Final Report for Carrybox with Ros Vacuum Gripper Plugin

Jiyuan Luo[1] and Siqi Ye[2] and Huiwen Xue[3]

Abstract— **This project explores the development of a robotic system designed to automate box handling in dynamic environments such as warehouses, logistics centers, and manufacturing facilities. The system integrates a ROS-controlled vacuum gripper, leveraging its versatility to securely lift and transport boxes of varying sizes and weights. Central to the system's functionality is the integration of motion planning, facilitated by the MoveIt framework, which enables the robotic arm to calculate optimal paths, avoid obstacles, and execute precise movements in real-time. This motion planning capability is crucial for ensuring safe and efficient operation in unpredictable environments.**

Through the use of simulation tools like Gazebo and advanced motion planning algorithms provided by MoveIt, the project addresses challenges such as real-time obstacle detection, path planning, and collision avoidance. MoveIt allows for the seamless integration of sensor feedback to dynamically adjust the robot's movements based on its surroundings, while Gazebo simulates real-world physics to create a realistic testing environment. These combined tools enhance the system's ability to operate efficiently in complex and constantly changing environments, ensuring precision and stability throughout operations.

The project aims to enhance material-handling efficiency by providing a scalable, modular, and adaptable solution. The ROS framework enables real-time sensor feedback and precise control, ensuring the system can adapt to unpredictable environments. The vacuum gripper is designed for minimal slippage, offering reliable and stable performance during transport. By simulating real-world physics in Gazebo, the system is tested and optimized in a controlled environment, reducing risks and development costs associated with physical trials.

This work builds upon existing research in robotics, combining proven methods with innovative design to achieve a flexible and robust system. Applications of the system include warehouse automation, e-commerce logistics, and industrial workflows, where it promises to reduce operational costs, improve safety, and streamline complex tasks. The outcomes of this project contribute to the broader field of robotics by demonstrating how simulation-driven design, integrated motion planning, and modular systems can be used to develop intelligent, scalable automation solutions tailored to modern industry needs.

I. Introduction

The increasing demand for automation in logistics, manufacturing, and e-commerce has highlighted the limitations of conventional material-handling systems, which often require extensive reconfiguration to adapt to new tasks or environments. With the rise of e-commerce, supply chains are becoming more complex, requiring flexible systems that can handle a wide variety of tasks efficiently and reliably. These trends necessitate the development of advanced robotic systems capable of meeting these challenges with minimal human intervention. Central to this development is motion planning, which enables robots to navigate dynamic environments with precision and adaptability. Motion planning allows robotic systems to calculate optimal paths, avoid collisions, and adapt to changes in real-time, making it a critical component of modern robotic solutions.

This project aims to address these issues by designing a robotic system equipped with a ROS (Robot Operating System)-controlled vacuum gripper. The system is intended to automate the handling and transportation of boxes between designated locations, providing a scalable solution for dynamic environments such as warehouses and logistics centers. The ROS framework facilitates modularity, allowing the system to evolve with changing requirements. By integrating tools such as Gazebo for simulation and MoveIt for motion planning, the project emphasizes real-time adaptability, precise control, and efficient path execution.

Key features of this system include its ability to perform collision avoidance through sensor feedback and execute dynamic path planning to respond to environmental changes. The integration of MoveIt, a motion-planning framework, enhances the system's ability to compute optimal trajectories, ensuring smooth and efficient robotic arm movements in complex environments. MoveIt's ability to plan and visualize the arm's movements ensures that the system can handle a wide range of tasks while maintaining safety and precision. The vacuum gripper itself is designed for versatility, capable of securely lifting boxes of various sizes and weights while minimizing slippage. This adaptability is critical in high-mix, low-volume workflows where conventional systems struggle to maintain throughput and reliability.

Additionally, the project leverages simulation tools to fine-tune system parameters before real-world deployment. Gazebo allows for the creation of realistic environments that replicate real-world physics, enabling thorough testing and optimization. This reduces development time and costs by minimizing trial-and-error in physical settings. Similarly, MoveIt provides an intuitive platform for planning and visualizing robotic arm movements, ensuring precision and safety in task execution.

By focusing on these advanced capabilities, the project not only addresses current inefficiencies in material handling but also contributes to the broader field of robotics and automation. It offers a flexible and intelligent framework for future developments, positioning itself as a meaningful step toward the next generation of smart, autonomous systems.

II. Related Work

In recent years, robotic systems have gained significant attention for automating material-handling tasks, particularly

in industries like logistics, manufacturing, and warehousing. Central to these systems is the use of vacuum grippers, which are valued for their ability to handle a diverse range of objects with varying shapes, sizes, and weights. Research has demonstrated the effectiveness of vacuum grippers in achieving secure and reliable manipulation, minimizing slippage, and adapting to the requirements of fragile or irregularly shaped items. These properties make vacuum grippers a preferred choice in environments demanding flexibility and precision.

The adoption of the Robot Operating System (ROS) has further revolutionized the development of robotic systems by offering a comprehensive framework for modular and scalable design. Studies have shown that ROS facilitates the integration of advanced functionalities such as real-time sensor feedback, dynamic control, and path planning. Motion planning, a key aspect of robotics, enables robots to compute optimal trajectories, avoid obstacles, and adapt to changing conditions in real-time. Projects utilizing ROS have successfully implemented motion-planning algorithms, enabling robotic arms to execute precise trajectory execution while avoiding collisions in complex environments. ROS's open-source nature has also encouraged widespread collaboration and innovation, resulting in a robust ecosystem of tools and plugins tailored to specific tasks.

MoveIt, a ROS-based motion-planning framework, has been widely used in robotic research for tasks requiring precise control and adaptability. Its ability to interface with sensor systems and perform real-time adjustments has proven critical in applications requiring safe and efficient operation in unpredictable environments. MoveIt allows for the calculation of optimal paths based on sensor feedback, and its advanced algorithms enable smooth motion, even in environments with dynamic obstacles. Its compatibility with simulation platforms like Gazebo further enhances its utility by enabling detailed testing and optimization before real-world deployment. Gazebo, with its ability to replicate realistic physics, such as friction, gravity, and collisions, provides an ideal environment for simulating and refining robotic behaviors. Studies combining MoveIt and Gazebo have shown significant improvements in system reliability, as they allow developers to fine-tune parameters and debug issues in a risk-free virtual environment.

Past research has also highlighted the importance of modular design in robotics, particularly for systems deployed in dynamic environments. Modular systems enable scalability and reconfiguration, allowing robotic solutions to evolve with changing operational demands. Motion planning, when coupled with modularity, provides the flexibility needed to adjust to new tasks with minimal reconfiguration. This approach has been successfully applied in various contexts, from warehouse automation to collaborative robotics, where adaptability is crucial for handling diverse tasks.

Building on these advancements, this project integrates the strengths of ROS, MoveIt, and Gazebo to create a system tailored for automated box handling in logistics environments. While existing studies provide a solid foundation for using vacuum grippers and motion-planning frameworks, this project extends these insights by focusing on real-time adaptability and scalability. By addressing specific challenges, such as minimizing slippage, optimizing gripping force, and enabling collision-free motion in dynamic settings, this work contributes to the growing body of knowledge in robotic material handling. Furthermore, it demonstrates how simulation-driven design, combined with motion planning, can enhance the development process, offering a practical approach to deploying intelligent and efficient robotic systems.

III. Method

A. Enhancing Realism in Box Modules for Virtual Worlds

In our initial modification, the basic box module was assigned colors to enhance visual distinctiveness. However, this implementation did not sufficiently emulate the appearance of a table and base. To address this limitation, we revised the specifications documented in the associated Word file. Specifically, the dimensions of the box modules (X, Y, and Z coordinates for length, width, and height) were adjusted to align more closely with the structural characteristics of a table and base. Following these refinements, confirming that the updates contributed to a more realistic and accurate representation of the virtual environment.

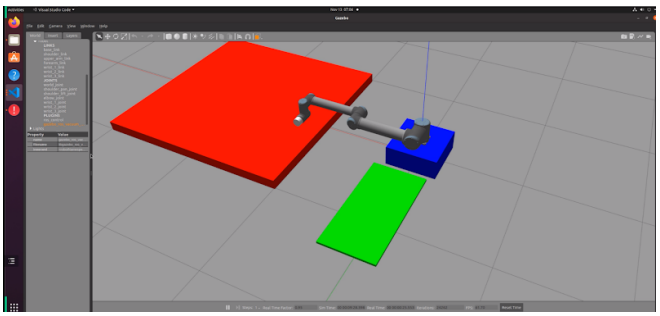

Fig. 1.

B. Development of Custom SDF Box Models for Simulation Automation

To enhance our simulation workflow, we shifted from directly importing pre-built models from gazebo to creating a custom box model by writing our own standalone SDF (Simulation Description Format) file. Using the official Gazebo tutorial as a guide (Gazebo Tutorial), we successfully designed and implemented our own box model. The primary objective of this effort was to enable automated integration of models into the simulated world through Python scripting. This approach aligns with real-world logistics operations, where packaging and handling rely on pipelines that continuously process incoming objects. Simulating such dynamics necessitates a system capable of managing multiple objects in succession, rather than isolating the movement of a single entity. By addressing this stretch goal, we significantly enhanced the realism and scalability of our simulation, achieving a high level of success in replicating real-world packaging workflows.

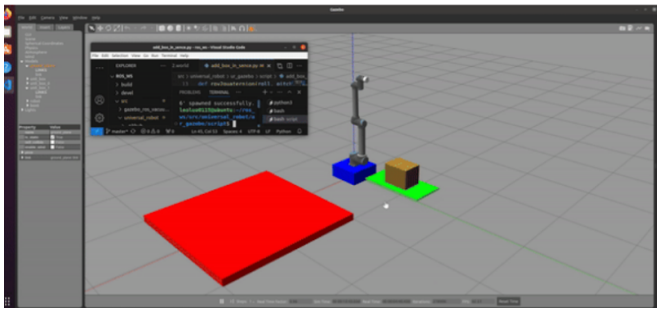

Fig. 2.

C. The Importance of Configuring MoveIt for Robotic Systems

Configuring MoveIt is a critical step in enabling efficient and precise robotic motion planning. This process involves setting up the **base and tip links**, which define the kinematic chain's reference frames, and verifying the **number of joints in the group**, ensuring alignment between the robot's physical model and its kinematic description. These configurations not only establish the structural foundation for the robot's motion, but also ensure compatibility with MoveIt's motion planning algorithms.

As a result of this setup, a tailored MoveIt package is generated, encapsulating the necessary configurations for the robot to function within the MoveIt framework. This new package enables seamless integration with simulation tools like Gazebo and RViz, providing a robust testing and visualization environment. By focusing on the configuration of **base/tip links** and **joint groups**, we establish a precise operational framework, ensuring the robot is ready for scalable and efficient task execution in real-world applications.

D. Practical Approach to Controlling the Vacuum Gripper in ROS

Controlling a vacuum gripper within the Robot Operating System (ROS) involves a series of steps to ensure reliable operation. ROS services, defined using srv files, serve as the foundation for this process. These files specify the request and response structure, creating a standardized interface for communication. A crucial first step is checking the availability of the service with rospy.wait-for-service(), which blocks execution until the service becomes active. This ensures that any service calls made later do not result in errors due to an unavailable service. Once the service is confirmed as available, it is accessed through a rospy.ServiceProxy, allowing for interaction with the gripper. The primary commands to control the gripper are to activate (turn on) or deactivate (turn off) the vacuum. Activating the gripper engages the vacuum mechanism, creating the necessary force to secure an object. Conversely, deactivating it releases the object, completing the manipulation task. To optimize gripper performance, various configuration parameters can be set in the plugin, such as `<maxForce>`, `<maxDistance>`, and `<minDistance>`. These settings are important for tailoring the gripper's operation to specific tasks. For example, the `<maxForce>` parameter defines the maximum force the gripper can exert, preventing damage to objects or overly strong suction that could affect handling. The `<maxDistance>` and `<minDistance>` parameters set the operational range of the gripper, ensuring that it can engage objects at the right distance for efficient operation.

By following these steps, developers can integrate and control vacuum grippers within their robotic systems, enabling precise handling and adaptability in a range of applications. The use of ROS services combined with configurable parameters ensures that the gripper can operate effectively in various scenarios, contributing to more dynamic and responsive robotic systems.

IV. Experiments

A. Troubleshooting Vacuum Gripper Failure in ROS

During the initial phase of our project, we successfully completed the motion planning, ensuring that the robotic arm accurately moved to a position directly above the box. However, despite activating the vacuum gripper service and setting it to "on," we observed that the gripper was unable to lift the box. Initially, we suspected that the issue might have been due to insufficient suction force or incorrect maximum and minimum distance settings. We adjusted these properties multiple times, but the problem persisted, and the arm still failed to lift the box properly. To further investigate, we examined the source code for the vacuum gripper. It became apparent that there was a discrepancy in the configuration of the joint type: the gripper's joint type was set as revolt, while our code was specified as fixed. We realized that using the fixed joint type was problematic. When set to fixed, the simulation merges the joint with the last connected joint, preventing the plugin from recognizing it as a separate joint entity. This led to the failure of the simulation and, consequently, the gripper's inability to function correctly. Upon changing the joint type to revolt, we tested the system again and found that the vacuum gripper was able to lift the box as expected. However, this approach introduced a new issue: during the lifting process, the box would rotate uncontrollably and eventually fall. This highlighted the need for further adjustments to ensure stable handling and prevent unintended rotations during operation.

B. Resolving the Rotation Issue

After conducting thorough research, we discovered two viable solutions to address the issue of the vacuum gripper causing the box to rotate during lifting. The first approach involved increasing the friction coefficient to enhance grip stability. However, we opted for the second solution—integrating a transmission motor into the joint—as it proved more effective in maintaining a stable and secure grip during the lifting process. To implement this, we added a 'transmission' configuration to the robot's URDF file. This configuration included defining a 'SimpleTransmission' type, associating the 'gripper-joint' with a 'PositionJointInterface' for precise control. Additionally, we connected an actuator, the 'vacuum-gripper-motor', specifying a 'mechanicalReduction' value of 1 and the same 'PositionJointInterface' for

```
+    <transmission name="vacuum_gripper_trans">
+        <type>transmission_interface/SimpleTransmission</type>
+        <joint name="gripper_joint">
+
  <hardwareInterface>hardware_interface/PositionJointInterface</hardwareInterface>
+        </joint>
+        <actuator name="vacuum_gripper_motor">
+            <mechanicalReduction>1</mechanicalReduction>
+
  <hardwareInterface>hardware_interface/PositionJointInterface</hardwareInterface>
+        </actuator>
+    </transmission>
```

Fig. 3.

the actuator's hardware interface. This setup ensured that the motor could provide continuous torque and control, allowing the gripper to resist rotational forces and maintain a secure hold on the object. This modification ultimately led to improved stability and precision in handling, ensuring that the box remained steady during movement and manipulation.

C. Challenges and Solutions

Throughout our experiments, we encountered several challenges that impacted the performance and stability of the robotic arm. One significant issue was the discrepancy between the real-time factor in the simulation and real-world time. This led to situations where a planned one-second pause in the simulation would only last 0.2 seconds, disrupting the stability needed for operations such as activating the vacuum gripper. To address this, we increased the RAM allocation for our virtual machine, aiming to align the simulation time as closely as possible with real-world time. This adjustment ensured that waiting for one second in the code corresponded to an actual one-second pause in the simulation, improving the arm's stability during operations. Another challenge we faced was related to control strategies for the robotic arm. Initially, the arm was unable to remain upright and collapsed entirely. After thorough research, we determined that switching from effort control to position control would be the simplest and most effective solution. Effort control requires setting specific forces or torques for each joint, which adds complexity and makes it more difficult to achieve stable motion. In contrast, position control simplifies the task by focusing on joint positions without needing to manage force or torque, resulting in more straightforward and reliable control. This change allowed the robotic arm to achieve a stable posture and perform tasks more effectively. By addressing these challenges and implementing the solutions, we ultimately succeeded in achieving continuous and reliable gripping with the vacuum gripper, as well as effective motion planning. This progress demonstrated the importance of optimizing simulation parameters, refining control strategies, and conducting thorough testing to ensure consistent and successful robotic system performance. The results of our successful implementation can be viewed through the submitted MP4 file or via the YouTube link provided.

V. Conclusion

This project highlighted the importance of careful configuration, thorough testing, and iterative refinement in robotic arm control and simulation. Key takeaways include the necessity of aligning the simulation environment's real-time factor with the real world to ensure reliable performance. By increasing the RAM allocation for our virtual machine, we were able to achieve better synchronization, which in turn improved the stability of the robotic arm's movements and its operations, such as gripping and lifting. This adjustment proved crucial in ensuring that the simulated environment closely mirrored real-world performance, reducing discrepancies that could otherwise disrupt task execution.

Additionally, the choice of control strategy played a significant role in enhancing system performance. The transition from effort control to position control simplified the control loop and enhanced the overall stability of the robotic arm. Position control eliminated the need for manually setting force or torque values for each joint, which significantly streamlined the system's operation and improved its precision. This shift made it easier to handle complex tasks, as the robot's movements became more predictable and easier to fine-tune, ensuring smoother task execution.

Beyond the technical improvements, this project also deepened our understanding of core robotics tools such as ROS, MoveIt, Gazebo, and RViz. Through hands-on experience with these tools, we became more proficient in navigating their functionalities, integrating them effectively for simulation, control, and visualization of robotic tasks. ROS, as the central framework, facilitated modular development and integration of sensor feedback, while MoveIt played a critical role in optimizing the robotic arm's trajectories and ensuring collision-free movement. Gazebo provided the platform for testing and refining the robotic system, while RViz allowed us to visualize the robot's movements in real-time, enabling more effective debugging and optimization.

The integration of these tools provided valuable insights into how different components of a robotic system interact with one another. We learned how to leverage their strengths to create a cohesive system, improving both performance and efficiency. Additionally, the project helped us recognize the importance of simulation-driven design, where testing in a controlled virtual environment can significantly reduce risks and development costs.

Through these lessons, we gained a deeper understanding of how subtle adjustments in simulation parameters, control strategies, and hardware configurations can make a significant difference in the overall functionality of robotic systems. The project demonstrated that rigorous research, trial-and-error, and adapting solutions to real-world constraints are vital for developing robust, reliable, and efficient robotic systems. Furthermore, this experience has equipped us with practical knowledge that will be invaluable in future projects, particularly in the areas of automation, material handling, and robotics. The ability to navigate challenges and iterate on designs is a key skill in the development of intelligent

systems, and this project has reinforced the importance of continuous testing, feedback, and adaptation in achieving optimal system performance.

REFERENCES

[1] Gazebo ROS Vacuum Gripper Debugger GitHub repository. Available: https://github.com/tatsuya-s/gazebo_ros_vacuum_gripper_debugger. Accessed: Nov. 1, 2024.

[2] ROS-Industrial Universal Robot GitHub repository. Available: https://github.com/lihuang3/ur5_ROS-Gazebo. Accessed: Nov. 1, 2024.

[3] Gazebo. (n.d.). *Building a model*. Retrieved from https://classic.gazebosim.org/tutorials?tut=build_model

[4] ROS Wiki. (n.d.). *rospy/Overview/Services*. Retrieved from https://wiki.ros.org/rospy/Overview/Services

[5] YouTube. (2024, November 1). *final demo video*. Retrieved from https://www.youtube.com/watch?v=s4nad_VvNz0