Team SO(n)RT: Autonomous in-place Sorting of Items on Shelves

Abhishek Iyer, Anirudh Shrihari, Keerthi Gaddobanahalli Vijayakumar and Prajwal Gurunath

***Abstract*— The ability to autonomously organize and sort objects in semi-structured environments is a fundamental step toward intelligent robotic automation in logistics, warehousing, and manufacturing. This paper presents an end-to-end robotic sorting system that seamlessly integrates perception, motion planning, and decision-making to efficiently arrange items on a structured shelf. Utilizing a Franka Emika Panda robotic arm and an Intel RealSense D415 camera, our system detects and localizes books through color-based segmentation and in real-world conditions. A minimum-swap sequence optimization approach determines the most efficient sorting order while minimizing redundant movements. The system dynamically updates its understanding of the workspace, adapting to variations in object placement. The proposed methodology demonstrates robust object detection, efficient manipulation, and adaptive decision-making. This work paves the way for scalable and adaptable robotic solutions in industrial automation.**

I. INTRODUCTION

The process of organizing and arranging physical objects within our daily workspaces may appear to be a straightforward task. However, it inherently involves a series of high-level cognitive and motor functions. These include the ability to prioritize the order in which items should be placed, effectively assessing and accommodating spatial constraints to ensure optimal organization, and executing dexterous manipulation to precisely position and handle objects with the required level of control and accuracy.

Replicating human-like manipulation in semi-structured environments remains a significant challenge, requiring the seamless integration of perception, planning, control, and decision-making. This project addresses the problem of automated item sortation in constrained indoor spaces using a Franka Emika robotic arm. The task involves detecting objects on a two-row shelf, generating collision-free motion plans to grasp and relocate items in place, and dynamically updating the system's understanding of the workspace using color-based perception. Through this project, we hope to gain insights into the viability of robotic arms for sortation and the challenges they pose in semi-structured environments like warehouses.

Hence, this project would contribute to advancing robotic automation in logistics, warehousing, and manufacturing by enhancing autonomous sortation in semi-structured environments. Improved perception, planning, and manipulation capabilities can streamline workflows, reduce reliance on manual labor, and increase operational efficiency. The insights gained could inform the development of more adaptable robotic systems for real-world industrial applications.

II. RELATED WORK

A. Perception in cluttered environments

Color thresholding remains a computationally lightweight method for object segmentation in controlled lighting, as demonstrated in warehouse automation systems [2]. However, its limitations in dynamic scenes have spurred hybrid approaches, such as combining thresholding with depth sensing for grasp-point estimation. ArUco [1] markers, widely used for pose estimation and spatial anchoring, offer reliable localization in structured shelves, reducing cumulative errors in repetitive tasks.

Deep learning-based object detection models, such as YOLOv7 [3] and Mask R-CNN [4], have demonstrated robustness in cluttered scenes by leveraging large-scale datasets for feature extraction and segmentation. Depth fusion techniques, combining RGB data with point clouds from stereo cameras or LiDAR, further enhance object discernment in occluded environments.

B. Motion Planning and Obstacle Avoidance

Sampling-based planners like RRT* and PRM* dominate high-DOF manipulator motion planning due to their probabilistic completeness in configuration spaces. Recent work integrates real-time sensor data with incremental planning, enabling dynamic obstacle avoidance without replanning overhead. For the Franka arm, optimization-based controllers have proven effective for trajectory smoothing in tight spaces.

Moveit [5] is a popular manipulation framework that provides various functionalities like motion planning, collision detection, avoidance, etc. It has various motion planners such as: OMPL (Open Motion Planning Library), Pliz Industrial Motion Planner, SBPL (Search-Based Planning Library) etc. We can use these planners for the Franka arm for generating trajectories in tight and confined spaces.

C. Integrated Autonomous Systems

Recent work on enabling integrated autonomy has emphasized the importance of decision-making and the decomposition of high-level goals into actionable sub-tasks. This decomposition is crucial for allowing robots to perform complex tasks in dynamic environments, particularly in industrial applications such as sorting, manipulation, and assembly.

Finite State Machines (FSMs) are based on a set of discrete states, each representing a specific action or

behavior, with predefined transitions between states [7]. These transitions are triggered by certain conditions or inputs, such as sensory feedback. While FSMs are effective for well-defined tasks in controlled environments, as the number of states and transitions grows, especially in dynamic, embodied robotic systems, they face scalability challenges.

To address some of these limitations, Behavior Trees (BTs) have emerged as a more flexible alternative for decision-making in autonomous systems [7]. Unlike FSMs, which treat all states and transitions as equally important, BTs focus on the status of actions—whether they are running, succeeding, or failing. The transitions between these actions are implicit. and dictated by the hierarchy of the tree and the status of the current task. While this provides an effective way to scale, the structure of the tree can become difficult to navigate when the tasks grow in complexity or when unexpected events require drastic changes in behavior.

The rise of Vision-Language Models (VLMs) offers promising solutions to some of these challenges. VLMs combine visual perception and natural language understanding to provide robots with the ability to interpret and reason about high-level, often ambiguous, instructions. By integrating language into the decision-making process, VLMs offer a more flexible approach to task execution that allows robots to better understand human intentions and respond to dynamic, real-world scenarios [8].

III. METHODOLOGY

The proposed methodology enables autonomous book sorting using a Franka Emika Panda robotic manipulator, equipped with an Intel RealSense D415 depth camera for visual perception and spatial localization. The system initializes by positioning the robot at a predefined home configuration, ensuring the entire bookshelf and randomly placed books fall within the camera's field of view (FoV). Object detection and segmentation are performed using OpenCV, where HSV thresholding generates a segmentation mask for isolating books based on their dominant color. Depth data from the RealSense camera is leveraged to compute the 3D world coordinates of each book's centroid.

The sorting task is formulated as a minimum-swap sequence optimization problem, determining the most efficient pick-and-place sequence to achieve the desired arrangement. To establish the "place" coordinates, the shelf is partitioned into six designated slots, with their centroids manually recorded by guiding the robotic arm to each location and storing the corresponding joint configurations. Franka Emika Panda's built-in motion planning framework generates dynamically feasible, collision-free trajectories based on the current configuration space. Each pick-and-place operation is executed via impedance-controlled manipulation, ensuring smooth and adaptive grasping. Upon completion of the sorting process, a final verification step validates the correct placement of all books. This methodology integrates robust perception, optimization-based sorting, and efficient motion planning to enable precise and autonomous book organization. Fig. 2 shows the entire system architecture in detail. Algorithm2 defines the Finite State Machine.

Fig. 1. SO(n)RT setup

A. Hardware Setup

The hardware setup comprises a Franka Emika Panda robotic arm, an Intel RealSense D415 depth camera, a custom-built shelf structure, and a high-performance computing unit. The robotic arm, providing precise 7-DoF (Degrees of Freedom) control, is securely mounted on a workstation, ensuring accurate and stable book manipulation. The RealSense D415 camera is rigidly mounted on the end effector, facilitating both RGB and depth-based visual perception for accurate book detection and localization. The custom shelf structure is designed with five designated slots, each serving as predefined target coordinates for book placement. The system is controlled by a desktop workstation running Ubuntu with ROS, OpenCV, and FrankaPy, enabling real-time perception, motion planning, and execution. Power supply units and safety mechanisms like an emergency stop button is present to ensure reliable operation of the robotic system.

B. Perception Subsystem

The Perception Subsystem begins by initializing the Franka robot arm to 2 pre-defined home positions to capture the entire scene of the shelf (top and bottom), including the randomly placed books. A dictionary is defined to specify the HSV (Hue, Saturation, Value) color boundaries for three target colors—red, green and blue — ensuring accurate segmentation. Each color range is associated with a corresponding BGR value for visualization purposes. The image is converted to the HSV color space, where a binary mask is created based on the defined color boundaries. Morphological operations such as opening and closing are applied to remove noise and enhance the mask's precision.

Subsequently, ROS messages for color and depth images are converted into OpenCV-compatible formats using CvBridge to facilitate further image processing. The segmentation process isolates the regions of interest (ROIs)

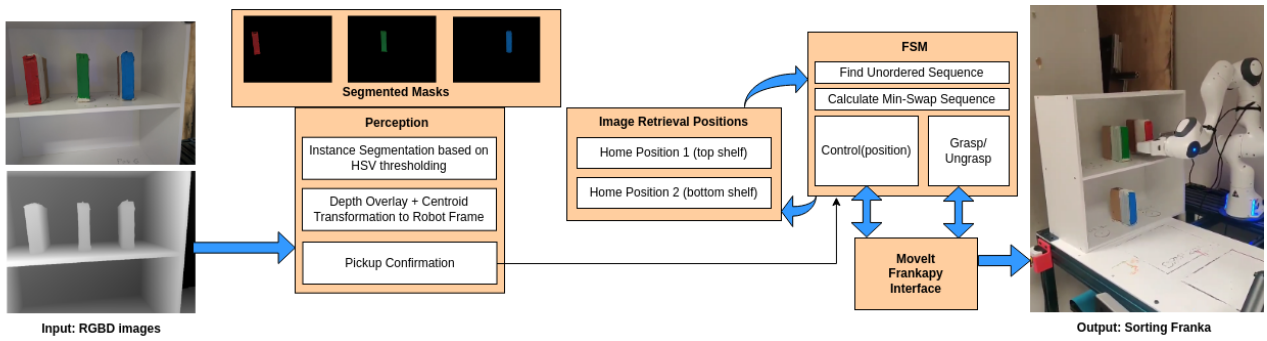

Fig. 2. SO(n)RT System Architecture

Fig. 3. Real-time tracking of Object Centroid

corresponding to the specified colors. For each color region, centroid calculation is performed using contour analysis to determine the geometric center. Depth values from the depth image are then extracted and used to back-project the 2D pixel coordinates into 3D space based on the camera's intrinsic parameters. These 3D world coordinates, along with their corresponding color values, are published as PointStamped messages, providing real-time spatial information. This data is subsequently utilized by the planning node for trajectory planning and task execution, ensuring seamless integration of perception with motion.

C. Sortation Algorithm

The Minimum-Swap Sortation Algorithm targets the efficient rearrangement of books in a shelf, for three target colors (e.g., 'R', 'G', 'B') and two empty spots (denoted as None). The primary objective is to achieve a predefined sorting order with minimal swap actions, transforming this task into an optimization problem that minimizes mechanical effort and computational overhead. For a scenario like sorting books, cyclic-sort is found to be highly efficient, non-comparative sorting algorithm. It operates by exploiting the principle of cyclic permutations of misplaced elements and sorting them in-place, which is very efficient.

Before any sorting begins, we define the target order for the books on the shelf which could be a specific sequence, such as ['R', 'G', 'B']. Further, we obtain the current order on the shelf from the perception node and traverse through it to identify where each book is located compared to its desired position. Each book that is not in its target position will be involved in a cycle. Each cycle involves a set of books that need to be swapped in a specific sequence to bring them to their correct positions. The empty spots (None) are utilized strategically such that when swapping books between positions, if the target position is occupied by another book, we can use an empty spot as a temporary holding place. Finally the algorithm outputs a sequence of move instructions including the pick-and-place indexes for each swap. These instructions are transmitted to the motion planning node, enabling seamless execution of the task with minimal physical movement and higher efficiency. Algorithm-1 provides a comprehensive overview of the sortation workflow.

For our stretch goals, we implemented a feature to check if the book has been grasped and placed successfully in its respective spots. Previously, it was an open-loop operation, but now we make sure that the book has reached its spot. We do this by getting visual feedback from the camera and matching that with the sequence in which the books are supposed to be placed. Another of our stretch goals was to dynamically sort books based on human-induced disruptions. We were able to achieve this by checking the position of the books after every place operation. We can then verify this with regard to the desired position of the books in the algorithm and redo the pick-and-place operation to achieve the required book positions.

D. Motion Planning

The motion planning node system utilizes the sequence of pick-and-place operations generated by the sortation algorithm, where each operation includes the slot index for both pick and place positions. To facilitate precise robot motion, these index-based positions are transformed into real-world coordinates. The "place" coordinate is derived by converting the index into 3D world coordinates, using the predefined joint angles associated with the centroids of each slot. For the "pick" operation, the perception node provides the 3D coordinates of the object along with its associated color. However, this coordinate is in the camera frame and needs to be converted into robot's base frame since the robot's motion planning function "goto_pose()" requires input this data in

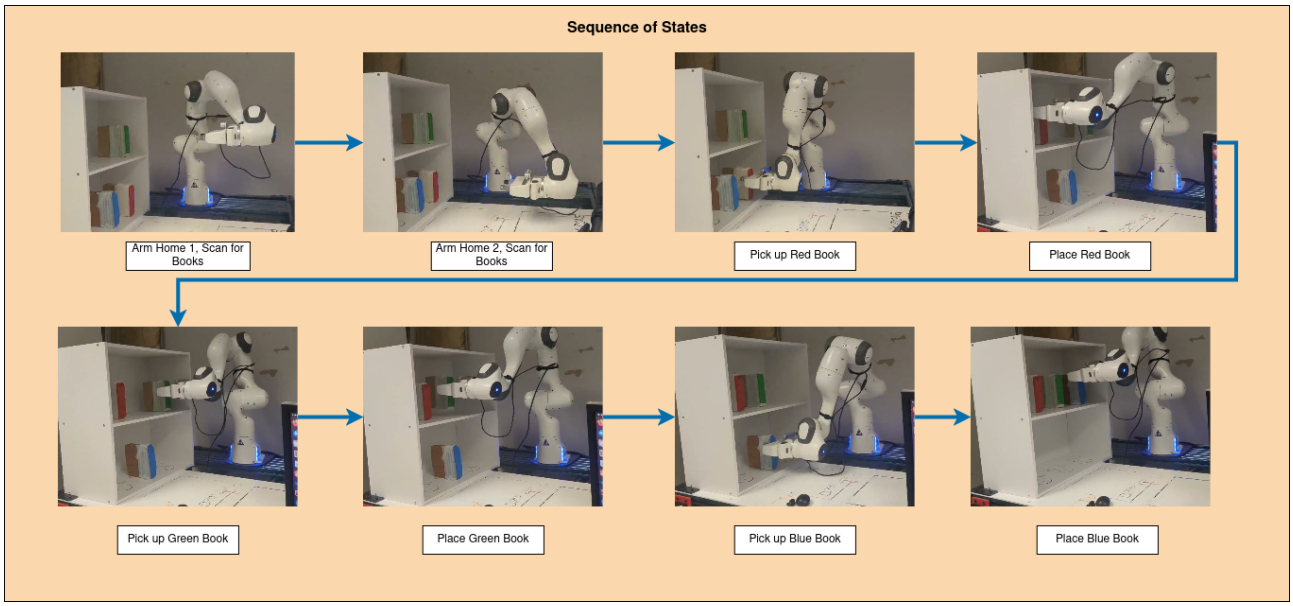

Fig. 4. Sequence of States for the FSM

the world frame. To achieve this we perform the following transformation (P refers to position coordinates):

$$\mathbf{P}_{\text{robot base}} = \text{transform}_{\text{camera}\to\text{robot}} \times \mathbf{P}_{\text{camera}}$$

$$\text{transform}_{\text{camera}\to\text{robot base}} = \text{transform}_{\text{camera}\to\text{gripper}} \times \text{transform}_{\text{gripper}\to\text{robot base}}$$

Fig. 5. Camera-frame to World-frame transformation

We obtain the transformation of the camera to robot base using ”get_pose()” function which essentially gives the pose of end-effector in the world frame. However, the transformation between the camera and the gripper frame is manually calculated by measuring the translation of the centroid of the camera relative to the gripper. The rotation matrix is considered to remain consistent between the gripper and the camera, as both are rigidly mounted on the gripper and share a fixed orientation relative to the robot base.

Algorithm 1 Min-Swap Book Sorting

Require: Shelf environment with Franka MoveIt controller
Ensure: Books sorted in target order with minimal swaps

Initialize Shelf environment
procedure INITIAL SCAN
    move_to_home_2
    get_centroids
    move_to_home_3
    get_centroids
end procedure
if ¬(red_centroid ∧ blue_centroid ∧ green_centroid) **then**
    error ”Failed to get all centroids”
end if
initial_order ← determine_centroid_positions
sorter ← BookSorter(*initial_order*)
(*moves*, *final_order*) ← *sorter*.sort_books()
parsed_moves ← Sort_MoveIt.parse_moves(*moves*)
for each (*color*, *initial_pos*, *target_pos*) in *parsed_moves* **do**
    Calculate *home_initial* based on *initial_pos*
    Calculate *home_target* based on *target_pos*
    move_to_home(home_initial)
    pickup(color)
    move_to_home(home_initial)
    move_to_home(home_target)
    place(target_pos)
    move_to_home(home_target)
end for

By applying these transformations, we can successfully convert the “pick” coordinates into the world frame (robot’s base frame). Once both the pick and place coordinates are computed, the motion planning process can commence, except, there is an additional step required to ensure an

optimal trajectory. Finally, the Franka arm plans an optimal trajectory while avoiding collisions and optimizing the path for efficiency.

IV. EVALUATION

The system was evaluated on a custom two-row shelf comprising six equally spaced slots (15 cm center-to-center). Each slot was populated with up to three color-coded books (red, green, blue) in randomized initial poses. For every trial, the target configuration was defined as [R, G, B, None, None, None], and a total of 10 independent runs were executed, with randomized book placements between trials.

The sort operation succeeded in 96 % of trials, with a single failure resulting from an incomplete grasp. This was due to an error by a miscalibrated camera-to-robot-base transform that introduced an unintended lateral offset in the planned gripper trajectory and caused the book to be dropped mid-transfer. The system does not verify successful grasps at the moment. However, we aim to include a method to verify successful grasps in the future to avoid such failure cases.

The overall results confirm that simple HSV thresholding yields reliable object segmentation under controlled lighting. The accurate computation of the 3D-world coordinates for the book centroids supports precise pick-and-place operations. The high success rate of the overall execution demonstrates that the minimum-swap optimizer and cartesian planner delivers efficient performance for sorting operation.

Further evaluations that would be interesting could use more occupied spots creating more cluttered environments, reducing the margin for error for grasps. One of our stretch goals is to include collision boxes on the fly by incorporating information from the perception node. These would also allow for more complex paths between picks and place instead of using an intermediary home position.

V. CHALLENGES

Our implementation encountered several significant technical obstacles during development. We faced persistent difficulties with the RRT-Connect motion planning algorithm, which frequently failed to generate any viable path plans.

Algorithm 2 Finite State Machine

```
Global State:
    shelf: Array[6]                          ▷ Current book positions
    target_order = [r, g, b, None, None, None]   ▷ Example Position
    moves: List                              ▷ Action recording
function FINDEMPTYSPOTS
    return [i | i ∈ 0..5, shelf[i] = None]
end function
function FINDBOOK(target_color)
    for i ← 0 to 5 do
        if shelf[i] = target_color then
            return i
        end if
    end for
    return -1                                ▷ Not found
end function
procedure MOVEBOOK(source, destination)
    book ← shelf[source]
    shelf[source] ← None
    shelf[destination] ← book
    APPEND(moves, "Move {book} from {source+1} to {destination+1}"))
end procedure
function SORTBOOKS
    for target_pos ← 0 to 5 do
        target_color ← target_order[target_pos]
        if target_color = None then continue
            current_pos ← FINDBOOK(target_color)
            if current_pos = target_pos then continue
                if shelf[target_pos] = None then
                    MOVEBOOK(current_pos, target_pos)
                else
                    empty_spots ← FINDEMPTYSPOTS
                    temp_pos ← empty_spots[0]
                    MOVEBOOK(target_pos, temp_pos)
                    MOVEBOOK(current_pos, target_pos)
                end if

                return (moves, shelf)
```

We ended up using a cartesian planner to alleviate these issues. The hand-eye calibration presented another major challenge when we discovered the provided transformation matrix contained inaccuracies, requiring us to manually measure the physical distance between the camera and the robot's end effector to get the proper transformation to convert the centroid of the book in end effector space from camera coordinates to robot base frame. We also struggled with overly restrictive virtual wall configurations that had been set well within the robot's actual working boundaries, severely limiting movement and triggering unnecessary safety stops.

VI. FUTURE WORK

In future iterations of our system, we aim to implement a one-shot continuous path planning strategy that eliminates

unnecessary intermediate steps, such as returning to a predefined home position between actions. Achieving this requires real-time, dynamic updates to the configuration space (C-space) at each planning checkpoint. Specifically, we plan to integrate continuous feedback from the perception module, which provides updated centroids for all objects within the environment. This will allow the planner to accurately reflect the current state of the workspace. During each task step, the object being manipulated (e.g., a specific book) will be excluded from collision consideration, while the remaining objects will be incorporated into the planning space as obstacles using the addbox() function. This approach is expected to improve planning accuracy and execution efficiency by maintaining a realistic and responsive model of the environment throughout task execution and we hope to implement this in the future.

VII. CONCLUSIONS

In this work, we have presented a fully integrated robotic sortation system that combines perception, motion planning, and a minimum-swap sorting strategy to autonomously organize books on a shelf in a semi-structured environment. By leveraging the Franka Emika Panda robotic arm and Intel RealSense RGBD camera, our system is capable of accurately detecting and localizing objects, optimizing the sequence of operations, and executing precise pick-and-place actions in real-world conditions. The proposed minimum-swap algorithm effectively reduces redundant motions, leading to improved operational efficiency and minimal mechanical effort.

Our results demonstrate the viability of color-based segmentation and cyclic sort-based planning for real-time manipulation tasks in structured shelf environments, offering a promising approach to scalable robotic automation. Moreover, the modular architecture allows for future extensions, including the incorporation of more complex object categories, dynamic re-planning under uncertainty, and learning-based perception or control methods.

This work moves toward practical robotic automation by bridging academic research and industrial deployment. Future directions include semantic reasoning for generalized sorting, robustness under occlusion, and scaling to multi-object, multi-arm tasks in dynamic settings.

VIII. ACKNOWLEDGEMENTS

We express our profound appreciation to the Robotics Institute at Carnegie Mellon University for providing the necessary resources and environment for this research. We are particularly grateful to Prof. Oliver Kromer, the instructor of the 16662 Robot Autonomy course, whose expert guidance was essential during this project's development. Additionally, we thank our teaching assistants, Madhusha Goonesekera and Shrudhi Ramesh, for their dedicated support and valuable feedback, which were critical to our success. Their efforts were key in addressing the challenges of designing and implementing robotic systems.

REFERENCES

[1] Garrido-Jurado, Sergio, et al. "Automatic generation and detection of highly reliable fiducial markers under occlusion." Pattern Recognition 47.6 (2014): 2280-2292.

[2] Malamas, Elias N., et al. "A survey on industrial vision systems, applications and tools." Image and vision computing 21.2 (2003): 171-188.

[3] Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023.

[4] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

[5] David Coleman, Ioan A. Șucan, Sachin Chitta, Nikolaus Correll, "Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study, Journal of Software Engineering for Robotics", 5(1):3–16, May 2014.

[6] Kapelyukh, Ivan, et al. "Dream2real: Zero-shot 3d object rearrangement with vision-language models." 2024 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024.

[7] R. Ghzouli, T. Berger, E. B. Johnsen, A. Wasowski and S. Dragule, "Behavior Trees and State Machines in Robotics Applications," in IEEE Transactions on Software Engineering, vol. 49, no. 9, pp. 4243-4267, Sept. 2023, doi: 10.1109/TSE.2023.3269081.

[8] Zeng, F., Gan, W., Wang, Y., Liu, N., Yu, P. S. (2023). Large Language Models for Robotics: A Survey. ArXiv. https://arxiv.org/abs/2311.07226