Grow Simplee's Route Planning for Optimized on Time Delivery

Mid Evaluation Report

Team 53

Inter IIT Tech Meet 11.0

Contents

1 Introduction

The modular nature of the given problem statement allowed us to partition it into three tasks, namely, the volume and weight scanning tool, the route optimization algorithm, and the application interface. Each of these tasks are independent of the other (up until the point of integration, when all three components will have to be merged into a final product).

One observation we made with regard to the route optimization algorithm was that a perfect, one-hundred percent optimal solution might not exist. Hence, we thought it wise to take up a comparative technique and test multiple approaches against one another. We realized there would always be room for optimization and that different approaches may give different results in varying scenarios.

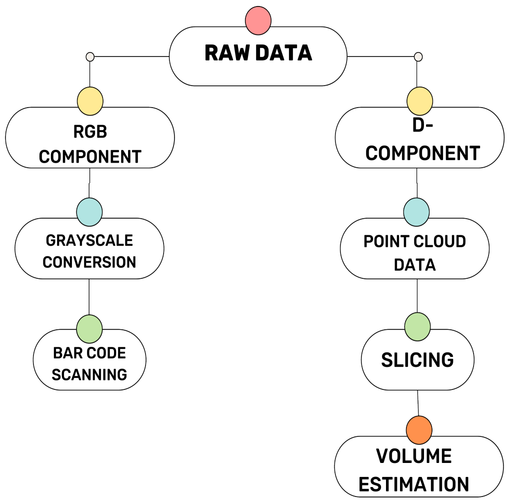
2 Volume and Weight measuring tool

2.1 Volume Estimation Method

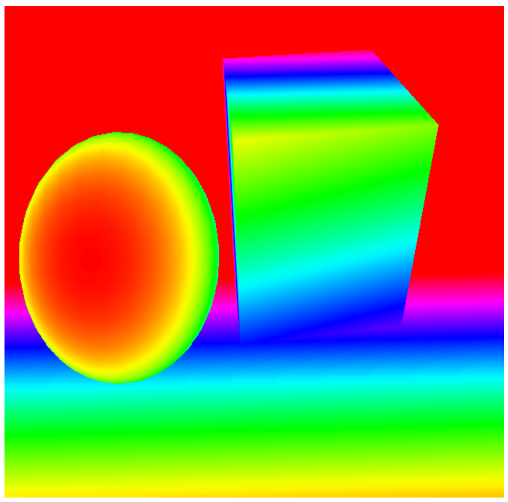
Volume Estimation in a contact-less manner with sensors and subsequent processing involves different techniques; prominent ones include image and depth data processing. Our approach to this is based on depth image processing using an RGB-D camera. The RGB-D camera is a type of depth camera that provides RGB(Red-Green-Blue) and D(Depth) data in real-time. The Depth information is retrievable through a depth map created by the camera. The depth data obtained is converted to a point cloud [1] and is further sliced, which gives us the area and volume using volume estimation algorithms.

We first need a full 360° view of the object to get through the above processes. This can be done by either rotating the camera or by rotating the object itself. After considering the complexity of the mechanism we chose to rotate the object. To capture discrete frames and use them for full 3D reconstruction [2], a rotary encoder is to be integrated with the turntable. For the bar-code part, the RGB feed is converted to grayscale to reduce computation and then with the use of pyzbar library, the product SKU can be identified. Having four channels on the camera, namely the Depth(D) and the Red-Green-Blue(RGB), with multi-threading, we can implement both the volume estimation and barcode scanning on the same camera. Thus, we covered the role of our camera and what features of the object will be extracted using our tool.

From the number of point clouds captured, the background is subtracted, and we extend the data using back and forward projection. Volume intersection is applied to the scattered 3D object data aquired by combining multiple views of the objects and processed with known coordinate transformations between the views.



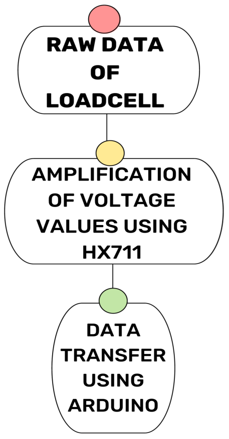
(a) Flow chart for finding the Volumetric weight and the SKU of the object



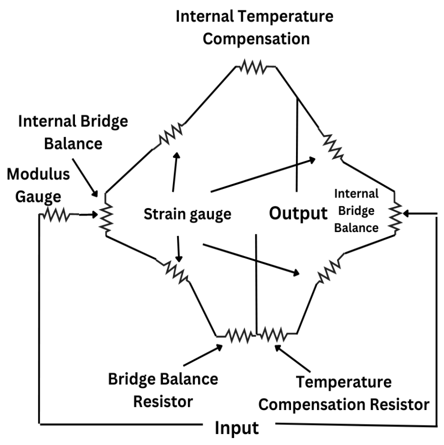
(b) A sample Depth feed of the camera

2.2 Weight Estimation Method

Using a load cell is the standard method to measure weight in any weight-measuring device. Here, we, too, use the same in the form of a strain gauge-based load cell. The way how a load cell works is that when an object is placed on it, the elastic membrane of the strain gauge deflects and creates a strain at those locations where the load is placed. There are four such load cells connected in a Wheatstone bridge-type connection. The weight is proportional to the imbalance voltage created at the 2 node points(as shown in the figure). The temperature compensation resistor also helps to reduce noise in the measured value. Considering the operating range and expected item weight, a 50kg load cell will be used. This transducer will then be interfaced with an HX711 sensor-amplifier that powers the load cell circuitry and scales the signal into definite readable values. The weight readings will be communicated from Arduino microcontroller to the main computer which is recording the values. The communication will be done through WiFi network, Bluetooth or direct USB.


RAW DATA OF LOADCELL
AMPLIFICATION OF VOLTAGE VALUES USING HX711
DATA TRANSFER USING ARDUINO

(a) Flow chart for finding the dead weight of the object



(b) Internal mechanism of the Load cell

3 Route Optimization Algorithm

3.1 Approach 1: Greedy Algorithm with Optimizations

1. Commence a loop over a threshold variable *max_possible_distance*, over the range.This threshold variable will be used in a later stage of the algorithm.
2. Commence a loop to iterate over all the items 'i' sorted in ascending order according to the Expected Date of Delivery (EDD).
3. Commence a loop to iterate over all the riders 'j'
4. For an item 'i' and rider 'j', the rider must have sufficient volume available in their bag to satisfy the order of deliveries and pickups preceding item 'i', as well as to add item 'i'. Also, the rider must reach the location of item 'i' before its EDD, and the distance between the current location of the rider and the location of item 'i' must be less than or equal to *max_possible_distance*.
5. Finally, we assign item 'i' to the rider that satisfies the required constraints and delivers the item in minimum time.

3.2 Approach 2: Dynamic Programming

1. Sort all the items in ascending order of Expected Date of Delivery (EDD).
2. Commence a loop to iterate over all the items 'i' sorted in ascending order according to the Expected Date of Delivery (EDD).
3. Commence a loop to iterate over all the riders 'j'.
4. Now, we use a two-state dynamic programming model defined as :

   dp[i][j] = the number of items successfully delivered till now, for item 'i', rider 'j'

   The state signifies that if item 'i' is assigned to rider 'j', then dp[i][j] is the maximum number of successful deliveries that are achievable preceding item 'i'.
5. For each state (i,j), we store three arrays accounting for the remaining bag capacity, the distance traveled by each rider and the rider's current location.
6. Commence a loop 'l' from 1 to (i-1) to calculate the current state of dp[i][j]
7. Commence a loop over all riders 'k' who delivered the l^{th} item.
8. Now for the state dp[i][j] we assume that the last successful delivery made was that of item 'l' and by rider 'k'.
9. Check if it is possible for rider 'j' to deliver item 'i'

   if possible,

   $$dp[i][j] = max(dp[i][j], dp[l][k] + 1) \tag{1}$$

   else,

   $$dp[i][j] = max(dp[i][j], dp[l][k]) \tag{2}$$
10. For each iteration of the innermost loop we update the remaining bag capacity, the distance traveled by each rider and the rider's current location considering the delivery of item 'i' by rider 'j'.
11. After iterating through all riders, we will get a value of 'j' for which dp[i][j] is the maximum. If multiple j have the same value, then choose the one for which the sum of the total distance traveled by all the riders is minimum. Thus, we will get a final order of delivery items for each rider.
12. To get the final answer, we store the rider 'j' who delivered item 'i' for all states dp[i][j].

3.3 Approach 3: K-Means Clustering Algorithm

1. If we have m-riders, we start with m locations that will act as centroids of the clusters.
2. We assign all the remaining locations to one of these 'm' clusters on the basis of the centroid which is at minimum distance from the current location.
3. For the m clusters obtain, we calculate a new centroid by calculating the variance for each cluster.
4. Using the newly obtained centroids, we will again obtain 'm' new clusters.
5. We continue this process until there is no change in any of the clusters' members

3.4 Approach 4: Greedy Algorithm

1. Iterate through the riders sorted in descending order of bag volume.
2. We start with rider 1 from the hub and move to the closest delivery location from the hub.
3. At every delivery location, we find the next closest delivery location from the current location.
4. We continue in this manner until the rider's bag cannot hold the next closest item.
5. After the rider's bag capacity has been exceeded, the rider returns to the hub, and now rider 2 begins from the hub in the same manner as the previous rider.
6. This process continues over all the riders until all the items have been assigned to some rider.

All three algorithms above serve the purpose of clustering the given 'n' items across 'm' riders. It may so happen that the order of deliveries assigned by these algorithms is not optimal with regard to the total distance traveled. To resolve this issue, we can use a modified version of the Travelling Salesman Algorithm, taking into account the EDD of the items. This will help us to minimize the distance traveled by each rider without affecting the percentage of on-time deliveries. We will also explore the idea of combining the aforementioned three algorithms in order to build a final cumulative algorithm which will produce the best results with maximum accuracy.

For handling the dynamic addition of pickup items, we will iterate through the order of deliveries assigned to each rider. For each rider's delivery sequence, we will calculate the extra distance/time that will be added if the pickup item is inserted between two deliveries. The bag volume of the rider and the time slot for executing the pick up will also be taken into account. Also, we will try to ensure that after catering to this pickup, the rider is still able to complete the initially assigned deliveries within time and with sufficient bag volume. Finally, we will choose the rider that follows these constraints and for whom the extra distance/time is minimum.

To test the above approaches, we generated random test cases considering number of deliveries in the range 50-100 and number of riders in the range 5-10. Item volumes were considered to be in the range 10-20% of the bag volumes. By comparing the results of our algorithm with the maximum number of achievable on-time deliveries feasible with the given bag sizes, we found that our algorithms gave an accuracy of 60-70% (successful deliveries).

In order to ensure correctness of our solution, we wrote a function that will test it against all the constraints to certify that none are violated and to calculate the percentage of on-time deliveries. Each approach has its own strengths and weaknesses, and the choice of which one to use will depend on the specific requirements of the problem at hand. It is important to consider factors such as the size of the input, the complexity of the constraints, and the desired level of optimality when choosing an approach for solving an item-rider assignment problem

4 Application Interface

4.1 Users of the apps

The app has three types of users:

1. Item Scanning Personnel
2. Warehouse Mangers
3. Delivery Riders

4.2 Functional Requirements

The item scanners, warehouse managers as well as riders will have to log in using an id and password before obtaining access to the following functionalities:

Item Scanning personnel

1. As items are placed within the scanning tool, data received in the sensors will be fed to the application and will be visible on the web/app interface, along with the inferences regarding the item's information.
2. The scanner's task is to monitor the sensor data and approve the item currently visible on the interface. The item will then be added to the list of all previously scanned items.
3. Scanners may remove any item from this list at any point in time if they wish to do so.
4. Finally, once the final list of scanned items is ready, the scanner will have to submit it via the interface. This action will add the items to the database.

Warehouse Managers

1. The manager's dashboard will display the list of scanned items as well as the information about the riders for that day.
2. In case of any inconsistencies, the manager will be able to alter the information about any rider or item.
3. Upon clicking the "Dispatch" button, the routing algorithm will start running in the back-end and will perform optimal clustering of items to riders, as well as finding the optimal route for each rider.
4. Finally, once the final list of scanned items is ready, the scanner will have to submit it via the interface. This action will add the items to the database.
5. On the navigation page, the manager will be able to view the current location of each rider and their projected route. This page will also support dynamic addition and removal of pickup points.

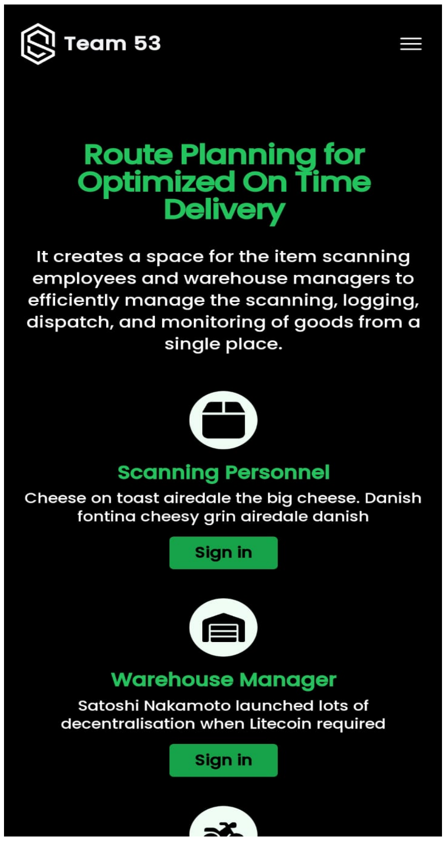
Delivery Riders

1. Each rider should be able to view the route they need to take in order to reach the next delivery/pickup point.
2. Riders will also be intimidated about the expected time within which they should reach that location

4.3 Map APIs Used

1. For Geocoding of Addresses : Google Maps Geocoding API
2. For Routing between two locations : Geoapify Routing API
3. For Map Tiling and Mapping : Geoapify Static Maps API

A Appendix : Application Design


Team 53

Route Planning for Optimized On Time Delivery

It creates a space for the item scanning employees and warehouse managers to efficiently manage the scanning, logging, dispatch, and monitoring of goods from a single place.

Scanning Personnel

Cheese on toast airedale the big cheese. Danish fontina cheesy grin airedale danish

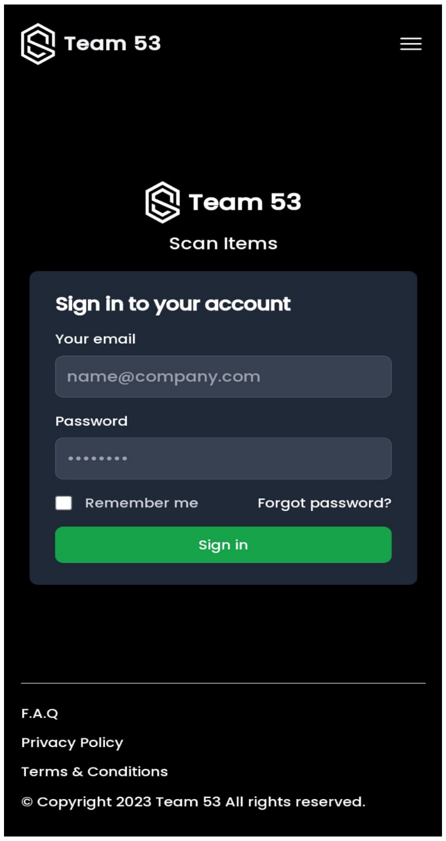
Sign in

Warehouse Manager

Satoshi Nakamoto launched lots of decentralisation when Litecoin required

Sign in

(a) Landing Page


Team 53

Team 53

Scan Items

Sign in to your account

Your email

name@company.com

Password

••••••••

Remember me Forgot password?

Sign in

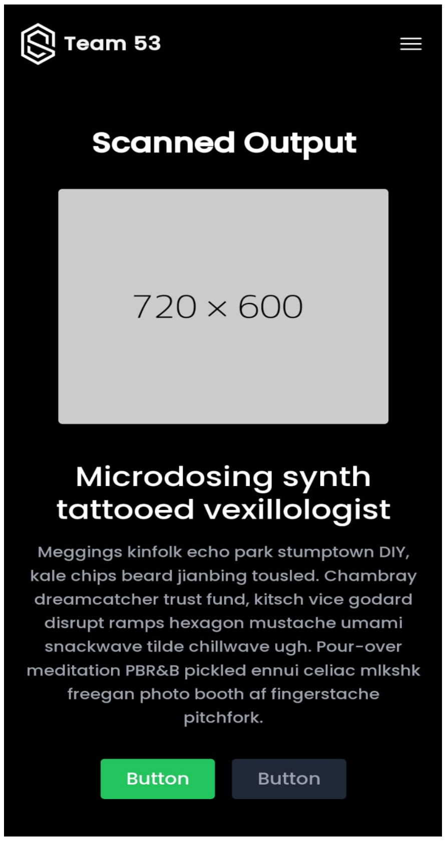
F.A.Q

Privacy Policy

Terms & Conditions

© Copyright 2023 Team 53 All rights reserved.

(b) Login Page


Team 53

Scanned Output

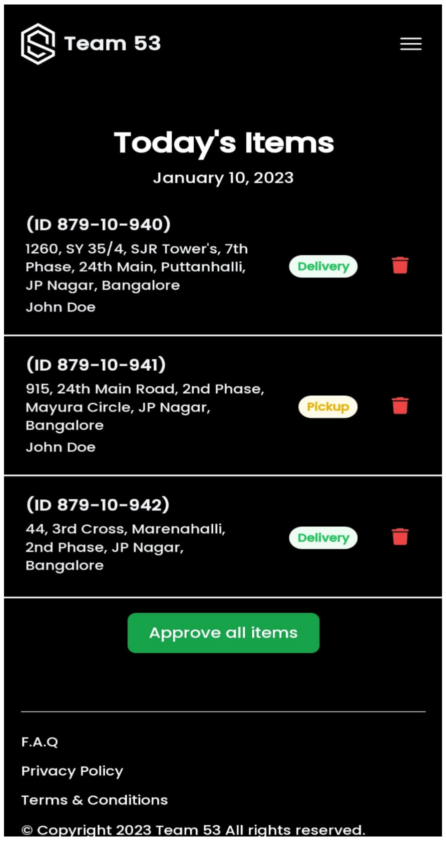
720 × 600

Microdosing synth tattooed vexillologist

Meggings kinfolk echo park stumptown DIY, kale chips beard jianbing tousled. Chambray dreamcatcher trust fund, kitsch vice godard disrupt ramps hexagon mustache umami snackwave tilde chillwave ugh. Pour-over meditation PBR&B pickled ennui celiac mlkshk freegan photo booth af fingerstache pitchfork.

Button Button

(c) Scanner Interface


Team 53

Today's Items

January 10, 2023

(ID 879-10-940)
1260, SY 35/4, SJR Tower's, 7th Phase, 24th Main, Puttanhalli, JP Nagar, Bangalore
John Doe
Delivery

(ID 879-10-941)
915, 24th Main Road, 2nd Phase, Mayura Circle, JP Nagar, Bangalore
John Doe
Pickup

(ID 879-10-942)
44, 3rd Cross, Marenahalli, 2nd Phase, JP Nagar, Bangalore
Delivery

Approve all items

F.A.Q

Privacy Policy

Terms & Conditions

© Copyright 2023 Team 53 All rights reserved.

(d) Warehouse Manger Interface

References

[1] Babette Dellen and Iván Andrés Rojas Jofre. Volume measurement with a consumer depth camera based on structured infrared light. In Proceedings of the 16th Catalan Conference on Artificial Intelligence, poster session, pages 1–10, 2013.

[2] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM, 65(1):99–106, 2021.