Bilkent University
Department of Computer Engineering

Senior Design Project

T2325

LibreBot

Analysis and Requirement 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

2022-11-11

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

Analysis and Requirement 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).

2 Current System

AuRoSS is a system proposed by Li, Huang, et al. [6] that strives to solve the library book tracking problem LibreBot also aims to solve. A significant difference between LibreBot and AuRoSS is how the systems identify books. While both LibreBot and AuRoSS use RFID tags, LibreBot will also use computer vision and text recognition when RFID is not available. If the book does not have RFID tags LibreBot requires that the book label is not damaged, not obstructed and the environment has sufficient lighting to extract the information. On the other hand, RFID tags require high-accuracy positioning of the scanner and enough range of the scanner [6]. In general, libraries already have labels on books but RFID tags are new technology (by library standards). While scanning RFID tags may be quicker, to use AuRoSS, libraries must tag the books, which costs time and money. Compared to AuRoSS, LibreBot is an end-to-end system, it handles the book-handling process from an electronic book request to its delivery. AuRoSS only handles missing and misplaced books while LibreBot handles gathering the books from shelves, placing back borrowed books, and correctly sorting misplaced books. LibreBot’s scope is much larger and it is a more complete system compared to AuRoSS.

3 Proposed System

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 Functional Requirements

3.2.1 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.

3.2.1.1 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.

3.2.1.2 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.

3.2.2 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.

3.3 Non-functional Requirements

3.3.1 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.

3.3.2 Usability

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

3.3.3 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.

3.4 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.

3.5 System Models

3.5.1 Scenarios

Request Book

- Use-case Name: Request Book
- Actor: Library User, Librarybot
- Entry Condition: User specifies the desired book's call ID to the system.
- Exit Condition: Book is successfully returned
- Flow of Events:
  - User's request is sent to the system
  - Librarybot check for any request from the system
  - Librarybot gets the new book request

    - For each book in the request
        - Librarybot looks up the call number of the book from the system
        - Librarybot gets the bookshelf location and calculates a path
        - Librarybot navigates to the location successfully
        - Librarybot picks up the book
        - Librarybot places the book in the cart
    - Librarybot navigates to the dropoff location
    - Librarybot picks up the book from the cart
    - Librarybot places the book to the dropoff location
- Alternative Flows:
    - User's request is sent to the system
    - Librarybot check for any request from the system
    - Librarybot gets the new book request
    - Requested book is not available in the library
    - Librarybot informs the user that book does not exist
- Alternative Flows:
    - User's request is sent to the system
    - Librarybot check for any request from the system
    - Librarybot gets the new book request
    - Librarybot looks up the call number of the book from the system
    - Librarybot gets the bookshelf location and calculates a path
    - Librarybot navigates to the location successfully
    - The book is not placed to the correct location
    - Librarybot informs the user that book does not exist

Return One Book

- Use-case Name: Return book
- Actor: Library User, Librarybot
- Entry Condition: User specifies the call ID of the book to be returned to the system.
- Exit Condition: Book is successfully returned to the library shelf.
- Flow of Events:
    - User's return request is sent to the system
    - Librarybot check for any request from the system
    - Librarybot gets the new book return request
    - Librarybot looks up the call number of the book from the system
    - Librarybot navigates to the pickup location
    - Librarybot picks the book to be returned and places it in the cart

- Librarybot gets the bookshelf location and calculates a path
- Librarybot navigates to the shelf location successfully
- Librarybot picks up the book from the cart
- Librarybot inserts the book the shelf

Request to Place Misplaced Books In a Specified Shelf Correctly

- Use-case Name: Request to Place Misplaced Books In a Specified Shelf Correctly
- Actor: Library Personnel, Librarybot
- Entry Condition: Personnel makes a request to place the books in the correct shelves.
- Exit Condition: Books are correctly placed on the shelves.
- Flow of Events:
    - Personnel's request is sent to the system
    - Librarybot checks for any request from the system
    - Librarybot gets the new place check request
    - Librarybot starts navigating through the specified shelf
    - For each book in a shelf, LibraryBot acquire the call number of the book using text recognition
    - Librarybot requests and gets the correct bookshelf from the call number
    - After checking all books, if the cart does not have any misplaced books then, LibraryBot will proceed to the next shelf.
    - If there is a book not placed on the correct bookshelf then move the book LibraryBot's cart.
    - After all books on a specific shelf are processed, calculate the correct shelf location from the system and calculate a path to the location
    - For each book in the cart, navigate to the correct location and place the book.
    - After the correction is completed for the specified shelf, signal all books .are placed on the correct shelves

New Path Calculation when Obstacle is Encountered

- Use-case Name: New Path Calculation when Obstacle is Encountered
    - Actor: Librarybot, An Obstacle
    - Entry Condition: Librarybot is navigating inside the library for a specific request.
    - Exit Condition: Librarybot proceeds to navigate to the target location.

- Flow of Events:
    - An obstacle (either human or nonhuman) is encountered on the path to the target location.
    - Librarybot decides whether the obstacle on the path is a human or a nonhuman object according to the data from the camera.
    - If the obstacle is a human, LibraryBot stops moving until the obstacle passes.
    - After the obstacle is no longer present, Librarybot continues navigating to the target location.

- Alternative Flows:
    - An obstacle (either human or nonhuman) is encountered in the path to the target location.
    - Librarybot decides whether the obstacle on the path is a human or a nonhuman object according to the data from the camera.
    - If the obstacle is nonhuman, LibraryBot starts calculating a new path that does not coincide with the obstacle
    - After the obstacle is no longer present, Librarybot continues navigating to the target location.

Accident and Collision Repost

- Use-case Name: Accident and Collision Repost
- Actor: Library User, Librarybot
- Entry Condition: The Librarybot is navigating
- Exit Condition: A collision report is generated.
- Flow of Events:
    - Librarybot is in operation
    - A user is also moving close to the robot
    - User is in the blind spot of Librarybot
    - Librarybot moves toward the user (who is still in the blindspot)
    - User enters Librarybot's field of vision or vice versa
    - Librarybot perceives the user and halts movement.
    - Because of the leftover inertia and momentum the robot hits a bookshelf
    - Data from Librarybot's sensors indicate a collision.

- Librarybot generates a collision report with the data from the manipulator's arm poses, inertia, momentum, and speed.
- Librarybot sends the collision report to the system

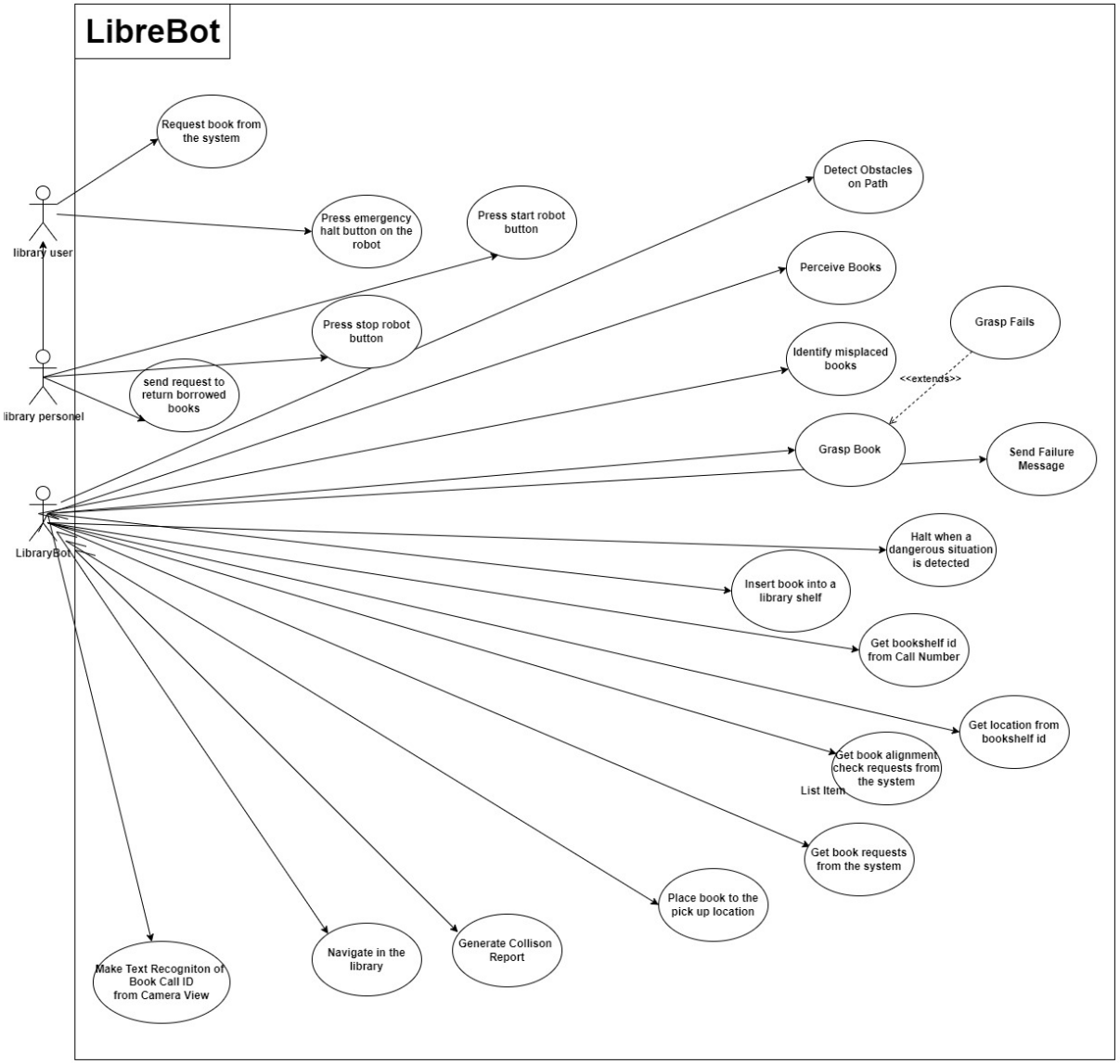
Change the Librarybot's state

- Use-case Name: Change the Librarybot's state
- Actor: Library User, Librarybot
- Entry Condition: User enters the system command of the desired state of the Librarybot
- Exit Condition: Librarybot's mode is changed.
- Flow of Events:
  - User specifies the turn-on mode
  - If Librarybot is not already turned on, Librarybot is turned on.
- Alternative Flows:
  - User specifies the turn-off mode
  - If Librarybot is not already turned off, Librarybot is turned off..

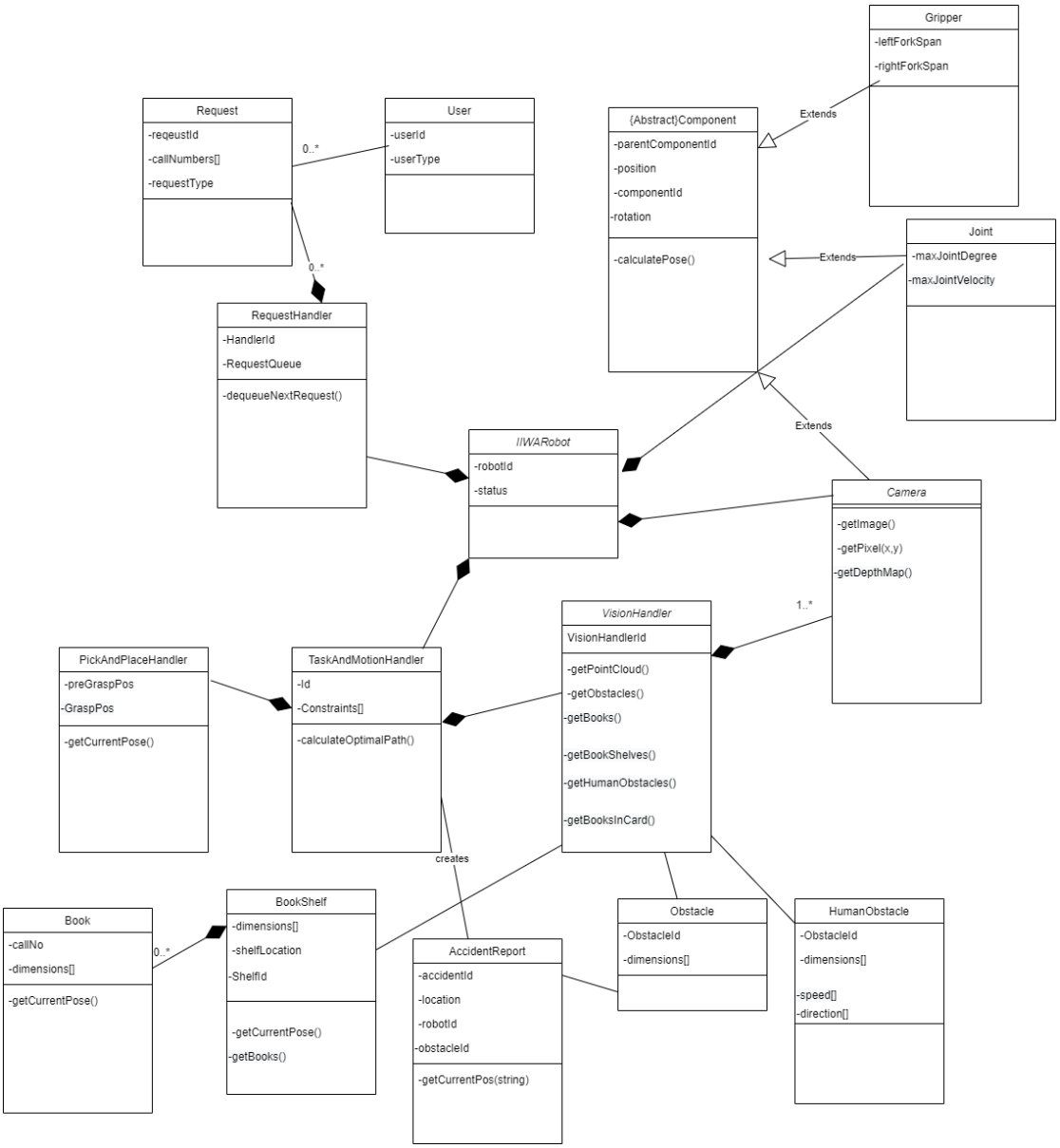
3.5.2 Use-Case Model

We prepared a use case diagram to summarize functionalities. However, beware that this project is not compatible OOP paradigm. So this diagram may not match what

would be expected from ordinary OOP project's diagram.



3.5.3 Object and Class Model



The IIWARobot class signifies the singular robot entity in the system. It has a unique identifier and operating status. It also has connections with components such as cameras, robot arms (joints) and grippers. Each of these components store their positions, rotation to calculate their poses relative to their parent (no parent means world is the parent) which combined with parents' relative poses allows to calculate the poses relative to the world frame.

Camera components are connected to the VisionHandler class which gets RGB images and depth maps from cameras and creates a point cloud which is then used to do semantic categorization. From the result of semantic categorization objects in the world such as books, bookshelves and obstacles (human or nonhuman) are initialized. Classes such as books and bookshelves correspond to their real life counterparts and store their dimensions and identity. Obstacle and HumanObstacle classes differ because we assume much of the obstacles will be stationary and could be maneuvered around but human obstacles are, generally, not stationary and can move around, in

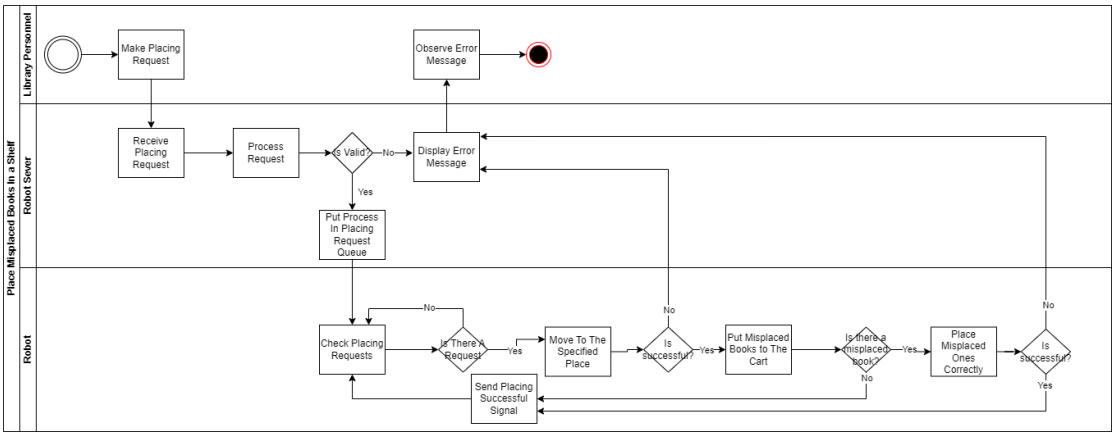
order to not put humans in danger the robot would keep a safe distance from the human obstacles and estimate their speed to take precautionary measures. TaskAndMotionHandler class controls almost all aspects of the robot, it gets the world objects created from the VisionHandler and dequeues a request from the RequestHandler. From then on, it keeps state and handles all aspects of the operation. It solves quadratic optimization programs to find optimal pseudoinverses and sends the objects to be grasped to PickAndPlaceHandler which calculates pregrasp poses, and communicates with TaskAndMotionHandler to manipulate the gripper. TaskAndMotionHandler also communicates with the robot which controls the components. Accident report class is for storing collisions or accidents and has a unique id and stores information about the accident such as location, id of the robot and obstacles involved.

RequestHandler handles communicating with the system to get requests from the users and queue them for increased efficiency. The request class has a unique id for each instance it stores its type (book request, return book request) and requesting user. The user class stores the user type (library user or library personnel).

3.5.4 Dynamic Models

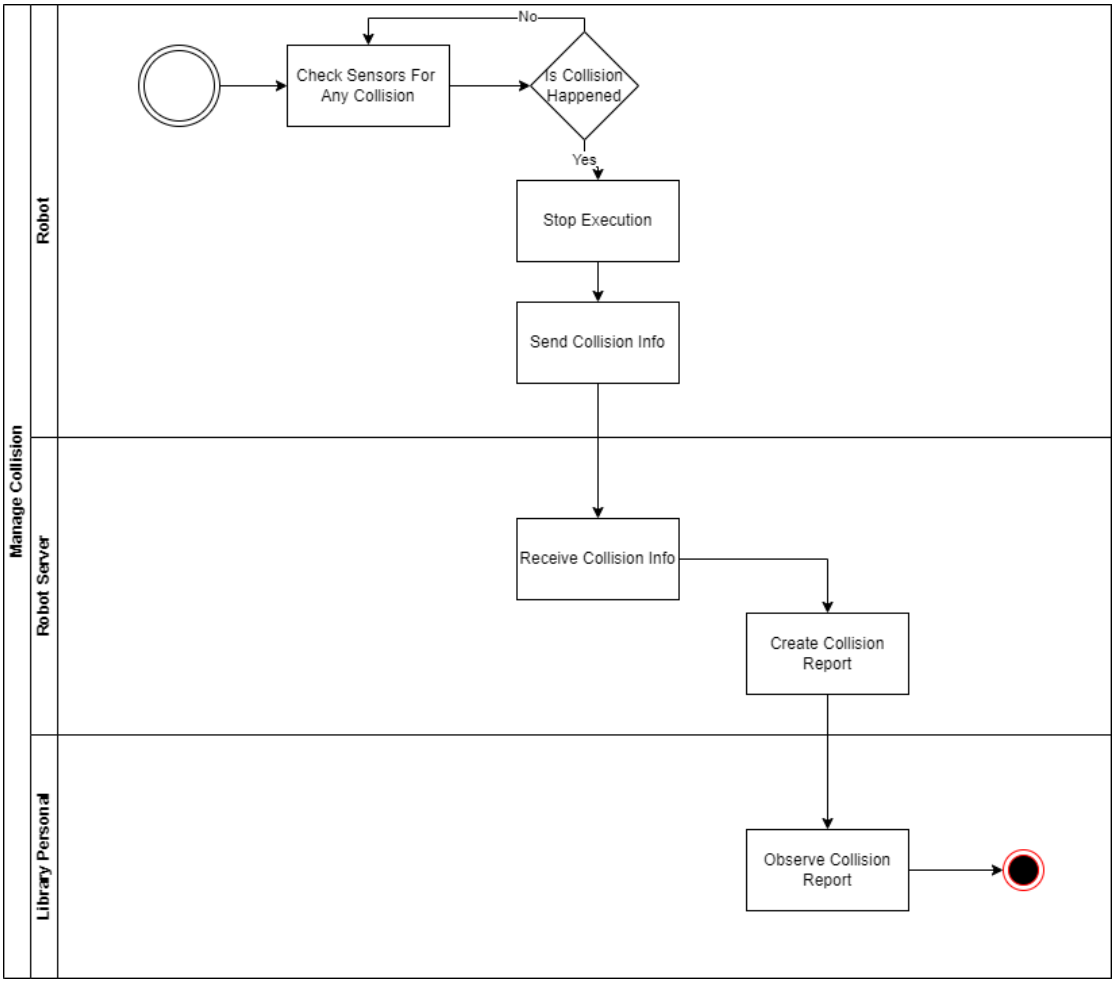
3.5.4.1 Activity Diagrams

3.5.4.1.1 Place Misplaced Books In a Shelf



Library personnel make a request to place the misplaced books on a certain shelf. The robot server and the robot manages this operation.

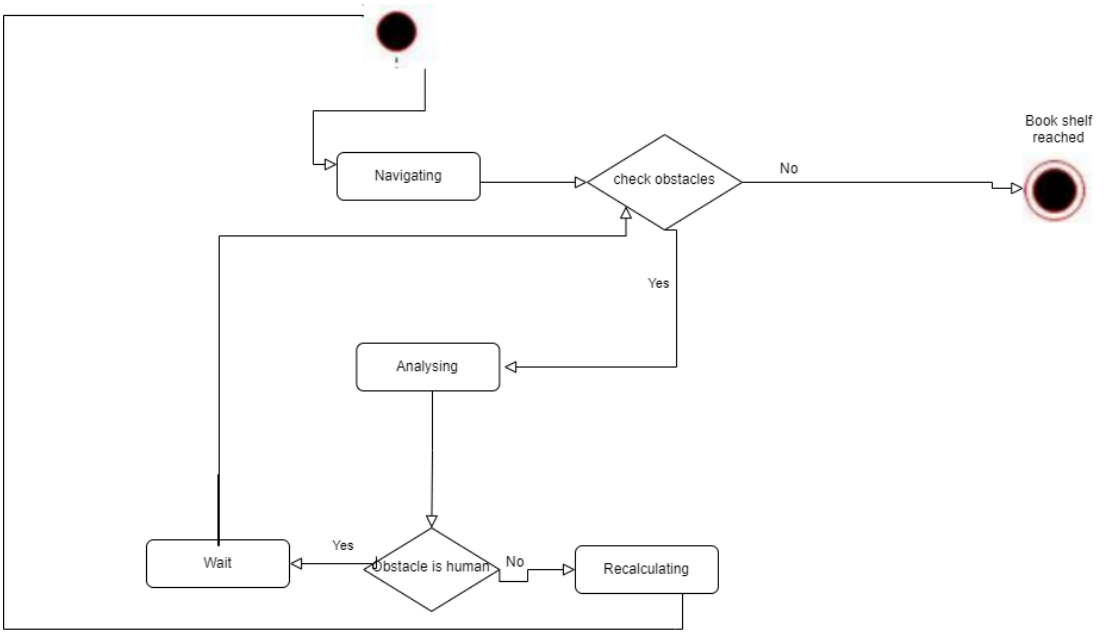
5.4.1.2 Manage Collision



If a collision happens LibreBot should stop working and send a collision report.

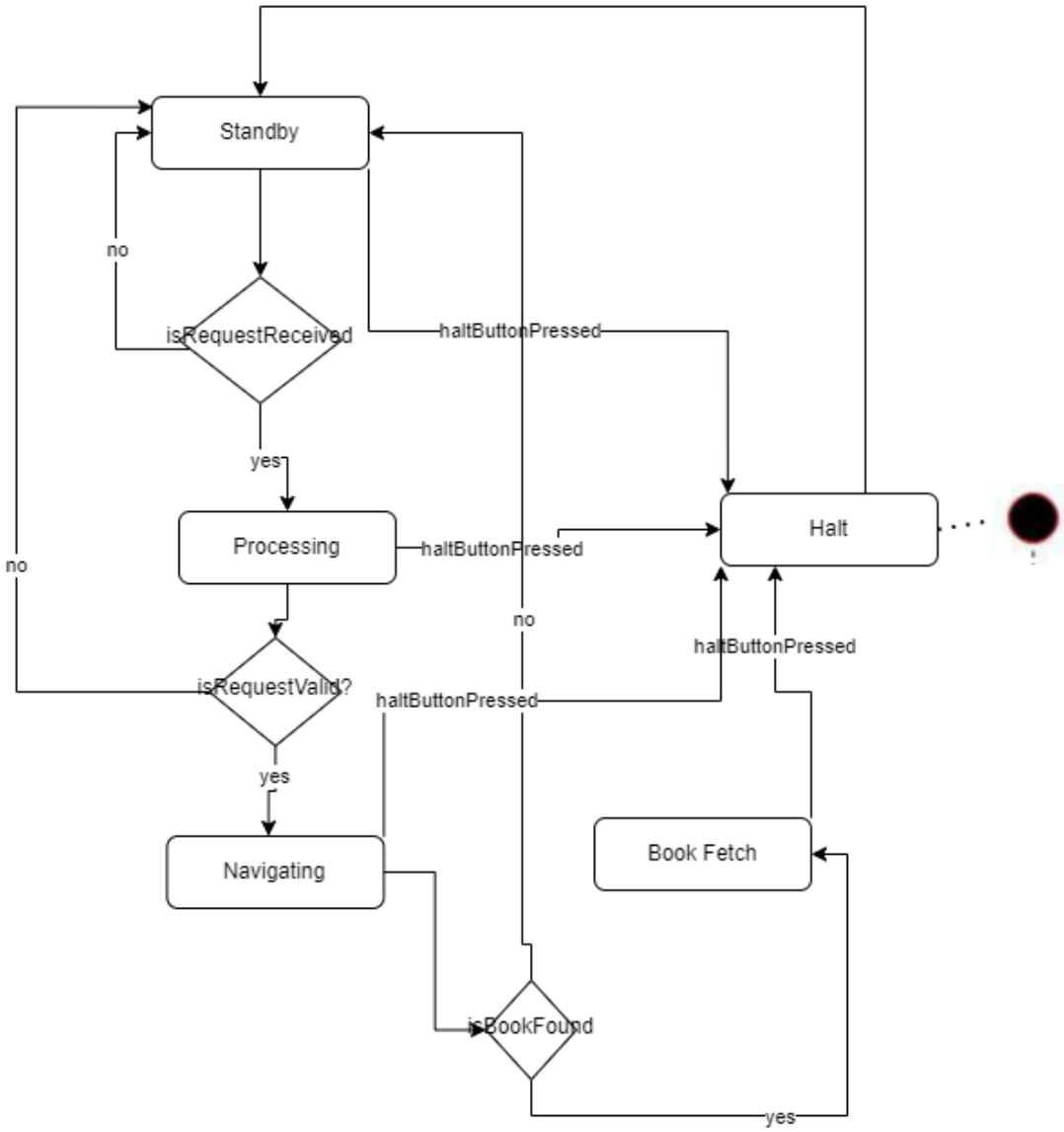
3.5.4.2 State Diagrams

3.5.3.2.1 Obstacle Detection



Obstacles should be detected by LibreBot. LibreBot tries to distinguish obstacles and behave accordingly.

3.5.3.2.2 Book Fetch



If a book fetch request is made. First, check the request. If it is valid, navigate to the bookshelf position and perform the book fetch operation. Then go back to the passive waiting state (standby).

4 Other Analysis Elements

4.1 Consideration of Various Factors in Engineering Design

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

4.1.1 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.

4.1.2. 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.

4.1.3. Public Welfare Considerations

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

4.1.4. 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.

4.1.5. Cultural Considerations

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

4.1.6. 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

4.1.7. 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.

4.1.8. 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.

4.2 Risks and Alternatives

LibreBot relies on text recognition and computer vision to detect books. There is a risk that the call number is not read fully or computer vision and text detection algorithms fail. In such cases, RFID technology will be used as a fallback plan.
Since there is a limited dataset that can be trained for the sake of this project, machine learning algorithms that will be used may not be as desired. In such cases, these parts will be hard coded.
The simulation tool, Drake, may not be capable of simulating the whole library or the LibreBot completely. In this scenario, the simulation will be performed in a less realistic environment.

Risk Name	Likelihood	Effect on the project	B Plan Summary
Computer vision and text detection fail	Medium	LibreBot fails to perform	Using RFID

ML algorithms work poorly	Medium	LibreBot works inefficient or sometimes fails to perform tasks	Hard coding
Drake simulation may not be enough for a realistic simulation	Medium	Applicability to the real life will be questioning	Creating less realistic simulation environment

Table 2: Risks

4.3 Project Plan

Table 3: List of work packages

WP#	Work package title	Leader	Members involved
WP1	CS 491 Reports	Muhammed	All Members
WP2	CS 492 Reports	Muhammed	All Members
WP3	CS 491 Demos	Giray	All Members
WP4	CS 492 Demos	Utku	All Members
WP5	CS 491 Problem Sets	Hikmet	All Members
WP6	CS 492 Problem Sets:	Yasir	All Members

Table 4: Detailed description of work packages

WP 1: CS 491 Reports			
Start date: *01.10.2022* **End date:** *13.11.2022*			
Leader:	*Muhammed*	**Members involved:**	All Members
Objectives: Delivering the project reports as a part of the CS491 Course			
Tasks: ***Task 1.1 Project Topic and Supervisor Selection*:** *To determine the project topic and the Supervisor. To prepare the Project information form for the submission.* ***Task 1.2 Project Specification Discussions*:** *To determine the scope of the project. Evaluate the risks and analyze market alternatives.To prepare the Project Specification document.* ***Task 1.3 Innovation Expert Meeting*:** *To present the project to an expert and receive feedback in regard to feasibility and innovativeness. To prepare the Innovation Expert Evaluation form.* ***Task 1.4 Analysis and Requirements Discussions*:** *To formalize the specifications of the project. To analyze market alternatives, and evaluate the risks and boundary conditions in detail.To prepare the Analysis and Requirements Report.*			
Deliverables ***D1.1:*** *Project Information form* ***D1.2:*** *Project Specification document* ***D1.3:*** *Innovation Expert Evaluation form* ***D1.4:*** *Analysis and Requirements Report*			
WP 2: CS 492 Reports			
Start date: *01.02.2023* **End date:** *mid-May 2022*			
Leader:	*Muhammed*	**Members involved:**	All Members
Objectives: Delivering the project reports as a part of the CS492 Course			
Tasks:			

Task 1.1 Detailed Design Discussion**:** *To formalize the project design in detail. To prepare Detailed Design Report.* ***Task 1.2 Final Report Discussion*****:** *To reflect on the final status of the project. To prepare a user's manual. To prepare the Final Report.*			
Deliverables ***D1.1:*** *Detailed Design Report* ***D1.2:*** *Final Report*			
WP 3: CS 491 Demos			
Start date: *01.11.2022* **End date:** *mid-December 2022*			
Leader:	*Giray*	**Members involved:**	All Members
Objectives: Demonstrating the project's progress to the supervisor, and the instructors of CS491 Course			
Tasks: ***Task 1.1 Pick and Place Demo*****:** *To prototype the robot model with the basic capability of picking a book from a cart and placing it on an empty shelf where the positions of all objects are assumed to be known.* ***Task 1.2 Preparation for Presentation and Prototype Demo*****:** *To prototype the robot model with the basic capability of recognizing the environment, locating the book and the shelfs to place the book on the shelf. The vision sensors will be usedTo*			
Deliverables ***D1.1:*** *The Jupyter Notebooks for Pick and Place Demo* ***D1.2:*** *The Jupyter Notebooks for Presentation and Prototype Demo*			
WP 4: CS 492 Demos			
Start date: *01.02.2023* **End date:** *mid-May 2023*			
Leader:	Utku	**Members involved:**	All Members
Objectives: Demonstrating the project's progress to the supervisor, and the instructors of CS491 Course			
Tasks: ***Task 1.1 Interim Progress Demo*****:** *To show the progress to the supervisor* ***Task 1.2 Preparation for Presentation and Demo*****:** *To finalize the project and prepare for the presentation and the demo.* ***Task 1.3 Preparation for CS Fair*****:** *To prepare for the CS Fair*			
Deliverables ***D1.1:*** *The Jupyter Notebooks for Presentation and Demo* ***D1.2:*** *The Slides for Presentation and Demo*			
WP 5: CS 491 Problem Sets			
Start date: *01.10.2022* **End date:** *December 2022*			
Leader:	Hikmet	**Members involved:**	All Members
Objectives: Learning the fundamental concepts for Robotic Manipulation tasks and gaining practice in Drake toolbox with PyDrake			
Tasks: ***Task 1.1 Study Pick-and-Pdlace of an Object with Full State Knowledge*****:** *To learn Kinematics and manipulation related formulations (geometric methods, physics, etc.) for the robot in the Drake toolbox.* ***Task 1.2 Study Geometric Perception*****:** *To learn Geometric Perception using vision sensors. To learn how robots perceive/see the environment with a camera in the Drake.*			
Deliverables			
WP 6: CS 492 Problem Sets			

Start date: *01.02.2023* **End date:** *May 2023*			
Leader:	Yasir	**Members involved:**	All Members
Objectives: Learning the state-of-art concepts and applications of machine learning in Robotic Manipulation and gaining practice in Drake toolbox with PyDrake			
Tasks: ***Task 1.1 Study Deep Perception and Motion planning.:*** *To learn Deep Perception and Motion planning.* ***Task 1.2 Study Applications of Machine Learning in Robotic Manipulation*:** *To learn how reinforcement learning can be used in robotic manipulation.*			
Deliverables			

Table 5: Gantt chart for the work packages

WP#	Work package title	Sept. 2022		Oct. 2022		Nov. 2022		Dec. 2022		Jan. 2023		Feb. 2023		Mar. 2023		Apr. 2023		May 2023	
WP1	CS 491 Reports																		
WP2	CS 492 Reports																		
WP3	CS 491 Demos																		
WP4	CS 492 Demos																		
WP5	CS 491 Problem Sets																		
WP6	CS 492 Problem Sets:																		

4.4 Ensuring Proper Teamwork

- Each member of the group is expected to attend weekly meetings.
- Each member of the group will take part in the project development lifecycle and decision-making process of the group.
- Each group member should research the topics related to his work. Group members are supposed to help each other and get help from the supervisor if they need it.
- The work division should consider the members' interests and should be made as just as possible.

4.5 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, preferably face-to-face group meetings will be scheduled with the team and the supervisor.

4.6 Planning for New Knowledge and Learning Strategies

For the project, we will model a robot for the use of libraries. None of the group members has previous experience or expertise in neither robot modeling nor in the libraries. So all group members need to learn the basics of robotic manipulation and the processes of libraries. We plan to follow the Robotic Manipulation course by Russ Tedrake [3] to understand the topics in robotic manipulation. By following the course we mean both studying the relevant chapters and solving the problem sets. Also note that a similar course is given by the Bilkent CS department with the name CS 449 Learning for Robotics by LiRA Lab. We will attend its tutorials and will be communicating with the members of the Lab. Both courses use a toolbox named Drake to simulate the robot [1]. We will also use this toolbox for our project. So we will also need to learn this toolbox. In addition to the these courses, we will also follow Drake's tutorials and examine its samples. Since it is software, naturally, we will use its API documentation frequently while learning the toolbox.

We plan to implement the project in Jupyter notebooks for python. We will use Google Colab platform for collaborating on writing notebooks. So we will also need to learn using the Deepnote. We also plan to use GitHub Projects to track project progress and manage task assignments.

We plan to use machine learning for task and motion planning. Although all of the team members have taken some introductory courses in machine learning, we still need to study its applications in robotics. For the text recognition of the book’s Call number from the camera input , our goal is to either train a model from scratch or to fine tune a state of art model for our purposes. The architecture of the model will be similar to the Mask R-CNN model , where a neural network that can process sequential data is preceded by a convolutional neural network. In our case, for the sequential neural network, we are planning to try different architectures as Long-Short Term Memory’s, Gated Recurrent Units and optimally the Transformers architecture. The second part where we use machine learning is the detection of an obstacle during the motion. We are planning to use a convolutional neural network as a classifier to determine whether there is an obstacle on the current path. Inferences will be made based on the image input from the camera. Another approach might be using an object detection model such as YOLO to detect whether an obstacle object is present in the image input.

We also need to research library classification systems, and investigate the needs of the libraries and what librarians might expect from the LibreBot. We plan to interview Bilkent University librarians to answer the previous questions and maybe obtain statistics on library usage (e.g. daily number of returned books, the distribution of the number of books borrowed/returned at once).

5 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.