T.C.

MARMARA UNIVERSITY

FACULTY of ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CSE4197 Engineering Project I Analysis and Design Document

Title of the Project

Fully Autonomous Supermarket

Group Members

150114027 Özge GÜNAY

150115823 Mert HASKAN

Supervised by

Prof. Çiğdem EROĞLU ERDEM

Table of Contents

1. Introduction

Autonomous marketing technology is a concept for improving shopping experience. Autonomous markets are supported artificial intelligence and machine learning techniques. Almost every day people do shopping and this action takes significant time in their lives. They must spare time to do shopping. In the crowded cities, shopping is a challenging daily task because of long lines at the cashier. Autonomous markets can be a solution for this challenge. In this work, we are planning to implement autonomous market technology.

1.1. Problem Description and Motivation

Nowadays, time is crucial for the people. Hence, the number of time-saving developments are increased. Before the mobile applications, the people had to go to branch offices of banks. The mobile applications created a new option, which is clients can do banking transactions anywhere and anytime. These transactions can be done in a few minutes. Most of the clients do not go to branch offices any more. This shows us that, they prefer a faster way of doing their tasks. Despite these developments, the shopping industry is still using old techniques which causes long lines. There are some improvements such as online shopping. However, online shopping cannot provide reliable shopping experience. The people who do not trust online shopping is forced to wait in long lines in supermarkets. As a result of long waiting times, people might give up on shopping and they must use online shopping to get things done in a short time. This situation causes loss of money to the markets. Also, it turns back as a loss of time and unreliable shopping to the customer.

With the evolution of technology, we believe that the shopping experience can be processed in a better way. We decided to design a shopping experience in which a customer can go to the supermarket and does not have to wait in the line for payment. There are many sectors which are using artificial intelligence and machine learning techniques to control their systems. We also want to use this approach in the shopping experience. Before these developments, technology was insufficient to solve these problems using artificial intelligence and machine learning. The recent developments take forward the progression of these technologies. They make autonomous markets feasible. By using computer vision, faster ways of detecting and tracking objects is possible, and we want to implement those for the shopping system. This system may lead to a new way of shopping in which is no customer has to wait in the line for payment. Instead of a cashier system, the customer can do shopping with a system that tracks the customer movements with one or more cameras and recognizes what the customer purchases automatically.

Figure 1: Long shopping lines. (For more details, the reader is referred to [1]*)*

1.2. Scope of the Project

The implementation of the autonomous market will be composed of a stationary cheap camera, several product shelves containing 15-20 different items. The camera will monitor the shelves continuously and detect a customer. Then, hands of the customer will be detected and tracked. Meanwhile object detection will be carried out around the hands. Detected objects will be tracked. When the object sufficiently far from its original position, then it will be added to the shopping list.

Before the shopping the mobile application will be mandatory for customers to do shopping. The customer will register with an account. After registration, the customer will be monitored, and his/her behavior will be tracked. The purchased items will be added to the account of the customer. When the shopping is finished, the total payment will be deducted from the credit card of the customer and the invoice of the shopping will be sent to the customer.

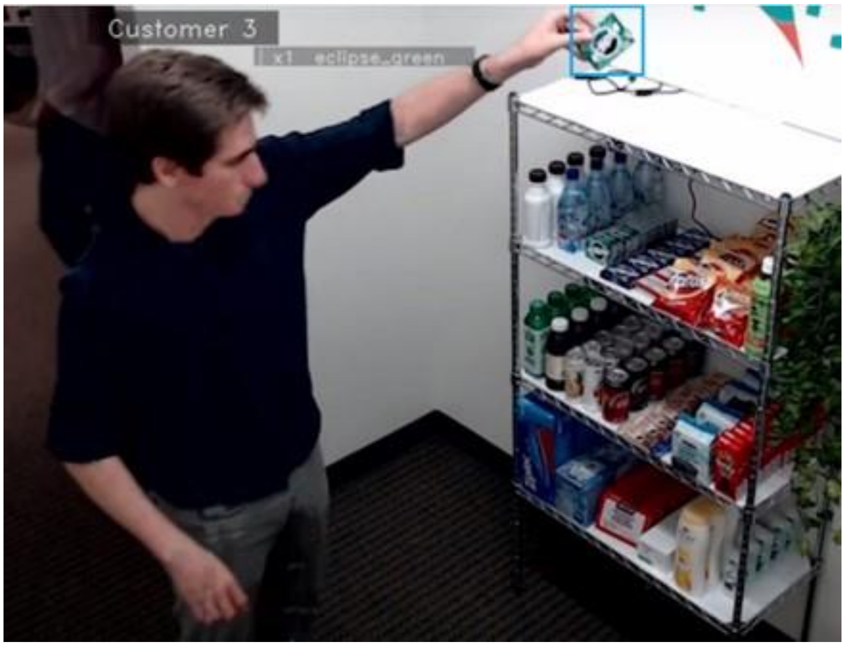


Figure 2: The detection of the product that customer grabs (For more details, the reader is referred to [2]*)*

1.2.1. Constraints:

- The mobile application runs at only Android OS.
- The sytem will track one customer at a time.
- Several items will be between 15 and 20.
- Shopping basket will not be used in this system.
- Customers can just put back items on the shelf only (not on the ground etc.)
- If the customer is not in the viewpoint of the camera, it is assumed that he/she bought all the items in the shopping list.
- The customer should not cover the item entirely.
- The item which will be bought must be in the customer's hands. Otherwise, it will be ignored.
- We assumed that the customer will not be in bad intention (robbery). He/She will always follow the rules.
- It is assumed that the customer will be with bare hands.
- The customer is not allowed to hold items which are not our product during shopping.
- To get the invoice for the shopping, the mobile phone of the customer should be connected to the internet.

1.3. Aims of the Project

Aims of the project are listed below.

- Autonomous supermarket application
  We are planning to develop an application to invoice the shopping list to the customer using camera-based tracking. The accuracy of shopping list creation is planned to be at least 90%.

The following aims are planned for the components of the whole system.

- Object detection and recognition
  Object detection will be used to detect the location of the object that customer grabs. The location of the object will be indicated by a rectangular box drawn around the object. Object recognition is used to classify the object that customer grabs. We need to unite the object detection and recognition for measuring performance of the model. The percentage of locating and identifying the right object (i.e. Label of the object) will be 50% at 0.5 IoU threshold on COCO [3] dataset.

- Object tracking
  Object tracking is used to follow the object that customer grabs. The percentage of tracking the object is 60% on real time.

- Hand detection
  Hand detection is used to find the hands of the customer. The percentage of locating the hand is 60% on Egohands dataset. [4]

1.4. Success Factors and Benefits

1.4.1. Success Factors

- The percentage of locating and identifying the right object (i.e. Label of the object) will be 50% at 0.5 IoU threshold on COCO dataset.
- Object tracking is used to follow the object that customer grabs. The percentage of tracking the object is expected to be 60% on the real-time.
- Hand detection is used to find the hands of the customer. The percentage of locating hands is 60% on Egohands Dataset.
- We would like to obtain a classification accuracy of at least 90% for purchased objects.

1.4.2. Benefits

The benefits of this projects can be listed as below:

- The customers can do shopping without waiting in a line.
- Markets can be run by less human resource.
- Markets' expenses can be decreased.
- The customer can be more satisfied and happier.
- This technological system provides statistical data about a customer's preference.
- Markets can infer which product is demanded most from sales and they arrange their product stocks according to that inference.

1.5. Definitions, acronyms and abbreviations

OS	: Operating System
COCO	: Common Objects in Context
R-CNN	: Region-based Convolutional Neural Networks
CNN	: Convolutional Neural Networks
SVM	: Support Vector Machine
VOC	: Visual Object Classes
mAP	: Mean Average Precision
SSD	: Single-Shot Multibox Detector
YOLO	: You Look Only Once
GOTURN	: Generic Object Tracking Using Regression Networks
FPS	: Frames Per Second
IOU	: Intersection Over Union
ILSVRC	: ImageNet Large Scale Visual Recognition Competition
ALOV	: Amsterdam Library of Ordinary Videos
CPU	: Central Processing Unit
YOLOv3	: Yolo Version 3
UML	: Unified Modeling Language
API	: Application Programming Interface
ICCV	: International Conference on Computer Vision
CVPR	: Conference on Computer Vision and Pattern Recognition
ECCV	: European Conference on Computer Vision

KLT	: Kanade–Lucas–Tomasi Feature Tracker
R-FCN	: Region-based Fully Convolutional Networks

2. Related Work

In the field of the autonomous markets, there are some systems which are currently available to use. At present, Amazon Go [5] is the most successful autonomous market. Amazon Go is supported by many camera systems. The main difference of that market from usual supermarkets is that there are no cashiers and no lines. Amazon Go provides a payment service via a mobile application with which customer can see the shopping list. Aipoly [2] and Standard Cognition [6] are other companies which are working on different approach in this area. These three companies are commercial companies. Therefore, they share limited know-how and techniques about their systems.

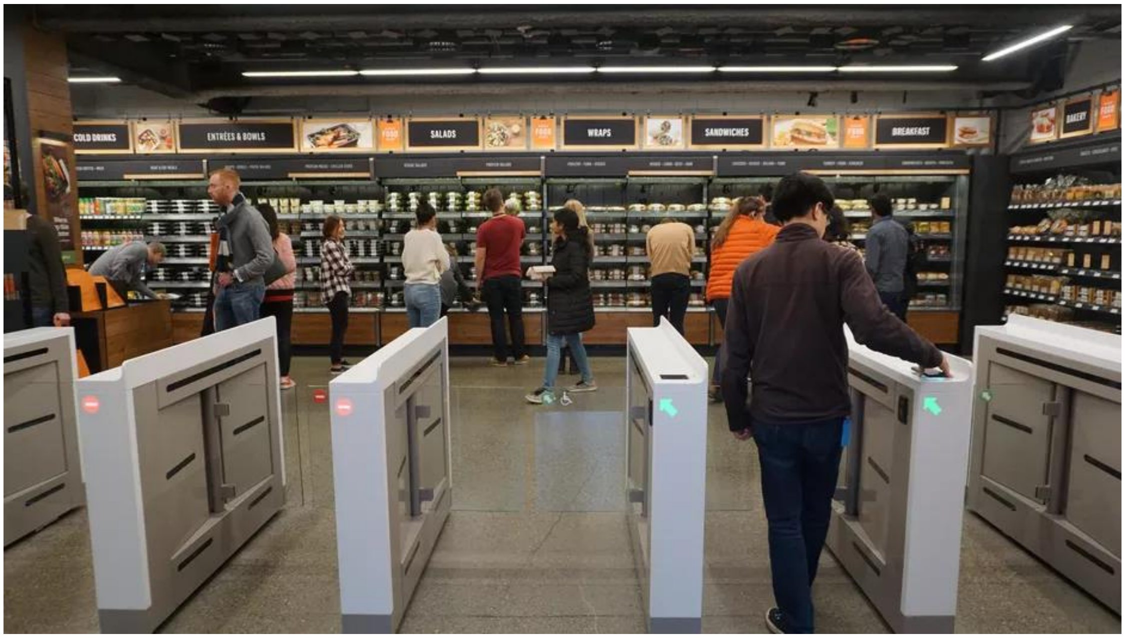


Figure 3: Amazon Go Supermarket (For more details, the reader is referred to [5]*)*

The object recognition is an important part of our project. The studies on this subject are summarized below.

- Rich feature hierarchies for accurate object detection and semantic segmentation (R-CNN)[7]:

  R-CNN is an object detection algorithm that created by combining regions with CNN. Their system consists of four steps. The first one is taking the image as an input, the second one is extracting around 2000 bottom-up region proposals, the third one is computing features for each proposal using a large convolutional neural network (CNN) and final step is classifying each region using class-specific linear SVMs. R-CNN's performance measured on the PASCAL VOC [8] dataset. "R-CNN achieves a mean average precision (mAP) of 53.7% on PASCAL VOC 2010."[7] The proposed detection algorithm has 30% improvement in terms of mean average precision with regard to previous best result on VOC 2012.

System Modules:

- Category-independent region proposals
- A fixed-length feature vector from each region by a large convolutional neural network
- A set of class-specific linear SVMs

- SSD: Single Shot MultiBox Detector [9]

  SSD is a method which uses single a deep neural network. For each object category, the network produces scores in each default box and it does adjustments for the better match the object shape. "SSD achieves 74.3% mAP on VOC2007."[9] SSD eliminates proposal generation and feature resampling stages. It capsulizes all computation in a single network. Systems which necessitate object detection can integrate SSD into their systems and train easily.

- You Only Look Once: Unified, Real-Time Object Detection (YOLO) [10]

  YOLO is a unified object detector which uses the single convolutional network. This convolutional network takes the entire image as an input and predicts each bounding box. As we can understand from the name of the article, this algorithm looks once at an image and predicts locations and types of objects. The system divides input image to an SxS grid and calculates class probabilities. The grid that corresponds to the center of an object is responsible for detecting that object. Each bounding box consists of 5 predictions: x, y, w, h,(x and y are coordinates, w is width and h is height) and confidence score which defines there is an object or not. [10] If there is not an object, the confidence score of the bounding box should be zero. Otherwise, the confidence score is equal the intersection over union (IoU) between the predicted box and the ground truth.[10] After calculating the class probabilities and predicting bounding boxes, YOLO combines these which results to final detections as shown in Figure 4. YOLO can process images at 45 FPS.

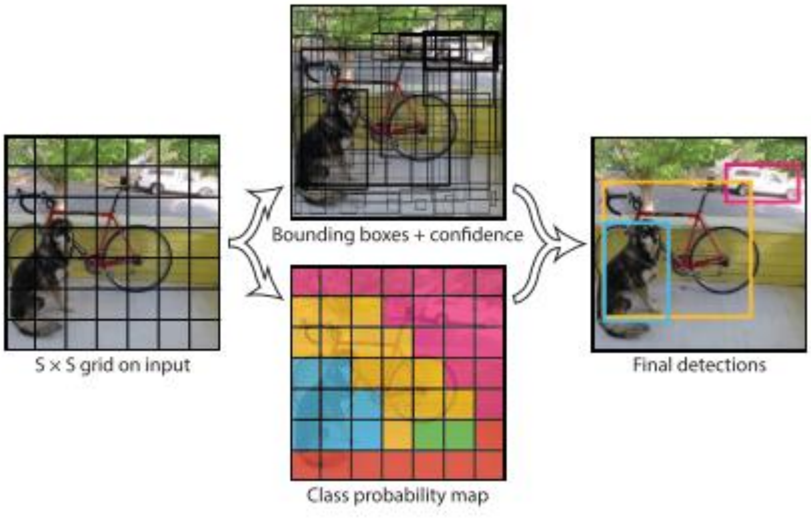


Figure 4: YOLO detection process (For more details, the reader is referred to [8, Fig. 2])

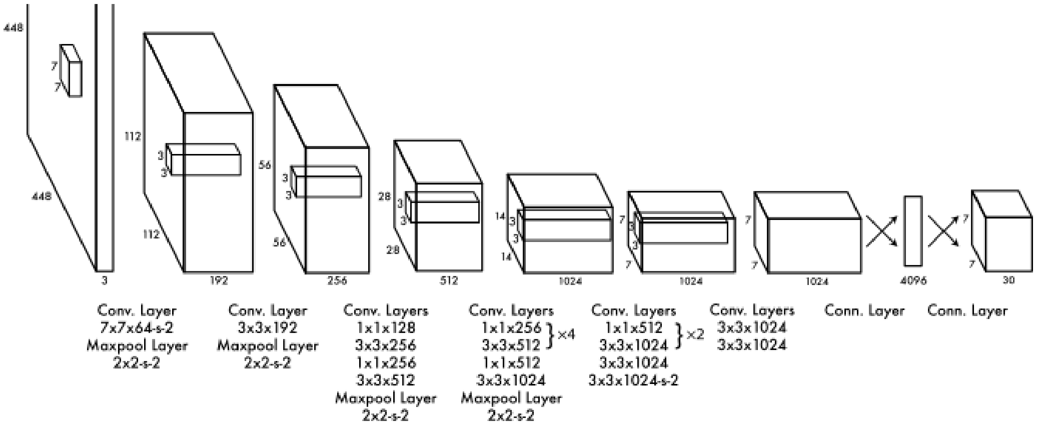
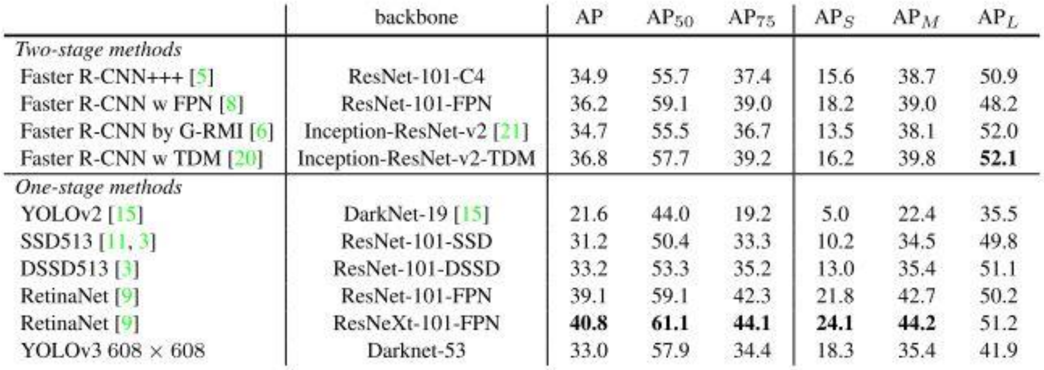


Figure 5: The structure of convolutional layers (For more details, the reader is referred to [8, Fig. 3])

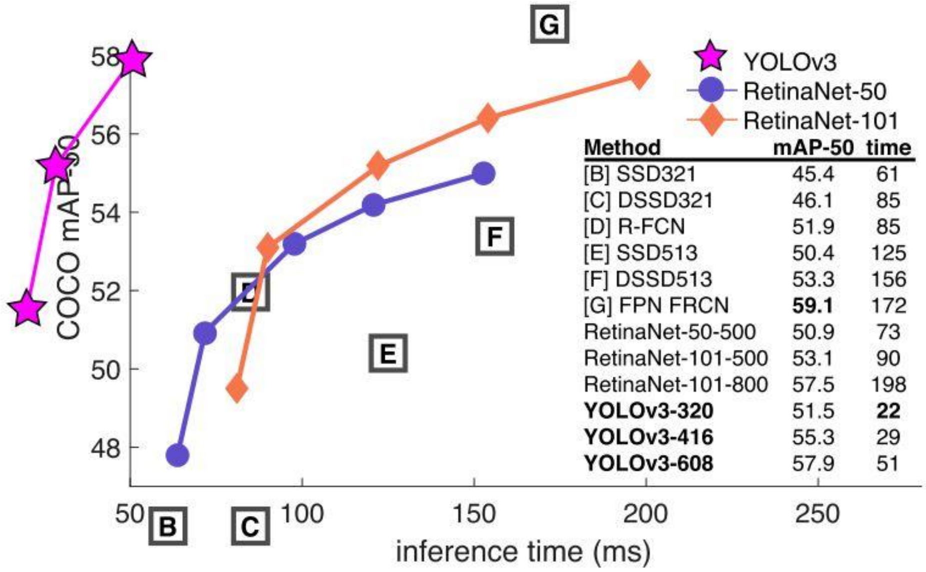
The network which is shown in Figure 5 has 24 convolutional layers and 2 fully connected layers.[10] Fully connected layers are responsible for the recognition and classification of objects. We plan to use COCO dataset for classification of objects. COCO dataset has fundamental object categories. The main problem of YOLO is incorrect localizations. The authors of the article worked on this problem and they presented new versions of YOLO.

The performance values of following versions are shown in Figure 6 and Figure 7.



	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Figure 6: Comparison of methods regarding the backbone and Average Precision. (For more details, the reader is referred to [11], Table. 3)



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Figure 7: Inference time vs. mAP on COCO Dataset. The smaller inference time and larger mAP identifies the better method.(For more details, the reader is referred to [11], Fig. 3)

Hand tracking is another crucial part of our project. Relative study about this topic is given below.

- Building a Real-time Hand-Detector using Neural Networks (SSD) on Tensorflow [12]

  This is the implementation of hand tracking with SSD detector. This implementation is trained on Egohands Dataset. This method provides working only about hands. Neural networks provide to train model that performed well. By using deep learning framework(such as Tensorflow [13]), the process of training a model for custom object detection is simplified. The improvement of neural network models like SSD or Fast R-CNN [14] makes neural networks an attractive applicant for real-time detection. Demonstration of hand tracker is shown in Figure 8.

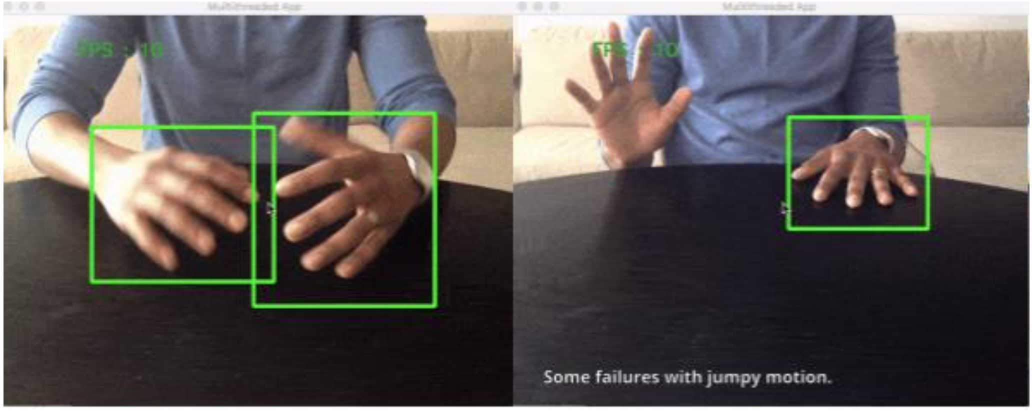


Figure 8: Hand Tracking (For more details, the reader is referred to [12]*)*

- Pose-conditioned Spatio-Temporal Attention for Human Action Recognition [15]

  Human action recognition is a field with many applications which are robotics, automated cars and others. Pose-conditioned spatio-temporal attention method propose two-stream approach. This method uses articulated pose and RGB frames for processing video and recognizing action. The pose stream is done with a convolutional model. The pose and RGB streams are shown in Figure 9.

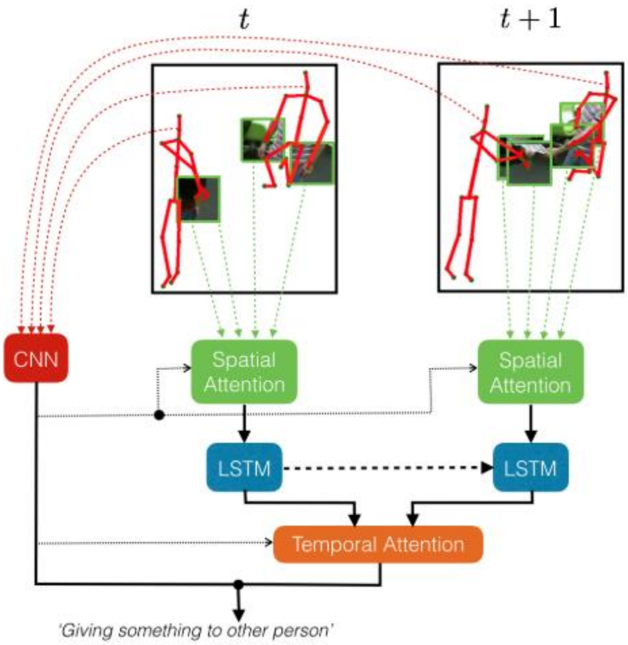


Figure 9: Human Action Recognition (For more details, the reader is referred to [15]*)*

Object tracking is also important method for our project. Relative studies about this topic is given below.

- Re^3: *R*eal-Time *R*ecurrent *R*egression Networks for Visual Tracking of Generic Objects [16]

  Re^3 is object tracker that uses convolutional layers to hold the object appearance and recurrent layers. It uses this information to remember the object appearance and its motion, and regression layer to output the location of the object. Object tracker starts with an initial bounding box. The goal of the tracker is detecting movements and tracking the bounding box continuously. Object tracker must keep tracking bounding box on each video frame. In each frame, tracker must locate the object and update location state of the bounding box to continue the tracking process in the future frames. In each frame, the network is fed a pair of crops from the image sequence. Model has previous and current image crops. When the object moved, it must be in the region of two times of previous bounding box with the reasonable speed and resolution. This process is shown in Figure 10. Re^3 tracks the objects at 150 FPS. In this implementation, Tensorflow is used to train and to test the networks. This model is trained with ILSVRC 2016 Object Detection from Video dataset (ImageNet [17]

Video Dataset) and the Amsterdam Library of Ordinary [18] Videos 300++ (ALOV). The computational power is used in tests that are published in the paper, Intel Xeon CPU E5-2696 v4 @ 2.20GHz and a Nvidia Titan X (Pascal).

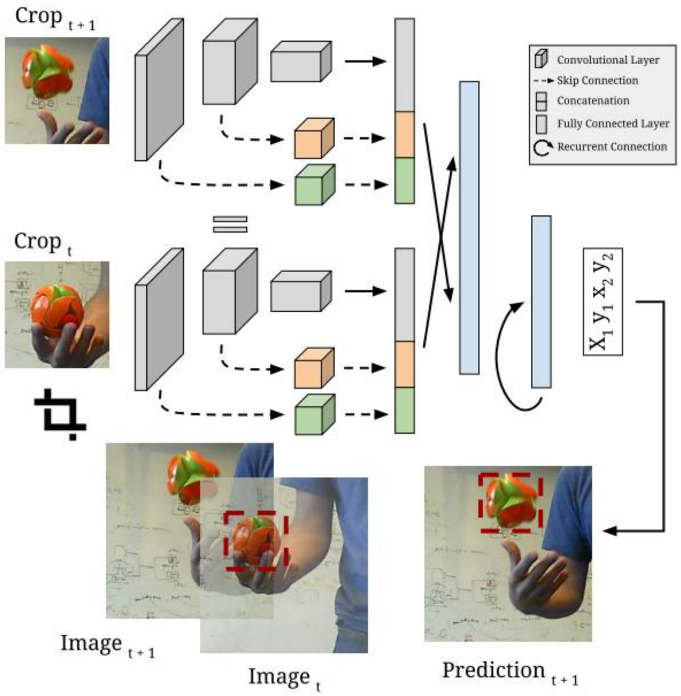


Figure 10: Network structure of the model. (For more details, the reader is referred to [16]*)*

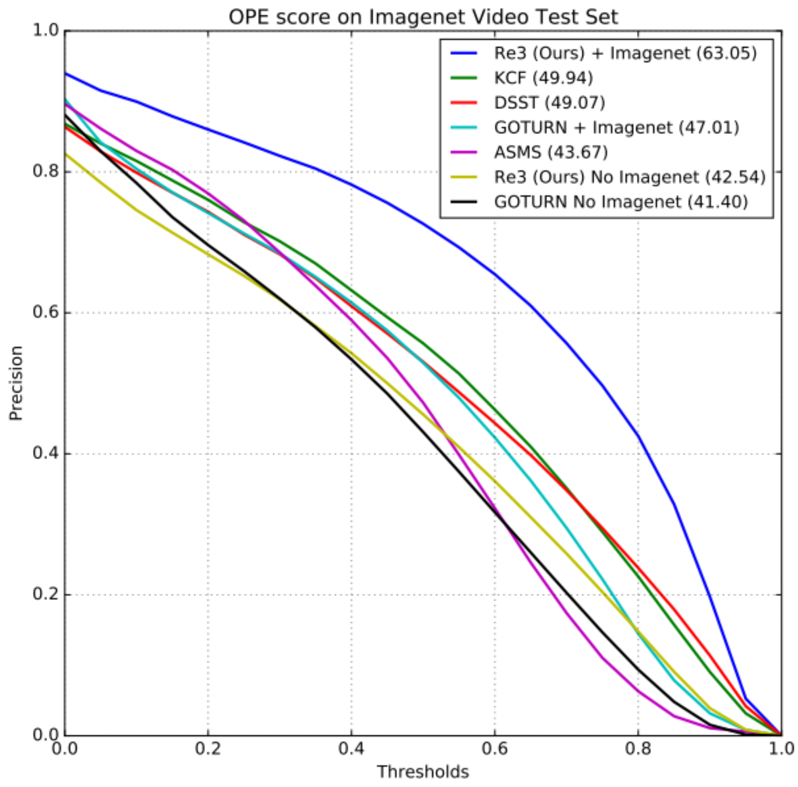


Figure 11: Several trackers evaluated on the ImageNet Video test set. (For more details, the reader is referred to [16]*)*

- Learning to Track at 100 FPS with Deep Regression Networks (GOTURN) [19]

  Generic Object Tracking Using Regression Networks (GOTURN) is a generic object tracker which is not specialized for determined classes of objects. GOTURN is an offline trained model. It learns to track objects from offline videos. GOTURN has a neural network for tracking generic objects in real-time. When this model is used, the network weights are frozen. GOTURN is the first generic object tracker which reaches 100 FPS and uses neural network.

3. System Design

3.1. System Model

When we research about autonomous markets, we come up with an approach that uses object detection and recognition to detect items and classify them, hand detection and tracking to catch the customer's actions. We seek methods to implement object detection which is one of the base components. When we investigated the solution for object detection, we came across many studies on this subject. The prominent studies are YOLOv3 [11], Fast R-CNN and SSD. The comparison of the precision performance of these studies is shown in Figure 6. We examined methods from the perspective of precision. We chose a few methods which are suitable for our system and precision is one of the criteria to choose.

Another important criterion is the inference time in the real-time because our system should be fast and as much precise as possible. Therefore, we should analyze from the viewpoint of inference time. The comparison of inference times of these methods is shown in Figure 7. We considered two aspects. We should choose the model which has the smallest inference time as much as possible and high precision. Although YOLO method is not the most precise method, it is the fastest method and precise enough for our needs. So, there is a tradeoff between the inference time and mean average precision. For this reason, YOLO method is the best fit model for our system, and we chose YOLO as the object detector method.

After we chose our object detector method, we sought a method to detect and track hands. We found two different methods for detecting and tracking hands which are based on human action recognition and real-time hand detection. Human Action Recognition is composed of human pose estimation and hand tracking methods. [15] Human pose estimation is a heavy process which is not necessary for our system. Thus, human action recognition method does not fit our system. Besides, Hand tracking method is only attaching importance to track hands. It is based on SSD with an inference time closed to YOLO. Because of these reasons, we plan to use hand tracking.

For developing a second approach, we looked for object tracking methods to track only objects. We searched tracking methods and we restricted the options into two tracking methods which are Real-time Recurrent Regression Networks for Visual Tracking of Generic Objects (Re^3) and Learning to Track at 100 FPS with Deep Regression Networks (GOTURN). When we compared the methods in terms of precision, we found out that Re^3 has higher precision value than GOTURN as shown in the Figure 11. According to the paper [16], Re^3 runs 150 FPS. It is faster than GOTURN and other tracking methods. It is a hybrid tracker which is composed of online and offline tracker. Online tracker is used for real-time performance. Offline tracker is used for stored videos and it is faster than online trackers, but it has a major problem. Instead of connecting the information from previous layers, it learns similarity between the pairs. In our system, we must work on the real-time. When we consider these performance metrics, Re^3 is a one of the recent successful method. Also, it has higher precision and lower inference time than the other trackers. Proposed accuracy and inference time values match with expectations for our system.

3.1.1. Our Proposed Systems

3.1.1.1. First Proposed System

- Our proposed system will consist of a camera, a product shelf, and 15-20 products. The camera will be set up a viewpoint which is clearly sighted product shelf and customers hands. Our system will take real-time input via the camera. This input will be processed immediately. This action will be the core of the project. In the first approach, we are planning to use object detection, recognition and hand tracking. Hand tracker detects customer's hands on the viewpoint of the camera. If the customer grabs an object, object detector will identify and then the system will add the product to the customer's list. When customer exits from the viewpoint, total payment will be deducted. We are planning to use YOLOv3 for object detection and recognition and Real Time Hand Detection for hand tracking.

The steps of the process are as follows:

- The system tries to detect a person in the input which comes from the camera. In this step, we can use a simplified version of YOLO. This step will continue until the system detects a person in the input.
- If the system detects a person in the input, it tries to detect the person's hands.
- The system tries to detect an object in the region of the hands simultaneously.
- If the object detector detects an object, it classifies that object and system will check the customer's list and the object.
- If the object is already on the list, the system does nothing. Otherwise, the system adds the object to the customer's list.
- If the object detector does not detect an object, the system checks the customer's list.
- If the customer's list is empty, the system does nothing. Otherwise, the system clears the customer's list.
- If the system does not detect a person anymore, the system checks whether there is any unpaid bill or not.
- If there is an unpaid bill, the system sends the invoice to the customer.

3.1.1.2. Second System

If hand tracker shows lower than our expected accuracy, we are planning to use this approach as a plan B. In this approach, we will use object detection, recognition and object tracking. Firstly, the system will recognize the products and tracker will start. Then, tracker can detect movement of the product. According to the direction of the movement, the system will add the product to the customer's list. When the customer exits from the viewpoint, total payment will be deducted. We are planning to use YOLOv3 for object detection and recognition same as the first approach and differently, Real-time Recurrent Regression Networks for object tracker. The steps of the process are as follows:

- The system tries to detect a person in the input which comes from the camera. In this step, we can use a simplified version of YOLO. This step will continue until the system detects a person in the input.
- If the system detects a person in the input, it tries to detect the objects and starts the object tracker for them.
- The object tracker tracks the object.
- If the object tracker detects a movement, the system identified the movement based on direction (towards the product shelves, away to the product shelf).
- The system decides to add product to the customer list or to remove from the customer list.
- If the system does not detect a person anymore, the system checks whether there is any unpaid bill or not.
- If there is an unpaid bill, the system sends the invoice to the customer.

3.2. Flowchart and/or pseudo code of proposed algorithms

3.2.1. First Proposed System

Our first proposed system consists of object detector and hand tracker models. In Figure 12, the system process is visualized by a flowchart. In Figure 13, there is a demonstration of shopping. There is a bottle and single customer. The system detects hands of the customer and tracks over in each frame. When customer grabs the bottle, the system identifies object with using object detector model. At this moment, the object is added to the customer shopping list.

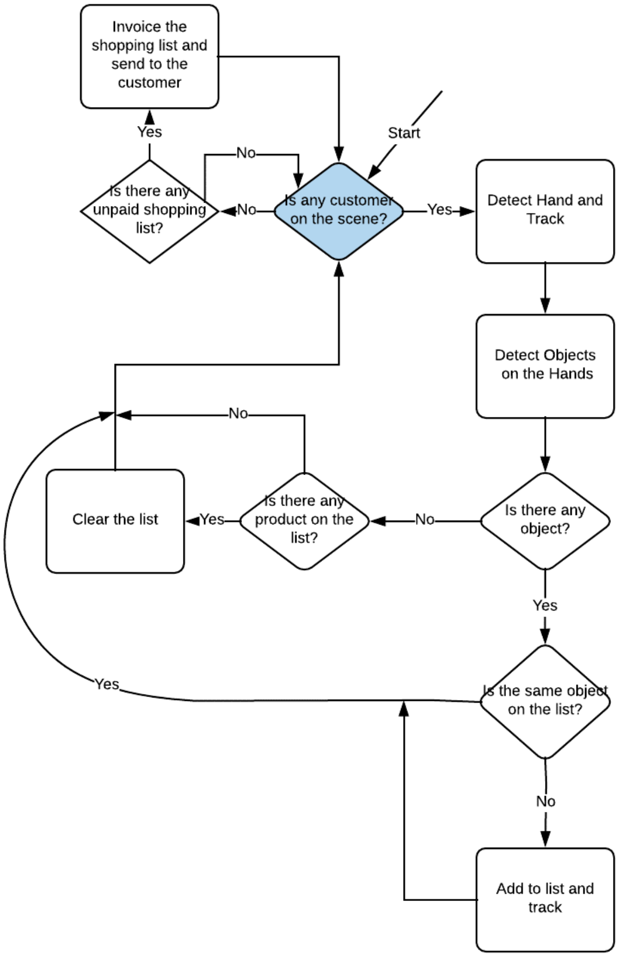

Invoice the shopping list and send to the customer
Yes
Start
No
Is there any unpaid shopping list?
No
Is any customer on the scene?
Yes
Detect Hand and Track
Detect Objects on the Hands
No
Clear the list
Yes
Is there any product on the list?
No
Is there any object?
Yes
Yes
Is the same object on the list?
No
Add to list and track

Figure 12: Flowchart of first proposed system

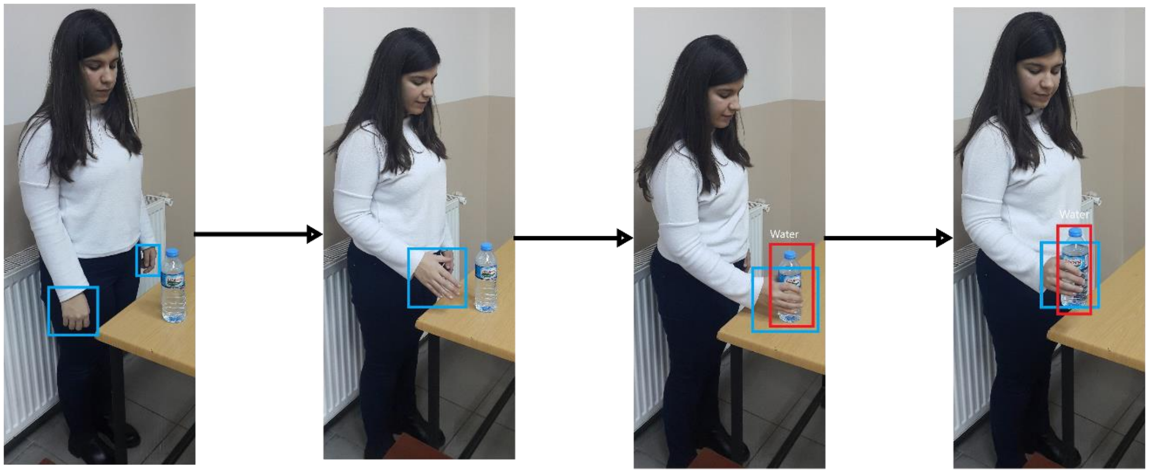


Figure 13: Hand tracking process

3.2.2. Second proposed system

Our second proposed system consists of object detector and hand tracker models. In Figure 14, the system process is visualized by a flowchart. In Figure 15, there is a demonstration of shopping. There is a bottle and single customer. The system identifies object and starts to tracking object over in each frame. When customer grabs the bottle, object tracker detects a movement. The system identifies the movement and takes an action.

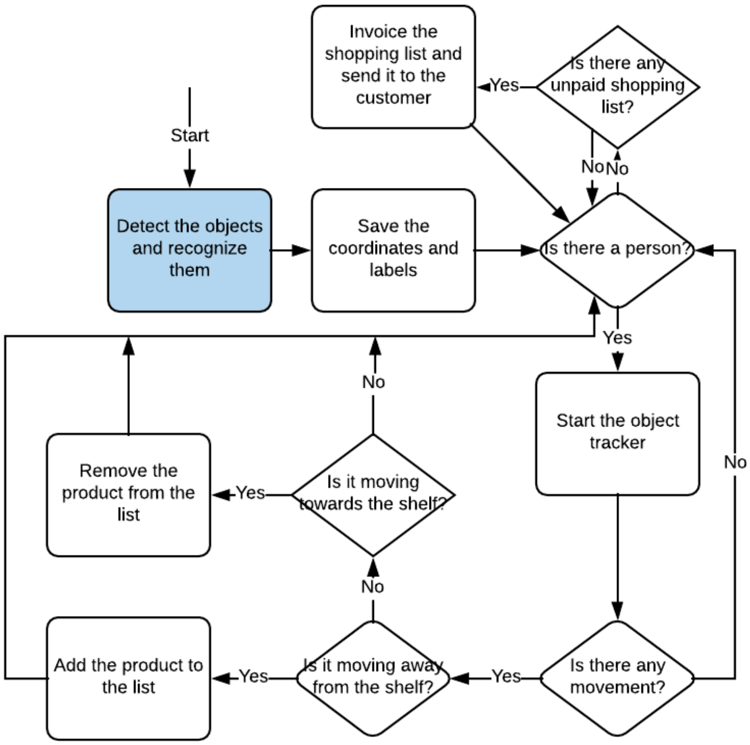

Invoice the shopping list and send it to the customer
Yes
Is there any unpaid shopping list?
Start
No
No
Detect the objects and recognize them
Save the coordinates and labels
Is there a person?
Yes
No
Start the object tracker
No
Remove the product from the list
Yes
Is it moving towards the shelf?
No
Add the product to the list
Yes
Is it moving away from the shelf?
Yes
Is there any movement?

Figure 14: Flowchart of the second proposed system

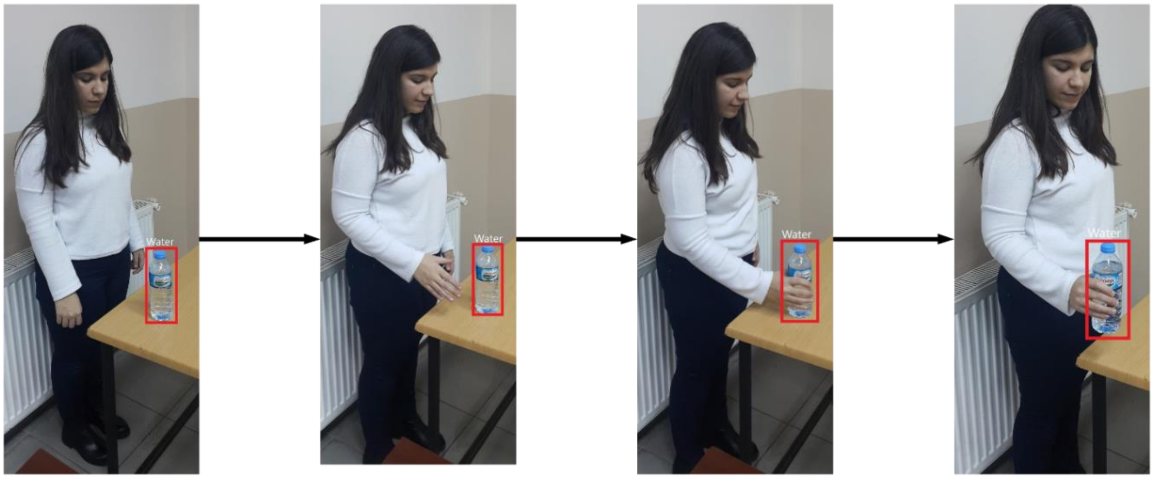


Figure 15: Object tracking process

3.3. Comparison metrics

Our system consists of object detector, object tracker or hand tracker. These methods have same performance metrics. Intersection over union (IoU) and mean average precision (mAP) are the performance metrics for these methods.

3.3.1. Intersection over union (IoU)

IoU measures the overlapping area which is calculated by the ratio of intersection area on an image and union area of our prediction box with ground truth.

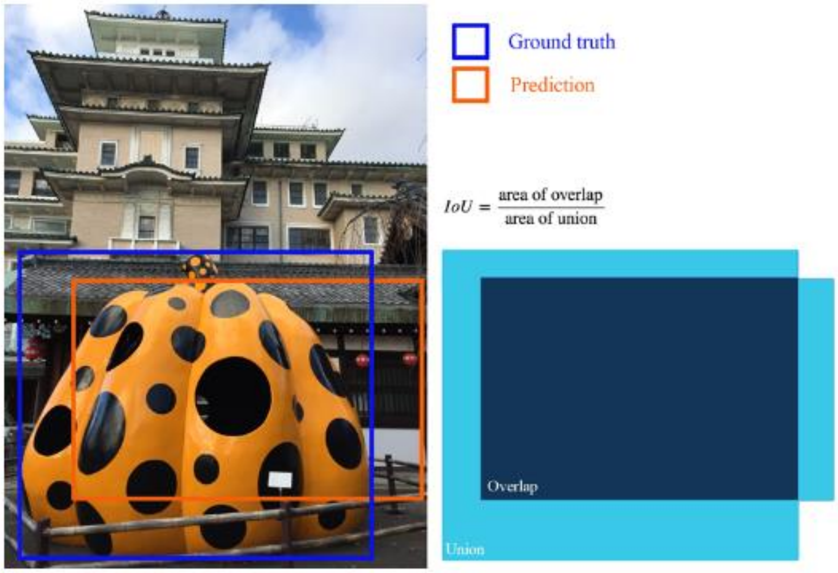


Figure 16: Intersection over Union. (For more details, the reader is referred to [20])

3.3.2. Mean average precision (mAP)

Precision measures accuracy of our prediction based on percentage of positive predictions. We use COCO mean average precision metric. Average precision is the average over multiple IoU values for COCO.

TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

$$\boldsymbol{Precision = \frac{TP}{TP + FP}}$$

3.4. Data sets or benchmarks

We are planning to create a dataset for a supermarket environment. The dataset will be composed of different items. Before working with our dataset, we want to work with a well-known dataset and we want to test our system with the specified dataset. After we created our dataset, we are planning to compare the results that are executed on the specified dataset and our dataset.

There are a few well-known datasets which are COCO, ImageNet and Pascal VOC. ImageNet are provides on average 1000 images to illustrate each synset. Each images are controlled and annotated by human. ImageNet has 21841 synsets. Pascal VOC dataset was created for visual object classes challenge. This challenge occurs between 2005 and 2012. The goal of this dataset is providing training data for supervised learning. COCO is a large-scale object detection dataset. COCO has 81 different object categories as shown in Figure 18. COCO objects are labeled using per-instance segmentations to aid in precise object localization. Some example of segmented objects is shown in Figure 17.

As a well-known dataset, we chose COCO. COCO dataset has enough categories for our system and provides several features to measure the detection performance. We are planning to use Yolov3 and this model is trained on COCO dataset. We chose several items from COCO dataset which are apple, orange, banana, teddy bear, bottle, cup, book and hair dryer. These items are chosen for testing our system. We are planning to create a dataset which includes milk, water, potato chip, shampoo, toilet paper, corn flakes and Nutella. We are planning to test our system with these items, and we will compare the results of COCO and our dataset.

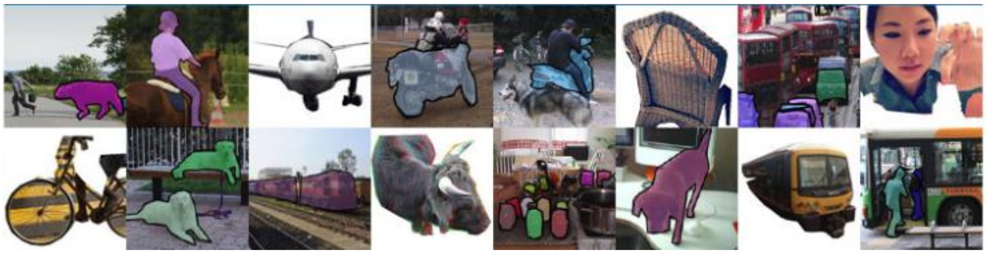


Figure 17: Coco Dataset example images

Figure 18: COCO Dataset Categories

3.5. Professional Considerations

3.5.1. Methodological considerations/engineering standards

We will use Git source code control system during development of the software. Git systems provide coordination and consistency among team members. We decided to use GitHub for repository management. We will create a private repository. The repository's collaborators will be our supervisor, and the team members.

We will use UML diagrams for planning the software architecture. UML diagrams provide agreement between team members and the supervisor. Also, it shows the structure of the software.

We will use Python programming language for the software. It has a large library support about the image processing and neural network areas. It provides simple syntax and easy coding.

3.5.2. Considerations

3.5.2.1. Economical

For markets, human resource is a costly expense. Our project can decrease the expense of markets. It collects customer data and customer’s buying lists. For markets, this system provides statistical data which helps to understand customer demands. With the help of statistical analyze, the markets can arrange their stocks. Also, these projects provide an opportunity which markets can give service to fleeing customer because there is no line anymore.

3.5.2.2. Health and Safety

Waiting on the line is a stressful situation for customers. Stress and anger levels increase in proportion to the waiting time. These are harmful to human health. Our main goal is to create a market without any line.

3.5.2.3. Social

One of the effects of the project is decreasing the stress and anger of customers. This is an element that reduces the fights experienced during shopping. It creates a change to re-establish a peaceful culture of society.

3.5.2.4. Ethical

The resources that we are planning to use are open source. There is no restriction to use for our project. We chose the libraries in our project by considering ethical concerns.

3.5.2.5. Sustainability

The system that we are planning to implement is maintainable. The fine-tune process is done once. After fine tune, the system can work without any change until the market needs to add new products.

3.5.3. Legal considerations

There is no legal issue for the projects. The researches which we used to base level of our project are free to use licenses. The databases we will use are publicly available for research purposes.

3.6. Risk Management

The first major risk is precision of object detection and tracking algorithm of first approach which can cause the whole system to fail. We proposed a second approach to eliminate the risk of failure.

The second major risk is real-time performance. Algorithms that we used require high calculation capacity. Originating from these algorithms, delays can occur which our system is an intolerance to them. This situation can cause to fail of our system. We will try to minimize these risks by avoiding unnecessary computations.

4. System Architecture

The system has two components which are object detector and hand tracker as seen in Figure 19. The system takes input from camera. Camera input is processed by object detector and hand tracker. This process is shown in Figure 12. The system output which is customer’s invoice is sent to our Web API. Web API is responsible of communication between mobile application and database. Also, it provides communication between database and proposed system.

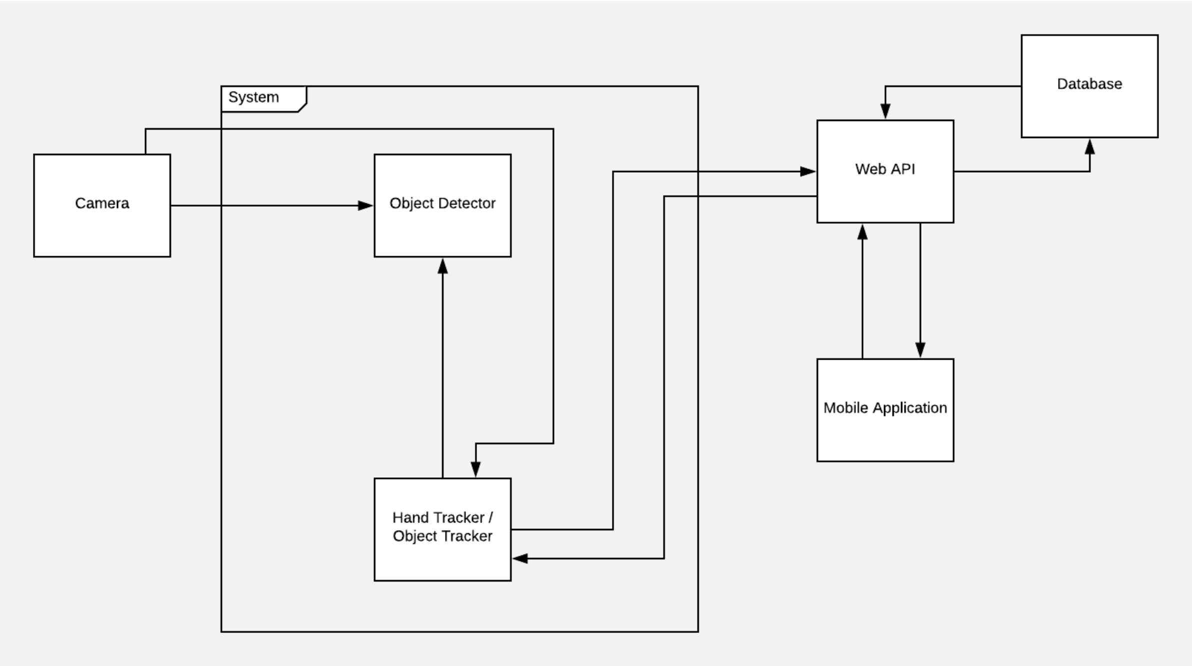


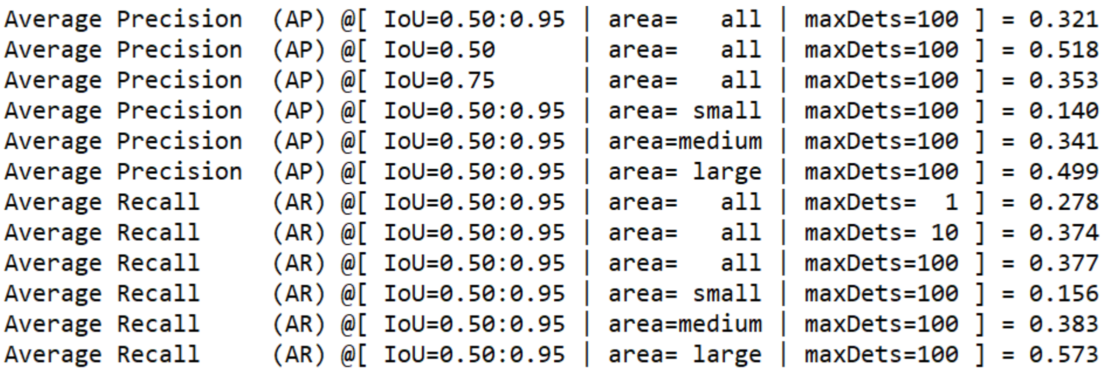
Figure 19: Integration of the system components

5. Experimental Study

5.1. Completed Experiments

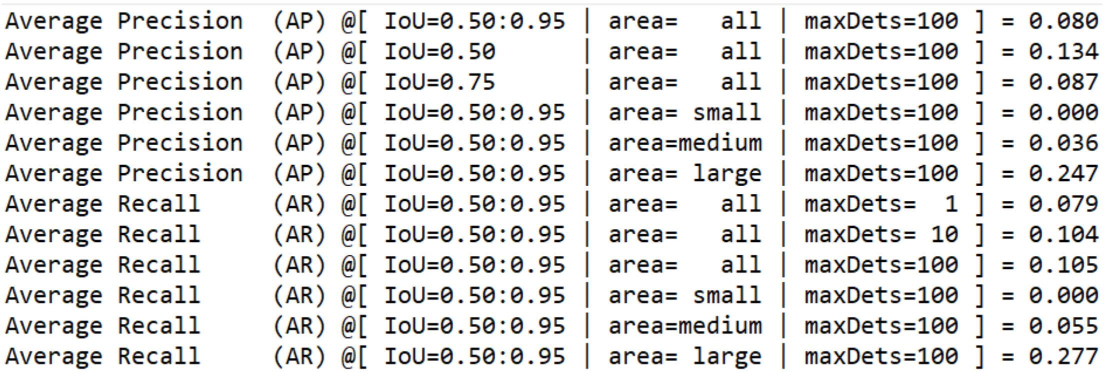
Experimental Setup: We prepare a test for object detector on our computer. We chose Yolov3 and Yolov3-tiny for testing models. These models are trained on COCO dataset. COCO shares some image and annotation sets for training, validation and test. These annotations hold information about images. There are 8 image and annotation sets which are published. We chose 2014 validation set for testing models and we pick randomly 2000 images. We want to analyze precision and inference time of models.

Experimental Results: When we test models with 2000 images, average precision of Tiny-Yolo with the IOU@0.5 is 0.134 and average precision of YOLOv3 with the IOU@0.5 is 0.518. We expected higher values than these results. We decided to test models on smaller image set. Results for testing with 500 images, average precision of Tiny-Yolo with IOU@0.5 is 0.134 and average precision of Yolov3 with the IOU@0.5 is 0.518. So, nothing is changed. The other criterion is speed. When we look at to the speed values of models, Tiny-Yolo is 4 times faster than Yolov3 in terms of Average time per image, but average precision of Tiny-Yolo is 4 times lower than Yolov3. We can say that there is a tradeoff between speed and precision. The average precision of Tiny-Yolo should be 33 mAP, but we get 13 mAP on our tests. Yolov3 has quite close results for the given mAP on the paper.



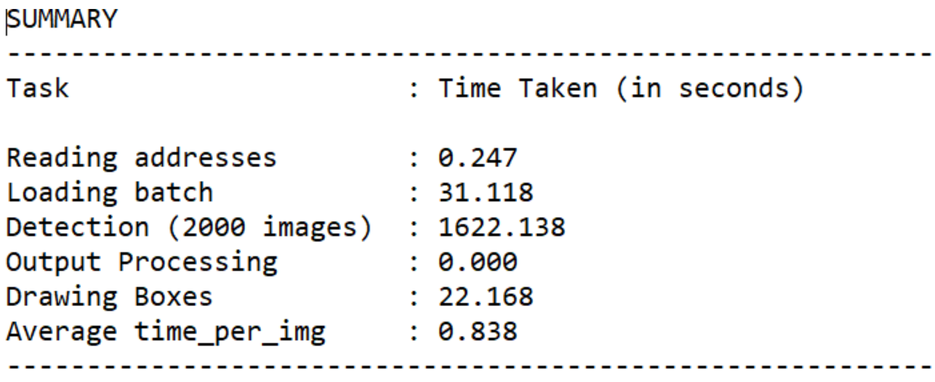
```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.321
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.518
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.353
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.140
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.341
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.499
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.278
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.374
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.377
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.156
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.383
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.573
```

Figure 20: Average Precision and Recall values of Yolov3 testing with 2000 images



```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.080
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.134
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.087
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.036
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.247
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.079
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.104
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.105
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.055
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.277
```

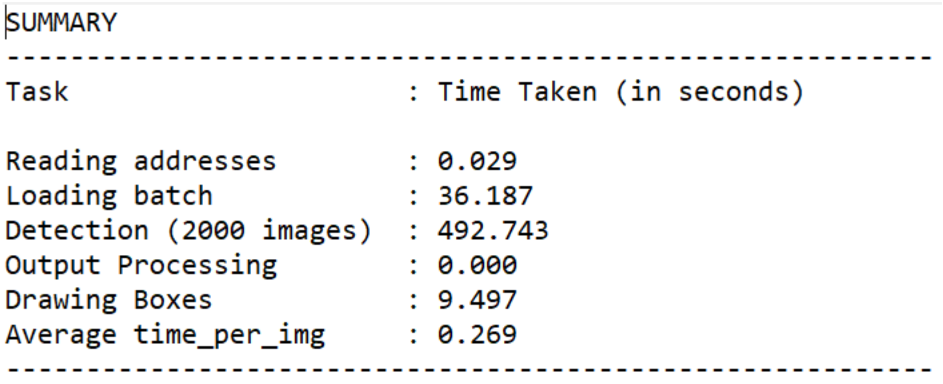
Figure 21: Average Precision and Recall values of Yolov3-tiny testing with 2000 images



```
SUMMARY
----------------------------------------------------------
Task                         : Time Taken (in seconds)

Reading addresses            : 0.247
Loading batch                : 31.118
Detection (2000 images)      : 1622.138
Output Processing            : 0.000
Drawing Boxes                : 22.168
Average time_per_img         : 0.838
----------------------------------------------------------
```

Figure 22: Inference times of Yolov3



```
SUMMARY
----------------------------------------------------------
Task                         : Time Taken (in seconds)

Reading addresses            : 0.029
Loading batch                : 36.187
Detection (2000 images)      : 492.743
Output Processing            : 0.000
Drawing Boxes                : 9.497
Average time_per_img         : 0.269
----------------------------------------------------------
```

Figure 23: Inference times of Yolov3-tiny

Discussions: According to the results that we measured, the performance of Yolov3 is close to expected precision. In terms of speed which is inference time, Yolov3 should be 22 ms but we got 1622 ms on our test. This result might be affected from our personal computers computational power. On the other hand, the performance of Tiny-Yolo is one-third of expected precision. In terms of speed, Tiny-yolo should be 5 ms but we got 492 ms on our test. Between two models, the ratio is 4 as expected. In conclusion, Tiny-Yolo has some precision issues and it can fail in the precision critical systems. Yolov3 has good prediction performance but it can fail on the real-time systems without high computational power.

5.2. Future Experiments

Experimental Setup: We are planning to perform test for our system. We choose items from COCO dataset which are apple, orange, banana, teddy bear, bottle, cup, book and hair dryer. We are planning to prepare test videos which includes these objects. This videos demonstrates our market shopping process.

Experimental Expectations: We are planning to get our system performance on the well-known dataset. We want to match the result with the project aims. After fine-tune operation, we are planning to do test with our dataset and comparing results with this experiment which is tested on COCO dataset.

6. Task Accomplished

6.1. Current state of the project

The completed tasks are shown below:

- Literature Survey for the components are done.
- Related works are examined.
- The well-known dataset is decided.
- We started to implement object detector model.
- We started to create test videos.
- We accomplished object detector test on COCO dataset.
- The categories of products are specified.
- Image labelling tools are searched.

6.2. Task Log

Meeting#1
Date: 19.09.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Research about frameworks

Decisions and Notes:

- PSD preparation should start.
- Deciding project specifications. Considered specifications are listed below:
    - What is the scope of mobile application?
    - Does face recognition should be a part of this system?
    - How many customers can do shopping?
    - Does shopping basket should be in this project?
    - How object alignment should be?
- Talking about project development and testing process. Processes are listed below:
    - Object dataset should be specified.
    - Human-pose dataset should be specified.
    - Video test set should be prepared.
- Aim formats and Aims are determined. Format and Aims listed below:
    - Object recognition performance on a well-known dataset: %xx
    - Face recognition performance on a well-known dataset: %xx
    - Human pose estimation performance on a well-known dataset: %xx
    - Object tracking performance?

Meeting#2
Date: 27.09.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Research topics
Decisions and Notes:

- Similar problems. You may mention about it in the motivation part. Similar problems:
- Search Web of Science: object recognition, object detection, human pose estimation
- OpenCV [21] with python
- Research frameworks:
    - TensorFlow Detection API
    - Caffe2 [22] Detection
- Research papers:
    - Object detection at 200 Frames Per Second Rakesh, Rakesh Mehta, 2018
    - YOLO9000: Better, Faster, Stronger Joseph Redmon, 2016
    - Girshick, “Fast R-CNN”, ICCV 2015

- Redmon et al, “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016
- Liu et al, “SSD: Single-Shot MultiBox Detector”, ECCV 2016
- Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, Zhe Cao

Meeting#3
Date: 03.10.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Project parts discussion
Decisions and Notes:

- Responsibilities are divided as below:
  - Özge will be responsible object detection and recognition
  - Mert will be responsible human pose estimation
- Tasks are defined as follows:
  - Fine tune will be searched for object detection
  - VGG [23] human pose estimation will be searched.
- Cloud computers for training models:
  - Google colab
  - Amazon deep learning servers

Meeting#5
Date: 11.10.2018
Location: Online
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Accomplished tasks and its outcomes
Decisions and Notes:

- Given papers discussed.
- The performance test of object detectors is discussed.
- Pose estimation implementation and performance results are discussed.

Meeting#6
Date:24.10.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Tracking methods discussion

Decisions and Notes:

- KLT tracker will be researched for tracking.
- Object tracking performance metrics should be researched.

Meeting#7
Date:31.10.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: To do
Decisions and Notes:

- Object detection performance is discussed.
- Starting to design test for object detector.
- Object detection fine tune process is discussed and starting to design fine tune process.
- Camera requirements should be defined.

Meeting#8
Date: 07.11.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Object detector test
Decisions and Notes:

- Image labelling tools should be search for preparing test set for object detector. Data augmentation for image classification can be suitable.
- Object detector test format and test plan are discussed.

Meeting#9
Date: 14.11.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Video test for object tracker
Decisions and Notes:

- Testing with camera should start for object detector.
- COCO and Pascal VOC dataset should search.
- Image labeler for video should search.
- Test video format is discussed. Video format should be as follows:

  - 10-15 different video.
  - Grabbing and release movement should be included.

Meeting#10
Date: 28.11.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Test video analyses
Decisions and Notes:

- Test video's viewpoint is confusing for object tracker. So, new videos should prepare with more restricted viewpoint.
- Test categories are specified. The categories are listed below:
  - Apple
  - Orange
  - Banana
  - Teddy bear
  - Bottle
  - Cup
  - Book
  - Hair dryer
- Mean average precision should search.
- Analysis and design document template are examined. Preparation should start.

Meeting#11
Date: 05.12.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Test video analyses
Decisions and Notes:

- Project presentation should be prepared
- System should be tested before fine tuning
- Hand tracking implementation errors are discussed. Risk analyses should be done for this problem.
- Object tracking can be plan B.
- Object tracking should be researched. Some methods are listed below:
  - Learning to track at 100fps
  - Re^3: Real-Time Recurrent Regression Networks

- R-FCN should be researched and be analyzed for object detector.

Meeting#12
Date: 13.12.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Project presentation
Decisions and Notes:

- Project presentation is analyzed.

Meeting#13
Date: 19.12.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Project presentation
Decisions and Notes:

- Final check of project presentation.

Meeting#14
Date: 26.12.2018
Location: MB646
Period: One week
Attendees: Özge GÜNAY, Mert HASKAN
Objectives: Project presentation
Decisions and Notes:

- Analysis and design document topics are discussed.

6.3. Task Plan with Milestones

The Gannt Chart is given in Figure 24. There are three milestones. The first milestone is Fine-tuning of object detection model. When we complete the system component, the first milestone is achieved. Second milestone is testing our system on real data inputs which comes from our dataset. The last milestone is testing our system with the supermarket environment that we created.

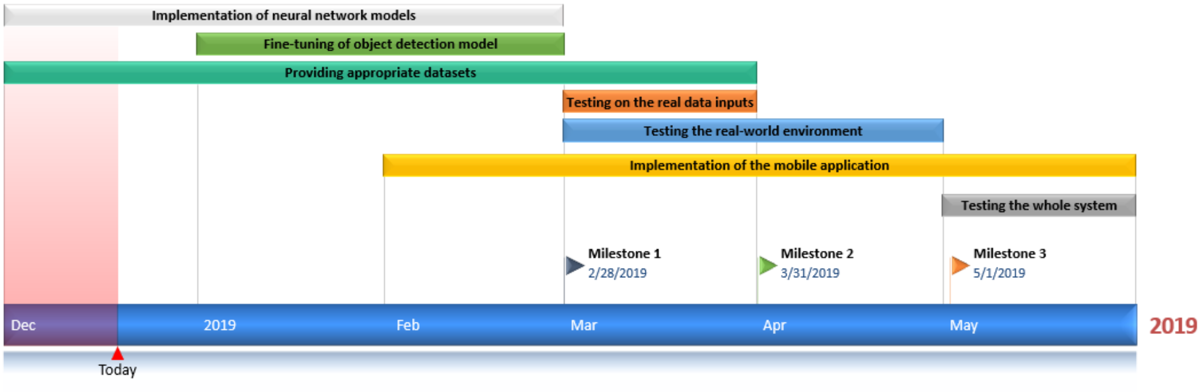


Figure 24: Gantt Chart with Milestones

- Implementation of neural network models:
  We are planning to implement neural networks of our system components.

- Fine-tuning of object detection model:
  We are planning to fine-tune our object detection model to recognize specified products.

- Providing appropriate datasets:
  The system will be tested on COCO dataset. After the testing on COCO, we are planning to create our dataset with specified products.

- Testing on the real data inputs:
  We are planning to create new data inputs from constituted supermarket environment.

- Testing the real-world environment:
  This phase is testing our system on the real-world environment.

- Implementation of the mobile application:
  For our system, we are planning to develop a mobile application.

- Testing the whole system:
  When all phases are done, we will test our system on constituted environment on real-time.

6.3.1. Division of responsibilities and duties among team members

Özge is responsible for object detection and recognition part. Mert is responsible for hand detection and tracking part. The implementation of mobile application and tests will be done together.

7. REFERENCES

[1] “İzmir Haberleri - CarrefourSA’dan, Balçova’ya 7,5 milyon liralık yatırım - Yerel Haberler.” [Online]. Available: http://www.hurriyet.com.tr/carrefoursadan-balcovaya-7-5-milyon-liralik-40686659. [Accessed: 17-Dec-2018].

[2] “Aipoly - Fully Autonomous Markets.” [Online]. Available: https://www.aipoly.com/. [Accessed: 21-Oct-2018].

[3] T.-Y. Lin *et al.*, “Microsoft COCO: Common Objects in Context,” 2014.

[4] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 1949–1957, 2015.

[5] “Amazon.com: : Amazon Go.” [Online]. Available: https://www.amazon.com/b?ie=UTF8&node=16008589011. [Accessed: 21-Oct-2018].

[6] “Standard Cognition - AI-powered checkout.” [Online]. Available: https://standard.ai/. [Accessed: 21-Oct-2018].

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 580–587, 2014.

[8] “The PASCAL Visual Object Classes Homepage.” [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/. [Accessed: 10-Jan-2019].

[9] W. Liu *et al.*, “SSD: Single shot multibox detector,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” 2015.

[11] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” 2018.

[12] D. Victor, “Real-time Hand Tracking Using SSD on Tensorflow ,” *GitHub repository*. GitHub, 2017.

[13] “TensorFlow.” [Online]. Available: https://www.tensorflow.org/. [Accessed: 21-Oct-2018].

[14] R. Girshick, “Fast R-CNN,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 1440–1448, 2015.

[15] F. Baradel, C. Wolf, and J. Mille, “Pose-conditioned Spatio-Temporal Attention for Human Action Recognition,” 2017.

[16] D. Gordon, A. Farhadi, and D. Fox, “Re3 : Real-Time Recurrent Regression Networks for Visual Tracking of Generic Objects,” pp. 1–8, 2017.

[17] “ImageNet.” [Online]. Available: http://www.image-net.org/. [Accessed: 10-Jan-2019].

[18] “Dataset Resources.” [Online]. Available: http://alov300pp.joomlafree.it/dataset-resources.html. [Accessed: 10-Jan-2019].

[19] D. Held, S. Thrun, and S. Savarese, “[GOTURN] Learning to Track at 100 FPS with Deep Regression Networks (ECCV 2016).pdf.”

[20] “mAP (mean Average Precision) for Object Detection – Jonathan Hui – Medium.” [Online]. Available: https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173. [Accessed: 20-Nov-2018].

[21] “OpenCV library.” [Online]. Available: https://opencv.org/. [Accessed: 20-Nov-2018].

[22] “Caffe2 | A New Lightweight, Modular, and Scalable Deep Learning Framework.” [Online]. Available: https://caffe2.ai/. [Accessed: 10-Jan-2019].

[23] “Very Deep CNNS for Large-Scale Visual Recognition.” [Online]. Available: http://www.robots.ox.ac.uk/~vgg/research/very_deep/. [Accessed: 10-Jan-2019].