Bilkent University
Department of Computer Engineering

Senior Design Project
T2325
LibreBot

Detailed Design Report

21902298, Giray Akyol, giray.akyol@ug.bilkent.edu.tr
21901779, Muhammed Can Küçükaslan, can.kucukaslan@ug.bilkent.edu.tr
21802880, Muhammet Hikmet Şimşir, hikmet.simsir@ug.bilkent.edu.tr
21803473, Mustafa Utku Aydoğdu, utku.aydogdu@ug.bilkent.edu.tr
21802530, Mustafa Yasir Altunhan, yasir.altunhan@ug.bilkent.edu.tr

Supervisor: Salih Özgür Öğüz
Course Instructors: Erhan Dolak, Tağmaç Topal

2023-05-19

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Contents

Detailed Design Report

LibreBot

1 Introduction

Organizing books in a library and keeping them organized is a tedious and time-consuming job that humans still perform. The time and effort put into finding a desired book in the library by a library user increases as the number of books and categories increase in the library. In the age of the fourth industrial revolution, where autonomous robots became a significant part of the workforce in many sectors, the use of robots in this daunting job is still very limited. Our project's purpose is to model a robot that can be used to automate the organization of library books taking advantage of state-of-the-art technologies. The robot will be able to perceive its environment and detect books using computer vision, identify books by radio frequency identification (RFID) tags or call numbers, and collect from or place them on appropriate shelves, according to the library classification system, using the state-of-the-art machine learning technologies for task and motion planning (TAMP).

1.1 Purpose of the system

Our project's purpose is to model a robot that automates the process of placing a delivered book on the shelves, finding and bringing the desired book to the user, and maintaining the book order of the library. When a book is returned the robot will be able to place the book in the correct shelf in the library. The robot also can collect unattended library books from the study desks and put them on the appropriate shelves. Besides those, it will be able to bring the desired book that was specified by its "call number" from a user. To specify a call number to the robot, the user will log in his/her library account, assuming the library management integrated control instructions of the robot to their library website with our help, and add its desired book to the queue of books to fetch. The robot will also scan through the shelves to check whether there are any books placed on wrong shelves or in wrong order regularly -with time interval specified by the library personnel. The library management and a normal user will have different privileges for the control of the robot. For instance, the library manager may command the robot to look for the shelves and adjust the order of the books if their order is not correct whereas a normal user may only be authorized -indirectly– to issue a book fetch request in the system.

Normally, building such a robot involves heavy work for both hardware and software components. However, our project is focused on the software part of the robot that involves but is not limited to computer vision, object detection, object recognition, decision-making for both low and high-level decisions, motion planning under specific constraints (either using solvers for linear and non-linear optimization problems or using reinforcement learning). We will use the drake robotics framework [1] to design the robot model, to simulate the behavior of the robot, to simulate the 3-D environment involving robots, bookshelves, books, desks, people, and other 3-D objects, and to test the functionalities.

1.2 Design goals

The LibreBot should be compatible and complementary with the existing library system:

- To avoid duplication or inconsistency of data between the robot and the library system
- To enable seamless communication and coordination between the robot and the library staff
- To enhance the efficiency and accuracy of the library operations
- To improve the user experience and satisfaction

The LibreBot should be reliable, safe and secure:

- To prevent malfunction or breakdown of the robot that could disrupt or damage the library services
- To protect the robot from unauthorized access or misuse that could compromise its functionality or data

- To ensure the safety of users and staff who may encounter or interact with the robot
- To comply with ethical and legal standards for using robots in public spaces

2 Requirements Details

2.1 Functional Requirements

2.1.1 System Functionalities

System functionalities refer to the library robot's capabilities developed for conducting specific tasks inside the library. The functionalities of the library robot consist of several core parts.

2.1.2 Perception

The library bot makes decisions according to the environment around it. Hence one of its core functionalities is the perception of its environment. The library robot should be able to:

- Perceive books with different shapes (e.g. thin, large)
- Perceive desks in the library from which it can collect the books
- Perceive bookshelves in the library to collect books from or place books back on to
- Perceive obstacles such as humans, chairs, and trash boxes to avoid any collision with these
- Recognize irrelevant objects on the desks, such as notebooks, computers, books that do not belong to the library, and other personal accessories of the students. This ability will prevent robots from taking away incorrect types of items from desks to bookshelves.
- Detect whether a book has a call number on it. The call number is the text or code used to identify books in the library classification systems. If this text is detected, the robot will attempt to take the book from the desk and put it on the library shelves; otherwise, it will not make such an attempt.
- Recognize the call number to identify books belonging to the library. This recognition of the unique text will enable the robot to perform many functionalities, such as distinguishing different books and understanding what part of the library a book belongs to.
- Recognize the texts written on the library shelves to find the correct shelve to insert (or take from) the book.
- Perceive the overall structure (e.g., whether books are placed horizontally or vertically) of a particular environment (e.g., desk or bookshelves) to make decisions regarding the actions that will be taken to place books.

2.1.3 Decision-Making and Interaction

The library robot work will continue on the decision-making and interaction stage after perceiving the environment around it. The library robot should be able to:

- Take the necessary command from the users (via integrated library system), as input to the LibreBot software system, to conduct the following:
- Bring the specific book, which is specified as input by the user, from the library shelves
- Put back a specific book, when the books are returned, from the returned book desk to the shelves
- Move from one location to another within the library to conduct necessary tasks.
- Stop the motion when an obstacle is observed along the path, change the path if needed, and continue to the motion when the obstacle is no longer in the path.
- According to the perception data regarding the positioning of the book, make the necessary move to grasp the book. This move might consist of several sequences such as first altering the current position of the book to convert it into a more desirable position for seizing it, followed by the actual seizing act.
- According to the perception data regarding the overall structure of the library shelves, make the necessary moves to insert the book. This move might consist of several sequences, such as first creating a free space between other books to insert the book, and then actually inserting the book.
- According to the text data written on books (unique "Call Number" in case of Bilkent University) or bookshelves, stop in the shelf that the book needs to be placed while moving through the bookshelves.

2.1.4 User Functionalities

The LibreBot system will be developed as a software system. Hence, users' interaction with a robot (i.e., commands and requests given to the robot) will be provided through the input sent by the user to the LibreBot system.
Users of the system interacting with the library robot will be able to:

- Request a book from the library system integrated into LibreBot's system. Then the call number and other relevant details will be conveyed to the robot by the system. Finally, the robot will bring the book using this information.
- Command the robot to put returned books back on the library shelves over the system. The robot will identify and pick the book, then request the relevant information, such as the shelf location, from the library system. Finally, it will carry the book back to its original place.
- Command the robot to scan shelves for misplaced books. If the robot encounters an incorrect ordering of the books, it collects the books breaking the order of the appropriate library classification system, and inserts them into the correct position after scanning.
- For the safety of the people in the vicinity of the robot, the robot will have emergency halt buttons.

2.2 Non-functional Requirements

2.2.1 Performance and Efficiency

The library robot system should operate in real-time to conduct interaction between the environment and itself. Therefore the robot is required to conduct perception and recognition tasks, then make the necessary decisions and act accordingly in real-time.

2.2.2 Usability

The library robot should be easy to communicate with, and the model should be integrated into the library's existing digital system.

2.2.3 Extensibility and Portability

- The library robot model should be able to adapt to different shelving arrangements in libraries
- The library robot model should be able to adapt to different library classification systems.

2.3 Pseudo Requirements

- Python programming language will be used.
- Drake framework will be used for simulation and symbolic equations.
- For mathematical optimization SPOPT solver will be used which is licensed to Drake framework.
- Pydrake bindings of Drake framework will be used to program the robot from python programming language.
- Deepnote Jupyter notebooks will be used for learning and experimenting with design and collaboration.
- Git with GitHub will be used as the version control system.
- 3D models of the robot and other entities that exist will be stored as SDF or URDF files.

3 Final Architecture and Design Details

3.1 Overview

Our project's purpose is to model a robot that automates the process of placing a delivered book on the shelves, finding and bringing the desired book to the user, and maintaining the book order of the library. When a book is returned the robot will be able to place the book in the correct shelf in the library. The robot also can collect unattended library books from the study desks and put them on the appropriate shelves. Besides those, it will be able to bring the desired book that was specified by its “call number” from a user. To specify a call number to the robot, the user will log in his/her library account, assuming the library management integrated control instructions of the robot to their library website with our help, and add its desired book to the queue of books to fetch. The robot will also scan through the shelves to check whether there are any books placed on wrong shelves or in wrong order regularly -with time interval specified by the library personnel. The library management and a normal user will have different privileges for the control of the robot. For instance, the library manager may command the robot to look for the shelves and adjust the order of the books if their order is not correct whereas a normal user may only be authorized -indirectly– to issue a book fetch request in the system.

Normally, building such a robot involves heavy work for both hardware and software components. However, our project is focused on the software part of the robot that involves but is not limited to computer vision, object detection, object recognition, decision-making for both low and high-level decisions, motion planning under specific constraints (either using solvers for linear and non-linear optimization problems or using reinforcement learning). We will use the drake robotics framework [1] to design the robot model, to simulate the behavior of the robot, to simulate the 3-D environment involving robots, bookshelves, books, desks, people, and other 3-D objects, and to test the functionalities.

One of the innovation types that our project is related to is service. After our project is finished and implemented in a library system, a user of a library will no longer need to look for the exact location of a book in the library system, walk into the chamber shelf where the desired book is located, take the book and register the book into the library system but simply issue a fetch request of a specific book in the library system. Therefore, it will ease the job of a user.

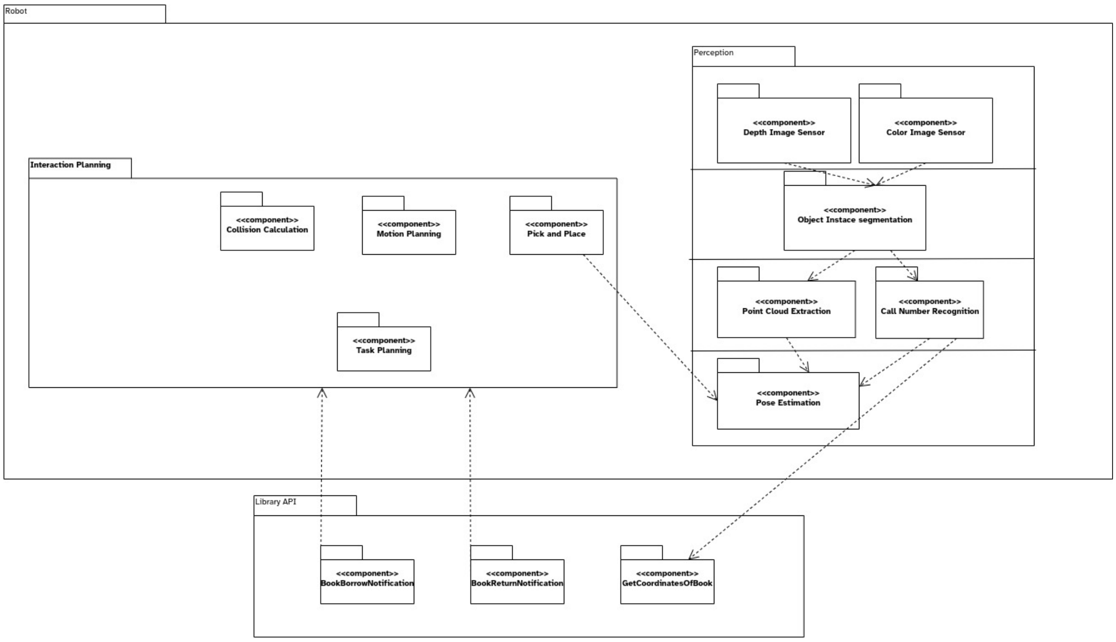
From time to time, users of the library may put a book on the wrong shelf or position so the library personnel may need to walk around and check the shelves to check if there are any books incorrectly located and put it into the correct position.

However, with the help of the robot, this task of the library personnel may be delegated to the robot which makes the life of the library personnel much easier.

The scope of the change of our project is transformation rather than optimization because its main aim is not to improve the productivity of an already existing mechanism of fetching and placing library books but to propose a totally new solution that take advantage of the state-of-art autonomous robot technologies to place returned books into the shelves, to fetch desired books from the shelves and to keep the correct order of the books in the shelves.

To illustrate how the robot works, consider a case where the robot places a book left on a table into the correct position of the library. First, the robot moves around the library to find any book left on a desk. When the robot encounters a desk with no one using it for at least 45 minutes (the amount of time can change according to a specific library rule), it stops next to the desk and runs an object detection algorithm for the items on the desk. The reason why the robot runs an object detection algorithm is that it should not pick a random object such as a pencil or water cup, but pick a book. After the execution of the object detection algorithm, it should pick any one of the books left using a motion planning algorithm to be able to grasp the object successfully without harming the environment. For instance, there may be people nearby so the robot should have a restricted working area and restricted angles for the movements of its arms in order not to hit any person or object nearby. Besides that, the book may not easily be grasped since there is no space between the book and the desk so the robot may move the book to a corner of the desk and grasp from there which requires different types of 6 or fewer degrees of freedom [2] in different parts of the motion. After that, an object recognition algorithm is run to check whether it is a user book or a library book. The plan, for now, is that the internal structure of the object recognition algorithm will benefit from the area of computer vision and look for a "call number" written on the book which is used to identify library books. After that, the robot needs to go to the correct position of the book in the library, which may or may not require the robot to go to a different floor or chamber. For the case of the robot going to a different flat, decision-making is used to decide which path to take in order to go to the specific flat and chamber. Feedback control is necessary to make sure the robot goes to the right place without harming or restricting its environment and itself. Each step or movement may give feedback to the robot so that it takes a better-optimized step or motion in consecutive movements. After finding and going to the right flat and chamber, the robot places the book into the correct position again using decision-making, motion planning, and object recognition (to find the exact spot of the correct position of the book by looking at the nearby books).

3.2 Subsystem decomposition



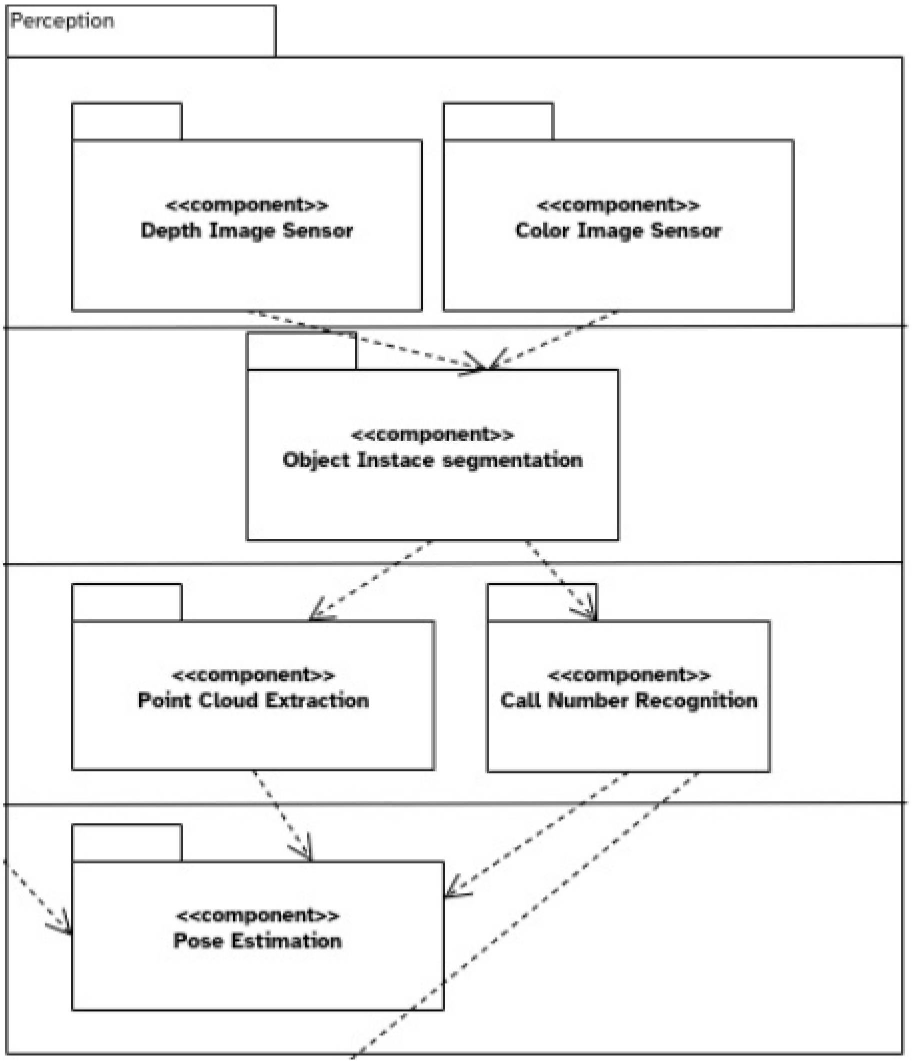
Based on functional requirements we decided to decompose our robotic software into two subsystems: Interaction Planning and Perception.

The Perception subsystem is designed to get data from the world and process it into a structured representation for other subsystems. It uses a layered architecture to organize data flow between components. Data comes from two sensors: Depth image sensor and Color image sensor. Sensor data flows into the segmentation subsystem which transforms it into a set of cropped images of objects such as book, shelf, obstacle, etc. In the next layer the images are fed into point cloud extraction which converts the 2D representation into a 3D representation. The call number recognition system extracts the call number of the book. The point cloud and call number are fed into the Pose estimation subsystem to calculate the pose of the objects in format usable by the interaction subsystem, this subsystem may also make a request for the coordinate information about this book to the library API.

Interaction Planning subsystem is designed to determine how the robot should perform in the physical environment. It has four components. Collusion calculation component is used to extract the collusion space of the environment given the point cloud of the environment as input. Motion planning is used to determine the optimal path that the robot should follow while avoiding the collusion space. After the robot follows an optimal path and arrives at a shelf, the pick and place component is used to determine the pre-pick and pre-put places of the book and to specify the necessary interpolation functions between the points. Task planning component is used to determine the order of commands to achieve an objective such as going from an initial point to the shelf where the desired book is located, picking the book and taking the book back to the initial point.

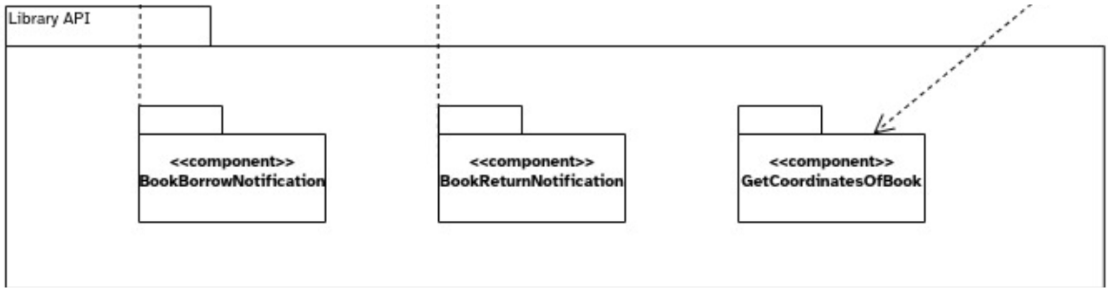
3.3 Subsystem Services

3.3.1 Perception



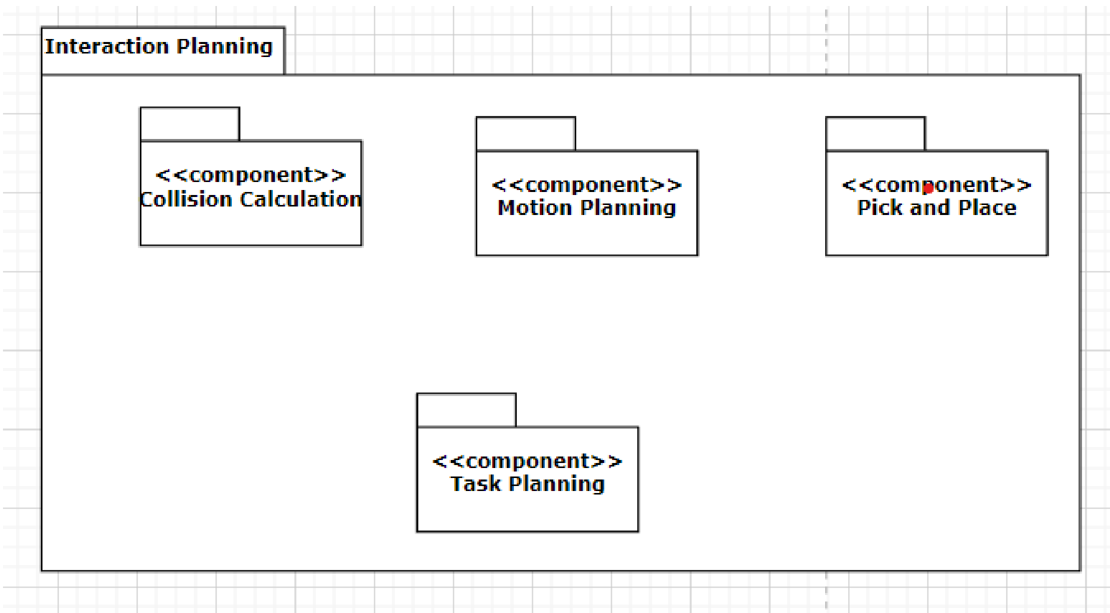
- Depth Image Sensor: Gather distance data and outputs a 2d array of depth metrics.
- Color Image Sensor: Gather color data and outputs a 2d array of RGB values.
- Object Instance Segmentation: Using ML extracts instance masks from the color image data and uses it to focus depth image to a certain object.
- Point Cloud Extraction: Converts masked 2D depth image to color points in 3D space.

- Call Number Recognition: uses OCR algorithms on the masked color image to recognize book call numbers.
- Pose Estimation: Using point cloud data iteratively calculates the 6 degree pose of the object for use by interaction subsystems.

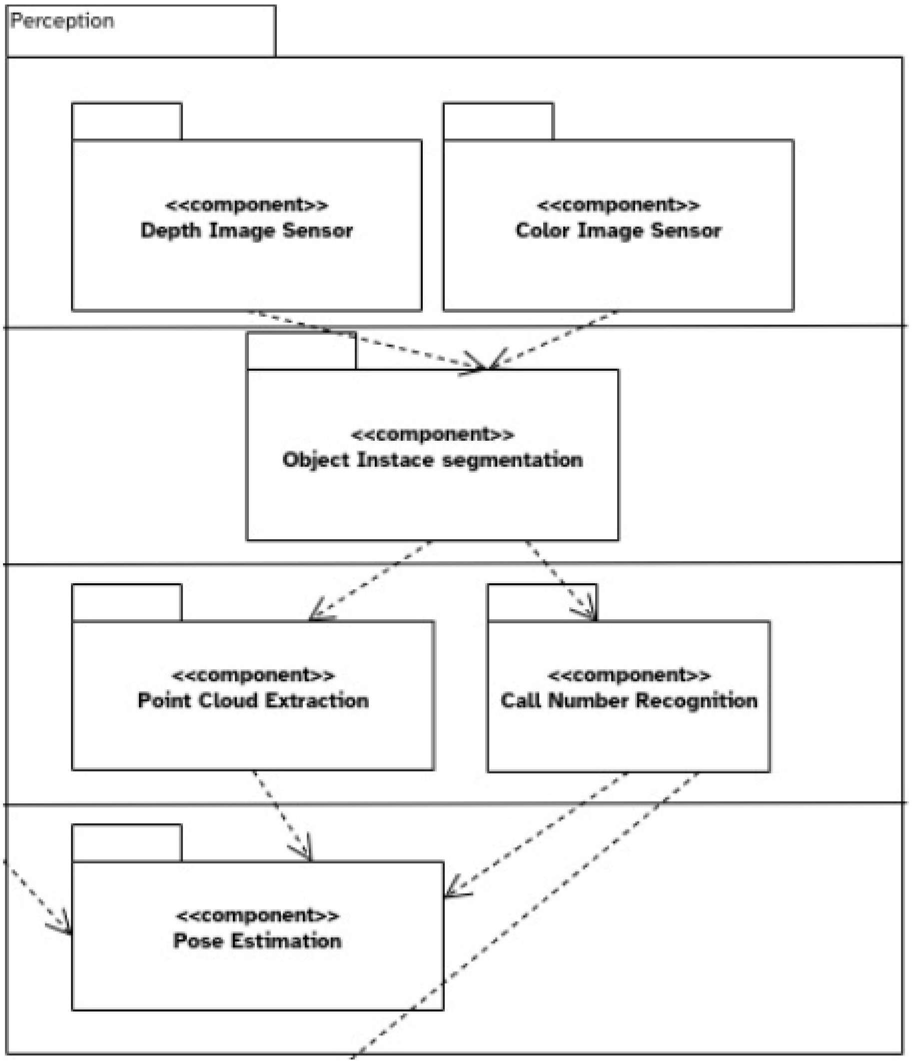


- BookBorrowNotification: When a user requests a book to be delivered this subsystem notifies the robot about the request.
- BookReturnNotification: When a user returns a book to the library this subsystem notifies the robot about the request.
- GetCoordinatesOfBook: Given a call number responds with where the book should be placed.

3.3.2 Interaction Planning



- Collusion Calculation: Using point cloud data obtained from the environment, it creates the collusion space.
- Motion Planning: Avoiding the collusion space and given the objective point for the robot to arrive, it calculates an optimal path.
- Pick and Place: Given initial and target pose of a book or object, it calculates the pre-pick, pre-put places and makes necessary interpolation between the desired points.
- Task Planning: It is used to specify the order of commands to give to the robot when the robot gets book borrow or book return notification.
- The submodules shown are for a single robot, there should be one of each for each robot, this was not shown for brevity.



4 Development/Implementation Details

- Python because of its strong interactive programming tools and first class Drake support.
- Drake robotics framework for all functionalities about the robot be it motion playing, task planning, perception and simulation.
- Meshcat for viewing simulations in a lightweight manner.
- Deepnote, Jupyter for interactive coding.
- Docker for packaging dependencies and a replicable setup.
- PyTorch for machine learning models such as the Segment Anything model.

- Mixed integer programming solver SPOPT for solving differential inverse kinematics problems.
- Git with GitHub will be used as the version control system.
- 3D models of the robot and other entities that exist will be stored as SDF or URDF files.

5 Test Cases and Results

We have provided you several test cases where each of them has its own specific category and field. The field is used to specify to which part or component of the project the test case belongs. The category is used to specify the type of issue or concept that the test case aims to address. Tables corresponding to the field and category are given below:

Field	
Image Segmentation and Classification Test Case	Seg-Class
Trajectory Calculation Algorithm in an environment with obstacles	Trajectory
Web	Web
Pick and Place	Pick-Place
Other Type of Test Case	Other

Category
Performance
Functionality
UI
Security

5.1 Image Segmentation Test Cases

In image segmentation, our main purpose in this project is to partition the images we take from the cameras inside the simulation into multiple segments or regions, each of which corresponds to a different object or region in the environment. The goal of image segmentation is to simplify and/or change the representation of an image into something more meaningful and easier to analyze. In our project, the main purpose is to extract depth images of the object around the robot, and take the corresponding action. One particular demonstration of this can be seen in the below figures. Our objective is to make a transition from the RGB Image on the left to the segmented image on the right.

During the image segmentation process, some of the faced problems are as following :

1) Ambiguity and uncertainty: Images can be ambiguous or uncertain, and the algorithm may struggle to segment them accurately. For example, objects with similar color or texture can be challenging to distinguish, and images with noise or low contrast can be difficult to segment.
2) Lack of labeled data: Image segmentation algorithms typically require labeled training data to learn the features and patterns of objects in the images. However, obtaining labeled data can be expensive and time-consuming, and may not be available for all types of images or applications.
3) Complex object shapes: Objects with complex shapes, such as those with concavities or non-convex boundaries, can be challenging to segment accurately.

The algorithm may also struggle to distinguish between objects that overlap or touch.

In our project, some of the problems are as following:

Test ID	Seg-Class_1	Category	Functionality	Severity	Crucial
Objective	Verify that the robot can recognize various objects (not just books) from the background from a segmented image.				
Input	An RGB image of a bookshelf with various books and objects on it.				
Output	A segmented depth image with each region or pixel labeled with a unique identifier.				
Test Criteria	The number and shape of regions or pixels should match the number and shape of books and objects in the input image.				
Date Result	May 9th, Pass				

Test ID	Seg-Class_2	Category	Functionality	Severity	major
Objective	Verify that the robot can recognize different sides of the same object instance (not just books) from a segmented image.				
Input	A segmented image of a bookshelf with each region or pixel labeled with an object instance identifier.				

Output	A labeled image with each region or pixel assigned a label.
Test Criteria	The labels should be correct and consistent for each region or pixel according to its appearance and features.
Date Result	May 9th, Pass

Test ID	Seg-Class_3	Category	Functionality	Severity	crucial
Objective	Verify that the robot can identify and classify all books in an image.				
Input	A labeled image of a bookshelf with each region or pixel assigned a label such as book or non-book.				
Output	A list of books identified and classified by their title from the input image. (Identification is based on color)				
Test Criteria	The list should contain all the books in the input image and their correct title and author.				
Date Result	May 9th, Pass				

Test ID	Seg-Class_4	Category	Functionality	Severity	middle
Objective	Verify that the robot can identify all non-book objects in an image such as book shelves that books are placed in, etc.				

Input	A labeled image of a bookshelf with each region or pixel assigned a label such as book or non-book.
Output	A list of non-book objects identified and classified by their type and category from the input image.
Test Criteria	The list should contain all the non-book objects in the input image and their correct type and category.
Date Result	May 9th, Fail

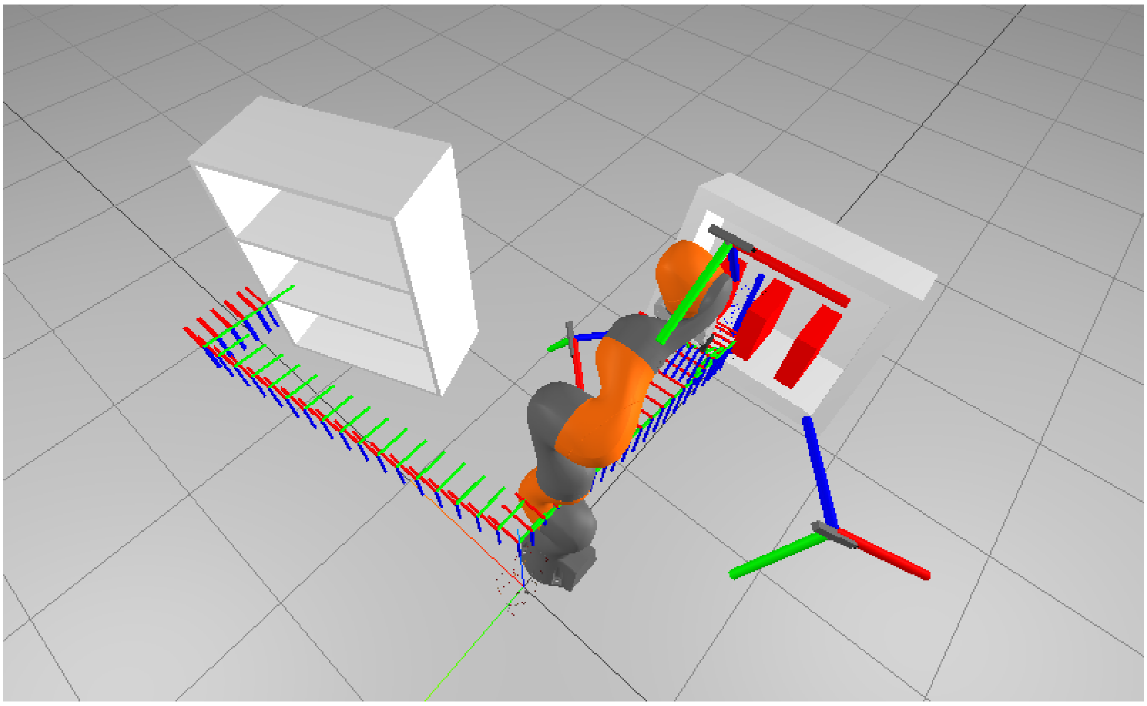
Test ID	Seg-Class_5	Category	Functionality	Severity	middle
Objective	Verify that the robot can handle different types of images that contain books or other objects in a library.				
Input	Different types of images that contain books or other objects in a library.				
Output	Segmented, labeled, identified and classified images for each type of input.				
Test Criteria	The results should be accurate and consistent for each type of input.				
Date Result	May 9th, Fail				

Test ID	Seg-Class_6	Category	Functionality	Severity	middle
Objective	Deal with camera noise that may affect the quality and consistency of image segmentation				
Input	Images with different levels of noise.				
Output	Segmented images for each level of difficulty.				
Test Criteria	The results should be robust and reliable for each level of difficulty.				
Date Result	May 9th, Pass				

Test ID	Seg-Class_7	Category	Functionality	Severity	middle
Objective	Deal with occlusion that may affect the quality and consistency of image segmentation				
Input	Images with different levels of occlusion.				
Output	Segmented images for each level of difficulty.				
Test Criteria	The results should be robust and reliable for each level of difficulty.				
Date Result	May 9th, Pass				

Test ID	Seg-Class_8	Category	Functionality	Severity	middle
Objective	Deal with camera position and camera angle changes that may affect the quality and consistency of image segmentation and recognition				
Input	Images with different levels of camera position and camera angle change.				
Output	Segmented, labeled, identified and classified images for each level of difficulty.				
Test Criteria	The results should be robust and reliable for each level of difficulty.				
Date Result	May 9th, Pass				

5.2 Planning/Kinematics Test Cases



Trajectory calculation can be performed with the utilization of the IRIS and GCS algorithms together. Given an obstacle in an area, calculating the path directly from one point to another without collision can be hard to calculate. There is a risk that there is an optimal solution since the area is not convex. To overcome this, the IRIS algorithm partitions the area into convex regions. In such convex regions, optimization problems are much easier to deal with. To make such partitions some seed points should be given to the algorithm. Testing the training of the algorithm with different seed points gives a better understanding of the IRIS algorithm and optimal seed points can be found. This makes the trajectory optimization problem easier to solve.

Test ID	Trajectory_1	Category	Performance, Functionality, Securtiy	Severity	Major
Objective	Finding the optimal seed points for a particular simulation setting. Plus, minimizing the training time of the algorithm.				
Input	Seed points and simulation environment				
Output	Set of partitioned regions				
Test Criteria	The set of convex regions are generated with respect to the seed points within a reasonable time interval.				
Date Result	TBD				

Test ID	Trajectory_2	Category	Performance, Functionality, Security	Severity	Major
Objective	Calculating a smooth trajectory				

Input	Simulation environment, regions created from the previous procedure, and joint position of the robot for the start and end of the trajectory
Output	Trajectory and run time of the movement
Test Criteria	A connection is constructed and the minimum cost is calculated.
Date Result	TBD

After finding convex regions, a graph algorithm (GCS) is used to find the least cost solution. This algorithm requires the joint positions of the robot at the start and end of the trajectory. Testing for different joint positions reveals that some joint positions lead to not connected graphs and there is no solution for such cases. Thus, for this algorithm, the joint position you provide to the algorithm determines the success of the test cases.

Test ID	Trajectory_3	Category	Functionality, Security	Severity	Middle
Objective	Evaluate whether the robot can pick up an object without squishing or putting stress on the object.				
Input	Simulation environment with an object in a bin with a given pose.				
Output	The robot has successfully picked up the object without damaging the object or the bin or itself.				

Test Criteria	The robot has not exerted too much force to damage the gripper or the joints. The robot has not exerted too much force to damage the object to be picked up. The robot has not exerted too much force to damage the bin.
Date Result	May 14th, Pass

Test ID	Trajectory_4	Category	Functionality, Security	Severity	Major
Objective	Evaluate whether the robot can place an object without squishing or putting stress on the object.				
Input	Simulation environment with an object in a bin with a given target pose.				
Output	The robot has successfully placed the object without damaging the object or the bin or itself.				
Test Criteria	The robot has not exerted too much force to damage the gripper or the joints. The robot has not exerted too much force to damage the object to be picked up. The robot has not exerted too much force to damage the bin.				
Date Result	May 14th, Pass				

Test ID	Trajectory_5	Category	Functionality, Security	Severity	Major
Objective	evaluate whether the robot transfer a picked up <1kg object without dropping				
Input	An object that is already inside the closed gripper. A trajectory to be followed				
Output	The robot has successfully transported the <1kg object without dropping it or damaging it.				
Test Criteria	The Robot should transfer one object at a time.				
Date Result	May 14th, Pass				

Test ID	Trajectory_6	Category	Security, Functionality	Severity	Major
Objective	evaluate whether the robot transfer a picked up >1kg object without dropping				
Input	An object that is already inside the closed gripper. A trajectory to be followed				
Output	The robot has successfully transported the >1kg object without dropping it or damaging it.				
Test Criteria	The Robot should transfer one object at a time.				

	Robot does not drop the object.
Date Result	May 19th, Fail

Test ID	Trajectory_7	Category	Security, Functionality	Severity	Major
Objective	evaluate whether the robot can limit the speed of the gripper.				
Input	A sequence of poses to be moved into				
Output	Robot has done the poses without going too fast.				
Test Criteria	Robot has done all the poses Robot has done poses in order Robot gripper has not gone over the speed limit.				
Date Result	May 14th, Pass				

Test ID	Trajectory_8	Category	Security, Functionality	Severity	Major
Objective	evaluate whether the robot can obey the angular speed limits of the joints.				
Input	A path to be followed by the gripper.				
Output	Gripper follows the specified trajectory.				

Test Criteria	Gripper does not deviate from the specified trajectory. Joint speed limits are respected
Date Result	May 19th, Pass

Test ID	Trajectory_9		Security, Functionality		Major
Objective	The gripper does not deviate from the trajectory.				
Input	A trajectory to be followed by the gripper.				
Output	Gripper follows the specified trajectory without deviation.				
Test Criteria	Gripper does not deviate from the specified trajectory. Joint speed limits are respected. Gripper speed limits are respected				
Date Result	May 19th, Pass				

Test ID	Trajectory_10		Security, Functionality		Major
Objective	The gripper follows the path under the time constraint.				
Input	A sequence of points to be visited by the gripper.				

Output	Gripper visits the points at the specified timestamp
Test Criteria	Gripper does not deviate from the specified trajectory. Joint speed limits are respected. Gripper speed limits are respected Points are hit according to their timestamp
Date Result	May 19th, Pass

5.3 User Interface (Website) Test Cases[1]

This test case corresponds to the user being able to search for a book by title, author, genre or keyword.

Test ID	Web_1	Category	UI, Functionality	Severity	Middle
Objective	verify that the user can find the book they want using different criteria				
Input	A valid title, author, genre, or keyword for a book				
Output	The website displays a list of matching books with their details and availability				
Test Criteria	The website returns relevant and accurate results based on the input				
Date Result	TBD				

[1] We assume that there is already a library website. So we only considered the test cases would be needed for integration of our robot (model) into the existing website/system.

Test ID	Web_2	Category	UI, Functionality	Severity	Middle
Objective	verify that the user can order a book and get it delivered by the robot				
Input	A valid book ID or call number from the search results				
Output	The website confirms the request and shows the estimated delivery time and location				
Test Criteria	The website sends the request to the robot and updates the status of the book				
Date Result	TBD				

Test ID	Web_3	Category	UI, Functionality	Severity	Middle
Objective	verify that the user can cancel a book order and get a confirmation from the robot				
Input	A valid book ID or call number from the request history				
Output	The website cancels the request and shows a confirmation message				
Test Criteria	The website notifies the robot and updates the status of the book				

Date Result	TBD

5.4 Pick and Place Test Cases

Test ID	Pick-Place_1	Category	Functionality, Security	Severity	Crucial
Objective	verify that the robot can pick up a book from a bookshelf according to its label.				
Input	A (red) book placed in a bookshelf.				
Output	The book is in the gripper.				
Test Criteria	The robot applies appropriate force and pressure to grasp. The book is recognized and the background is masked.				
Date Result	May 19th, Pass				

Test ID	Pick-Place_2	Category	Functionality, Security	Severity	Crucial
Objective	Verify that the masked depth image has enough points in the cloud for icp to work.				
Input	An RGB and depth image of a (red) book placed in a bookshelf.				

Output	masked depth image Book’s point cloud.
Test Criteria	The background is masked correctly Book is recognized ICP is run.
Date Result	May 19th, Pass

Test ID	Pick-Place_3	Category	Functionality, Security	Severity	Crucial
Objective	Verify that the camera is not too far (>10m) for the depth image to be correctly taken				
Input	A (red) book placed in a bookshelf. A depth camera				
Output	Book’s depth image				
Test Criteria	Book is identifiable in the image				
Date Result	May 19th, Pass				

Test ID	Pick-Place_4	Category	Functionality, Security	Severity	Crucial

Objective	Verify that the camera is not too close (<10cm) for the depth image to be correctly taken
Input	A (red) book placed in a bookshelf. A depth camera
Output	Book’s depth image
Test Criteria	Book is identifiable in the image
Date Result	May 19th, Pass

5.5 Other Test Cases

Test ID	Other_1	Category	Functionality	Severity	Crucial
Objective	verify that the robot can scan and recognize the book correctly				
Input	A book with a valid call number or RFID tag (will be simulated using color) on a shelf				
Output	The robot displays or announces the title and author of the book				
Test Criteria	The robot matches the book information with its database				
Date Result	May 19th, Pass				

Test ID	Other_2	Category	Functionality, Security	Severity	Major

Objective	verify that the robot can move smoothly
Input	A destination shelf or location for the book
Output	The robot reaches the destination shelf following a clearance point.
Test Criteria	The robot uses its sensors and map to plan and execute its motion
Date Result	May 19th, Pass

Test ID	Other_3	Category	Functionality, Security	Severity	Major
Objective	verify that the robot can sort and arrange the books correctly				
Input	A destination shelf number or location for the book				
Output	The robot puts the book on the designated spot on the shelf according to its category or sequence				
Test Criteria	The robot follows library rules and standards for shelving books				
Date Result	May 19th, Fail				

Test ID	Other_4	Category	Functionality, Security	Severity	minor
Objective	verify that the robot can transport 3 books within a timeframe (<10 min) and without dropping.				

Input	A number of books with valid call numbers or RFID tags on a shelf
Output	The robot carries the books in its gripper, arm, basket, or tray without dropping or damaging them
Test Criteria	The robot maintains balance and stability while carrying the books
Date Result	May 19th, Pass

Test ID	Other_5	Category	Functionality	Severity	middle
Objective	verify that the robot can handle invalid or unavailable inputs gracefully				
Input	A destination shelf number or location that is either non-existent, occupied, blocked, or out of reach for the book				
Output	The robot displays or announces an error message and asks for a new input or returns the book to its original shelf				
Test Criteria	The robot does not attempt to place the book on an invalid or unavailable destination shelf or location				
Date Result	May 19th, Fail				

Test ID	Other_6	Category	Functionality	Severity	middle

Objective	verify that the robot can recognize and select only books for shelving
Input	A mix of books and non-books items with valid RFID tags on a table
Output	The robot scans and identifies only the books and ignores the non-books items
Test Criteria	The robot does not grasp or lift any non-books items
Date Result	May 19th, Fail

Test ID	Other_7	Category	Performance	Severity	middle
Objective	evaluate the efficiency of two robot arm robots and a single storage and transport robot in retrieving requested books from designated locations				
Input	Simulation environment with two iiwa robots that are placed in different locations within the library, a storage and transport robot that is placed in a central location within the library, a set of requests from different users for specific books located throughout the library				
Output	The robots have successfully worked together to retrieve all requested books				

	The storage and transport robot has transported the books to a location where they can be picked up by one of the two robot arm robots
Test Criteria	The storage and transport robot must plan an efficient route for transporting all requested books to a location where they can be picked up by one of the two robot arm robots. The two robot arm robots must independently plan an efficient route for picking up each book from the designated location. The mobile storage robot must reach each designated location within a reasonable amount of time without getting stuck or taking an inefficient path. The two robot arm robots must pick up each book without dropping it or damaging it
Date Result	TBD

6 Maintenance Plan and Details

Our project consists of many components where each of them has its own state of art technology. Since technology related to computer science improves rapidly, the related components should also be updated. For the segmentation, the Segment Anything model was used. However, in future if a better model that is suitable to the project is found, then it will be replaced. Similarly, ICP was used to predict the pose of an object, differential inverse kinematic was used to predict the trajectory of the gripper. If better alternatives are found in future, then the team will update its components accordingly to keep up with the new technology. Besides those, our project relies on heavy mathematical functions implemented in python libraries. Therefore, software version updates will be performed regularly to improve the efficiency of the tools used in the project. The robot will use script to automate the updating process instead of manually updating every process and connect to the internet to download the necessary packages and updates.

7 Other Project Elements

7.1 Consideration of Various Factors in Engineering Design

In this section, many aspects that may affect the LibreBot design will be discussed.

7.1.1 Public Health Considerations

Since many diseases can be transferred through human contact, it is desired that LibreBot is as autonomous as possible. This decreases the risk of infection. LibreBot should not require the help of library personnel frequently.

7.1.2 Public Safety Considerations

LibreBot should not make any moves that can damage people or make bookshelves to fall to the ground. So, its vision should be wide enough to see surroundings, and high enough quality to detect objects, especially humans. Pick and place and pathfinding algorithms should be robust and consider any human contact while performing these operations.

7.1.3 Public Welfare Considerations

There is no direct effect of public welfare that influences the LibreBot design.

7.1.4 Global Considerations

Since LibreBot will be used by people speaking various languages, it should be designed to work with different languages. Also, vision text extraction should be able to work with different alphabets. Different book ordering systems should be recognized by LibreBot. There should be language support for different languages.

7.1.5 Cultural Considerations

From culture to culture, the interior design of libraries and bookshelves may change. Path finding algorithms should work with different library designs.

7.1.6 Social Considerations

Since libraries are social places where people prefer being quiet, LibreBot should not create much noise. To not disturb people, distracting elements such as lights, excessive movement, and vibrations should be minimized

7.1.7 Environmental Considerations

Since the robot will be active during the working hours of the library and constantly consuming energy, energy consumption should be minimized to make it more environmentally friendly. To achieve this, the design of the robot should achieve the followings:

- Energy-efficient motors and lightweight materials should be preferred. The use of heavy metals that damage the environment should be minimized, recyclable materials should be used instead.
- If the robot has more than one book to place, the robot should cover a minimal distance between bookshelves.
- In order to train the ML algorithms considerable computational power is needed which uses a significant amount of energy in the form of electricity. This energy may come from fossil or non-renewable sources that impact the environment negatively. Inefficient training algorithms and hardware should not be preferred.

7.1.8 Economic Considerations

Number of library bots in the library should be the minimum number that can satisfy the request and alignment requests in the library. That's why processing and performing requests should be handled efficiently.

	Effect Level	Effect

Public health	2	More autonomy
Public Safety	3	Better camera, pick and place and pathfinding algorithms
Public welfare	0	No effect
Global factors	4	Text recognition will be trained for different alphabets, language support
Cultural factors	1	Pathfinding algorithm trained for various library designs
Social factors	3	Less noise, movement, light, vibration
Environmental Factors	6	Less energy consumption
Economic Factors	7	Efficient request processing and performing the requests

Table 1: Factors that can affect analysis and design.

7.2 Ethics and Professional Responsibilities

- The source code will be accessible privately to the group members and graders in Deepnote and GitHub. The private source code should not be shared with third parties until the end of the project.
- The software frameworks and libraries should be used and given credit in compliance with the license requirement.
- Every Friday during Fall semester and every Wednesday (?) during spring semester , preferably face-to-face group meetings were scheduled with the team and the supervisor.

7.3 Teamwork Details

It is aimed that every person in the group will contribute to the project. In order to ensure this division of the work with respect to the skills and capabilities of the group members is important.

7.3.1 Contributing and functioning effectively on the team

- Consulted with the group members and chose the most sensible and popular options together as a group
- Contributed equally to the preparations of reports together as a group
- Participated in the project demo together as a group
- Task and Motion Planning Modules are implemented by Mustafa Yasir with the help of Muhammed and Giray.
- Image segmentation, using Segment Anything model, module is designed and implemented by Utku with the help of Giray and Hikmet
- Image segmentation, using Segment Anything model, module is designed and implemented by Utku with the help of Giray and Hikmet
- IRIS and GCS module is being designed and implemented by Hikmet with the help of Giray and Utku
- The very first prototype of pick and place demos was implemented by Giray with the help of other members
- The book storage was prepared by Mustafa and Muhammed
- The mobile iiwa was prepared by Muhammed and Hikmet
- The mobility of the book storage and multiple iiwa were done by Hikmet, Muhammed and Giray
- Weekly meetings with the project supervisor were held to get rapid feedback and keep everyone on the same page on what has been done and what is the next step.
- Designers and implementers of modules gathered in a meeting to collectively ensure the absence of any incorrect assumptions during the module integration process.

7.3.2 Helping creating a collaborative and inclusive environment

Throughout the project, we demonstrated an effort to establish the following:

1) Designating clear and realistic goals and expectations for the project , which can be implemented in the given time and equal amount of workload among the team members. This helped us to avoid any frustration and disappointment we faced during the project phases.
2) We have assigned roles and responsibilities based on each member's interests and strengths. This separation of work based on the member's interests increased the total amount of time each member spent on the project drastically.
3) Although we mostly divided the group work into independent parts in order to increase the parallel efficiency of the group, we did not only focus on our own parts but examined each other's work so that our independent parts are coherent in the overall project and gave feedback to our teammates so that the work of each team member was corrected and improved by other teammates.
4) Throughout the project, we respected each other's opinions and backgrounds, and came into a compromise when different ideas conflicted. We helped each other when a theoretical concept was not understood by one of the team members. Similarly, we helped each other during the implementation phases, which increased the comradeship inside the team.

7.3.3 Taking lead role and sharing leadership on the team

- The simulation environment was set up by Muhammed Can Kucukaslan and Mustafa Yasir Altunhan.
- Giray Akyol led the research on the algorithms and supported the team in implementation and research.
- Mustafa Yasir Altunhan oversaw some the paperwork.
- Mustafa Utku Aydoğdu led the design of ML algorithms for the vision part.
- Muhammet Hikmet Şimşir integrated the vision part of the project and implemented the IRIS & GCS algorithm.
- Muhammed maintained project website

7.3.4 Meeting Objectives

1. First goal of the semester was to transition from MASK-RCNN, which was working poorly, to a robust image segmentation module. One to two team members, depending on the schedule of the members, were assigned to this task. After testing numerous segmentation models Segment Anything was chosen. Integration of the model was done by 3 people because it was a bottleneck since it was blocking other modules.
2. Second goal, concurrent with the first, was to add a storage box robot to transport books. Again, this required considerable research into the drake framework to find the necessary methods/components. One person was assigned to make the storage box configurable.
3. Third goal was to implement the motion/task planning for the robots. Since our initial approach assumed only a single robot this required some architectural changes which prompted us to meet and discuss the approaches. After finalizing the new design, implementation work began swiftly and progressed rapidly.
4. Final goal was to integrate and fine tune all the modules, add scenarios for demos and such. Since this coincided with a busy week for courses it was worked on asynchronously by each member with quick progress updates.
5. We can add and control multiple iiwa arms
6. We can add storage (with arbitrary dimensions/capacity) and control its movement
7. The robot can recognize books with their colors (which is substitution for real world RFID solutions) and locate them
8. We can change the color of the iiwa arm
9. The robot compute and follow pick and place trajectories for arbitrary number of books

7.4 New Knowledge Acquired and Applied

- Python programming language, gained experience with working with classes and inheritance.

- Drake framework, we had literally no experience with this framework. Drake is not just a robotics framework it is a general dynamical system simulation and control system. We believe that this experience is transferable to other ones since drake is based on systems architecture like Simulink, which is a widely used system for developing dynamical systems.
- SPOPT solver for mathematical optimization, we have only used mathematical solvers on the IE400 course on a small scale. In our project math. solvers are a critical component and we wrote and debugged constraints and objective functions.
- Deepnote Jupyter notebooks for learning and collaboration, nearly all of our work was done via python notebooks. This interactive programming has significant advantages and considerable disadvantages. We had to have better harmony as a team to not break already working code.
- We gained experience with working with work-in-progress Drake library. We have found many inconsistencies and worked around them. There is nearly no documentation so we had to read source code which is in between a rewrite in C++ so we had to read source code to even understand basic functions.

We have gained an appreciation for choosing simple and boring solutions to problems because on a complex system it is already hard to find the causes of bugs. We have also learned to prioritize a result driven workflow to keep making progress.

8 Conclusion and Future Work

LibreBot is a cutting-edge project that aims to automate the organization of library books using advanced technologies. The proposed system features a fleet of robots that can navigate through the library, identify books, and place them in their designated locations. The system utilizes perception and decision-making functionalities to ensure efficient and accurate book placement.

Looking forward, there are several areas where future work can be done to improve LibreBot. For example we plan to commercialize the software by integrating it into the existing electronic library systems. To that end we need to create an API to allow the library's existing system to communicate with it. At the moment we only have the simulation, so we don’t use RFID tags to recognize books. But in a commercialized system we would need to. LibreBot’s path planning algorithm could be made more robust to obstacles, and we are planning to improve it before commercialization. A more powerful task planning engine could be integrated to decrease the time and energy spent in operation.

Overall, we believe that LibreBot has great potential in revolutionizing the way libraries organize their books. With further development and refinement, it could become an essential tool in libraries worldwide.

9 User Manual and Installation Guide

To install and use the project:

Clone the project from https://github.com/Fall22CS491/LibreBot

```
$ git clone  https://github.com/Fall22CS491/LibreBot
```

To execute the notebooks you need to use the docker image provided at https://hub.docker.com/r/hikmetsimsir/visionstorage2iiwa

You can use prepared script to do it

```
$ sudo ./docker_run_notebook.sh .
```

Then you can use the given jupyter notebook server to use the corresponding parts of the project

10 Glossary

Book Classification System: A system used to categorize books in a library based on their subject matter.

Call Number: A text or code used to identify books in the library classification systems. It is a unique identifier assigned to each book in the library, which helps in locating and organizing books on the shelves.

Image Segmentation: Separating the image into regions (that may partially cover an object)

Image Recognition: Reconstructing object masks from an image (masks must cover only a single instance)

11 References

[1] Russ Tedrake and the Drake Development Team, "Drake: Model-Based Design And Verification For Robotics," 2019. [Online]. Available: https://drake.mit.edu. [Accessed: 16-Oct-2022].

[2] Wikipedia, "Underactuation," Wikipedia, 17-Oct-2022. [Online]. Available: https://en.wikipedia.org/wiki/Underactuation. [Accessed: 16-Oct-2022].

[3] R. Tedrake, "Robotic Manipulation: Perception, Planning, and Control." [Online]. Available: http://manipulation.mit.edu. [Accessed: 16-Oct-2022].

[4] A. Quart, "Automation Is A Real Threat. How Can We Slow Down The March Of The Cyborgs?," The Guardian, 08-Aug-2017. [Online]. Available: https://www.theguardian.com/us-news/2017/aug/08/humans-v-robots-defending-jobs. [Accessed: 16-Oct-2022].

[5] I. Wladawsky-BergerGuest, "Automation And The 2030 Job Hunt," WSJ, 09-Feb-2018. [Online]. Available: https://www.wsj.com/articles/automation-and-the-2030-job-hunt-1518198504?tesla=y. [Accessed: 16-Oct-2022].

[6] R. Li, Z. Huang, E. Kurniawan, and C. K. Ho, "AuRoSS: An Autonomous Robotic Shelf Scanning system," in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sep. 2015, pp. 6100–6105. doi: 10.1109/IROS.2015.7354246.