ECE 599 Final Report: Autonomous Capabilities for Spot in Manufacturing Environments

Yung-Ching Sun
May 10, 2025

***Abstract*—Boston Dynamics' Spot robot [1] has been widely adopted in research and various applications, such as search and rescue operations in hazardous terrain or inspection tasks within manufacturing environments. Recently, a 6-DOF robotic arm was added to Spot, further enhancing its capabilities for more advanced tasks. This project is a research collaboration with Nestlé Purina, aiming to explore the potential and safety of the Spot arm for executing inspection tasks in a manufacturing plant. In this report, I will focus on the integration of computer vision techniques for autonomous bucket and valve manipulation. Specifically, it investigates how Spot leverages its onboard cameras to detect the location and orientation of target objects for autonomous manipulation, and evaluates the reliability of this methods. Furthermore, this report outlines key research directions and open challenges for future work.**

I. Introduction

A. Motivation

Advancements in mobile robotics have become increasingly valuable in industrial applications, particularly for inspection tasks in manufacturing environments and industrial plants. Autonomous mobile robots offer numerous benefits, including automating routine inspections [2, 3], accessing hazardous environments which are dangerous for humans [4], utilizing various sensors to detect anomalies [5, 6], or systemically recording and analyzing inspection data [3]. Boston Dynamics' Spot robot [1] has been widely adopted for inspection and monitoring tasks due to its exceptional mobility and multi-sensing capabilities. With the recent integration of a 6-DOF robotic arm, Spot is now capable of performing complex manipulation tasks, allowing it to interact with and manipulate objects in its environment, rather than merely observing them.

This project, collaborating with Nestlé Purina, aims to explore the potential of Spot with an arm's autonomous inspection and manipulation capabilities in manufacturing environments. One potential application under investigation is a feeder test, which involves comparing the actual and intended flow rate of liquids. Automating this task with Spot and its arm would require the robot to identify and locate the target bucket in the environment, pick up the bucket and put it to the specified position, rotate the ball valve to release liquid and wait for a specified duration, then close the valve, and move the bucket to a scale for weighing. To reliably achieve these goals, several key capabilities of the robot require further investigation and evaluation.

B. Background

Mobile robots are widely deployed for autonomously routine inspections, equipment monitoring, anomaly detection, and data collection and analysis to support maintenance. Equipped with various sensors, including optical cameras, thermal cameras, pan-tilt-zoom (PTZ) cameras, acoustic sensors, laser scanners [3], these robots provide comprehensive inspection capabilities. For instance, [6] demonstrated the use of mobile robotic system equipped with various sensors, including LiDAR, stereo, UV/IR/RGB cameras, and an electronic nose for autonomous industrial plant inspection, allowing autonomous navigation to identify anomalies. [7] outlined the ability of mobile robots to traverse difficult environments during power substation inspections, using sensors and computer vision techniques for identifying equipment status.

Boston Dynamics' Spot robot has become the industry's preferred mobile robot for autonomous inspection tasks [8] due to its exceptional mobility, user-friendly design, and a variety of sensors. For example, National Grid has utilized Spot to autonomously navigate areas with high electrical field and detect hot spots and oil leak on the equipment [4], Michelin has employed Spot to detect air leaks and thermal issues with motors [9], and bp sent Spot to inspect hard-to-reach areas for inspection tasks like gauge reading and noise anomaly detection to enhance industrial safety [10].

Despite the promising advantages, several challenges remain in applying mobile robots for industrial inspections. One key challenge is the complexity and unstructured nature of industrial and manufacturing environments. For example, in environments with equipments that obstruct computer vision systems would be difficult for autonomous navigation [7]. Furthermore, in cluttered and dynamic settings with humans or complex machinery, ensuring safety is a significant concern [11]. Safe autonomous navigation in environments and safe operation of robotic arms are essential for preventing accidents in manufacturing environments. Another challenge is the

robustness of the detection and manipulation capabilities of mobile robots [7].

Moreover, most autonomous inspection tasks performed by mobile robots rely primarily on sensors for observation. However, many inspection and maintenance tasks in manufacturing plants require physical interaction. For example, checking for loose fasteners on assembly lines [12], performing feeder tests in automated production systems, operating a handheld partial discharge (PD) sensor in substations to measure acoustic and transient earth voltage data for inspections [13], or autonomously tripping and racking circuit breaker [2].

These delicate manipulation tasks demand fine motor skills and precise control of the robotic manipulator. Nevertheless, the reliability, capability, and limitations of quadruped robot with robotic arms in performing these kind of delicate tasks has not been fully explored. Further investigation into the integration of precise manipulation for physically interactive inspection tasks is essential for advancing real-world deployment.

C. Contribution

This project utilized Boston Dynamics' Spot robot [1] to consider the feeder test as an study case to explore the reliability, limitations, and opportunities of quadruped robots with manipulators for autonomous inspection tasks in manufacturing environments. This report investigates the feasibility of using Spot's onboard cameras to identify target objects in the environment without relying on AprilTags. Additionally, it proposes a method for performing valve manipulation using Spot's vision system and robotic arm. Finally, this report outlines potential directions for future work and discusses possible solutions to the remaining challenges.

D. Overview

This report is structured as follows: Section II includes the approaches and result discussions for autonomous bucket manipulation without using Apriltags. Section III presents the approach for estimating the ball valve rotation angle. Section IV gives a brief summary of my current work, and Section V proposes the future directions and potential solutions.

II. Bucket Manipulation without Apriltags

In the previous work [14], the detection of the bucket position relied on an AprilTag attached to the bucket. However, in real-world applications, the placement of the AprilTag might not always be in Spot's field of view. If the AprilTag is attached to the side of the bucket that Spot cannot see, the bucket cannot be detected, leading to failure of the manipulation tasks. Moreover, eliminating the reliance on Apriltags for detection would enhance the flexibility of autonomous manipulation tasks, allowing it to be extended to unmarked objects and broader applications in the future. This section proposes solutions for Spot to identify the target bucket in the environments without relying on AprilTags, and evaluate the reliability of the proposed method.

A. Problem Statement

Given a quadruped robot equipped with 360-degree RGB cameras and integrated with computer vision techniques, whether the robot can reliably identify and autonomously approach the target object in a manufacturing-like environment.

B. System Overview

The workflow for autonomous bucket manipulation without Apriltags is shown in Fig. 1. RGB videos captured from Spot's onboard cameras will serve as the input for a custom bucket detection model. The detection model will then detect the bucket in the scene and calculate the center pixel position of its bounding box. Using the camera models provided in the Spot SDK, we can calculate the bucket's position in the world relative to Spot based on its center pixel position and used this position to command Spot to walk to the front of the bucket. In addition, to simplify the task, I made the following assumption.

Assumption 1. *We assume that the target bucket is always the same type of orange bucket, and there is only one target bucket in the environment.*

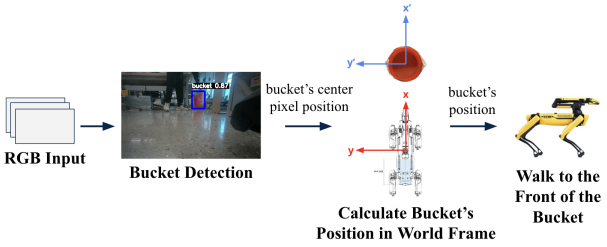

Fig. 1. The workflow of bucket manipulation without using Apriltags.

C. Bucket Detection with a Pretrained Model

I adopted Ultralytics YOLO11 [15] model for bucket detection. YOLO11 is the latest model of the YOLO series. Since I could not find any publicly available datasets specifically for bucket detection, and considering the fact that this task might be expanded to various objects in the future, I initially chose to use yolo11n, a light weight object detection model pretrained on the COCO dataset [16], for detecting the bucket. The COCO dataset includes 80 common object classes and 118287 training images, making yolo11n a strong model for general object detection. I assumed that leveraging this pretrained model would provide a robust detection framework for

our goal while allowing a greater flexibility for the future expansion to other objects. However, one of the biggest challenges is that there is no "bucket" class in the COCO dataset. After some testing, I found that yolo11n model tends to identify the bucket as "cup". Therefore, I temporarily treated "cup" as the detection class for "bucket" during the initial test.

Leveraging the yolo11n pretrained model and OpenCV functions in Python, I developed Algorithm 1 to detect the orange bucket in images. First of all, I use the yolo11n model to detect objects in the RGB image (line 1) and extract the bounding box region of the "cup" class as the Region of Interest (ROI) (line 4-6). Next, I aim to develop a stable method for detecting the bucket's color. I first apply the Contrast Limited Adaptive Histogram Equalization (CLAHE) method on L-channel in LAB color space to improve image contrast and reduce noise interference (line 8). This approach helps mitigate the impact of uneven lighting and noise on color, while preserving color authenticity, as the processing primarily targets the Light channel.

Then, the ROI is converted to the HSV color space, and *cv2.InRange()* is applied to create a binary mask for the orange region, using the lower bound or_{lb} and upper bound or_{ub} for the orange color in the HSV space (line 10). However, I observed that text and plots in different color on the bucket or noise might affect the result. To address this, I applied a morphological operation with *cv2.MorphologyEx()* to refine the noise or small unwanted regions, enhancing the target orange bucket area and making the mask cleaner (line 11). Finally, the ratio of the orange mask to the full ROI was calculated to check if this bucket is in orange or not (line 13-17), where the ratio threshold for orange object is $\epsilon_{or} = 0.7$. The orange bucket detection results is shown in Fig. 2.

Fig. 2. Detection results using a yolo11n pretrained model with color check.

D. Bucket Detection with Custom Detection Models

The method described in the previous subsection (section II-C) performed well when tested on images taken with a smartphone. However, when I tested on Spot, the detection performance was unstable. Several factors could have contributed to this issue: (1) using the "cup" class to represent the "bucket" is not an effective approach, if other orange object similar to a cup is in the scene, it may be incorrectly identified as an orange bucket; (2) the cluttered background in the lab environment may interfere with the yolo11n model, which was trained on the COCO dataset and is too general, making it easy to be distracted by surrounding objects. Given these concerns, I decided to collect images of the bucket used in our experiments within the lab environment to train a custom detection model during spring break.

Algorithm 1 Orange Bucket Detection Using yolo11n

Require: RGB image img;

1: $detections$ = **DetectionModel**(img);
2: **for** $d_i \in detections$ **do**
3: $bbox$, cls = bounding box coordinate and class of d_i;
4: **if** cls = "cup" **then**
5: $ROI \leftarrow$ Subregion of img from coordinates of corresponding bounding box;
6: **end if**
7: ROI_{lab} = **ConvertColorToLAB**(ROI);
8: $ROI_{enhanced}$ = **ApplyCLAHE**(ROI_{lab});
9: ROI_{hsv} = **ConvertColorToHSV**($ROI_{enhanced}$);
10: $MASK_{or}$ = **InRange**(ROI_{hsv}, or_{lb}, or_{ub});
11: $MASK_{refined}$ = **MorphologyEx**($MASK_{or}$);
12: r_{or} = **AREA**($MASK_{or}$)/**AREA**(ROI);
13: **if** $r_{or} > \epsilon_{or}$ **then**
14: $color$ = "orange";
15: **else**
16: $color$ = "not orange";
17: **end if**
18: $result \leftarrow img$ with detected bucket's bounding box and color label;
19: **end for**
20: **return** $result$

To train a custom bucket detection model with good performance, I tried different datasets and using transfer learning with a pretrained yolo11n model. Firstly, I utilized my phone to capture 85 images of bucket under various lighting conditions, distances, angles, and locations within the lab to ensure that the detection model would perform robustly in the lab environment. The images were then labeled using *labelImg* and the dataset were split into training data (70%), validation data (15%), and testing data (15%). After the preparations, the models were trained with the configurations shown in Appendices VI-A Table III. To elaborate, I wanted to make the model more robust in various conditions, so I employed these hyperparameters to optimize the detection model's performance: *scale*, which randomly scale the size of the images during training to help the model detect objects at different scales; *degrees*, which randomly rotate the images to help model recognize objects from various

orientations; *hsv_s* and *hsv_v*, which randomly adjust saturation and brightness value of images in HSV space to improve the model's robustness in different lighting conditions and exposure levels [15].

Fig. 3 shows the detection results of the first version of bucket detection models, it is able to detect the target bucket in various environments. I then tested the detection model with Spot; however, as shown in Fig. 4, the model still struggled to detect the bucket stably when using Spot's onboard cameras. This issue may be due to the significant differences in image quality and color saturation between images captured by a smartphone and those taken by Spot's cameras.

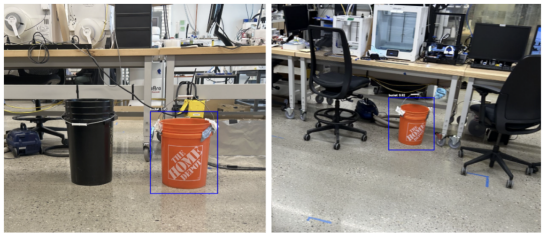

Fig. 3. Bucket detection results with the detection model trained on images captured by a smartphone.

Fig. 4. Bucket detection results using Spot's onboard cameras with the detection model trained on images captured by a smartphone.

To address this, a straightforward solution is to use images captured by Spot's cameras for training. I wrote a script, *capture_rgb.py*, to automatically capture images at regular intervals using Spot's onboard cameras. With this script, I collected 213 images of bucket from Spot's different cameras under various conditions and prepared them as the second dataset for training. The dataset preparation, training method, and configurations remained the same as described earlier.

E. Spot Walk to Bucket

We now get a reliable bucket detector, the next step is to integrate Spot's vision system with the bucket detector, calculate the bucket's position in the world frame through transformations, and enable Spot autonomously navigate to the front of the bucket. Fig. 1 shows the flowchart of this task, the implementation can be found in Algorithm 2. In addition, to make the task simpler in this step, I made this assumption.

Assumption 2. *One target bucket is directly in front of Spot and within its field of view.*

When the task begins, an RGB image is captured by one of Spot's cameras as input. (In this part, I assume the bucket is in front of Spot, so I use the front-left camera to capture the image). The bucket detection model then detect the position of the target bucket in the image. According to Assumption 1, there should be only one target bucket in the scene. Therefore, the detection results are checked to ensure that only one bucket will be the target bucket (line 2-6). The pixel coordinates of the center of the target bucket's bounding box are stored as **p**. Next, using the camera model and the frame transformations T_s provided in the Spot SDK, the bucket's position in the world frame **w** can be calculated from the its pixel location **p** in the image. To be more specific, the bucket's position in the pixel frame will first be transformed into camera frame using the camera intrinsic matrix K_{cam}. Then, it will be transformed from the camera frame to the body frame, and finally to the world frame using the relationships between body frame, vision frame, and odometry frame [17]. Lastly, using **w** along with a predefined offset distance from the target bucket, a walk request is sent to Spot, making it to move and stop in front of the bucket.

Algorithm 2 Spot Walk to Bucket Without Apriltags

Require: RGB image img;
1: $detections$ = **DetectionModel**(img);
2: **if** $detections.size()$ = 1 **then**
3: **p** ← pixel coordinates of the detected bucket's bounding box center;
4: **else**
5: **p** ← pixel coordinates of the bounding box center of the detected bucket with the highest confidence;
6: **end if**
7: **w** = **FindObjectWorldPosition(p,** T_s**,** K_{cam}**)**;
8: **WalkRequest(w,** $offset$**)**;

F. Results

Using the detection model trained on Spot's own images, Spot was able to detect the bucket reliably in the lab. A demo of real-time bucket detection using Spot's vision can be found at this link: *Demo Video*.

Table I and Fig. 5 show the performance of bucket detection models trained on different datasets. In the result comparison, I also trained a model using a combination of images from both Spot and a smartphone. Table I shows that the detection models trained on all three datasets achieved high Precision (P), recall rate (R), mean average precision at 50% IoU (mAP50), and mean average precision across IoU thresholds from 50% to 95% (mAP50-95). This is because the model was

trained and tested on single-class datasets (bucket), and the images in the datasets were captured in the same environment. However, this training approach is only robust for detecting the same type of bucket in similar environments, it will not stably detect the bucket under different conditions from the dataset. This can also be observed in Fig. 5, which I will discuss later. Since this experiment aims to demonstrate the feasibility of removing Apriltags and using Spot's vision for autonomous tasks, this limitation is not a current concern.

Fig. 5 shows that the detection model trained with Spot's images achieve better detection performance when testing on images from Spot. There is a significant difference in image quality and color saturation between images taken with a smartphone and Spot's cameras, as shown in Fig. 3 and Fig. 4. As previously mentioned, since the test images here were captured using Spot, the disparity between Spot's camera and the smartphone camera caused the model trained on smartphone images to struggle with detecting buckets.

TABLE I
PERFORMANCE OF CUSTOM BUCKET DETECTION MODELS

Dataset	P	R	mAP50	mAP50-95
85 images from an iPhone 14	0.988	1.000	0.995	0.881
213 images from Spot's body cameras	0.998	1.000	0.995	0.842
298 images from an iPhone 14 and Spot	0.977	0.993	0.994	0.853

This *demo video* shows that Spot is able to use its own vision to detect and walk to the bucket autonomously. However, the current approach assumes that the bucket is within Spot's field of view. In Algorithm 2, the detection relies on a single captured image to determine the bucket's position before moving to it. This approach might not work if the bucket is outside the field of view. To extend this task to actively searching for buckets within the environment in the future, this method will need to be replaced using real-time detection and tracking approach.

III. VALVE MANIPULATION WITHOUT APRILTAGS

Previously, the ball valve manipulation task required an AprilTag attached on the valve handle to determine its position and rotation angle, with Spot starting from a predefined location. This section propose a solution to enable the Spot operate the valve without relying on AprilTags on the valve handle and can start from varying initial positions.

A. Problem Statement

Given a quadruped robot equipped with a robotic arm, where the robot's body is mounted with 360-degree RGB cameras and the gripper is equipped with an additional RGB camera. The objective is to enable the robot to autonomously estimate a ball valve status and operate it using its vision system and arm. This section aims to evaluate the reliability and limitations of Spot using its perception system to estimate the valve's rotation angle and plan corresponding arm motions for valve manipulation.

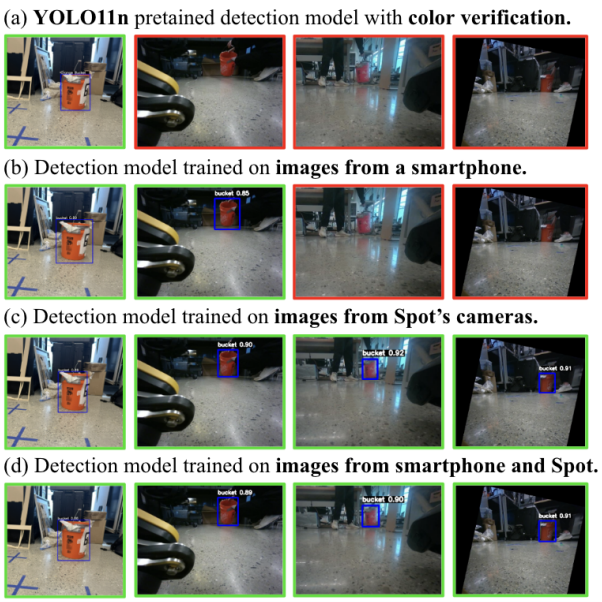

Fig. 5. Detection results using custom models trained on different datasets with same training configurations. The model trained with images from Spot achieved the most stable performance on all images. Green boxes indicate successful detections with high confidence scores (≥ 0.8), while red boxes represent failed detections or low confidence scores (< 0.8).

B. System Overview

Fig. 6 shows the overall workflow for Spot to approach and operate the valve without relying on AprilTags on the valve handle for orientation estimation.

According to Hanna's previous work [14], Spot's initial position would significantly affects the success rate of valve operation. Moreover, I found that the cameras on Spot's body cannot see the valve at its default height. Therefore, at this stage, an AprilTag is placed below the valve to help Spot localize the valve position more accurately (Fig. 6a). Investigating the feasibility and robustness of using a pre-built map and Spot's navigation system to locate the valve could be a direction for future research. Next, since we will estimate the valve orientation in a 2D plane, the camera viewing angle can impact the estimation accuracy. Therefore, we should ensure that the camera is perpendicular to the valve's rotation axis. To achieve this, we use the gripper camera to detect the valve position and move the gripper

so that the camera aligns with the valve's center of rotation (Fig. 6bc). Once aligned, the valve orientation is estimated using oriented bounding box (OBB) detection (Fig. 6d). Finally, Spot operates the valve based on the estimated orientation (Fig. 6e).

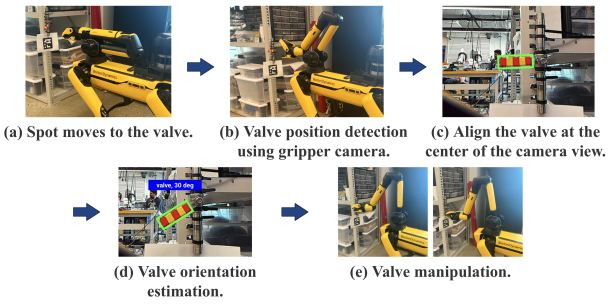

Fig. 6. Overall workflow of autonomous ball valve manipulation.

C. Valve Orientation Estimation

To operate the valve with varying orientations, Spot must estimate the valve handle orientation with its vision system. Since the valve rotates within a 2D plane, estimating its 2D rotation angle is sufficient. To achieve 2D object orientation estimation, I apply Oriented Bounding Box (OBB) detection.

Unlike the general object detection model used in section II, which only detects the axis-aligned bounding boxes of objects, OBB detection is able to identify the objects along with their rotation. In standard YOLO11 object detection, bounding boxes are defined using the format *xywh*, where *x* and *y* are the center coordinates of the bounding box, and *w* and *h* are the width and height aligned with the image axes. As a result, the detected bounding boxes do not reflect the object's orientation.

In contrast, Oriented Bounding Box (OBB) detection uses different label formats that includes orientation information. As illustrated in Fig. 7, OBB detection provides both object oriented bounding boxes, allowing us to estimate their rotation angles. By integrating OBB detection with Spot's vision system, Spot can accurately detect both the valve's position and its orientation.

I adopted YOLO11n-OBB model [15] to train a custom OBB detection model for valve detection and valve orientation estimation due to its robustness and lightweight. To collect training data, I used the script *capture_rgb.py* to automatically capture images of the valve in various orientations and from different viewing angles using Spot's onboard cameras. However, the labeling application I previously used for bucket detection datasets does not support oriented bounding box labeling. Therefore, I found another labeling tool, *X-AnyLabeling* [18], to annotate the valve in the images.

While labeling the valve images captured by Spot's cameras, I noticed that the original dark blue ball valve

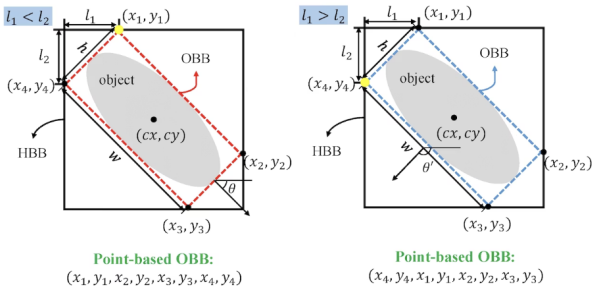

Fig. 7. Examples of label formats for YOLO11 Oriented Bounding Box (OBB) detection [15].

was not visually distinctive in the scene, as shown in Fig. 8a. Using these images would not result in a well-performing valve OBB detection model. To address this issue, I initially wrapped the valve with yellow tape, as shown in Fig. 8b; however, it is still not obvious enough. Therefore, I replaced it with yellow-and-red striped tape to enhance the valve's visibility while avoiding confusion with the nearby red bar, as shown in Fig. 8c. After adjusting the valve color, the dataset was recollected and trained using YOLO11n-OBB model [15].

Fig. 8. Valve in different color.

A similar issue observed in the bucket detection task also arises in the valve detection case. The RGB camera on the gripper differs significantly in quality from Spot's body cameras. As a result, when the OBB detection model is trained using images captured from Spot's body cameras, it struggles to reliably detect the valve in images taken by the gripper camera. As shown in Fig. 10, the model fails to detect the valve in the gripper camera's image. Therefore, I collected the dataset with the gripper camera to train the valve OBB detection model.

Once the valve's oriented bounding box is detected, its angle can be calculated using the *cv2.meanAreaRect()* function, which returns the smallest enclosing rectangle of the input OBB points. The output includes the center coordinates, width, height, and the angle of rotation relative to the horizontal axis. Note that in *cv2.meanAreaRect()*, the width represents the first edge encountered when rotating counterclockwise from the horizontal axis. Since I designed the system such that the valve is at 0 degrees when the handle is aligned with the tube (valve open) and at 90 degrees

when closed, I applied a conditional statement based on the width and height to adjust the output angle from *cv2.meanAreaRect()* to match the desired setup.

D. Results

Fig. 9 shows the OBB detection and valve rotation angle estimation using a model trained on images captured by the gripper camera. The test images were also collected using the gripper camera. The results demonstrate that the custom OBB detection model can reliably detect the valve's oriented bounding box and estimate its rotation angle. Fig. 10 and Table II present the performance of the valve OBB detection models trained on different datasets. Similar to the bucket detection task, differences in resolution and color saturation between cameras affect detection performance. Therefore, reliable detection requires training datasets captured with the same camera used during the detection task.

Fig. 9. Valve OBB detection and rotation angle estimation results.

(a) OBB detection model trained on images **from body cameras.**

(b) OBB detection model trained on images **from gripper camera.**

(c) OBB detection model trained on images **from body and gripper cameras.**

Fig. 10. Valve oriented bounding box (OBB) detection results using models trained on different datasets with same training configurations. Green boxes indicate successful detections with high confidence scores (≥ 0.8), while red boxes represent failed detections or low confidence scores (< 0.8).

IV. Conclusions

This report presents an investigation into the reliability of using Spot's vision system, combined with computer vision techniques, to perform a feeder test in a manufacturing-like environment. Experimental results demonstrate that a custom-trained object detection model (using YOLO11n [15] model in our case) can reliably identify the target bucket. With coordinate transformations, the robot is able to autonomously navigate to the front of the bucket without relying on AprilTags. However, reliable detection requires datasets collected using Spot's onboard cameras under conditions similar to the target environment. Future work could be integrating this bucket detection pipeline with Sammie's novel bucket pick-and-place maneuver, allowing Spot to carry the bucket on its back and place it beneath the valve for subsequent operations.

TABLE II
Performance of Valve Oriented Bounding Box (OBB) Detection Models

Dataset	P	R	mAP50	mAP50-95
273 images from Spot's body cameras	0.998	1.000	0.995	0.914
117 images from gripper camera	0.996	1.000	0.995	0.951
390 images from Spot's body and gripper cameras	0.964	1.000	0.994	0.937

This report also explored a method for estimating the valve's orientation. By modeling valve rotation as occurring in a 2D plane, using oriented bounding box (OBB) detection allows reliable estimation of both the valve's position and its rotation angle. Similar to the bucket detection task, the consistency between the training dataset collecting and deployment cameras is critical for robust performance. Future research could investigate whether incorporating depth information, rather than relying solely on 2D images, may improve the robustness of orientation estimation, as orientation estimation in 2D images is susceptible to projection distortion. Another interesting direction would be removing the AprilTag below the valve and instead leveraging a pre-built map and Spot's navigation system to localize and approach the valve autonomously.

In addition to these future directions, the following section outlines other areas that require further investigation in developing a reliable quadruped robot with a robotic arm for autonomous inspections.

V. Future Work

For a quadruped robot equipped with a robotic arm to reliably perform autonomous inspections and monitoring tasks that require physical interaction, several key capabilities and limitations still require further investigation. Taking the feeder test as an example, there are multiple critical abilities of the robot that need to be further explored:

- Motion planning for valve manipulation.
- Estimating bucket handle orientation.
- Manipulating bucket with varying handle orientations.
- Safe navigation for object search in manufacturing environments.

- Safe navigation in manufacturing environments with moving workers.
- Safe motion planning for Spot's arm.

In this section, I categorize the future work into three main directions: (1) motion planning for valve manipulation, (2) bucket handle orientation estimation, and (3) safe planning for Spot and its arm. I will discuss the key challenges that need to be addressed under each category.

A. Motion Planning for Valve Manipulation

While this work has shown that estimating valve rotation angle within 2D images allows for sufficient reliable estimation result, the robustness of the valve rotation operation still requires further experimental validation. Some interesting directions for future work are incorporating depth information to improve the accuracy and reliability of valve orientation estimation and enhance motion planning for valve manipulation.

To elaborate, during my experiments, I observed that the gripping position and grasp depth would significantly affect the success rate of valve operation. When the gripper grasps the valve closer to its end, the arm can exert a greater rotational effect with less torque due to the principle of leverage; however, this configuration is also more tend to slippage. In contrast, gripping near the valve's rotational center requires more torque, which may lead to failure turning operation.

Furthermore, the depth of the grasp is also a critical factor. A shallow grip might increase the risk of slippage, while an overly deep grip can result in collisions with the wall behind the valve. These findings highlight the importance of refined grasp motion planning strategies. Further investigation in these areas is needed to improve the overall reliability and effectiveness of autonomous valve manipulation.

B. Bucket Handle Orientation Estimation

In previous work, it was assumed that the bucket handle is always pointing upward, allowing Spot to use a predefined arm trajectory to pick up the bucket. To relax this assumption, two main challenges must be addressed: estimating the orientation of the bucket handle and arm motion planning for bucket manipulation with varying handle orientation. This section proposed a solution for estimating the bucket handle orientation.

To simplify the bucket handle orientation estimation task, the following assumption could be made.

Assumption 3. *The handle of the bucket will be on the side visible to Spot. In other words, when Spot detects the bucket and approaches it, it should not fail to find the handle, as it will not be occluded by being on the opposite side of the bucket.*

A potential solution is to use point cloud registration techniques. Specifically, the CAD model of the target bucket can be used to generate ground truth point clouds for both the bucket body and its handle. Assuming that when the handle is pointing upward, the z-axes of the bucket body and the handle are aligned, and their x- and y-axes are parallel. Once the bucket detector identifies the bucket and the robot approaches it, Spot can use its depth cameras to scan the bucket and separately match the scanned point clouds to the ground truth point clouds of the bucket body and handle. This registration process allows us to compute the relative transformation between the bucket body and handle, thereby enabling us to determine the handle's orientation. Once the handle orientation is estimated, Spot can plan a trajectory for the arm to pick up the bucket based on different handle orientation. This step would require the integration of the arm collision avoidance to ensure the safety of the robot arm and the surrounding equipment.

C. Safe Planning for Spot and Its Arm

Safety is a critical concern in industrial environments. In manufacturing plants with equipment and moving humans in a cluttered space, mobile robots must ensure they do not collide with sensitive equipment or workers during inspections and operations. However, Spot's built-in obstacle avoidance system and its moving obstacle detection capabilities still have limitations. A brief overview of Spot's obstacle avoidance system is summarized below. All the following information is referenced from [19, 20, 21].

Spot's Obstacle Avoidance System: Spot's built-in obstacle avoidance system relies on five depth cameras mounted on its body, two at the front, one at the rear, and one on each side of Spot. Each depth camera provides an approximate 90-degree field of view. However, due to the placement of these cameras, there are gaps in the overall coverage, as shown in Fig. 11.

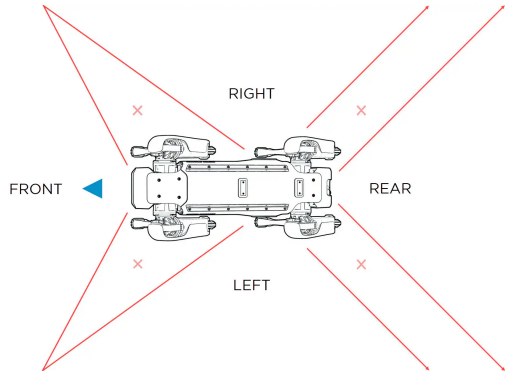

Fig. 11. Field of view of Spot's depth cameras [19].

The built-in obstacle avoidance system maps the Spot's surroundings during movement for obstacle avoid-

ance. During operation, Spot keeps a minimum distance of 7.5 cm between itself and nearby obstacles. However, there are some limitations of the built-in obstacle avoidance system.

- It may fail to detect moving obstacles.
- Obstacles that remain in the gaps in Spot's field of view during its entire path may not be detected.
- The depth cameras' sensing range is approximately 2 meters, and the minimum area and width of obstacles it could detect is 3 cm.
- It cannot detect obstacles directly above Spot.

Spot's Moving Obstacle Avoidance System: Spot, equipped with an Enhanced Autonomy Payload 2 (EAP2), is capable of detecting certain moving objects, such as people and vehicles. Spot's built-in moving object avoidance system can be utilized within an Autowalk mission by incorporating a Spot Crosswalk Action. When a moving object is detected within the defined crosswalk area, Spot will stop and wait for a user-predefined timeout period (with a minimum threshold of 2 seconds) to allow the object to clear. If the moving object leaves within the specified time, Spot will resume its mission. However, if the object remains, Spot can either reroute to find an alternate path without moving objects or start the obstacle avoidance system to navigate through the area, depending on user configuration. In manufacturing and industrial environments, finding another clear path would be a preferred option because it reduces the risk of collisions with nearby equipment or humans.

Although Spot is equipped with built-in static and moving obstacle avoidance systems, several limitations still remain. The following outlines some open challenges and interesting research directions for future exploration.

1) Safe Navigation for Object Search: Currently, the bucket detection and manipulation tasks assume that the target bucket is within Spot's field of view. However, if the bucket or other target object is not visible to Spot, Spot must be capable of autonomously and safely navigating through the environment to search for the target. Ensuring the searching process is efficiently completed without collisions with surrounding obstacles and humans is an essential requirement for deployment in manufacturing and industrial environments.

2) Safe Navigation in Manufacturing Environments with Moving Workers: In manufacturing plants, ensuring that autonomous mobile inspection robots do not collide with human workers is essential for preventing accidents and injuries. Although Spot is equipped with a built-in moving obstacle avoidance system, its current default behavior is relatively inefficient; it simply stops and waits for the moving object to clear before resuming its task. Therefore, one key remaining challenge is enabling Spot to effectively navigate around moving human workers while continuing its path without colliding with surrounding equipment. To achieve this goal, it requires pedestrian motion prediction, real-time re-planning and decision-making, accurate perception of the environment, and the application of conservative safety constraints. Furthermore, if the originally planned path becomes overly crowded, having Spot re-plan an alternative, safer, and efficient route is another option. Defining the constraint for an overcrowded path is critical for balancing operational efficiency while maintaining safety.

3) Spot Arm Safety: Spot's built-in obstacle avoidance system is designed only for its body and does not account for payloads such as the robot arm and EAP2. As a result, Spot arm and EAP2 lack collision avoidance capabilities. When Spot and its arm performs autonomous inspections or manipulations, there is a risk that the arm or EAP2 may unintentionally collide with nearby objects or people, causing damage or injury.

In manufacturing environments that have sensitive, fragile equipment and walking humans, it is crucial that the arm can interact with target objects effectively while avoiding collisions with its surroundings. To enable this capability, several key factors must be achieved:

- Spot's onboard cameras should be able to perceive objects within the arm's reachable space.
- The system should have the precise models of both the robot arm and EAP2, and accurately model the arm's motion.
- It should distinguish between the target objects that are intended for manipulation and those that must not be touched or collided with.
- Safety constraints for arm motion planning that ensure safe execution of manipulation tasks.

In summary, while Spot offers basic obstacle avoidance capabilities for its body, its performance is limited in complex, dynamic, and crowded environments, and it lacks collision avoidance for the arm and payloads. Enhancing safe navigation during object search, enabling real-time re-planning in dynamic environments, and extending collision avoidance capabilities to include the arm and payloads are essential steps toward achieving fully autonomous and safe operation.

For a mobile robot with a manipulator to operate safely within an environment, accurate environmental perception and robust motion planning are critical. Verify Spot's environmental perception and mapping accuracy should be a preliminary step toward developing advanced safe planning capabilities. Achieving these objectives requires the integration of multiple sensors.

Currently, I have been attempting to integrate the system using the Spot ROS 1 driver [22]. However, this has led to several unexpected issues. After consulting with students from another lab, I learned that the Spot

ROS 1 driver is unstable and buggy, and is generally not recommended. Therefore, the next steps could be exploring alternative approaches, such as using the Spot SDK [17] directly, switching to the Spot ROS 2 driver [23], or developing custom ROS publishers to retrieve data from the Spot SDK client library.

ACKNOWLEDGMENT

I would like to thank Professor Dawn Tilbury and Professor Kira Barton for offering me this opportunity and for providing valuable advice and insightful guidance throughout the project. I also appreciate Samantha Staudinger for discussing ideas with me and for collaborating on the ICRA submissions and this project. Thanks to Hanna Chapin for her prior work and instructions, which helped me quickly get up to speed with the project. I would also like to thank Alyssa Carter and Thomas Richards for their guidance and feedback.

REFERENCES

[1] Boston Dynamics, "Spot - The Agile Mobile Robot," 2024. [Online]. Available: https://bostondynamics.com/products/spot/.

[2] E. Robotics, "Automating industrial inspection with autonomous mobile robots," *Robotics Tomorrow*, 2022.

[3] Boston Dynamics, "A new approach to predictive maintenance challenges," 2024. [Online]. Available: https://bostondynamics.com/blog/a-new-approach-to-predictive-maintenance-challenges.

[4] Boston Dynamics, "Spot Becomes Part of the Team at National Grid," 2024. [Online]. Available: https://bostondynamics.com/case-studies/spot-becomes-part-of-the-team-at-national-grid/.

[5] M. Schneier, M. Schneier, and R. Bostelman, "Literature review of mobile robots for manufacturing," 2015.

[6] G. K. Fischer, M. Bergau, D. A. Gómez-Rosal, A. Wachaja, J. Graeter, M. Odenweller, U. Piechottka, F. Höflinger, N. Gosala, N. Wetzel, *et al.*, "Evaluation of a smart mobile robotic system for industrial plant inspection and supervision," *IEEE Sensors Journal*, 2024.

[7] S. Lu, Y. Zhang, and J. Su, "Mobile robot for power substation inspection: A survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 4, pp. 830–847, 2017.

[8] Boston Dynamics, "The New Standard for Industrial Inspection," 2024. [Online]. Available: https://bostondynamics.com/solutions/inspection/.

[9] Boston Dynamics, "Spot at Michelin," 2024. [Online]. Available: https://bostondynamics.com/case-studies/spot-at-michelin/.

[10] Boston Dynamics, "A retrospective on uses of boston dynamics' spot robot," 2024. [Online]. Available: https://bostondynamics.com/blog/retrospective-on-boston-dynamics-spot-robot-uses/.

[11] F. Silveira, R. M. Gago, R. L. De Moura, and G. M. Freitas, "A framework for selecting and implementing ground mobile inspection robots in industrial applications," in *2024 Latin American Robotics Symposium (LARS)*, pp. 1–6, IEEE, 2024.

[12] G. Jing, X. Qin, H. Wang, and C. Deng, "Developments, challenges, and perspectives of railway inspection robots," *Automation in construction*, vol. 138, p. 104242, 2022.

[13] E. Pearson, B. Mirisola, C. Murphy, C. Huang, C. O'Leary, F. Wong, J. Meyerson, L. Bonfim, M. Zecca, N. Spina, *et al.*, "Robust autonomous mobile manipulation for substation inspection," *Journal of Mechanisms and Robotics*, vol. 16, no. 11, p. 115001, 2024.

[14] H. Chapin, "ROB 590 Final Report: Evaluating the Performance of Spot in Automating Manufacturing Flow Rate Testing." University of Michigan, 2024.

[15] G. Jocher and J. Qiu, "Ultralytics YOLO11," 2024. [Online]. Available: https://github.com/ultralytics/ultralytics.

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pp. 740–755, Springer, 2014.

[17] Boston Dynamics, "Spot SDK - Spot 4.1.1 documentation," 2024. [Online]. Available: https://dev.bostondynamics.com/.

[18] W. Wang, "Advanced auto labeling solution with added features," 2023. [Online]. Available: https://github.com/CVHub520/X-AnyLabeling.

[19] Boston Dynamics, "How Spot Avoid Obstacles," 2024. [Online]. Available: https://support.bostondynamics.com/s/article/How-Spot-Avoids-Obstacles-49928.

[20] Boston Dynamics, "Nominal operating environments for spot," 2024. [Online]. Available: https://support.bostondynamics.com/s/article/Nominal-Operating-Environments-for-Spot-49930.

[21] Boston Dynamics, "Moving Object Detection," 2024. [Online]. Available: https://support.bostondynamics.com/s/article/Moving-Object-Detection-72044.

[22] Clearpath Robotics, "Spot ROS User Documentation," 2020. [Online]. Available: https://heuristicus.github.io/spot_ros/html/index.html.

[23] Mobile Autonomous Systems and Cognitive

Robotics Institute, "Spot ROS 2," 2025. [Online]. Available: https://github.com/bdaiinstitute/spot_ros2.

[24] J. Sanchez-Cubillo, J. Del Ser, and J. L. Martin, "Toward fully automated inspection of critical assets supported by autonomous mobile robots, vision sensors, and artificial intelligence," *Sensors*, vol. 24, no. 12, p. 3721, 2024.

VI. Appendices

A. Bucket Detection Models Training Configurations

The custom bucket detection models were trained with the following configurations:

TABLE III
CUSTOM BUCKET DETECTION MODELS TRAINING CONFIGURATIONS

Configurations	Value
device	MPS
model	yolo11n
batch size	16
number of epochs	100
image size	640*640
scale	0.5
degrees	45.0
translate	0.5
hsv_s	0.8
hsv_v	0.8

B. Valve OBB Detection Models Training Configurations

The custom valve oriented bounding box (OBB) detection modesl were trained with the following configurations:

TABLE IV
CUSTOM VALVE OBB DETECTION MODELS TRAINING CONFIGURATIONS

Configurations	Value
device	MPS
model	yolo11n
batch size	16
number of epochs	200
image size	640*640
scale	0.5
degrees	45.0
translate	0.5