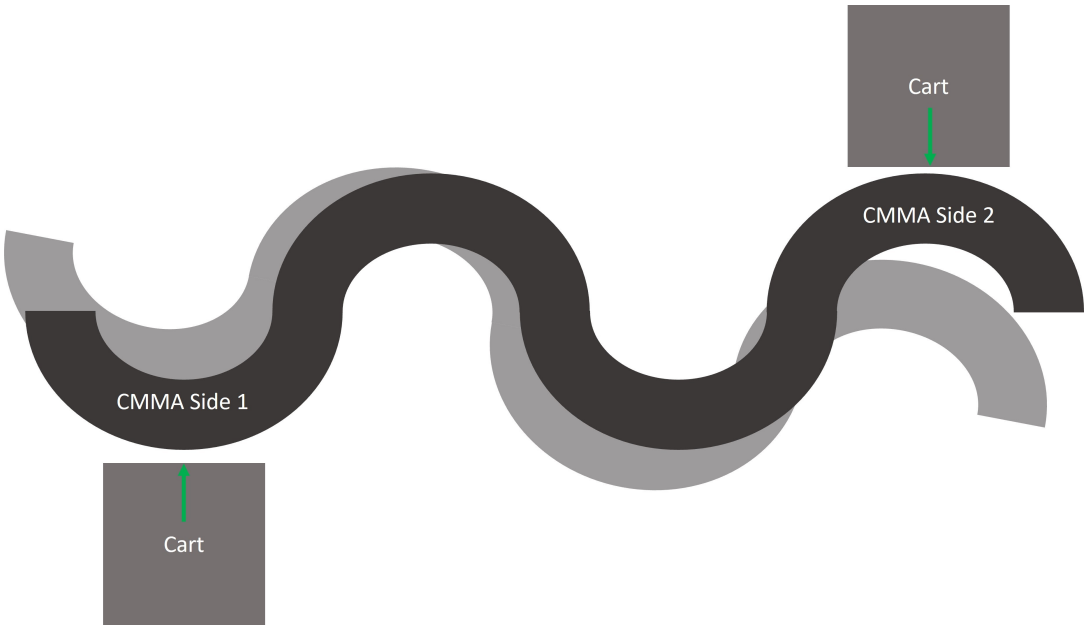

Fig. 5. Drawing of the cart docking configuration of the cart relative to the Type B CMMA. Note that it is assumed the Type B CMMA can be bumped or moved as the AMR-CT travels it or as the manipulator positions itself to detect retro-reflective fiducials.

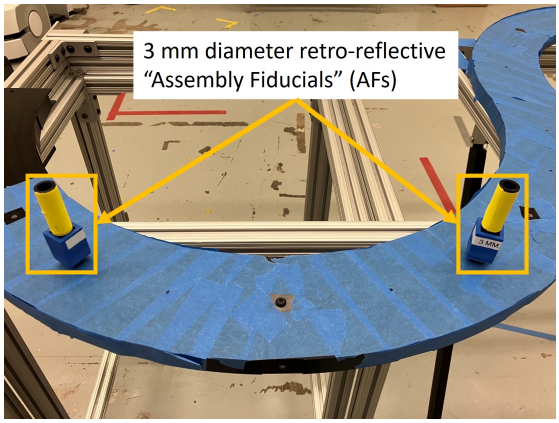

Fig. 6. One of the Type B CMMA arcs configured with two 3 mm diameter retro-reflective targets [26].

tracking three sample rigid bodies are shown in Fig. 9 [85]. Of particular note was that the reported values for the "system" latency (i.e., the total time between camera exposure and data being fully solved) did not appear to align with reported values for the "final rate" (i.e., the data acquisition rate of the OTS) [85]. For example, 6.9 ms corresponded to a frequency of approximately 145 Hz, which was higher than the 120 Hz final rate and lower than the 180 Hz final rate settings, respectively. Additionally, 5.2 ms corresponded to a frequency of approximately 192 Hz, which was lower than the final rate setting of 250 Hz.

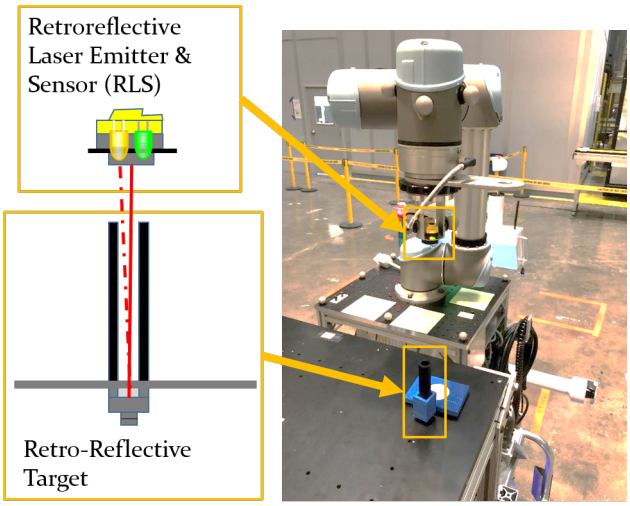

Fig. 7. The manipulator using the RLS to detect an AF [82].

Although not included for the prototype test implementation, the overall system latency of the OTS is planned to be re-verified using standard test method ASTM E3124-17 [86]. The cameras were previously placed to maximize the field-of-view (FOV) within the lab space, which had dimensions of 9 m x 22 m x 7 m (width x length x height) [25, 26]. Eight of the 20 cameras, including one ceiling-mounted camera, were focused directly on the CMMA. As recommended by the manufacturer and implemented in prior continuous performance experiments, the laboratory windows were previously covered using blackout curtains to block sunlight that might otherwise introduce Infrared (IR) spectrum reflections or occlusions [29, 87]. However, it was also discovered in previous work that the IR light emitted by the cameras themselves could introduce stray reflections off of the floor surface. Although not implemented for the prototype, this is planned to be mitigated by re-painting the lab floors with non-reflective paint.

As was done for previous projects, the OTS required calibration using a manufacturer specified procedure to optimize data capture quality [88]. In summary, the calibration procedure required three main steps. First, as many reflections as possible were removed or covered in the lab (including floor reflections) using an assortment of tarps. Reflections that could not be covered, such as those originating from the cameras, were instead masked via software. Second, sample data OTS data was recorded and generated by waving a calibration wand artifact in front of the cameras [89]. After this step, the OTS software displayed a calibration summary containing an ordinal quality value, mean ray error (in mm), and mean wand error (also in mm). A screen capture of this summary was recorded for each OTS calibration so that the quality could later be referenced and only calibrations labeled "exceptional" (i.e., the best quality), as determined by the manufacturer software, were accepted [88]. Third, the OTS coordinate system origin and ground plane (shown in Fig. 10) were set by placing and leveling a manufacturer-supplied square artifact within the lab. To ensure (more or less) the ground plane was set in a repeatable position, an outline of the

Fig. 8. OTS camera configuration (10 out of 20) shown.

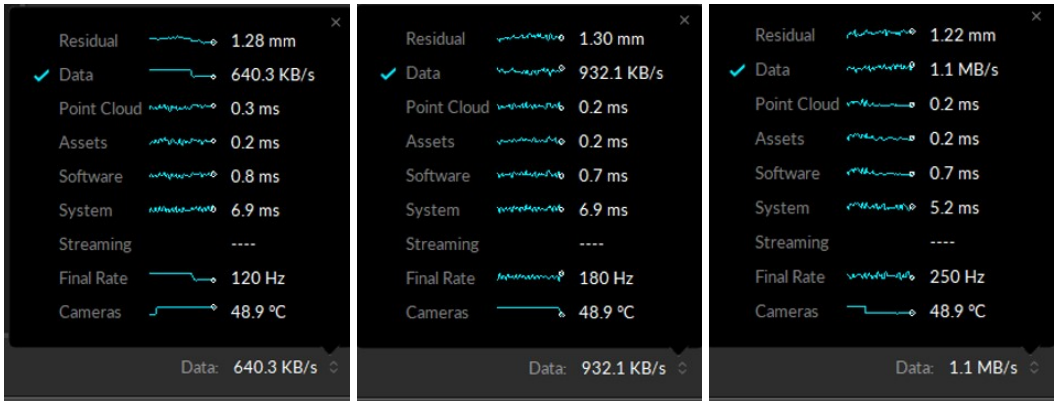

Fig. 9. Screenshot of the OTS software status pane showing example latency when setting the data acquisition rate to 120 FPS (left), 180 FPS (middle), and 250 FPS (right). Screenshots were taken when using the OTS to track three rigid bodies and other reflections are physically masked. Note that the processing required to package the data for streaming over the SDK (discussed later in the section) adds an additional 0.2 ms of latency to the system [84].

artifact had been previously drawn on the floor using a permanent marker for the prototype test implementation[9]. The square also has built in bubble levels to verify proper leveling.

[9]For the final implementation, a fixed mount embedded into the concrete will be installed within the lab.

A software feature that automatically updated the calibration was also used after initial calibration of the OTS [90]. The feature relied on fixed markers to automatically detect and improve calibration quality. Therefore, passive, 19-mm diameter OTS markers were super glued to a steel I-beam to utilize this feature. The layout and labeling of these markers is depicted in Fig. 11.

Fig. 10. Example of OTS ground-plane placement during calibration.

The manufacturer supplied control software was upgraded from version 3.0.0 Beta 1 to then current version 3.0.1 Final [91]. While again not included for the prototype test implementation, the OTS measurement uncertainty is planned to be re-evaluated using ASTM E3064-16 to determine if the OTS measurement uncertainty changed from the previously determined 0.022 mm (positional) and 0.00040143 rad (angular) values [20, 30, 92]. The computer used to run the OTS software, had the following specifications: An eight-core processor with a clock-rate of 3.70 GHz, 64 GB of RAM, a graphics card using driver version of 27.21.14.5671, and running Windows 10 build 19042 [93, 94]. Since the main focus of the testbed was to allow for the streaming of tracked rigid body data to the mobile manipulator during run-time, then-latest version 4.0.0 of a software development kit (SDK) supplied by the manufacturer was used to stream pose data to the custom applications controlling the AMR-CT and manipulator as clients [95, 96]. Only pre-compiled versions of SDK libraries were available and the executables targeted only x86 and x64 CPU architectures. Therefore, the library could be run on the SoM only using an emulator [57–59]. Since the data streaming library had to run in an emulator, but the manipulator control software could be run natively on the SoM, additional IPC code, using a shared memory scheme (see Ref. [97]), was implemented to bridge the two pieces of software.

Additionally, for the final test implementation beyond the prototype, the code will be revised to follow real-time programming guidelines, such as those presented in Refs. [98, 99], to maximize performance. The SDK supported real-time streaming of tracking data over both multicast and unicast User Datagram Protocol (UDP), however unicast data stream-

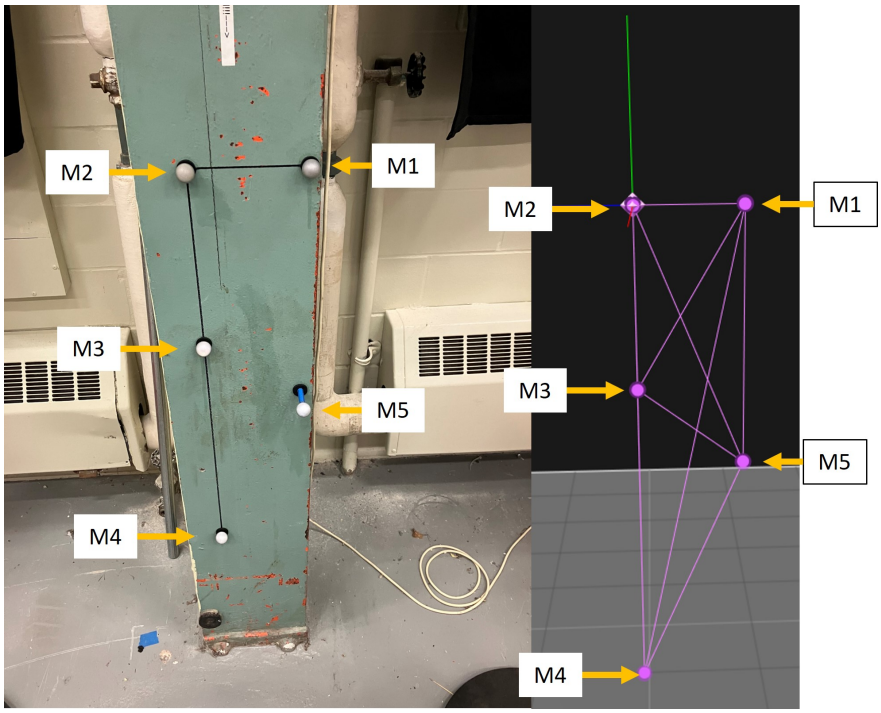

Fig. 11. The marker position and labeling of the fixed markers used for the automatic, continuous calibration update feature provided by the OTS software (left). Screenshot of the corresponding rigid body in the OTS tracking software (right).

ing was preferred since the functionality, according to the manufacturer, was specifically intended to improve performance over wireless networks by reducing packet sizes and the number of clients engaged in streaming [100, 101]. It is also important to note that, according to the manufacturer, packaging the data for streaming using the SDK adds an additional 0.2 ms of latency to the overall system [84]. Reference [84] provided additional guides on how to evaluate the streaming latency using the SDK, and the streaming code was modified to use the Welford Algorithm to conduct online computation of aggregate latency metrics including the mean and standard deviation latency [102]. The key functionality needed to integrate OTS feedback was to stream the full 6 DoF pose data for each of the rigid bodies described in Sec. 3.5. Aside from the online API documentation, which detailed how to use the Unicast mode to establish data filters (e.g., to stream only the pose of specific rigid bodies or labeled markers), several sample applications were included with the SDK that were used as code development references [95, 96, 101]. For example, the `WinForms` application was used to test and verify correct configuration/operation of basic data streaming functionality and the `SampleClient` application was used as a template for adding OTS streaming clients to the custom C++ applications previously written to control the AMR-CT and manipulator, respectively.

Before moving on to discuss the rigid body configurations, a few final points regarding the

OTS data output format are noted. Unlike the default coordinate system axis convention used by the OTS, in which the y axis corresponds to height, all rigid body tracking files (exported in CSV format) and data streams used a right-handed coordinate axis convention such that the x and y axes were parallel to ground plane and the z axis was perpendicular to the ground plane (i.e., a rotation of $\frac{\pi}{2}$ rad about the x axis compared to the default axis convention) [100, 103, 104]. This was configured in the "Tracking data export settings" for the former case and in the "Advanced network settings" of the "Data streaming pane" for the latter case [100, 103, 104]. For rigid body tracking files, each frame of rigid body tracking data was configured to include a frame number, timestamp, the 6 DoF pose of each tracked rigid body centroid, and the Cartesian position of individual markers that make-up each rigid body. The OTS expressed Cartesian positions along the x, y, and z axes and rigid body tracking files had the option to export orientations in either quaternion or Euler angle format [103]. However, it is important to note that the SDK would output rigid body orientations only as quaternions [104]. Therefore, seven floating point numbers per rigid body were needed at minimum for streaming over the network using the SDK, and conversion to alternate rotation formats, such as Euler angles, needed to be performed client-side [104]. Such conversions were implemented using the NIST RCS library, which was done for the mobile manipulator code base for previous projects [26, 105, 106]. Since quaternions can be readily composed with fewer operations than rotation matrices and do not suffer from singularities, like Euler angles, this format of rotation was favored for pose composition calculations whenever possible [107, 108].

3.5. Rigid Body Configuration

In this section, the tracked OTS rigid bodies, including marker labels and approximate coordinate frame origins for the associated tracked objects, are defined and depicted in Fig. 12 - 14. Rigid bodies were created to track the manipulator cart base, the CMMA, and the EOAT. To construct the tracked rigid bodies, 19 mm diameter passive retro-reflective markers were used for their balance between marker visibility and finer tracking of robot movements. Furthermore, it was discovered during prior use of the OTS that, in addition to the floor, the IR light emitted by the OTS cameras themselves could reflect off of the aluminum optical breadboards used for the CMMA and manipulator-on-a-cart. To mitigate these reflections, the top surface of the CMMA and the manipulator breadboard mount were covered with painter's tape (as shown in Fig. 13 (bottom) and Fig. 12 (left)). The EOAT mount was also covered with painters tape to reduce IR reflections, as depicted in Fig. 14.

The marker configuration for each rigid body was selected based on manufacturer guidelines and prior hands-on experience to improve tracking quality [26, 109, 110]. First, since rigid bodies are assumed to not be deformable, care was taken, especially for the EOAT rigid body, to ensure that markers did not span across any movable robot joints. Second, to minimize marker and rigid body mislabeling, the marker placement followed an asymmetric configuration within rigid bodies and was selected to not be congruent between

different rigid bodies. Third, it was ensured that not all markers of a rigid body were coplanar. This was done to: 1) Further minimize the potential for rigid body congruency 2) Improve tracking quality, since past experience with the OTS has shown that rigid body tracking uncertainty can be improved by varying the height of markers within rigid bodies. To accomplish this, multiple 1 in long Hex standoffs compatible with M4 and $\frac{1}{4} - 20$ in screws were used for mounting where needed. Furthermore, OTS markers were fixtured directly to the CMMA using $\frac{1}{4} - 20$ in to M4 screw adapters. Fourth, the number of markers per rigid body were kept between the recommended four and 12, inclusive. According to the manufacturer, rigid bodies having less than four markers can be more susceptible to occlusion, and rigid bodies having more than 12 markers may result in markers overlapping in the camera views. Finally, the markers used to discern orientation were spread out as far as possible from each other within the rigid body, as this was recommended by the manufacturer to improve 3D orientation tracking accuracy.

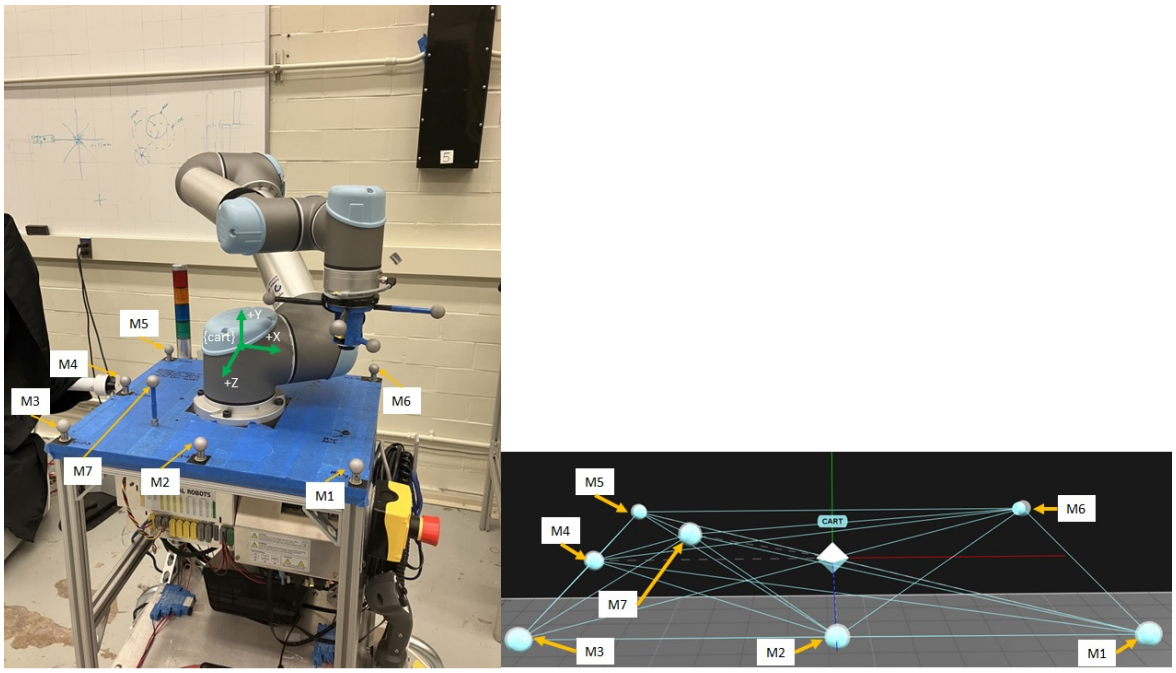

Fig. 12. The marker position and labeling on the manipulator support structure, as well as the approximate pose of the manipulator cart base coordinate frame origin (left). Screenshot of the corresponding rigid body in the OTS tracking software (right).

The centroid pose of the CART, CMMA, and EOAT rigid bodies were modified using a configuration option in the OTS software [110]. Specifically, the CART rigid body centroid was translated to coincide with the midpoint of markers M1, M3, M5, and M6 (see Fig. 12), the CMMA rigid body centroid was translated to coincide with marker M5 (see Fig. 13), and the EOAT rigid body centroid was translated to coincide with the midpoint of markers M3, M4, M5, and M6 (see Fig. 14). The CART rigid body orientation was adjusted by first modifying the roll component such that the positive x axis pointed in the direction facing

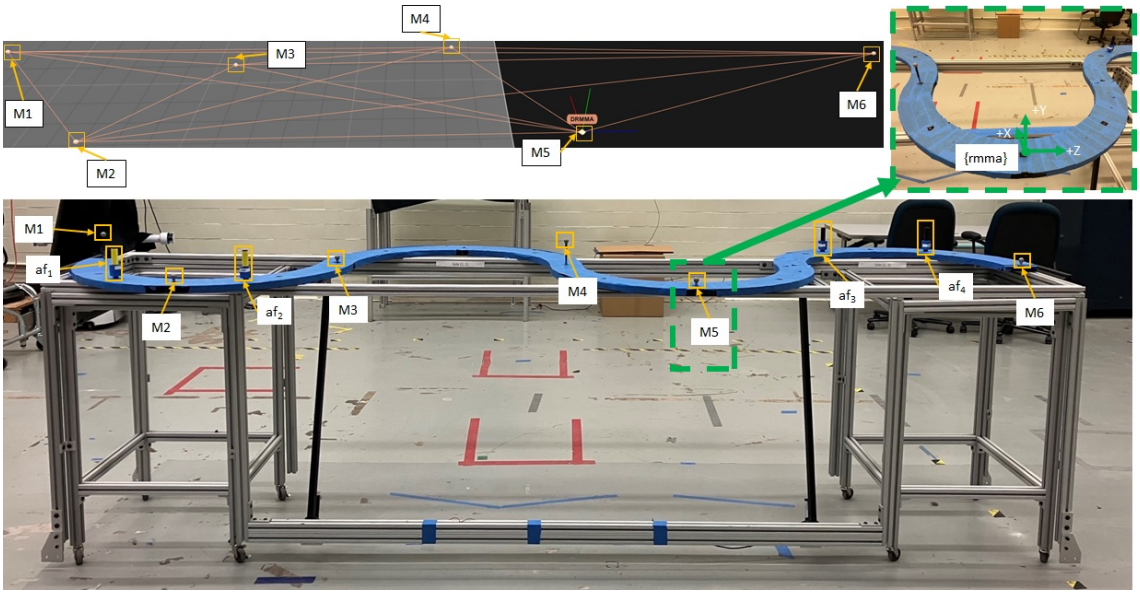

Fig. 13. Screenshot showing a top-view of the marker position and labeling for the CMMA rigid body in the OTS tracking software (top-left), corresponding marker configuration (bottom), and the approximate pose of the 6 DoF CMMA coordinate frame origin (top-right).

away from marker M4, followed by modifications to the pitch and yaw angles such that positive z axis pointed towards marker M2. Similarly, the CMMA rigid body orientation was adjusted by modifying the roll angle such that the positive z axis pointed in the direction facing away from marker M2. Subsequently, the pitch and yaw angles were adjusted by pointing the positive x axis in the direction facing towards from the midpoint between markers M3 and M6. The EOAT rigid body centroid orientation was adjusted by modifying the roll angle such that the positive x axis pointed toward marker M4 and the positive y axis pointed downward. Subsequently, the pitch and yaw angles were adjusted such that the positive z axis pointed toward marker M3.

3.6. Network Time Synchronization

This section describes the time synchronization requirements and architecture planned to align the AMR-CT and manipulator position for closed-loop control (see Fig. 2). With new commercially available industrial internet of things (IIoT) technologies, the objective was to improve the synchronization requirements to the order of 1 ms or better.

In prior experiments [19, 26], NTP was used to synchronized the OTS, manipulator, and AMR-CT controller systems. In a conservative estimate, Network Time Protocol (NTP) provides 100 ms to 1 s synchronization precision between nodes in a wireless environment.

In the closed-loop testbed design, the key improvements from Ref. [26] included: (a) reduced number of wireless hops, (b) reduced one-way path delay variability using hardware

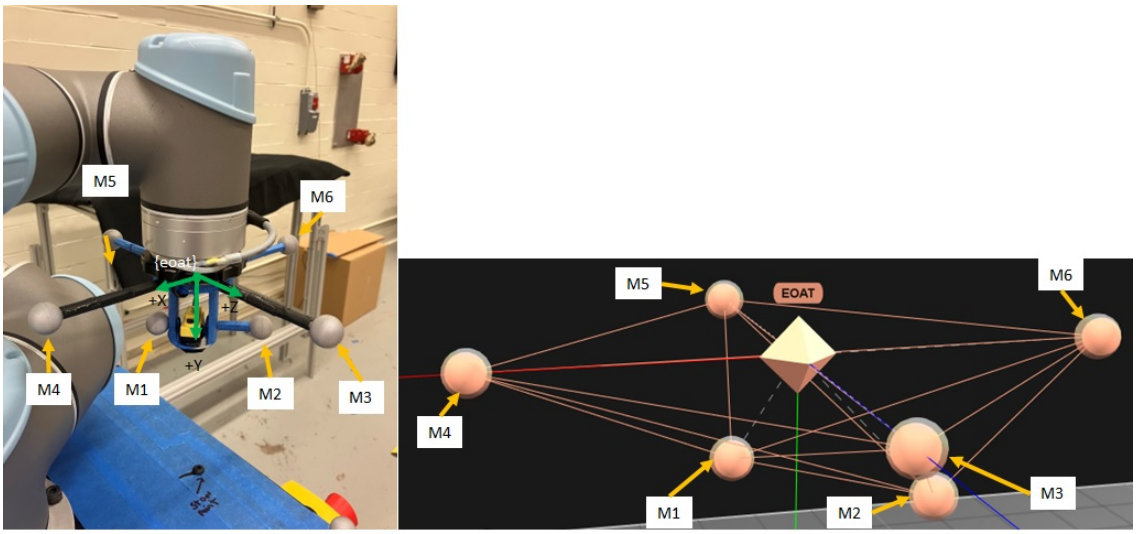

Fig. 14. The marker position and labeling on the EOAT, as well as the approximate pose of the EOAT coordinate frame origin (left). Screenshot of the corresponding rigid body in the OTS tracking software (right).

timestamping at the network nodes where feasible by replacing (1) switches with transparent clocks, (2) OTS node with Precision Time Protocol (PTP) Peripheral Component Interconnect Express (PCIe) card for hardware timestamping of Ethernet packets, and (3) providing a Linux-based onboard compute and controller module also with hardware timestamping and a real-time clock.

The timing requirement of 1 ms was established based on the constraints of the closed-loop system. The OTS had a maximum sampling frequency 180 Hz or one 6 DoF position data for each rigid body approximately every 5.6 ms. In addition, the real-time interface of the manipulator had a maximum frequency of 125 Hz, or a control frequency resolution of 8 ms. The manipulator speed was initially limited to 0.1 mm/s for the closed-loop prototype.

Beyond the prototype test implementation, a key part of later evaluation will be to characterize the delays and delay variabilities in the closed loop system in order to improve the control precision and response. Each component of the system contributes to both delay and delay uncertainty. The characterization includes measuring the sources of delays from (1) a target change in position event to the OTS data capture timestamping; (2) translating the coordinate system; (3) path planning; and (4) position command to start of AMR-CT or manipulator response.

4. Coordinate System Transformations and Registrations

The coordinate transformations required to integrate OTS feedback for closed-loop control of the manipulator and the AMR-CT are now detailed. Since the AMR-CT capabilities

did not support continuous pose command updates and the ± 100 mm position, ± 0.035 rad orientation docking uncertainty of the AMR-CT (see Sec. 3.1) was known to be much larger than the sub-millimeter position repeatability of the manipulator (see Sec. 3.2), direct registration between the manipulator cart base, the EOAT, and the CMMA using the OTS was preferred after the cart has been docked to the CMMA [52, 60]. For this purpose, two cases of coordinate registration are described, including a 3 DoF coordinate registration and a 6 DoF coordinate registration. It was expected that the 3 DoF coordinate registration would be simpler to implement. However, since the 3 DoF coordinate registration relies on the assumption that the mobile manipulator and the CMMA lie on a flat plane, this coordinate registration method may be less robust to environmental conditions such as floor tilt/undulation (as discussed in Sec. 2) and the rotation of the top surface of the CMMA must remain flat (i.e., no tilt). Since the implementation of the 3 DoF and 6 DoF cases are very similar, with the main key difference being that height, roll and pitch components are ignored for the former, the procedure described will be assumed to be applicable to both cases unless otherwise noted.

4.1. Manipulator Cart Base to CMMA Registration

To implement the registration between the manipulator cart base and the CMMA, streamed data from the OTS rigid body presented in Fig. 13 was used each time the manipulator was docked with the CMMA. Five coordinate frames were involved (as depicted in Fig. 15), including the OTS coordinate frame (denoted OTS), the manipulator cart base coordinate frame (denoted *cbase*), the CMMA coordinate frame (denoted *cmma*), the AF coordinate frames (denoted af_i where $i \in \{1,2,3,4\}$ is the fiducial target number), and the coordinate frame of the docking goal of the AMR-CT (denoted $dock$). Again, since the Cartesian pose and linear servo API calls of ur_rtde expected the pose of the EOAT to be specified relative to the manipulator cart base, the goal was to determine the coordinate transformation chain needed to express the commanded pose of the EOAT to intercept the AFs using the RLS [111]. This pose was denoted ${}^{\{cbase\}}\xi_{\{af_i\}}$ where $i \in \{1,2,3,4\}$ is the fiducial target number and is interpreted as "the pose of the AF relative to manipulator cart base". Note that since the AF was rigidly fixtured to the CMMA, the pose of the AF relative to the CMMA, denoted ${}^{\{cmma\}}\xi_{\{af_i\}}$ where $i \in \{1,2,3,4\}$, was fixed and known a priori. Furthermore, the height and yaw components of ${}^{\{cmma\}}\xi_{\{af_i\}}$ were set to arbitrary constant values, while the roll was set constant to π rad such that the commanded EOAT pose would face downward with the RLS pointing towards the AF. Additionally, since the AMR-CT controller presented in Sec. 5.1 required the ability to detect when the cart has arrived at the docking goal, an additional registration was needed to determine the measured pose of the docking goal relative to manipulator cart base (i.e., ${}^{\{cbase\}}\xi_{\{dock\}}$). The coordinate transformation describing the pose of manipulator cart base relative to the OTS, denoted, ${}^{\{OTS\}}\xi_{\{cbase\}}$, and the pose of the CMMA relative to the OTS, denoted, ${}^{\{OTS\}}\xi_{\{cmma\}}$, were streamed directly from the OTS. Therefore, the ideal pose of the EOAT relative to the manipulator cart base such that the EOAT can intercept the AFs was determined by Eq. 1.

$$ {}^{\{cbase\}}\xi_{\{af_i\}} = ({}^{\{OTS\}}\xi_{\{cbase\}})^{-1}\,{}^{\{OTS\}}\xi_{\{cmma\}}\,{}^{\{cmma\}}\xi_{\{af_i\}} \tag{1} $$

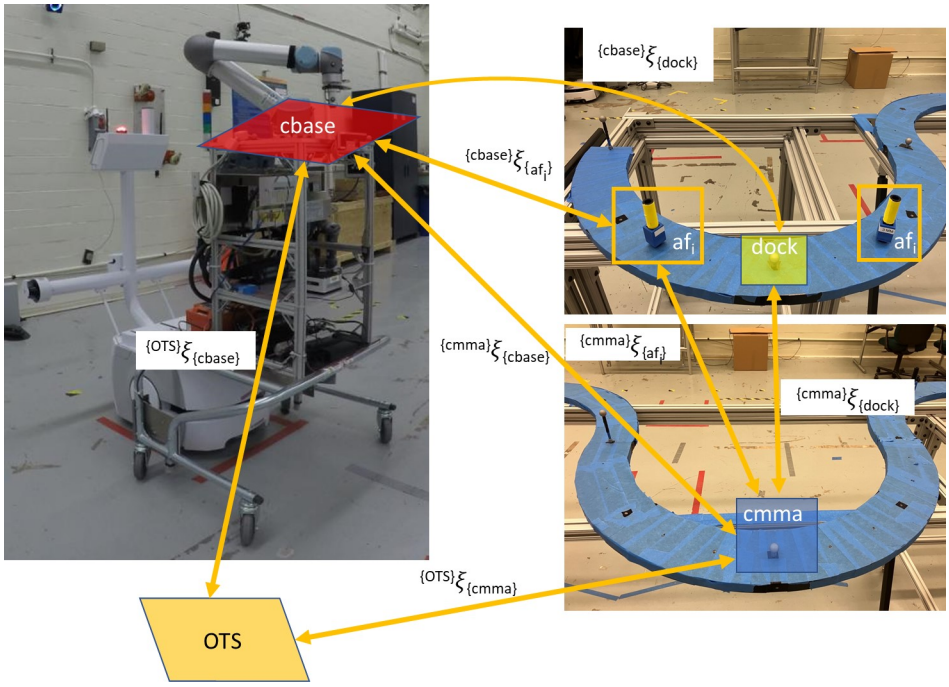

Fig. 15. Diagram showing the coordinate system transformations needed to conduct a 6 DoF registration between the manipulator cart base and the CMMA.

Now, to determine the additional registration between manipulator cart base and the docking goal, ${}^{\{cbase\}}\xi_{\{dock\}}$, an additional relative pose describing the pose of the docking goal relative to the CMMA, denoted ${}^{\{cmma\}}\xi_{\{dock\}}$, was defined. The transformation, ${}^{\{cmma\}}\xi_{\{dock\}}$, was fixed and pre-defined, which means ${}^{\{dock\}}\xi_{\{cbase\}}$ was determined by Eq. 2.

$$ {}^{\{cbase\}}\xi_{\{dock\}} = ({}^{\{OTS\}}\xi_{\{cbase\}})^{-1}\,{}^{\{OTS\}}\xi_{\{cmma\}}\,{}^{\{cmma\}}\xi_{\{dock\}} \tag{2} $$

4.2. EOAT to Manipulator Cart Base Registration

To implement the coordinate system registration between the EOAT and the manipulator cart base, the following coordinate systems depicted in Fig. 16 were needed. Again, let *OTS* denote the OTS coordinate frame and the *cbase* denote manipulator cart base coordinate frame, as was the case in Sec. 4.1. Additionally, let *eoat* denote the coordinate frame associated with the EOAT (depicted in Fig. 14). The objective was to determine the coordinate transformation chain needed to express the measured pose of the EOAT relative to the manipulator cart base, denoted ${}^{\{cbase\}}\xi_{\{eoat\}}$, so it could be compared to its commanded

pose in orientation. Then, the relative poses, ${}^{\{OTS\}}\xi_{\{cbase\}}$ and ${}^{\{OTS\}}\xi_{\{eoat\}}$, denoted the measured pose of manipulator cart base and EOAT, respectively, relative to the OTS coordinate system, and the measured pose of the EOAT relative to the manipulator cart base was given by Eq. 3.

$$ {}^{\{cbase\}}\xi_{\{eoat\}} = ({}^{\{OTS\}}\xi_{\{cbase\}})^{-1}\,{}^{\{OTS\}}\xi_{\{eoat\}} \tag{3} $$

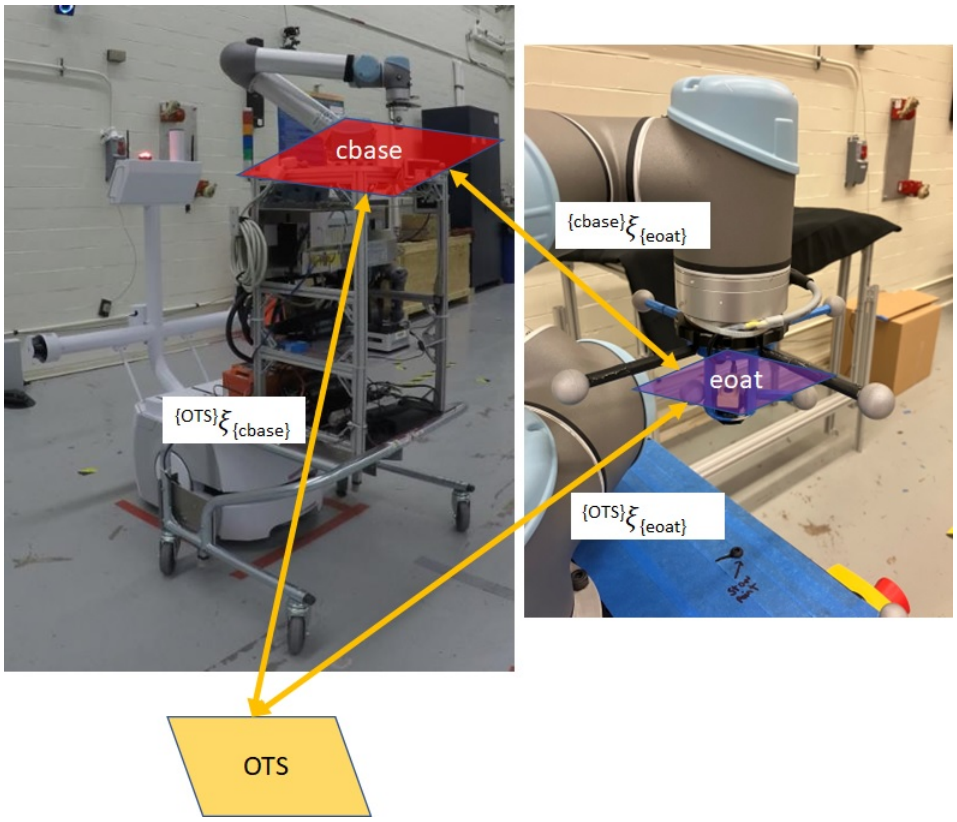

Fig. 16. Diagram showing the 6 DoF coordinate system transformations needed for closed-loop to register between the EOAT to the manipulator cart base.

4.3. EOAT to CMMA Registration

To implement the coordinate system registration between the EOAT and the CMMA, the following coordinate transformations depicted in Fig. 17 were needed. Again, let *OTS* denote the OTS coordinate frame, *cbase* denote manipulator cart base coordinate frame, and *eoat* denote the EOAT coordinate frame, as was the case in Sec. 4.1 and 4.2. Additionally, again let *cmma* denote the coordinate frame associated with CMMA, as was the case in Sec. 4.1. Similar to Sec. 4.1, the goal was to determine the coordinate transformation chain needed to express the ideal pose of EOAT to intercept the retro-reflective targets with the RLS, except this time relative to the EOAT coordinate frame rather than relative to the

manipulator cart base. This pose was given by ${}^{\{eoat\}}\xi_{\{af_i\}}$ where $i \in \{1,2,3,4\}$ is again the fiducial target number. This was because the Cartesian controller presented later in Sec. 5 required this transformation to determine when the EOAT has reached the commanded pose and the Proportional-Integral-Derivative (PID) controller in Fig. 23 of Appendix C took the error between the current EOAT pose and the commanded pose as input for velocity-based control. The relative pose, ${}^{\{OTS\}}\xi_{\{cmma\}}$, again denoted the measured pose of the CMMA relative to the OTS coordinate system, which was obtained directly from the OTS rigid body tracking. The pose, ${}^{\{OTS\}}\xi_{\{eoat\}}$, describing the measured pose of the EOAT relative to the OTS coordinate system was also obtained directly from the OTS. Finally, the pose of the AF relative to the CMMA is given by ${}^{\{cmma\}}\xi_{\{af_i\}}$ where $i \in \{1,2,3,4\}$ is the fiducial target number. Again, this pose is constant and known a priori.

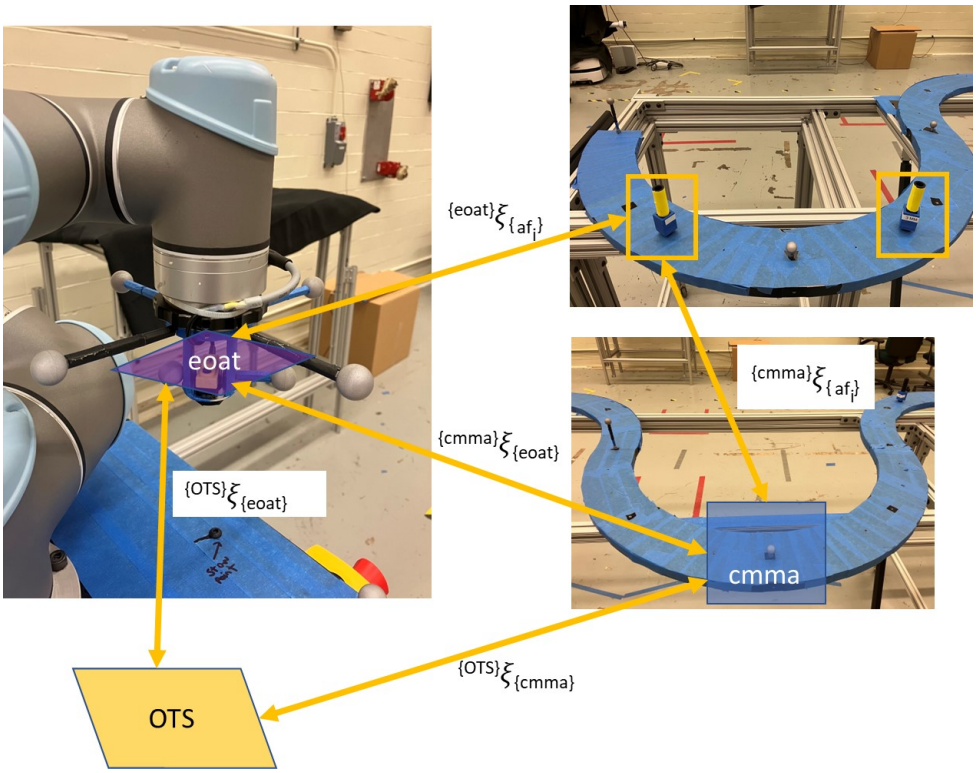

Fig. 17. Diagram showing the 6 DoF coordinate system transformations needed for closed-loop fine positioning of the EOAT over the AF.

Therefore, the retro-reflective fiducial positions relative to the EOAT were determined by Eq. 4.

$$ {}^{\{eoat\}}\xi_{\{af_i\}} = ({}^{\{OTS\}}\xi_{\{eoat\}})^{-1}\,{}^{\{OTS\}}\xi_{\{cmma\}}\,{}^{\{cmma\}}\xi_{\{af_i\}} \tag{4} $$

4.4. Cart Transporter Map to OTS Registration

The coordinate transformations needed to coarsely pose the AMR-CT to dock with the CMMA are now explained and depicted in Fig. 18. Similar to Sec. 4.1, three coordinate frames for the OTS, manipulator cart base, and the CMMA were defined and denoted *OTS*, *cbase*, and *cmma*, respectively. The coordinate frame of the AMR-CT itself was denoted *vbase*. The AMR-CT map coordinate system was represented by up to two additional coordinate frames (depending on if 3 DoF or 6 DoF are used). The first was *MAP_2D*, which, in the 3 DoF case, corresponded to the 2D AMR-CT map coordinate system or, in the 6 DoF case, embeded the 2D AMR-CT map coordinate system into 3D by assuming zero height, roll, and pitch. The second coordinate system was a 3D projection of 2D AMR-CT map, denoted *MAP_3D*, which was only needed for the 6 DoF case. Additionally, *dock* denoted the coordinate frame of the ideal docking goal near the CMMA. Since, the `goToPoint` clear-text command was used to position the AMR-CT near the CMMA, the goal was to obtain ${}^{\{MAP\_2D\}}\xi_{\{vbase\}}$, which denoted the commanded position of the AMR-CT coordinate frame in the *MAP_2D* coordinate frame.

The following additional relative poses are now defined. First, let

${}^{\{MAP\_3D\}}\xi_{\{vbase\}}$ and ${}^{\{OTS\}}\xi_{\{vbase\}}$

denote the relative poses describing the commanded pose of the AMR-CT relative to the *MAP_3D* and *OTS* coordinate frames, respectively. Furthermore, ${}^{\{OTS\}}\xi_{\{cmma\}}$ denoted the transformation describing the pose of the CMMA relative to the *OTS* coordinate frame, which is again given directly from the streamed OTS data. The transformation, ${}^{\{cmma\}}\xi_{\{dock\}}$ and ${}^{\{cbase\}}\xi'_{\{dock\}}$, denoted the transformations describing the commanded pose of the docking goal relative to the CMMA and cart base, respectively. These transformations were fixed and pre-defined such that the front of manipulator cart base directly faced the CMMA, the center of the cart aligned with the center of the docking arc, and the cart was initially 1000 mm away from the CMMA in x. However, the difference in height between manipulator cart base and the configured height of the CMMA also needed to be factored into the height component of these transformations in the 6 DoF case. Finally, ${}^{\{vbase\}}\xi_{\{cbase\}}$ denoted the fixed pose of manipulator cart base relative to the AMR-CT base, which was previously measured as part of the work in Ref. [29].

Therefore, for the 6 DoF case, the transformations ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_3D}\}}$ and ${}^{\{\mathbf{MAP\_2D}\}}\xi_{\{\mathbf{MAP\_3D}\}}$ were the only potentially unknown coordinate transformations. However, a 6 DoF calibration procedure for the transformation AMR-CT map coordinate system and OTS was devised and implemented in Ref. [29]. As a quick summary (full details will be available in Ref. [29]), the calibration procedure involved parking the AMR-CT at seven, level (as measured by digital levels attached to the AMR-CT) locations throughout the lab, recording pose data from the AMR-CT controller and OTS rigid body data, and applying closed-form solutions to the collected pose data to calibrate the unknown coordinate transformations [29, 112–114]. The same procedure without modification was used again to solve

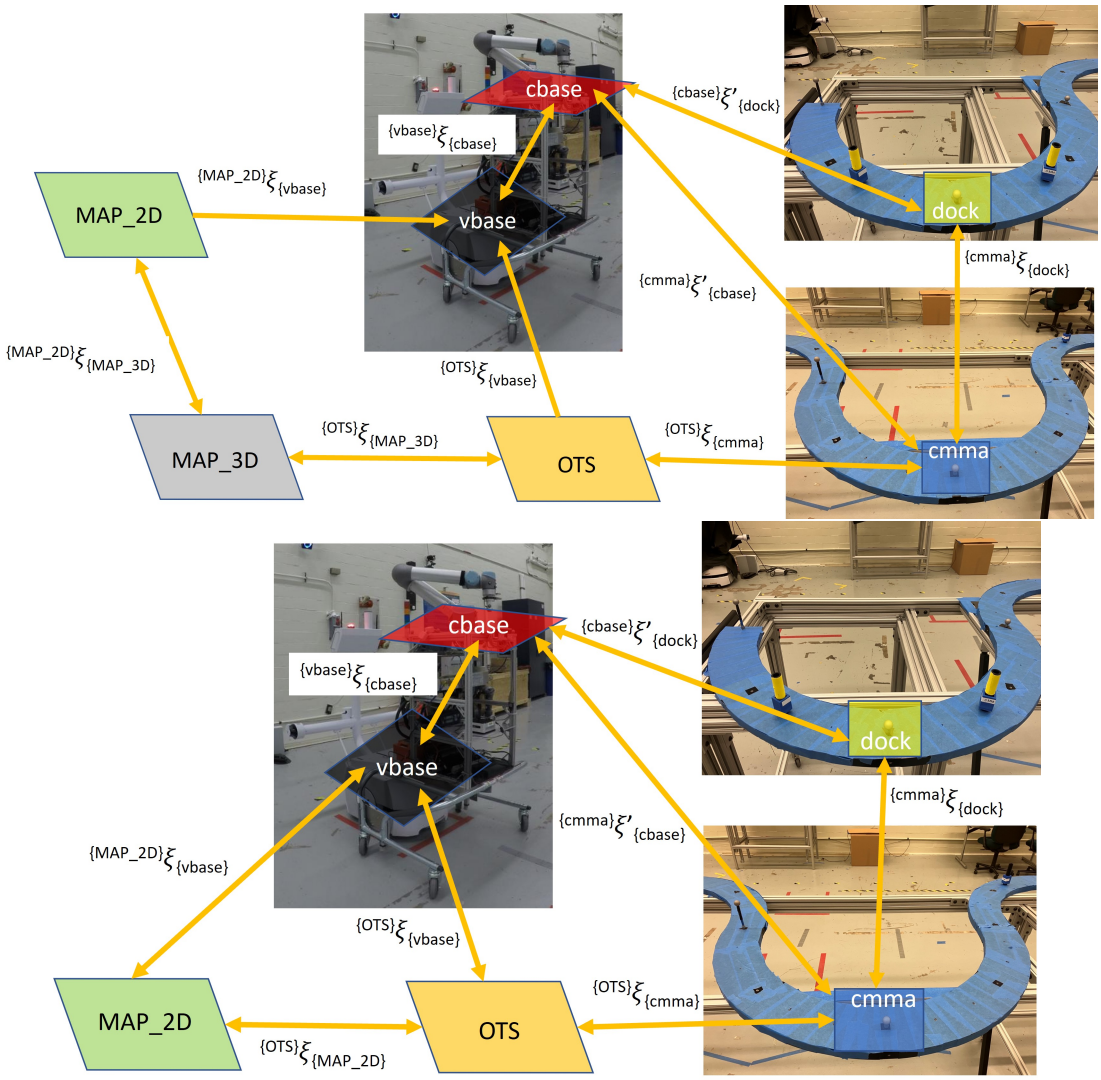

Fig. 18. Diagram showing the 6 DoF (top) and 3 DoF (bottom) coordinate system transformations needed to command the AMR-CT to dock with the CMMA.

for ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_3D}\}}$ and ${}^{\{\mathbf{MAP\_2D}\}}\xi_{\{\mathbf{MAP\_3D}\}}$, which meant the complete coordinate system transformation chain for the 6 DoF case was expressed by Eq. 5.

$$
\begin{aligned}
{}^{\{MAP\_2D\}}\xi_{\{vbase\}} = {}^{\{\mathbf{MAP\_2D}\}}\xi_{\{\mathbf{MAP\_3D}\}}({}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_3D}\}})^{-1} \\
{}^{\{OTS\}}\xi_{\{cmma\}}\,{}^{\{cmma\}}\xi_{\{dock\}}({}^{\{cbase\}}\xi'_{\{dock\}})^{-1}({}^{\{vbase\}}\xi_{\{cbase\}})^{-1}
\end{aligned} \tag{5}
$$

For the 3 DoF case, only the transformation that goes directly between the 2D projected OTS coordinate system and the 2D AMR-CT map (i.e., ${}^{\{OTS\}}\xi_{\{MAP\_2D\}}$) was potentially unknown. However, the calibration procedure from the 6 DoF case was easily adapted to fit the 3 DoF case. The first main difference in the procedure was the need to collect only 2D Cartesian points of the AMR-CT in each coordinate system (i.e., ${}^{\{OTS\}}\vec{v}_j$ and ${}^{\{MAP\_2D\}}\vec{v}_j$) and at level locations throughout the lab. Since the two vectors are assumed to track approximately the same point, a solution to the absolute orientation problem, ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_2D}\}}\,{}^{\{MAP\_2D\}}\vec{v}_j \approx {}^{\{OTS\}}\vec{v}_j$, by Arun et. al. was applied [115, 116]. This solution was previously used to solve a similar calibration problem involving the EOAT in Ref. [26]. Therefore, the complete coordinate system transformation chain for the 3 DoF case was expressed by Eq. 6.

$$
\begin{aligned}
&{}^{\{MAP\_2D\}}\xi_{\{vbase\}} = \\
&({}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_2D}\}})^{-1}\,{}^{\{OTS\}}\xi_{\{cmma\}}\,{}^{\{cmma\}}\xi_{\{dock\}}({}^{\{cbase\}}\xi'_{\{dock\}})^{-1}({}^{\{vbase\}}\xi_{\{cbase\}})^{-1}
\end{aligned} \tag{6}
$$

Note that, since ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_3D}\}}$, ${}^{\{\mathbf{MAP\_2D}\}}\xi_{\{\mathbf{MAP\_3D}\}}$, and ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{MAP\_2D}\}}$ were measured based on samples of a tracked point associated with the AMR-CT base, these transformations (or their respective inverses) were directly applicable only to AMR-CT poses expressed in either the *MAP_2D*, *MAP_3D*, or *OTS* coordinate frames.

5. Closed-Loop Controller Design

This section describes the algorithm and data flow for achieving the operations outlined in Fig. 19. First, the event-based AMR-CT control loop (Fig. 20 top) provided coarse posing (navigation) of the mobile manipulator to a goal near the CMMA using the OTS. If the goal was successfully reached, then the second control loop proceeded, which provided refined docking of the AMR-CT close to the CMMA such that the manipulator could reach the CMMA. If the CMMA were to be moved during this process, control would return back to the coarse pose AMR-CT control loop, otherwise the control flow proceeded upon successful docking. Finally, the closed-loop manipulator control algorithm and data flow using Cartesian/linear servo pose commands are described in Fig. 21, which posed the manipulator EOAT over the AFs to simulate mock peg-in-hole assembly. Note that an alternative

design, which used a PID controller and tool-center-point (TCP) control via velocity-based commands is presented in Appendix C and was considered as a back-up design in case the Cartesian/linear servo pose controller exhibited limitations in responsiveness.

Before describing each control algorithm individually, the steps common to all three controllers are now addressed. First, the poses for the needed rigid bodies (i.e., the CART, EOAT, and CMMA rigid bodies for the Cartesian pose controller), just the EOAT and CMMA rigid bodies for the velocity controller, and just the CMMA rigid body for the AMR-CT control algorithm) were streamed over UDP from the OTS to the SoM mounted on the cart using the manufacturer SDK. The poses arrived at the SoM in vector-quaternion pair format. As mentioned in Sec. 3.4, the OTS was configured to stream data at 125 FPS. The data streaming client, running on the SoM then received the data and pre-processed it to verify that no significant occlusions, dropped packets, or mislabeling occurred. If a network or data quality error occurred, the client would wait until clean data arrives. Otherwise, the controllers proceeded to the next processing steps.

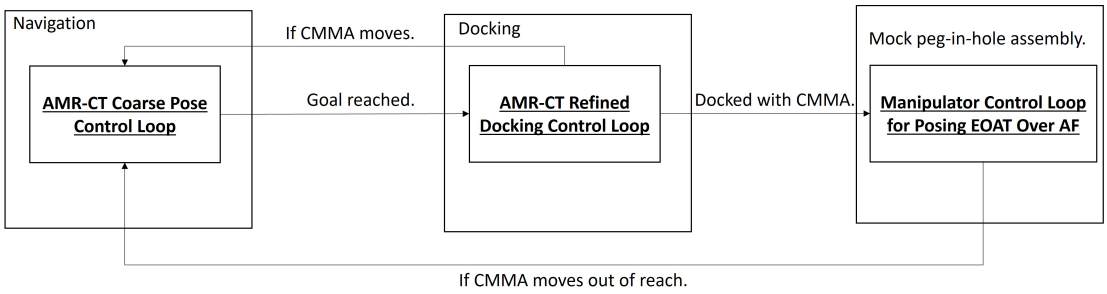

Fig. 19. Diagram showing high-level control flow between the two closed-loop controllers for implementing coarse AMR-CT posing (navigation), and fine docking near the CMMA and the manipulator controller to perform mock peg-in-hole-assembly through detection of the AFs using the RLS.

5.1. Cart Transporter Controller Design

Recall from Sec. 3.1 that the dead-reckoning `move` command was needed to dock the AMR-CT close to the CMMA without detecting it as an obstacle and the `deltaHeading` command was used to implement heading corrections. As such, two control loops were needed for AMR-CT control: The first control loop coarsely posed the AMR-CT 1 m away from the intended docking position, while the second control loop allowed the AMR-CT to dock close the CMMA while simultaneously correcting the AMR-CT heading. The former control loop is detailed first.

After receiving the marker position data from the OTS and computing the needed rigid body poses as per the beginning of Sec. 5, the AMR-CT controller executed the following steps. First, ${}^{\{cbase\}}\xi_{\{dock\}}$ was computed using Eq. 2, vector-quaternion pose composition, and quaternion inverses where applicable [107, 108]. The controller used this transforma-

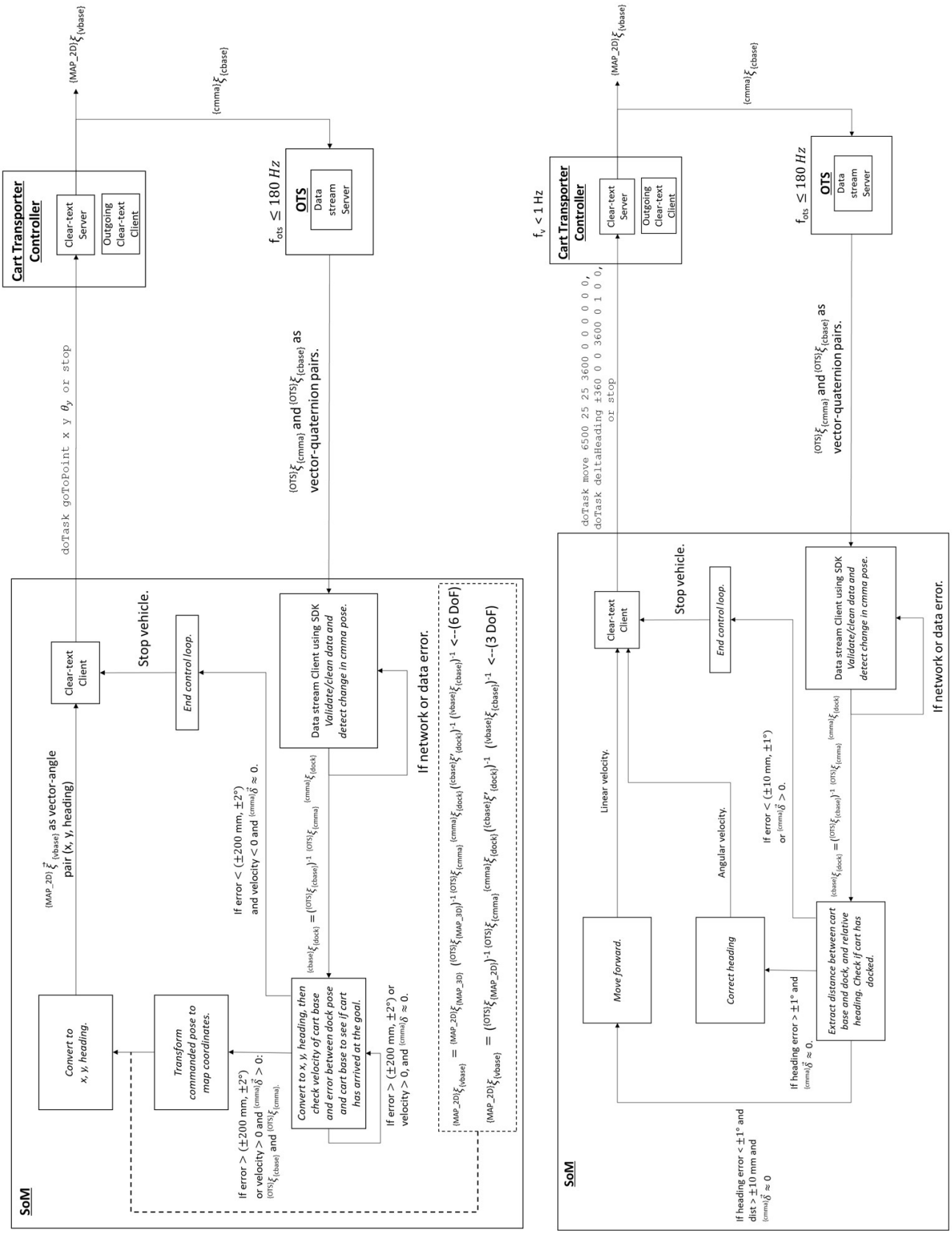

Fig. 20. Diagram showing the control algorithm design for coarse AMR-CT posing (top) and for refined docking near the CMMA with AMR-CT heading corrections (bottom). Note that f_v and f_{ots} denote the control frequencies of the AMR-CT controller and OTS, respectively. Additionally, ξ is used to denote poses in a described representation.

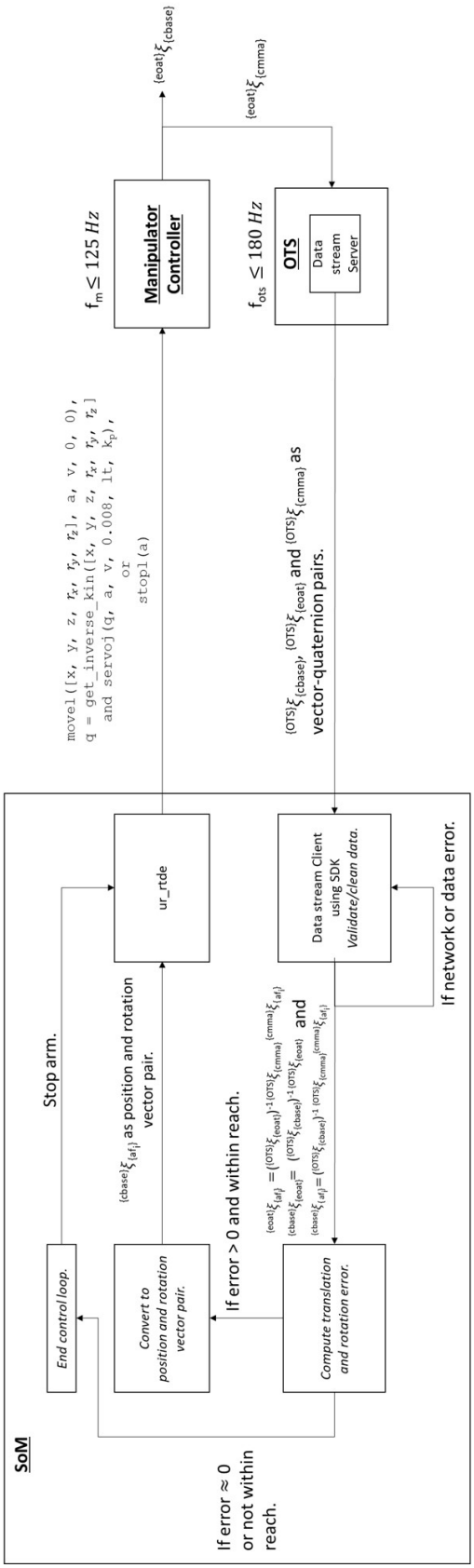

Fig. 21. Diagram showing the closed-loop manipulator controller design for the Cartesian/linear servo pose controller. Note that f_m and f_{ots} denote the control frequencies of the manipulator controller and OTS, respectively.

tion, which represented the error between the current pose of manipulator cart base and the docking goal, to check if the AMR-CT had arrived at the docking goal near the CMMA. Additionally, the speed of manipulator cart base and the change in pose of the CMMA between the current loop execution and the preceding loop execution (denoted ${}^{\{cmma\}}\vec{\delta}$) was monitored. If the error between manipulator cart base pose and docking goal was greater than a pre-set threshold [10], and the velocity of manipulator cart base[11] and ${}^{\{cmma\}}\vec{\delta}$ was less than another pre-defined threshold[12], then the controller would do nothing. However, if the error between manipulator cart base pose and docking goal, the velocity of manipulator cart base, and ${}^{\{cmma\}}\vec{\delta}$ all fell below their respective pre-set thresholds, then the control loop would end, a `stop` command would be sent to the vehicle through the clear-text API, and control was handed over to the manipulator. Otherwise, the control loop continued and updated the commanded pose of the AMR-CT based on the updated CMMA pose. This step was completed using either Eq. 6 or Eq. 5 (again using vector-quaternion pose composition, and quaternion inverses where applicable [107, 108]) depending on whether 3 DoF or 6 DoF were used for the registrations. The pose containing the new commanded AMR-CT pose, ${}^{\{MAP\_2D\}}\xi_{\{vbase\}}$ was then converted to a vector format consisted of Cartesian position components and a heading. The vector components could then be directly substituted as input parameters for a `doTask goToPoint` clear-text command, which updated the AMR-CT path planning with a new goal. As the vehicle traveled to the new goal, the pose of the manipulator cart base relative to the CMMA, (i.e., ${}^{\{cmma\}}\xi_{\{cbase\}}$) would change and the control loop repeated.

The second control loop is now described. The steps to acquire ${}^{\{cbase\}}\xi_{\{dock\}}$ were identical to the preceding control loop. From this transformation, the relative yaw angle between manipulator cart base and the CMMA was extracted. If the heading error was greater than a preset threshold[13], then a heading correction was applied to the vehicle using a `deltaHeading` command to turn the AMR-CT in the direction opposite to the error. The `deltaHeading` command would be allowed to execute until the heading error fell below the error threshold. After the heading constraint had been satisfied, if the distance between manipulator cart base and CMMA was greater than another preset threshold[14], then the AMR-CT would be commanded to move forward to dock with the CMMA. The move command would be allowed to run until manipulator cart base reached the desired docking position. Note that control could shift between docking and correcting the heading at any time as needed. When both the heading and distance constraints have been satisfied, the control loop would terminate successfully. However, another termination condition was included: If, at any time, the CMMA was moved, then the docking control loop terminated and control was returned to the coarse pose control loop described in the previous

[10] For the prototype, the thresholds were set to ± 200 mm in position and ± 0.035 rad in heading based on the specs of the AMR-CT in Sec. 3.1 and manual tuning [52].
[11] For the prototype, this threshold was set to 1 mm/s
[12] For the prototype, this threshold was set to ± 5 mm.
[13] This threshold was set to ± 0.0175 rad for the prototype.
[14] This threshold was wet to ± 10 mm for the prototype.

paragraph. Again, since the control loop updates were based on specific conditions, the effective overall control rate of the AMR-CT was expected to be slower than 1 Hz.

5.2. Manipulator Controller Design

The control loop steps to pose the manipulator EOAT such that the AF was intercepted by the RLS is now described. After the common steps outlined at the beginning of Sec. 5, the manipulator pose controller that used Cartesian/linear servo commands then computed the needed relative poses. Specifically, these poses were ${}^{\{eoat\}}\xi_{\{af_i\}}$ (using Eq. 4), ${}^{\{cbase\}}\xi_{\{eoat\}}$ (using Eq. 3) and ${}^{\{cbase\}}\xi_{\{af_i\}}$ (using Eq. 1). The Cartesian/linear servo controller then checked the error between the EOAT and its commanded pose. The translation error was computed as the Euclidean norm of the translational component of ${}^{\{eoat\}}\xi_{\{af_i\}}$ and the orientation error was computed as $\phi_3 = arccos(|{}^{\{cbase\}}q_{\{eoat\}} \cdot {}^{\{cbase\}}q_{\{af_i\}}|)$ where q is the quaternion representation of the respective relative poses [117]. Note that ϕ_3 mapped to an angle between 0 and $\frac{\pi}{2}$ rad. A slightly different metric, $\phi_4 = |{}^{\{cbase\}}q_{\{eoat\}} \cdot {}^{\{cbase\}}q_{\{af_i\}}|$, could have been used instead to further improve computational efficiency [117]. If the error was less than a pre-defined threshold[15] or the commanded pose of the EOAT exceeded the 850 mm reach of the manipulator, then the control loop was terminated and either the "stopL()" or "servoStop()" API call was made. For both API calls, a `stopl` script command was forwarded to the manipulator [111, 118, 119] . If the control loop terminated for the former reason, then the arm had successfully reached the commanded pose, and, as has been done in the past, a spiral search trajectory was traced by the EOAT using open-loop control to verify how much, if any, error existed between the commanded EOAT pose and the actual location of the AF [26]. However, if the control loop terminated for the latter reason, control was returned to the AMR-CT (i.e., Fig. 20) so that the manipulator could be re-docked within reachable distance of the CMMA.

Otherwise, the closed manipulator control loop proceeded and the pose, ${}^{\{cbase\}}\xi_{\{af_i\}}$, was converted to an alternative vector format containing the Cartesian position components and orientation components as a rotation vector [120, 121]. The components of this vector were directly passed as arguments to the ur_rtde library, which had API calls, "moveL()", which directly corresponded to a `movel` script command on the manipulator controller, and "servoL()", which corresponded to a `get_inverse_kin` command followed by a `servoj` command on the manipulator controller[16] [111, 118, 119, 122]. Again, from Sec. 3.2, the manipulator controller was expected to be able to execute these commands at a frequency of no greater than 125 Hz, and for the "`servoL()`" API call, the look-ahead time and proportional gain parameters needed to be tuned in a fashion similar to that of the PID controller gains used in the velocity controller (see Appendix C.1). This resulted in the arm being actuated, which changed the pose of the EOAT relative to the CMMA (i.e.,

[15] For the prototype, the threshold was set to between 1 mm and 5 mm distance for the translation and ± 0.0209 rad for the orientation.

[16] Note again that the real-time control example from Ref. [78] will be referred to in the future for refining the control loop.

${}^{\{cmma\}}\xi_{\{eoat\}}$). The OTS would then observe the new pose and the control loop repeated until the error fell below the threshold.

6. Conclusion

In this report, the design of a closed-loop control systems for mobile manipulators that integrates feedback from an OTS was presented. This included a comprehensive overview of the hardware and software specifications for the AMR-CT, the manipulator, the OTS, and the CMMA. The specifications also included the estimated maximum control frequencies, where known and applicable. For example, the AMR-CT was expected to be able to support only event-based pose updates (at a rate of less than 1 Hz), the OTS was expected to acquire data no faster than 180 FPS according to the manufacturer specifications (with further verification of these specs planned to be conducted as per ASTM ASTM E3124-17 [86]), and the manipulator was expected have a maximum control frequency of 125 Hz for velocity-based commands over the robust real-time interface and 20-30 Hz for Cartesian commands over the non-robust real-time interface, again according to manufacturer specification. For the OTS, the marker layouts required for rigid body tracking were provided. Additionally, the network topology and design of the time synchronization using PTP was detailed, with these being critical for evaluating the unknown system latencies and validating the cited system latencies. The 3 DoF and 6 DoF coordinated system transformations between the AMR-CT base, the manipulator cart base, the EOAT, the CMMA and the AFs/commanded docking locations were mapped such that all coordinate system transformations required for the closed-loop control system could be determined. Finally, the design of the software-based closed-loop controllers themselves were detailed, which include two alternate designs for the manipulator controller (i.e., using Cartesian-based pose commands and velocity-based commands determined by a PID controller), as well as two, event-based controllers for the vehicle (one for coarse positioning and one for fine docking).

The implementation of this design as a prototype followed a multi-step plan that started with the simplest performance scenario/design in Table 1 and built up to the most complex scenario/design. Ultimately, the prototype test implementation was found to address each of the desired mobile manipulator capabilities in Table 1, including scenarios in which the CMMA was disturbed continuously within reach of the manipulator. A few additional refinements to the system remain prior to full evaluation. Namely, the prototype was run on a separate laptop as opposed to the SoM. However, all software dependencies were tested to ensure they would run on the SoM. Furthermore, once this is done, the manipulator control code should be modified to run with real-time process priorities, as outlined in Sec. 3.2, to achieve maximum performance. The floor of the lab is also planned to be re-painted with non-reflective paint to minimize reflections caused by the IR emitted by the OTS cameras, as discussed in Sec. 3.4. Finally, also discussed in Sec. 3.4, the measurement uncertainty and latency of the OTS will be re-evaluated using the appropriate ASTM test methods. As the refinements are made, the latencies of the system can be further characterized and used

to make small improvements to the controller design and/or implementation.

Future extensions of this work could expand mobile manipulator closed-loop controller test implementations to mobile manipulators with on-board vision systems (i.e., using the OTS closed-loop controller as a ground-truth reference) or to include more advanced capabilities such as a moving mobile manipulator operating on an also moving target for part hand-off (e.g., an AMR-CT or another mobile manipulator). This latter expansion has grounding in existing research cited in Sec. 2, such as Refs. [43–45]. These advanced test implementations will be used to further validate developing performance test methods for mobile manipulation.

References

[1] Bostelman RV, Hong T, Marvel J (2016) Survey of research for performance measurement of mobile manipulators. *Journal of Research (NIST JRES)* 121:342–366. https://doi.org/10.6028/jres.121.015

[2] Li G, Huang Y, Zhang X, Liu C, Shao W, Jiang L, Meng J (2020) Hybrid maps enhanced localization system for mobile manipulator in harsh manufacturing workshop. *IEEE Access* 8:10782–10795. https://doi.org/10.1109/ACCESS.2020.2965300

[3] Gmerek AJ, Plastropoulos A, Collins P, Kimball M, Amit AW, Liu J, Karfakis PT, K Shah J, Carroll, Virk GS (2018) A novel holonomic mobile manipulator robot for construction sites. *Robotics Transforming the Future – Proceedings of the 21st International Conference on Climbing and Walking Robots and the support Technologies for Mobile Machines (CLAWAR 2018)*, pp 375 – 384.

[4] Bo T, XingWei Z, Han D (2019) Mobile-robotic machining for large complex components: A review study. *SCIENCE CHINA-TECHNOLOGICAL SCIENCES* 62(8, SI):1388–1400. https://doi.org/10.1007/s11431-019-9510-1

[5] Wang J, Tao B, Gong Z, Yu S, Yin Z (2021) A mobile robotic measurement system for large-scale complex components based on optical scanning and visual tracking. *Robotics and Computer-Integrated Manufacturing* 67:102010. https://doi.org/https://doi.org/10.1016/j.rcim.2020.102010. Available at https://www.sciencedirect.com/science/article/pii/S0736584520302210 (Accessed July 17, 2023)

[6] Li F, Jiang Y, Li T (2021) A laser-guided solution to manipulate mobile robot arm terminals within a large workspace. *IEEE/ASME Transactions on Mechatronics* 26(5):2676–2687. https://doi.org/10.1109/TMECH.2020.3044461

[7] Dhanaraj N, Yoon YJ, Malhan R, Bhatt PM, Thakar S, Gupta SK (2022) A mobile manipulator system for accurate and efficient spraying on large surfaces. *Procedia Computer Science* 200:1528–1539. https://doi.org/https://doi.org/10.1016/j.procs.2022.01.354. Available at https://www.sciencedirect.com/science/article/pii/S1877050922003635 (Accessed July 17, 2023) 3rd International Conference on Industry 4.0 and Smart Manufacturing

[8] Zhou K, Ebenhofer G, Eitzinger C, Zimmermann U, Walter C, Saenz J, Castaño LP,

Hernández MAF, Oriol JN (2014) Mobile manipulator is coming to aerospace manufacturing industry. *2014 IEEE International Symposium on Robotic and Sensors Environments (ROSE) Proceedings*, pp 94–99. https://doi.org/10.1109/ROSE.2014.6952990

[9] Bhatt PM, Malhan RK, Shembekar AV, Yoon YJ, Gupta SK (2020) Expanding capabilities of additive manufacturing through use of robotics technologies: A survey. *Additive Manufacturing* 31:100933. https://doi.org/10.1016/j.addma.2019.100933. Available at http://www.sciencedirect.com/science/article/pii/S2214860419312266 (Accessed July 17, 2023)

[10] Sustarevas J, Butters D, Hammid M, Dwyer G, Stuart-Smith R, Pawar VM (2018) MAP - a mobile agile printer robot for on-site construction. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 2441–2448. https://doi.org/10.1109/IROS.2018.8593815

[11] Zhang X, Li M, Lim JH, Weng Y, Tay YWD, Pham H, Pham QC (2018) Large-scale 3D printing by a team of mobile robots. *Automation in Construction* 95:98–106. https://doi.org/https://doi.org/10.1016/j.autcon.2018.08.004. Available at https://www.sciencedirect.com/science/article/pii/S0926580518304011 (Accessed July 17, 2023)

[12] Tiryaki ME, Zhang X, Pham QC (2019) Printing-while-moving: a new paradigm for large-scale robotic 3D printing. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 2286–2291. https://doi.org/10.1109/IROS40897.2019.8967524

[13] Sustarevas J, Kanoulas D, Julier S (2021) Task-consistent path planning for mobile 3D printing. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 2143–2150. https://doi.org/10.1109/IROS51168.2021.9635916

[14] Xu X, Wang Z, Feng C (2021) Projector-guided non-holonomic mobile 3D printing. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp 8039–8045. https://doi.org/10.1109/ICRA48506.2021.9561719

[15] Sustarevas J, Kanoulas D, Julier S (2022) Autonomous mobile 3D printing of large-scale trajectories. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 6561–6568. https://doi.org/10.1109/IROS47612.2022.9982274

[16] (2018) National Institute of Standards and Technology (NIST) robotic systems for smart manufacturing. Available at www.nist.gov/programs-projects/robotic-systems-smart-manufacturing-program (Accessed July 17, 2023).

[17] (2018) National Institute of Standards and Technology (NIST) measurement science for manufacturing robotics. Available at https://www.nist.gov/programs-projects/measurement-science-manufacturing-robotics (Accessed July 17, 2023).

[18] (2018) National Institute of Standards and Technology (NIST) mobility performance of robotic systems. Available at https://www.nist.gov/programs-projects/mobility-performance-robotic-systems (Accessed July 17, 2023).

[19] Bostelman RV, Aboul-Enein O, Yoon S, Li-Baboud YS (2022) Design and appli-

cation of the Reconfigurable Mobile Manipulator Artifact (RMMA) (100 Bureau Drive Gaithersburg, MD 20899), https://doi.org/https://doi.org/10.6028/NIST.AMS.100-46. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=933475 (Accessed July 17, 2023)

[20] Bostelman RV, Hong T, Marvel JA (2015) Performance measurement of mobile manipulators. *Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2015*, ed Braun JJ International Society for Optics and Photonics (SPIE), Vol. 9498, pp 97 – 106. https://doi.org/10.1117/12.2177344. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=918268 (Accessed July 17, 2023)

[21] Bostelman RV, Foufou S, Legowik SA, Hong TH (2016) Mobile manipulator performance measurement towards manufacturing assembly tasks. *Proceedings of the 13th IFIP International Conference on Product Lifecycle Management (PLM16)*, eds Harik R, Rivest L, Bernard A, Eynard B, Bouras A (Springer International Publishing), Vol. 492, pp 411–420. https://doi.org/10.1007/978-3-319-54660-5_37. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=920470 (Accessed July 17, 2023)

[22] Bostelman RV, Hong T, Legowik S (2016) Mobile robot and mobile manipulator research towards ASTM standards development. *Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2016*, ed Braun JJ International Society for Optics and Photonics (SPIE), Vol. 9872, pp 111 – 120. https://doi.org/10.1117/12.2228464

[23] Bostelman RV, Eastman R, Hong TH, Aboul-Enein O, Legowik SA, Foufou S (2016) Comparison of registration methods for mobile manipulators. *Advances in Cooperative Robots*, eds Tokhi MO, Virk GS (World Scientific), pp 205–213. https://doi.org/10.1142/9789813149137_0026. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=921002 (Accessed July 17, 2023)

[24] Bostelman RV, Li-Baboud YS, Legowik SA, Foufou S (2017) Mobile manipulator performance measurement data (National Institute of Standards and Technology (NIST)), https://doi.org/10.6028/NIST.TN.1965

[25] Bostelman RV, Li-Baboud Y, Yoon S, Shah M, Aboul-Enein O (2020) Towards measurement of advanced mobile manipulator performance for assembly applications (100 Bureau Drive Gaithersburg, MD 20899), https://doi.org/10.6028/NIST.TN.2108

[26] Aboul-Enein O, Bostelman R, Li-Baboud YS, Shah M (2022) Performance measurement of a mobile manipulator-on-a-cart and coordinate registration methods for manufacturing applications (100 Bureau Drive Gaithersburg, MD 20899), https://doi.org/https://doi.org/10.6028/NIST.AMS.100-45r1. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=934492 (Accessed July 17, 2023)

[27] Amoako-Frimpong SY, Messina M, Medeiros H, Marvel J, Bostelman R (2018) Stochastic search methods for mobile manipulators. *Procedia Manufacturing* 17:976–984. https://doi.org/10.1016/j.promfg.2018.10.106. Available at https:

//www.sciencedirect.com/science/article/pii/S235197891831223X (Accessed July 17, 2023) 28th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2018), June 11-14, 2018, Columbus, OH, USAGlobal Integration of Intelligent Manufacturing and Smart Industry for Good of Humanity

[28] Jiayu X (2021) *Stochastic Modeling for Mobile Manipulators* Master's thesis Marquette University 1515 W. Wisconsin Ave. Milwaukee, WI 53233. Available at https://epublications.marquette.edu/theses_open/687/ (Accessed July 17, 2023).

[29] Aboul-Enein O, Medeiros H, Shah M, Li-Baboud YS, Bostelman R, Virts A Continuous mobile manipulator performance measurement data. Publication pending–title subject to change.

[30] Bostelman RV, Falco JA, Shah M, Hong TH (2016) *Autonomous Industrial Vehicles: From the Laboratory to the Factory Floor* (ASTM International), Chapter Dynamic Metrology Performance Measurement of a Six Degree-Of-Freedom Tracking System used in Smart Manufacturing, pp 91–105. https://doi.org/10.1520/STP1594-EB

[31] Godil A, Bostelman R, Saidi K, Shackleford W, Cheok G, Shneier M, Hong T (2013) 3D ground-truth systems for object/human recognition and tracking. *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp 719–726. https://doi.org/10.1109/CVPRW.2013.109

[32] Norman AR, Schoenberg A, Gorlach IA, Schmitt R (2013) Validation of iGPS as an external measurement system for cooperative robot positioning. *INTERNATIONAL JOURNAL OF ADVANCED MANUFACTURING TECHNOLOGY* 64(1-4):427–446. https://doi.org/10.1007/s00170-012-4004-8

[33] Mosqueira G, Apetz J, Santos K, Villani E, Suterio R, Trabasso L (2012) Analysis of the indoor GPS system as feedback for the robotic alignment of fuselages using laser radar measurements as comparison. *Robotics and Computer-Integrated Manufacturing* 28(6):700–709. https://doi.org/https://doi.org/10.1016/j.rcim.2012.03.004. Available at https://www.sciencedirect.com/science/article/pii/S0736584512000373 (Accessed July 17, 2023)

[34] Maisano D, Jamshidi J, Franceschini F, Maropoulos P, Mastrogiacomo L, Mileham A, Owen G (2008) Indoor GPS: System functionality and initial performance evaluation. *Int J Manufacturing Research* 3. https://doi.org/10.1504/IJMR.2008.019214

[35] Flannigan C (2012) Mobile manipulation robotics & automation engineering, Southwest Research Institute. Available at http://www.swri.org/4org/d10/msd/automation/mobile-manipulator.htm (Accessed July 17, 2023).

[36] Shneier M, Bostelman R (2014) Literature review of mobile robots for manufacturing (National Institute of Standards and Technology (NIST), 100 Bureau Drive Gaithersburg, MD 20899), https://doi.org/10.6028/NIST.IR.8022

[37] Bartlett G, Hvass P (2011) SwRI internal research and development 2011: Metrology referenced roving accurate manipulator phase 2 (MR ROAM 2), 10-r8205. Available at https://www.swri.org/sites/default/files/IRD2011.pdf (Accessed July 17, 2023).

[38] Bartlett G, Hvass P (2012) SwRI internal research and development 2012: Metrol-

ogy referenced roving accurate manipulator phase 2 (MR ROAM 2), 10-r8205. Available at https://www.swri.org/sites/default/files/IRD2012.pdf (Accessed July 17, 2023).

[39] Bartlett G, Hvass P (2013) Large-scale mobile robotic manipulation. Available at https://www.youtube.com/watch?v=K0Ol5vv8tsE (Accessed July 17, 2023) Youtube Video.

[40] Meng X, He Y, Gu F, Yang L, Dai B, Liu Z, Han J (2016) Design and implementation of rotor aerial manipulator system. *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp 673–678. https://doi.org/10.1109/ROBIO.2016.7866400

[41] Vasilopoulos V, Topping TT, Vega-Brown W, Roy N, Koditschek DE (2018) Sensor-based reactive execution of symbolic rearrangement plans by a legged mobile manipulator. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 3298–3305. https://doi.org/10.1109/IROS.2018.8594342

[42] Ubezio B, Sharma S, der Meer GV, Taragna M (2019) Kalman filter based sensor fusion for a mobile manipulator. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. Volume 5A: 43rd Mechanisms and Robotics Conference. https://doi.org/10.1115/DETC2019-97241. Available at https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/IDETC-CIE2019/59230/V05AT07A043/6453506/v05at07a043-detc2019-97241.pdf (Accessed July 17, 2023) V05AT07A043.

[43] Schierl A, Angerer A, Hoffmann A, Vistein M, Reif W (2016) On structure and distribution of software for mobile manipulators. *Informatics in Control, Automation and Robotics 12th International Conference, ICINCO 2015 Colmar, France, July 21-23, 2015 Revised Selected Papers*, eds Filipe J, Madani K, Gusikhin O, Sasiadek J (Springer International Publishing, Cham), pp 209–227.

[44] Schierl A (2017) *Object-Oriented Modeling and Coordination of Mobile Robots*. Doctoral thesis Universität Augsburg, Universität Augsburg, Universitätsstr. 2, 86159 Augsburg. Available at https://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/index/index/docId/3915 (Accessed July 17, 2023).

[45] Obregón-Flores J, Arechavaleta G, Becerra HM, Morales-Díaz A (2021) Predefined-time robust hierarchical inverse dynamics on torque-controlled redundant manipulators. *IEEE Transactions on Robotics* 37(3):962–978. https://doi.org/10.1109/TRO.2020.3042054

[46] Kuka youBot specifications. Available at https://www.generationrobots.com/img/Kuka-YouBot-Technical-Specs.pdf (Accessed July 17, 2023).

[47] Dong K, Pereida K, Shkurti F, Schoellig AP (2020) Catch the ball: Accurate high-speed motions for mobile manipulators via inverse dynamics learning. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp 6718–6725. https://doi.org/10.1109/IROS45743.2020.9341134

[48] Yao Q, Terakawa T, Morita Y, Komori M (2023) Mobile parallel manipulator

consisting of two nonholonomic carts and their path planning. *Journal of Advanced Mechanical Design, Systems, and Manufacturing* 17(2). https://doi.org/10.1299/jamdsm.2023jamdsm0020

[49] Tao B, Zhao X, Yan S, Ding H (2022) Kinematic modeling and control of mobile robot for large-scale workpiece machining. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 236(1-2):29–38. https://doi.org/10.1177/0954405420933708

[50] (2023) Raspberry Pi Compute Module 4 datasheet. Available at https://datasheets.raspberrypi.com/cm4/cm4-datasheet.pdf (Accessed July 17, 2023).

[51] (2022) Raspberry Pi Compute Module 4 IO board datasheet. Available at https://datasheets.raspberrypi.com/cm4io/cm4io-datasheet.pdf (Accessed July 17, 2023).

[52] (2020) LD series mobile robot datasheet. Available at https://assets.omron.com/m/1bf71ca06f961b4b/original/LD-Series-Mobile-Robot-Datasheet.pdf (Accessed July 17, 2023).

[53] Omron Adept Technologies 4225 Hacienda Drive, Pleasanton, CA 94588 (2016) *Mobile Robot Software Suite User's Guide*, I614-E-01 Ed. Available at https://assets.omron.com/m/0c55a28223f13734/original/Mobile-Robot-Software-Suite-User-s-Guide.pdf (Accessed July 17, 2023).

[54] Omron Adept Technologies 4225 Hacienda Drive, Pleasanton, CA 94588 (2021) *LD Platform Peripherals User Guide*, I613-E-03 Ed. Available at https://assets.omron.com/m/5f7d3ae7376db288/original/LD-Platform-Peripherals-User-s-Guide.pdf (Accessed July 17, 2023).

[55] Omron Adept Technologies 4225 Hacienda Drive, Pleasanton, CA 94588 (2016) *Advanced Robotics Command Language Reference Guide*, I617-E-01 Ed. Available at https://assets.omron.com/m/6f13d7c70932964d/original/Advanced-Robotics-Command-Language-Reference-Guide.pdf (Accessed July 17, 2023).

[56] Proctor F (2019) ULAPI download. Available at https://github.com/frederickproctor/ulapi (Accessed July 17, 2023).

[57] ptitSeb (2022) Box86 Linux userspace x86 emulator with a twist. Available at https://box86.org/ (Accessed July 17, 2023).

[58] ptitSeb (2022) Box64 GitHub. Available at https://github.com/ptitSeb/box64 (Accessed July 17, 2023).

[59] ptitSeb (2022) Box64 GitHub: Compile/installation. Available at https://github.com/ptitSeb/box64/blob/main/docs/COMPILE.md (Accessed July 17, 2023).

[60] (2016) Universal Robots UR5 technical specifications. Available at https://www.universal-robots.com/media/50588/ur5_en.pdf (Accessed July 17, 2023).

[61] (2022) WORLD-BEAM® QS18LLP series datasheet. Available at http://info.bannerengineering.com/cs/groups/public/documents/literature/118900.pdf (Accessed July 17, 2023).

[62] (2016) Pro 3600 owner's manual. Available at https://www.leveldevelopments.com/wp/wp-content/uploads/documents/PRO3600_Instructions.pdf (Accessed July 17,

2023).
[63] (2010) ML900-24 specification sheet. Available at http://powerbright.com/manuals/specs/ML900_spec_sheet.pdf (Accessed July 17, 2023).
[64] (2011) XE and EP batteries range summary. Available at https://www.enersys.com/493c0d/globalassets/documents/product-documentation/genesis/pure-lead/apac/en-xe-ep-rs-005_0911.pdf (Accessed July 17, 2023).
[65] (2006) UL(AC), DL(DC) series datasheet. Available at https://assets.alliedelec.com/v1620727753/Datasheets/b181f9d764d2f836322247b876112d39.pdf (Accessed July 17, 2023).
[66] (2022) Overview of client interfaces. Available at https://www.universal-robots.com/articles/ur/interface-communication/overview-of-client-interfaces/ (Accessed July 17, 2023).
[67] (2015) Remote control via TCP/IP. Available at https://www.universal-robots.com/articles/ur/interface-communication/remote-control-via-tcpip/ (Accessed July 17, 2023).
[68] (2019) Real-Time Data Exchange guide. Available at https://www.universal-robots.com/articles/ur/interface-communication/real-time-data-exchange-rtde-guide/ (Accessed July 17, 2023).
[69] Marvel J, Zimmerman M, Bagchi S, Wyk KV, Kumbla N, Antonishek B, Aboul-Enein O (2022) Collaborative Robot Programming Interface download. Available at https://github.com/usnistgov/CRPI (Accessed July 17, 2023).
[70] Shackleford W (2009) Real-time Control System software and documentation. Available at https://www.nist.gov/ctl/smart-connected-systems-division/networked-control-systems-group/real-time-control-systems (Accessed July 17, 2023).
[71] Shackleford W (2014) RCS library installation instructions. Available at https://www.nist.gov/ctl/smart-connected-systems-division/networked-control-systems-group/rcs-library-installation (Accessed July 17, 2023).
[72] Shackleford W (2020) rcslib GitHub. Available at https://github.com/usnistgov/rcslib/blob/master/README (Accessed July 17, 2023).
[73] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) ur_rtde source code. Available at https://gitlab.com/sdurobotics/ur_rtde (Accessed July 17, 2023).
[74] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) ur_rtde documentation introduction. Available at https://sdurobotics.gitlab.io/ur_rtde/introduction/introduction.html (Accessed July 17, 2023).
[75] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) ur_rtde documentation FAQ. Available at https://sdurobotics.gitlab.io/ur_rtde/faq/faq.html (Accessed July 17, 2023).
[76] (2023) Raspberry Pi Linux kernel. Available at https://github.com/raspberrypi/linux/

tree/rpi-5.15.y (Accessed July 17, 2023).
[77] (2022) PREEMPT-RT kernel patch mirror. Available at https://cdn.kernel.org/pub/linux/kernel/projects/rt/5.15/older/ (Accessed July 17, 2023).
[78] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) ur_rtde documentation guides. Available at https://sdurobotics.gitlab.io/ur_rtde/guides/guides.html (Accessed July 17, 2023).
[79] (2022) Raspberry Pi documentation: The Linux kernel. Available at https://www.raspberrypi.com/documentation/computers/linux_kernel.html (Accessed July 17, 2023).
[80] Windynation LIN-ACT1-XX linear actuator manual. Available at https://cdn.shopify.com/s/files/1/0636/9289/8526/files/Actuator_Manual_R3.pdf?v=1666281336 (Accessed July 17, 2023).
[81] Bostelman RV (2018) *Performance measurement of mobile manipulators*. Ph.D. thesis. Université Bourgogne Franche-Comté, 32 Av. de l'Observatoire, 25000 Besançon, France. Available at https://tel.archives-ouvertes.fr/tel-01803721 (Accessed July 17, 2023).
[82] Aboul-Enein O, Jing Y, Bostelman R (2020) Formalizing performance evaluation of mobile manipulator robots using CTML. *Proceedings of the ASME 2020 International Mechanical Engineering Congress and Exposition (IMECE 2020)* (ASME), Vol. Volume 2B: Advanced Manufacturing. https://doi.org/10.1115/IMECE2020-23234. Available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=928432 (Accessed July 17, 2023) V02BT02A004.
[83] (2012) Prime 41 data sheet. Available at https://d111srqycjesc9.cloudfront.net/Prime%2041%20Data%20Sheet.pdf (Accessed July 17, 2023).
[84] (2022) Motive documentation latency measurements. Available at https://docs.optitrack.com/developer-tools/natnet-sdk/latency-measurements (Accessed July 17, 2023).
[85] (2022) Motive documentation status panel. Available at https://docs.optitrack.com/motive-ui-panes/status-panel (Accessed July 17, 2023).
[86] (2022) ASTM E3124-17: Standard test method for measuring system latency performance of optical tracking systems that measure six degrees of freedom (6DOF) pose. https://doi.org/10.1520/E3124-17. in Book of Standards Volume 10.04 and developed by subcommittee E57.50.
[87] (2022) Motive documentation prepare setup area. Available at https://docs.optitrack.com/hardware/prepare-setup-area (Accessed July 17, 2023).
[88] (2022) Motive documentation calibration. Available at https://docs.optitrack.com/motive/calibration (Accessed July 17, 2023).
[89] (2015) CW-500 technical drawing. Available at https://d111srqycjesc9.cloudfront.net/CW-500TechnicalDrawing.pdf (Accessed July 17, 2023).
[90] (2022) Motive documentation continuous calibration. Available at https://docs.optitrack.com/motive/calibration/continuous-calibration (Accessed July 17, 2023).
[91] (2022) Motive download. Available at https://optitrack.com/support/downloads/

motive.html (Accessed July 17, 2023).
[92] (2022) ASTM E3064-16: Standard test method for evaluating the performance of optical tracking systems that measure six degrees of freedom (6DOF) pose. https://doi.org/10.1520/E3064-16. in Book of Standards Volume 10.04 and developed by subcommittee E57.50
[93] (2022) Intel Xeon W-2145 processor specifications. Available at https://www.intel.com/content/www/us/en/products/sku/126707/intel-xeon-w2145-processor-11m-cache-3-70-ghz/specifications.html (Accessed July 17, 2023).
[94] (2019) NVIDIA Titan RTX product overview. Available at https://www.nvidia.com/content/dam/en-zz/Solutions/titan/documents/titan-rtx-for-creators-us-nvidia-1011126-r6-web.pdf (Accessed July 17, 2023).
[95] (2022) NatNet SDK 4.0. Available at https://docs.optitrack.com/developer-tools/natnet-sdk/natnet-4.0 (Accessed July 17, 2023).
[96] (2022) NatNet SDK 4.0 download. Available at https://optitrack.com/support/downloads/developer-tools.html#natnet-sdk (Accessed July 17, 2023).
[97] Matthew N, Stones R (2008) *Beginning Linux Programming* (Wiley Publishing), Chapter 14, 4th Ed., pp 577–606.
[98] (2016) Introduction to real-time systems. Available at https://design.ros2.org/articles/realtime_background.html (Accessed July 17, 2023).
[99] (2023) Understanding Linux real-time with PREEMPT_RT training. Available at https://bootlin.com/doc/training/preempt-rt/preempt-rt-slides.pdf (Accessed July 17, 2023).
[100] (2023) Motive documentation data streaming. Available at https://docs.optitrack.com/motive/data-streaming (Accessed July 17, 2023).
[101] (2022) NatNet class/function reference. Available at https://docs.optitrack.com/developer-tools/natnet-sdk/natnet-class-function-reference (Accessed July 17, 2023).
[102] Welford BP (1962) Note on a method for calculating corrected sums of squares and products. *Technometrics* 4:419–420. https://doi.org/10.1080/00401706.1962.10490022
[103] (2023) Motive documentation data export. Available at https://docs.optitrack.com/motive/data-export (Accessed July 17, 2023).
[104] (2023) Motive documentation data streaming pane. Available at https://docs.optitrack.com/v/v2.3/motive-ui-panes/settings-streaming (Accessed July 17, 2023).
[105] (2022) NatNet sample projects. Available at https://docs.optitrack.com/developer-tools/natnet-sdk/natnet-sample-projects (Accessed July 17, 2023).
[106] (2009) Real-Time Control System library download. Available at https://www.nist.gov/el/intelligent-systems-division-73500/networked-control-systems-group/real-time-control-systems (Accessed July 17, 2023).
[107] Corke P (2017) *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer International Publishing AG), Chapter 2, 2nd Ed., pp 44–45.

https://doi.org/10.1007/978-3-319-54413-7

[108] Corke P (2017) *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer International Publishing AG), Chapter 2, 2nd Ed., p 47. https://doi.org/10.1007/978-3-319-54413-7

[109] (2022) Motive documentation markers. Available at https://docs.optitrack.com/motive/markers (Accessed July 17, 2023).

[110] (2022) Motive documentation rigid body tracking. Available at https://docs.optitrack.com/motive/rigid-body-tracking (Accessed July 17, 2023).

[111] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) ur_rtde documentation API reference. Available at https://sdurobotics.gitlab.io/ur_rtde/api/api.html (Accessed July 17, 2023).

[112] Shah M (2011) Comparing two sets of corresponding six degree of freedom data. *Computer Vision and Image Understanding* 115(10):1355 – 1362. https://doi.org/https://doi.org/10.1016/j.cviu.2011.05.007. Available at https://www.sciencedirect.com/science/article/pii/S1077314211001378 (Accessed July 17, 2023)

[113] Shah M, Eastman RD, Hong T (2012) An overview of robot-sensor calibration methods for evaluation of perception systems. *Proceedings of the Workshop on Performance Metrics for Intelligent Systems* PerMIS '12 (Association for Computing Machinery, New York, NY, USA), p 15–20. https://doi.org/10.1145/2393091.2393095

[114] Shah M (2014) Calibration and registration techniques for robotics. Available at http://faculty.cooper.edu/mili/Calibration/index.html (Accessed July 17, 2023).

[115] Arun K, Huang T, Blostein S (1987) Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9(5):698–700. https://doi.org/10.1109/TPAMI.1987.4767965

[116] Shah M (2014) Arun.m source code. Available at http://faculty.cooper.edu/mili/Calibration/RT/arun.m (Accessed July 17, 2023).

[117] Huynh DQ (2009) Metrics for 3D rotations: Comparison and analysis. *Journal of Mathematical Imagine and Vision* 35:155–164. https://doi.org/10.1007/s10851-009-0161-2

[118] Lindvig AP, Nowak C, Iturrate I, Lorenzen KH, Sorensen K, Sorensen LC, Lukas, Denisa M, Simon, Thulesen TN (2022) rtde_control.script source code. Available at https://gitlab.com/sdurobotics/ur_rtde/-/blob/master/scripts/rtde_control.script (Accessed July 17, 2023).

[119] Universal Robots A/S Energivej 25, DK-5260 Odense, Denmark (2016) *Script manual - CB-Series - SW3.3*, 3.3.4 Ed. Available at https://s3-eu-west-1.amazonaws.com/ur-support-site/22198/scriptManual.pdf (Accessed July 17, 2023).

[120] Corke P (2017) *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer International Publishing AG), Chapter 2, 2nd Ed., pp 41–42. https://doi.org/10.1007/978-3-319-54413-7

[121] Carlo T (2013) Vector representation of rotations. Available at https://courses.cs.duke.edu/fall13/compsci527/notes/rodrigues.pdf (Accessed July 17, 2023).

[122] Universal Robots A/S Energivej 25, DK-5260 Odense, Denmark (2021) *Script man-*

ual - E-Series - SW5.11, 5.11 Ed. Available at https://s3-eu-west-1.amazonaws.com/ur-support-site/22198/scriptManual.pdf (Accessed July 17, 2023).

[123] Corke P (2017) *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer International Publishing AG), Chapter 2, 2nd Ed., pp 38–40. https://doi.org/10.1007/978-3-319-54413-7

[124] Hauser K (2018) 3D rotations. Available at motion.pratt.duke.edu/RoboticSystems/3DRotations.html (Accessed July 17, 2023).

[125] Corke P (2017) *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (Springer International Publishing AG), Chapter 9, 2nd Ed., p 261. https://doi.org/10.1007/978-3-319-54413-7

[126] Ellis G (2012) *Control System Design Guide* (Elsevier Inc), Chapter 6, 4th Ed., pp 97–119.

[127] Ellis G (2012) *Control System Design Guide* (Elsevier Inc), Chapter 8, 4th Ed., pp 145–163.

[128] Tilbury D, Messner B, Hill R, Taylor J, Das S (2021) Introduction: PID controller design. Available at https://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID#25 (Accessed July 17, 2023).

[129] Ellis G (2012) *Control System Design Guide* (Elsevier Inc), Chapter 5, 4th Ed., pp 73–95.

[130] Case JC, White EL, Kramer RK (2016) Sensor enabled closed-loop bending control of soft beams. *Smart Materials and Structures* 25(4):045018. https://doi.org/10.1088/0964-1726/25/4/045018

Appendix A. List of Symbols, Abbreviations, and Acronyms

Appendix A.1. Variable Conventions

Variable definitions use the following conventions. Scalars are denoted by lowercase Latin or Greek letters with no arrow (e.g., i and x). Vectors are denoted by lowercase Latin letters with an arrow (e.g., $\vec{f}_1$), and matrices are denoted by capital Latin letters, with H specifically used to denote a homogeneous transformation matrix. For matrices and vectors, the applicable coordinate frames(s), if any, are denoted as either a superscript or subscript in curly braces (e.g., ${}^{\{OTS\}}\vec{v}_i$ and ${}^{\{\mathbf{OTS}\}}\xi_{\{\mathbf{cmma}\}}$). Otherwise, an arbitrary, Cartesian coordinate system is assumed. The Greek letter ξ denotes a pose in an arbitrary format. Boldface is used to denote potentially unknown vectors or matrices in calibration problems (e.g., $\mathbf{t}_i$ and $\mathbf{R}_i$).

Appendix A.2. Acronyms

2D Two-Dimensional. 9, 31, 33

3D Three-Dimensional. 15, 24, 31

AF Assembly Fiducial. iii, iv, 19, 27, 30, 33, 38, 55

AFs Assembly Fiducials. iv, 16, 27, 34, 39

AGV Automatic Guided Vehicle. 2, 3, 5, 9

AM Additive Manufacturing. 1, 7

AMR Autonomous Mobile Robot. 2, 5, 9

AMR-CT Autonomous Mobile Robot Cart Transporter. iii, iv, 2, 4, 8–14, 18, 21, 22, 25–27, 31–35, 37–40, 55

API Application Programming Interface. 22, 27, 37, 38, 53

ARC average rate-of-change. 55

CMMA Configurable Mobile Manipulator Apparatus. ii–iv, 1–6, 8, 9, 14–19, 23–25, 27–35, 37–39, 55

CNC Computer Numerical Control. 1

CPU central processing unit. 14, 21

CRPI Collaborative Robot Programming Interface. 15, 16

CSV Comma Separated Value. 23

DNN Deep Neural Network. 7

DoF Degrees of Freedom. iii, iv, 3–5, 22, 23, 25–33, 37, 39, 55

eMMC Embedded Multi-Media Card. 8

EOAT End-of-Arm Tool. iii, iv, 2, 15, 23–30, 33, 34, 38, 39, 53–55, 57

FK forward kinematics. 6

FOV field-of-view. 17, 19

FPS Frames-per-Second. iii, 17, 20, 34, 39

GPS Global Positioning System. iii, 5, 11

I/O Input/Output. 8

iGPS Indoor Global Positioning System. 5

IIoT industrial internet of things. 25

IMU inertial measurement unit. 6

IPC Inter-process Communications. 21

IR Infrared. 19, 23, 39

MP Megapixel. 17

MPC model predictive controller. 7

mutexes mutual exclusions. 14, 15

NIST National Institute of Standards and Technology. v, 1, 3–5, 15, 23

NTP Network Time Protocol. 25

OS operating system. 16

OTS Optical Tracking System. iii–v, 2–9, 11, 13–31, 33–36, 39, 40, 53–56

P Proportional. 57

P/I Proportional/Integral. 8, 57

PC personal computer. 8

PCIe Peripheral Component Interconnect Express. 26

PID Proportional-Integral-Derivative. iv, v, 8, 30, 34, 38, 39, 53, 55–57

POE Power over Ethernet. 8

pose position and orientation. iii, iv, 1–6, 9, 14, 21–31, 33–39, 53, 55, 57

PTP Precision Time Protocol. iii, 11, 26, 39

QP Quadratic Programming. 7

RAM Random Access Memory. 8, 21

RCS Real-time Control System. 15, 23

RLS Retro-reflective Laser Sensor and Emitter. iii, iv, 15, 16, 19, 27, 29, 34, 38

RMMA Re-configurable Mobile Manipulator Artifact. 1, 2, 16

RPY roll-pitch-yaw. 53

SDK software development kit. iii, 20–23, 34

SLAM simultaneous localization and mapping. 7

SoM System on Module. 8, 9, 14–16, 21, 34, 39

SQP Sequential Quadratic Programming. 7

TCP tool-center-point. 34, 53

TCP/IP Transmission Control Protocol / Internet Protocol. 13–15

TCPP Task-Consistent Path Planning. 7

UDP User Datagram Protocol. 21, 34

ULAPI User-Level Application Programming Interface. 14, 15

UTC Coordinated Universal Time. iii, 11

Appendix B. Updated EOAT Mount Technical Drawing

The engineering design drawing for the updated EOAT mount, which was modified to include additional mounting holes for OTS markers is presented in Fig. 22.

Appendix C. Alternate Velocity-based PID Manipulator Controller Design

As mentioned in Sec. 5.2, during the process of designing the closed-loop manipulator pose controller, an alternative design utilizing a PID controller and linear velocity commands that would control the manipulator in linear Cartesian space relative to the TCP was considered. The alternate controller design is presented in Fig. 23.

Picking up from the steps outlined at the beginning of Sec. 5, the Velocity controller executes as follows. The controller would compute the relative pose it needs in order to compute the error between the measured EOAT pose and the commanded pose (i.e., ${}^{\{eoat\}}\xi_{\{af_i\}}$). Note that this relative pose would also be the only one needed to control the arm since the "speedL()" API call/script command controls the manipulator relative to the TCP in linear Cartesian space [111, 119]. Additionally, the pose ${}^{\{cbase\}}\xi_{\{eoat\}}$ would be needed, just to ensure the EOAT does not exceed the joint limits of the robot (i.e., that the commanded pose would be kept within the reach of the manipulator). The controller then would convert the orientation of the relative pose into the RPY[17] representation. The translational error would be computed as the Euclidean norm of the translational component of ${}^{\{eoat\}}\xi_{\{af_i\}}$ and rotation error would be computed as the component-wise difference of the RPY angles. If all of the errors are less than a pre-defined threshold or the commanded

[17] To avoid singularities and non-unique representations, the rotation of the EOAT relative to the manipulator cart base could be limited to $\pm\frac{\pi}{2}$, non-inclusive [123, 124]

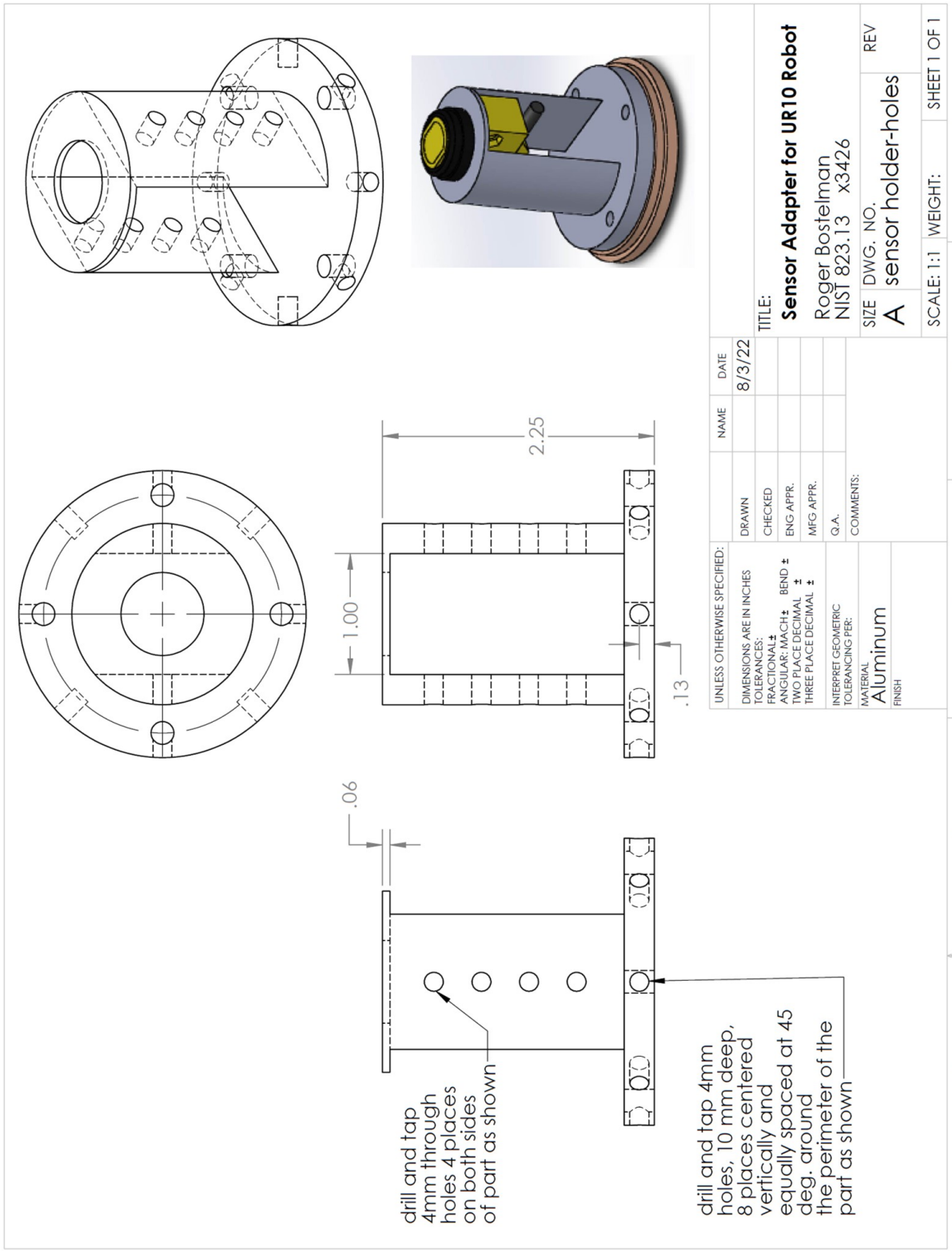

Fig. 22. Technical drawing for updated EOAT mount with additional holes for fixturing OTS markers.

pose of the EOAT is not within reach, then the control loop would terminated. If the control loop terminates for the former reason, then, as in the Cartesian/linear servo controller in Sec. 5.2, the arm would have successfully reached the commanded pose, and, as has been done in the past, a spiral search trajectory would be traced by the EOAT using open-loop control to verify how much, if any, error exists between the commanded EOAT pose and the actual location of the AF [26]. However, if the control loop terminates for the latter reason, as in the Cartesian/linear servo controller in Sec. 5.2, control would be returned to the AMR-CT (i.e., Fig. 20) so that the manipulator can be re-docked within reachable distance of the CMMA.

Otherwise, the control loop would proceed and the component-wise pose error values, denoted e_x, e_y, e_z e_{roll}, e_{pitch}, and e_{yaw}, would be each passed to a separate PID controller (more details in Sec. C.1) each having their own separate separate controller gains. Each set of controller gains could be tuned online using the strategy outlined in Sec. C.1. The PID controller would then issue a `speedl` command to the manipulator controller [119]. Using the ur_rtde library, the manipulator should be able to execute these commands at a rate of 125 Hz. This would result in the arm being actuated, which would also change the pose of the EOAT relative to the CMMA (i.e., ${}^{\{cmma\}}\xi_{\{eoat\}}$). The OTS would then observe the new pose and the control loop would repeat until the error lowers below the threshold.

Appendix C.1. PID Controller and Gain Tuning

The PID controller design was explored as an alternative choice to implement velocity-based position control of the EOAT due its ubiquitous use in control systems and for its simple implementation [125, 126]. However, depending on the magnitude of the delay variability after implementation, more complex controller designs, such as feed-forward controllers may need to be considered to further improve responsiveness [127]. The equation for a generic PID controller in the time domain is given by: $u(t) = k_p e(t) + k_i \int e(t)dt + k_d \frac{de}{dt}$, where $u(t)$ denoted the control signal (in this case the component-wise linear velocities used to control the manipulator), k_p, k_i, and k_d denoted the proportional, integral, and derivative controller gains, respectively, and $e(t)$ denoted the error signal measured using the OTS [128]. The proportional term can scale the control signal with respect to the error signal, the integral term can further compensate for steady-state error, and the derivative term can be used to further tune controller responsiveness [126, 128]. Furthermore, the derivative term can be approximated using an average rate-of-change (ARC) between two error signal samples across adjacent time steps, while the integral term can be approximated using one of several common approximations depending on accuracy needs (i.e., Riemann sums, the Trapezoidal rule, or Simpson's rule) [129].

A total of 9 controller gains for the 3 DoF case and 18 controller gains for the 6 DoF case (i.e., separate proportional, integral, and derivative gains for up to six separate PID controllers, or one controller per pose component) would be adjusted using a combination of an online, manual tuning strategy based on that from Ref. [130] and the Ziegler-Nichols

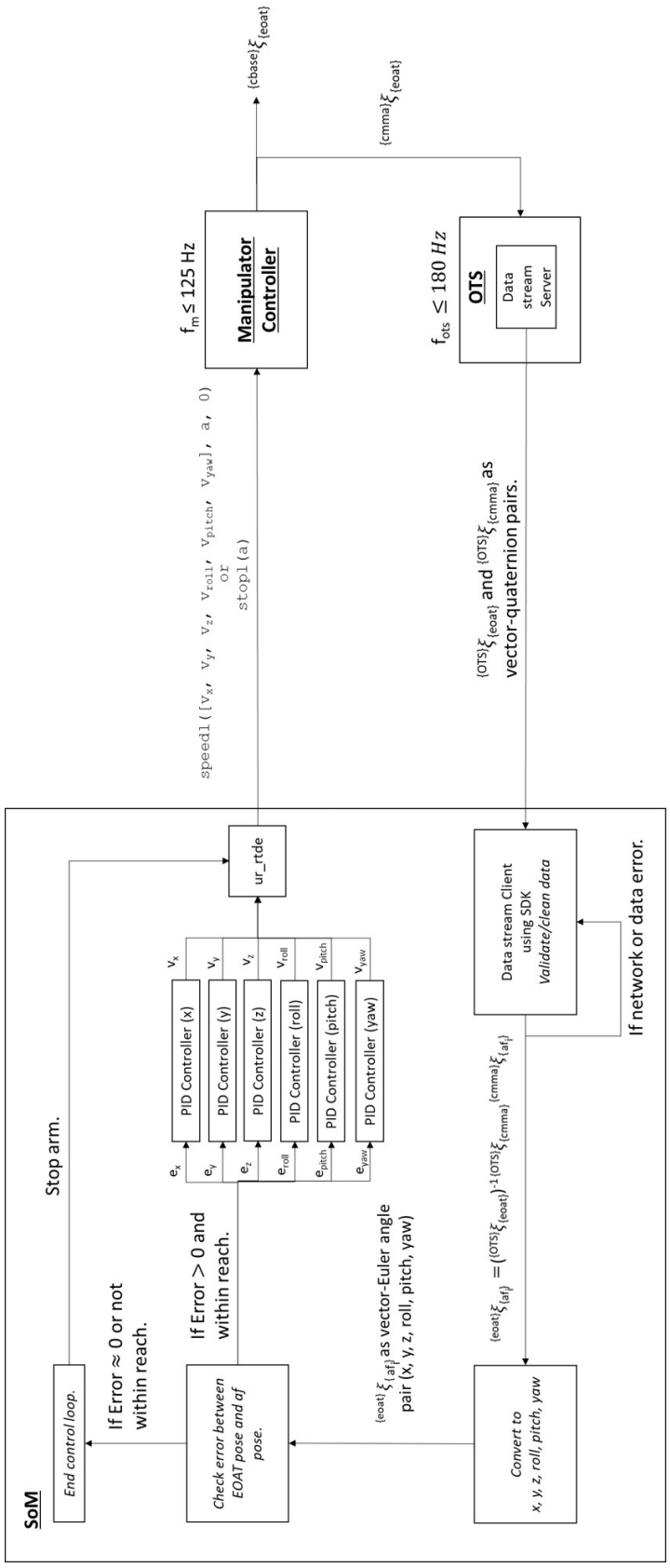

Fig. 23. Diagram showing the alternate manipulator closed-loop controller design, which utilizes a PID controller and velocity commands. Note that f_m and f_{OTS} denote the control frequencies of the manipulator controller and OTS, respectively.

method [126]. Gains belonging to separate controllers can be tuned independently, but simultaneously using this strategy. For example, the proportional gain for the x component PID controller can be adjusted simultaneously with the separate proportional gain for the y component PID controller, etc. First, the proportional gains would be set to a small, arbitrary value while the integral and derivative gains would be held constant at zero. The maximum velocity command outputted by the controller would also be limited to a safe speed, and the frequency of the PID controllers would be set to a fixed value that falls comfortably within the limitations of the system latency. A tuning iteration would then consist of using the PID controllers to position the arm to an arbitrary pose for three trials. The performance of each controller for each trial would be measured by monitoring the cumulative absolute error among each pose component as the manipulator EOAT reaches the commanded pose. Optimizing the PID controller can therefore be done by adjusting the gain value to minimize the average cumulative error across all three trials for each training iteration. It is expected that, as the proportional gain is increased, the average cumulative error would decrease across each PID controller until target overshoot or sustained oscillation is observed. The gain value and average cumulative error would be recorded for each training iteration, and the gain value adjusted until satisfactory performance is achieved. Furthermore, the oscillation frequency, which, if oscillation occurs, can be computed by detecting how fast the error changes from positive to negative. The oscillation frequency, combined with the value of the proportional gains when instability occurs can then be used to fractionally scale the proportional gain and set the Integral and Derivative gains using the Ziegler-Nichols method [126]. If the proportional gain or proportional and integral gains alone result in overall satisfactory stability and responsiveness, the remaining gains would be left at zero, and the PID controller would be reduced to a Proportional (P) or Proportional/Integral (P/I) controller. As the Ziegler-Nichols method is known to be an aggressive tuning strategy, a purely manual tuning strategy can be alternatively repeated for the integral and derivative gains as well [126].