Bilkent University
Department of Computer Engineering

Senior Design Project
T2323
GroceryBee

Analysis and Requirement Report

21901548, Efe Beydoğan, efe.beydogan@ug.bilkent.edu.tr
21903350, Arda Önal, arda.onal@ug.bilkent.edu.tr
21901645, Mert Barkın Er, barkin.er@ug.bilkent.edu.tr
21903100, Emir Melih Erdem,
melih.erdem@ug.bilkent.edu.tr
21902615, Eren Polat, eren.polat@ug.bilkent.edu.tr
Supervisor:
Özgür S. Öğüz
Course Instructors:
Selim Aksoy
Tağmaç Topal
Erhan Dolak

13.11.2022

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

Contents

Analysis and Requirement Report

Project Short-Name: GroceryBee

1 Introduction

Online grocery shopping has been offered by numerous markets for years now, however, especially with the advent of the COVID virus and the following quarantine period, there has been a considerable surge in the number of online grocery orders. Specifically, Turkey's online market grew by 434% and reached $244.9 million in the first half of 2020 [1], during the time when COVID first became a part of our lives. The supermarket chain "Migros" was already offering an online shopping option, and with the popularity of online grocery shopping booming, new companies have also started investing in online grocery shopping options for their platforms, such as Morhipo and Trendyol. It was reported that the growth in fast delivery startups, such as Getir and Banabi, exceeded 50% and the provinces Migros delivers to have gone from 58 to practically all provinces in Turkey. During the pandemic, the demand for virtual markets almost quadrupled [1]. In the US, 70% of shoppers stated they would be buying groceries online by 2022, and Walmart's US e-commerce sales grew by 79% in the third quarter of 2020. It is estimated that e-commerce warehouses will be fully automated by the year 2029 [2].

It is apparent from the aforementioned information that demand for online grocery shopping has skyrocketed in the past couple of years, and investing in this field is surely a worthwhile engagement. To this end, we will be developing a robot, named "GroceryBee", to automate and expedite the task of gathering grocery items from the shelves of supermarkets for online orders. GroceryBee is a robot designed to assist grocery stores in keeping up with their online orders. GroceryBee will help grocery store employees by collecting the items which are required for an online order from the store shelves, thus speeding up the delivery process. Additionally, GroceryBee will restock shelves that are empty and keep updated stock information in order to inform customers on whether a product is available in the store or not. Consequently, supermarket employees will have received an assist in their jobs, and they will be able to work simultaneously with GroceryBee to serve customers. There are similar projects for accomplishing tasks akin to what we are trying to achieve. However, GroceryBee will not only safely and accurately gather grocery items, but it will at the same time help grocery store employees with keeping the markets in order. Moreover, although there are robots that are engineered to accomplish tasks such as managing inventory or notifying employees about shelves that need to be restocked [3], none of these robots are designed to do all of these, and even more, altogether, unlike GroceryBee.

In the following parts of this report, first, the proposed system will be introduced. After an overview of the system, functional, non-functional, and pseudo requirements will be discussed. Afterward, system models, such as the use case diagram of the system, object model, and dynamic models, will be given. Finally, other analysis elements, including consideration of various factors in design, risks and alternatives, project plan, and ethical and professional responsibilities, will be presented.

2 Proposed System

2.1 Overview

For the automation and acceleration of gathering grocery items from the shelves of supermarkets for online orders, we will be developing our robot “Grocery Bee”. For this, we aim to leverage theories from different fields in science and engineering. GroceryBee will have to make decisions in an uncertain environment, and in this uncertainty, it is not possible to plan everything precisely beforehand, therefore our robot will exploit the real-time information from its camera (both RGB and depth) to make real-time decisions and changes in planning (a robot might see a person blocking its path, and change its path to the destination, etc.). For perception, connectivist approaches which have achieved state-of-the-art results will be employed. Therefore, neural models will be used and perhaps trained according to our purposes; both for the detection of the product to be handled and for sequential decision-making. For example, the robot might have to analyze the situation of the item and check if it is stuck, or has other items around it. Deep learning approaches will pave the way for the robot to assess the situation and segment the item of interest from a window of frames. Different architectures have been shown to achieve SOTA results (such as transformers) in generative and understanding tasks regarding computer vision. Robot actuators and grippers apply control theory in motion/trajectory planning and as mentioned above to leverage the sequential information gathered from the feedback system. This is important for the robot to execute sequential decision-making and be responsive to its environment. We will also use simulations (Drake) that use control theoretic design and analysis to enhance the robustness of the physics engine. Open problems in motion planning, perception, and decision-making will be worked on using these approaches from various disciplines.

2.2 Functional Requirements

- GroceryBee should be able to collect required items in response to the orders.
- GroceryBee should be able to navigate in the supermarket aisles without crashing.
- GroceryBee should have access to locations of products within aisles and navigate to the location when an order is requested.
- GroceryBee should be able to detect the requested object on the shelf from the camera frames, identify relative 3D coordinates, use the gripper to pick up the object safely, and bring the object to the delivery point without damaging the item.
- GroceryBee should be able to calculate the most optimum path for requests with multiple items and collect them accordingly.
- GroceryBee should inform the employees about stock information and alert them if stock is low when it arrives on the shelf.
- GroceryBee should be able to restock the shelves in the supermarket.
- GroceryBee should be simulated in a simulation rather than a physical robot.

2.3 Non-functional Requirements

Usability

The application should require only the information of the items such as the images, names, and locations. GroceryBee should be able to collect the requested items and bring them to the delivery point within the supermarket.

Efficiency

When multiple items are requested in the same order, GroceryBee should calculate the shortest path and collect the orders to save time, energy, and money.

When GroceryBee arrives at the aisle to collect the item, it should calculate the most optimum way for its arm to move to grab the item.

Extensibility

GroceryBee should be designed so it could be operated in environments with multiple robots which could communicate with each other. In the future, if a system with multiple robots is introduced, GroceryBee should easily be converted to a multi-robot application.

In the future, it could also work in environments that have humans.

Reliability

GroceryBee should collect, carry and deliver items without damaging them. When an order is requested, GroceryBee should ensure successful delivery.

Data should be processed securely to ensure that it is protected and not leaked to third parties.

2.4 Pseudo Requirements

1. Drake is a module that is available in C++ and Python, but Python will be used for implementation.
2. Github will be used for version control.
3. scikit-learn, PyTorch, pandas, NumPy, and matplotlib modules will be used for perception purposes.
4. The robot will be free for the users, but will be paid for customers:
5. The website of GroceryBee will be hosted on GitHub.
6. Visual Studio Code will be our IDE.

2.5 System Models

2.5.1 Use-Case Model

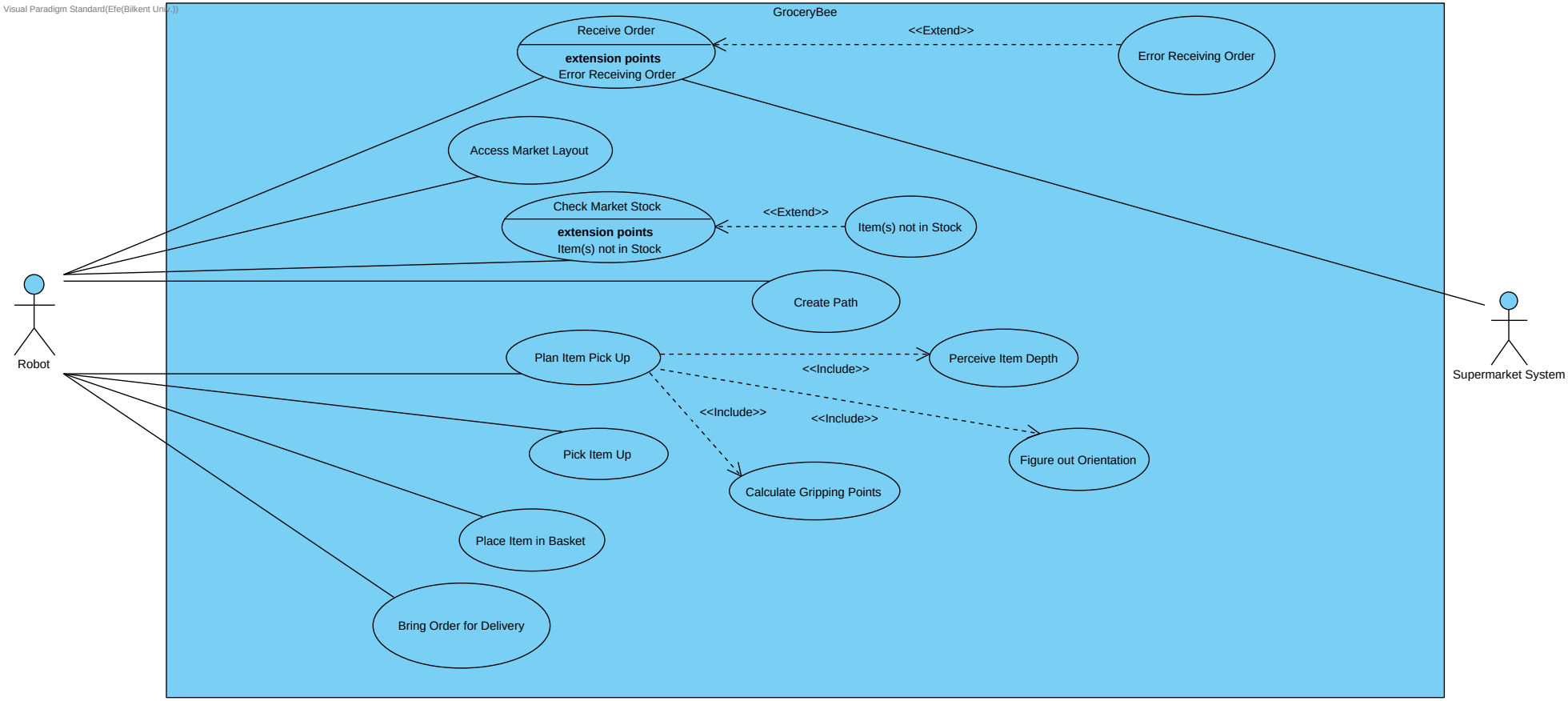

Figure 1: Use-Case Model

2.5.2 Use-Case Scenarios

Use Case: Receive Order

Participating Actor(s): Robot, Supermarket System

Entry Conditions:

- Customer places an order on the supermarket system
- Robot is not busy with another task

Exit Conditions: Robot receives the order through the system

The Flow of Events:

- A customer uses the supermarket system to place an order
- The order is relayed to the robot system
- The robot receives the order
- Robot initiates the actions to collect the required items

Use Case: Error Receiving Order

Participating Actor(s): Robot

Entry Conditions: This use case extends the “Receive Order” use case. It is initiated by the robot system whenever an error occurs during the communication of an order to the robot through the supermarket system.

Exit Conditions: The admins are notified or the order is successfully received by the robot.

The Flow of Events:

- Robot faces an error when trying to receive an order.
- Robot system notifies the administrators about the issue.

Use Case: Access Market Layout

Participating Actor(s): Robot

Entry Conditions: The robot successfully receives an order.

Exit Conditions: The robot accesses the market layout.

The Flow of Events:

- Robot accesses the market layout which is preloaded in its memory.

Use Case: Check Market Stock

Participating Actor(s): Robot

Entry Conditions: The robot successfully receives an order.

Exit Conditions: The robot checks the market stock to determine the availability of required items for the order.

The Flow of Events:

- Robot accesses the market stock.
- Robot compares the market stock against the order list to see if all items are available.

Use Case: Item(s) not in Stock

Participating Actor(s): Robot

Entry Conditions: This use case extends the “Check Market Stock” use case. It is initiated by the robot system whenever the robot determines that an item in the current order list is not in the market stock.

Exit Conditions: The market employees are notified about the situation.

The Flow of Events:

- Robot finds out an item in the order is out of stock.
- Robot lets the employees know about the situation and registers the problem so the shopper can also be notified.

Use Case: Create Path

Participating Actor(s): Robot

Entry Conditions:

- Robot successfully receives an order.
- Robot accesses the market layout.
- Robot determines which items are in stock and can be collected.

Exit Conditions: Robot plans the quickest path to collect all of the items.

The Flow of Events:

- Robot utilizes the market layout to determine the most efficient and quick path to collect all of the available items in the delivery list.

Use Case: Plan Item Pick Up

Participating Actor(s): Robot

Entry Conditions: The robot reaches an item in the list.

Exit Conditions: The robot successfully determines a way to pick the item up.

The Flow of Events:

- Robot successfully locates and approaches an item in the list.
- This use case includes the “Perceive Item Depth”, “Figure out Orientation” and “Calculate Gripping Points” use cases. All of these use cases are executed one by one to complete the item pick-up planning process.
- Robot completes planning the item pick up.

Use Case: Perceive Item Depth

Participating Actor(s): Robot

Entry Conditions: Robot approaches an item

Exit Conditions: Robot identifies the depth of the item using its camera

The Flow of Events:

- The robot approaches the item next on the list.
- Using its camera, the software calculates the depth and the position of the item.

Use Case: Figure out Orientation

Participating Actor(s): Robot

Entry Conditions: Robot perceives an item’s depth

Exit Conditions: Robot calculates the orientation of the item

The Flow of Events:

- The robot calculates and stores the orientation of the item on the list using its camera.

Use Case: Calculate Gripping Points

Participating Actor(s): Robot

Entry Conditions: Robot calculates the orientation of the item

Exit Conditions: Robot calculates the gripping points of the object

The Flow of Events:

- Using the orientation and the depth of the item calculated in the steps before, the robot calculates the optimal gripping points for the item.

Use Case: Pick Item Up

Participating Actor(s): Robot

Entry Conditions: Robot plans picking up the item

Exit Conditions: Robot picks the item up

The Flow of Events:

- Using the optimal gripping points for the item, the robot picks up the item.

Use Case: Place Item in Basket

Participating Actor(s): Robot

Entry Conditions: Robot picks the item up

Exit Conditions: Robot places the item in the basket

The Flow of Events:

- After picking up the item, the robot carefully places the picked up item in the basket such that the placed item will not fall down.

Use Case: Bring Order for Delivery

Participating Actor(s): Robot

Entry Conditions: Robot places all the necessary items in the basket

Exit Conditions: Robot brings the order for delivery.

The Flow of Events:

- After picking up all the items on an order, the robot goes back to its starting position to deliver the order.

2.5.3 Object and Class Model

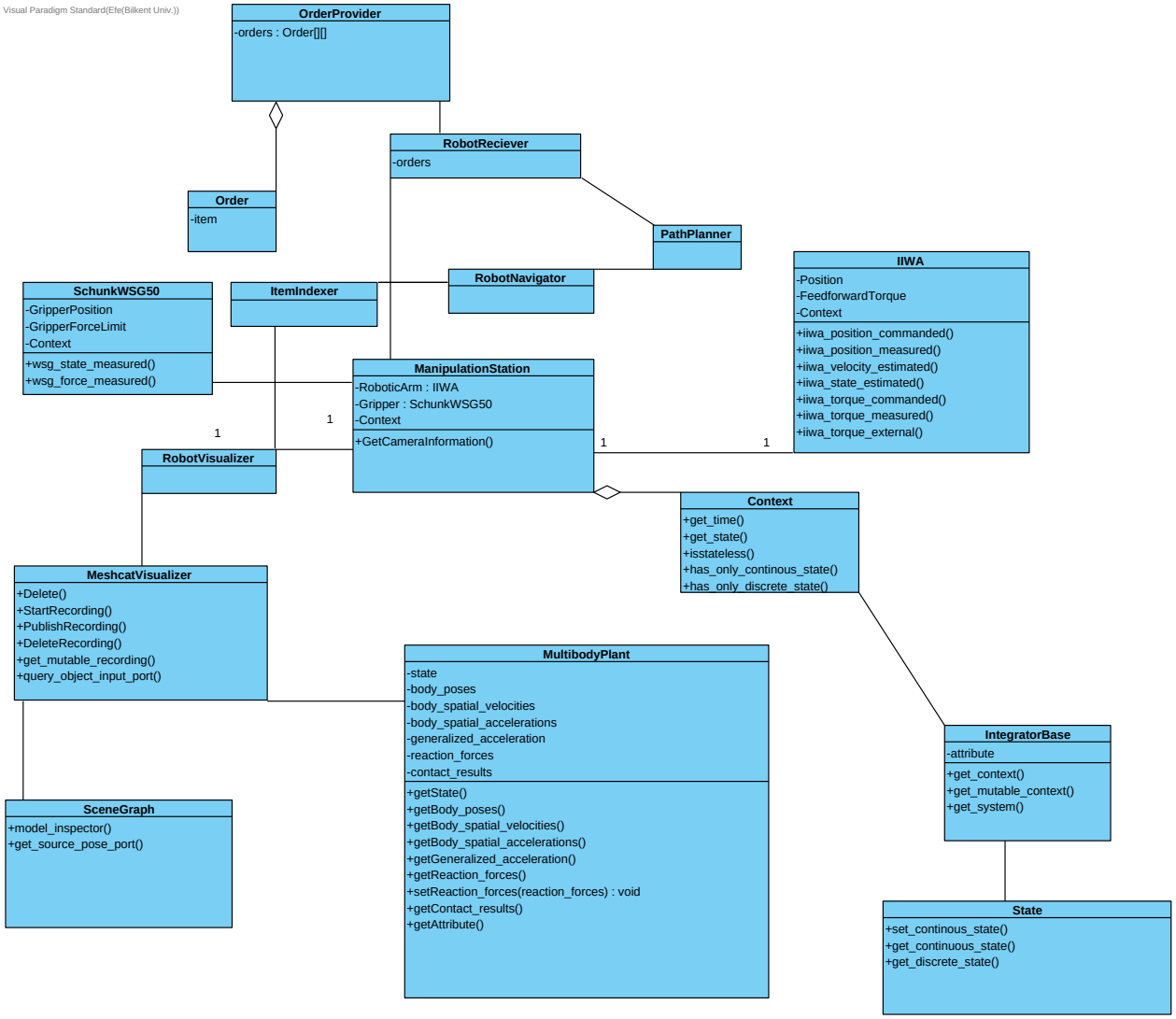

Figure 2: Object and Class Model

Order: Class representing a specific order by the customer, involving items

ItemIndexer: Responsible for indexing the order to specific locations where items reside.

OrderProvider: The linker class connecting the order by the customer to the robot-market infrastructure.

RobotReciever: Class that receives the order, processes it by communicating with the navigator, and plans the path.

PathPlanner: Responsible for finding an optimal path around multiple ordered items to stop by. This may use external modules for mathematical optimization, in a combinatorial setting with the usage of heuristics or leveraging the real-time perception of the market.

RobotNavigator: Navigating the robot with respect to the orders of the path planner, but may contribute to the decision-making process by feedback from the real-time action (e.g. if a road is blocked, it may recognize it with the camera of the robot and recommend the path planner to change direction).

ManipulationStation: "A system that represents the complete manipulation station, including exactly one robotic arm (a Kuka IIWA LWR), one gripper (a Schunk WSG 50), and anything a user might want to load into the model". [4]

IIWA: "Wires up Drake systems between an LCM interface and the actuation input ports of a MultibodyPlant. This simulates the role that driver software and control cabinets would take in real life." [5]

Context: "Context is an abstract class template that represents all the typed values that are used in a System's computations: time, numeric-valued input ports, numerical state, and numerical parameters." [6]

MultibodyPlant: "MultibodyPlant is a Drake system framework representation for the model of a physical system consisting of a collection of interconnected bodies." [7]

IntegratorBase: "An abstract class for an integrator for ODEs and DAEs as represented by a Drake System." [8]

State: "State is a container for all the data comprising the complete state of a particular System at a particular moment." [9]

SchunkWS50: Gripper class to handle the objects in the simulation.

MeshcatVisualizer: "Provides an interface to Meshcat (https://github.com/rdeits/meshcat)" [10] to visualize the Meshcat scene.

SceneGraph: "Creates an instance of PlanarSceneGraphVisualizer, adds it to the diagram, and wires the scene_graph pose bundle output port to the input port of the visualizer." [11]

2.5.4 Models

2.5.4.1 Activity Diagram

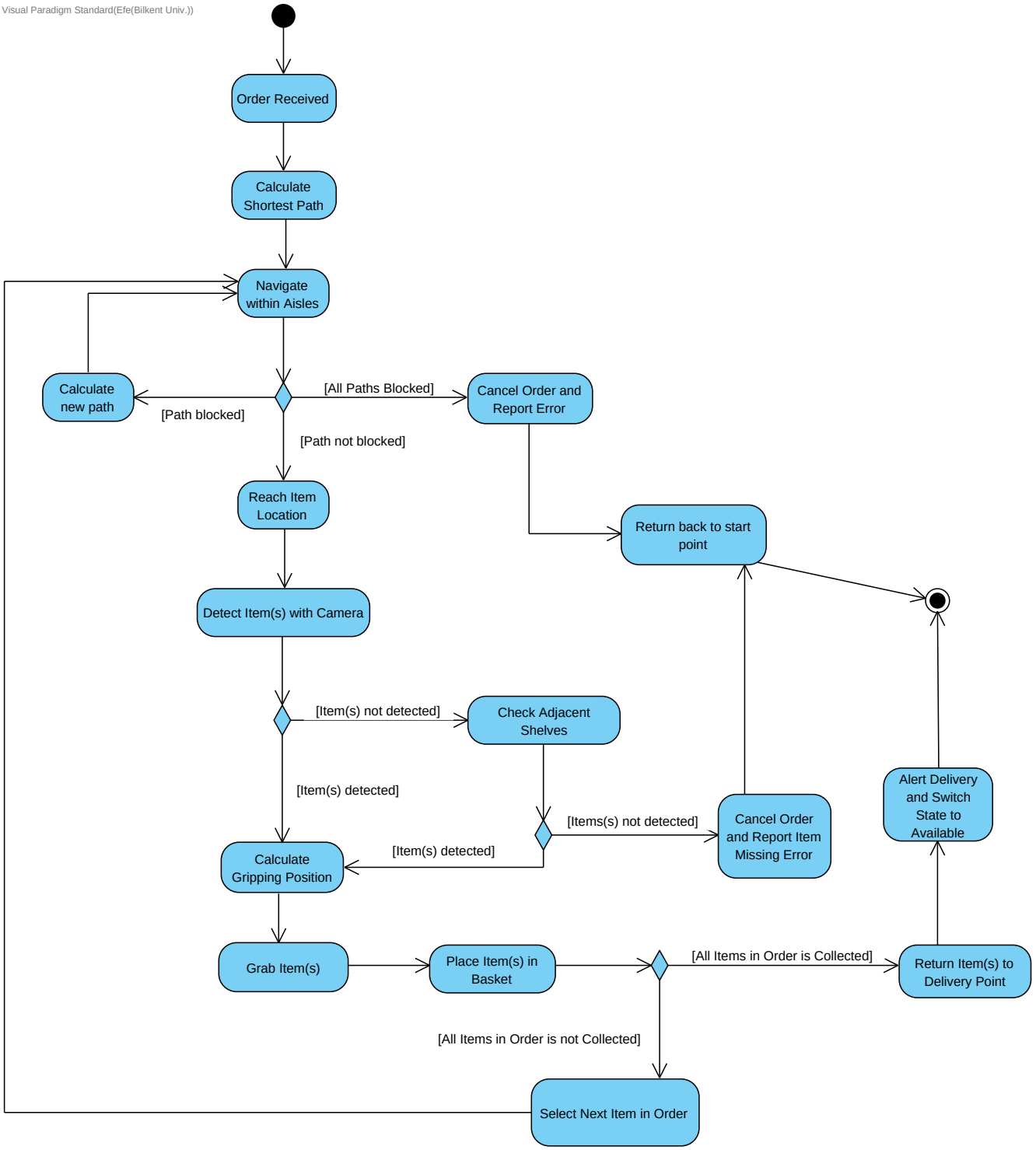

Figure 3: Activity Diagram of item collection and delivery of GroceryBee

The previous figure (Figure 3) represents the activity diagram of GroceryBee. The robot enters the activity diagram as it receives an order. The robot will then find the appropriate (possibly shortest) path to the ordered item and it will navigate itself within the aisles. Depending on whether the path is blocked (by another robot, human, etc.) or not, it will decide to continue on that path to reach the item location; or calculate a new path. If it assesses that all paths are blocked, it will cancel the order and report the error. Afterward, it will return back to its starting point to wait for an order. After finding the proper path, it will reach where the item is and detect the item with its camera. If it doesn't find it, it will look for it on the adjacent shelves. If it still can't find it, it will again return back to its starting point by reporting an error. If the robot determines where the item is, it will grab the item and put it in the basket to deliver it to the delivery point if no more items are left. If all items in the order are not collected, it will select the next item in the order and repeat the activity all over again.

2.5.4.2 State Diagram

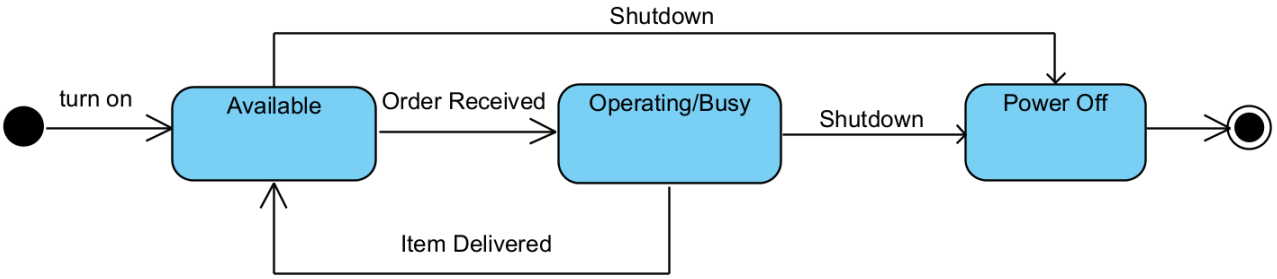

Figure 4: State Diagram of GroceryBee

The above figure is the state diagram of GroceryBee. It showcases the possible states of the robot in which it can be. When the robot is turned on, it is immediately available which means it can accept orders so it can collect and deliver the items. When an order is received when it is available, it is immediately accepted, and the state switches to an operating/busy state. During this state, the robot navigates through the aisles of the supermarket and goes to the correct location which the is located, detects the real location of the item through its camera and grabs it, and navigates back to the delivery point with the collected item. After the item is delivered, the robot switches back to the available state. At any point, the robot could be shut down through a system interrupt which puts the robot in a power-off state.

2.5.5 User Interface

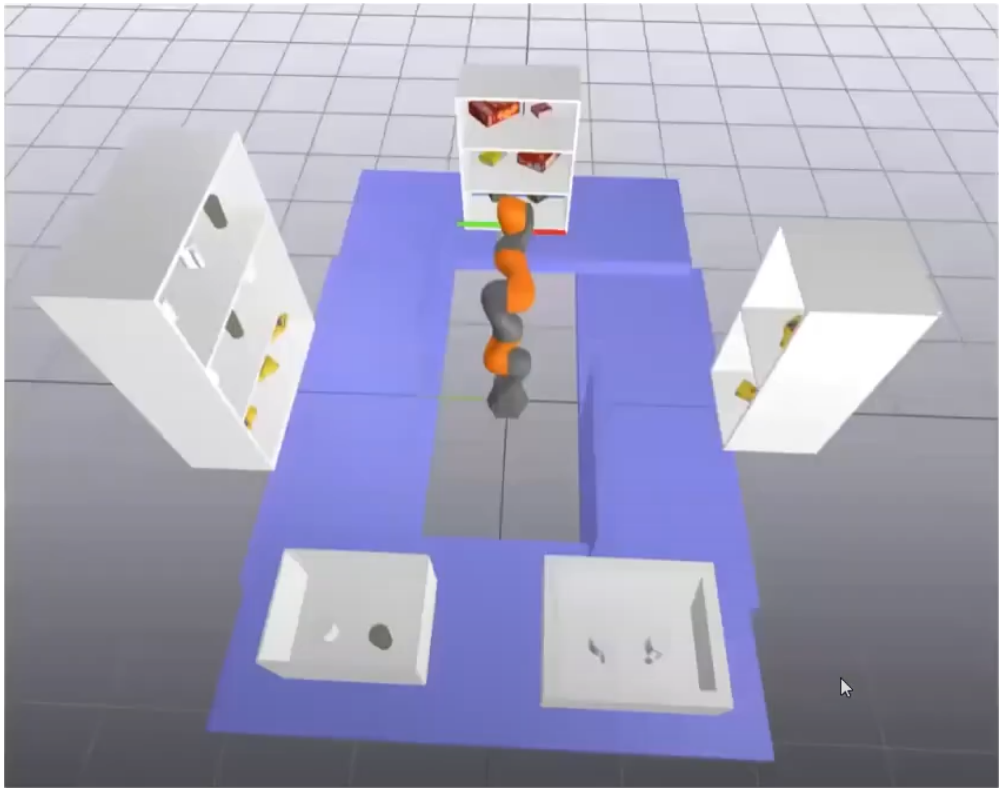

Figure 5.1: Example Simulated Supermarket Setting 1 [12]

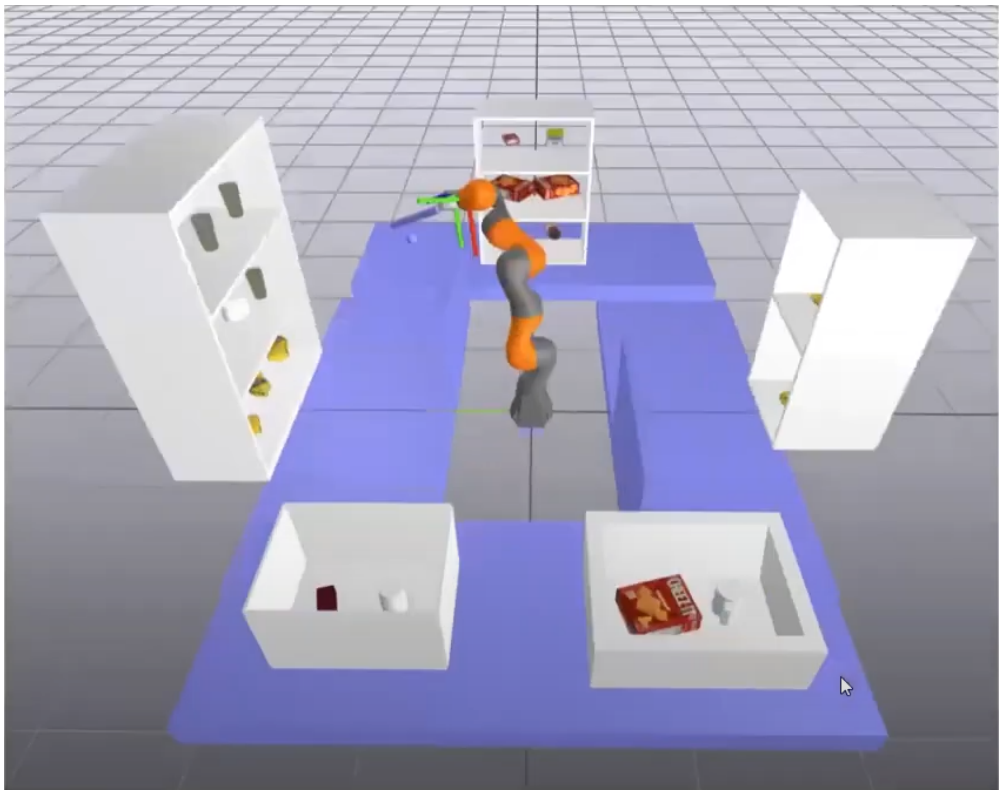

Figure 5.2: Example Simulated Supermarket Setting 2 [12]

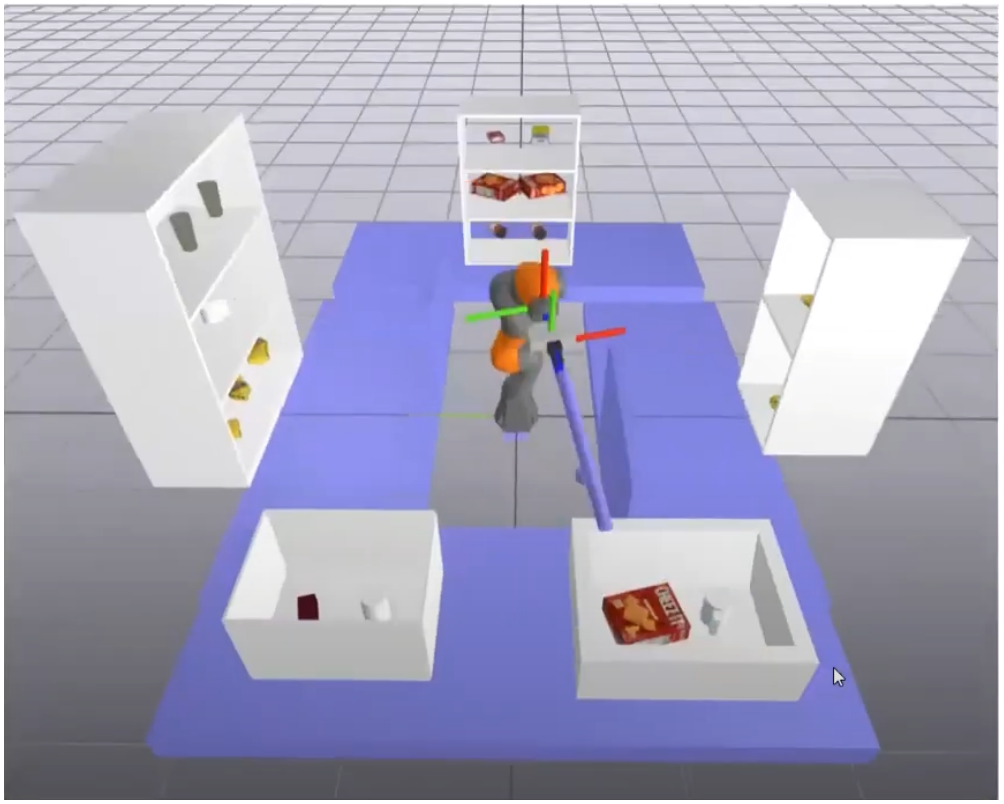

Figure 5.3: Example Simulated Supermarket Setting 3 [12]

3 Other Analysis Elements

3.1 Consideration of Various Factors in Engineering Design

Here the constraints on the project (i.e implementation, design, time, language) and the needs of various factors like health, safety, environmental, and economic factors will be discussed. The section will mention how these factors have affected the analysis process and will affect the project in the future.

Implementation

- The robot will be developed using Drake—a simulation toolbox developed at MIT Computer Science and Artificial Intelligence Lab (CSAIL) that enables model-based design and verification for robotics. The perception of the robot will be built in Drake.
- Python will be the main language for developing robotic skills, through Drake's Python integration.
- OpenGL API will be used to render 2D and 3D vector graphics.
- MeshCat will be used to remotely run and view 3D simulations on the web.
- For computer vision, learning will be enabled by the PyTorch framework while the Open3D library will be used to facilitate geometric queries.
- Libraries such as IPOPT, SNOPT, Gurobi, and MOSEK will be utilized as optimizers.

Design

- The robot cannot be too wide for it to fit in between the shelves in the supermarkets.
- The robot must be able to move between shelves without the need for external support.

Time

- The reports and demos as part of the course CS-491 will be prepared and handed in before the deadlines stated in the course requirements.
- The robot will be virtually implemented before the project delivery deadline and ready to be showcased at CSFair in Spring 2023.

Language

- The robot will only recognize item names in English.
- The shopping list will be provided only in English.

Economic Factors

- Building a physical robot or buying a premade one would cost well over the budget of this project and are out of the project's main focus. For that reason, the robot will only be built virtually through simulations since the gap between simulation and reality is sufficiently small in robotics nowadays.
- The Drake simulator and other libraries and APIs are free to use, so will not affect the project's budget.

Health and Safety Factors

Health and safety of the customers and employees in a supermarket environment is a concern that affects the project's design.

- The robot should avoid any kind of collision, especially with humans, for the safety of the customers and employees around.
- The robot cannot carry out sudden, unexpected maneuvers since that would threaten the individuals nearby.
- The robot will not have pointy, sharp edges or parts made of harmful materials that may cause harm when in touch with human skin.

Global Factors

Since there are no global standards in supermarket plans, it is discussed that the project has to adapt to different supermarket environments and plans. To develop and demonstrate the project, an example virtual setting will be designed in MeshCat and used throughout the stages. However, the market plan will be changeable and the robot should be able to function in different environments.

Environmental Factors

Sustainability is a topic that is getting more and more prominent globally. With correct engineering practices that account for environmental factors, it is possible to sustain the engineering processes without harming the environment and wasting excessive amounts of energy. Sustainability in robotics is therefore gaining importance too. So, in the analysis, we had to consider how environmental factors may affect our design.

- Firstly, the energy consumption of the robots will be reduced by optimizing the paths the robots will take [13].
- Next, the use of unsustainable materials in the production of the robot can be avoided, with the exception of sensors, computer chips, etc. [14], [15].
- Lastly, to ensure sustainable development, no prototypes or physical components will be built. All testing and development will be carried out through virtual simulations.

Social Factors

Since replacing human supermarket employees would be highly unethical and harmful to the social order, robots should support the employees instead of replacing them. That brings the necessity of a human-robot interaction aspect to the project. The robots should be able to collaborate well with the employees, and assist them whenever possible.

3.2 Risks and Alternatives

Time may not be estimated perfectly, or in other words, time allocated for learning the robotics tools may exceed the schedule and leave less space for the implementation of the project. Therefore, the project has its priorities set in terms of functionalities, which will be mentioned in the next section.

The aspects of the project (i.e. perception algorithms, simulation technologies) that are initially considered to be premade by others and therefore neglected in the task allocation may in fact require extra work and time to be built. These concerns will be clarified as we learn the necessary tools like Drake, and the assistance provided by them.

Wrong or unrealistic estimations of what a robot can achieve within the boundaries of modern technology and tools available to us are also one of the risks that may leave some of the features hanging. For instance, we have observed from our competitors that modern robotics technology doesn't allow robots to function at high speeds, especially when they have to navigate in an unstructured environment and perform actions. Therefore, we cannot expect our robot to be extremely swift at collecting items from the shelves.

- First specific risk is that detecting items, picking them up correctly, and placing them into the basket may already be a difficult and time-consuming task to complete since these processes involve complex perception algorithms to obtain successful results. In such a case, an alternative is to set aside the mobility of the robot within the scope of this project and focus on how to improve the pick and place process. As a temporary solution, that may result in a stable robot arm between the shelves, which correctly identifies supermarket products, grasps them properly based on their physical properties, and successfully carries them.

- Next, an initial idea was to build a multi-robot system, where multiple Bees in the same environment interact with each other and the task allocation is done collectively. This would require the pathfinding algorithms to be enhanced by accounting for multiple agents and optimizing the total outcome. Moreover, the robots would have to avoid any collision with each other while working together. However, this may be a more challenging task than we estimate, and due to the risks related to time, a safe alternative is to go with one robot within the context of this project.

- We have also considered a human interaction aspect for the robot, where it would inform customers and employees on whether an item is still in stock, and if it is, where it could be found. However, considering the complexity of this task, it is risky to include this function initially as part of the senior design project. An alternative to this is to store the item stock information within the robot's system, but leave the human interaction aspect as a possible extension of the GroceryBee in the future.

- Learning will be used to enhance the perception of the robot, and help it better understand the environment and detect the objects properly. However, this keeping the accuracy high and above a certain threshold may be difficult, especially considering the sensory noise and object-specific properties. Therefore, if we fail to have a proper detection model, alternatively, the objects used in the simulations may be simplified and typified to ease the perception. For instance, cubic objects can be used to have a more successful simulation.

3.3 Project Plan

Table 1: Factors that can affect analysis and design.

	Effect level (out of 10)	Effect
Economic factors	10	The robot will be simulated in a virtual environment instead of building a physical robot.
Public health & safety	8	The robot's movements should be programmed to be limited and controlled, considering the safety of humans around the robot in the supermarket.
Environmental factors	5	Energy consumption of the robot should be minimized and sustainable engineering practices should be considered during development.
Social factors	5	The robot should collaborate well with human employees.

Table 2: Risks

	Likelihood (out of 10)	Effect on the project	B Plan Summary
Risk 1: Pick and Place Operation is More Challenging Than Estimated	4	Focus more and spend more time on improving the detection, pick and place processes.	Set aside the mobility of the robot, and build a stable arm instead, focusing more on other operations.
Risk 2: Failing to Build an Efficient Multi-robot System	7	Have an inefficient system of multiple robots that cannot collaborate well with each other.	Focus on building one single robot that operates alone instead of optimizing the pathfinding algorithms for multiple agents.
Risk 3: Human Interaction Aspect Limited due to Time Constraints	9	Build a half-completed human interaction UI and system for the robot.	Instead of working on the human-interaction aspect, complete the stock management system first.
Risk 4: Inaccuracy in Learning Models	3	The robot may fail to detect and grab the objects properly.	The objects used may be simplified to ease the perception, i.e cubic objects may be used.

Below is the intended plan that will be followed during the development of GroceryBee.

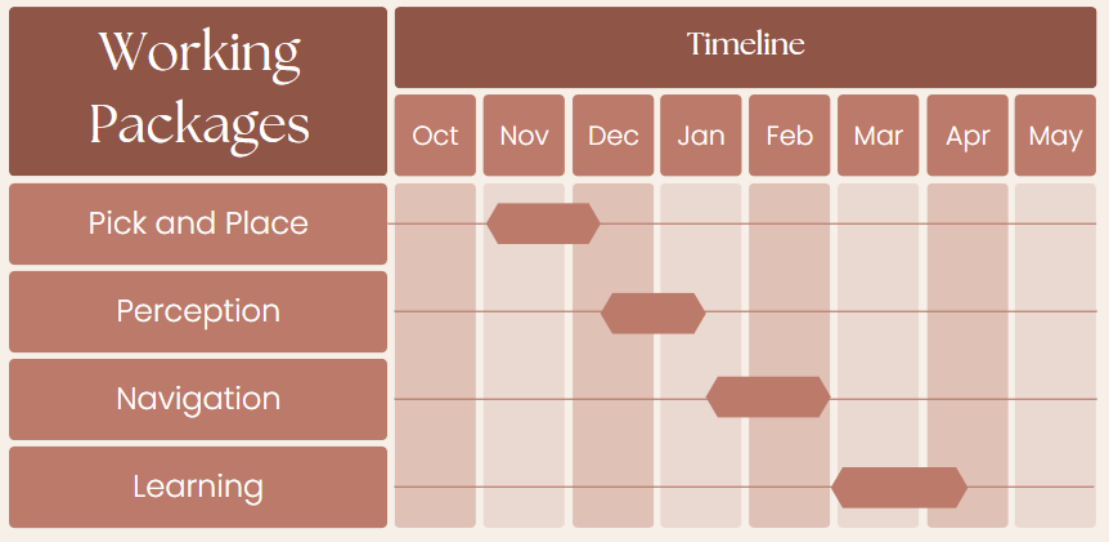

Figure 5: GroceryBee Development Gantt Chart

Table 3: List of work packages

WP#	Work package title	Leader	Members involved
WP1	Pick and Place	Efe	Emir, Mert, Eren, Arda
WP2	Perception	Arda	Eren, Emir, Mert, Efe
WP3	Navigation	Emir	Arda, Efe, Mert, Eren
WP4	Learning	Eren	Mert, Efe, Arda, Emir

Table 4: Pick and Place work package details

WP 1: *Pick and Place*			
Start date: *15/11/2022* **End date:** *15/12/2022*			
Leader:	*Efe Beydoğan*	**Members involved:**	*Arda Önal* *Eren Polat* *Efe Beydoğan* *Emir Melih Erdem* *Mert Barkın Er*
Objectives: *The main objective of this work package is to implement the main functionality of picking up the object with known coordinates and allowing the robot to pick and place objects.*			
Tasks: ***Task 1.1 Robot structure*:** *The kind of robot and robot arms that will be used should be designed and implemented.* ***Task 1.2 Forward Kinematics*:** *When the robot arm wants to grab an object, the joint angles, the velocity, and other parameters should be properly set so the robot can reach and its gripper could be aligned with the object. This task will implement the mathematical calculation required to pick an object.* ***Task 1.3 Gripper*:** *When an object is tried to be gripped, its gripping positions, the amount of force, etc. should be calculated. In this section, a control theory problem could be implemented by adding sensors to the point of contact of the grippers to implement a gripper similar to the human hand that senses the hardness from the pressure measured and applies the required force which wouldn't break the object.*			
Deliverables ***D1.1:*** *Designing and setting up the robot* ***D1.2:*** *Implementing the robot's picking and placing*			

Table 5: Perception work package details

WP 2: *Perception*			
Start date: *15/12/2022* **End date:** *15/02/2022*			
Leader:	*Arda Önal*	**Members involved:**	*Arda Önal* *Eren Polat* *Efe Beydoğan* *Emir Melih Erdem* *Mert Barkın Er*
Objectives: *The first part of pick and place is done with known object coordinates. The purpose of this part is to find the object through frames obtained from the depth camera which will be mounted on top of the object. The object's 3D coordinates relative to the robot will be detected, calculated, and fed into the pick and place functions so the robot can detect and pick locations of unknown objects.*			
Tasks: ***Task 2.1 Object detection:*** *The depth image that comes from the depth camera gives blobs that will be used to classify what could be the object in the given camera frame. From these blobs, the pixel values corresponding to the object desired will be calculated and therefore the borders of the object will be known in the 2D image of the camera.* ***Task 2.2 Point cloud generation*****:** *With point cloud generation techniques, the relative 3D coordinates of all the pixels corresponding to the object will be calculated. By doing this, we will know the 3D shape of the object that will be used for calculating the correct gripping position. See Figure 6 for an example of a point cloud image of a bunny rabbit.* ***Task 2.3 Grasp selection:*** *This task is for identifying which point will be the point of gripping from the given point cloud so the robot can pick up the object correctly. It will have subtasks such as point cloud pre-processing, estimating normals and local curvature (see Figure 7), evaluating a candidate grasp, and generating grasp candidates. See Figure 8 for a better understanding of the problem which will be solved with this task.*			
Deliverables ***D2.1:*** *Integration of pick and place with the camera mounted on top of the robot to extend the pick and place functionality to real-life situations by completing all the tasks mentioned above.*			

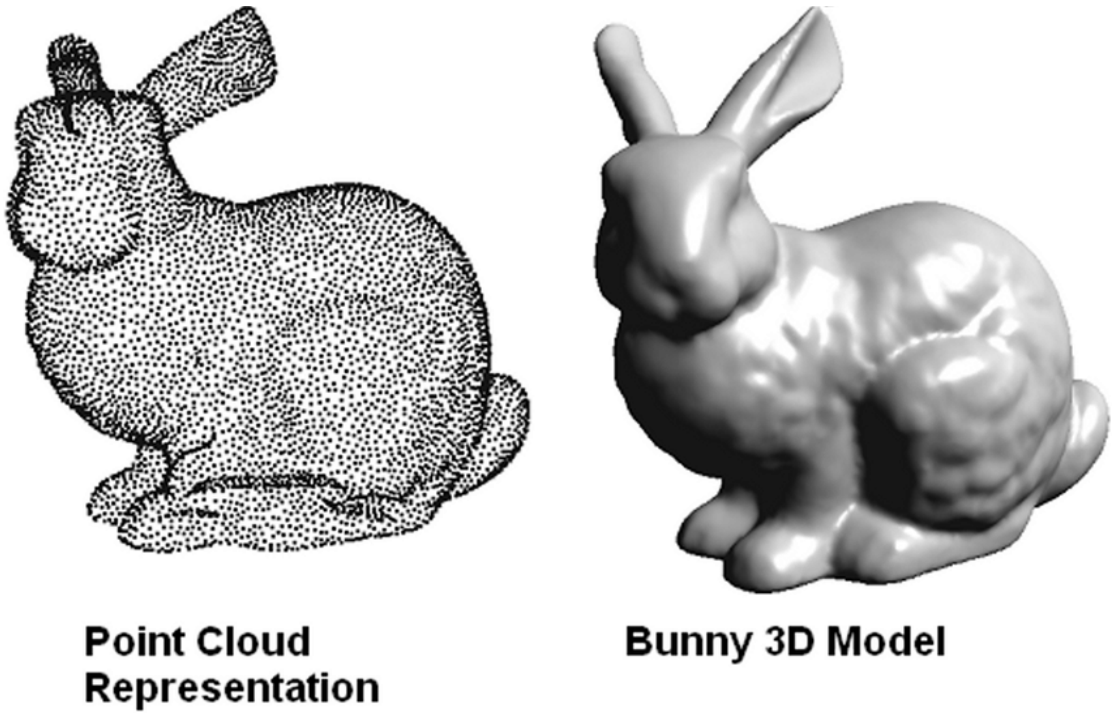

Figure 6: Point Cloud Representation of Stanford Bunny Model [16].

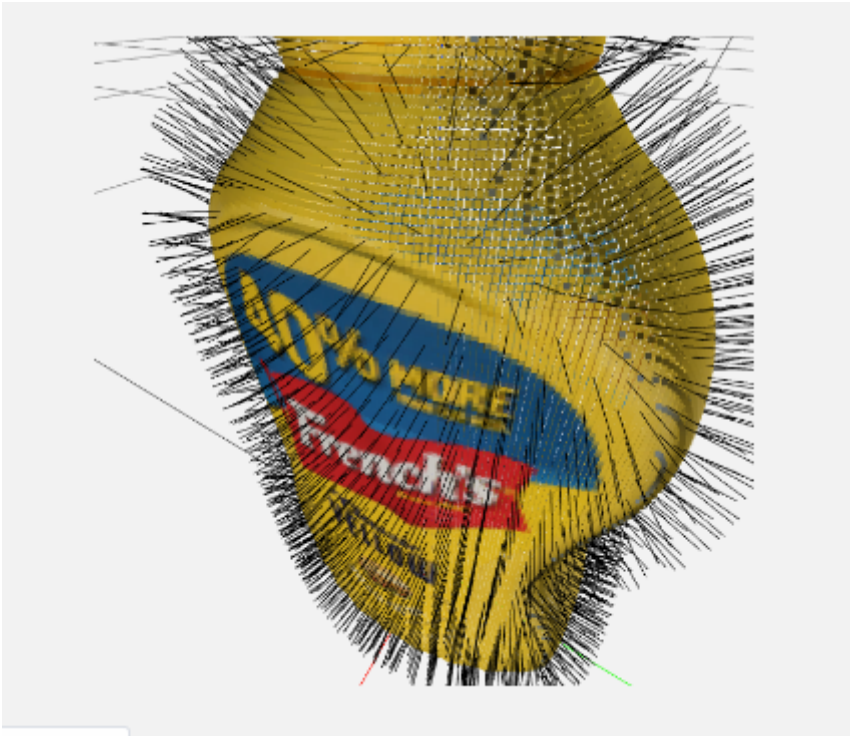

Figure 7: Normal vectors of a sample object on each point of the point cloud [17].

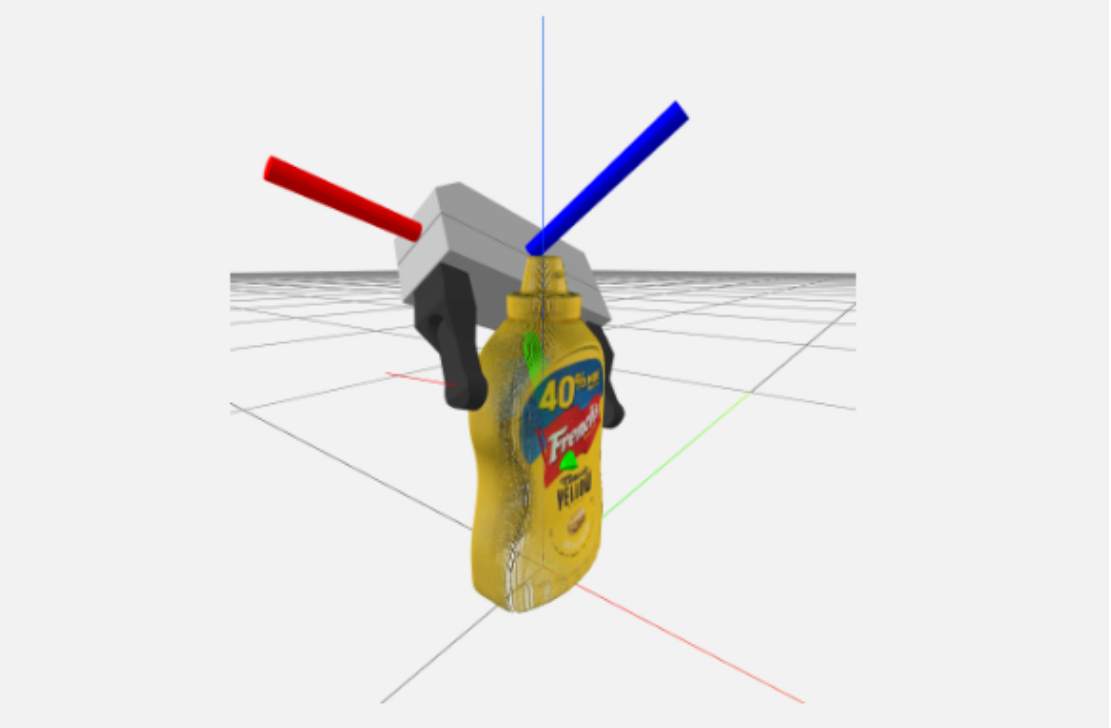

Figure 8: Gripper gripping a sample object from a different grasp calculated from the point cloud [17].

Table 6: Navigation work package details

WP 3: *Navigation*			
Start date: *15/02/2022* **End date:** *15/03/2022*			
Leader:	*Emir Melih Erdem*	**Members involved:**	*Arda Önal* *Eren Polat* *Efe Beydoğan* *Emir Melih Erdem* *Mert Barkın Er*
Objectives: *When the robot is able to detect and pick up objects, it has to be able to move around the supermarket aisles without crashing to go to the desired location and come back to the delivery point.*			
Tasks: ***Task 3.1 Navigation:*** *The robot should be able to move within the defined supermarket area. The main task is to add a navigation module on top of the robot so that when the robot receives an instruction such as "go to aisle A, row K", it should be able to go there.* ***Task 3.2 Shortest Path Calculation/Task Planning*****:** *When multiple orders are made at the same time or if an order includes multiple items, it should plan the shortest path in which all items will be collected consecutively. There are also heuristic motion planning approaches that could be tested and implemented for faster decision-making.*			

Task 3.3 Environment generation: *The supermarket environment should be created so that the robot can actually pick up objects from shelves after its pick and place functionalities are implemented.*
Deliverables ***D3.1:*** *Generating the environment which will be used for the simulation* **D3.2:** *The robot should be able to navigate within the generated supermarket environment without crashing and go to the desired locations(s).*

Table 7: Learning work package details

WP 4: *Learning*			
Start date: *15/03/2022* **End date:** *15/04/2022*			
Leader:	*Eren Polat*	**Members involved:**	*Arda Önal* *Eren Polat* *Efe Beydoğan* *Emir Melih Erdem* *Mert Barkın Er*
Objectives: *A machine learning model should be implemented to detect the desired objects with the camera frames. Therefore, the robot could be trained to detect and deliver local grocery items.*			
Tasks: ***Task 4.1 Deep Learning:*** *The machine learning model/architecture should be determined for object detection and trained based on the selected grocery items.* ***Task 4.2 Integration*****:** *The trained model should be integrated with the robot so that the camera frames will be fed to the model and the desired object's coordinates are determined in the 2D camera. The object will be picked up after this with the previous working packages.*			
Deliverables ***D4.1:*** *Train a machine learning model to do grocery item detection.* **D4.2:** *Integrate it with the current system.*			

After the WPs are completed, the remaining time will be used for the optimization of every aspect for improvements.

3.4 Ensuring Proper Teamwork

We believe that teamwork is an integral part of any successful project, so we meet up every week to discuss our progress, what we did that week and what we need to do/learn in the next week. We also have a weekly meeting with our supervisor so that our supervisor is also aware of our progress, and can give us valuable information whenever we are stuck somewhere. Whenever someone finishes up their work, they ask the whole team to check on their work so that everyone is familiar with our work. There are non-scheduled meetings in the team to make sure that everyone is up-to-date on the current state of the project.

3.5 Ethics and Professional Responsibilities

- Any and all libraries, frameworks, and tools used in GroceryBee should be licensed properly.
- The orders given to the robot should be securely stored to ensure the safety of all customers.
- Regardless of the security of the system, the users should be notified that their data is being processed in GroceryBee.
- The project might result in several lost jobs, but this might get fixed by the fact that GroceryBee might start new job areas of its own.
- The project is proprietary for now, however, it might be open source in the future.
- The data processing should be compliant with data protection laws such as GDPR.

3.6 Planning for New Knowledge and Learning Strategies

First and foremost, Drake documentation is of paramount importance for making this project work. It involves the environment to make the simulations and all the mathematical optimization modules and fundamentals of robotics that we may use some of them with a black-box approach, as it would be impossible to mathematically deduce them throughout the project. The MIT course on robotics lectured by Russ Tedrake is a great source and as a group, we are following it quite closely. For basic tasks such as pick/place and movement in an ideal setting, the framework provides us with tools to work on. Although, robotics is unavoidably related to many other disciplines. Different parts of this project will require different disciplines to borrow from, these include but not limited to task/motion planning (RL, automated planning, path-finding), perception (traditional CV algorithms, Deep Learning), and control. For perception, we may have to review state-of-the-art approaches using neural models.

These may appear in different architectures, and we may have to read articles regarding Transformer architectures, and perhaps watch lectures from universities regarding these subjects. We may use pre-trained models to fine-tune them on a specified dataset and customize the SOTA models giving top metric results for our work to achieve better perception. We may also use already fine-tuned models by just importing them and using them for inference by following instructions given by researchers that trained the model. These are often in open-source GitHub source code (such as BERT, CodeT5, BART, etc.) for other researchers

to use and exploit the optimized model. If we plan to work on task/motion planning, we may have to learn more about path-finding algorithms and work on finding novel approaches to leverage the current algorithms and develop better ones. To do this, one would need to learn the heuristic search approaches currently used for robot task/motion planning. We may check work in literature, such as Barto/Sutton's book on RL, Task and Motion planning by Dantam.

4 Glossary

Heuristic: A heuristic involves the approach to solve a problem leveraging a calculated guess from previous experiences. In order to find an (near) optimal solution, instead of trying/calculating all possible solutions to a problem, say a traversal problem, heuristic approaches enable agents to make a decision without having to consider all paths, eliminating unnecessary amounts of work.

Forward Kinematics: Forward Kinematics is the process of using kinematic equations, rigid transforms, and transformations to calculate the position and the velocity of the end-effector (in our case the gripper) when the joint angles and the angular velocities of the robot arm are given.

Point cloud: A set of points in 3D space that represent a 3D shape, this point cloud can be generated by the camera on the robot.

Perception: Perception in the robotic context involves the algorithms and approaches to enable the robots to understand data received from sensors and cameras. These algorithms may involve traditional Vision algorithms or ML/DL approaches that have been used in the current.

Normal vector: The normal vector, often simply called the "normal," to a surface is a vector that is perpendicular to the surface at a given point. When normals are considered on closed surfaces, the inward-pointing normal (pointing towards the interior of the surface) and outward-pointing normal are usually distinguished. [18]

5 References

[1] “Turkey’s online supermarket grows 434% in H1, reaching TL 1.8B,” *Daily Sabah*, Sep. 01, 2020. [Online]. Available: https://www.dailysabah.com/business/economy/turkeys-online-supermarket-grows-434-in-h1-reaching-tl-18b. [Accessed: Oct. 15, 2022].

[2] M. Djordjevic, “20 Incredible Online Grocery Shopping Statistics for 2022,” *SaveMyCent*, May 19, 2022. [Online]. Available: https://savemycent.com/online-grocery-shopping-statistics/. [Accessed: Oct. 15, 2022].

[3] K. Matthews, “5 Robots Now in Grocery Stores Show the Future of Retail,” *Robotics Business Review*, Apr. 10, 2020. [Online]. Available: https://www.roboticsbusinessreview.com/retail-hospitality/5-robots-grocery-stores-now/. [Accessed: Oct. 15, 2022].

[4] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1examples_1_1manipulation__station_1_1_manipulation_station.html#details. [Accessed: 13-Nov-2022].

[5] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/pydrake/pydrake.manipulation.kuka_iiwa.html. [Accessed: 13-Nov-2022].

[6] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1systems_1_1_context.html. [Accessed: 13-Nov-2022].

[7] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1multibody_1_1_multibody_plant.html. [Accessed: 13-Nov-2022].

[8] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1systems_1_1_integrator_base.html. [Accessed: 13-Nov-2022].

[9] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1systems_1_1_state.html. [Accessed: 13-Nov-2022].

[10] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/doxygen_cxx/classdrake_1_1geometry_1_1_meshcat.html. [Accessed: 13-Nov-2022].

[11] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://drake.mit.edu/pydrake/pydrake.systems.planar_scenegraph_visualizer.html. [Accessed: 13-Nov-2022].

[12] “6.843 final project: Online search and retrieve,” YouTube, 09-Dec-2021. [Online]. Available: https://youtu.be/XyR78zb1Z_w. [Accessed: 11-Nov-2022].

[13] B. Lennartson and K. Bengtsson, “Reducing energy consumption through optimised robot systems,” *Open Access Government*, Sept. 22, 2017. [Online]. Available: https://www.openaccessgovernment.org/reducing-energy-consumption-robot-systems/28061/. [Accessed: Oct. 16, 2022].

[14] J. Snow, “How to Build a More Sustainable Robot,” *The Wall Street Journal*, Oct. 19, 2021. [Online]. Available: https://www.wsj.com/articles/how-to-build-sustainable-robot-11634585544. [Accessed: Oct. 16, 2022].

[15] F. Hartmann, M. Baumgartner, and M. Kaltenbrunner, “Becoming sustainable, the New Frontier in Soft Robotics,” *Advanced Materials*, vol. 33, no. 19, p. 2004413, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/adma.202004413. [Accessed: Oct. 16, 2022].

[16] P. Agarwal and B. Prabhakaran, “Robust blind watermarking of point-sampled geometry,” *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 1, pp. 36–48, 2009. [Accessed: 13-Nov-2022].

[17] R. Tedrake, “Robotic Manipulation Perception, Planning, and Control,” *Robotic manipulation*. [Online]. Available: https://manipulation.csail.mit.edu/index.html. [Accessed: 13-Nov-2022].

[18] “Normal Vector,” Wolfram MathWorld. [Online]. Available: https://mathworld.wolfram.com/NormalVector.html. [Accessed: 13-Nov-2022].