Robust Pipeline for Mobile Brick Picking

Simon Fryc, Liyang Liu, and Teresa Vidal-Calleja
Centre for Autonomous Systems at the Faculty of Engineering and IT, University of Technology Sydney
simon.fryc@uts.edu.au, liyang.liu@uts.edu.au, teresa.vidalcalleja@uts.edu.au

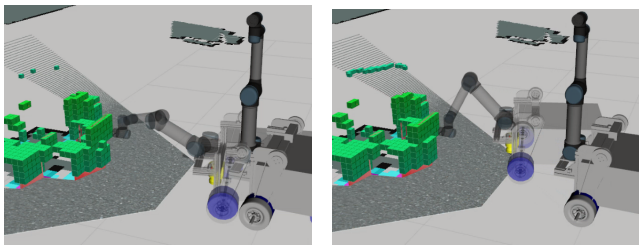
Abstract

In this work, we are interested in the problem of picking a single brick shaped object from an unstructured pile using a mobile manipulator and a 3D camera system. We propose a robust multi-stage pipeline for efficient, collision-free brick picking given the pose of a target object. The key contribution of this work is a scoring function used to find the most suitable configuration considering the integrated kinematic chain of a mobile base and manipulator arm. Realistic simulation results show the proposed pipeline has 100% success rate as opposed to a standard off-the-shelf solution, which has high-failure rates.

1 Introduction

Although significant progress has been made towards the development of autonomous mobile manipulators in recent years, it remains highly desirable to develop robots that can navigate and manipulate objects autonomously in unstructured and dynamic environments such as outdoors scenarios. The work presented in this paper is motivated by the 2020 Mohamed Bin Zayed International Robotics Challenge (MBZIRC) [mbz, 2019] which requires to develop mobile manipulators capable of autonomously locating and picking different types of bricks from unorganised piles to build a predefined structure. These types of robotic systems present new opportunities in a range of commerce and research applications including disaster response, logistics, manufacturing, and construction.

We refer to mobile manipulation as a robotic system that typically consist of a robotic manipulator arm mounted on a mobile platform equipped with sensors, which removes the work space limitations imposed by static bases by extending the manipulators total number of degrees of freedom. The robot's mobile base gives it the freedom to manoeuvre around its environment while its manipulator enables it to achieve the precision required to perform complex manipulation tasks. Mobile robots are typically divided into two distinct classes based on their drive mechanism. Robots with a differential drive mechanism have non-holonomic constraints which prevent them from moving perpendicular to their wheels. Conversely, the driving mechanisms of omni-directional robots allow them to become holonomic, meaning that they are capable of driving in any direction in a 2D plane.



(a) (b)

Figure 1: Given the two inverse kinematic solutions shown in the simulated environment, we can see (a) would be unfavourable because the robot manipulator is overly outstretched resulting in a low manipulability score, while in the (b) the robot has a much higher ability to move freely in any direction. Since the solution in (a) is more favourable it will obtain a higher score.

A key challenge in mobile manipulation is the interaction between motion planning and perception that will deliver stable and efficient solutions for the purposes of the application in place. In this paper, we are interested in the problem of picking a single brick shaped object from an unstructured pile using a mobile manipulator and a 3D camera system. To achieve this there are a number of sub-problems that need to be considered, such as where the robot should position itself to reach the target object, and the generation of efficient and collision-free motion plans.

The simplest approach to solve the brick picking problem may be for the robot to drive toward a target object until it reaches a predefined offset from which it assumes it can reach the object. Although this simplistic approach may work in practice, there is no guarantee that the manipulator will be able to reach a target object on every occasion. This means the robot would have to re-position itself by planning and executing a new trajectory, leading to significant losses in system efficiency. Moreover, mobile manipulators introduce multiple degrees of freedom and a virtually infinite number of configurations that are achievable in a given unconstrained environment.

In this paper we present a robust multi-stage pipeline for efficient, stable and collision-free brick picking. Our approach combines the kinematic model of the robot base with that of the manipulator. The proposed method generates motion plans for the entire robot while considering 3D environment mapping, collision avoidance and trajectory execution. The main contribution of this work is a manipulability score used to find the most suitable configuration for the entire mobile manipulator. Figure 1 shows an example of this task with two types of manipulability score. Off-the-shelf packages such as MoveIt [Chitta *et al.*, 2012] can provide such functionality with seamless integration of perception, motion planning and position control given the full kinematic chain. However, this package is commonly slow and prone to failure when generating motions spanning large distances with redundant manipulators.

The proposed multi-stage pipeline aims to prevent these planning failures and improve system robustness via the scoring function and by breaking down the task into smaller planning problems, while considering the surroundings. The key insight is to exploit collision free planning and decoupled motion execution at different stages of the task. Our method uses efficient predefined trajectories to move the arm to a position that has been pre-qualified as having a high planning success rate to the most likely grasp configurations.

The proposed approach is evaluated in a realistic simulated environment. The pipeline considers all aspects to enable full autonomy to pick the objects while maintaining an awareness of its environment for collision detection and avoidance. A UR5 manipulator is mounted on a differential mobile base. Perception is incorporated into the system using a customised implementation of the OctoMap 3D mapping framework [Hornung *et al.*, 2013] using a 3D camera. The results show the proposed pipeline produces no failures, while standard MoveIt! framework [Chitta *et al.*, 2012] has a high-failure rate.

2 Related work

According to [Khatib *et al.*, 1996] mobile manipulators could be used for a broad range of robotic applications including, but not limited to, space, construction and service industries. The authors leverage their knowledge of traditional fixed place systems to develop new methodologies for the analysis and control of cooperative mobile manipulation systems, with a focus on decentralised control. It is highlighted how a mobile robot can be considered as a redundant mechanism that utilises the coarse and adaptable base for coarse positioning, and the fast and accurate arm for fine manipulation.

The authors in [Hamner *et al.*, 2009] support the view that mobile robots have immense potential for assembly tasks in both terrestrial and extra-terrestrial locations due to their inherent flexibility and ability to manage system uncertainties and unknowns which conventional robots are unlikely to encounter in highly structured factory applications. The authors recognise that in order to reach their full potential these robots must be equipped with the appropriate sensors and control algorithms so that they can perceive and react appropriately to dynamic changes in their environment. Their research endorses a control scheme that uses coordinated motion of the base and arm in combination with visual feedback and force servoing.

A common application of mobile manipulators is in civilian and military missions that involve hazardous environment, as shown by [Dong Hun Shin *et al.*, 2003]. These applications often invoke a decoupled representation of the mobile base and arm to maintain the ease of human tele-operation. This implies that the robot base must first be driven to some desired location before being parked by the operator who would then switch to manipulator control. Such a system would not be considered as autonomous and [Hamner *et al.*, 2009] argues that by treating a mobile robot as two independent robots the system is liable to inefficiencies, due to a discontinuous and interrupted workflow.

[Yamamoto and Xiaoping Yun, 1994] considered the implications of robot base positioning for manipulation by observing the way in which humans position their bodies when writing on a whiteboard. It makes sense that a human would position their body in such a way as maintain a high measure of manipulability in their arm while writing to avoid an overly outstretched configuration. Similarly, when carrying heavy objects, humans tend to adopt a pose that minimises the load on their body. [Yamamoto and Xiaoping Yun, 1994] discerned that "it is desirable to bring the manipulator into certain preferred configurations by appropriately planning the motion of the mobile platform. This is particularly challenging if the trajectory of the manipulator is not known a priori, in which case an autonomous system

must perform planning for the base in real time. To address this problem, the authors developed an algorithm that accounts for the preferred operating region of the arm by observing its joint positions.

Although many existing approaches also divide mobile manipulation into a series of sub-problems it is often the case that the base and manipulator are decoupled and optimised independently [Berenson *et al.*, 2008]. There are few examples of robust motion planning methods such as ours that utilise the complete kinematic chain of a mobile manipulator. Even fewer examples exist of approaches that combine the efficiency of pre-defined trajectories with collision-free motion planning within dynamic environments.

3 Problem definition

Our method aims to address the problem of efficient mobile manipulation for the purposes of brick picking from a random pile. The proposed pipeline takes advantage of a specific setup to quickly plan and execute a series of efficient, collision-free, collision-aware and fine adjustments motion plans. We define the mobile manipulation problem for the purposes of brick picking from a random pile formally as follows.

Considering the manipulator and mobile base as a single kinematic chain, let us define a homogeneous transform from a world frame $T_W^B \in \mathrm{SE}(3)$ to the robot base. The intent is to find a solution for the transform T_W^B that, when coupled with the configuration of the arm T_B^E, transforms the robot end effector from the world frame to the target pose T_W^E such that,

$$T_W^E = T_W^B T_B^E \in \mathrm{SE}(3)\,. \tag{1}$$

Given that forward kinematics describes the relationship between the set of joint positions q in a robot arm and the pose of the end effector ξ_E, the function K determines the pose of the end effector with respect to the manipulator's base as a function of the manipulator's joint angles by applying homogeneous transformations between the frame of each of the links N,

$$\begin{gathered} \xi_N = K(q) \\ \text{where, } q = \{q_j,\, j \in [1 \ldots \mathrm{N}]\}\,. \end{gathered} \tag{2}$$

Thus, Inverse Kinematics (IK) is used to determine the joint configurations q that satisfies the final term in Equation (1) in order to allow the end effector to reach the target pose ξ_E such that:

$$q = K^{-1}(\xi_E)\,. \tag{3}$$

Determining the joint angles q^* can be posed as an optimization problem where we want to minimise the relative pose error of the end effector between the current pose $K(q)$ and the desired pose ξ^*:

$$q* = \underset{q}{\operatorname{argmin}} \left| K(q) \ominus \xi^* \right|. \tag{4}$$

We define a vector containing the pose of the base (x, y, ϕ) as well as the joint angle vector of the manipulator θ to be:

$$q = (x,\, y,\, \phi,\, \theta) \tag{5}$$

By considering the manipulator and mobile base as a single kinematic chain the inverse kinematic planning returns motion plans for the full configuration on the robot qR including joint trajectories for both the manipulator qM and base qB such that:

$$\begin{aligned} qB(x, y, \phi) &\subset qR \\ qM(\theta) &\subset qR \end{aligned} \tag{6}$$

By defining the mobile base as a planar joint the resulting trajectory assumes that the robot base is omni-directional, which might not be the case for many mobile robots. To resolve this issue, our approach allows the components for the motion of the base platform and the manipulator to be extracted from the trajectory independently. That is to say that only the final waypoint, i.e. the final x and y positions within the base trajectory are to be considered for navigation. Note that for an omni-directional platform the trajectory can be executed in full, taking the robot from its initial configuration to a collision-free grasping configuration.

4 Multi-Stage Approach

The proposed approach considers partition of the positioning task for brick picking. The full picking task is subdivided into five discrete logical stages:

- Stage 1: Generation and scoring a set of plans for entire mobile manipulator model
- Stage 2: Motion execution for mobile base only
- Stage 3: Octomap and target pose update with respect to new base location
- Stage 4: Motion execute a pre-planned, collision-free trajectory for manipulator only
- Stage 5: Re-planning and motion execution for manipulator considering obstacles to a grasp pose q^*

In the first stage we model the mobile manipulator as a single kinematic chain where the mobile base is represented as a planar joint connecting the base of the manipulator to the world frame, as shown in Figure 2. An OctoMap representation of the environment and a target pose are required as inputs before inverse kinematics is applied to the chain to generate a set of motion plans

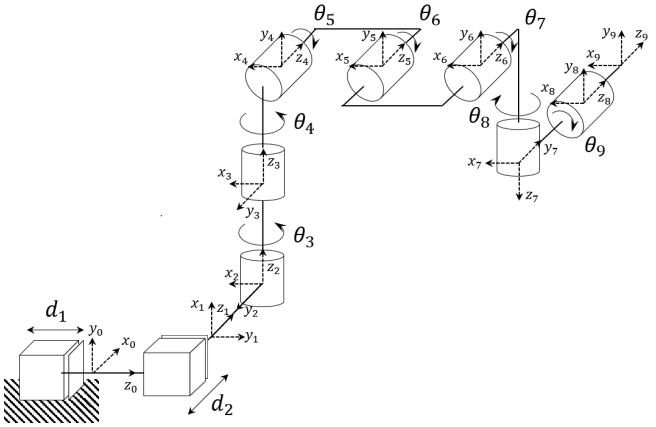


Figure 2: The kinematic chain of a mobile UR5 manipulator using the D-H model [Hartenberg and Denavit, 1955]. The D-H model is a concise way of describing the spatial relationships between serial link mechanisms. The notation allows to define the offsets between the coordinate frames of 2 links using just four parameters, θ, d, a and α. The D-H parameters for the mobile UR5 robot are listed in Table 1.

for the entire robot model. A scoring function is used to both validate and evaluate each motion according to the final manipulability index of the manipulator and the trajectory duration. In the case of an omni-directional robot the full motion plan is forwarded to the controllers for execution. For differentially driven robots the motion plan for the base is extracted and sent to the base controller.

Once the base has reached its goal, its frame with respect to the target will have changed, so perception is run again to update the target pose. At this point the manipulator proceeds to execute a pre-defined, collision-free trajectory that moves the gripper to the left or right side of the robot based on the relative position of the target. A conventional planner is used to generate the final motion plan that will take the gripper to the target pose considering all obstacles.

5 Scoring Function

Our approach takes advantage of the fact that a mobile manipulator is a redundant mechanism which has a greater number of degrees of freedom available than are required of a manipulation task in 3D space. It follows then that the inverse kinematics solution is not unique and there may be multiple joints configurations that enable the robot to reach a target pose.

The proposed method uses a simple optimisation algorithm to validate and score a set of motion plans. The algorithm works by generating a set of 10 motion plans and scoring each plan based on two criteria. The first

Joint	**Type**	$\boldsymbol{\theta}$	**d**	**a**	$\boldsymbol{\alpha}$
1	P	$\pi/2$	d_1	$\pi/2$	0
2	P	-$\pi/2$	d_2	-$\pi/2$	0
3	R	θ_3	0	0	0
4	R	θ_4	0.089	-0.425	$\pi/2$
5	R	θ_5	0	-0.393	0
6	R	θ_6	0	0	0
7	R	θ_7	0.109	0	$\pi/2$
8	R	θ_8	0.095	0	-$\pi/2$
9	R	θ_9	0.082	0	0.000

Table 1: D-H parameters for a UR5 Mobile Manipulator that define the geometric relationships between the frames of each robot link.

criterion is the manipulability index of the arm and the second is the time it would take to execute the trajectory. The objective of the function is to find the plan where the trajectory execution time is minimised, and the manipulability index is maximised. The score of each plan S is calculated using a Sigmoid function [Kros *et al.*, 2006] to map the raw metrics of arbitrary range into a well controlled range of [0...1] thereby equalising their contribution:

$$\begin{aligned} T_{score} &= \frac{2}{1+\exp\left(\frac{t_T}{10}\times 5\right)} \\ M_{score} &= \frac{4}{1+\exp -\left(\frac{m}{0.1}\times 5\right)} - 2 \end{aligned} \tag{7}$$

where, M_{score} is clamped to zero if $m < 0.04$, then

$$S = T_{score} \times M_{score} \tag{8}$$

If a base location is chosen which results in the arm having a low manipulability index, then the capability of the arm in performing pick and place manoeuvres efficiently would be greatly reduced. We consider that the manipulability index is of greater importance than the trajectory execution time so double the weighting is applied to its contribution towards the score. This warrants that a longer trajectory that results in a higher manipulability index is chosen over a shorter one.

The measure of manipulability [Yoshikawa, 1985] is used in our approach as one of the main scoring criteria when determining the optimal robot base position. A manipulator that has a high measure of manipulability can readily move in any direction with ease, whereas with a low measure of manipulability the configuration of the robot constrains its motion and it may be very difficult for the robot to move in certain directions. During the planning process, kinematic solutions that offer a higher manipulability are favoured over those that result in a lower measure so that the arm retains its agility in its

workspace as depicted in Figure 1. For this process we use the determinant of the Jacobian matrix to calculate the robots measure of manipulability m as follows,

$$m = \sqrt{\det JJ^T(\theta)}\,. \tag{9}$$

6 Implementation

Our approach was implemented using the MoveIt! software platform [Chitta *et al.*, 2012] within the Robot Operating System (ROS) development environment. MoveIt! has the ability to integrate kinematics, motion planning, trajectory execution, collision avoidance and environment monitoring in a single framework. The framework has been featured in a range of solutions in well-known robotics challenges requiring unstructured autonomous pick and place and mobile manipulation, such the Amazon picking challenge and DAPRA robotics challenge [Chitta *et al.*, 2012].

In our implementation the functionality offered by MoveIt! was employed and built upon using its C++ API user interface called the MoveGroup class. This is the main class responsible for handling most operations such as processing pose requests, creating motion plans, executing motion and modifying the robot collision model and its environment. Tailored algorithms were developed using this C++ API to demonstrate our approach.

The D-H parameters from Section 4 were used to develop the Universal Robot Descriptor Format (URDF) model of the mobile base with an integrated UR5 manipulator. Note that the brick pose is considered given and therefore outside of the scope of this paper. For the purposes of testing our proposed method, AR markers are used to provide the pose of a target brick.

6.1 Mobile Base Planning

In our experiments inverse kinematics was implemented using MoveIt's TRAC_IK plugin which runs both an enhanced version of KDL's joint-limited pseudoinverse Jacobian solver [Beeson and Ames, 2015] and a Sequential Quadratic Programming IK formulation.

Given a target pose, a motion planning request is generated and passed to a motion planning node. In here, we constrain the final orientation of the robot colinear to the target object. Thus, the end-effector is constrained to always be facing perpendicular to the brick pile.

In our implementation Moveit! is set to use either the robot manipulator alone or the manipulator in conjunction with the mobile base. The resulting motion plan is a trajectory that will move the mobile manipulator from one configuration to another while following the maximum joint velocities and accelerations. Each waypoint in the trajectory stipulates the position, velocity and acceleration for all the joints in the kinematic chain at the given instant.

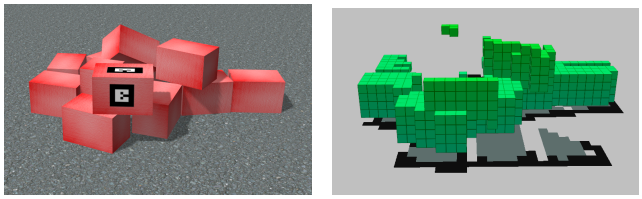


Figure 3: An OctoMap representation of the random brick pile for collision-free planning.

6.2 Perception

Perception is incorporated into the system using a depth sensor and a custom implementation of the OctoMap 3D mapping framework [Hornung *et al.*, 2013] which utilises the octree data structure [Franklin and Akman, 1985] to provide probabilistic 3D occupancy mapping. The probabilistic map gives the robot an understanding of its environment which it can use for motion planning and collision avoidance as shown in Figure 3. The algorithm operates by subdividing 3D space into volumetric pixels called voxels that can then be labelled as either free or occupied based upon probabilistic estimations of 3D sensor measurements. For our method a simulated Intel RealSense RGB-D camera was used to provide the depth images and point cloud data required by the OctoMap node (Figure 3).

OctoMap Customisation

OctoMap uses ray casting to determine which cells to mark as occupied and which to mark as unoccupied (Figure 4a). A significant challenge when using the OctoMap framework is that it only incorporates free space into the map when it can ray trace through an occupied cell to another occupied cell behind it (Figure 4b), or when the distance between a point and the sensor origin is larger than the sensors maximum range as defined in the configuration files. In an open, outdoor environment there is a high chance that there might not be any objects for the sensor to detect. This means that if an object is moved the space which it previously occupied in the OctoMap will not update to unoccupied (Figure 4c). In the case that the any of the sensors pixels fails to receive valid depth readings the depth value of these pixels if filled with undefined values (i.e. NaN, not a number). To overcome this issue a virtual wall is created (Figure 4d) by in the first converting any NaN pixel values from the original point cloud (Figure 5a) to the sensors maximum range (Figure 5b).

A random sample consensus (RANSAC) [Alehdaghi *et al.*, 2015] algorithm is run on this point cloud using the Point Cloud Library (PCL) to segment the foreground from the ground plane and the background. Ray casting is computationally expensive so in an effort to reduce the

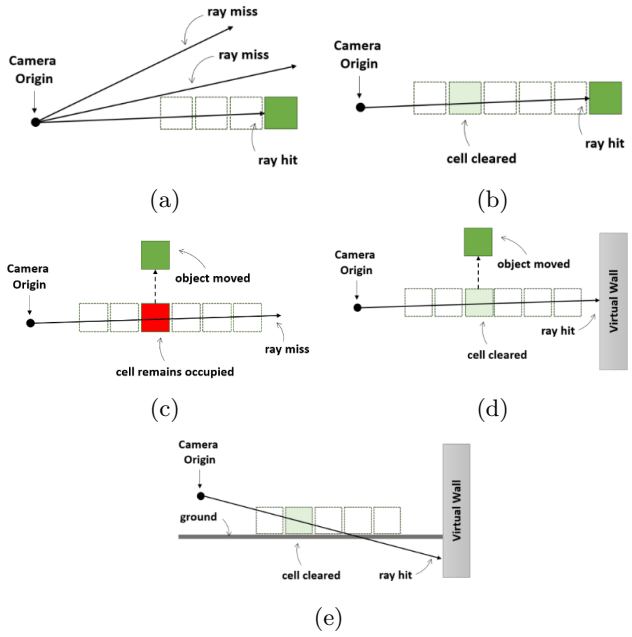


Figure 4: Ray casting is a process that involves projecting a ray between the sensor origin and each point in the point cloud while observing which cells in the Octree it intersects.

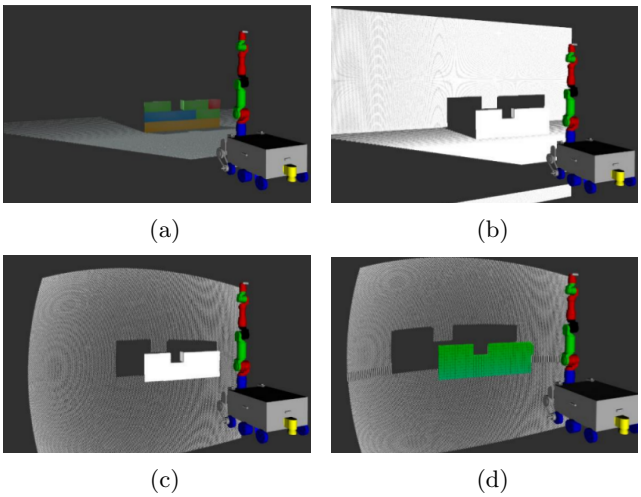


Figure 5: OctoMap Point cloud processing.

amount of ray casting required any points on the ground plane and background are projected onto a sphere whose radius is equal to the depth sensor's maximum range (Figure 5c). The ground plane must be projected onto the sphere so that occupied cells close to the ground are cleared (Figure 4e).

It is not necessary to process the background at a high resolution, so to maintain a high level of fidelity in the final OctoMap only objects in the foreground are processed at a higher resolution. An example of the resulting occupancy map that is capable of incorporating dynamic changes in the environment is shown in Figure 5d.

6.3 Pre-planned Trajectory Execution

Once the robot base has reached the goal coordinates in the X-Y plane a grasp pose for the end effector can be calculated based on the updated robot position. Mobile robots often leave the manipulation in a tucked position when driving from one location to another to keep the robot compact and prevent collisions with the environment. Rather than generating a motion plan using MoveIt! between such a tucked configuration and the grasp configuration, which has a relatively high failure rate, our method uses efficient predefined trajectories to move the arm to a position that has been pre-qualified to find a successful plan. In this approach two pre-defined trajectories are used to position the end effector to the front left or front right side of the robot base in preparation for picking. The use of pre-planned trajectories assumes that the robots immediate environment is static and that the IK solution for the base will not position the robot in such a way as to introduce collisions along the pre-planned trajectory. The latter problem can be alleviated using a prior knowledge of the robot environment and the yaw joint constraints discussed previously to ensure the robot is perpendicular to the target object.

6.4 Collision free motion planning

Moveit! is used within our approach in the conventional manner to move the arm from the pre-planned pose to the grasp pose considering the objects within the environment. This final stage takes full advantage of Moveit! collision avoidance using the Open Motion Planning Library (OMPL). OMPL is a motion planning interface in MoveIt that offers a selection of randomized sampling-based motion planners. The planner chosen for the manipulator was the RRT Connect planner [Kuffner and Lavalle, 2000] which is an extension of the Rapidly Exploring Random Tree algorithm (RRT) proposed by [Lavalle, 1998]. The algorithm is designed to be an efficient single-query path planner [Kuffner and Lavalle, 2000]. It works by constructing two RRT structures at rooted at the start pose and goal pose that incrementally converge by randomly exploring the workspace.

7 Simulated Results

The mobile UR5 manipulator and randomised brick piles were simulated in the Gazebo 3D simulator which is designed to model dynamic outdoor environments, and interfaces directly with ROS. The backbone of Gazebo is the Open Dynamics Engine (ODE) physics engine that simulated rigid body dynamics and collision detection.

To evaluate our approach six different pile configurations were generated with the target brick in different locations with varying grasping difficulty as shown in Figure 6. Our 5-stage planning approach was evaluated against the simplistic approach where the robot drives to a fixed offset from the pile and attempts to grasp the target brick using the conventional MoveIt planning pipeline.

The quantitative data in Table 2 confirms that our approach was able to solve the mobile manipulator planning problem successfully for each unique pile configuration. Whereas the simplistic approach often failed due to either a lack of reach, or an inability to generate a successful motion plan.

Furthermore, the data shows that the 5-stage approach often resulted in the manipulator obtaining a higher manipulability index. It follows then that our approach would be suitable for manipulation tasks where the gripper must maintain a high level of dexterity.

Our results show that generating successful motion plans using our approach often results in a longer duration for the picking process which can be attributed to two factors. The first is that the simplistic approach only commands the robot to drive forward along a straight line trajectory, whereas the base solution generated by our planner often require the robot to navigate the environment to a position further away from its starting position. Secondly, the average time taken to generate a single plan is in the region of 0.552 seconds which means that in order to generate a set of 10 plans is approximately 5 seconds, during which time the robot is stationary. It should be noted that the planning time is directly proportional to the complexity of the planning problem and the physical distance between the robot base and the target object.

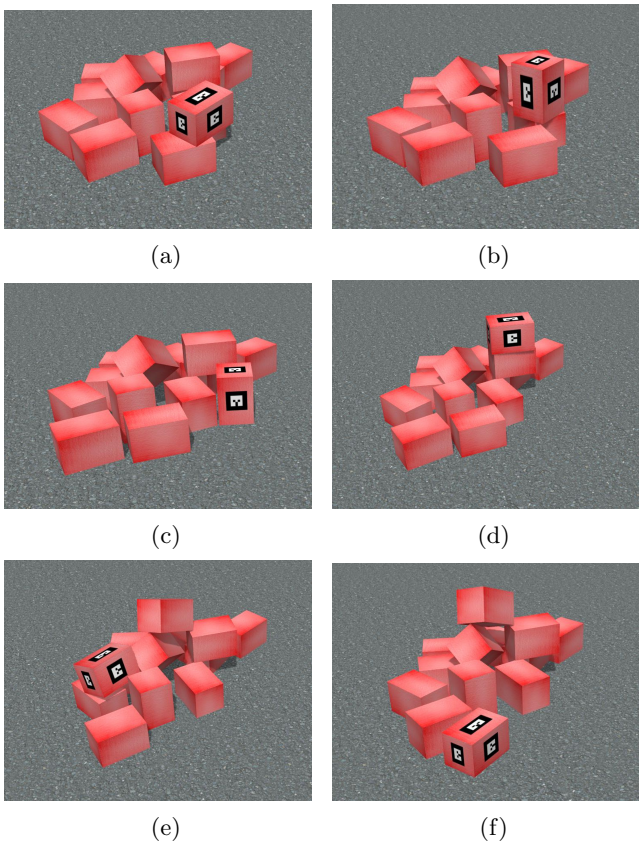


Figure 6: Randomly oriented brick piles generated in a simulator where the target brick's pose is returned using an AR marker.

8 Conclusions

This paper presents a robust multi-stage pipeline for mobile manipulation. The application is a brick picking from an unstructured pile task, where the pose of target brick is given. Our approach considers the full kinematic model of a mobile base and a 6-DOF manipulator. A key aspect of our approach is the design of a scoring function that provides the best configuration of this redundant system. Our 5-stage pipeline includes: 1) collision-aware planning with scoring system for the full mobile manipulator, 2) base-only motion execution, 3) obstacle map and brick pose update, 4) collision-free motion execution for manipulator-only and finally, 5) re-planning and collision-aware execution for manipulator-only.

The performance of the proposed approach is evaluated in using realistic simulation with multiple brick piles configurations. The result shows that our algorithm is more robust than the Moveit! framework. Although in some case it takes a longer time, it is guaranteed not to fail.

In future work, we will test our approach with the real mobile manipulator that will be used in the MBZIRC 2020 competition.

	Success		Duration (s)		Manipulability		Avg Planning Time (s)	Planning Success Rate
Pile	5-Stage	Simple	5-Stage	Simple	5-Stage	Simple	5-Stage	5-Stage
A	Yes	Yes	26.595	29.418	0.04	0.03	0.334	1.00
B	Yes	Yes	21.155	19.254	0.08	0.05	0.590	1.00
C	Yes	Yes	26.389	12.019	0.08	0.04	0.504	1.00
D	Yes	No	22.411	-	0.05	-	0.407	1.00
E	Yes	No	29.147	-	0.06	-	0.784	1.00
F	Yes	No	25.526	-	0.06	-	0.693	1.00

Table 2: A summary of our simulation results showing the comparison between the proposed pipeline which has a 100% success rate as opposed to a simplistic approach where the robot drives to a fixed offset from the pile.

References

[Alehdaghi *et al.*, 2015] M. Alehdaghi, M. A. Esfahani, and A. Harati. Parallel ransac: Speeding up plane extraction in rgbd image sequences using gpu. In *2015 5th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 295–300, Oct 2015.

[Beeson and Ames, 2015] Patrick Beeson and Barrett Ames. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In *Proccedings of HUMANOIDS*, 11 2015.

[Berenson *et al.*, 2008] D. Berenson, J. Kuffner, and H. Choset. An optimization approach to planning for mobile manipulation. In *2008 IEEE International Conference on Robotics and Automation*, pages 1187–1192, May 2008.

[Chitta *et al.*, 2012] S. Chitta, I. Sucan, and S. Cousins. Moveit! [ros topics]. *IEEE Robotics Automation Magazine*, 19(1):18–19, March 2012.

[Dong Hun Shin *et al.*, 2003] Dong Hun Shin, B. S. Hamner, S. Singh, and Myung Hwangbo. Motion planning for a mobile manipulator with imprecise locomotion. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 1, pages 847–853 vol.1, Oct 2003.

[Franklin and Akman, 1985] Wm. Randolph Franklin and Varol Akman. Octree data structures and creation by stacking. In Nadia Magnenat-Thalmann and Daniel Thalmann, editors, *Computer-Generated Images*, pages 176–185, Tokyo, 1985. Springer Japan.

[Hamner *et al.*, 2009] Brad Hamner, Seth Koterba, Jane Shi, Reid Simmons, and Sanjiv Singh. An autonomous mobile manipulator for assembly tasks. *Autonomous Robots*, 28(1):131, Sep 2009.

[Hartenberg and Denavit, 1955] Richard S Hartenberg and Jacques Denavit. A kinematic notation for lower pair mechanisms based on matrices. *Journal of applied mechanics*, 77(2):215–221, 1955.

[Hornung *et al.*, 2013] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 2013. Software available at `http://octomap.github.com`.

[Khatib *et al.*, 1996] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal. Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*, volume 2, pages 546–553 vol.2, Nov 1996.

[Kros *et al.*, 2006] John F Kros, Mike Lin, and Marvin L Brown. Effects of the neural network s-sigmoid function on kdd in the presence of imprecise data. *Computers & operations research*, 33(11):3136–3149, 2006.

[Kuffner and Lavalle, 2000] James J. Kuffner and Steven M. Lavalle. Rrt-connect: An efficient approach to single-query path planning. In *Proc. IEEE Intl Conf. on Robotics and Automation*, pages 995–1001, 2000.

[Lavalle, 1998] Steven M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998.

[mbz, 2019] MBZIRC challenge. `https://www.mbzirc.com/`, 2019. Accessed: 2019-09-30.

[Yamamoto and Xiaoping Yun, 1994] Y. Yamamoto and Xiaoping Yun. Coordinating locomotion and manipulation of a mobile manipulator. *IEEE Transactions on Automatic Control*, 39(6):1326–1332, June 1994.

[Yoshikawa, 1985] Tsuneo Yoshikawa. Manipulability of robotic mechanisms. *The International Journal of Robotics Research*, 4(2):3–9, 1985.