# Robotic Autonomous Trolley Collection with Progressive Perception and Nonlinear Model Predictive Control

Anxing Xiao[†], Hao Luan[†], Ziqi Zhao[†], Yue Hong, Jieting Zhao, Weinan Chen,
Jiankun Wang[*], *Member, IEEE*, Max Q.-H. Meng[*], *Fellow, IEEE*

*Abstract*— **Autonomous mobile manipulation robots that can collect trolleys are widely used to liberate human resources and fight epidemics. Most prior robotic trolley collection solutions only detect trolleys with 2D poses or are merely based on specific marks and lack the formal design of planning algorithms. In this paper, we present a novel mobile manipulation system with applications in luggage trolley collection. The proposed system integrates a compact hardware design and a progressive perception and planning framework, enabling the system to efficiently and robustly collect trolleys in dynamic and complex environments. For perception, we first develop a 3D trolley detection method that combines object detection and keypoint estimation. Then, a docking process in a short distance is achieved with an accurate point cloud plane detection method and a novel manipulator design. On the planning side, we formulate the robot's motion planning under a nonlinear model predictive control framework with control barrier functions to improve obstacle avoidance capabilities while maintaining the target in the sensors' field of view at close distances. We demonstrate our design and framework by deploying the system on actual trolley collection tasks, and their effectiveness and robustness are experimentally validated. (Video[1])**

## I. INTRODUCTION

Robots have become popular in people's lives because they can complete tedious and complex tasks autonomously or collaboratively. In this paper, we discuss a robotic autonomous trolley collection system designed for trolley collection at airports.

At airports, passengers usually use luggage trolleys to help carry luggage between gates and arrival/departure. For example, Hong Kong International Airport (HKG) has an annual passenger flow of more than 72.9 million passengers and has approximately 13,000 luggage trolleys[1]. Naturally, for the convenience of passengers, collecting and redistributing these trolleys has become a vital but laborious task. Most airports, including HKG, still require considerable human

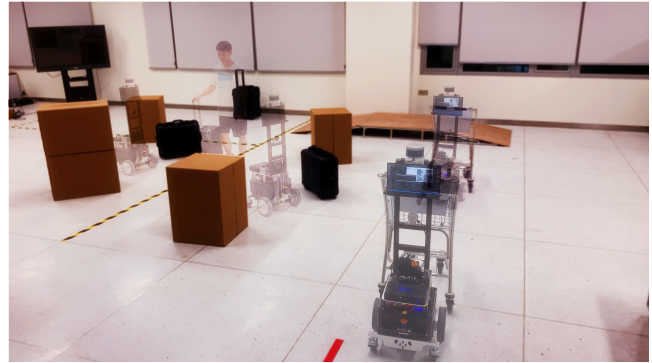[1]Video demonstration: https://youtu.be/6SwjgGvRtno.



Fig. 1: An autonomous robot detecting a trolley, safely navigating itself among people and obstacles, and collecting the trolley and transporting it to a designated location.

resources to collect trolleys and return them to designated locations for continued use. However, the labor costs incurred by such human-driven operations are huge and continue to grow, especially in developed regions. Therefore, the robotic autonomous trolley collection system provides a promising and cost-effective solution for tedious and expensive tasks.

In addition, with the outbreak of COVID-19, large international airports such as HKG have become high-risk areas for the spread of the virus. Airport staff working with trolleys are at risk of infection because the coronavirus can last for several days, even on inanimate objects including trolleys. Therefore, without human contact and intervention, robotic autonomous trolley collection will be another effort to break the chain of virus transmission as the pandemic escalates. In this paper, we focus on developing a robotic autonomous trolley collection system that integrates mechanical design, perception and planning, with the ability to navigate and reliably collect trolleys in complex environments.

### A. Related Work

An autonomous mobile manipulation robot that can find and manipulate objects, as shown in Fig. 1, is an engineering challenge featuring sophisticated incorporation of multiple modules, including mechanical design, perception, planning, and control. The design of mobile manipulation platforms has been an active research area. A few researchers take the approach of equipping mobile robots with robot arms, represented by the DLR-HIT-Hand[2], PR2[3] and TIAGo[4]. However, these platforms are designed for a universal purpose using their sophisticated manipulators, so they lack reliability when performing repetitive tasks, e.g., collecting trolleys at airports. The first robotic trolley collection solution

is introduced in [5]. The developed prototype has several sensors and a fork manipulator to lift a trolley. Nonetheless, the lifting process is open-loop since there is no feedback sensor in the manipulator. In our new design, we add sensors to introduce feedback detecting sudden impacts encountered by the manipulator.

For visual perception, learning-based 2D object detection such as Fast R-CNN[6] and YoloV5[7] has been well investigated in recent years. These real-time object detection models endow mobile robots with the ability to localize specific targets in complex environments. In [5], the authors use a trained R-CNN model to detect a trolley, and the robot moves towards the trolley while maintaining a bounding box of the target in the middle of the image. For accurate manipulation tasks, however, merely perceiving 2D information of the target is not enough. The method in [8] relies on markers pasted on the trolley despite its exploration in monocular 3D trolley detection. In autonomous driving, many researchers attempt to fully explore the potential of RGB images for 3D detection by recovering 3D objects from key points[9], [10]. A key drawback of such methods is that when the target is too close to the camera, the limitation of the camera's field of view will cause failure in detection. In our method, to address the above shortcomings, we incorporate 2D detection and key point detection to estimate the 3D pose of a trolley at a long distance and leverage LiDARs to detect the backplane of the trolley at a short distance.

For planning and control, there are some previous efforts at the trolley collection task. In [11], [5], the adopted method is visual servoing. The main disadvantage of such method is that it does not consider obstacle avoidance and safety, leaving another critical task on the to-do list upon field deployment. The work of [12], [1] focuses on task assignment and smooth-path generation for multiple robots with nonholonomic constraints, but safety is not a general consideration in the planning framework and the final docking error is not taken into account. Recently, optimization-based planning strategies such as model predictive control (MPC) have gained their prevalence in mobile robot planning and control[13]. There are also breakthroughs on tackling real-time safety guarantees for MPC with control barrier functions (CBFs)[14], [15], [16]. CBFs are useful tools for integrating safety considerations as constraints into the general MPC framework. In our work, we formulate our motion planning problem under an MPC framework with obstacle avoidance constraints and field-of-view constraints, and it is validated more efficient and robust than the state of the arts concerning robotic autonomous trolley collection.

### B. Contributions

This work offers the following contributions:

1) A novel robotic autonomous trolley collection system integrating a mechanical system and an efficient autonomy framework.
2) A progressive perception strategy involving long-distance keypoints-based monocular 3D detection and short-distance accurate pose estimation using LiDARs.

3) A safety-critical motion planner formulation under a nonlinear model predictive control framework with CBFs considering obstacle avoidance and field-of-view constraints.
4) Experimental demonstration in complex and dynamic environments of our system detecting target trolleys and safely collecting the trolleys.

## II. SYSTEM DESIGN

The robots for collecting luggage trolleys in the airport need to replace trolley collectors to complete many tasks. Most of these tasks are very complex and challenging for a robotic system, so we specially designed a highly integrated hardware and software system suitable for trolley collection tasks in crowded environments.
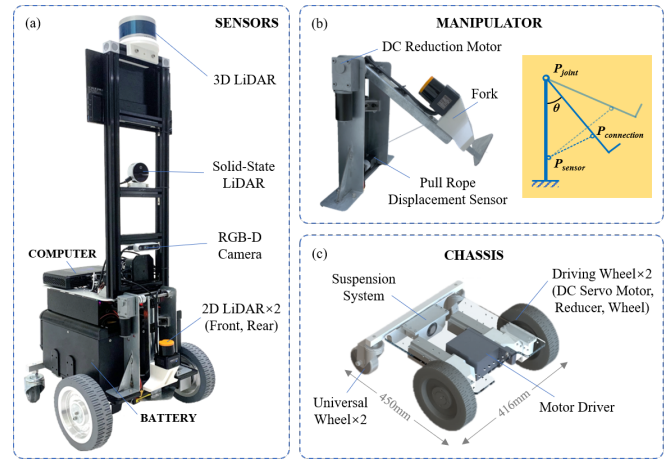
### A. Mechanical System



Fig. 2: The robot consists of three main functional modules, namely, the chassis module (Fig. 2(c)), the sensors module (Fig. 2(a)), and the manipulator module (Fig. 2(b)). In addition, the robot is equipped with a high-performance computer and a large-capacity battery to support stable operation (see Fig. 2(a)).

The developed robot for luggage trolley collection, shown in Fig. 2, is 1.2m high with 0.07m ground clearance, 0.45m long, and 0.416m wide. As an integrated robotic system with mobile operation functions, the performance of movement, loading, and operation must be considered in the design.

*1) Chassis:* The two front wheels are driving wheels, shown in Fig. 2(c). Each driving wheel comprises a DC servo motor, a reducer, and a wheel, producing 31.75N.m of torque. The rear wheels are two universal wheels connected to the car body by a suspension system.

*2) Sensors:* For perception and localization, the robot is equipped with a 3D LiDAR, two 2D LiDARs, a solid-state LiDAR, and an RGB-D camera, as shown in Fig. 2(a). Due to adequate battery performance and sufficient onboard computing power, further extension of sensors and other equipment can be installed as required.

*3) Manipulator:* The manipulator is particularly designed for catching a luggage trolley in airports, shown in Fig. 2(b). It is installed at the front of our robot and consists of a support base, a fork, a draw-wire encoder , and a DC reducer

motor. The motor lifts the fork in a rotating manner around a pivot, and the draw-wire encoder serves as a feedback source for calculating the position of the fork based on the length of the wire.

The length variation $\Delta l$ of the wire reflects the speed and state of the fork movement. The length of the wire $l$ can be used to judge whether the fork is close to the designated position, and $\Delta l$ can be used to judge whether the fork is blocked or has grasped the trolley. Hence, by periodically detecting $l$ from the draw-wire encoder and through differential calculation, we can construct a feedback control system for the manipulator.
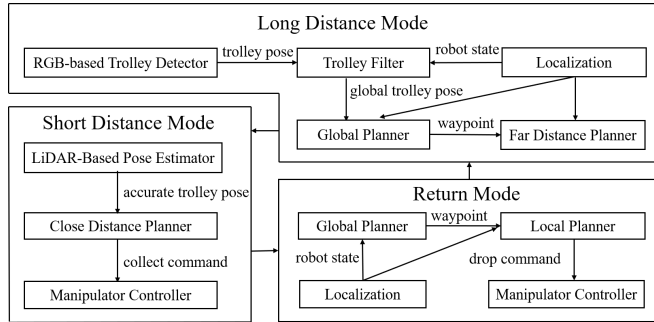
### B. Autonomy Framework



Fig. 3: Autonomy framework overview.

Fig. 3 illustrates our navigation and collection autonomy. We propose a hierarchical framework to break the robotic autonomous trolley collecting process into three stages.
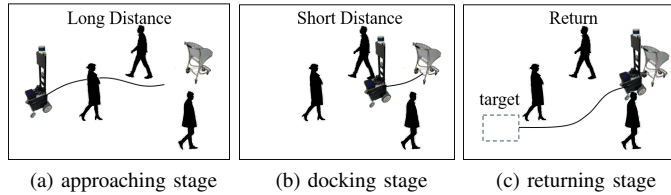


Fig. 4: Illustration of the three stages of our framework. (a) At the approaching stage, the robot detects the trolley at long distances and navigates safely in crowded environments. (b) At the docking stage, the robot catches the trolley with fine motions. (c) At the returning stage, the robot transports the trolley to a returning spot.

In the approaching stage shown in Fig. 4(a), the robot moves in a crowded environment and finally gets to the back of a target trolley and shares the same orientation as the trolley. An RGB-D camera is used at this stage to perceive the trolley's position and orientation at a fairly long distance. The planner generates a motion trajectory to approach the trolley while avoiding obstacles in the dynamic environment. In the docking stage (see Fig. 4(b)), the robot continues to move towards an ideal docking position precisely and then catches the trolley with its manipulator. At this stage, the robot should accurately get to the docking position so that the final aligning error can be small enough for a successful catch. When the robot arrives at the exact docking location, the planner will give the low-level manipulator controller an action command to perform the final catch.

After successfully capturing the target trolley, the robot carries the trolley to a designated returning spot, as Fig. 4(c) shows. During all stages, an occupancy grid map is built with the Gmapping package[17], and the AMCL[18] localization is utilized to estimate the robot's states in the world frame.

## III. PROGRESSIVE PERCEPTION AND PLANNING STRATEGY

In this section, we characterize our trolley collection strategy as a two-stage process based on the distance between the trolley and the robot. The collection task is simplified to a planar model since we assume that the trolley and the robot are in the same plane. At long distances, we utilize a monocular camera to detect the trolley and roughly estimate its three dimensional pose $\mathbf{q}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}, \theta_{\text{tar}}]^T$, wherein $\mathbf{p}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^T$ denotes the trolley's position and $\theta_{\text{tar}}$ represents its orientation. At short distances, the trolley's pose is precisely estimated by using a LiDAR.

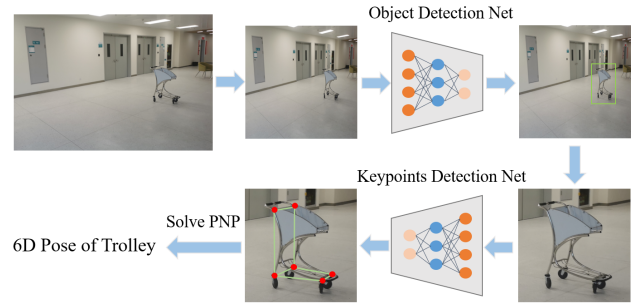### A. Monocular 3D Trolley Detection at Long Distance



Fig. 5: In the monocular 3D pose detection, a source image is first downsampled and put into an object detection net, then the detected trolley is cropped out from the original image and put into a keypoint detection net, and eventually the detected keypoints are used to solve a PnP problem.

The monocular 3D detection framework consists of three parts as shown in Fig. 5. First, we use an object detection network to find a 2D bounding box of the target trolley from a downsampled RGB image $I_s \in \mathbb{R}^{W_s \times H_s \times 3}$. Then, from the original high-resolution image $I \in \mathbb{R}^{W \times H \times 3}$, we crop the trolley image with the bounding box and get a new image $I_c \in \mathbb{R}^{W_c \times H_c \times 3}$. Second, instead of predicting the 3D pose of the trolley directly, we use a deep neural network to predict six 2D keypoints (red points in Fig. 5) in the cropped image $I_c$. Finally, with the *a priori* known 3D coordinates of the 2D keypoints, we can calculate the corresponding 6D pose by minimizing the reprojection error of the keypoints in the original image $I$.

Specifically, in the first part, we choose YOLOV5[7] as our network model for real-time trolley detection due to its efficiency in object detection. In the second part, inspired by the human pose estimation[19], we adopt the stacked hourglass network structure to estimate the heatmaps of six 2D keypoints $\hat{p}_i^c = [\hat{x}_i^c, \hat{y}_i^c]^T$ for $i = 0, 1, \ldots, 5$, in the cropped image $I_c$. Then we can get the corresponding homogeneous 2D keypoints $\hat{p}_i = [\hat{u}_i, \hat{v}_i, 1]^T$ for $i = 0, 1, \ldots, 5$, in image

coordinates of the original image $I$. In the third part, we solve a perspective-n-point (PnP) problem[19] to obtain the pose of the trolley. According to the perspective projection model for cameras, we have the following relationship

$$s_i p_i = K X_i^c = K \left[ R \mid T \right] X_i^t, \quad i = 0, 1, \ldots, M \quad (1)$$

where $X_i^c = [x_i^c, y_i^c, z_i^c, 1]^T$ and $X_i^t = [x_i^t, y_i^t, z_i^t, 1]^T$ represent the homogeneous 3D coordinates of the keypoints in the camera's frame and the trolley's frame, respectively; $K$ is the intrinsic camera matrix that projects $X_i^t$ to the image point $p_i = [u_i, v_i, 1]^T$ in homogeneous image coordinates; $s_i$ is a scale factor. Then, we solve the PnP problem to get a 3D rotation $R$ and a 3D translation $T$ from the trolley's frame to the camera's frame by adopting the EPnP algorithm[20]. Eventually, with localization information of the robot base and the relative pose of the trolley in the camera frame, we can calculate the global state of the trolley $\mathbf{q}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}, \theta_{\text{tar}}]^T$. Moreover, a filter is performed to avoid sudden changes in detection results since the trolley should be static most of the time. If the trolley is not in the camera's field of view (FoV), the state of the trolley is set to be the same as the last time when it was within the FoV.

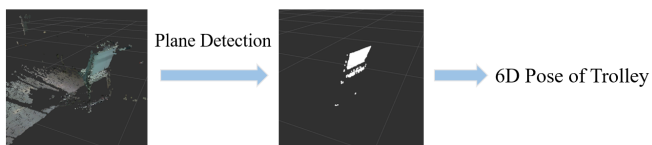### B. LiDAR-Based Pose Estimation in Short Distance



Fig. 6: Accurate trolley pose estimation enabled by plane detection using a solid-state LiDAR.

During docking, accurate perception of the trolley's pose is vitally crucial. Fig. 6 illustrates our perception strategy at this stage. We use a solid-state LiDAR to yield a point cloud, and then perform plane detection and fitting with those points of the cloud to estimate the precise pose of the trolley. The obtained point cloud is noted by $P = \{p_1, p_2, \ldots, p_n\}$ where $p_i \in \mathbb{R}^3$ for $i = 1, \ldots, n$, with $n$ being the total number of points. To get an ideal point cloud characterizing the backplane of a trolley, the robot should be at a suitable pose. Empirically, we set the robot facing the backplane close behind the trolley (0.3m~2m). Since the trolley's pose obtained at the approaching stage is with a decimeter-level precision, equipped with a good enough motion planner, which we will show in Section III-C, our LiDAR's FoV is large enough to ensure the backplane will be presented entirely in the point cloud.

To estimate the trolley's pose $\mathbf{q}_{\text{tar}}$ at a centimeter-level precision, we filter out interference points by setting a threshold. After that, we conduct plane fitting through the RANSAC algorithm[21] and get a set of plane points $P_{\text{plane}} = \{p_1, p_2, \ldots, p_M\} \subseteq P$ and 4 parameters $a$, $b$, $c$, and $d$ in the plane equation $aX + bY + cZ + d = 0$. With the plane parameters and the plane points, we may estimate the center point and the yaw angle of the back plane by calculating the center point of the filtered point cloud and the normal

vector of the plane. After obtaining the trolley's 3D pose $\mathbf{q}_{\text{tar}}$, we can then figure out a manipulation pose for the robot to collect the trolley.

### C. Safety-Critical Motion Planning with FOV Constraints

This planning part considers two main problems, videlicet, generating a feasible state and control trajectory, and avoiding unsafe actions in crowded environments. In both stages of the collection process, the robot needs to move to a given goal state $\mathbf{x}_{\text{goal}} = [x_{\text{goal}}, y_{\text{goal}}, \theta_{\text{goal}}]^T$. Concretely, the goal state at the approaching stage is at a position behind the trolley and an orientation same as the trolley, while at the docking stage, the goal state is an ideal pose for the robot to operate manipulator.

In this work, we characterize the safe set $\mathcal{C}$ of states as the zero-superlevel set of a continuously differentiable function $h : \mathcal{X} \subseteq \mathbb{R}^3 \to \mathbb{R}$

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\}. \quad (2)$$

Safety, in our case, has a physical meaning of avoiding all static and dynamic obstacles. To do so, we can keep the distance between the robot and any obstacles beyond a specific range. Therefore, it is natural to define the following function to construct our safe set $\mathcal{C}$

$$h_{\text{ob}}(\mathbf{x}) = (x - x_{\text{ob}})^2 + (y - y_{\text{ob}})^2 - d_{\text{safe}}^2 \quad (3)$$

where $\mathbf{x}_{\text{ob}} = [x_{\text{ob}}, y_{\text{ob}}]^T$ denotes the position of any obstacle and $d_{\text{safe}}$ a predefined safety distance.
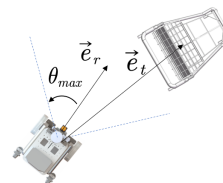


Fig. 7: Illustration of the maintain-field-of-view requirement.

At the docking stage, it is preferable to let the trolley remain in the FoV of the solid-state LiDAR. As is shown in Fig. 7, $\vec{\mathbf{e}}_t$ is a unit vector starting from the robot and pointing at the trolley; $\vec{\mathbf{e}}_r$ represents a unit vector in the direction of the robot's orientation; $\theta_{\text{max}}$ is the maximal angle between these two vectors at which the trolley stays within the LiDAR's FoV. To meet this requirement of maintaining observation, we define the following function that we will use later in our planning formulation:

$$h_{\text{view}}(\mathbf{x}) = \vec{\mathbf{e}}_t \cdot \vec{\mathbf{e}}_r - \cos \theta_{\text{max}}. \quad (4)$$

Then, we introduce CBF constraints [14], [15]

$$\Delta h(\mathbf{x}_k, \mathbf{u}_k) + \lambda_k h(\mathbf{x}_k) \geq 0, \quad (5)$$

where $\Delta h(\mathbf{x}_k, \mathbf{u}_k) := h(\mathbf{x}_{k+1}) - h(\mathbf{x}_k)$ with $\lambda_k \in (0, 1]$. This kind of constraints ensures $h$ becomes a discrete-time CBF, which means the safe set $\mathcal{C}$ defined in (2) is invariant along the trajectories of a discrete-time dynamic system. Also, one can find that (5) guarantees the lower bound of $h$ decreases exponentially at time $k$ with the rate $1 - \lambda_k$.

We formulate the planning task as a nonlinear model predictive control (NMPC) problem. At the approaching stage, the formulation has the following form:

$$\min_{\{\mathbf{x}_k, \mathbf{u}_k\}} \ \|\mathbf{x}_N - \mathbf{x}_{\text{goal}}\|_{P_f}^2 + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|_{Q_u}^2 \tag{6a}$$

$$\text{s.t. } \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{6b}$$

$$\mathbf{x}_0 = \mathbf{x}_{\text{init}} \tag{6c}$$

$$\mathbf{x}_k \in \mathcal{X}, \mathbf{u}_k \in \mathcal{U} \tag{6d}$$

$$\Delta h_{\text{ob}}^i(\mathbf{x}_k, \mathbf{u}_k) + \lambda_k h_{\text{ob}}^i(\mathbf{x}_k) \geq 0 \tag{6e}$$

where $\|\mathbf{x}\|_A := \sqrt{\frac{1}{2}\mathbf{x}^T A \mathbf{x}}$, and the two positive definite matrices $P_f$ and $Q_u$ are respectively coefficients measuring terminal costs and running control costs. (6a) minimizes the quadratic cost function over a horizon of $N$ steps. In (6b), we use the differential-drive model as the robot's system model. (6d) constrains the states and control inputs in a reachable state set and an admissible control set, respectively. Constraint (6e) is for obstacle avoidance.

At the docking stage, the formulation is similar:

$$\min_{\{\mathbf{x}_k, \mathbf{u}_k, \delta_k\}} \ \|\mathbf{x}_N - \mathbf{x}_{\text{goal}}\|_{P_f}^2 + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|_{Q_u}^2 + w\delta_k^2 \tag{7a}$$

$$\text{s.t. } \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{7b}$$

$$\mathbf{x}_0 = \mathbf{x}_{\text{init}} \tag{7c}$$

$$\mathbf{x}_k \in \mathcal{X}, \mathbf{u}_k \in \mathcal{U} \tag{7d}$$

$$\Delta h_{\text{ob}}^i(\mathbf{x}_k, \mathbf{u}_k) + \lambda_k h_{\text{ob}}^i(\mathbf{x}_k) \geq 0 \tag{7e}$$

$$\Delta h_{\text{view}}(\mathbf{x}_k, \mathbf{u}_k) + \mu_k h_{\text{view}}(\mathbf{x}_k) \geq \delta_k \tag{7f}$$

At this stage, we describe the states at which the trolley remains in the robot's FoV as a safe set defined by joining (2) and (4). Similar to the obstacle avoidance constraint (7e), the maintaining observation requirement is formulated as the constraint (7f). To avoid infeasibility, we introduce a slack variable $\delta$ and minimize it by the cost term $w\delta^2$.

Upon implementation, these optimization problems are formulated in CasADi[22] and solved with IPOPT[23].

## IV. IMPLEMENTATION AND EXPERIMENTS

### A. Perception System Evaluation

*1) Data Set:* To ensure the robustness of the detection task at long distances, we built our own data set for network training. For object detection, we collected 1,200 pictures of the trolleys with different illumination, backgrounds, angles, etc. These pictures were all labeled with 2D bounding boxes around the trolleys. The data set was divided into three parts, namely, 800 for training, 200 for validation, and 200 for testing. For key points detection, we prepared 800 pictures of the cropped trolleys images with accurate key point labels. Similarly, we arranged 600 images for training, 100 for validation, and another 100 for testing.

*2) Implementation Details:* We implemented and trained our monocular 3D trolley detection networks offline using PyTorch on an Intel machine with an i7-9750H CPU and an NVIDIA GTX 1660Ti GPU. Our 2D detection network is adapted from the official code releases of YOLOV5 [7]. Training this network on our own data set, we adopted the SGD optimizer[24] for 300 epochs in total with a batch size of 16. The base learning rate was 0.01, and we reduced it to 0.001 from the 150th epoch and to 0.0001 from the 200th epoch. The training stage of the object detection net lasted for roughly 14 hours. For key points detection, we used a stacked hourglass network with PyTorch upon implementation. To improve the generalization ability of our model, we leveraged data augmentation techniques such as random scaling, cropping, flipping, and color transformation. During training, we run the Adam[25] optimizer with a base learning rate of 0.0001 for the first 200 epochs, and reduced it at a decreasing rate of 0.95 every 10 epochs later on. Finally, it took about 4 hours to train our key points detection network with a batch size of 8. At short distances, our method based on plane detection does not involve learning, so we implemented it with the Point Cloud Library[26].
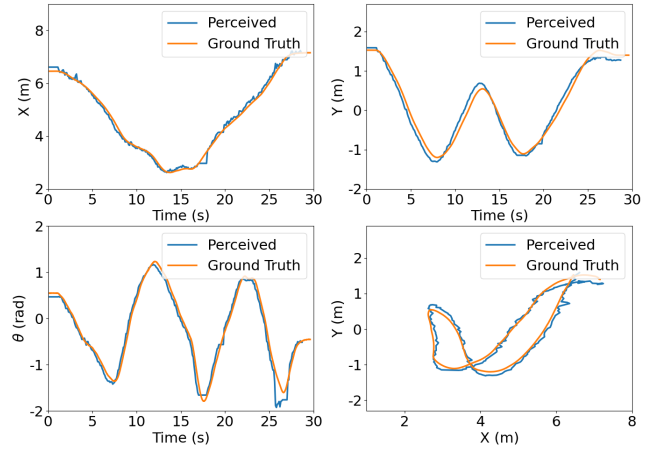


Fig. 8: Comparisons between the ground truth pose of a moving trolley and results of our 3D monocular method. These four subfigures represent the $x$ coordinate, $y$ coordinate, orientation $\theta$, and position trajectory respectively.

*3) Perception Results Evaluation:* To verify the effectiveness of our perception strategy, we conducted experiments moving a trolley in irregular motions and comparing the 3D poses detected by the robot with ground truths measured by a motion capture system. First, we tested our 3D monocular method used in long-distance perception, and we show comparisons between the perceived poses of a moving trolley and the ground truth in Fig. 8. The average estimate error in position is 0.17m with a variance of 0.0097m$^2$, and the average estimated angle error is 0.11rad with a variance of 0.0085rad$^2$. Then the proposed short-distance LiDAR method based on plane detection is also validated, and the results are presented in Fig. 10. The average estimate error in position is 0.03m with a variance of 0.0002m$^2$ and the average estimate error in orientation is 0.02rad with

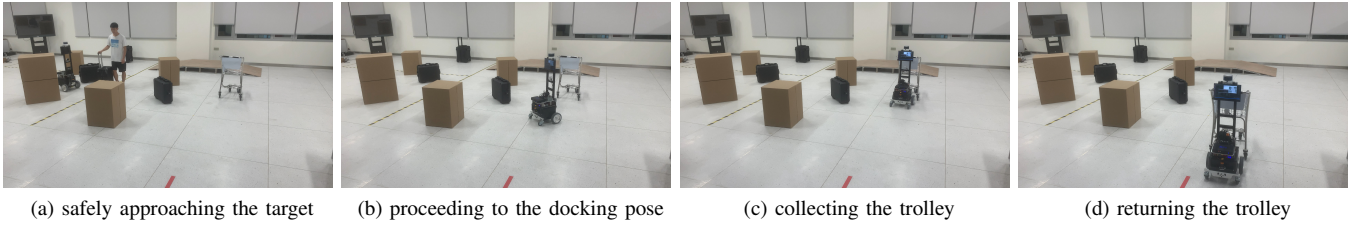| (a) safely approaching the target | (b) proceeding to the docking pose | (c) collecting the trolley | (d) returning the trolley |

Fig. 9: Snapshots of demonstration of our system conducting an actual trolley collection task. (a) When the robot was approaching the trolley, a human with a suitcase moved across the robot's route right in front of it. (b) The robot then slowed down, adjusted its route to avoid the human and other obstacles, and arrived at the goal position for docking. (c) The robot reached the exact manipulation pose based on its own perception and planning in real time. (d) After a successful capture, the robot carried the trolley to the returning spot.
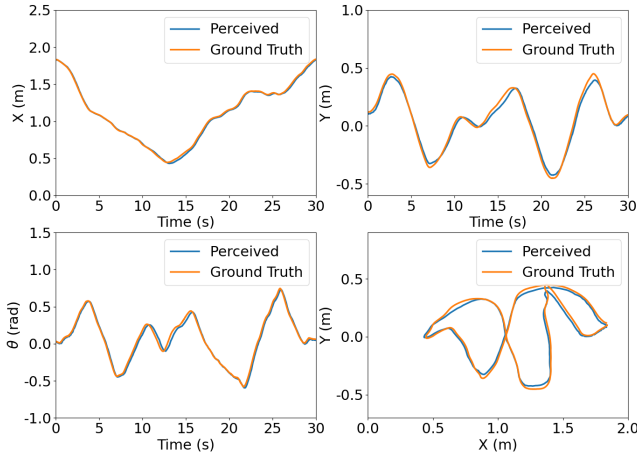


Fig. 10: Comparisons between the ground truth pose of a moving trolley and results of our LiDAR-based plane detection method.

a variance of $0.00036\text{rad}^2$. In all, the perception module can provide accurate information for further planning and manipulation.

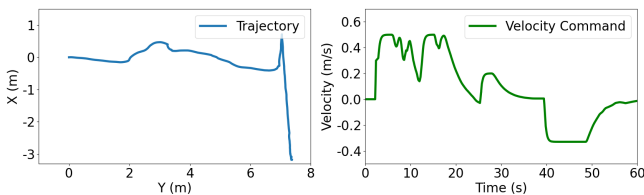### B. Autonomous Trolley Collection Demonstration



Fig. 11: The position trajectory of the robot and the velocity commands yielded by its motion planner.

Using the approaches described throughout this paper, we demonstrate our system in an actual autonomous trolley collection task. The robot is supposed to detect and localize a target trolley, safely navigate itself to the trolley's back, catch the trolley with its manipulator, and finally carry it to a designated returning spot. Our hardware setup is shown in Fig. 2, and all the algorithms above were integrated with the Robot Operating System (ROS) environment and run in real time on the robot's onboard computer with an i7-1165G7 CPU and an NVIDIA GTX 2060 GPU. In the demonstration shown in Fig. 9, we put the target trolley at different locations with different orientations, far from several initial locations of the robot, and let the robot perform

the collection autonomously. In the space between the robot and the target trolley, we set up multiple static obstacles to block the robot's direct route to the goal. Fig. 11 shows the position trajectory of the robot in the demonstration and velocity commands produced by our planner over time. In the velocity commands plot in Fig. 11, the first big crest happens between $t = 10$s and $t = 20$s is caused by avoiding the moving human in Fig. 9(a); the second crest at $t = 27$s means the robot has passed the approaching stage and begins docking (see Fig. 9(c)); and the sudden change at $t = 39$s indicates the start of the return stage shown in Fig. 9(d).

## V. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a mobile manipulation system for robotic autonomous trolley collection in complex and dynamic environments. To detect target trolleys and estimate their poses, the robot uses a learning-based 3D detection method involving object and key points detection at long distances, and adopts an accurate point cloud plane detection method at short distances. For safe motion planning and control, we model this real-time task as an NMPC problem. With CBFs, the obstacle avoidance and field-of-view maintaining requirements are composed into the planning framework as constraints. The incorporation of the novel design of mechanical system and autonomy framework together with the progressive perception and planning strategy forms an efficient and robust robotic solution to autonomous trolley collection. We demonstrate our system in hardware on an actual trolley collection task with static obstacles and moving humans. Experimental results reveal that our solution clearly outperforms most state of the arts regarding the collection task. Our future work will focus on developing global decision-making strategies and multi-robot collaboration.

## REFERENCES

[1] J. Wang and M. Q.-H. Meng, "Real-time decision making and path planning for robotic autonomous luggage trolley collection at airports," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–10, 2021.

[2] H. Liu, P. Meusel, G. Hirzinger, M. Jin, Y. Liu, and Z. Xie, "The modular multisensory DLR-HIT-Hand: Hardware and software architecture," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 4, pp. 461–469, 2008.

[3] J. Bohren, R. B. Rusu, E. G. Jones, E. Marder-Eppstein, C. Pantofaru, M. Wise, L. Mösenlechner, W. Meeussen, and S. Holzer, "Towards autonomous robotic butlers: Lessons learned with the PR2," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5568–5575.

[4] J. Pages, L. Marchionni, and F. Ferro, "TIAGo: The modular robot that adapts to different research needs," in *International workshop on robot modularity, IROS*, 2016.

[5] C. Wang, X. Mai, D. Ho, T. Liu, C. Li, J. Pan, and M. Q.-H. Meng, "Coarse-to-fine visual object catching strategy applied in autonomous airport baggage trolley collection," *IEEE Sensors Journal*, vol. 21, no. 10, pp. 11 844–11 857, May 2021.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[7] G. Jocher, A. Stoken, J. Borovec, NanoCode012, A. Chaurasia, TaoXie, L. Changyu, A. V, Laughing, tkianai, yxNONG, A. Hogan, lorenzomammana, AlexWang1900, J. Hajek, L. Diaconu, Marc, Y. Kwon, oleg, wanghaoyang0106, Y. Defretin, A. Lohia, ml5ah, B. Milanko, B. Fineran, D. Khromov, D. Yiwei, Doug, Durgesh, and F. Ingham, "ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations," Apr. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.4679653

[8] J. Lin, H. Ma, J. Cheng, P. Xu, and M. Q.-H. Meng, "A monocular target pose estimation system based on an infrared camera," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019, pp. 1750–1755.

[9] P. Li, H. Zhao, P. Liu, and F. Cao, "RTM3D: Real-time monocular 3D detection from object keypoints for autonomous driving," in *European Conference on Computer Vision*. Cham: Springer, 2020, pp. 644–660.

[10] T. He and S. Soatto, "Mono3D++: Monocular 3D vehicle detection with two-scale 3D hypotheses and task priors," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8409–8416.

[11] J. Pan, X. Mai, C. Wang, Z. Min, J. Wang, H. Cheng, T. Li, E. Lyu, L. Liu, and M. Q.-H. Meng, "A searching space constrained partial to full registration approach with applications in airport trolley deployment robot," *IEEE Sensors Journal*, vol. 21, no. 10, pp. 11 946–11 960, May 2021.

[12] J. Wang and M. Q.-H. Meng, "Path planning for nonholonomic multiple mobile robot system with applications to robotic autonomous luggage trolley collection at airports," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2020, pp. 2726–2733.

[13] S. Yu, Y. Guo, L. Meng, T. Qu, and H. Chen, "MPC for path following problems of wheeled mobile robots," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 247–252, 2018.

[14] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *2021 American Control Conference (ACC)*, May 2021, pp. 3882–3889.

[15] J. Zeng, Z. Li, and K. Sreenath, "Enhancing feasibility and safety of nonlinear model predictive control with discrete-time control barrier functions," *arXiv:2105.10596*, May 2021.

[16] S. He, J. Zeng, B. Zhang, and K. Sreenath, "Rule-based safety-critical control design using control barrier functions with application to autonomous lane change," in *2021 American Control Conference (ACC)*, 2021.

[17] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[18] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, no. 343-349, pp. 2–2, 1999.

[19] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 483–499.

[20] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *International Journal of Computer Vision*, vol. 81, no. 2, p. 155, 2009.

[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, Jun. 1981.

[22] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[23] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.

[24] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Heidelberg: Physica-Verlag HD, 2010, pp. 177–186.

[25] D. P. Kingma and J. Ba, "ADAM: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[26] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1–4.