Bilkent University
Department of Computer Engineering

Senior Design Project
T2323
GroceryBee

Detailed Design 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, E. 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

<13.03.2023>

Contents

1 Introduction **3**
1.1 Purpose of the System 3
1.2 Design Goals 4
1.3 Definitions, acronyms, and abbreviations 5
1.4 Overview 7
2 Proposed Software Architecture **8**
2.1 Overview 8
2.2 Subsystem Decomposition 8
Subsystem Explanations 9
3 Test Cases **10**
4 Consideration of Various Factors in Engineering Design **18**
5 Teamwork Details **19**
5.1 Contributing and functioning effectively on the team 19
5.2 Helping create a collaborative and inclusive environment 20
5.3 Taking lead role and sharing leadership on the team 20
6 References **22**

Detailed Design 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.

1.1 Purpose of the System

We are developing "GroceryBee" for the automation and acceleration of gathering grocery items from the shelves of supermarkets for online orders. 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 are using the Drake framework, which uses control theoretic design and analysis to enhance the robustness of the physics engine. Open problems in motion planning, perception, and decision-making are being worked on using these approaches from various disciplines.

GroceryBee will be able to collect required items in response to the orders while navigating between the supermarket aisles without crashing. The robot will have access to the locations of products within aisles and navigate to the location when an order is requested. It will 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 will also calculate the most optimal path to collect the orders, ensuring the fastest delivery possible for customers.

1.2 Design Goals

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 it 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, the GroceryBee should easily be converted to a multi-robot application.
In the future, it could 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.

Performance

GroceryBee should calculate the most optimum path with a low computation time. When an order is made, the robot should move fast and try to collect items as fast as possible to make sure that the order is completed in a short time span.

Marketability

GroceryBee as a software is designed with a real physics engine and uses models of robots of “KUKA” which is a robotics company. The final product can be advertised with its ease to use in real life scenarios by stating the fact that it can be used in real life. There are millions of supermarkets across the globe and having a robot which can collect items from shelves would certainly attract these supermarket companies. GroceryBee should be designed so it is cheap, robust and accurate so that it could be used in real supermarkets.

Scalability

GroceryBee should be built so that it could be scaled to systems with multiple robots. These robots can be operated in an efficient manner to complete an order in the shortest time possible. Furthermore, it should be adaptable to environments with human beings for its incorporation to daily supermarkets.

Maintainability

GroceryBee should be implemented following object-oriented programming and other programming principles to ensure its ease to maintain.

1.3 Definitions, acronyms, and abbreviations

SOTA: State-Of-The-Art. It refers to the current best level of performance achieved in a particular field or task and is commonly used in the context of technology and research.

Heuristic: A heuristic involves the approach to solving a problem leveraging a calculated guess from previous experiences. In order to find a (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.

Sequential decision-making: Sequential decision-making is the process of making a series of decisions over time, where the outcome of each decision affects the available options for subsequent decisions. In the context of robotics, it is a key aspect as robots often need to make decisions based on sensory input and previous actions. This process involves gathering information about the environment, determining the current state of the robot, selecting an action based on this information and state, and then updating the state based on the outcome of the action.

End-effector: In robotics, the end-effector is the tool that is attached to the end of a robotic arm or manipulator, which is designed to interact with the environment. In our context, the end-effector is the gripper.

Motion/path planning: Motion planning, also known as path planning, is the process of determining a sequence of valid motions that meet certain movement constraints to achieve a desired goal, and perhaps, optimize the movement in certain ways. The goal can be a desired end-effector position or a more complex task, such as navigating through an environment or manipulating an object.

Kinematics: Kinematics in robotics is a branch of study that deals with the motion of robots without taking into account the forces or torques that cause that motion. It focuses on analyzing the position, velocity, and acceleration of different parts of a robot, as well as how they change over time. By understanding the kinematics of a robot, engineers can optimize its movement, ensure that it moves efficiently, and make it perform specific tasks accurately. There are two main types of kinematics in robotics: forward kinematics and inverse kinematics. **Forward kinematics** is concerned with determining the position and orientation of a robot's end-effector (gripper in our case) based on the joint angles of the robot. In contrast, **inverse kinematics** is concerned with finding the joint angles of a robot that will result in a desired end-effector position and orientation.

Differential inverse kinematics (DiffIK): Differential inverse kinematics is a technique used in robotics to control the motion of a robot's end-effector based on changes in its position or orientation. While inverse kinematics involves computing the joint angles required to achieve a desired end-effector position, differential inverse kinematics involves computing the change in joint angles required to move the end-effector in a desired direction. This results in a smooth motion of the end-effector since DiffIK takes into account the current state of the robot, including the velocity and acceleration of the end-effector.

Kinematic trajectory optimization: Kinematic trajectory optimization is a technique used in robotics for computing the optimal trajectory of a robot's end-effector over time, subject to constraints on the robot's motion. The goal is to find a motion that minimizes some cost function while satisfying constraints on the robot's movement, such as joint limits, joint velocities, or avoiding collisions with obstacles.

Sampling-based methods: Sampling-based methods are a set of techniques for motion planning which involve randomly sampling the robot's state space to explore potential configurations, and finding a collision-free path from the robot’s current position to the desired goal by connecting those. They offer an efficient way to plan the robot’s motion.

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.

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.

RGBD: Red Green Blue Depth. An RGBD camera is a type of camera that captures both color and depth information.

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 [4].

Segmentation: Segmentation is the process of dividing an image into different regions, based on some criteria such as color, texture, or shape. In our context, it is necessary to identify and separate different objects within a scene and determine how to grasp them.

1.4 Overview

In the rest of this report, multiple issues related to the design of our project will be discussed. First, the current software architecture will be described by taking into account competitors, alternatives and our current solutions. Competitors to GroceryBee exist, as mentioned in the introduction, however we intend to make GroceryBee a robust and efficient robot with minimal error rate, so it can significantly enhance the process of the delivery of online orders. Secondly, the proposed software architecture will be discussed with the help of a “Subsystem Decomposition” diagram. The essential subsystems of our project will be clarified. Following this, functional and non-functional tests for our project will be given. We intend to make sure our project passes all of these test cases in its final form, as to ensure a smooth experience for any users in the future. Finally, the various factors we considered when engineering our design will be discussed and teamwork details, including contributing and functioning effectively on the team, helping create a collaborative and inclusive environment, and taking the lead role and sharing leadership on the team, are going to be elaborated on.

2 Proposed Software Architecture

2.1 Overview

In order to make the development of GroceryBee easier, we have divided the system into subsystems, all of which are critical to the project's success, and also complex enough to be separated from each other. The "Subsystem Decomposition" diagram shows the subsystems we came up with for our project, as well as the relationships between them, as in which subsystem communicates with which other ones to obtain the necessary data it needs. With this design, we are aiming to simplify the maintenance of our system, as well as ensure its performance.

2.2 Subsystem Decomposition

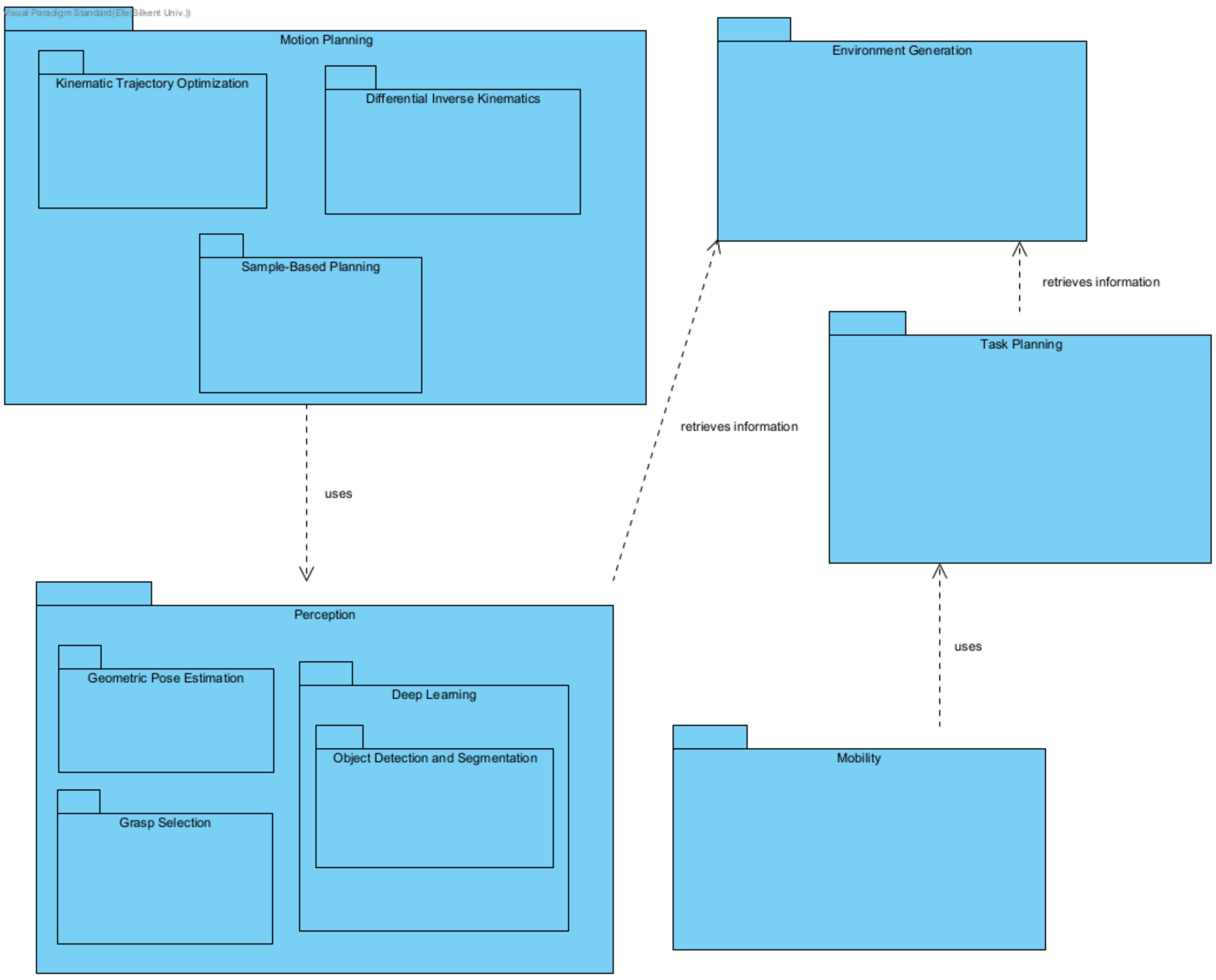

Figure 1. Subsystem Decomposition Diagram

Subsystem Explanations

Environment Generation:
The “Environment Generation” subsystem consists of functionalities responsible for generating a supermarket environment, which GroceryBee will traverse. The environments will be generated dynamically depending on different supermarket interiors. GroceryBee will adapt to the given environment and plan its paths accordingly to ensure a fast and efficient delivery.

Task Planning:
The “Task Planning” subsystem includes the functionalities for the robot to find an optimum order for tasks in any generated supermarket environment. To do so, this subsystem retrieves information from the “Environment Generation” subsystem. The information regarding the placement of shelves, items, etc. is received, and the “Task Planning” subsystem utilizes this information to come up with an optimum path to gather items as quickly as possible.

Mobility:
The “Mobility” subsystem is responsible for integrating the two dimensional planar movement of the robot to move along the aisles and navigate within the supermarket. This subsystem will be built separately and then integrated to the robot. It will use the “Task Planning” subsystem to plan the order of items that will be picked up to navigate with the optimum path.

Perception:
The “Perception” subsystem will be solely responsible for the understanding of the robots environment. In the real world, the robot will not know the locations of the items therefore, camera systems are required for the robot to perceive its surroundings. Cameras and depth sensors will be added to the system. The images and the information retrieved from them will be used with image processing and deep learning algorithms to detect and pick up the objects. It will consist of three subsystems. The “Geometric Pose Estimation” subsystem is the system responsible for understanding the orientation of the object with regards to the world. This information is crucial to determine how to pick the object up. The “Grasp Selection” subsystem is the system for deciding the pinching points of the gripper for picking up the item. As the shapes and the sizes of the items vary, sophisticated algorithms are necessary for finding valid grasp candidates. Lastly, the “Deep Learning” subsystem is the implementation of neural networks and deep learning algorithms to detect and segment the objects with the images and the depth information retrieved from the cameras.

Motion Planning:
The “Motion Planning” subsystem includes the functionalities that will let GroceryBee calculate the best ways to pick an item from the shelves and put them in the bin for online delivery orders. This subsystem consists of 3 further subsystems. The first one, named “Kinematic Trajectory Optimization”, encompasses the functions for optimizing the movement of the robot’s arm when reaching objects on the shelves. By optimizing this process, we ensure that the robot doesn’t touch the shelves by mistake when picking up an item, or it doesn’t knock over anything while reaching or retracting to put an item in the bin. The second subsystem is “Differential Inverse Kinematics”. Differential IK is necessary for GroceryBee to calculate where to place its gripper when picking up an item, so the robot will know exactly which movements to make prior to reaching the shelves, and thus the possibility of mistakes will have been minimized. The final subsystem is called “Sample-Based Planning”. This subsystem consists of functionalities that are both fast and efficient as they don’t require fully exploring the configuration space. Rather, random configurations are sampled and checked for plausibility, finally helping the robot determine a solution. The “Motion Planning” subsystem uses

information from the “Perception” subsystem to make informed decisions when optimizing the picking up of items.

3 Test Cases

Test ID	Test Type	Objective	Procedure	Expected Results	Priority
1	Functional	Robot should fit in the aisle	Check if the robot is smaller than the aisle	Robot fits in the aisle	Critical
2	Functional	Items should fit in the shelves	Check if the items safely fit in the shelves	Items fit in the shelves	Critical
3	Functional	Item not found on shelf	The robot is tasked with picking up an item which doesn’t exist	The robot skips picking up the item/checks nearby shelves to see if the item is there	Major
4	Functional	Motion path occluded	An obstacle is added to the robot’s path	The robot chooses another path for itself	Major
5	Functional	Shelf higher than robot	The robot is tasked with picking up an item which is higher than the robot	The robot skips picking up the item	Major
6	Functional	Item too deep in the shelf	The robot is tasked with picking up an item which is too far deep in the shelf	The robot skips over picking up the item	Major
7	Functional	Item blocked	The robot is	The robot	Major

		by another object	tasked with picking up an item which is blocked by another object	skips over picking up the item/places the object and places it away to successfully pick the item	
8	Functional	Robot movement path displayed on screen	The robot creates a movement path after getting tasked	The movement path is displayed correctly	Minor
9	Functional	Motion time is correctly displayed	The robot creates a motion path after getting tasked	The motion path time is displayed correctly	Minor
10	Functional	Robot drops the item before placing in basket	The robot picks up an item, then drıps the item while in the process of dropping it in the basket	The robot picks the item back up and places it in the basket	Major
11	Functional	Basket full	The robot fills the basket up with items it was tasked with picking up	The robot fills the basket successfully, empties it and then continues picking up whatever items left in the list	Major
12	Functional	Robot drops other item when picking up from shelf	The items are placed such that when the robot picks up the item, it drops	The robot does not drop the other item.	Major

			another while bringing it back		
13	Functional	Environment information correctly transferred to robot	The robot is started, then the environment information is checked to see if it’s correct	The environment information is correctly transferred to the robot, and the robot is able to process the information.	Critical
14	Functional	Trajectory correctly calculated by robot when reaching item	The robot is given an item to retrieve, the trajectory is checked after the robot calculates it	The trajectory calculated by the robot is correct and the item is retrieved successfully	Critical
15	Functional	Robot doesn’t hit the edge of shelves when picking item	The robot is given a list of items to pick up	The robot does not hit the shelves when retrieving the items	Major
16	Functional	Robot’s camera is functioning correctly	The robot is given an item to retrieve, the camera is connected to the robot for it to calculate a trajectory	The camera works correctly and the robot retrieves the item by using the image retrieved from the camera	Critical
17	Functional	Robot’s joints are functioning correctly	The robot is given an item to retrieve	The robot’s joints function correctly and the robot successfully picks the item	Critical

				up.	
18	Functional	Robot correctly collects all of the items from shelves	The robot is given a list of items to retrieve. The basket is checked after the robot is done.	The robot collects all of the items given in the list.	Critical
19	Functional	Robot calculates the most optimal path for collecting items	The robot is given a list of items to retrieve	The robot calculates a path and starts picking the items up	Critical
20	Functional	Robot is unable to find item because item is out of stock	The robot is tasked with picking up an item which does not exist in the shelves	The robot skips over picking up the object	Critical
21	Functional	Robot cannot grasp item	The item is placed such that the robot is unable to pick the item on the first try	The robot retries to pick the item again	Major
22	Functional	Item texture is incorrect	The item’s texture is changed before making the robot pick it up	The robot picks up another item on the shelves as the same kind as the item given to the robot	Minor
23	Functional	Item geometry not perceived correctly	The item is placed such that the robot’s	The robot picks up another item that can be	Major

			camera cannot perceive the item correctly	perceived	
24	Functional	Item damaged	The item is placed such that the camera can perceive that the item is damaged	The robot picks up another item that is not damaged	Major
25	Functional	Delivery point unavailable	The delivery point is set up such that the robot is unable to find a path there	The robot waits for the delivery point to be available	Critical
26	Functional	Previous delivered basket not picked up and new delivery is attempted	Delivery system is set up such that orders are made before previous basket is picked up	Hang on until the basket is picked up	Major
27	Functional	Position in front of shelf is too far	Place the robot to a far position from the shelf	Get closer to shelf	Critical
28	Functional	Item type is not suitable for being picked up by two grippers	Recognize certain items that are not suitable to be picked up by two grippers	Use a different kind of gripper (such as vacuum)	Major
29	Functional	Differential inverse kinematics	See if pick objective is reached	Try to pick up using trajectory	Critical

		not working properly	when the task is given	optimization	
30	Functional	Segmentation not working as expected	Detect if segmentation fails with the constructed point cloud	Use alternative approaches such as deep learning	Major
31	Functional	Environment generation shouldn’t lead to unreachable areas	Generate areas that are unreachable by the robot	Calculate the reachability and not take the assigned task of pickup	Critical
32	Functional	Robot falls over	Place the robot lying down	Shuts itself down	Critical
33	Functional	One item unavailable but other items remain on the list	Recognize that the item is unavailable and plan accordingly	Proceeds to collect other items	Major
34	Functional	Robot crashes into shelf	Place the item so that a pickup would be likely to cause a collision	Optimize the trajectory to avoid collisions	Critical
35	Non-functional	An item is attempted to be picked up at most 5 times	Count the times an item is attempted to be picked up	Try to pick up another object	Minor
36	Non-functional	The process of picking up an item with all attempts should take at most 2 minutes	Measure the pick up time for each item and print/check it	Skip the item, try alternative perception methods such as deep learning	Minor

37	Functional	Valid grasp candidate is found and optimum trajectory is calculated	Check the logs for the trajectory and grasps	Robot moves its joints to pick up the object	Major
38	Functional	Object has been picked up	Observe that the object is picked up	Robot places object in the basket	Major
39	Non-functional	Completion of an order shouldn't take more than 20 minutes	Measure the time it takes for an order to arrive	Cancel the order if more than 20 mins	Minor
40	Functional	Basket is too small for the whole order	Fill the basket by manually placing items in it	Robot should take several tours to complete the order	Major
41	Functional	Item is fragile	Place a fragile item such as a glass container	Item should be placed in basket with care	Minor
42	Non-functional	Robot mobility speed shouldn't exceed 5 km/h	Check if the robot speeds up more than 5 km/h	Robot should slow down	Minor
43	Non-functional	Joint speed shouldn't exceed 0.2 m/s	Measure the speed of the joint	Report the time, make it faster	Minor
44	Functional	Stop button is pressed	Check if robot stops	Robot should stop	Minor
45	Functional	Return robot button is pressed	Robot stops whatever its doing safely and returns	Robot should return to its dock and wait until it is given another instruction	Minor

46	Functional	Robot out of power (prizden cikti)	Cut the power to the robot	Robot should shut down but should stay as it is to prevent the items from falling down	Minor
47	Functional	Items are on top of each other or are in contact, which impacts segmentation	Place items on top of each other to test the robot	Adjust an algorithm for the blocks world-like problem	Minor
48	Functional	Item depth not perceived properly	Check if the gripper reaches the correct depth	Adjust the mobile	Major
49	Functional	Order cannot be received	Check if the order is received from the outside stakeholder s of the system	Report an error to outside users	Minor
50	Non-functional	Environment generation shouldn’t take longer than 5 minutes	Measure the environment generation time	Report the time	Minor

4 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 the 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 in order 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-492 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

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 [5]. Next, the use of unsustainable materials in the production of the robot can be avoided, with the exception of sensors, computer chips, etc. [6], [7]. Lastly, to ensure sustainable development, no prototypes or physical components will be built. All testing and development will be carried out through virtual simulations.

5 Teamwork Details

5.1 Contributing and functioning effectively on the team

Efe: The responsibilities were divided among all of the team members in the implementation phase. Efe took part in creating the environment for the robot, pick and place without cameras, and placing the RGBD cameras to receive real time information so items could be picked up with information about the environment. He also contributed to implementing a mobile base for the robot. When writing the reports, he took active roles in all of them and helped with the parts that were assigned to him, like other members. All in all, as the rest of the group, he has fulfilled his responsibilities in every part of the project.

Arda: Arda contributed to creating the simulation environment, setting up the initial basic pick and place approach, configuring the robot's path for picking up objects from shelves, setting up the point clouds that are obtained from cameras to exclude parts that didn't belong to shelves, converting the code from the deep note environment to local version and inclusion of git and version control and lastly, implementation of the mobile base of the robot.

Eren: Eren contributed to the initialization of the project, robot and the simulation environment. He has worked on basic pick and place, creating the shelves/placing objects of the simulation.

He is now currently working on using deep learning models for instance segmentation to replace classical methods used in perception.

Mert: Mert contributed to initialization of the project, the first pick-and-place algorithms. He helped set up the simulation environments and created objects to place in the simulation environment. He also worked on making the robot mobile and trajectory optimization.

Emir: Emir contributed to building the simulation environment, initiating basic pick & place, planning the trajectory with DiffIK, and implementing the grasp selection system while ensuring that the robot functions reliably and collision-free in the first half of the project. Currently, he is working on motion planning—implementing kinematic trajectory optimization—and helping develop a mobile base.

5.2 Helping create a collaborative and inclusive environment

Efe: To create a collaborative environment, Efe helps his teammates whenever they need it and offers knowledge. Further, in any situation he makes sure to listen to his teammates' ideas on how to tackle an issue, and makes decisions accordingly. He is able to utilize his team members' strength and asks for help whenever he requires it, in order to save time and effort. All in all, he tries to maintain an inclusive environment for his teammates where everyone feels valued and listened.

Arda: Arda was always enthusiastic to involve others in trying to implement certain aspects of the project, even when they didn't believe in themselves that they could do it. He made sure that the workload was distributed equally amongst members and asked other members to join him in the implementation of crucial tasks to ensure teamwork. He listened to and considered everyone's suggestions while solving problems with regards to the project and debated ideas to make sure that everyone feels involved and their opinion is valued.

Mert: Mert helped his teammates whenever he could. He made sure that he was not doing less work than he was supposed to, and he helped other teammates if they had more work assigned to them than they were capable of doing.

Emir: Emir is always eager to help his friends in their tasks, provide his teammates with his understanding of Drake and other frameworks, and request support whenever he needs technical help or new ideas to progress. By working together with others in many aspects of the project and constantly exchanging ideas, he is helping create a collaborative and inclusive environment.

Eren: Eren is always helpful to his friends when they need it, and provide both emotional and technical support during the problems they faced throughout the project. He tried to work with others in an inclusive and accepting way to develop new ideas and solve problems.

5.3 Taking lead role and sharing leadership on the team

Efe: For the report aspect of the project, Efe usually shares leadership with Arda to assign responsibilities and make sure everything is done on time and in an acceptable way. For the implementation part, Efe, like other members of the team, leads a certain section, such as implementing various parts of the robot such as a joint for the mobile base. He usually shares the leadership of implementation with another person to develop the project effectively and collaboratively.

Arda: Arda took responsibility in ensuring that the deliverables are implemented before the deadlines and distributed project’s pieces to team members according to their interests. He also migrated the entire project from DeepNote (a collaborative web based notebook) which was inefficient in collaboration and version control to local development environments which reduced the loading time of the simulation significantly and improved version control. To do this, everyone was forced to dual boot their computers to the Linux operating system since the framework that we were using only worked on Linux. Lastly, in the reports he separated the sections to group members with Efe to make sure that everyone had equal work to do.

Mert: Mert took a leadership role at the beginning of the project, helping set up the initial robot and the environment.

Emir: Emir shared leadership during most of the implementation process, helping navigate the progress on DeepNote and then the git repository. Currently, he is leading the “motion planning” aspect, which is a major subsystem of the project.

Eren: Eren took a leadership role at the beginning of the project, helping set up the initial robot and the environment. Currently, he is leading the “deep learning” aspect of the project.

6 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: Mar. 5, 2023].

[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: Mar. 5, 2023].

[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: Mar. 5, 2023].

[4] “Normal Vector,” Wolfram MathWorld. [Online]. Available: https://mathworld.wolfram.com/NormalVector.html. [Accessed: Mar. 7, 2023].

[5] 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: Mar. 8, 2023].

[6] 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: Mar. 8, 2023].

[7] 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: Mar. 8, 2023].