DECLARATION

I, **Nihal Nihalani**, hereby declare that this dissertation work entitled **"Smart Garbage Collection"**, has been carried out under the guidance of **Prof**. **Nita K**, Professor, ***Computer Science and Engineering department, S.D.M College of Engineering & Technology, Dharwad,*** in partial fulfillment of the degree of ***Bachelor of Engineering in Computer Science and Engineering*** *from* ***Visvesvaraya Technological University, Belgaum***, during the academic year ***2017-18.***

I also declare that I have not submitted this dissertation to any other University for the award of any other degree.

Place: Dharwad.	NAME: Nihal Nihalani
Date: 26/05/2018	USN: 2SD14CS065
	Signature:

ACKNOWLEDGEMENT

I consider it a privilege to express my sincere gratitude and respect to all those who guided and inspired me throughout this project.

I express my heartfelt thanks to my guide **Prof. Nita K,** Professor**,** Department of Computer Science and Engineering, S.D.M College of Engineering and Technology, Dharwad, for his/her valuable guidance and suggestions during the course of this project. The successful completion of this project, owes to his/her coordination and streamlining of the project progress.

I extend my gratitude to our Project Coordinator(s) **Dr. S M Joshi, Dr. U P Kulkarni, Prof. Ranganath G Yadawad, and Prof. Indira R Umarji,** Department of Computer Science and Engineering, S.D.M College of Engineering and Technology, Dharwad, for their valuable guidance and suggestions during the course of this project.

I am thankful to **Dr. S. M. Joshi,** Head of the Department, Department of Computer Science and Engineering, SDM College of Engineering and Technology, Dharwad, for his encouragement and help throughout the BE course.

I acknowledge gratitude to **Dr. Shrikant B. Vanakudre,** Principal, S.D.M College of Engineering and Technology, Dharwad, for his timely help and inspiration during the tenure of the course.

I would like to thank all teaching and non-teaching staff of CSE department, SDMCET, for helping me to carry out my project successfully.

Last but not the least I am very much thankful to all my family members, classmates and friends for their valuable support during the period of my study.

Name: Nihal Nihalani

USN: 2SD14CS065

ABSTRACT

Many times, in our city we see that the garbage bins or dustbins placed at public places are overloaded. It creates unhygienic conditions for people as well as ugliness to that place leaving bad smell. To avoid all such situations we are going to implement a project called Smart Garbage collection system.

We have designed and developed a prototype of autonomous robots which can segregate and collect recyclable plastic waste from land.. The prime technologies used include Internet of Things (IoT), Tensorflow, Digital Image/Video Processing, Machine Learning and Robotics. As we know, recycled plastic produces about 3.5 kg less CO2 compared to 6 kg of CO2 for new plastic and 6,137 tons/day waste remain uncollected and littered across India viz. Delhi- 690, Chennai- 430, Kolkata-425, Mumbai-408 ton/day. After collecting the waste, it can be sent to relevant industries for recycling. We believe these robots assist in realizing our Government's Swatch Bharat Abhiyan and Make in India schemes along with implementing Smart Cities

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

PROBLEM STATEMENT:

The main problem that is being addressed here is collection and segregation of the recyclable waste (garbage) which is usually littered and uncollected on the roads, rivers and railway tracks in many developing and underdeveloped countries. As we know that, it is a time consuming task and more over a tedious task to segregate the waste manually, as a result there is a huge impact on the environment by the harmful elements in the waste like the plastic, which is a main cause for environment pollution.

The traditional methods employed by the organizations have the following limitations:

1. The garbage keeps lying near bins and on roads.

2. It serves as a breeding ground for stray dogs and various animals.

3. The collection mechanism is not efficient and it is a tedious job.

The collection will be done by a robot, so that the garbage does not keep lying near the bins or roads. It will be an efficient system and will save huge amount of fuel.

CHAPTER 1

INTRODUCTION

Solid waste management is a big challenge in urban areas for most of the countries throughout the world. An efficient waste management is a prerequisite to maintain a safe and green environment as there is increase in all kinds of waste disposal. Many technologies are used for waste collection as well as for well managed recycling. The concurrent effects of a fast national growth rate, of a large and dense residential area and a pressing demand for urban environmental protection create a challenging framework for waste management. The complexity of context and procedures is indeed a primary concern of local municipal authorities due to problems related to the manual collection, transportation and processing of residential solid waste, which takes a lot of effort and is time consuming.

Figure 1. Garbage Affecting the Animal Life(left), Garbage on Land (right)

Internet of Things (IoT) interconnects people, machines, and sensors throughout the world. IoT has paved the way for smart cities, smart environment, smart water system etc. IoT has evolved from the convergence of wireless technologies, micro-electromechanical systems and the Internet. By collaborating IoT with robotics and machine learning aspects, we can construct a system that has the potential to bestow intelligence to the things and enabling them to diagnose any glitches in their operational status and upon detection of the same, giving them the capability to resolve it without human intervention.

Over the last twenty years robots have been increasingly used in many specialized application areas and industrial processes from medicine to military and manufacturing. More recently, increasing attention is being given to the use of robots in social contexts, as their use is becoming more widespread. Researchers and engineers have mostly been focusing on how to make the robots smarter, but there is now a shift to thinking about the automatic garbage collection and segregation with the help of robots.

The waste segregation is not only the responsibility of the government organizations like the municipalities but many rag-pickers and few NGOs like I Got Garbage and private organizations like 'ExJunk' in India, DustCart, an autonomous robot for door-to-door garbage collection in Peccioli, Italy are also concerned. In India daily around 6,137 tons remain uncollected and littered. By 2025, the world will produce garbage at a rate of 2.5 billion tons per year. Poor waste management in many developing countries is a threat to human health and the environment. Fig 3 shows the typical garbage share.

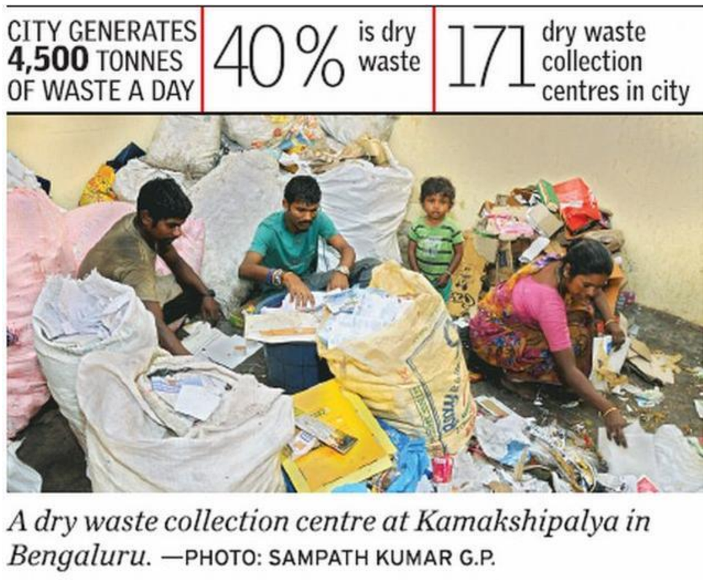

CITY GENERATES 4,500 TONNES OF WASTE A DAY
40 % is dry waste
171 dry waste collection centres in city
A dry waste collection centre at Kamakshipalya in Bengaluru. —PHOTO: SAMPATH KUMAR G.P.

Figure 2. Segregating Waste by Rag Pickers

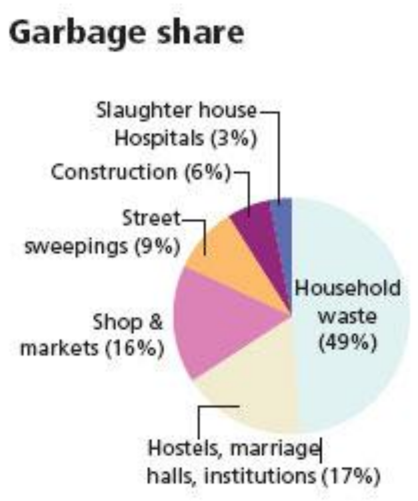


Figure 3. Garbage Share

CHAPTER 2

LITERATURE SURVEY

2.1. Referred Papers for robot design and existing automated garbage collection methods

- *A State of the Art review on Internet of Things by P. Suresh, Vijay. Daniel, R.H. Aswathy, Dr. V. Parthasarathy*. It gave the idea of IoT subject and addition details about IoT. The proper smart environment and various applications
- *Smart Garbage Management System by Vikrant Bhor, Pankaj Morajkar, Maheshwar Gurav,Dishant Pandya*. It provided us with additional details and designs needed for flow and management of garbage while collection.
- *Hayati, S., et. al., "The Rocky 7 Rover: A Mars Science craft Prototype", Proceedings of the 1997 IEEE International Conference on Robotics and Automation, pp. 2458-64, 1997.* It provided the design and implementation of rocker bogie.
- *D. S. Chinchkar, et. al.,"Design of Rocker Bogie Mechanism", International Advanced Research Journal in Science, Engineering and Technology,2017*. This paper provides us the information about mobility design, select of acceleration angle for robot and how rocker bogie is able to move on any terrain.

2.2. Referred Papers for Object Detection Classifiers

- *Alexander C. Berg,et. al., "SSD: Single Shot MultiBox Detector", Cornell University :* This paper helps us to understand and implement Single Shot MultiBox Detector.This tells us about SSD framework and how its faster than now state of art YOLO (You only look once).As in paper our team has implemented Convolutional Predictors for Prediction. Our team present a method for detecting objects in images using a single deep neural network. Our approach, named SSD, discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location. At prediction time, the network generates scores for the presence of each object category in each default box and produces adjustments to the box to better match the object shape. Additionally, the network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes.

- *Joseph Redmon,et. al., "You Only Look Once: Unified, Real-Time Object Detection", Cornell University*

  In this paper we understand that underlying architecture is extremely fast. The base YOLO model processes images in real-time at 45 frames per second. A smaller version of the network, Fast YOLO, processes an astounding 155 frames per second while still achieving double the mAP of other real-time detectors. Compared to state-of-the-art detection systems, YOLO makes more localization errors but is far less likely to predict false detections where nothing exists. Finally, YOLO learns very general representations of objects. It outperforms all other detection methods, including DPM and R-CNN, by a wide margin when generalizing from natural images to artwork on both the Picasso Dataset and the People-Art Dataset.

- *Jian Sun,et. al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", Cornell University :*

  In this paper we understand State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet and Fast R-CNN have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck. In this work, we introduce a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. An RPN is a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. We further merge RPN and Fast R-CNN into a single network by sharing their convolutional features.

CHAPTER 3

UML DIAGRAMS

3.1. Interaction Overview Diagram:

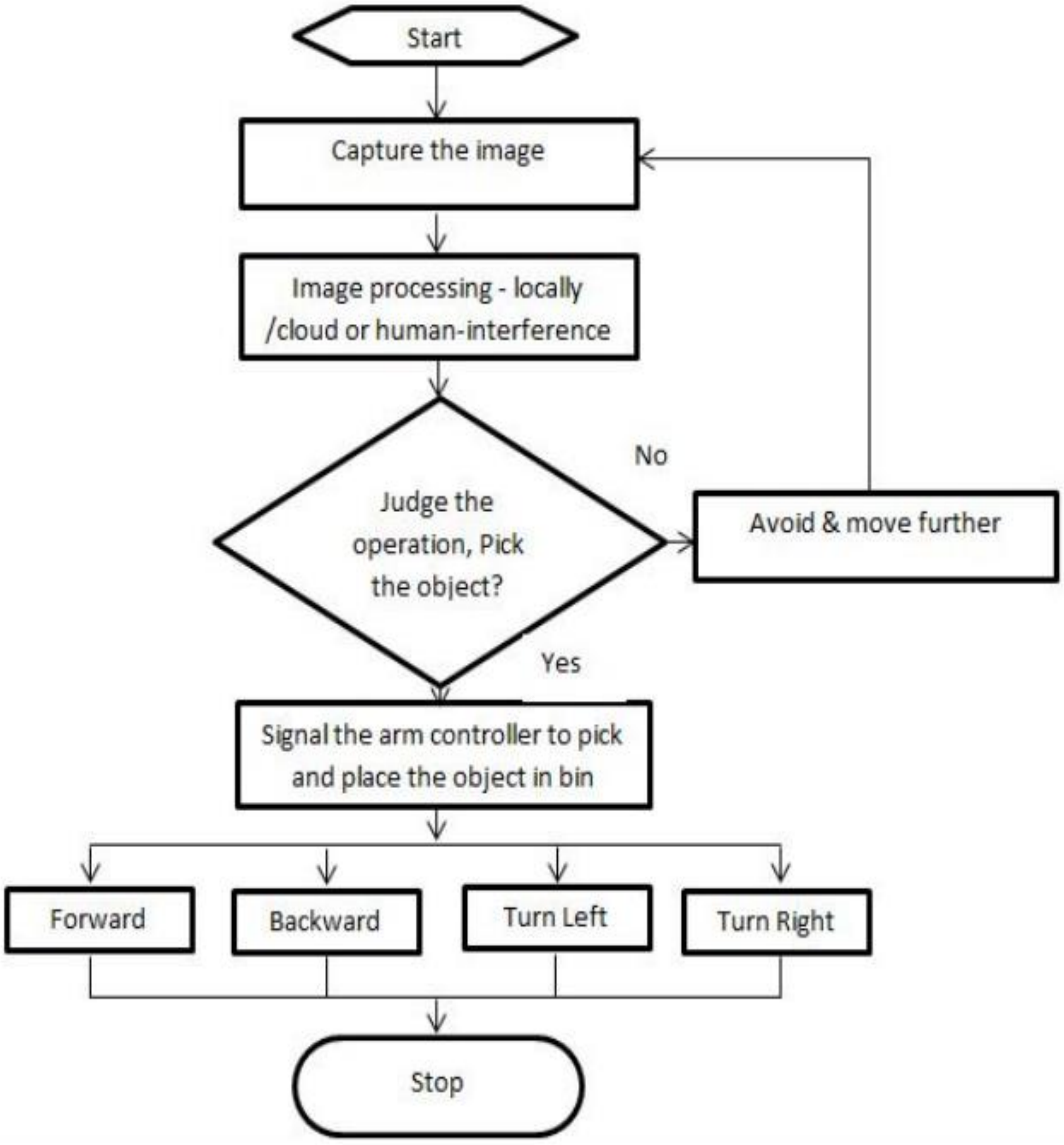

Start
Capture the image
Image processing - locally /cloud or human-interference
Judge the operation, Pick the object?
No
Avoid & move further
Yes
Signal the arm controller to pick and place the object in bin
Forward
Backward
Turn Left
Turn Right
Stop

Figure 4. Interaction Overview Diagram

3.2. Sequence Diagram:

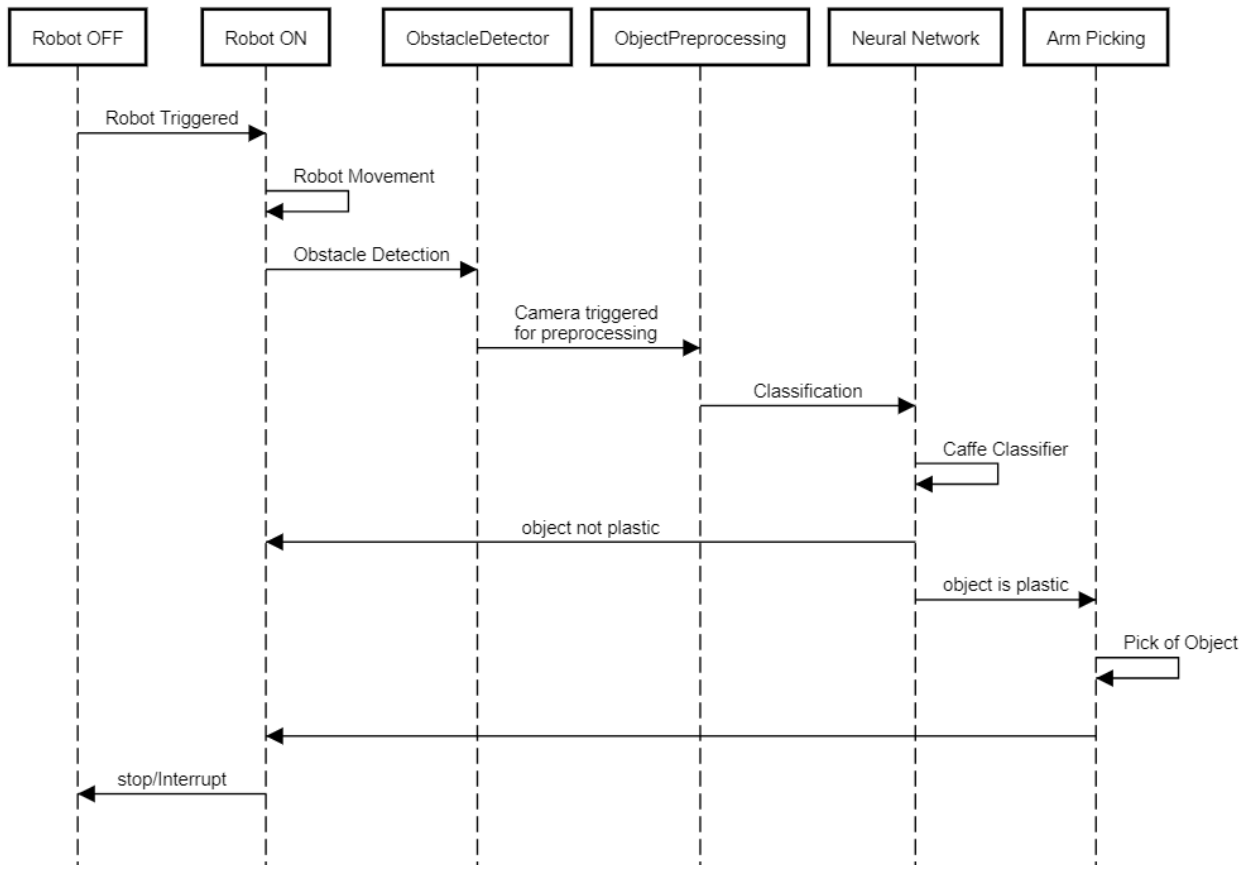


Figure 5. Sequence Diagram

CHAPTER 4

REQUIREMENTS

Requirement R1:Caffe for Object detection and Classification

R1.1 : OpenCV

OpenCV is a library of programming functions mainly aimed at real-time computer vision.

R1.2 : Numpy

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices.

R1.3 : Matplotlib

Matplotlib is a plotting library for the Python programming language.

R1.4 : Imutils

It includes a series of OpenCV + convenience functions that perform basics tasks such as translation, rotation, resizing, and skeletonization

Requirement R2: Autonomous all terrain Robot

R2.1 : Design of rocker bogie

The dimensions (lxbxh) of the Rocker bogie, size of each part, angle between the legs of the bogie.

R2.2 : 3D printing of motor holders

The design and printing of the custom holders for the DC motor.

R2.3 : Building of rocker bogie

Assembling all the parts of the bogie and making sure all fits properly.

R2.4 : Installation of Ultrasonic Sensor.

Installation of Ultrasonic Sensor and configuring them to our needs so as to make our robot autonomous.

Requirement R3: Robotic arm controlled by Raspberry Pi

R3.1 : Gripper

The design and 3D printing of the gripper.

R3.2 : Installation of required library

The servo controller requires installation of the Adafruit library in order to control the servo.

R3.3 : Controlling single servo motor

The controlling of single servo motor using the servo controller to check and understand the working of servo motor.

R3.4 : Controlling the arm

Controlling the 5 servo motors which is nothing but controlling the robotic arm in order to pick the object.

CHAPTER 5

PROJECT ANALYSIS

Our team has applied the standard product development approach to help us decide on the best solution for this project, it begins with a list of needs and this will be the basis for the project scope. The items shall define the goals of the project and in no way hint to a solution.

5.1. List of Needs

- No human interaction
- On-board computation
- Detect garbage
- Move garbage
- Avoid obstacles

5.2. Function Specification

5.2.1. External

The robot must be able to run automatically without interaction from external controller, and persist its stable behaviours to external noise and disturbance from the surrounding environment.

- No human interaction
- Robustness to external noise and disturbance.

5.2.2 Internal

The robot must be able to move itself in the arena on wheels. It must detect and avoid obstacles, find the bottles and transport as many as possible back to the designated area in 10 minutes.

- Being able to move
- Power autonomy for at least 10 minutes
- Localization
- Find the recycling area
- Object detection

- Differentiate between litter and obstacles
- Avoid obstacles
- Move the litter

5.3. Critical Technical Points:

The critical technical points were (in order): the locomotion, obstacle avoidance, bottle detection and bottle manipulation. We proceeded in this order to solve all these problems so that we could concentrate on one problem at a time.

5.4. Solutions Identification

5.4.1. Movement

The first consideration is the robot's locomotion. Several strategies for its movement are listed in table 1, including the specific principles, advantages, disadvantages and risks corresponding to the strategies.

Table 1. Movement Strategy

Strategy	Principle	Advantage	Disadvantage
Rocker bogie	6 wheeled robot	Powerful and can navigate on any terrain	Less spacious
Rover 5	Differential mobile based on tracks	Can navigate on any terrain	Must be ordered
Hexapod	6 legged robot	Navigate through complex terrain	Hard to program, slow, lots of parts
Biped	2 legged robot	Can access all terrains	Extremely difficult to program and control. not Statically stable
2-wheel different robot	2 wheeled robot	Easy to control only 2 motors	Needs a 3rd passive wheel

Strategy	Principle	Advantage	Disadvantage
Swedish wheels	Wheels on wheels.	Allows movement in any direction	Not as precise as Conventional wheels, odometery is very difficult. Cannot climb slopes

In line with the design concept of "as simple as possible", and through the analysis of the advantages, disadvantages for each listed strategy, the team has decided to go with Rocker bogie due to its ability to move on any terrain and power it delivers.

5.4.2. Object Detection

It is crucial for the robot to do object detection, including the bottles, the obstacles (bricks) and surrounding walls, so that it could do the sequence of behaviours containing avoiding the obstacles, avoiding the walls, and finding the bottles. Several strategies for object detection are listed in table 2, including the advantages, disadvantages and risks corresponding to the relative strategies.

Table 2. Object Detection Strategy

Strategy	Advantage	Disadvantage	Risk
Ultrasound	Linear response with distance, not affected by target materials, surface and colour. Can detect small objects over long operating distance. Resistance to external disturbances such as vibration infrared radiation, ambient noise, and EMI radiation.	Must view a surface (especially a hard, flat surface) squarely (perpendicularly) to receive ample sound echo. Require time for the transduce to stop ringing after each transmission burst before they are ready to receive returned echoes. Have a minimum sensing distance. Change in environment, target of density, smooth of	False positive outputs due to large operating angles, detecting an object other than desired target.

Strategy	Advantage	Disadvantage	Risk
Laser rangefinders	Better accuracy and more quickly. Easy alignment by employing visible red laser beam. Detect of very small targets due to small measuring spot size	Suffer from laser noise, stray light, and speckle effects interferences	Detect an object other than desired target
Camera	Cheaper, more informative and more compact	Limited of its view fields	Might not detect the whole target space
Tactile sensors	Guaranteed obstacle detection. Allow physical interaction with objects	Must be close enough to the obstacle, cannot avoid without physical interaction	Hit the obstacle while detecting
Infrared	Detect infrared light from far distance over a large area. In real time and detect movement.	Incapable of distinguishing between objects.	Strong infrared sources might be detected as obstacles.

At first the team wanted to use only one camera to do everything in order to keep a simple system. The camera could do different kinds of image processing and in theory detect and differentiate all the objects. When testing we saw that the camera could not differentiate between the floor and walls using the color information and was actually quite slow when processing the video stream, this meant we had to use other sensors to complement the camera. In the end our team chose to use the camera only for the bottle detection as it could do it reliably and use ultrasonic sensors for obstacle detection.

5.4.3. Bottle Grasping

After the achievement of finding bottles, the actuator should have the capacity of grasping and storing the bottles, so that the robot could complete the recycling target. There are a lot of practical ways for the robot to the grasp bottles, and several strategies are listed in table 3, including the principles, advantages, disadvantages and risks corresponding to the relative strategies.

Table 3. Bottle Grasping Strategy

Strategy	Principle	Advantage	Disadvantage	Risk
Robotic arm	Arm with a few DDLs mounted onto the robot	Allows picking up bottle in every position and in every terrain	Difficult mechanical realization and time consuming program	Complex implementation High possibility that mechanism won't work as intended for different situation
Clamp	Grabbing a bottle in front with a clamp	Depending on DDLs wanted, can be very easy to release	Need good precision in positioning to grab a bottle	Not being able to positioned the robot to pick up the bottle
Suction	Suction mechanism to hold bottles	Easy pickup and release	Require compressor, energy consuming, can be hard to position on bottle	No good seal between bottle and suction mechanism, therefore not being able to apply enough suction
Pushing	Bottle being rolled with the robots chassis	No extra mechanical parts	Can be hard to perform complex movement while keeping bottle pushed. Hard when bottle positioned close to edge or corner	Losing the bottle underway, hence loosing time with trying to recuperate it if it is even possible

Strategy	Principle	Advantage	Disadvantage	Risk
Storage bay for single bottle	Robot ingests bottle inside a storage bay	Easy mechanical implementation, carrying bottles around relative easy	Must return to base with every single bottle, must be well aligned with the bottle	Low risk
Deployable cage	Deploying a cage to surround the objects, and bring it back to the base	Very easy to implement, bottle position doesn't have to be exact	Can't bring bottle over rough terrains, only one bottle at a time.	Low risk

In accordance with the concept of "as simple as possible", and through the analysis of the advantages, disadvantages, and risks for each listed strategy, Robotic Arm becomes the final decision, which calls for much more mechanical structure, and actuator motion to grasp and store bottles, with less cost and risk but more availability and reliability.

CHAPTER 6

SOFTWARE DESIGN

Object Detection using Deep Learning

6.1. Why Deep Learning for Object Detection?

Deep learning has proven useful for many purposes and not just one single task. This is the point of the learning end-to-end, i.e. learning the whole problem from input to output. The same toolkit can work for different domains whether vision, speech, text, or control and robotics.

We'll focus on vision, and next we'll look at core visual recognition tasks and the standard benchmarks for each problem. Deep learning approaches have delivered dramatic improvements across these and many other tasks.

6.2. Visual Recognition Tasks:

6.2.1. Classification: Classification is the fundamental visual task of recognizing what is in an image or what type of image it is.

Classification is challenging because of the many differences in appearance seen in the visual world, like lighting, pose, style, and so on. Clutter or noise can obscure the information to be extracted from the image. Fine-grained categorization is a further difficult.

6.2.2. Detection: Detection is the task of recognizing not only what but where: both the identity and location of each object need to be predicted. While classification considered only presence or absence, detection demands the recognition of every instance in an input frame. Localization is difficult especially for interacting or articulated object. Even multiple instance of similar object and small object in a frame adds up to the challenge in detection further.

6.2.3. Semantic Segmentation: semantic segmentation is a visual recognition task that asks the identity of every pixel. For things this means what kind of object the pixel part is of. In this task there is a tension between recognizing what globally and where locally. Computational cost can be an obstacle now that there is a decision to be made for every pixel.

6.3. Object Detection in our Project using Caffe:

6.3.1. What is Caffe?

Caffe is a Deep Learning Framework made with expression, speed, modularity in mind.

6.3.2. Why Caffe?

Expressive architecture encourages application and innovation. Models and optimization are defined by configuration without hard-coding. Switch between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity clusters or mobile devices.

Extensible code fosters active development. In Caffe's first year, it has been forked by over 1,000 developers and had many significant changes contributed back. Thanks to these contributors the framework tracks the state-of-the-art in both code and models.

Speed makes Caffe perfect for research experiments and industry deployment. Caffe can process over **60M images** per day with a single NVIDIA K40 GPU. That's 1 ms/image for inference and 4 ms/image for learning and more recent library versions and hardware are faster still. We believe that Caffe is among the fastest convnet implementations available.

6.3.3. Caffe in our Project:

In this project, we're presently using

1. MobileNet (A computationally efficient CNN architecture).
2. Single Shot Multibox Detector Algorithm.

When combined together these methods can be used for super-fast, real-time object detection on resource constrained devices like raspberry-pi

.

6.3.3.1. MobileNet:

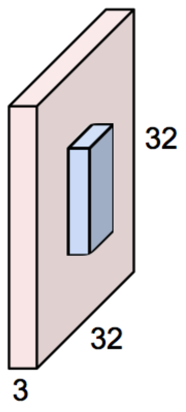
MobileNet: MobileNets are Efficient Convolutional Neural Networks for Mobile Vision Applications. Convolution means fewer parameters for more efficient learning. CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output.

Unlike neural networks, where the input is a vector, here the input is a multi-channeled image (3 channeled in this case).

We take the 5x5x3 filter and slide it over the complete image and along the way take the dot product between the filter and chunks of the input image. For every dot product taken, the result is a scalar.

Diagrammatic depiction of working of Convolution Layer in MobileNets:

1. A filter is a spatially local and cross-channel template, CNN filters are learned**.**



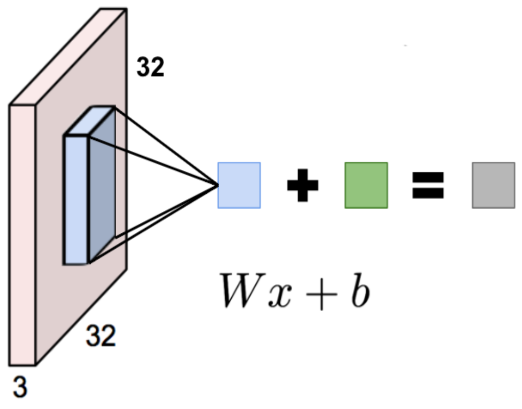
Input is 3x32x32 data.

- A color image (3 RGB channels) and square frame (32x32)

Filter is 3x5x5 weights.

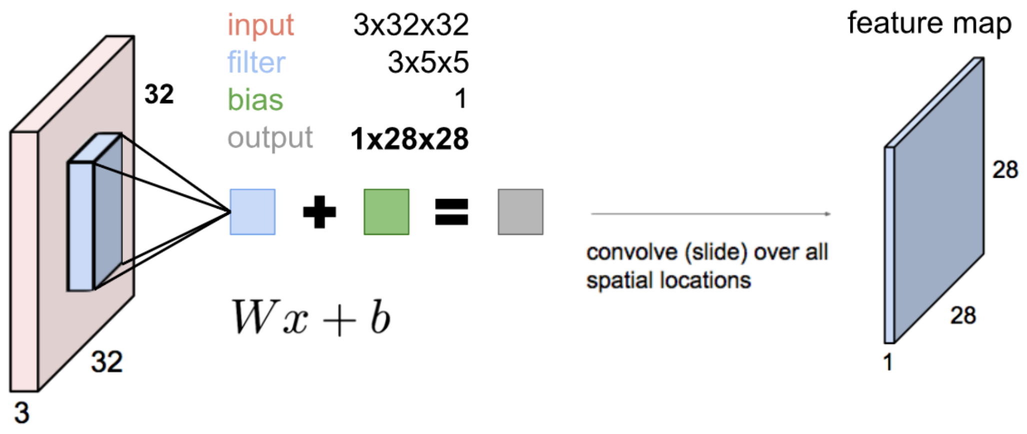
- spatially local: kernel size is 5x5
- cross-channel: connected across all input channels

2. One filter evaluation is a dot product between the input window and weights + bias.

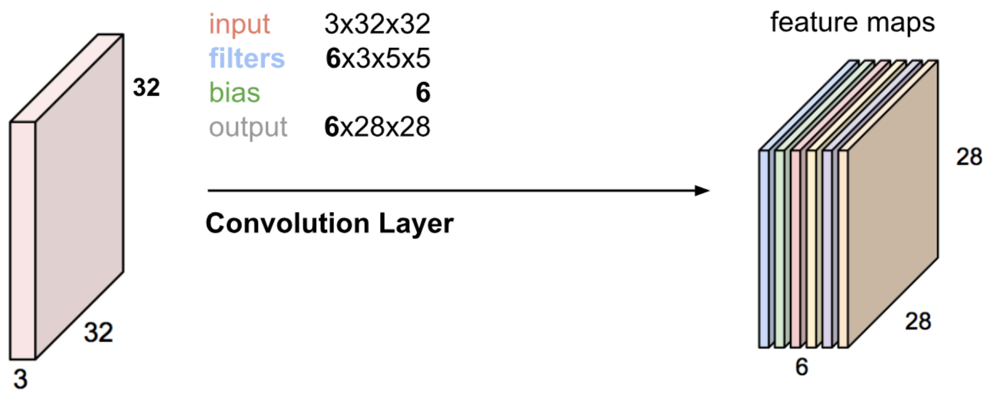


input: 3x32x32
filter: 3x5x5
bias: 1
output: 1

3. Convolving the filter with the input gives a feature map.



4. Convolution layers have multiple filters for more modeling capacity.



ReLU Layer:

ReLU is the abbreviation of Rectified Linear Units. This layer applies the non-saturating activation function f(x)=\max(0,x). It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer.

Other functions are also used to increase nonlinearity, for example the saturating hyperbolic tangent f(x)=\tanh(x), f(x)=\tanh(x)| and the sigmoid function f(x)=(1+e^{-x})^{-1}} f(x)=(1+e^{-x})^{-1}. ReLU is often preferred to other functions, because it trains the neural network several times faster without a significant penalty to generalization accuracy.

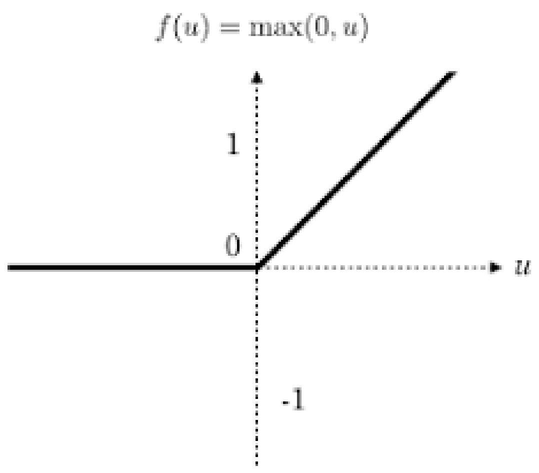


Figure 6. ReLU Function Curve (Source: Wikipedia)

Pooling Layer: Pooling is a form of non-linear down-sampling. The goal of the pooling layer is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. There are several functions to implement pooling among which max pooling is the most common one. Pooling is often applied with filters of size 2x2 applied with a stride of 2 at every depth slice. A pooling layer of size 2x2 with stride of 2 shrinks the input image to a 1/4 of its original size.

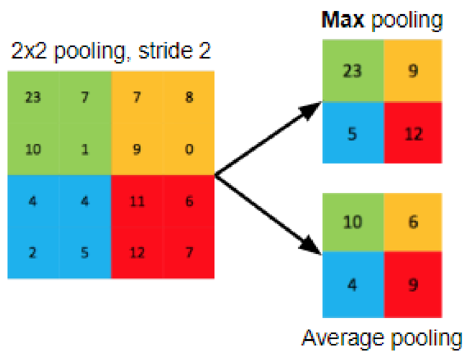


Figure 7. Pooling with 2x2 filter with stride = 2

Convolution Neural Network Architecture:

The simplest architecture of a convolutional neural networks starts with an input layer (images) followed by a sequence of convolutional layers and pooling layers, and ends with fully-connected layers. The convolutional layers are usually followed by one layer of ReLU activation functions.

The convolutional, pooling and ReLU layers act as learnable features extractors, while the fully connected layers acts as a machine learning classifier. Furthermore, the early layers of the network encode generic patterns of the images, while later layers encode the details patterns of the images.

Note that only the convolutional layers and fully-connected layers have weights. These weights are learned in the training phase.

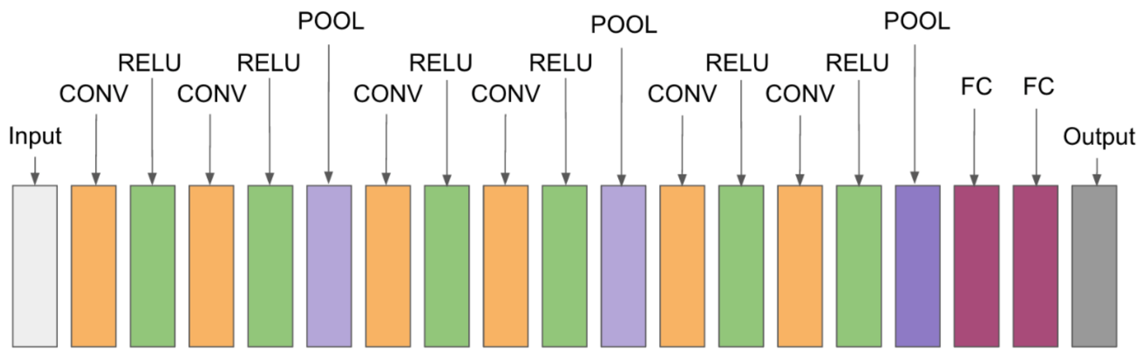


Figure 8. Example of CNN Architecture. (Source: Wikipedia)

6.3.3.2. Single Shot MultiBox Detector Algorithm:

The paper about SSD: Single Shot MultiBox Detector (by C. Szegedy et al.) was released at the end of November 2016 and reached new records in terms of performance and precision for object detection tasks, scoring over 74% mAP (mean Average Precision) at 59 frames per second on standard datasets such as PascalVOC and COCO. To better understand SSD, let's start by explaining where the name of this architecture comes from:

- Single Shot: this means that the tasks of object localization and classification are done in a single forward pass of the network
- MultiBox: this is the name of a technique for bounding box regression developed by Szegedy et al. (we will briefly cover it shortly)
- Detector: The network is an object detector that also classifies those detected objects.

SSD Architecture:

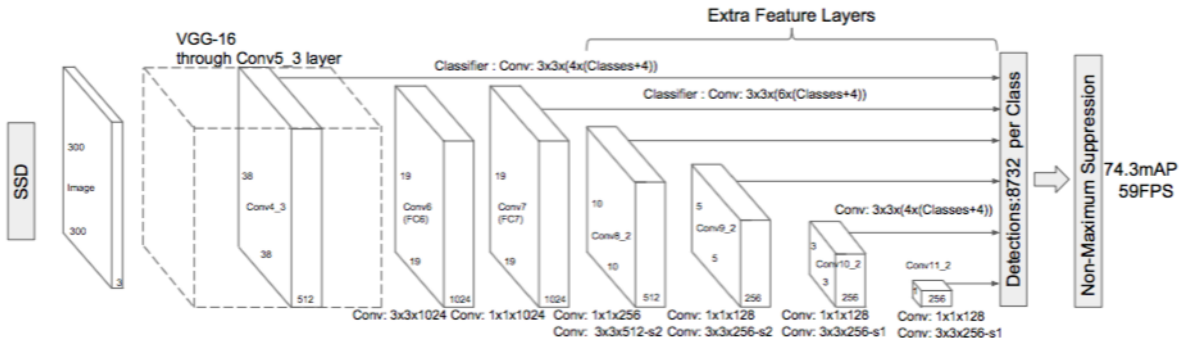


Figure 9. SSD Architecture (Source: MultiBox by Szegedy et al.)

As you can see from the diagram above, SSD's architecture builds on the venerable VGG-16 architecture, but discards the fully connected layers. The reason VGG-16 was used as the base network is because of its strong performance in high quality image classification tasks and its popularity for problems where transfer learning helps in improving results. Instead of the original VGG fully connected layers, a set of auxiliary convolutional layers (from conv6 onwards) were added, thus enabling to extract features at multiple scales and progressively decrease the size of the input to each subsequent layer.

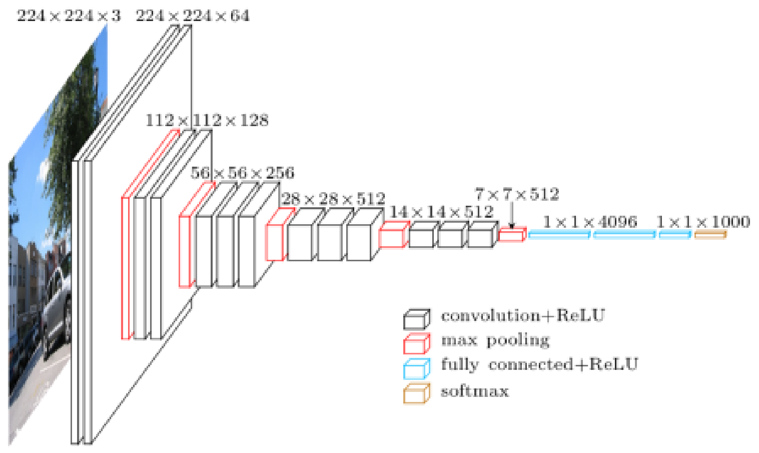


Figure 10. VGG Architecture (Input is 224x224x3)
(Source: Towards Data Science)

MultiBox: The bounding box regression technique of SSD is inspired by Szegedy's work on MultiBox, a method for fast class-agnostic bounding box coordinate proposals. Interestingly, in the work done on MultiBox an Inception-style convolutional network is

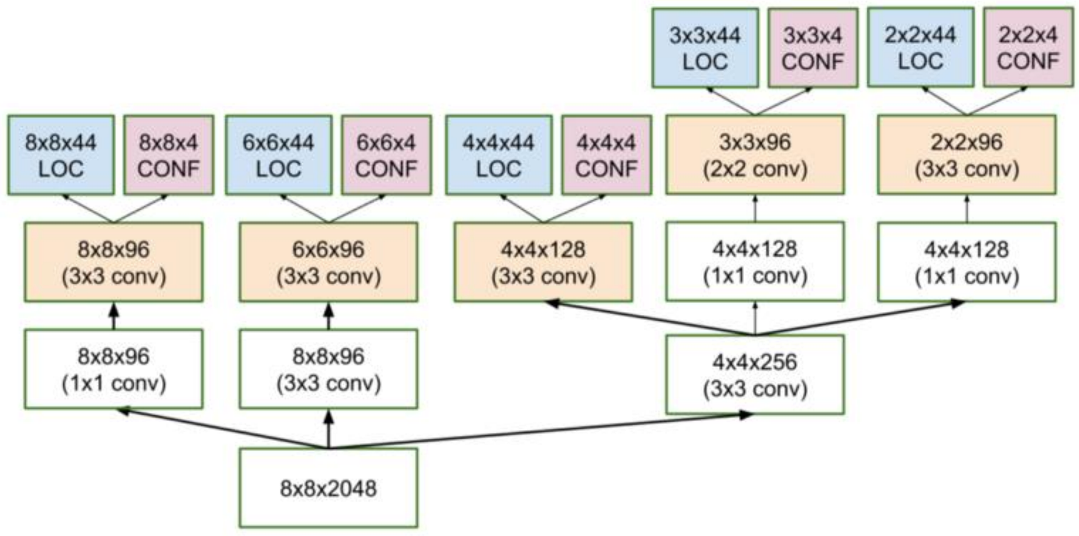


Figure 11. Architecture of multi-scale convolutional prediction of the location and confidences of Multibox
(Source: Towards Data Science)

used. The 1x1 convolutions that you see below help in dimensionality reduction since the number of dimensions will go down (but "width" and "height" will remain the same).

MultiBox's loss function also combined two critical components that made their way into SSD:

- Confidence Loss: this measures how confident the network is of the objectness of the computed bounding box. Categorical cross-entropy is used to compute this loss.
- Location Loss: this measures how far away the network's predicted bounding boxes are from the ground truth ones from the training set.

The expression for the loss, which measures how far off our prediction "landed", is thus:

$$multibox\_loss = confidence\_loss + alpha * location\_loss$$

The alpha term helps us in balancing the contribution of the location loss. As usual in deep learning, the goal is to find the parameter values that most optimally reduce the loss function, thereby bringing our predictions closer to the ground truth.

MultiBox Priors and IoU:

In MultiBox, the researchers created what we call priors (or anchors in Faster-R-CNN terminology), which are pre-computed, fixed size bounding boxes that closely match the distribution of the original ground truth boxes. In fact those priors are selected in such a way that their Intersection over Union ratio (aka IoU, and sometimes referred to as Jaccard index) is greater than 0.5. As you can infer from the image below, an IoU of 0.5 is still not good enough but it does however provide a strong starting point for the bounding box regression algorithm—it is a much better strategy than starting the predictions with random coordinates. Therefore MultiBox starts with the priors as predictions and attempt to regress closer to the ground truth bounding boxes.

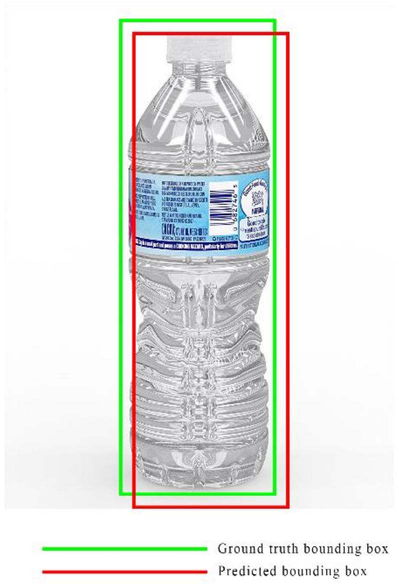


Figure 12. Bounding boxes

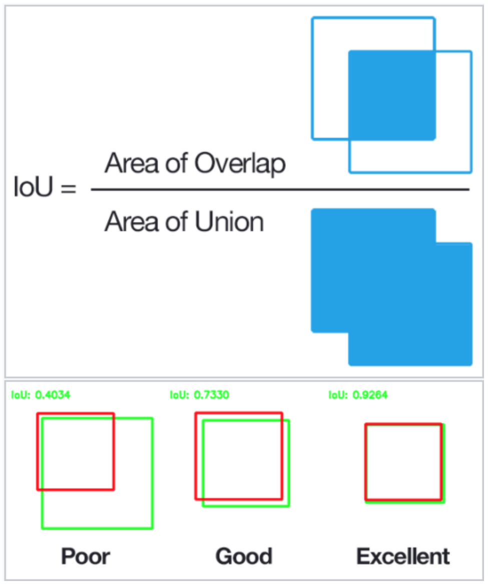


Figure 13. Intersection over Union Factor (Source: Wikipedia)

The resulting architecture (check MultiBox architecture diagram above again for reference) contains 11 priors per feature map cell (8x8, 6x6, 4x4, 3x3, 2x2) and only one on the 1x1 feature map, resulting in a total of 1420 priors per image, thus enabling robust coverage of input images at multiple scales, to detect objects of various sizes. At the end, MultiBox only retains the top K predictions that have minimized both location (LOC) and confidence (CONF) losses.

Training Dataset:

Total Data = 70000 images

- Plastic bottle images = 25000
    - Test Set = 75% of Total Plastic bottle images = 18750
    - Train Set = 15 % of Total Plastic bottle images = 3750
    - Validation Set = 10 % of Total Plastic bottle images = 2500
- Plastic bag images = 25000
    - Test Set = 75% of Total Plastic bag images = 18750
    - Train Set = 15 % of Total Plastic bag images = 3750
    - Validation Set = 10 % of Total Plastic bag images = 2500

- Plastic cup images = 20000
    - Test Set = 75% of Total Plastic cup images = 15000
    - Train Set = 15 % of Total Plastic cup images = 3000
    - Validation Set = 10 % of Total Plastic cup images = 2000

The SSD paper makes the following additional observations:

- More default boxes results in more accurate detection, although there is an impact on speed
- Having MultiBox on multiple layers results in better detection as well, due to the detector running on features at multiple resolutions
- 80% of the time is spent on the base VGG-16 network: this means that with a faster and equally accurate network SSD's performance could be even better
- SSD confuses objects with similar categories (e.g. animals). This is probably because locations are shared for multiple classes
- SSD-500 (the highest resolution variant using 512x512 input images) achieves best mAP on Pascal VOC2007 at 76.8%, but at the expense of speed, where its frame rate drops to 22 fps. SSD-300 is thus a much better trade-off with 74.3 mAP at 59 fps.
- SSD produces worse performance on smaller objects, as they may not appear across all feature maps. Increasing the input image resolution alleviates this problem but does not completely address it.

CHAPTER 7

HARDWARE DESIGN

7.1. Robot Design:

The team has made robot chassis based on Rocker bogie design as it is stable and is more preferable for our project over the other designs. It is a six wheel drive with all terrain rubber wheels. Each wheel also has cleats, providing grip for working in all terrains.

The rocker-bogie system is the suspension arrangement used in the Mars rovers (mechanical robot) introduced for the Mars Pathfinder and also used on the Mars Exploration Rover (MER) and Mars Science Laboratory (MSL) missions. It is currently NASA's favored design.

The term "rocker" comes from the rocking aspect of the larger links on each side of the suspension system. These rockers are connected to each other and the vehicle chassis through a differential. Relative to the chassis, when one rocker goes up, the other goes down. The chassis maintains the average pitch angle of both rockers. One end of a rocker is fitted with a drive wheel and the other end is pivoted to a bogie. The term "bogie" refers to the links that have a drive wheel at each end. Bogies were commonly used as load wheels in the tracks of army tanks as idlers distributing the load over the terrain. Bogies were also quite commonly used on the trailers of semi-trailer trucks. Both applications now prefer trailing arm suspensions.

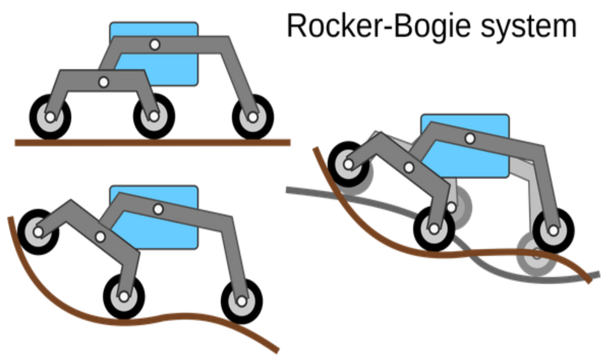


Figure 14. Rocker Bogie Design (Source: Wikipedia)

Our chassis is completely based on rocker bogie concept. We have made use of PVC pipes, as pipes are light and strong. For joints we have used PVC L and T joints. To attach the motors to the chassis we have made use of 3D printed mount specific to size of motor so that it is easy to connect it to PVC pipes on one end.

In order to go over a vertical obstacle face, the front wheels are forced against the obstacle by the center and rear wheels. The rotation of the front wheel then lifts the front of the vehicle up and over the obstacle. The middle wheel is then pressed against the obstacle by the rear wheels and pulled against the obstacle by the front until it is lifted up and over. Finally, the rear wheel is pulled over the obstacle by the front two wheels. During each wheel's traversal of the obstacle, forward progress of the vehicle is slowed or completely halted.

Bogie Specification

Size	50x45x30 (lxbxh) in cms
Weight	10kg
Chassis material	UPVC pipes
Wheels	130 mm all terrain wheels
Motor holders	Custom 3D printed
Controller Box	Custom sheet metal box

Table 4. Rocker Bogie Specification

7.2. Hardware

7.2.1. Motor

The team has used 200 rpm Johnson DC motor to move our rocker bogie. These are high torque motors. The specification of these motors is as follows

Rotations per minute:	200 rpm
Output torque range:	5kg-cm to 7kg-cm.
No-load current:	800 ma(max)
Load current:	upto 9.5 a(max)
Motor Weight:	90 gms
Brush Material:	Precious metal

Table 5. Johnson DC Motor Specification

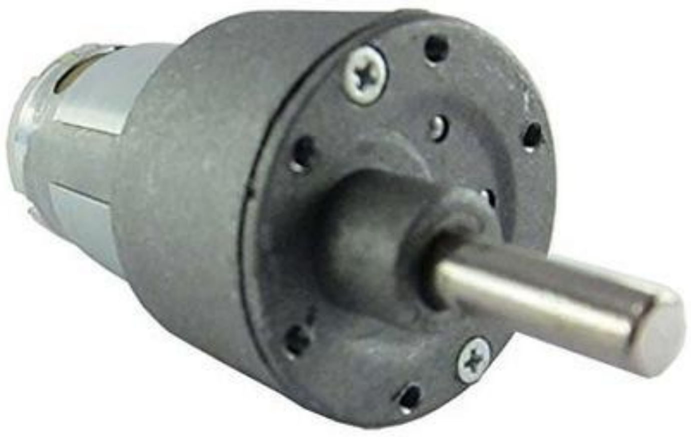


Figure 15. 200rpm Johnson DC Motor (Source: Robokits)

7.2.2. Servo Motors

In our bogie, the team has used NRS-995 metal gear servo motors of Nex Robotics for the movement of arm. The team has used these motor because these motors have high torque which helps our arm to pick objects. The specification of these servo motors is as follows.

Dimension:	40.7mm x 20.5mm x39.5mm
Torque:	15.5kg/cm at 4.8V, 17kg/cm at 6V
Bearings:	Dual bearing with metal gear
Operating speed:	0.15sec/60 degree
Operating voltage:	4.8V to 6V
Temperature range:	0-55C
Motor weight:	60gms

Table 6. Servo Motor Specification

Figure 16. Servo Motor (Source: NEX Robotics)

7.3. Electronics

7.3.1. Raspberry Pi 3

The Raspberry Pi is a very cheap and relatively powerful small 700MHz computer. The team chose it because we already had some experience working with it and it has a significant community behind it that could help us out in case of problems. The specification of Raspberry Pi 3 is as follows,

	Raspberry Pi 3 Model B
Introduction Date	2/29/2016
SoC	BCM2837
CPU	Quad Cortex A53 @ 1.2GHz
Instruction set	ARMv8-A
GPU	400MHz VideoCore IV
RAM	1GB SDRAM
Storage	micro-SD
Ethernet	10/100
Wireless	802.11n / Bluetooth 4.0
Video Output	HDMI / Composite
Audio Output	HDMI / Headphone
GPIO	40
Price	$35

Table 7. Raspberry Pi 3 Model B Specification (Source: raspberrypi.org)

7.3.2. Ultrasonic Sensor

The team has used ultrasonic sensor HC-SR04 for detecting obstacles in front of the robot. The specification of ultrasonic sensor is as follows

Supply voltage	5 v
Global Current Consumption	15 mA
Ultrasonic Frequency	40k Hz
Maximal Range	400 cm
Minimal Range	3 cm
Trigger Pulse Width	10 µs
Outline Dimension	43x20x15 mm

Table 8. Ultrasonic Sensor Specification

Working of Ultrasonic Sensor: The Ultrasonic Sensor sends out a high-frequency sound pulse and then times how long it takes for the echo of the sound to reflect back. The sensor has 2 openings on its front. One opening transmits ultrasonic waves, (like a tiny speaker), the other receives them, (like a tiny microphone). The speed of sound is approximately 341 meters (1100 feet) per second in air. The ultrasonic sensor uses this information along with the time difference between sending and receiving the sound pulse to determine the distance to an object. It uses the following mathematical equation:

Distance = Time x Speed of Sound / 2

Time = the time between when an ultrasonic wave is transmitted and when it is received. Dividing this number by 2 because the sound wave has to travel to the object and back.

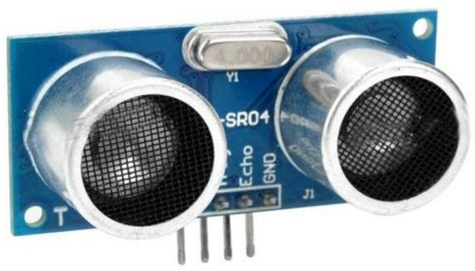


Figure 17. Ultrasonic Sensor

7.3.3. Motor Driver

Our Rocker bogie consists of 6 DC motors which drag too much current when operated simultaneously. The use of normal 9685IC motor controller won't be of useful for us. Therefore we are using RKI 1341 motor control for controlling the DC motors of our robot which is capable of handling up to 20A current. Some of the features of this controller are

- Simple connectivity to IO pins of any MCU.
- Compatible with motors rated 6V - 18V
- Can easily deliver 20A of current during normal operation
- Braking feature included without affecting the performance of an MCU

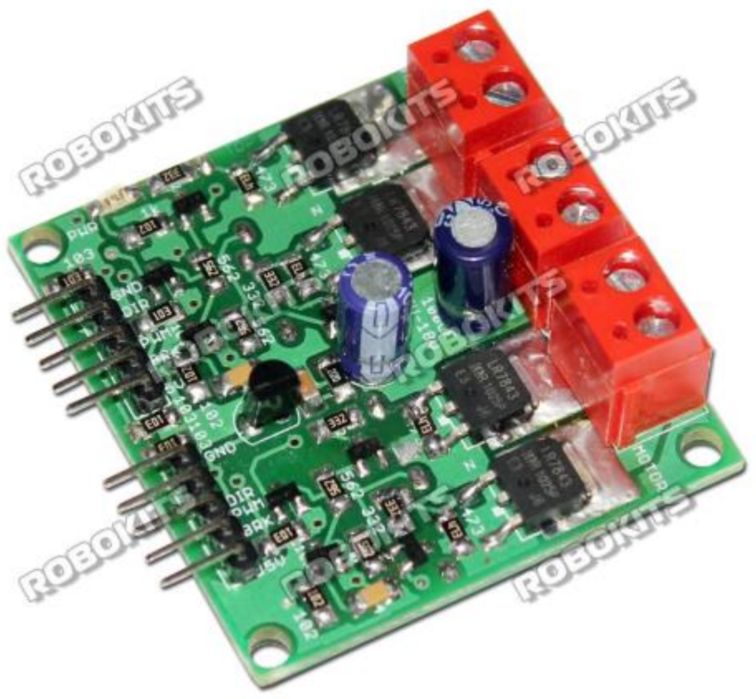


Figure 18. Motor Controller (Source: Robokits)

7.3.4. Servo Controller

The robotic arm consists of high torque metal gear servo motor, controlling these servo with general purpose input output pins of raspberry pi is hard. So, we are using PCA 9685 16 channel servo controller which has ability to control 16 servo motors. It's an i2c-controlled PWM driver with a built in clock. That means that, unlike the TLC5940 family, you do not need to continuously send it signal tying up your microcontroller, its completely free running. It is 5V compliant, which means you can control it from a 3.3V microcontroller and still safely drive up to 6V outputs, however we are using external power supply for the robotic arm.

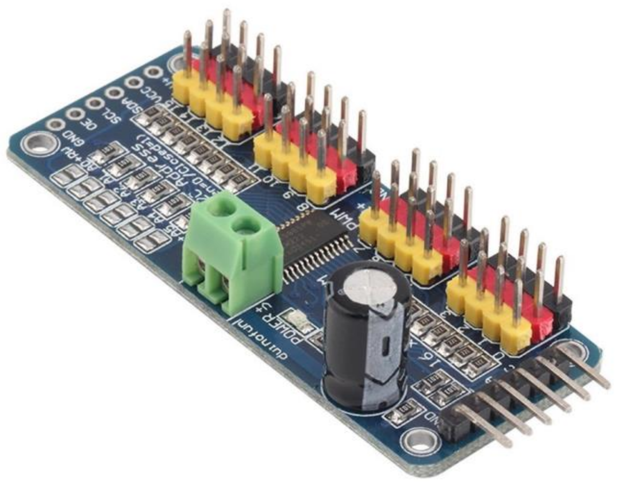


Figure 19. PCA 9685 Servo Controller (Source: Robokits)

CHAPTER 8

IMPLEMENTATION

8.1. Object Classifier Implementation

Input : Captured frame from camera.

Output : Correctly classified objects.

Real time video capture:

First we start the VideoStream , then we wait for the camera to warm up , and finally we start the frames per second counter (Line 37). The VideoStream and FPS classes are part of my imutils package.

```
# initialize the video stream, allow the cammera sensor to warmup,
# and initialize the FPS counter
print("[INFO] starting video stream...")
time.sleep(5.0)
vs = VideoStream(0).start()
time.sleep(2.0)
fps = FPS().start()

# loop over the frames from the video stream
# grab the frame from the threaded video stream and resize it
# to have a maximum width of 400 pixels
frame = vs.read()
frame = imutils.resize(frame, width=1000,height=1000)
```

Data Preprocessing (Image Pre-processing):

The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task.

```
def resize_image(image_path):
      im=Image.open(image_path)
      resizedImage = im.resize((int(299), int(299)), Image.ANTIALIAS)
      resizedImage.save(image_path)
```

Converting image to blob:-

A Binary Large OBject (BLOB) is a collection of binary data for better processing of images.

```
# grab the frame dimensions and convert it to a blob
(h, w) = frame.shape[:2]
blob = cv2.dnn.blobFromImage(frame, 0.007843, (300, 300), 127.5)
```

Loading the model:

Next, we load our serialized model from disk, initialize the list of class labels MobileNet SSD was trained to detect, then generate a set of bounding box colors for each class.

```
print("[INFO] loading model...")
net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])
CLASSES = ["bottle", "plastic_cup", "plastic_bag"]
COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))
```

Image Classification:

Now we go frame by frame and check the bounding boxes for finding whether the detections are present.

```
for i in np.arange(0, detections.shape[2]):
      # extract the confidence (i.e., probability) associated with
      # the prediction
      confidence = detections[0, 0, i, 2]
      # filter out weak detections by ensuring the `confidence` is
      # greater than the minimum confidence
      if confidence > args["confidence"]:
            idx = int(detections[0, 0, i, 1])
            box = detections[0, 0, i, 3:7] * np.array([w, h

      if confidence > args["confidence"]:
            idx = int(detections[0, 0, i, 1])
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")
            print(startX," ",startY)
            print(endX," ",endY)
            if (endX-startX)<(endY-startY):
                  print("vertical")
            elif (endX-startX)>(endY-startY):
                  print("Horizontal")
            part_width=600/10
            iterator=0
            while(iterator<=600):
                  if(startX<iterator):
                        x=iterator
                        break
                  iterator=iterator+part_width
            print(x/part_width)

            label = "{}: {:.2f}".format(CLASSES[idx],
                  confidence * 100)
            cv2.rectangle(frame, (startX, startY), (endX, endY),
                  COLORS[idx], 2)
            y = startY - 15 if startY - 15 > 15 else startY + 15
            cv2.putText(frame, label, (startX, y),
                  cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)
            mystring= label.split(":")


                  print(mystring[0]+" "+mystring[1])
                  cv2.imshow("Frame", frame)
                  cv2.waitKey(0)
```

8.2. Robot automation implementation

Pin configuration: In this stage we configure the left wheel,right wheel pwm pins

```
GPIO.setup(12, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(19, GPIO.OUT)
GPIO.setup(21, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)
pwm=GPIO.PWM(15,255)
pwm.start(pwmValue)
```

Distance Calculation: In this stage we calculate the distance of obstacle using ultrasonic sensor

```
def distance(GPIO_TRIGGER,GPIO_ECHO):
    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.0001)
    GPIO.output(GPIO_TRIGGER, False)
    StartTime = time.time()
    StopTime = time.time()
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()
    TimeElapsed = StopTime - StartTime
    distance = (TimeElapsed * 34300) / 2
    return distance
```

Robot movement (Forward & Backward): Robot movement forward and backward

```
def moveLeftWheelForward(duration,pwmVal):
    GPIO.output(13,0)
    pwm.ChangeDutyCycle(pwmVal)
    GPIO.output(12,turn)
    time.sleep(duration)

def moveRightWheelForward(duration,pwmVal):
    GPIO.output(21,0)
    pwm1.ChangeDutyCycle(pwmVal)
    GPIO.output(19,not(turn))
    time.sleep(duration)

def moveForward(duration):
    moveLeftWheelForward(0,pwmValue)
    moveRightWheelForward(0,pwmValue)
    time.sleep(duration)
```

```
def moveBackward(duration):
      moveLeftWheelBackward(0,pwmValue)
      moveRightWheelBackward(0,pwmValue)
      time.sleep(duration)
```

Robot movement (Left & Right): Robot movement left and right

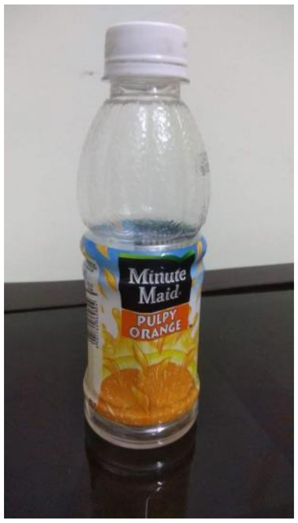
```
def moveRightWheelBackward(duration,pwmVal):
      GPIO.output(21,0)
      pwm1.ChangeDutyCycle(pwmVal)
      GPIO.output(19,turn)
      time.sleep(duration)

def stopRightWheel():
      GPIO.output(21,1)

def stopLeftWheel():
      GPIO.output(13,1)

def moveRight():
      stopRightWheel()
      moveLeftWheelBackward(0.5,25)
      stopRightWheel()
      stopLeftWheel()

def moveLeft():
      stopLeftWheel()
      moveRightWheelBackward(0.5,25)
      stopRightWheel()
      stopLeftWheel()
```

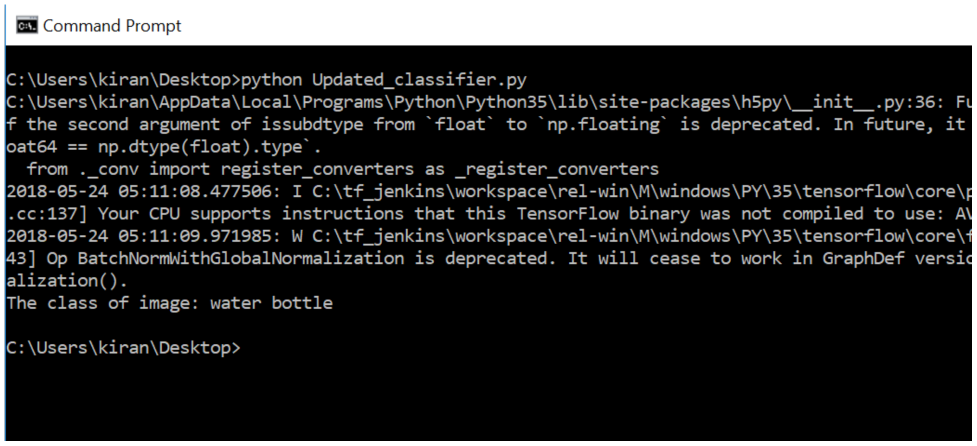
CHAPTER 9

TESTING

Our team tested with plenty of plastic bottles, paper cups and plastic bags, and every time we got accurate results of the particular category of the object which is fed to camera frame.

Below are the some examples of each category's input and output.

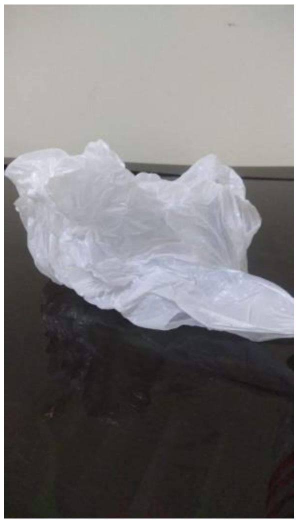


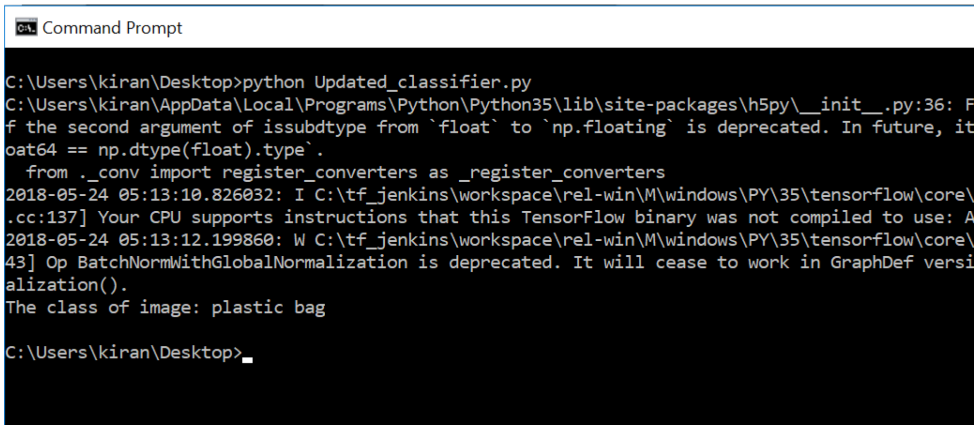


Command Prompt

```
C:\Users\kiran\Desktop>python Updated_classifier.py
C:\Users\kiran\AppData\Local\Programs\Python\Python35\lib\site-packages\h5py\__init__.py:36: Fu
f the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, it
oat64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
2018-05-24 05:11:08.477506: I C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core\p
.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AV
2018-05-24 05:11:09.971985: W C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core\f
43] Op BatchNormWithGlobalNormalization is deprecated. It will cease to work in GraphDef versio
alization().
The class of image: water bottle

C:\Users\kiran\Desktop>
```

Figure 20. Plastic Bottle Input (Right), Suitable Output of terminal (Left)

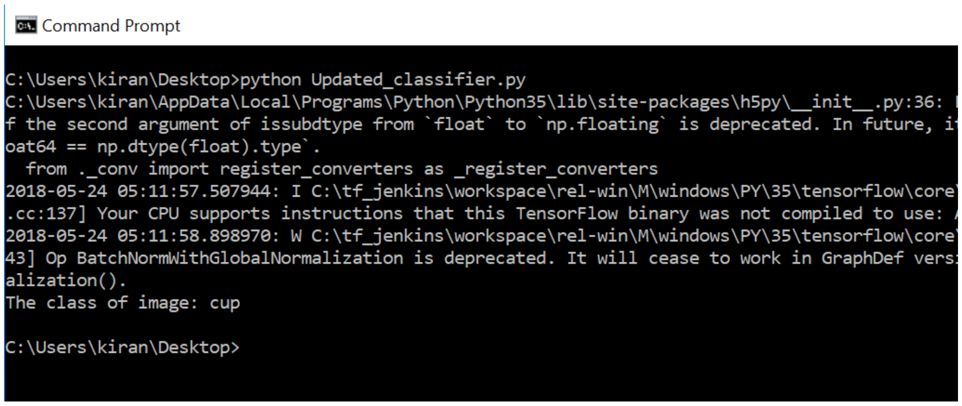




Command Prompt

```
C:\Users\kiran\Desktop>python Updated_classifier.py
C:\Users\kiran\AppData\Local\Programs\Python\Python35\lib\site-packages\h5py\__init__.py:36: F
f the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, it
oat64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
2018-05-24 05:13:10.826032: I C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core\
.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: A
2018-05-24 05:13:12.199860: W C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core\
43] Op BatchNormWithGlobalNormalization is deprecated. It will cease to work in GraphDef versi
alization().
The class of image: plastic bag

C:\Users\kiran\Desktop>
```

Figure 21. Plastic Bag Input (Right), Suitable Output of terminal (Left)



Command Prompt

```
C:\Users\kiran\Desktop>python Updated_classifier.py
C:\Users\kiran\AppData\Local\Programs\Python\Python35\lib\site-packages\h5py\__init__.py:36:
f the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, i
oat64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
2018-05-24 05:11:57.507944: I C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core
.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use:
2018-05-24 05:11:58.898970: W C:\tf_jenkins\workspace\rel-win\M\windows\PY\35\tensorflow\core
43] Op BatchNormWithGlobalNormalization is deprecated. It will cease to work in GraphDef vers
alization().
The class of image: cup

C:\Users\kiran\Desktop>
```

Figure 22. Paper cup Input (Right), Suitable Output of termina (Left)l

CHAPTER 10

CONCLUSION AND FUTURE SCOPE

Conclusion:

- The most considerable purpose of Internet of Things (IoT) based Smart Garbage Collection Systems is the collection and segregation of garbage automatically by robots.
- The segregated waste can be further recycled and reused. For example the plastic waste which cannot be either segregated manually or decomposed into the environment can be segregated using advanced technology and given for recycling industries for reuse.
- Our team believes that these robots will help in realizing our Government's Swachh Bharat Abhiyan and Make in India schemes along with implementation of Smart Cities.
- This project was a very challenging one, we learned to apply the things we were taught in some courses and we had to design, develop, program and test a robot from scratch.

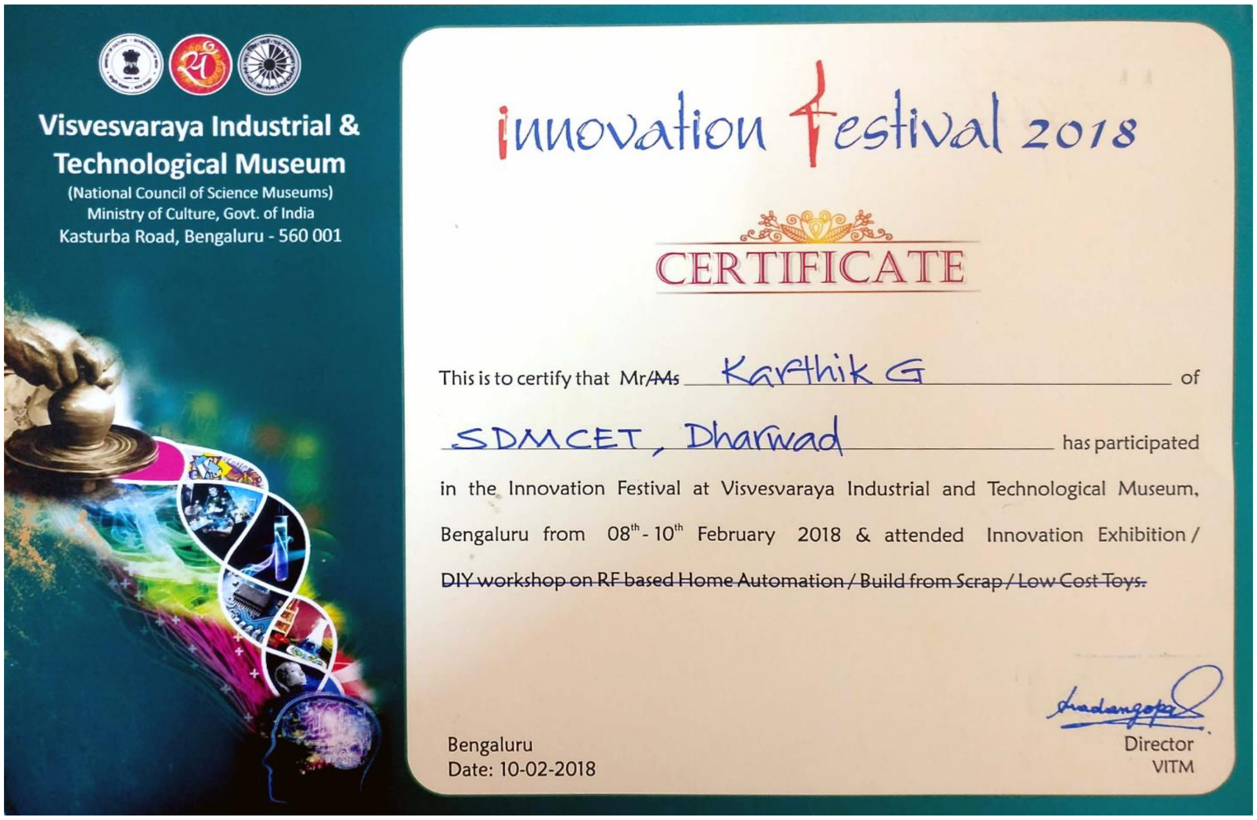
Future Scope:

- In future we can install multiple arms on the robot so that we can collect multiple object at a time and thus we can collect lot of waste in less time.
- We can upgrade the processor (raspberry pi) so that video processing is possible. Also for garbage collection on large scale we need better processor for speeding up the classification process.
- We can upgrade our servo motors with industrial grade high torque servo motors so that the robotic arm can lift more heavy objects.
- We can apply this same technology for garbage collection using drones and also for cleaning of water bodies.

ACHIEVEMENTS AND APPRECIATIONS

Our project is incubated under Department of ITBT, Govt. of Karnataka. We participated in many of the competitions and exhibitions. Following are few of the achievements gained by the project in course of previous academic year.

1. Selected as "Top 10 ITBT funded projects" in VITM, Bengaluru. Certificates issued to all the team members.



Visvesvaraya Industrial & Technological Museum
(National Council of Science Museums)
Ministry of Culture, Govt. of India
Kasturba Road, Bengaluru - 560 001

innovation festival 2018

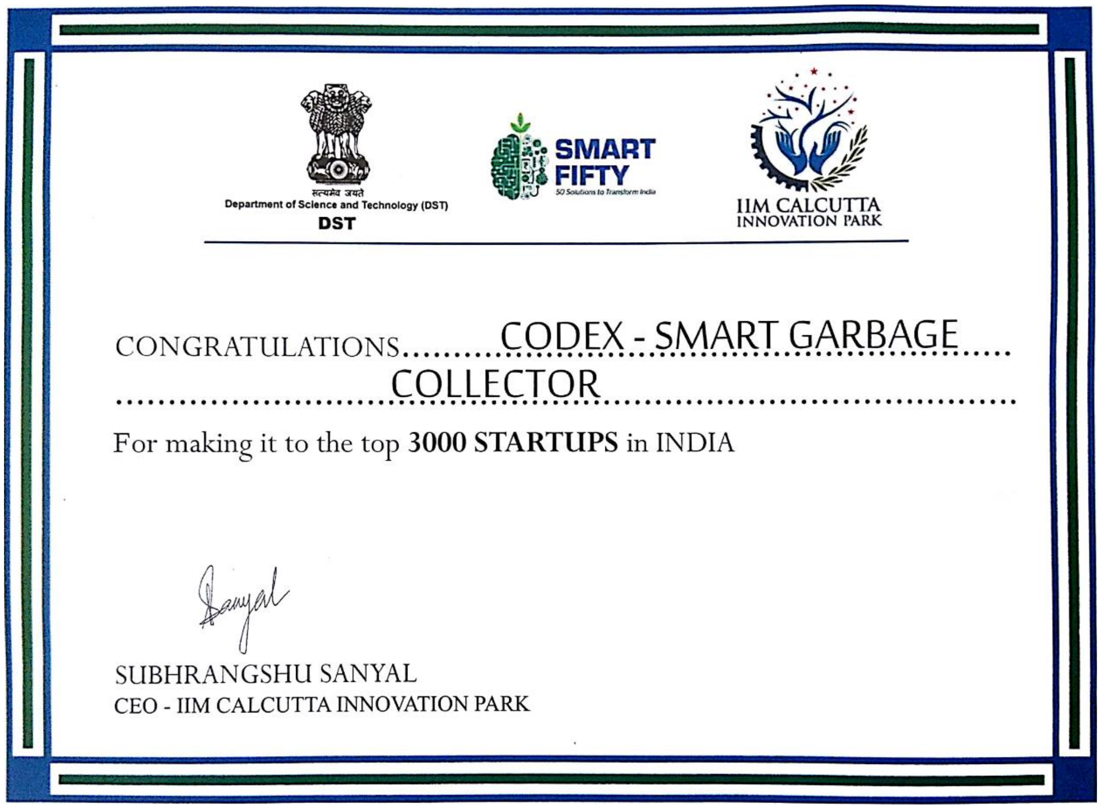
CERTIFICATE

This is to certify that Mr/~~Ms~~ Karthik G of SDMCET, Dharwad has participated in the Innovation Festival at Visvesvaraya Industrial and Technological Museum, Bengaluru from 08th - 10th February 2018 & attended Innovation Exhibition / ~~DIY workshop on RF based Home Automation / Build from Scrap / Low Cost Toys.~~

Bengaluru
Date: 10-02-2018

[illegible]
Director
VITM

2. Selected as "Top 3000 startups in India", An initiative by IIM Calcutta



सत्यमेव जयते
Department of Science and Technology (DST)
DST

SMART FIFTY
50 Solutions to Transform India

IIM CALCUTTA
INNOVATION PARK

CONGRATULATIONS........CODEX - SMART GARBAGE.....
........................COLLECTOR........................

For making it to the top **3000 STARTUPS** in INDIA

Sanyal

SUBHRANGSHU SANYAL
CEO - IIM CALCUTTA INNOVATION PARK

REFERENCES

[1] Andrew G. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", Cornell University, Submitted on 17 Apr 2017.

[2] Wei Liu et al., “SSD: Single Shot MultiBox Detector”, Cornell University, Submitted on 8 Dec 2015 (v1), last revised 29 Dec 2016 (this version, v5)

[3] Joseph Redmon et al., “You Only Look Once: Unified, Real-Time Object Detection”, Cornell University, Submitted on 8 June 2015 (v1), last revised 9 May 2016 (this version, v5).

[4] Shaoqing Ren et al., “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, Cornell University, Submitted on 4 Jun 2015 (v1), last revised 6 Jan 2016 (this version, v3)

[5] Ray Jarvis, Nghia Ho, "Robotic Cybernavigation in Natural Known Environments", Cyberworlds (CW) 2010 International Conference on, pp. 338-345, 2010

[6] A State of the Art review on Internet of Things by P. Suresh, Vijay. Daniel, R.H. Aswathy, Dr. V. Parthasarathy.

[7] Smart Garbage Management System by Vikrant Bhor, Pankaj Morajkar, Maheshwar Gurav,Dishant Pandya.

[8] Hayati, S., et. al., “The Rocky 7 Rover: A Mars Science craft Prototype”, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, pp. 2458-64, 1997

[9] D. S. Chinchkar, et. al.,“Design of Rocker Bogie Mechanism”, International Advanced Research Journal in Science, Engineering and Technology,2017