A General Mobile Manipulator Automation Framework for Flexible Tasks in Controlled Environments

Can Pu, Chuanyu Yang, Jinnian Pu, Robert B. Fisher

Abstract

To enable a mobile manipulator to perform human tasks from a single teaching demonstration is vital to flexible automation. We call our proposed method MMPA (Mobile Manipulator Process Automation with One-shot Teaching). Currently, there is no effective and robust MMPA framework which is not influenced by the mobile base's parking precision. The proposed MMPA framework consists of two stages: collecting data (mobile base's location, environment information, end-effector's path) in the teaching stage for robot learning; letting the end-effector repeat the nearly same path as the reference path in the world frame to reproduce the work in the automation stage. More specifically, in the automation stage, the robot navigates to the specified location without the need of a precise parking. Then, based on colored point cloud registration, the proposed IPE (Iterative Pose Estimation by Eye & Hand) algorithm could estimate the accurate 6D relative parking pose of the robot arm base without the need of any marker. Finally, the robot could learn the error compensation from the parking pose's bias to modify the end-effector's path to make it repeat a nearly same path in the world coordinate system as recorded in the teaching stage. Hundreds of trials have been conducted with a real mobile manipulator to show the superior robustness of the system and the accuracy of the process automation regardless of the parking precision. For the released code, please contact AI@amigaga.com.

Keywords
Mobile manipulator, automation framework, one-shot teaching, iterative pose estimation, point cloud registration, adaptive online path learning

1 Introduction

Mobile manipulators (def: a mobile platform carrying a robot arm) have attracted much interest in universities and industry because of the combination of flexible locomotion and dexterous manipulation (Ramasubramanian et al., 2022; Yang et al., 2019a). A mobile manipulator is able to undertake multiple different tasks in a large workspace (rather than at a fixed workstation), and can be widely used in many flexible tasks, such as quality inspection in factories, workpiece loading and unloading in workshops, and painting. It is cumbersome and time-consuming to reprogram for each robot application. Thus, it is vital for flexible automation to minimize the human effort spent on teaching the robot, and to ensure the mobile manipulator is able to automate and redo multiple desired tasks in a large workspace flexibly with one-shot teaching from a human operator. This one-shot teaching and automation pipeline of a mobile manipulator for flexible tasks is referred to as Mobile Manipulator Process Automation with One-shot Teaching (MMPA)[1].

Current TMMA (short for traditional mobile manipulator automation) approaches (e.g. Nair et al. (2019)) consist of two stages: in the first stage the mobile base's location and the robot arm's path are recorded; in the second stage the mobile manipulator will park at the same location and replay the arm's path as recorded in the first stage. Such approaches rely heavily on precise parking of the robot base and are easily affected by ground conditions. As the robot arm's path is replayed in the exact same manner as recorded in the teaching stage, a minor error of the parking

[1] All the abbreviations in this manuscript are listed in Appendix C. List of Abbreviations.

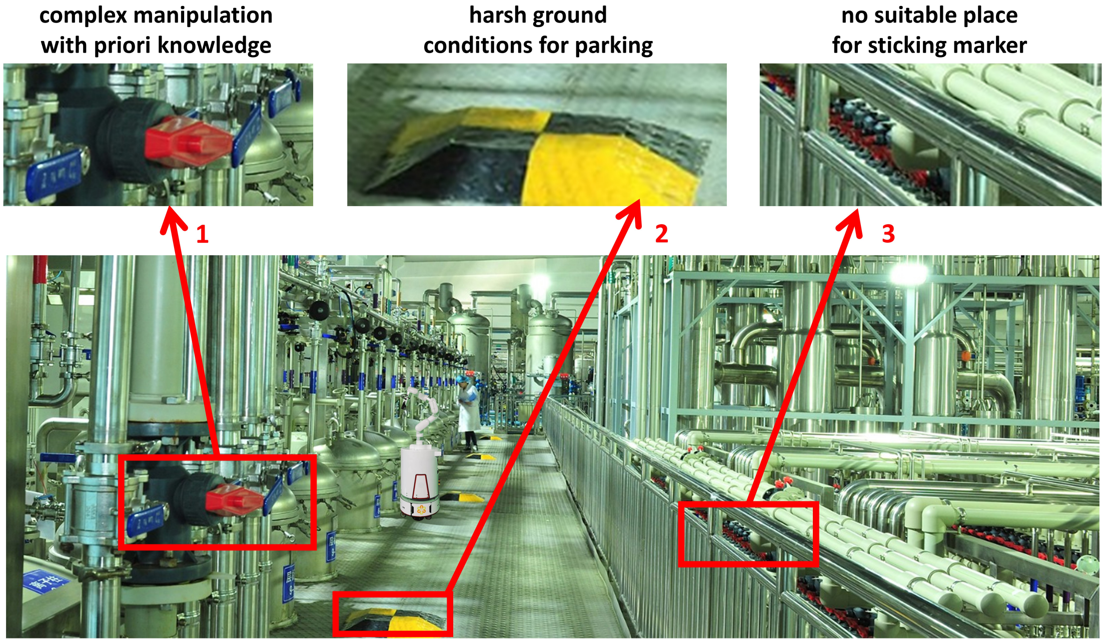


Figure 1: A typical workshop example with harsh conditions and our mobile manipulator. 1 The mobile manipulator is required to perform complex manipulation in 3D space with priori knowledge such as how to switch the valve rather than simple workpiece loading and unloading. 2 The ground consists of not only 2D plane but also 3D speed bump where a working robot parks on. 3 The guard bars and pipes are not suitable to stick any marker for error compensation.

position will lead to an incorrect robot arm manipulation. However, parking precisely is hard to achieve in the real world. For example, a rough, cratered, slippery ground will affect robot control accuracy[2] definitely (Li et al., 2018b; Nampoothiri et al., 2021; Nie et al., 2021; Wang et al., 2022). Additionally, the localization accuracy from SLAM (Simultaneous Localization And Mapping) (Bai et al., 2021, 2022; Grisetti et al., 2007; Hess et al., 2016; Khan et al., 2021; Nubert et al., 2022; Van Nam and Gon-Woo, 2021) is not high enough for precise parking in MMPA. Matching the 2D template marker's images from the teaching and automation stages (Meng et al., 2021) could provide a relative 3D pose for parking error compensation but can only be used to move the robot base on a 2D plane rather than a 3D surface. Lastly, there is no existing 6D pose recovery algorithm (Ao et al., 2021; Gu et al., 2022; Huang et al., 2022, 2021; Junior et al., 2022; Liu et al., 2021; Myronenko and Song, 2010; Park et al., 2017; Pu et al., 2018; Wu et al., 2021; Yang et al., 2015; Zeng et al., 2017; Zhou et al., 2016) that provides an accurate and robust 6D pose estimation (The 6D pose is [translation on x axis, translation on y axis, translation on z axis, roll, pitch, yaw]) for parking pose error compensation to meet the requirement of MMPA. As described above, Figure 1 shows a real typical workshop example with a harsh environment where our developed robot is working. The two articles (Wong et al., 2017, 2018) give a good overview of the design of the autonomous robotics system against the harsh environment in real-life settings.

The proposed MMPA method is capable of dealing with the problem of bad ground conditions with a novel iterative pose estimation approach to compensate for the error introduced by imprecise parking. The proposed MMPA framework has two stages, a one-shot teaching stage and an automation stage. In the one-shot teaching stage, a human teaches the mobile manipulator to perform a specific task. Information such as the location of the robot base, working environment information and the path of the robot arm's end-effector are collected, to allow for

[2] Note: on the market, a mobile base with high control accuracy (millimeter level) is 10+ times more expensive than a regular mobile base (centimeter level) we use.

the reproduction of the task in the automation stage. In the automation stage, the robot first navigates to the specified location without the need of precise parking. Then, based on the global colored point cloud registration, the proposed IPE (Iterative Pose Estimation by Eye & Hand) algorithm provides an accurate relative 6D parking pose without the need of any marker. Finally, the mobile manipulator uses the relative 6D parking pose to calibrate and adjust the path of the robot arm's end-effector for performing complex tasks taught by the human operator during the one-shot learning stage. In this work, hundreds of trials have been conducted with a real mobile manipulator to show that our proposed MMPA framework and IPE algorithm are capable of calibrating and adjusting the path of the robot arm end-effector to compensate for the parking error and guarantee the robustness of the system and the accuracy of the process automation without worrying about ground conditions and parking precision.

The proposed MMPA could bring much benefit to robot developers and customers who need the joint automation of flexible locomotion and dexterous manipulation. Currently, robot developers need to spend much time in programming robots for various tasks, which limits the robot application development efficiency and increases the development fee drastically. With the proposed MMPA, robot developers could develop simple robot automation tasks with human demonstration, which is also important to the customers without professional knowledge of robotics. The customers could adapt the mobile manipulator flexibly to a new task by themselves without the need of professional on-the-spot service, which reduces the maintenance fee and waiting time. With the proposed MMPA technique, one mobile manipulator could replace at least three workers[3] day and night, which decreases the labor cost. Additionally, compared to human workers, robot workers do not require human resource management and can work in hazardous environments, reducing the management burden for managers. Currently, the proposed MMPA technique could be used to develop many real applications in various areas. For example, currently welding large components still requires human work because fixed robot arms for welding have limited working range and the traditional mobile manipulator automation TMMA (Nair et al., 2019) could not meet the accuracy requirement, but the proposed MMPA technique could solve this defect in the welding factories. The mobile manipulator equipped with the proposed MMPA could also be used to load or unload the workpiece or electronic components and convey them to different designated places in some semi-conductor factories or machining workshops. In an e-commerce warehouse, the mobile manipulator with the proposed MMPA could be used to fetch different goods on the shelves, pack them and deliver them to the postmen. In a hospital, the mobile manipulator with the proposed MMPA could be used to dispose the harmful medical waste and convey it to an appointed place without any intervention from humans. Many more real applications are being developed on the way.

The remainder of this paper is structured as follows. Section 2 presents previous research about MMPA's progress, MMPA's impact factors, and 6D pose recovery algorithms. Section 3 presents the proposed algorithm for mobile manipulator process automation by one-shot teaching. Section 4 demonstrates the robustness of the system and assesses the accuracy of the process automation regardless of the ground conditions and parking precision based on hundreds of trials using the real mobile manipulator. Section 6 presents a summary of the work.

Contributions in this paper:

(1) An accurate 6D pose estimation algorithm: IPE (Iterative Pose Estimation by Eye & Hand);

(2) An adaptive online path learning method to calibrate the end-effector's trial in 3D space effectively;

(3) An effective and robust MMPA framework that is able to compensate for the errors caused by ground conditions and low parking precision;

[3] The robot could work 24 hours a day while a human worker only works for 8 hours a day

2 Related Works

Mobile manipulators combine the advantages of both the mobile base's mobility (Fragapane et al., 2021) and the articulated arm's dexterity, which makes it an ideal tool to perform multiple manipulation tasks that cover different locations in a large workspace. Research into mobile manipulators has a long history, dating back to the last century (Khatib et al., 1996; Mason et al., 1999; Yamamoto and Yun, 1992). The most recent surveys (Ramasubramanian et al., 2022; Yang et al., 2019a) give an explicit progress overview of the mobile manipulator's hardware system, software system and new applications. One of the interesting advanced techniques is mobile manipulator process automation because it requires the robot to autonomously reproduce the work taught by humans. Currently, the main ideas to realize mobile manipulator process automation in a large workspace are: (1) In the teaching stage, record the mobile base's location data and the arm's manipulation path data into the database. (2) In the automation stage, let the mobile base navigate and park at the location from the teaching stage with high accuracy (e.g.: position accuracy on the x, y axis ¡ 0.2 cm, orientation accuracy ¡ 0.1°) first; then let the robot arm replay the previously recorded manipulation path data from the database to reproduce the work that humans taught. Given the actuation accuracy of most economic collaborative robot arms is not above 0.1 mm, parking the mobile platform extremely accurately is the key to mobile manipulator process automation.

To make the robot base park precisely, the researchers have to deal with many aspects within the robot system, such as: sensor calibration (Ali and Mailah, 2019; Bai et al., 2021, 2022; Li et al., 2018a), localization (Bai et al., 2021, 2022; Grisetti et al., 2007; Hess et al., 2016; Khan et al., 2021; Nubert et al., 2022; Van Nam and Gon-Woo, 2021), control accuracy (Li et al., 2018b; Nampoothiri et al., 2021; Nie et al., 2021; Wang et al., 2022), error compensation (Meng et al., 2021; Yan et al., 2019). Single sensor calibration could rectify the output value by using a noise model to make it more accurate. For example, the researchers (Li et al., 2018a) improve the accuracy of the odometer (which can provide the velocity or mileage of a vehicle) through the use of Coriolis effects from three perspectives (scale factor, misalignment and level arm with inertial measurement unit). Multiple sensor calibration (Ali and Mailah, 2019; Bai et al., 2021, 2022) provides the relative pose among multiple different sensors for sensor fusion to improve the accuracy, such as using preintegration theory for IMU (Inertial Measurement Unit) and odometer self-calibration (Bai et al., 2021, 2022), calibrating gyroscope and magnetometer for data fusion (Ali and Mailah, 2019). However, sensor calibration could slightly reduce the error given the sensor's inherent properties or mounting positions. Thus, it contributes little to precise parking. In order to park accurately, mobile base localization is important. Currently, there are many popular SLAM algorithms, especially, the Lidar-based SLAM (Grisetti et al., 2007; Hess et al., 2016). The main feature of the Lidar-based SLAM (Khan et al., 2021; Van Nam and Gon-Woo, 2021) is that it uses Lidar scanner to input the position data for mapping and localization. The Lidar-based SLAM consists of two main categories: 2D Lidar SLAM (e.g.: Gmapping (Grisetti et al., 2007), Cartographer (Hess et al., 2016)) to generate a 2D map for localization (usually) indoors and 3D Lidar SLAM (e.g.: Lego-loam (Shan and Englot, 2018)) to generate a 3D map for localization (usually) outdoors. In order to get a more robust and accurate localization, it is a trend to fuse data from multiple sensors (e.g.: lidar, odometer, imu, etc.) using Kalman Filter (Urrea and Agramonte, 2021; Welch et al., 1995), Graph-based methods (Bai et al., 2021, 2022; Nubert et al., 2022) etc. . Nevertheless, their final localization accuracy is still limited to centimetre-level, which is still too large to meet the requirements of mobile manipulator process automation. Besides the above sensor calibration and localization, the mobile platform's control accuracy in the workspace is vital as well. Much research (Li et al., 2018b; Nampoothiri et al., 2021; Nie et al., 2021; Wang et al., 2022) about it has been done in recent years. For example, some researchers studied the terrain property (Li et al., 2018b; Nampoothiri et al., 2021) for the robot's control to handle the robot's slide etc. Some researchers proposed different improved control methods, such as fuzzy control (Nie et al., 2021), impedance control (Wang et al., 2022), etc. However, it is challenging to model the errors of the driving mobile base on different unknown terrain in a uniform way to ensure a robust and accurate control accuracy. Thus, parking precisely is hard in real-world scenarios where the environmental situation is unknown, which limits the mobile manipulator process automation's reliability.

Even if the mobile manipulator drives on a well known plane with high control accuracy, the localization accuracy still might not meet the requirements of the mobile manipulator process automation. In order to detect the localization's system error, iterative-learning error compensation methods (Meng et al., 2021; Yan et al., 2019) have been proposed recently. In this research (Meng et al., 2021), the researchers developed an extra eye-in-hand vision system to assist the mobile platform's localization. More specifically, they put an RGB camera at the end of the robot arm for perception and attached a QR (Quick Response) code on the workbench. In the teaching stage, the eye-in-hand vision system (the RGB camera on the arm) takes a photo of QR code using a fixed arm pose. In the automation stage, the mobile manipulator reaches the workbench and takes a photo of the previous QR code again using the previous same fixed arm pose. By matching those 2D QR code templates, the mobile base's relative 3D pose (position bias on the x, y axis and orientation angle) could be estimated. Then, the new relative 3D pose was used to rectify the mobile base's parking pose to make it park more precisely. Matching 2D QR code templates and rectifying the parking pose in an iterative way could make the mobile manipulator park precisely enough for a process automation. However, this method (Meng et al., 2021) suffers from two big problems. Given the pose from 2D QR code template matching has only 3 degrees of freedom, the mobile manipulators could only be moved on a good 2D plane terrain rather than a cratered 3D surface. The second problem is that the condition of the 2D plane for robot base moving should be good enough for accurate robot base control in order to park precisely.

To solve the two problems mentioned above (Meng et al., 2021), we use a depth sensor (stereo vision camera) on the arm to estimate the relative 6D pose of the mobile base by point cloud matching (Huang et al., 2021), which enables the estimation of the robot's pose, even on a rough 3D surface. By changing the robot arm end-effector's working path based on the relative 6D pose of the mobile base, our method does not require the robot base to park precisely, which eliminates the need of high control accuracy of the robot base and good quality ground condition.

Currently, there are three categories of point cloud registration algorithms for estimating the relative 6D pose. The recent survey (Huang et al., 2021) introduces the taxonomy of the point cloud registration methods and their progress. A new cross-source point cloud benchmark (Huang et al., 2021) is developed to evaluate the point cloud registration algorithms to solve cross-source challenges. The first category of methods(e.g.: (Gu et al., 2022; Junior et al., 2022; Park et al., 2017; Yang et al., 2015)) is based on ICP (Iterative Closest Point) (Besl and McKay, 1992), which estimates the rigid relative 3D pose between two point clouds in different views by minimizing the Euclidean distance between the corresponding points. Exact point-to-point correspondences seldom exist, which leads to the low accuracy of the ICP-based methods. The second category is feature-based methods (Ao et al., 2021; Wu et al., 2021; Zeng et al., 2017; Zhou et al., 2016), which extract the local descriptors from two point clouds first and then match them to recover the relative pose of the two point clouds. These methods are sensitive to strong noise and low density of the point clouds. That is, the noise and density of the point cloud influence the local descriptors' extraction and can even cause the algorithm to crash if the noise is too strong or the density is too low. The third class uses probabilistic models (Huang et al., 2022; Liu et al., 2021; Myronenko and Song, 2010; Pu et al., 2018) for point cloud registration. The probabilistic models are used to represent the structure of the point cloud, encoding the geometry distribution of the point cloud in 3D space. By calculating the maximum likelihood of the two probabilistic models, the relative pose can be estimated. This category of methods is more robust and accurate than the first and second category but its computation efficiency is low and limits many real-time applications. Although those three categories of point cloud registration algorithm can estimate the relative pose from different views, accuracy and robustness decreases with the increase of the initial rotation and translation error between the two point clouds, which might not be able to meet the requirement for the high accuracy in the mobile manipulator process automation. In order to solve this problem, we propose a method called IPE:Iterative pose estimation by eye & hand in Section 3.2.2.

Path planning, also known as motion planning, is an optimization problem that aims to find a valid collision-free trajectory to move the robot end-effector from a start point to an end point while satisfying certain constraints. Path planning problems are commonly solved using probabilistic sampling based algorithms such as rapidly exploring random tree (RRT*) (Karaman and Frazzoli, 2011), probabilistic roadmap (PRM) (Kavraki et al., 1996). Advance in path planning algorithms has allowed robotic arms to perform more complex and flexible tasks (Schulman et al., 2014; Sucan et al., 2012). Traditionally, the operation path of industrial robot arms within a factory is manually taught or programmed by a human with a dedicated teach pendant, which is tedious and time-consuming. In recent years, there has been an increase in interest on applying path planning algorithms to eliminate the time consuming manual teaching process. Path planning is typically used for manufacturing applications that involve processing industrial products with complex 3D geometry. Such applications include welding(Zhou et al., 2022), painting(Chen et al., 2020), grinding(Lv et al., 2020), polishing(Mohsin et al., 2017), cutting(van Sosin et al., 2019), and inspection(Alexis et al., 2016). Coverage path planning is a type of path planning problem, which aims to determine a collision-free trajectory that uniformly passes through all points of an area or volume of interest while minimizing travel time, energy, and other costs(Galceran and Carreras, 2013). Manufacturing applications that require coverage over a specific area of interest include painting(Chen et al., 2020), grinding(Lv et al., 2020), and inspection(Engemann et al., 2021).

3 Methodology

The main goal of our proposed MMPA is to ensure that the robot arm's end-effector repeats the same path recorded during the teaching stage in the world coordinate system to reproduce the desired task even if the parking precision is low. **A simple practical demo video to show the pipeline (teaching stage and automation stage) is available: `https://youtu.be/N5EQNMO_vJ8`.** The whole pipeline consists of two stages: (1) data collection by one-shot teaching (Section 3.1) to make the robot learn to perform a specific task; (2) replaying the newly-learned data by the mobile manipulator process automation (Section 3.2) to enable the robot to finish the task successfully and flexibly on its own.

Figure 2 shows different coordinate systems[4]: mobile base coordinate system $XYZO_{MB}$, robot arm base coordinate system $XYZO_B$, depth camera coordinate system $XYZO_C$, world coordinate system $XYZO_W$, the joint coordinate system from $XYZO_1$ to $XYZO_6$.

3.1 One-shot Teaching

To realize mobile manipulator process automation in the automation stage, the mobile platform's location information in the pre-built map, the working environment information (colored 3D point cloud of the workbench, tools, etc.) and the robot arm's end-effector path information are collected in real-time through one-shot teaching in this stage in this sub section.

In a known workspace (that is, the navigation map has been built beforehand by some SLAM algorithm, such as Gmapping (Grisetti et al., 2007), Cartographer (Hess et al., 2016) and so on), the mobile manipulator is controlled to finish a task denoted by $tk_i (i = 1, 2, 3, ...)$. Each task tk_i contains the mobile platform location information $L_i(vehicle)$, the initial working environment information - a colored 3D point cloud $X_i(O)$ and the robot end-effector's path $Path_i(O)$ when performing the task tk_i. The location information $L_i(vehicle)$ is recorded only once, which consists of the position and Euler angle of the mobile platform in the world coordinate system $XYZO_W$. The colored 3D point cloud $X_i(O)$ is acquired by the depth sensor mounted on the robot arm in its depth sensor coordinate system $XYZO_C$. The recorded path information[5] $Path_i(O)$ consists of multiple sample points containing information of the pose of the end-effector. The path is recorded with a time frequency of $f_i(O)$ Hz. The position and Euler angles of the end-effector are recorded in the robot arm's base coordinate frame $XYZO_B$.

[4] All the symbols in this manuscript are listed in Appendix D. List of Symbols.

[5] We perform one-shot teaching of the desired path by making a human operator manually guide and apply force on the end-effector on the robot arm. Collaborative robot arms have a zero gravity compensation mode, where the robot exerts just enough torque to compensate for the force applied by gravity. Under this mode, a human operator is able to move and guide a robot by applying force directly on the robot body.

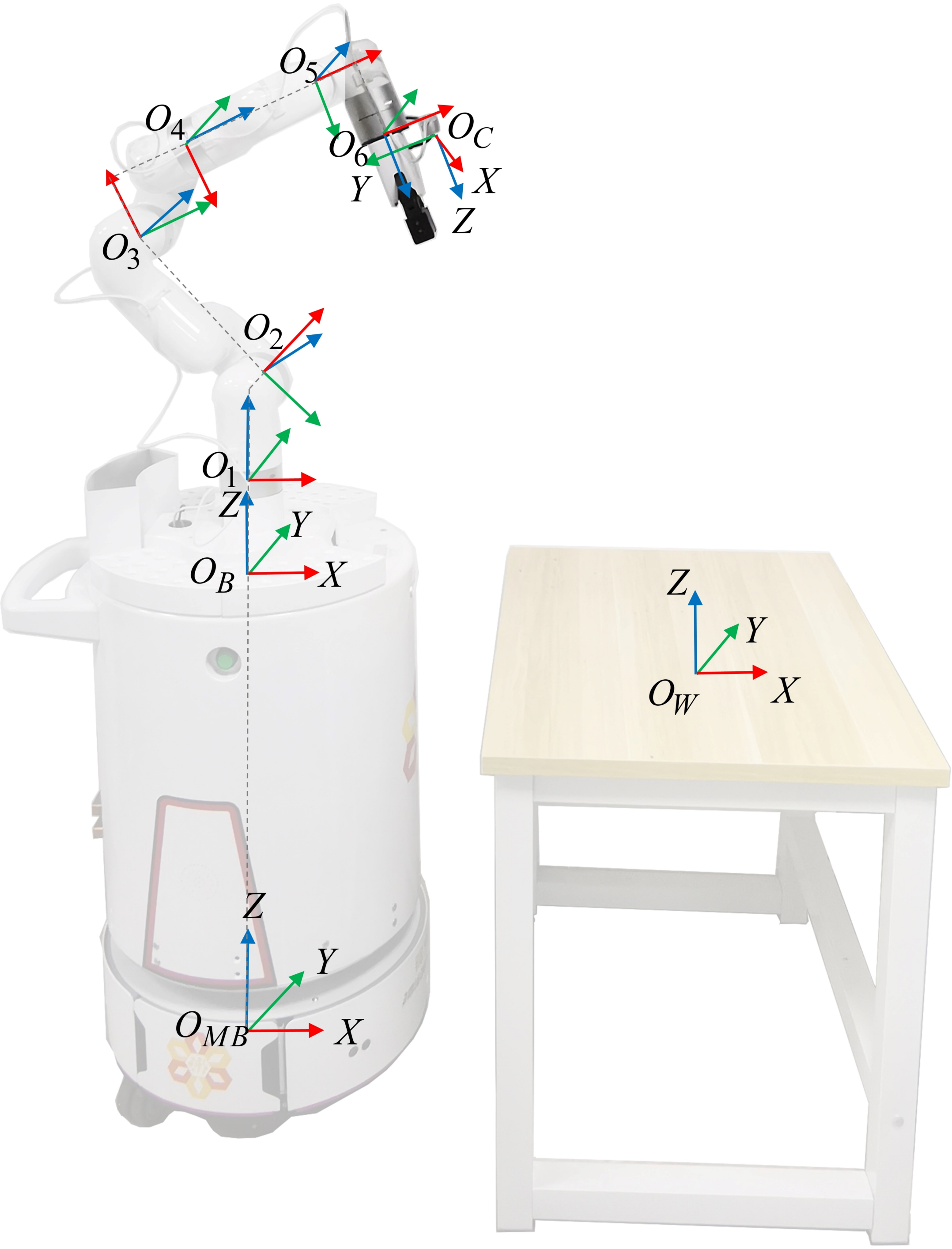


Figure 2: The figure shows the various coordinate frames in a mobile manipulator.

After collecting the above data for task tk_i only once, the one-shot teaching comes to an end.

3.2 Mobile manipulator process automation

In the current stage, to finish the task tk_i, the mobile manipulator process automation needs to accomplish the following four steps: (1) Firstly, the mobile platform navigates to the location $L_i(vehicle)$ and finally parks at the location $\tilde{L}_i(vehicle)$ autonomously (Section 3.2.1). (2) Then, the IPE algorithm calculates the relative 6D parking pose between the location $L_i(vehicle)$ and $\tilde{L}_i(vehicle)$ in the robot arm's base coordinate frame (Section 3.2.2). (3) Next, the adaptive online path learning part learns the difference in pose and modifies the initial robot end-effector path information $Path_i(O)$ into the new one $\widetilde{Path}_i(arm)$ in the robot arm's base coordinate frame[6] (Section 3.2.3). (4) Finally, the mobile manipulator carries out the task tk_i using the newly-learned path $\widetilde{Path}_i(arm)$ (Section 3.2.4).

3.2.1 Robot navigation

During the automation stage, the mobile manipulator will attempt to navigate to the desired parking location autonomously $L_i(vehicle)$. However, due to errors within the system, the robot will eventually park at a different location $\tilde{L}_i(vehicle)$ that is close to $L_i(vehicle)$. Due to the uneven ground, slippery floor, dynamic obstacles or some other random factors in the real working environment, the mobile platform usually cannot park precisely at the location $L_i(vehicle)$. Previous work using the 2D QR code template matching (Meng et al., 2021) could only provide the mobile platform's relative 3D parking pose between the location $L_i(vehicle)$ and $\tilde{L}_i(vehicle)$. Given the acquired pose from the 2D QR code template matching has only three degrees of freedom, the pose could only be applied to the mobile platform to move on the 2D plane, which couldn't be used to calibrate the robot arm's path in 3D space. To get the extremely-accurate robot arm base's relative 6D pose between the location $L_i(vehicle)$ and $\tilde{L}_i(vehicle)$, the iterative 6D pose estimation by eye & hand interaction will be proposed in the following part.

3.2.2 IPE:Iterative pose estimation by eye & hand

Figure 3 shows the iterative process when performing the IPE algorithm. The robot arm's base coordinate system is $XYZO_{B_i}$ when reaching the location $\tilde{L}_i(vehicle)$. The robot arm will move the mounted depth camera iteratively to search for the initial sampling pose from the teaching stage. After k^{th} movement of the robot arm, the camera coordinate system becomes $XYZO_{C_k}$. Figure 4 illustrates the flowchart of the proposed IPE algorithm. When the mobile manipulator reaches the working place in the automation mode, the IPE algorithm starts. The robot arm's new sampling pose is set to the initial one in the robot arm base coordinate frame when sampling the working environment's colored 3D point cloud in the one-shot teaching stage. Taking the system safety into account, if the new sampling pose of the end-effector is unreachable by the robot arm, the IPE algorithm will exit with the "False" execution flag. If the new sampling pose of the end-effector is available, the robot arm will move the end-effector to the new sampling pose to sample the working environment's colored point cloud $X_i(k), k = 1, 2, 3, ...$ (k represents the k^{th} movement of the robot arm in IPE). Then we calculate the relative 6D pose $[\Delta R_{C_k},\ \Delta t_{C_k}]$ between the sampled point cloud $X_i(O)$ from teaching stage and the sampled point cloud $X_i(k)$ in the current view in the automation stage through the colored point cloud global registration algorithm (See **II Method: colored point cloud global registration** for more details). Then we will update the relative 6D parking pose $[\Delta R_B(k),\ \Delta t_B(k)]$ of the robot arm's base (For more details, see **I Justification: dynamic pose update in different coordinate systems**). According to the updated relative parking pose $[\Delta R_B(k),\ \Delta t_B(k)]$, the next new sampling pose is set. If the pose difference $[\Delta R_{C_k},\ \Delta t_{C_k}]$ is below the threshold $\boldsymbol{\alpha}$, IPE will exit and return the relative 6D pose $[\Delta R_B(k),\ \Delta t_B(k)]$ in the robot arm's base coordinate system with a "True"

[6] $Path_i(O)$ and $\widetilde{Path}_i(arm)$ are nearly the same in the world coordinate system.

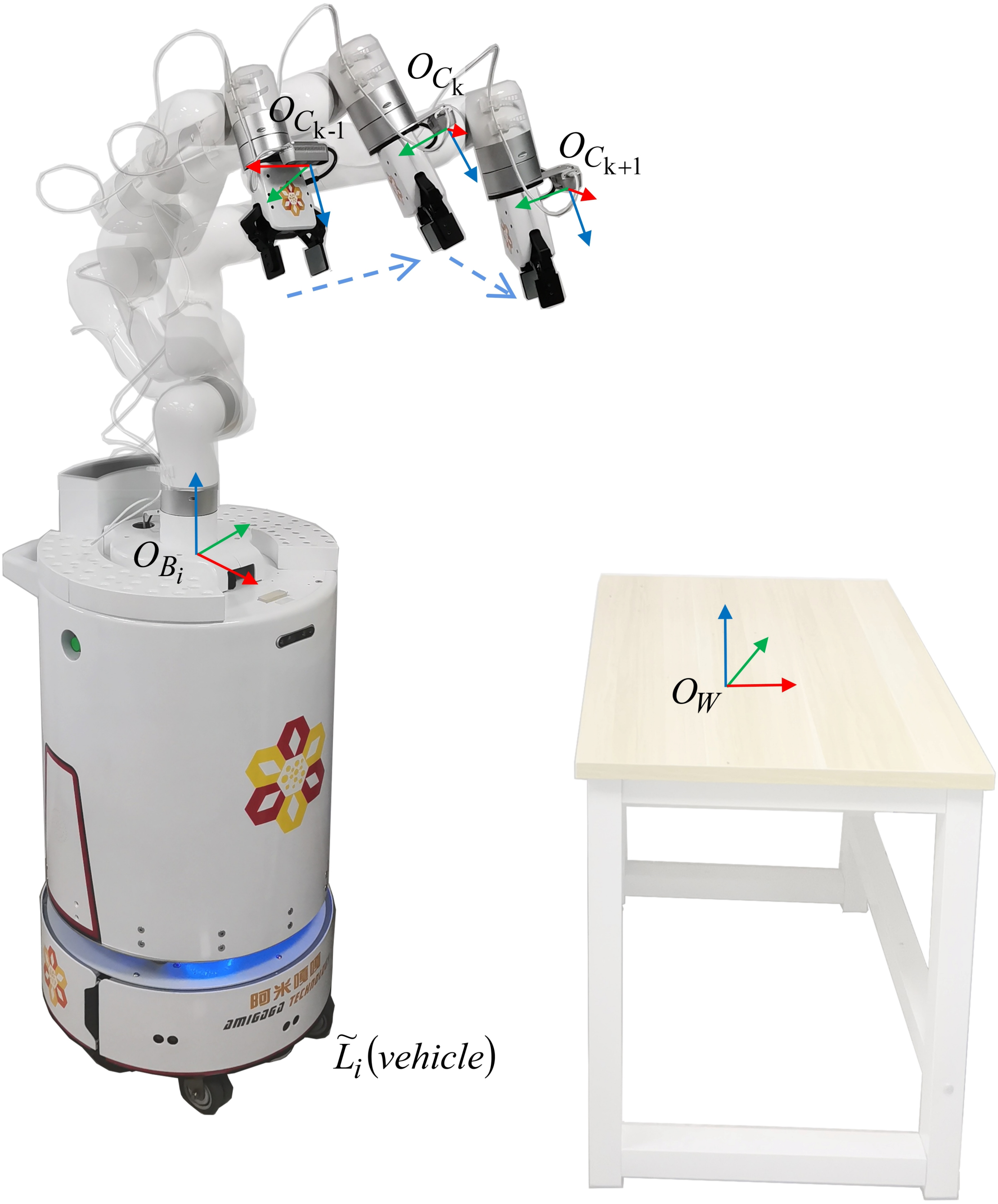


Figure 3: The iterative process by eye-hand interaction to search for the same sampling pose in the teaching stage in world frame.

execution flag. If the pose difference $[\Delta R_{C_k}, \Delta t_{C_k}]$ is above the threshold $\boldsymbol{\alpha}$ and the loop count is below the threshold β, IPE will run into the next loop and check if the new sampling pose is reachable or not. If the loop count is above the threshold β, the IPE algorithm will exit with the "False" execution flag.

I Justification: dynamic pose update in different coordinate systems

IPE performs iterative pose estimation multiple times, which adjusts the sampling pose of the mounted depth camera incrementally by moving the robot arm to match the initial sampling pose of the camera in the one-shot teaching stage in the world frame. IPE relies on a unique property, that is, if the new sampling pose of the depth camera in the automation stage is closer to the initial sampling pose of the depth camera in one-shot teaching stage in the world frame, the newly sampled point cloud in automation stage will be more similar to the initial sampled point cloud in the one-shot teaching stage. It means there will be more common features for the colored point cloud registration to match thus achieving a higher matching accuracy. Thus, in IPE algorithm, the robot arm tends to move the depth camera in an eye-hand iterative style[7] for multiple times to position the depth camera in its initial sampling pose in the one-shot teaching stage. Experiments (see Figure 9) show that with each iteration, the IPE rotation and translation errors gradually decrease. The result from the final iteration is more accurate than the result obtained from the first iteration.

Due to the eye-hand dynamic iteration process, the robot arm has to change its configuration to reposition the depth camera constantly, therefore the spatial relationship between the depth camera and robot arm's base is constantly changing. In this part, given the dynamic camera pose and robot arm's dynamic movement, we will deduce the robot arm's base relative 6D pose between the teaching stage and the current automation stage.

Figure 5 shows the relative 6D pose between different coordinate systems. $R_{B_o}^{C_o}$ and $t_{B_o}^{C_o}$ represent the rotation and translation matrix from the camera coordinate system to the robot arm's base coordinate system when sampling the working environment's colored point cloud in the one-shot teaching stage in task tk_i. $R_{B_k}^{C_k}$ and $t_{B_k}^{C_k}$ represent the rotation and translation matrix from the camera coordinate system to robot arm's base coordinate system after the $k^{th}, k = 1, 2, 3, ...$ movement of the robot arm in IPE. S_{C_o} represents one 3D point in the camera coordinate system, which corresponds to the 3D point S_{B_o} in the robot arm's base coordinate system, in one-shot teaching stage. S_{C_k} represents one 3D point in camera coordinate system, which corresponds to the 3D point S_{B_k} in robot arm's base coordinate system, after the k^{th} movement of the robot arm in the automation stage. ΔR_{C_k} and Δt_{C_k} represent the camera's relative rotation and translation matrix between the teaching stage and the current automation stage after the k^{th} movement of the robot arm. ΔR_{B_k} and Δt_{B_k} represent the robot arm base's estimated relative rotation and translation matrix when sampling the working environment's point cloud in one-shot teaching stage and after the k^{th} movement of the robot arm during the iterative adjustment process.

$$R_{B_o}^{C_o} S_{C_o} + t_{B_o}^{C_o} = S_{B_o} \tag{1}$$

$$R_{B_k}^{C_k} S_{C_k} + t_{B_k}^{C_k} = S_{B_k} \tag{2}$$

$$\Delta R_{C_k} S_{C_o} + \Delta t_{C_k} = S_{Ck} \tag{3}$$

$$\Delta R_{B_k} S_{B_o} + \Delta t_{B_k} = S_{B_k} \tag{4}$$

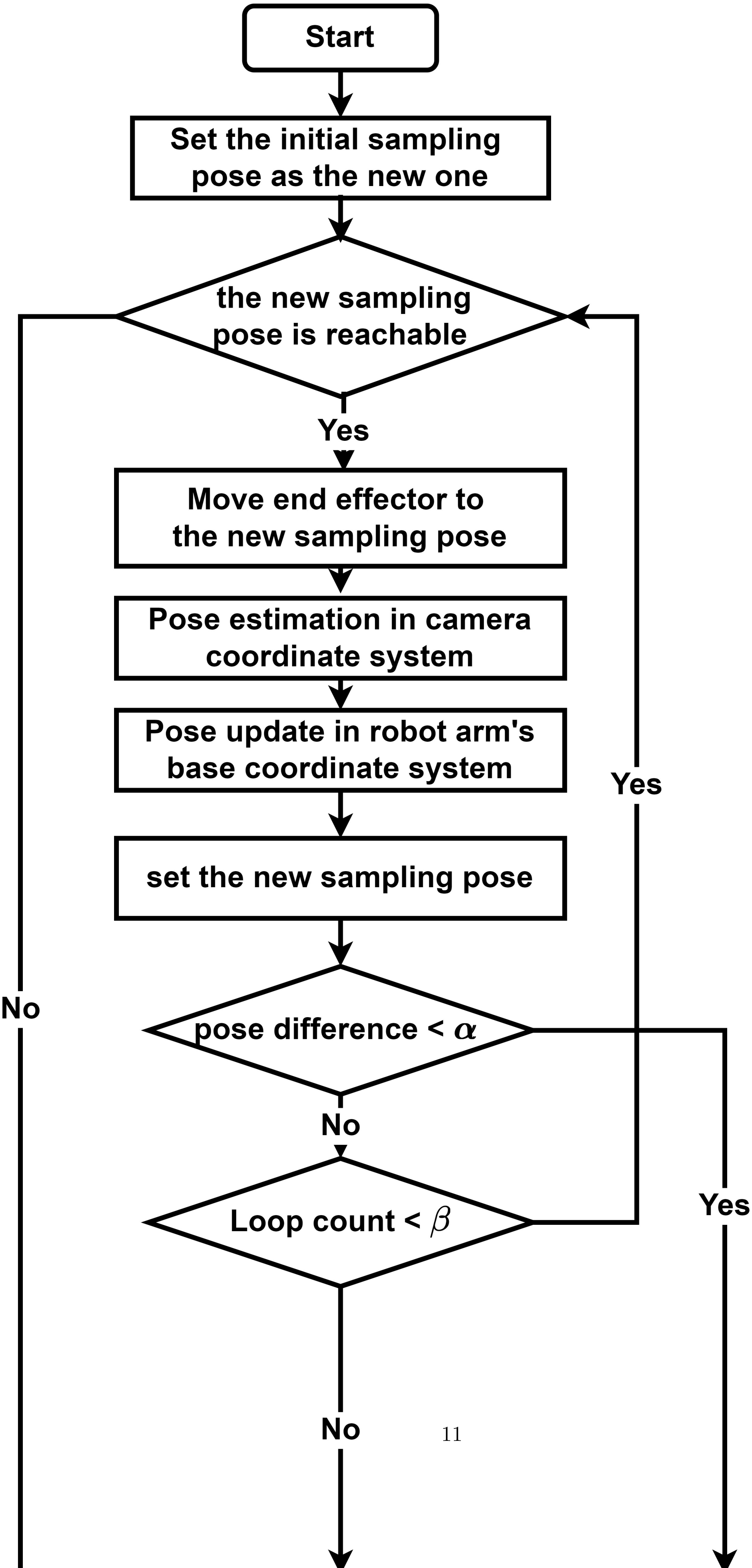
From equation (1) and (2) we get

$$S_{C_o} = R_{B_o}^{C_o -1} S_{B_o} - R_{B_o}^{C_o -1} t_{B_o}^{C_o} \tag{5}$$

$$S_{C_k} = R_{B_k}^{C_k -1} S_{B_k} - R_{B_k}^{C_k -1} t_{B_k}^{C_k} \tag{6}$$

Substitute S_{C_o} and S_{C_k} in equation (3) with equation (5) and (6). We get

[7] Eye-hand iterative style means doing registration and moving the mounted camera by the robot arm iteratively.


Start
Set the initial sampling pose as the new one
the new sampling pose is reachable
Yes
Move end effector to the new sampling pose
Pose estimation in camera coordinate system
Pose update in robot arm's base coordinate system
set the new sampling pose
pose difference < α
No
Loop count < β
Yes
No
Yes
No

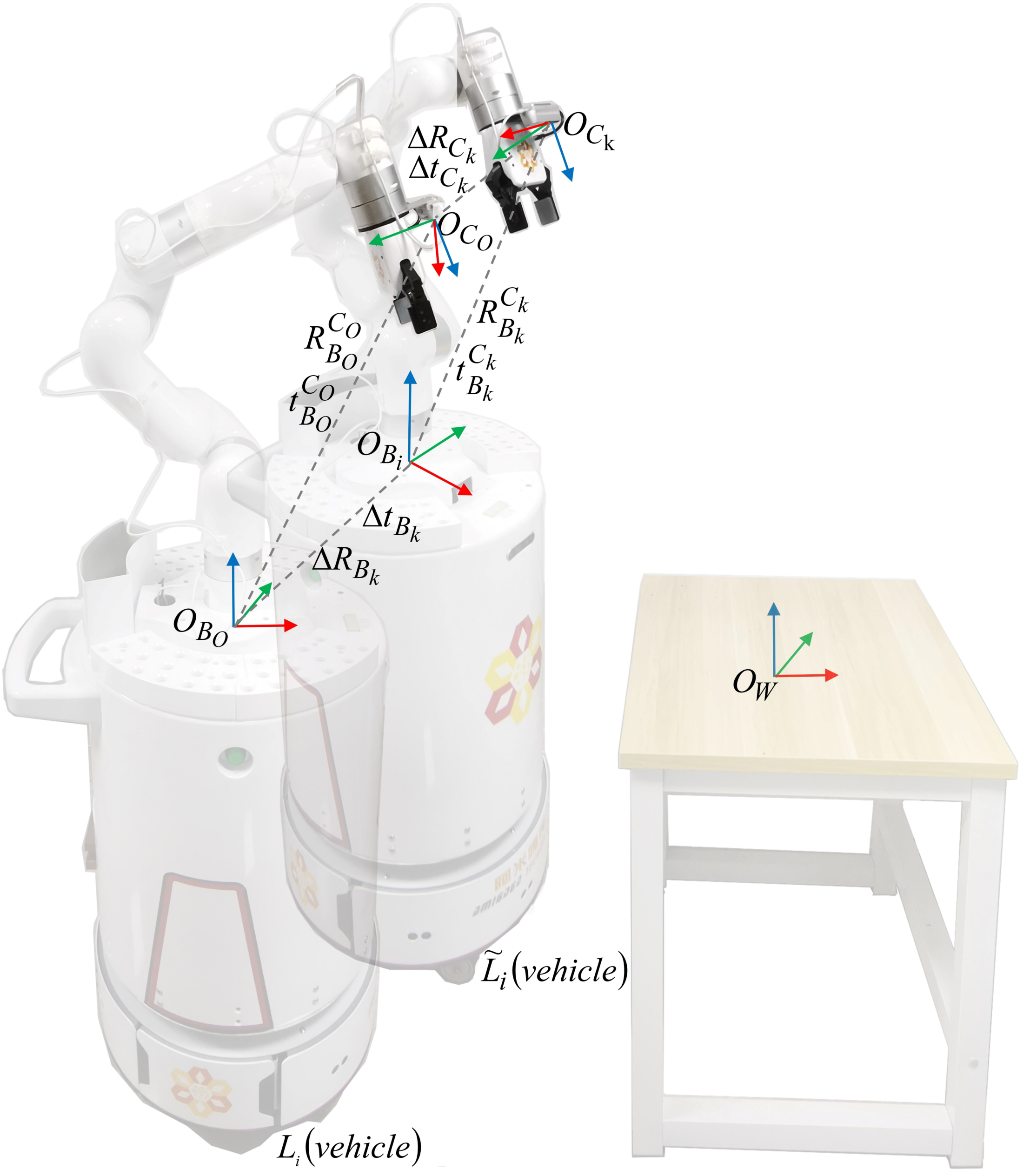


Figure 5: The figure illustrates how the relative parking pose of the robot arm's base can be calculated given the spacial relationship of the camera and base and the relationship between different camera poses.

$$\begin{aligned}&\Delta R_{C_k}\left(R_{B_o}^{C_o-1}S_{B_o}-R_{B_o}^{C_o-1}t_{B_o}^{C_o}\right)+\Delta t_{C_k}=\\&R_{B_k}^{C_k-1}S_{B_k}-R_{B_k}^{C_k-1}t_{B_k}^{C_k}\end{aligned}\tag{7}$$

Reformulating the equation (7) by multiplying $R_{B_k}^{C_k}$ on the both sides, we get:

$$\begin{aligned}&R_{B_k}^{C_k}\Delta R_{C_k}R_{B_o}^{C_o-1}S_{B_o}+(-R_{B_k}^{C_k}\Delta R_{C_k}R_{B_o}^{C_o-1}t_{B_o}^{C_o}\\&+R_{B_k}^{C_k}\Delta t_{C_k}+t_{B_k}^{C_k})=S_{B_k}\end{aligned}\tag{8}$$

Comparing equation (8) and equation (4), we get:

$$\Delta R_{B_k}=R_{B_k}^{C_k}\Delta R_{C_k}R_{B_o}^{C_o-1}\tag{9}$$

$$\Delta t_{B_k}=-R_{B_k}^{C_k}\Delta R_{C_k}R_{B_o}^{C_o-1}t_{B_o}^{C_o}+R_{B_k}^{C_k}\Delta t_{C_k}+t_{B_k}^{C_k}\tag{10}$$

II Method: colored point cloud global registration

In order to get the mounted camera's relative rotation matrix ΔR_{C_k} and translation matrix Δt_{C_k}, the colored point cloud global registration method[8] is proposed in this part by simply combining the fast global registration (Zhou et al., 2016) and the local colored point cloud registration (Park et al., 2017) with some improved engineering techniques (e.g.: point cloud preprocessing, parameter fine-tuning, multi-scale matching, etc.). More specifically, the coarse global pose from the fast global registration (Zhou et al., 2016) is fed into the colored point cloud registration (Park et al., 2017) algorithm to avoid the local minimum during the registration. The local algorithm (Park et al., 2017) takes not only the geometry information but also the color information into account to achieve a better registration accuracy. The proposed method with the simple strategy above provides a robust and accurate 6D pose estimation for IPE framework, which meets the mobile manipulator process automation's localization requirement.

3.2.3 Adaptive online path learning

The robot end-effector's path for performing task tk_i is recorded in the robot arm's base frame during one-shot teaching. When the robot reaches the same working place again in the automation stage, the robot arm base's pose will differ from that of the one-shot teaching stage because of motion or system errors or some other random factors. To ensure the robot end-effector follows the desired path in the world frame to finish the manipulation task tk_i, the robot has to learn to adjust the reference path $Path_i(O)$ obtained in teaching stage to the new path $\widetilde{Path_i}(arm)$ in the robot arm's base coordinate frame in the automation stage.

Considering a sampled point S_{B_o} from the recorded path $Path_i(O)$ in the teaching stage, the position of the end-effector is transferred from the teaching stage to the automation stage in the robot arm's base coordinate system using the following equation

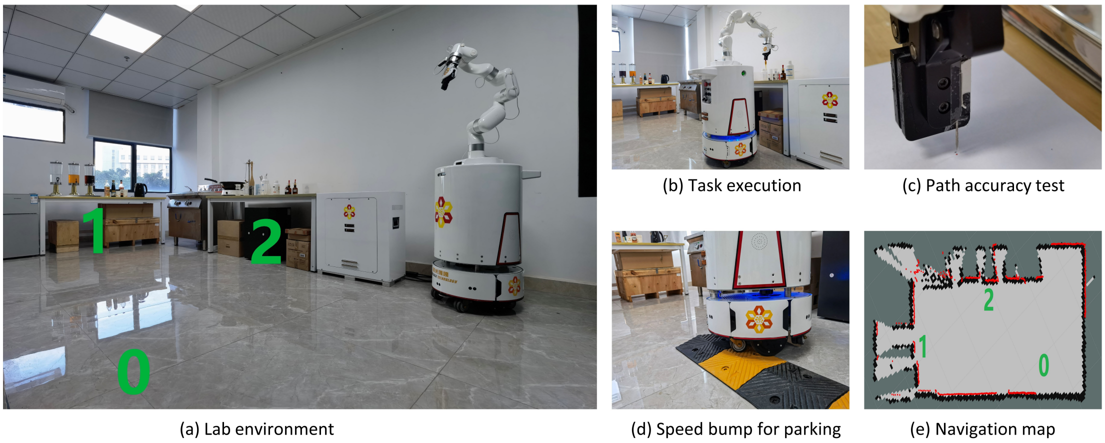
$$\Delta R_{B_k}S_{B_o}+\Delta t_{B_k}=S_{B_k}\tag{11}$$

where ΔR_{B_k} and Δt_{B_k} could be calculated using equation (9) and equation (10). The sampled point S_{B_k} is from the new execution path $\widetilde{Path_i}(arm)$ in the automation stage.

3.2.4 Task Execution

After obtaining the newly-adjusted robot end-effector path $\widetilde{Path_i}(arm)$, some motion planning algorithm (e.g.: RRT* (Karaman and Frazzoli, 2011), PRM (Kavraki et al., 1996)) could be used to compute the trajectory containing the target joint angles which are used to control the robot joints during execution. The acceleration, torque, velocity, and position limits of the robot joints are taken into account by the motion planning algorithm. If the motion planning algorithm could compute a valid trajectory, the robot will execute the task. If not, the robot will abandon this task.

[8] Readers could also use other 3D point cloud registration algorithms to replace the colored point cloud global registration in this part as well.



(a) Lab environment

(b) Task execution

(c) Path accuracy test

(d) Speed bump for parking

(e) Navigation map

Figure 6: Experiment configuration. (a) shows the experiment environment. The mobile manipulator will conduct experiments at site 0, site 1, site 2, site 2' (site 2 and site2' are in the same location). (b) shows the robot executing a task. (c) shows the executed path's accuracy test using a needle. (d) shows a speed bump installed at the parking place to emulate uneven terrain. (e) shows the pre-built map for robot navigation. The black lines or dots in the occupancy map represent obstacles ("occupied"). The red lines or dots represent the laser scan which is used for localization. '0', '1' and '2' represent site 0, site 1 and site 2.

4 Experiment

4.1 Experiment Configuration

The mobile manipulator "Gagabot DR-03"[9] is the platform to demonstrate the performance of the proposed method. Gagabot DR-03 consists of a two-wheeled differential mobile platform and a 6-DOF (Degree of Freedom) robot arm. A depth camera "Intel Realsense D435" is mounted at the end of the robot arm to perceive the environment. Inside Gagabot DR-03, an integrated industrial computer (Nvidia AGX Xavier) is used for the complex computation (e.g.: navigation, perception, planning, control, etc.). The Nvidia AGX Xavier (105 mm x 105 mm x 65 mm) is equipped with a 512-core Volta GPU, a 8-core ARM v8.2 64-bit CPU and 32GB memory. All the experiments in this paper were conducted with the Nvidia AGX Xavier (See Figure 6). Figure 6, (a) shows an image of the lab where we conduct the experiments. The robot will navigate from Site 0 to Site 1, Site 2, Site 2' to perform the experiments (Site 2 and Site2' are in the same location). (b) shows an example of the robot executing a task. (c) shows the executed path's accuracy test by using a needle. (d) shows a speed bump installed on the parking position to emulate uneven terrain. (e) shows the pre-built map from Gmapping (Grisetti et al., 2007) for the robot navigation.

The experiments are divided into 3 groups given different working sites (Site 1, Site 2, Site 2'). Three different kinds of experiments and analysis are conducted for each group: 1) parking pose accuracy in Section 4.2, 2) IPE performance in Section 4.3 and 3) executed path's accuracy in Section 4.4. Site 1[10] emulates a plant extract workshop to make the robot perform complex manipulation with a flat floor, such as switching the valve, fetching and returning the cup. Site 2[11] emulates a quality inspection center in a factory with a flat floor to test the liquid products

[9] For more details, please refer to `http://www.amigaga.com/en/index.php?id=111`

[10] Site 1 consists of cups, big liquid containers with valves, small bottles containing different liquid, desks, logos and letters printed on the tape.

[11] Site 2 consists of glass and plastic reagent bottles (containing sodium hydroxide, hydrochloric acid, phenolphthalein, ferrum, seperately) , droppers, a waste disposal container, test tubes, a test tube stand, a desk, logo and letters printed on the tape.

Table 1: Number of Different Experiment Trials

Factor	*Site 1*	*Site 2*	*Site 2'*
parking pose accuracy	30 times	30 times	30 times
IPE performance	30 times	30 times	30 times
executed path's accuracy	30 times	30 times	30 times

by chemical reaction. Site 1 and Site 2 consists of only flat floor. However, in the real world, there are many harsh working environments where the floor is uneven. Taking real-world floor conditions into consideration, a speed bump[12] is installed at site 2 to emulate uneven terrain. The new site is denoted as site 2'. Thus, we could get various 6D parking poses on a 3D surface for the experiments. We define this group of experiments with a speed bump to park on (see Figure 6 d) are at Site 2'. Each experiment is conducted 30 times. For an explicit overview of all the experiments, please see Table 1. We have used two formats to describe the 6D pose. The first way is using a 6D vector $[tx, ty, tz, r, p, y]$ (short for [translation on x axis, translation on y axis, translation on z axis, *roll*, *pitch*, *yaw*]). The unit is meter for tx, ty, tz and degree for r, p, y. We use the difference between the estimated 6D vector and the ground truth to describe the 6D pose's error. The second way is using rotation matrix $\boldsymbol{R}$ and translation matrix $\boldsymbol{t}$. The equations for estimating the accuracy of the 6D poses are from (Huynh, 2009):

$$E_R = ||\boldsymbol{I} - \boldsymbol{R_{gt}}\boldsymbol{R_{est}^{-1}}||_F \tag{12}$$

$$E_t = ||\boldsymbol{t_{gt}} - \boldsymbol{t_{est}}||_F \tag{13}$$

where $\boldsymbol{t_{est}}, \boldsymbol{R_{est}}$ are the estimated values and $\boldsymbol{t_{gt}}, \boldsymbol{R_{gt}}$ are the ground truth respectively. $|| \bullet ||_F$ is the Frobenius norm. Readers should know that: Although the error estimation methods and their error values for the two formats above are different, they all could describe the 6D pose's accuracy.

4.2 Parking Pose Accuracy

Analysis is performed to observe if there is any relationship between the parking pose's accuracy and IPE performance. In each experiment, the robot will be required to navigate to a specified location. To get the ground truth of the relative parking pose, the end-effector is pointed to a fixed point with a fixed pose when the mobile base parks at the specified location. The transformation matrix from the robot arm's base to the fixed point is T_0. When the mobile base navigates to the specified location again, we let the end-effector point to that fixed same point with that same pose. The current transformation matrix from the robot arm's base to the fixed point is T_1. Given the robot arm is extremely accurate with a repeatability of 0.1mm, we use the product of those transformation matrixes $T_0T_1^{-1}$ to represent the ground truth of the relative parking pose.

We designate the position (2.0415, -1.375, 0.0) with orientation (0.0, 0.0, -0.2970, 0.9548) (represented by unit quaternion) as Site 1 in the map (See Figure 6 e). We specify the position (0.1251, -0.7638, 0.0) with orientation (0.0, 0.0, -0.8785, 0.4776) (represented by unit quaternion) as Site 2. The starting point's position (site 0) is (-0.3498, 0.9449, 0.0) with an orientation (0.0, 0.0, -0.5367, 0.8437). The mean and standard deviation of the relative parking pose are listed in

[12] The speed bump's shape is a triangular prism, with the size $1000mm * 350mm * 50mm$ ($length * width * height$). It weighs 11 kg and has a surface with uneven 3D texture for increased friction.

Table 2 by using the 6D vector $[tx, ty, tz, r, p, y]$ format. If we use Equation (12) and Equation (13) to describe the error, the mean rotation error is 0.0617 at Site 1, 0.0614 at Site 2, 0.2910 at Site 2'. The standard deviation of the rotation error is 0.0418 at Site 1, 0.0433 at Site 2, 0.2213 at Site 2'. The mean translation error is 0.1018 m at Site 1, 0.0799 m at Site 2, 0.1304 m at Site 2'. The standard deviation of the translation error is 0.0176 m at Site 1, 0.0316 m at Site 2, 0.0594 m at Site 2'. Note that this is the uncorrected initial base positioning error.

Table 2: Initial Parking Pose Accuracy

Factor	*Site 1*	*Site 2*	*Site 2'*
$\overline{tx}$ / (m)	0.0011	0.0505	0.0779
σ_{tx} / (m)	0.0281	0.0287	0.0569
$\overline{ty}$ / (m)	-0.0995	0.0539	0.0261
σ_{ty} / (m)	0.0179	0.0287	0.1060
$\overline{tz}$ / (m)	-0.0012	-0.0011	-0.0042
σ_{tz} / (m)	0.0013	0.0014	0.0054
$\overline{r}$ / (deg)	0.2151	0.1155	0.0079
σ_r / (deg)	0.1593	0.0784	1.2595
$\overline{p}$ / (deg)	-0.0036	-0.1733	0.7932
σ_p / (deg)	0.1439	0.1295	2.6967
$\overline{y}$ / (deg)	0.0023	1.6706	-1.9701
σ_y / (deg)	2.9214	2.7079	14.7211

4.3 IPE Performance

From Equation 9 and Equation 10, the relative parking pose (ΔR_{B_k}, Δt_{B_k}) and the relative camera pose (ΔR_{C_k}, Δt_{C_k}) could be deduced or calculated from each other. Thus, we will only test the accuracy of the relative parking pose (ΔR_{B_k}, Δt_{B_k}) in this part. The method for obtaining the ground truth of relative parking pose is the same in Section 4.2. A threshold has to be set to indicate convergence. We set the convergence threshold $\boldsymbol{\alpha}$ of the sampling pose difference $(x, y, z, roll, pitch, yaw)$ as (0.002m, 0.002m, 0.002m, 0.5°, 0.5°, 0.5°). The maximum iteration number β is set as 5 for Site 1, Site 2 and 10 for Site 2' because the situation at Site 2' is harder than those at Site 1, Site 2. This is because the uneven terrain emulated by the speed bump will introduce more changes for point cloud matching in the z axis, roll and pitch orientation, compared to the flat terrain case of Site 1 and Site 2.

Figure 7 shows one example of the IPE matching. In Figure 7, (a) - (c) show the point clouds (from the depth camera) after k^{th} movement of the robot arm in IPE. (d) is the point cloud sampled during the teaching stage. (e) shows the scene before matching the point cloud (a) and (d). (f) shows the scene before matching point cloud (b) and (d). (g) shows the scene before matching the point cloud (c) and (d). Note the improved alignment. (h) is the RGB image of the scene. (i) - (l) are the local patches from (e) - (h). With each iteration k, the scene [(a), (b) (c)] gets closer to the original (d). That is, the current depth camera pose gets closer iteratively to the sampling pose from the teaching stage. With each additional iteration of the IPE, the registration result [(e),(f),(g)] gradually improves. (i) - (k) shows the details of the registration's accuracy. Compare the word "AMIGAGA TECHNOLOGY" in (j) and (k), some parts are missing in (j).

In Table 3, the mean $[\overline{\Delta tx}, \overline{\Delta ty}, \overline{\Delta tz}, \overline{\Delta r}, \overline{\Delta p}, \overline{\Delta y}]$ and standard deviation $[\sigma_{\Delta tx}, \sigma_{\Delta ty}, \sigma_{\Delta tz}, \sigma_{\Delta r}, \sigma_{\Delta p}, \sigma_{\Delta y}]$ of the error between the IPE estimation (ΔR_{B_k}, Δt_{B_k}) and ground truth are given. While using the rotation matrix and translation matrix criteria shown in Equation (12) and Equation (13) as error measures, the mean rotation error between IPE estimation and ground truth is 0.0169 at Site 1, 0.0183 at Site 2, 0.0225 at Site 2'. The standard deviation of the rotation error is 0.0074 at Site 1, 0.0063 at Site 2, 0.0103 at Site 2'. The mean translation error is 0.0049 m at Site 1, 0.0054 m at Site 2, 0.0089 m at Site 2'. The standard deviation of the translation error is 0.0025

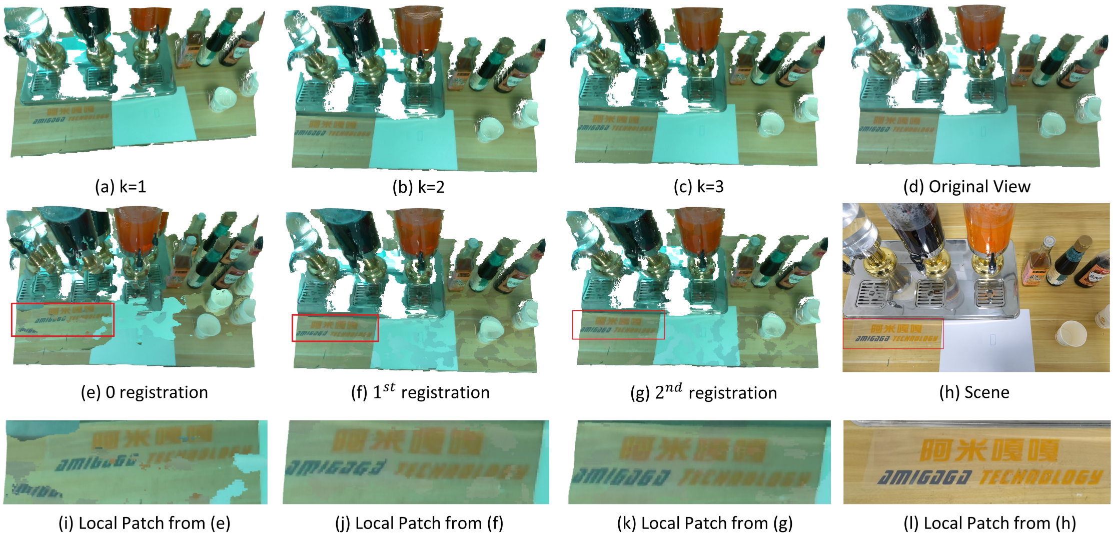
Table 3: The error of IPE's estimation (ΔR_{B_k}, Δt_{B_k})

Factor	*Site 1*	*Site 2*	*Site 2'*
$\overline{\Delta tx}$ / (m)	0.0001	0.0000	-0.0004
$\sigma_{\Delta tx}$ / (m)	0.0004	0.0002	0.0005
$\overline{\Delta ty}$ / (m)	0.0009	0.0010	-0.0005
$\sigma_{\Delta ty}$ / (m)	0.0010	0.0009	0.0007
$\overline{\Delta tz}$ / (m)	0.0007	-0.0009	-0.0010
$\sigma_{\Delta tz}$ / (m)	0.0012	0.0008	0.0010
$\overline{\Delta r}$ / (deg)	0.0190	-0.0106	0.0088
$\sigma_{\Delta r}$ / (deg)	0.1472	0.0123	0.0469
$\overline{\Delta p}$ / (deg)	0.0495	-0.1218	-0.1046
$\sigma_{\Delta p}$ / (deg)	0.1317	0.1119	0.1090
$\overline{\Delta y}$ / (deg)	-0.1338	-0.1423	0.0311
$\sigma_{\Delta y}$ / (deg)	0.1413	0.1221	0.0744

m at Site 1, 0.0019 m at Site 2, 0.0044 m at Site 2'. From the results above, it can be seen that IPE exhibits high accuracy.

Figure 8 shows the error relationship between the actual parking pose and IPE's estimation. Figure 8 (a) shows the error relationship between the actual parking's rotation and the estimated rotation from IPE. Figure 8 (b) shows the error relationship between the actual parking's rotation and the estimated translation (unit: m) from IPE. Figure 8 (c) shows the error relationship between the actual parking's translation (unit: m) and the estimated rotation from IPE. Figure 8 (d) shows the error relationship between the actual parking's translation (unit: m) and the estimated translation (unit: m) from IPE.

Generally, when doing point cloud registration, with every increment of the initial rotation and translation misalignment, the 6D pose estimation error will increase as well. However, with IPE, this is not the case. From Figure 8, we find that the initial rotation and translation change resulting from the parking pose doesn't influence the IPE estimation (i.e. no linear relationship). No matter what the initial rotation and translation is, the IPE's translation estimation error is always below 0.02m and its rotation estimation error is always below 0.02. Note that the IPE error is considerably smaller than the 'Parking' error in Section 4.2. The reason is that IPE has used an iterative way to move the robot arm to sample multiple point clouds at different views until the sampling pose is nearly the same with that at the teaching stage. The final accuracy of IPE will be influenced by the 6D pose estimation method inside IPE (e.g.: we use the colored point cloud registration in this paper). Given that the colored point cloud global registration is proposed in other work (Park et al., 2017; Zhou et al., 2016) and is not the contribution of this paper, experiments for testing the performance of the global colored point cloud registration are omitted. Readers could be directed to the original two papers (Park et al., 2017; Zhou et al., 2016) for more information. From this part, the experiment shows the IPE's strong robustness and accuracy, which is not influenced by the initial rotation and translation.

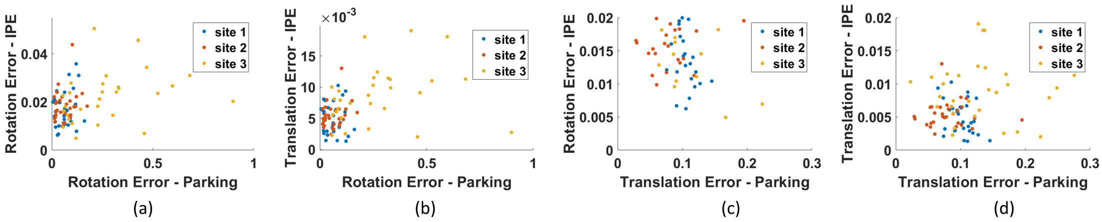


(a) k=1 (b) k=2 (c) k=3 (d) Original View

(e) 0 registration (f) 1^{st} registration (g) 2^{nd} registration (h) Scene

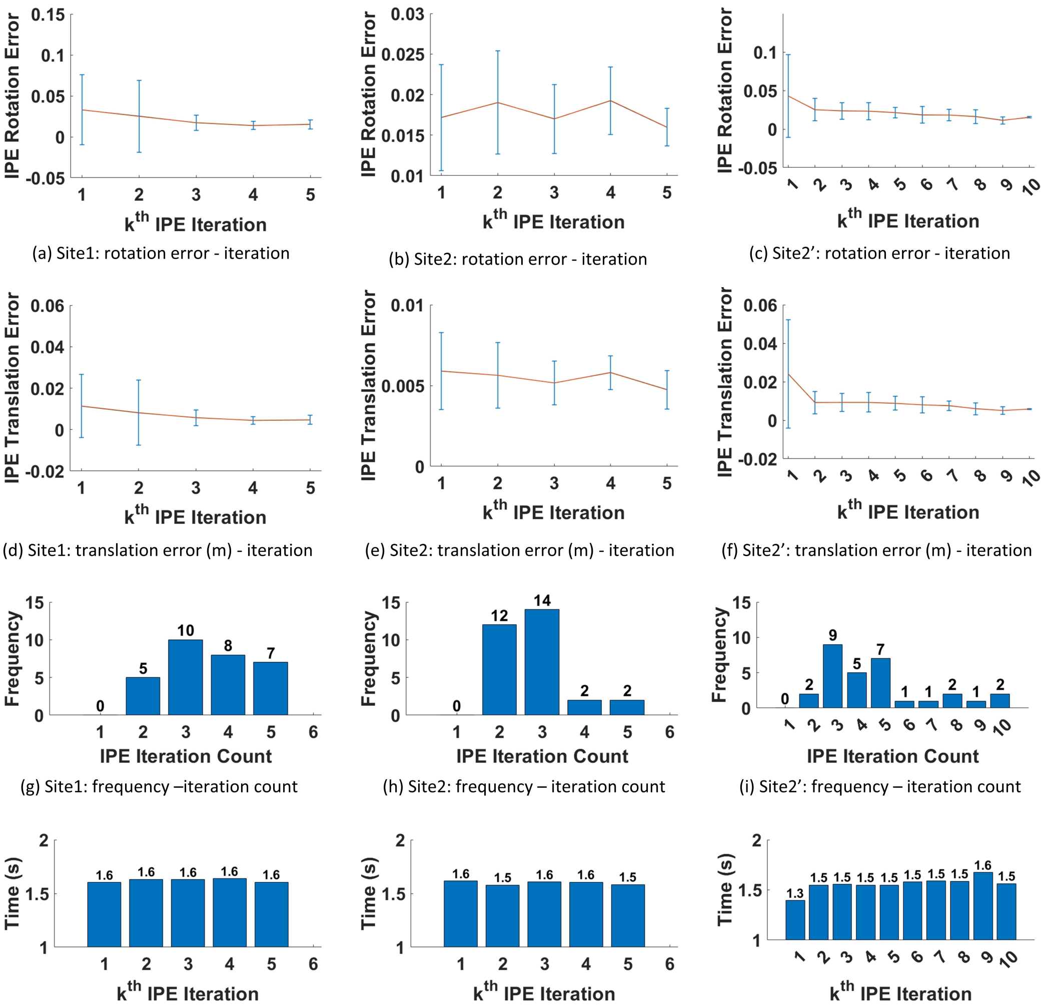
(i) Local Patch from (e) (j) Local Patch from (f) (k) Local Patch from (g) (l) Local Patch from (h)

Figure 7: IPE process. (a) - (c) show the point clouds (from the depth camera) after k^{th} movement of the robot arm in IPE. (d) is the point cloud sampled during the teaching stage. (e) shows the scene before matching the point cloud (a) and (d). (f) shows the scene before matching point cloud (b) and (d). (g) shows the scene before matching the point cloud (c) and (d). (h) is the RGB image of the scene. (i) - (l) are the local pathes from (e) - (h).(Readers are encouraged to view the electronic version of the paper for clearer visual details.)



(a) (b) (c) (d)

Figure 8: Error Relationship between Parking and IPE.



(a) Site1: rotation error - iteration

(b) Site2: rotation error - iteration

(c) Site2’: rotation error - iteration

(d) Site1: translation error (m) - iteration

(e) Site2: translation error (m) - iteration

(f) Site2’: translation error (m) - iteration

(g) Site1: frequency –iteration count

(h) Site2: frequency – iteration count

(i) Site2’: frequency – iteration count

(j) Site1: running time - iteration

(k) Site2: running time -iteration

(l) Site2’: running time - iteration

Figure 9: IPE performance within each iteration.

Figure 9 shows the performance of the IPE algorithm at each iteration. Figure 9 (a) - (c) shows the IPE's estimated rotation error at each iteration for Site 1, Site 2, Site 2'. (d) - (f) shows the IPE's estimated translation error at each iteration for Site 1, Site 2, Site 2'. (g) - (i) shows the distribution of the number of iterations to convergence at Site 1, Site 2, Site 2'. (j) - (l) shows the average running time for registration at k^{th} iteration. From (a) - (f), we find that with every increasing iteration, both the mean and standard deviation of the rotation error and translation error gradually decreases, indicating an increase in accuracy and robustness. The reason is that IPE adjusts the sampling pose of the mounted depth camera incrementally by moving the robot arm to match the initial sampling pose of the camera in the one-shot teaching stage in the world frame. Thus, the newly sampled point cloud in the latest iteration in the automation stage will be more similar to the initial sampled point cloud in the one-shot teaching stage. It means there will be more common features for the colored point cloud registration to match to get a higher matching accuracy. From (g) - (i), we find that IPE needs at least 2 iterations to converge and the majority of the experiments converge by iteration 3-5. The result from the final iteration (in which IPE converges) is more accurate than the result obtained from the first iteration. It proves that the eye-hand iterative strategy in IPE could efficiently reduce the pose estimation error by incrementally positioning the depth camera closer to its initial sampling pose in the one-shot teaching stage. From (j) to (l), we find that the average running time of each registration is stable, usually ranging from 1.5s - 1.7s. It shows IPE's strong robustness against the rotation and translation perturbation. In future work, GPU-based acceleration could be implemented to speed up the point cloud registration algorithm inside IPE in each iteration.

All the experiments on Site 1, Site 2, Site2' are successful and don't exit from IPE algorithm with failure (see Figure 4, "Execution flag = False") because the robot arm's operating range (radius = 0.8 m) is big enough to compensate for the parking error. If the robot parks too far away from the taught position (e.g. ¿ 0.8 meter) and the robot arm fails to find a feasible solution to move the depth camera to the desired pose in the next iteration, IPE will abort with failure definitely. If the colored point cloud global registration could not get converged to make the pose difference fall below the threshold $\boldsymbol{\alpha}$ even when the IPE's loop count exceeds the maximum iteration number β, IPE will abort with failure as well. Given that the colored point cloud global registration is proposed in other work (Park et al., 2017; Zhou et al., 2016) and is not the contribution of this paper, experiments for testing the performance of the global colored point cloud registration are omitted. The two original papers (Park et al., 2017; Zhou et al., 2016) have given their performance (robustness and accuracy) against noise, density, occlusion, overlapping rate, rotation and translation perturbation of the input point clouds.

4.4 Executed Path's Accuracy

Table 4: The position accuracy of the end-effector's path (unit: milimeter)

	Site 1		**Site 2**		**Site 2'**	
Name	**TMMA**	**Ours**	**TMMA**	**Ours**	**TMMA**	**Ours**
mean distance bias	85.1 mm	**1.0 mm**	67.6 mm	**1.1 mm**	null	**1.2 mm**
SD of the distance bias	22.0 mm	**0.5 mm**	32.0 mm	**0.5 mm**	null	**0.6 mm**
maximum distance bias	118.3 mm	**2.0 mm**	130.8 mm	**2.4 mm**	null	**2.8 mm**

In this part, the position accuracy of the end-effector's path $\widetilde{Path_i}(arm)$ will be tested. A needle is attached to the end-effector (See Figure 6 c). The needle tip is pointed at a specified point on the paper (recorded by a red dot) during the teaching stage and the needle tip is required to point to the same point during the automation stage. The bias of the needle tip's position (recorded by a black dot at each trial) from the specified target point (recorded by a red dot) over 30 trials during the automation stage is used to show the accuracy of the executed path's position. Figure 10 records the needle tip's positions in different trials. Because the recorded dots crowd in a very small area (e.g.: a small circle with 2 - 3 mm radius) it is not easy to calculate

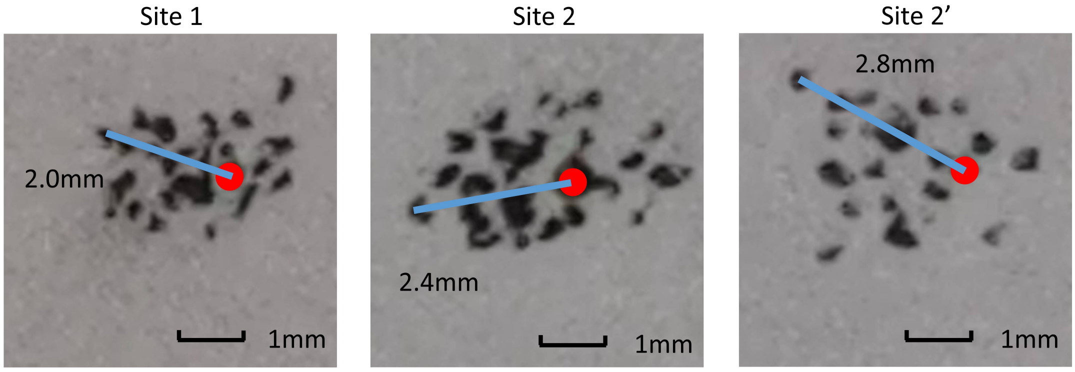


Figure 10: The record of needle tip's positions at different sites. (10x magnification) (Readers are encouraged to view the electronic version of the paper for clearer visual details.)

the distance between each black dot and the red dot accurately enough. Thus, we only measure the maximum distance bias, the mean distance bias and its standard deviation (SD) to describe the position accuracy of the point on the executed path in a rough and approximated manner. According to the experiment results (maximum distance bias is 2.0 mm at Site 1, 2.4 mm at Site 2, 2.8 mm at Site 2'; mean distance bias is 1.0 mm at Site 1, 1.1 mm at Site 2, 1.2 mm at Site 2'; standard deviation of the distance bias is 0.5 mm at Site 1, 0.5 mm at Site 2, 0.6 mm at Site 2'), the end-effector's position accuracy could meet the accuracy requirement of the most applications in daily life. The factors that contribute to the executed path's high accuracy are the high pose estimation accuracy of IPE and the high repeatability of the collaborative robot arm (±0.1mm). This experiment only measures the position accuracy of the executed path, because the orientation of the end-effector[13] is not as important as the position for many manipulation tasks.

Currently, the traditional mobile manipulator automation TMMA (Nair et al., 2019) consists of two stages: in the first stage the mobile base's location and the robot arm's path are recorded; in the second stage the mobile manipulator will park at the same location and replay the arm's path as recorded in the first stage. Compared with TMMA (Nair et al., 2019) using the same experiment setting above, our proposed method achieves much better performance and Table 4 lists the maximum distance bias, the mean distance bias and its standard deviation (SD) using TMMA (Nair et al., 2019) and Ours. Given that the parking place on Site 2' is not even, using the TMMA (Nair et al., 2019) to replay the arm's recorded path without any correction is dangerous (e.g. the robot arm would collide with the desk possibly). Thus, we gave up the TMMA trials on Site 2'. After reviewing Section 4.2 and Table 4, we could find that the position accuracy of the end-effector's path using TMMA is positively correlated with the initial base parking error. Using more expensive and accurate lidar scanners or motor controllers for the base localization or control would boost the parking accuracy, which will increase the TMMA accuracy (Nair et al., 2019) but will also increase the hardware cost drastically in return. Our proposed MMPA method is able to achieve high accuracy and robustness without increasing the hardware cost.

5 Discussion

The core mechanism of MMPA framework is: making the mobile manipulator's end-effector in the automation stage repeat a nearly same path of the end-effector as recorded in the teaching stage in the world coordinate system. Thus, the working environments and the IPE algorithm's performance are the most important factors while implementing MMPA applications in the real world. In Appendix A we did an ablation study to test the IPE's robustness and accuracy with

[13] Except the position of the end-effector, the rest parameters of the robot arm's joints are not fixed and calculated by some motion planning algorithm (e.g.: RRT* (Karaman and Frazzoli, 2011), PRM (Kavraki et al., 1996)).

the change of different related factors in the real world. In Appendix B we reviewed various environments in the real world, and pointed out the difficulty and feasibility of designing real MMPA applications.

Besides the two most important factors above, there are several other factors below, which would help improve the robustness, accuracy and flexibility of the proposed MMPA framework and are left as future work.

In this work, we collect human demonstration data for one-shot teaching through kinesthetic guidance, i.e. guiding the robot through physical contact (Zhu and Hu, 2018). Recent advance in deep learning and imitation learning has allowed the robot arm to learn to perform manipulation tasks by extracting knowledge from human demonstration within videos, eliminating the need of a human to teach the robot through physical contact, thus increasing efficiency and safety (Yang et al., 2019b; Yu et al., 2018). In the future, we will further improve our one-shot teaching procedure by exploring state-of-the-art deep learning and imitation learning approaches.

The colored point cloud registration used in the proposed IPE algorithm for recovering 6D parking pose is accurate but slow. Furthermore, the cheap ”Intel Realsense D435” depth sensor used in the paper is sensitive to the ambient lighting and temperature, which will have a negative impact on the retrieved point cloud's quality, resulting in bad point cloud registration results. In future work, a faster and more robust 6D pose estimation algorithm will be explored. Depth fusion algorithms (e.g. (Pu and Fisher, 2019; Pu et al., 2019)) will be explored using a cheap depth sensor (e.g.: Intel Realsense D435) rather than the expensive sensors (e.g.: Pickit 3D, Zivid) to improve the input point cloud's quality as well. Additionally, the proposed framework in this paper will be further tested in more realistic environments that accurately resemble real-world scenarios such as factory manufacturing, food servicing, quality inspection, etc..

Obstacle avoidance is necessary for the robot arm to work in unpredictable dynamic environments where safety is crucial, such as environments with constant human-robot interaction(Lin et al., 2017). The work in this paper focuses on unmanned controlled environments such as automated factories. Considering that there is little human interaction within these environments, the environments remain unchanged throughout the operation. As long as the trajectory is planned and executed properly, it is unlikely for collisions to happen. Thus, collision avoidance was not necessary and therefore not the focus of this work. Additionally, the robot arm has collision detection functionality built-in within the controller as a safety mechanism by the robot arm manufacturer. Whenever the robot arm senses a collision with an object, it will perform an emergency stop to prevent damage (Haddadin et al., 2017). For future work, collision avoidance can be implemented to allow the mobile manipulator to operate in environments that have uncertain changes over time due to human interactions, such as restaurants. Further hardware upgrades will be done to include more depth cameras, as the current depth camera mounted on the robot arm is not able to construct the real-time 360° 3D map of the surroundings needed for collision avoidance algorithms.

A proper physical gripper is important for a successful manipulation of the rigid or non-rigid items. In the real-life setting, the engineers have to design the proper grippers for different kinds of items which need to be manipulated. The manual design process is time-consuming and expensive. In future work, we will explore the possibility of developing algorithms to autonomously design the physical grippers given the 3D model of the target items.

6 Conclusions

This paper presents a framework for flexible automation that allows a robot to redo multiple tasks at different sites in a controlled environment, after one-shot teaching by a human operator. The framework consists of two stages: the teaching stage and the automation stage. During the teaching stage, the location of the robot base and the path of the robot arm's end-effector have been recorded. During automation stage, the aim is to make the robot arm's end-effector repeat a path as same as the reference path in the world frame. The key for mobile manipulator process automation is the accurate estimation of the relative 6D parking pose between the teaching stage and the automation stage, which is used to adjust the path of the robot arm end-effector. Thus, we propose the IPE algorithm to estimate an accurate relative 6D pose by registering to the camera's initial sampling pose (at teaching stage) in the world coordinate system iteratively.

Acknowledgement

The research funding is from Shenzhen Amigaga Technology Co. Ltd. by the Gagabot2022 project (Grant Agreement No. P987001), from the Human Resources and Social Security Administration of Shenzhen Municipality by Overseas High-Caliber Personnel project (Grant NO. 202102222X, Grant NO. 202107124X) and from Human Resources Bureau of Shenzhen Baoan District by High-Level Talents in Shenzhen Baoan project (Grant No. 20210400X, Grant No. 20210402X) .

Appendix A. Ablation Study

In this part, we will test the impact on the IPE's performance from the pose difference threshold $\boldsymbol{\alpha}$, the IPE's maximum iteration number β and the point clouds from the scenes used in IPE. In each trial, the experiment setting is the same as that in Section 4.3 except for the following factors that are to be compared: the pose difference threshold, the IPE's maximum iteration number, and the point cloud for registration from the scene. Equation (12) and Equation (13) are used to evaluate the rotation and translation error briefly rather than the mean and standard deviation of the 6D pose vector $[tx, ty, tz, r, p, y]$ (short for [translation on x axis, translation on y axis, translation on z axis, roll, pitch, yaw]). We define that successful trial as a trial whose final pose difference is below the pose difference threshold $\boldsymbol{\alpha}$ and whose total iteration number does not exceed the IPE maximum iteration number β. We use the success rate *suc* to represent the rate between the number of the successful trials (which have converged successfully) and the total number of the trials. The average iteration number $\overline{L}$ represents the average iteration number required for the successful trials. $\overline{E_R}$ and σ_{E_R} are the mean and standard deviation of the rotation error only considering the successful trials. $\overline{E_t}$ and σ_{E_t} are the mean and standard deviation of the translation error (unit: meter) only considering the successful trials.

A.1 The pose difference threshold $\boldsymbol{\alpha}$

In this module, the threshold $\boldsymbol{\alpha}$ of the pose difference $(x, y, z, roll, pitch, yaw)$ will be set as $\boldsymbol{\alpha_1}$ = (0.001m, 0.001m, 0.001m, 0.25°, 0.25°, 0.25°), $\boldsymbol{\alpha_2}$ = (0.0015m, 0.0015m, 0.0015m, 0.375°, 0.375°, 0.375°), $\boldsymbol{\alpha_3}$ = (0.002m, 0.002m, 0.002m, 0.5°, 0.5°, 0.5°), $\boldsymbol{\alpha_4}$ = (0.0025m, 0.0025m, 0.0025m, 0.625°, 0.625°, 0.625°), $\boldsymbol{\alpha_5}$ = (0.003m, 0.003m, 0.003m, 0.75°, 0.75°, 0.75°). The $\boldsymbol{\alpha}$ original setting is $\boldsymbol{\alpha_3}$ = (0.002m, 0.002m, 0.002m, 0.5°, 0.5°, 0.5°) in Section 4.3. On each site, we have done 30 trials for each $\boldsymbol{\alpha}$ setting separately. Table A.1, Table A.2 and Table A.3 show how $\boldsymbol{\alpha}$ affects the IPE performance. With $\boldsymbol{\alpha}$ increasing, the success rate *suc* will increase and the average iteration number $\overline{L}$ will decrease. The IPE accuracy increases when $\boldsymbol{\alpha}$ decreases. Considering the tradeoff among the success rate, average iteration number and accuracy, the optimal value should be $\boldsymbol{\alpha_3}$ = (0.002m, 0.002m, 0.002m, 0.5°, 0.5°, 0.5°).

A.2 IPE maximum iteration number β

In this module, the threshold $\boldsymbol{\beta}$ of IPE's maximum iteration number will be set as $\boldsymbol{\beta_1} = 3$, $\boldsymbol{\beta_2} = 4$, $\boldsymbol{\beta_3} = 5$, $\boldsymbol{\beta_4} = 6$, $\boldsymbol{\beta_5} = 7$ for site 1 and site 2. The threshold $\boldsymbol{\beta}$ of IPE's maximum iteration number will be set as $\boldsymbol{\beta_1} = 8$, $\boldsymbol{\beta_2} = 9$, $\boldsymbol{\beta_3} = 10$, $\boldsymbol{\beta_4} = 11$, $\boldsymbol{\beta_5} = 12$ for site 2'. The β original setting is 5 for site 1, site 2 and 10 for site 2' in Section 4.3, which is equal to $\boldsymbol{\beta_3}$ on the corresponding site. On each site, we have done 30 trials for each β setting separately. Table A.4, Table A.5 and Table A.6

Table A.1: The factor $\boldsymbol{\alpha}$ ablation study for Site 1

Factor	$\boldsymbol{\alpha_1}$	$\boldsymbol{\alpha_2}$	$\boldsymbol{\alpha_3}$	$\boldsymbol{\alpha_4}$	$\boldsymbol{\alpha_5}$
suc	0.8333	0.9667	1.0000	1.0000	1.0000
$\overline{L}$	4.0800	3.8276	3.5667	3.2667	2.9667
$\overline{E_R}$	0.0168	0.0168	0.0169	0.0199	0.0215
σ_{E_R}	0.0074	0.0075	0.0074	0.0104	0.0115
$\overline{E_t}$ / (m)	0.0048	0.0049	0.0049	0.0050	0.0052
σ_{E_t} / (m)	0.0024	0.0025	0.0025	0.0026	0.0027

Table A.2: The factor $\boldsymbol{\alpha}$ ablation study for Site 2

Factor	$\boldsymbol{\alpha_1}$	$\boldsymbol{\alpha_2}$	$\boldsymbol{\alpha_3}$	$\boldsymbol{\alpha_4}$	$\boldsymbol{\alpha_5}$
suc	0.8667	0.9667	1.0000	1.0000	1.0000
$\overline{L}$	3.5385	2.9310	2.8000	2.5000	2.3667
$\overline{E_R}$	0.0175	0.0177	0.0183	0.0185	0.0186
σ_{E_R}	0.0063	0.0064	0.0063	0.0065	0.0065
$\overline{E_t}$ / (m)	0.0054	0.0054	0.0054	0.0055	0.0055
σ_{E_t} / (m)	0.0018	0.0019	0.0019	0.0020	0.0022

Table A.3: The factor $\boldsymbol{\alpha}$ ablation study for Site 2'

Factor	$\boldsymbol{\alpha_1}$	$\boldsymbol{\alpha_2}$	$\boldsymbol{\alpha_3}$	$\boldsymbol{\alpha_4}$	$\boldsymbol{\alpha_5}$
suc	0.9667	0.9667	1.0000	1.0000	1.0000
$\overline{L}$	5.8966	5.1379	4.8000	4.1667	4.0000
$\overline{E_R}$	0.0224	0.0224	0.0225	0.0227	0.0228
σ_{E_R}	0.0102	0.0103	0.0103	0.0105	0.0106
$\overline{E_t}$ / (m)	0.0088	0.0088	0.0089	0.0089	0.0090
σ_{E_t} / (m)	0.0042	0.0044	0.0044	0.0045	0.0045

show how β affects the IPE performance. With the increase of β, the success rate *suc* will increase. The average iteration number $\overline{L}$ will increase drastically from β_1 to β_3 and stay steady from β_3 to β_5 (Note: $suc = 1$ for β_3, β_4 and β_5). IPE's accuracy stays nearly the same from β_1 to β_5 because they use the same $\boldsymbol{\alpha}$ setting and $\boldsymbol{\alpha}$ controls the convergence accuracy. Considering the tradeoff among the success rate, average iteration number and accuracy, the optimal value should be $\beta = \beta_3$. It should be noted that increasing β does not guarantee correct convergence. There may exist a case where the IPE is unable to converge correctly (e.g.: in the scene - a plane of pure color), increasing β in such a case will only waste time.

Table A.4: The factor β ablation study for Site 1

Factor	β_1	β_2	β_3	β_4	β_5
suc	0.9000	0.9000	1.0000	1.0000	1.0000
$\overline{L}$	2.8148	3.2593	3.5667	3.5333	3.6000
$\overline{E_R}$	0.0172	0.0170	0.0169	0.0168	0.0169
σ_{E_R}	0.0080	0.0073	0.0074	0.0074	0.0073
$\overline{E_t}$ / (m)	0.0052	0.0050	0.0049	0.0048	0.0048
σ_{E_t} / (m)	0.0029	0.0025	0.0025	0.0024	0.0023

Table A.5: The factor β ablation study for Site 2

Factor	β_1	β_2	β_3	β_4	β_5
suc	0.8667	0.9333	1.0000	1.0000	1.0000
$\overline{L}$	2.5385	2.6429	2.8000	2.8333	2.8000
$\overline{E_R}$	0.0184	0.0183	0.0183	0.0183	0.0183
σ_{E_R}	0.0065	0.0063	0.0063	0.0064	0.0063
$\overline{E_t}$ / (m)	0.0056	0.0054	0.0054	0.0054	0.0055
σ_{E_t} / (m)	0.0021	0.0020	0.0019	0.0018	0.0018

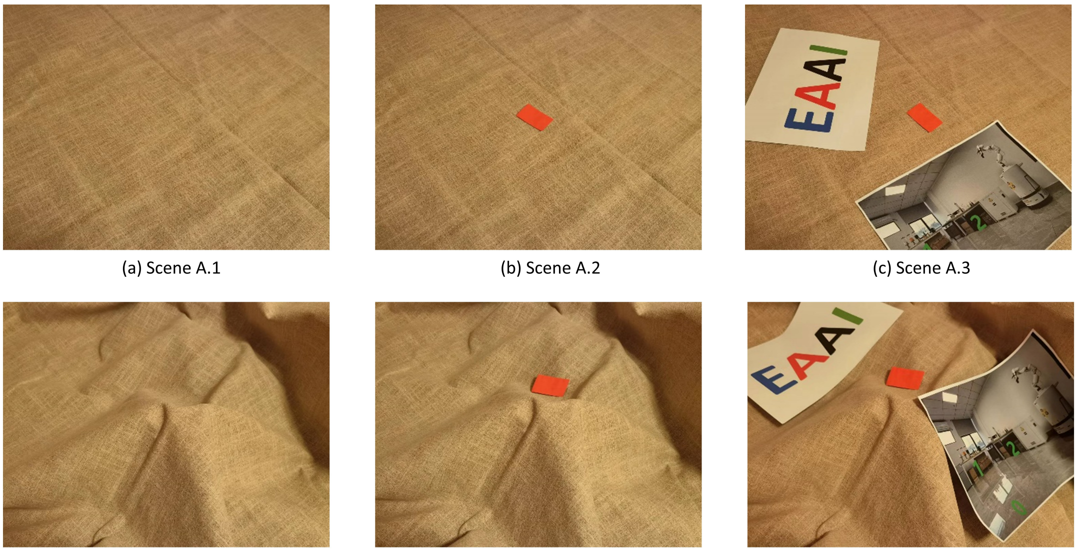
Table A.6: The factor β ablation study for Site 2’

Factor	β_1	β_2	β_3	β_4	β_5
suc	0.9333	0.9667	1.0000	1.0000	1.0000
$\overline{L}$	4.3929	4.6207	4.8000	4.8333	4.8333
$\overline{E_R}$	0.0226	0.0225	0.0225	0.0225	0.0225
σ_{E_R}	0.0104	0.0104	0.0103	0.0102	0.0103
$\overline{E_t}$ / (m)	0.0091	0.0090	0.0089	0.0089	0.0088
σ_{E_t} / (m)	0.0046	0.0044	0.0044	0.0044	0.0044

A.3 The point clouds from the scenes

In this module, more scenes have been used to test the robustness and accuracy of IPE by considering the geometry and color of the scene. In Figure A.1: (a) Scene A.1 shows the scene containing a single plane with single pure color; (b) Scene A.2 shows a scene based on the Scene A.1 with a minor variation of the color using a red dot (which is printed on a thin paper); (c) Scene A.3 shows a scene based on Scene A.1 with rich color and textures (which are printed on three pieces of thin paper); (d) Scene B.1 shows the scene with more geometry features but still with pure color; (e) Scene B.2 shows a scene based on Scene B.1 with a minor variation of the color by using a red dot (which is printed on a thin paper); (f) Scene B.3 shows a scene based on Scene B.1 with rich color using many textures (which are printed on three pieces of thin paper); .

There are 6 groups of experiments where each group of experiments corresponds to a scene. Experiment group Ex A.1, Ex A.2, and Ex A.3 correspond to Scene A.1, A.2, A.3 respectively. Experiment group Ex B.1, Ex B.2, and Ex B.3 correspond to Scene B.1, B.2, B.3 respectively.



(a) Scene A.1 (b) Scene A.2 (c) Scene A.3

(d) Scene B.1 (e) Scene B.2 (f) Scene B.3

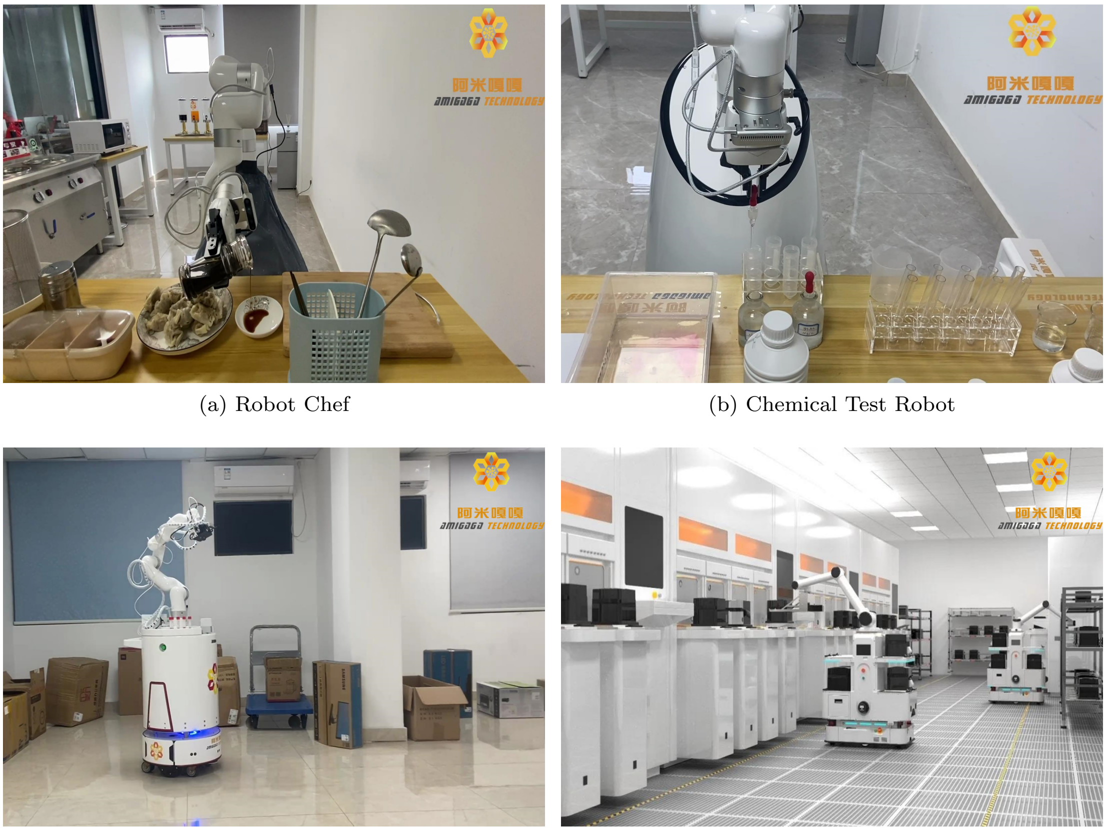
Figure A.1: More Scenes for Test.

There are 2 additional groups of experiments where the color of the scene is not used in IPE[14]. We remove the color information of the point cloud from Scene A.1 and Scene B.1, and treat the corresponding data in Experiment group Ex A.0 and Ex B.0 respectively. Each group of experiments consists of 10 trials. $\boldsymbol{\alpha}$ = (0.002m, 0.002m, 0.002m, 0.5°, 0.5°, 0.5°), β =5. Table A.7 lists the performance result of IPE with different point cloud inputs from various scenes. Ex A.0 - Ex A.2 shows the performance of IPE will collapse when the geometry feature is too poor and there is no or little colorful texture. Ex A.3 shows IPE could work when there is rich colorful texture but poor geometry feature. The reason why the success rate is low in Ex A.3 is that the global point cloud registration (Zhou et al., 2016) in IPE breaks down when the geometry features are poor. If the global registration algorithm could not provide an initial transform to the local colored point cloud registration algorithm (Park et al., 2017), the colored point cloud registration algorithm is easy to fail when the initial transformation between the two registered point clouds is big. Ex B.0 - Ex B.3 shows rich geometry features contribute to a higher success rate. The rich colorful texture contributes to a higher precision.

Table A.7: IPE performance on different scenes

Factor	Ex A.0	Ex A.1	Ex A.2	Ex A.3	Ex B.0	Ex B.1	Ex B.2	Ex B.3
suc	0	0	0	0.5000	0.8000	0.9000	0.9000	**1.0000**
$\overline{L}$	Null	Null	Null	3.6000	3.7500	**3.5556**	3.6667	3.6000
$\overline{E_R}$	Null	Null	Null	0.0160	0.0217	0.0204	0.0206	**0.0155**
σ_{E_R}	Null	Null	Null	0.0078	0.0118	0.0089	0.0073	**0.0072**
$\overline{E_t}$ / (m)	Null	Null	Null	0.0047	0.0048	0.0049	0.0047	**0.0046**
σ_{E_t} / (m)	Null	Null	Null	**0.0023**	0.0025	0.0026	0.0025	**0.0023**

[14] Given the input point cloud has no color and the colored point cloud registration algorithm (Park et al., 2017) is not capable of dealing with it, we replace the colored point cloud registration algorithm used in IPE with ICP (Besl and McKay, 1992).



(a) Robot Chef

(b) Chemical Test Robot

(c) Sampling Robot

(d) Fetching Robot

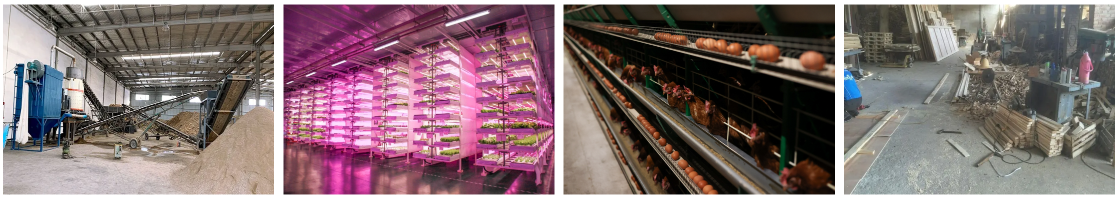
Figure B.1: The figure shows some real applications equipped with the proposed MMPA in unmanned controlled environments. (a) shows a mobile manipulator for fast food cooking in a small unmanned kitchen. (b) shows a mobile manipulator for the acid-base test of the liquid in a quality inspection room of a chemical plant. (c) shows a sampling robot for Covid-19 in one office of the customs. (d) shows a mobile manipulator for fetching and delivering the materials inside a chip-making factory. Please notice that the unmanned environments in these applications are controlled and all the objects related to MMPA are set to fixed positions.

Appendix B. Various Real Environments

Given that the proposed MMPA technique could be applied to various scenarios, we will discuss the required environment characteristics when implementing the proposed MMPA applications in this part. The environment could be classified into two categories: controlled environment and uncontrolled environment. Figure B.1 shows some controlled environments where we could implement the proposed technique. Figure B.2 shows the uncontrolled environments in which the proposed MMPA technique would fail. There are three essential conditions or rules for the proposed MMPA applications.

- **Rule 1**: Ensure that the mobile base could remain steady (not slippery or shaky) on the ground when the robot arm is moving or manipulating objects.
- **Rule 2**: Ensure that IPE could work in that environment (The IPE's ablation study about the environment is shown in Appendix A.3).
- **Rule 3**: Ensure that the positions of the manipulated objects are fixed.

Figure B.1 shows some possible applications in the environments which could meet those three rules above. Here are two short demos to show the mobile manipulators working in an unmanned



(a) Sand Factory (b) Vegetable Factory (c) Chicken Raising Factory (d) Wooden Factory

Figure B.2: The figure shows some uncontrolled environments which would fail the proposed MMPA technique.

kitchen, a quality inspection room of a chemical plant, an office of the customs. The demo video for Figure B.1 (d) in the chip-making factory is not released because of the confidentiality agreement. Sampling robot demo in the office of the customs: `https://youtu.be/6OrD1i11RyE`. Robot chef and chemical test robot demos could be viewed here: `https://youtu.be/mkWIWoaqAAI`. In the demo videos, all the rest tasks are finished by the proposed MMPA technique, except for the recognition and segmentation of the dumplings to be grasped and the goods to be sampled.

Figure B.2 shows some uncontrolled environments which break the three rules required by the proposed MMPA. In Figure B.2 (a), there is a lot of sand on the ground, which will make the robot base slip when the robot arm is moving or manipulating (against Rule 1). In Figure B.2 (b), the lighting will change color (which is unknown to the robot) and the geometry is repetitive and similar, which will collapse the IPE algorithm (against Rule 2). In Figure B.2 (c), the eggs are not fixed and their final positions are unpredictable, which makes the mobile manipulator fail to collect the eggs by the proposed MMPA technique (against Rule 3). In Figure B.2 (d), the floor is slippery because of the sawdust (against Rule 1), the scenes related IPE change constantly and unpredictably because of human activities (against Rule 2), and the tools for robot manipulation are moved constantly by human (against Rule 3).

In conclusion, real environments are complex, therefore making a mature commercial robot application requires a set of advanced techniques including MMPA, instance segmentation, intelligent object grasping and manipulation, etc. The proposed MMPA is one of the basic techniques to make the robot finish various tasks autonomously in a large space and does not require the users to know much professional knowledge of robotics.

Appendix C. List of Abbreviations

MMPA	Mobile Manipulator Process Automation with One-shot Teaching
TMMA	Traditional Mobile Manipulator Automation
SLAM	Simultaneous Localization And Mapping
IPE	Iterative Pose Estimation by Eye & Hand
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
QR	Quick Response
DOF	Degree of Freedom
RRT*	Rapidly Exploring Random Tree
PRM	Probabilistic Roadmap
SD	Standard Deviation

Appendix D. List of Symbols

$XYZO_{MB}$	Mobile Base Coordinate System
$XYZO_B$	Robot Arm Base Coordinate System
$XYZO_C$	Depth Camera Coordinate System
$XYZO_W$	World Coordinate System
$XYZO_n$	Coordinate System for n^{th} Robot Joint
tk_i	Task i
$L_i(vehicle)$	Mobile Platform Parking location for Task i in Teaching Stage
$X_i(O)$	Colored 3D Point Cloud for Task i in Teaching Stage
$Path_i(O)$	Robot Arm Path Information for Task i in Teaching Stage
$\tilde{L}_i(vehicle)$	Mobile Platform Parking location for Task i in Automation Stage
$X_i(k)$	Colored 3D Point Cloud for Task i in k^{th} Iteration in Automation Stage
$\widetilde{Path}_i(arm)$	Adapted Robot Arm Path for Task i
ΔR_{C_k}, Δt_{C_k}	Camera's Relative Rotation and Translation Matrix between Teaching Stage and k^{th} Iteration in Automation Stage
ΔR_{B_k}, Δt_{B_k}	Robot Arm's Estimated Relative Rotation and Translation Matrix between Teaching Stage and k^{th} Iteration in Automation Stage
$R^{C_o}_{B_o}$, $t^{C_o}_{B_o}$	Rotation and Translation Matrix from Coordinate $XYZO_C$ to $XYZO_B$ in Teaching Stage
$R^{C_o}_{B_k}$, $t^{C_k}_{B_k}$	Rotation and Translation Matrix from Coordinate $XYZO_C$ to $XYZO_B$ after k^{th} Iteration Automation Stage
S_{C_o}	One 3D Point in Coordinate $XYZO_C$ in Teaching Stage
S_{B_o}	One 3D Point in Coordinate $XYZO_B$ in Teaching Stage
S_{C_k}	One 3D Point in Coordinate $XYZO_C$ in k^{th} Iteration Automation Stage
S_{B_k}	One 3D Point in Coordinate $XYZO_B$ in k^{th} Iteration in Automation Stage
$\boldsymbol{t_{est}}, \boldsymbol{R_{est}}$	Estimated Translation and Rotation Matrix
$\boldsymbol{t_{gt}}, \boldsymbol{R_{gt}}$	Ground Truth Translation and Rotation Matrix
E_R	Rotation Error
E_t	Translation Error
α_n	Pose Difference Threshold for Test Case n
β_n	Pose Difference Threshold for Test Case n
suc	Success rate
$\overline{L}$	Average Iteration Number Required for Success
$\overline{E_R}$, σ_{E_R}	Mean and Standard Deviation of the Rotation Error Considering Only Successful Trials
$\overline{E_t}$, σ_{E_t}	Mean and Standard Deviation of the Translation Error Considering Only Successful Trials

References

Alexis, K., Darivianakis, G., Burri, M., Siegwart, R., 2016. Aerial robotic contact-based inspection: planning and control. Autonomous Robots 40, 631–655.

Ali, M.A., Mailah, M., 2019. Path planning and control of mobile robot in road environments using sensor fusion and active force control. IEEE Transactions on Vehicular Technology 68, 2176–2195.

Ao, S., Hu, Q., Yang, B., Markham, A., Guo, Y., 2021. Spinnet: Learning a general surface descriptor for 3d point cloud registration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11753–11762.

Bai, S., Lai, J., Lyu, P., Cen, Y., Ji, B., 2021. Improved preintegration method for gnss/imu/in-vehicle sensors navigation using graph optimization. IEEE Transactions on Vehicular Technology 70, 11446–11457.

Bai, S., Lai, J., Lyu, P., Cen, Y., Wang, B., Sun, X., 2022. Graph-optimisation-based self-calibration method for imu/odometer using preintegration theory. The Journal of Navigation , 1–20.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-d shapes, in: Sensor fusion IV: control paradigms and data structures, Spie. pp. 586–606.

Chen, W., Li, X., Ge, H., Wang, L., Zhang, Y., 2020. Trajectory planning for spray painting robot based on point cloud slicing technique. Electronics 9, 908.

Engemann, H., Cönen, P., Dawar, H., Du, S., Kallweit, S., 2021. A robot-assisted large-scale inspection of wind turbine blades in manufacturing using an autonomous mobile manipulator. Applied Sciences 11, 9271.

Fragapane, G., De Koster, R., Sgarbossa, F., Strandhagen, J.O., 2021. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. European Journal of Operational Research 294, 405–426.

Galceran, E., Carreras, M., 2013. A survey on coverage path planning for robotics. Robotics and Autonomous systems 61, 1258–1276.

Grisetti, G., Stachniss, C., Burgard, W., 2007. Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE transactions on Robotics 23, 34–46.

Gu, X., Tang, C., Yuan, W., Dai, Z., Zhu, S., Tan, P., 2022. Rcp: Recurrent closest point for point cloud, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8216–8226.

Haddadin, S., De Luca, A., Albu-Schäffer, A., 2017. Robot collisions: A survey on detection, isolation, and identification. IEEE Transactions on Robotics 33, 1292–1312.

Hess, W., Kohler, D., Rapp, H., Andor, D., 2016. Real-time loop closure in 2d lidar slam, in: 2016 IEEE international conference on robotics and automation (ICRA), IEEE. pp. 1271–1278.

Huang, X., Li, S., Zuo, Y., Fang, Y., Zhang, J., Zhao, X., 2022. Unsupervised point cloud registration by learning unified gaussian mixture models. IEEE Robotics and Automation Letters .

Huang, X., Mei, G., Zhang, J., Abbas, R., 2021. A comprehensive survey on point cloud registration. arXiv preprint arXiv:2103.02690 .

Huynh, D.Q., 2009. Metrics for 3d rotations: Comparison and analysis. Journal of Mathematical Imaging and Vision 35, 155–164.

Junior, E.M.O., Santos, D.R., Miola, G.A.R., et al., 2022. A new variant of the icp algorithm for pairwise 3d point cloud registration. American Academic Scientific Research Journal for Engineering, Technology, and Sciences 85, 71–88.

Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. The international journal of robotics research 30, 846–894.

Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H., 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE transactions on Robotics and Automation 12, 566–580.

Khan, M.U., Zaidi, S.A.A., Ishtiaq, A., Bukhari, S.U.R., Samer, S., Farman, A., 2021. A comparative survey of lidar-slam and lidar based sensor technologies, in: 2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC), IEEE. pp. 1–8.

Khatib, O., Yokoi, K., Chang, K., Ruspini, D., Holmberg, R., Casal, A., 1996. Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96, IEEE. pp. 546–553.

Li, L., Sun, H., Yang, S., Ding, X., Wang, J., Jiang, J., Pu, X., Ren, C., Hu, N., Guo, Y., 2018a. Online calibration and compensation of total odometer error in an integrated system. Measurement 123, 69–79.

Li, Y., Ding, L., Zheng, Z., Yang, Q., Zhao, X., Liu, G., 2018b. A multi-mode real-time terrain parameter estimation method for wheeled motion control of mobile robots. Mechanical Systems and Signal Processing 104, 758–775.

Lin, H.C., Liu, C., Fan, Y., Tomizuka, M., 2017. Real-time collision avoidance algorithm on industrial manipulators, in: 2017 IEEE Conference on Control Technology and Applications (CCTA), IEEE. pp. 1294–1299.

Liu, W., Wu, H., Chirikjian, G.S., 2021. Lsg-cpd: Coherent point drift with local surface geometry for point cloud registration, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15293–15302.

Lv, Y., Peng, Z., Qu, C., Zhu, D., 2020. An adaptive trajectory planning algorithm for robotic belt grinding of blade leading and trailing edges based on material removal profile model. Robotics and Computer-Integrated Manufacturing 66, 101987.

Mason, M.T., Pai, D.K., Rus, D., Taylor, L.R., Erdmann, M.A., 1999. A mobile manipulator, in: Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C), IEEE. pp. 2322–2327.

Meng, J., Wang, S., Li, G., Jiang, L., Zhang, X., Liu, C., Xie, Y., 2021. Iterative-learning error compensation for autonomous parking of mobile manipulator in harsh industrial environment. Robotics and Computer-Integrated Manufacturing 68, 102077.

Mohsin, I., He, K., Cai, J., Chen, H., Du, R., 2017. Robotic polishing with force controlled end effector and multi-step path planning, in: 2017 IEEE International Conference on Information and Automation (ICIA), IEEE. pp. 344–348.

Myronenko, A., Song, X., 2010. Point set registration: Coherent point drift. IEEE transactions on pattern analysis and machine intelligence 32, 2262–2275.

Nair, V.V., Kuhn, D., Hummel, V., 2019. Development of an easy teaching and simulation solution for an autonomous mobile robot system. Procedia manufacturing 31, 270–276.

Nampoothiri, M., Vinayakumar, B., Sunny, Y., Antony, R., 2021. Recent developments in terrain identification, classification, parameter estimation for the navigation of autonomous robots. SN Applied Sciences 3, 1–14.

Nie, J., Wang, Y., Miao, Z., Jiang, Y., Zhong, H., Lin, J., 2021. Adaptive fuzzy control of mobile robots with full-state constraints and unknown longitudinal slipping. Nonlinear Dynamics 106, 3315–3330.

Nubert, J., Khattak, S., Hutter, M., 2022. Graph-based multi-sensor fusion for consistent localization of autonomous construction robots. arXiv preprint arXiv:2203.01389 .

Park, J., Zhou, Q.Y., Koltun, V., 2017. Colored point cloud registration revisited, in: Proceedings of the IEEE international conference on computer vision, pp. 143–152.

Pu, C., Fisher, R.B., 2019. Udfnet: Unsupervised disparity fusion with adversarial networks, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 1765–1769.

Pu, C., Li, N., Tylecek, R., Fisher, B., 2018. Dugma: Dynamic uncertainty-based gaussian mixture alignment, in: 2018 International Conference on 3D Vision (3DV), IEEE. pp. 766–774.

Pu, C., Song, R., Tylecek, R., Li, N., Fisher, R.B., 2019. Sdf-man: Semi-supervised disparity fusion with multi-scale adversarial networks. Remote Sensing 11, 487.

Ramasubramanian, A.K., Mathew, R., Preet, I., Papakostas, N., 2022. Review and application of edge ai solutions for mobile collaborative robotic platforms. Procedia CIRP 107, 1083–1088.

Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., Pan, J., Patil, S., Goldberg, K., Abbeel, P., 2014. Motion planning with sequential convex optimization and convex collision checking. The International Journal of Robotics Research 33, 1251–1270.

Shan, T., Englot, B., 2018. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 4758–4765.

van Sosin, B., Bartoň, M., Elber, G., 2019. Accessibility for line-cutting in freeform surfaces. Computer-Aided Design 114, 202–214.

Sucan, I.A., Moll, M., Kavraki, L.E., 2012. The open motion planning library. IEEE Robotics & Automation Magazine 19, 72–82.

Urrea, C., Agramonte, R., 2021. Kalman filter: historical overview and review of its use in robotics 60 years after its creation. Journal of Sensors 2021.

Van Nam, D., Gon-Woo, K., 2021. Solid-state lidar based-slam: A concise review and application, in: 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), IEEE. pp. 302–305.

Wang, H., Peng, J., Zhang, F., Zhang, H., Wang, Y., 2022. High-order control barrier functions-based impedance control of a robotic manipulator with time-varying output constraints. ISA transactions .

Welch, G., Bishop, G., et al., 1995. An introduction to the kalman filter .

Wong, C., Yang, E., Yan, X.T., Gu, D., 2017. An overview of robotics and autonomous systems for harsh environments, in: 2017 23rd International Conference on Automation and Computing (ICAC), IEEE. pp. 1–6.

Wong, C., Yang, E., Yan, X.T., Gu, D., 2018. Autonomous robots for harsh environments: a holistic overview of current solutions and ongoing challenges. Systems Science & Control Engineering 6, 213–219.

Wu, B., Ma, J., Chen, G., An, P., 2021. Feature interactive representation for point cloud registration, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5530–5539.

Yamamoto, Y., Yun, X., 1992. Coordinating locomotion and manipulation of a mobile manipulator, in: [1992] Proceedings of the 31st IEEE Conference on Decision and Control, IEEE. pp. 2643–2648.

Yan, Q., Cai, J., Ma, Y., Yu, Y., 2019. Robust learning control for robot manipulators with random initial errors and iteration-varying reference trajectories. IEEE Access 7, 32628–32643.

Yang, J., Li, H., Campbell, D., Jia, Y., 2015. Go-icp: A globally optimal solution to 3d icp point-set registration. IEEE transactions on pattern analysis and machine intelligence 38, 2241–2254.

Yang, M., Yang, E., Zante, R.C., Post, M., Liu, X., 2019a. Collaborative mobile industrial manipulator: a review of system architecture and applications, in: 2019 25th International Conference on Automation and Computing (ICAC), IEEE. pp. 1–6.

Yang, S., Zhang, W., Lu, W., Wang, H., Li, Y., 2019b. Learning actions from human demonstration video for robotic manipulation, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 1805–1811.

Yu, T., Finn, C., Xie, A., Dasari, S., Zhang, T., Abbeel, P., Levine, S., 2018. One-shot imitation from observing humans via domain-adaptive meta-learning. arXiv preprint arXiv:1802.01557 .

Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T., 2017. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1802–1811.

Zhou, Q.Y., Park, J., Koltun, V., 2016. Fast global registration, in: European conference on computer vision, Springer. pp. 766–782.

Zhou, X., Wang, X., Xie, Z., Li, F., Gu, X., 2022. Online obstacle avoidance path planning and application for arc welding robot. Robotics and Computer-Integrated Manufacturing 78, 102413.

Zhu, Z., Hu, H., 2018. Robot learning from demonstration in robotic assembly: A survey. Robotics 7, 17.