



Masters Thesis
ROBINLAB, University of Jaume I

Deep Learning for Object Recognition in picking tasks
Arijit Mallick

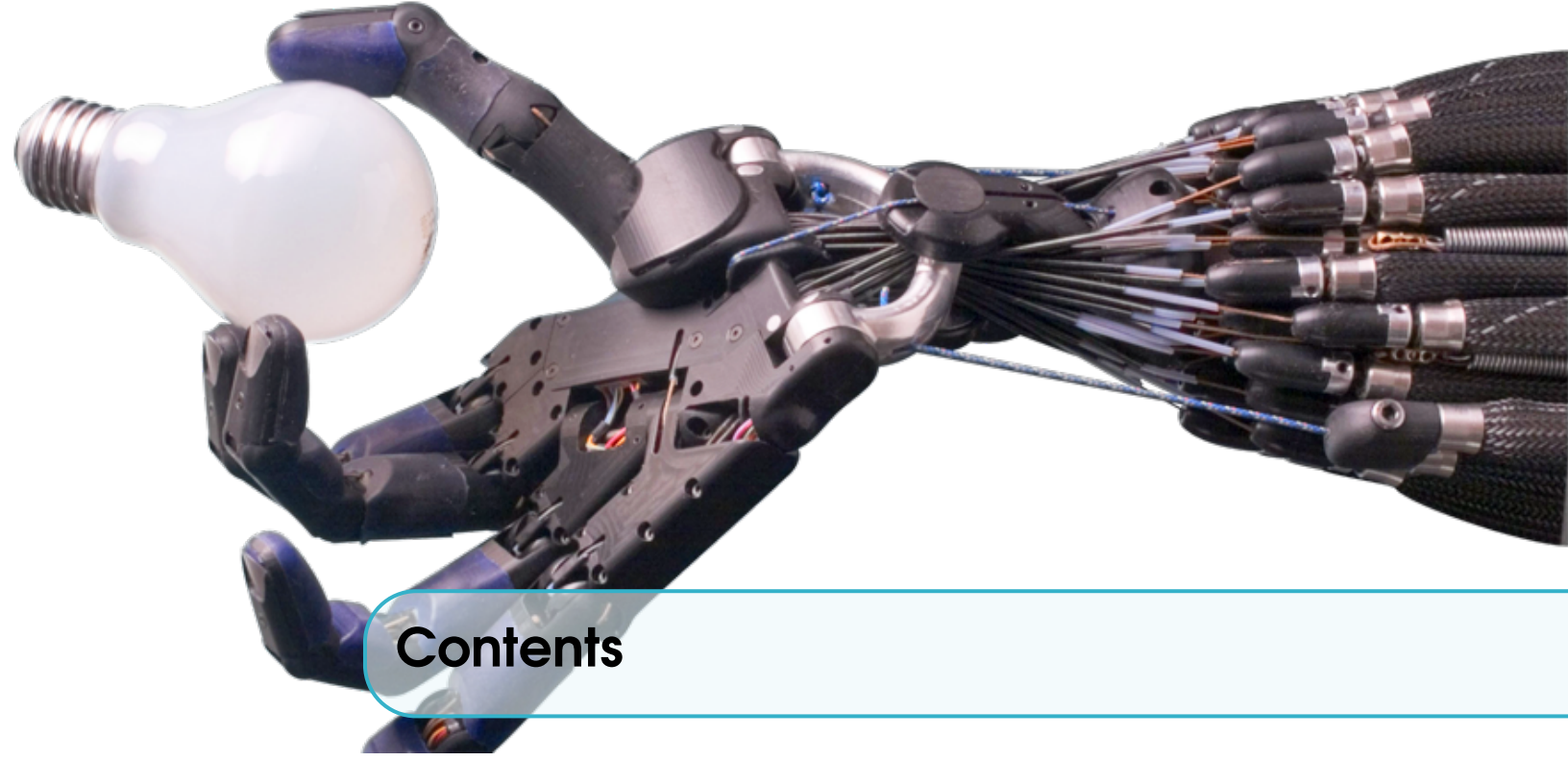
Advisor:
Prof. Enric Cervera
Co-Advisor:
Prof. Olivier Kermorgant

July 10, 2017

ERASMUS MASTERS IN ADVANCED ROBOTICS, UNIVERSITY OF JAUME I

This research was done under the supervision of Dr. Enric Cervera Mateu and Dr. Angel del Pobil of Robotic Intelligence Lab, Spain within a total of 22 weeks, from February to July of 2017.

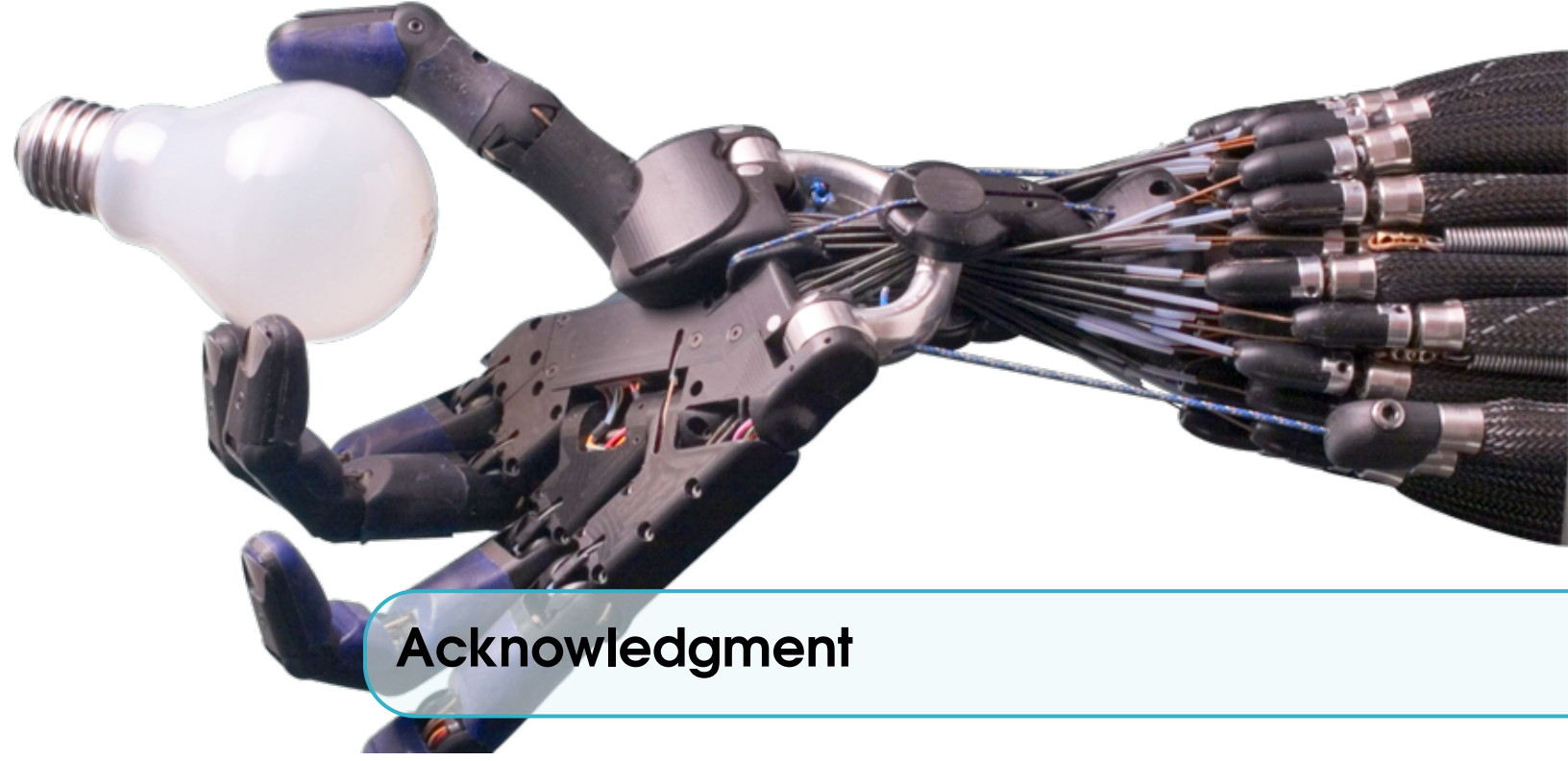
Selected date of thesis defense: 13th July 2017



Contents

1	Introduction	9
1.1	Motivation	9
1.2	Objective	10
1.3	Research Background	11
1.4	Useful definitions	11
2	System Description	13
2.1	Hardware	13
2.2	Software	15
2.3	Work Pipeline	15
3	Convolutional Neural Network and Vision	17
3.1	Brief introduction to Machine Learning	17
3.2	Deep Learning	18
3.2.1	Convolutional Neural Network (CNN)	18
3.3	Problem of Localization and initial approaches	20
3.4	Single shot detector : YOLO	20
3.5	Semantic segmentation and RESNET	22
4	Vision Setup for training	27
4.1	Calibration and setup	27

4.2	DATASET specification	28
4.3	Training time and practical constraints	28
4.4	Dataset input and labels	29
5	Object Recognition for picking	31
5.1	The challenge and the plan: revisited	31
5.2	YOLO: Observations	31
5.3	RESNET: Observations	32
5.4	Results and Evaluation	32
5.4.1	Evaluation metric	32
5.4.2	Evaluation of YOLO	33
5.4.3	Evaluation of RESNET	33
5.5	Discussions	36
6	Conclusions and Future Work	43
6.1	General Conclusions	43
6.2	Future Work	43
7	APPENDIX I	49



Acknowledgment

This work would not have been possible without the financial support of the Robotic Intelligence Lab (ROBINLAB) of University of Jaume I, Spain. I am especially indebted to Prof. Enric Cervera Mateu and Prof. Angel Pobil, not only they encouraged me to pursue the current line of research, but also kind enough to consider me as a deserving candidate for the team representing ROBINLAB, Spain for the prestigious AMAZON robotics challenge 2017.

I am grateful to all of those with whom I have had the pleasure to work during this and other related projects. Each of the members of my team, which includes my classmates and the PhD researchers at ROBINLAB has provided me extensive personal and professional guidance and taught me a great deal about both scientific research and life in general. I would particularly like to thank my classmate Monica and PhD researchers Angel Duran and Majd for their hard work and determination. Without them this work would have not been well implemented and I am particularly grateful for the photos and videos provided by them. I am also indebted to my classmate Iliia Vasylev, with the help of whom we prepared the synthetic dataset for the semantic segmentation.

I would also like thank the ERASMUS MUNDUS foundation for providing a full scholarship throughout my masters and took care of my financial and medical needs in time. Without them, this masters program and the current work would have been impossible. Overall, I would like to thank all the members of ROBINLAB for their sheer determination and providing an enthusiastic work environment, including my parents who played a pivotal role in encouraging and supporting me throughout some of the hard times I was facing during my work.

Yours gratefully
ARIJIT
Castellon, Spain

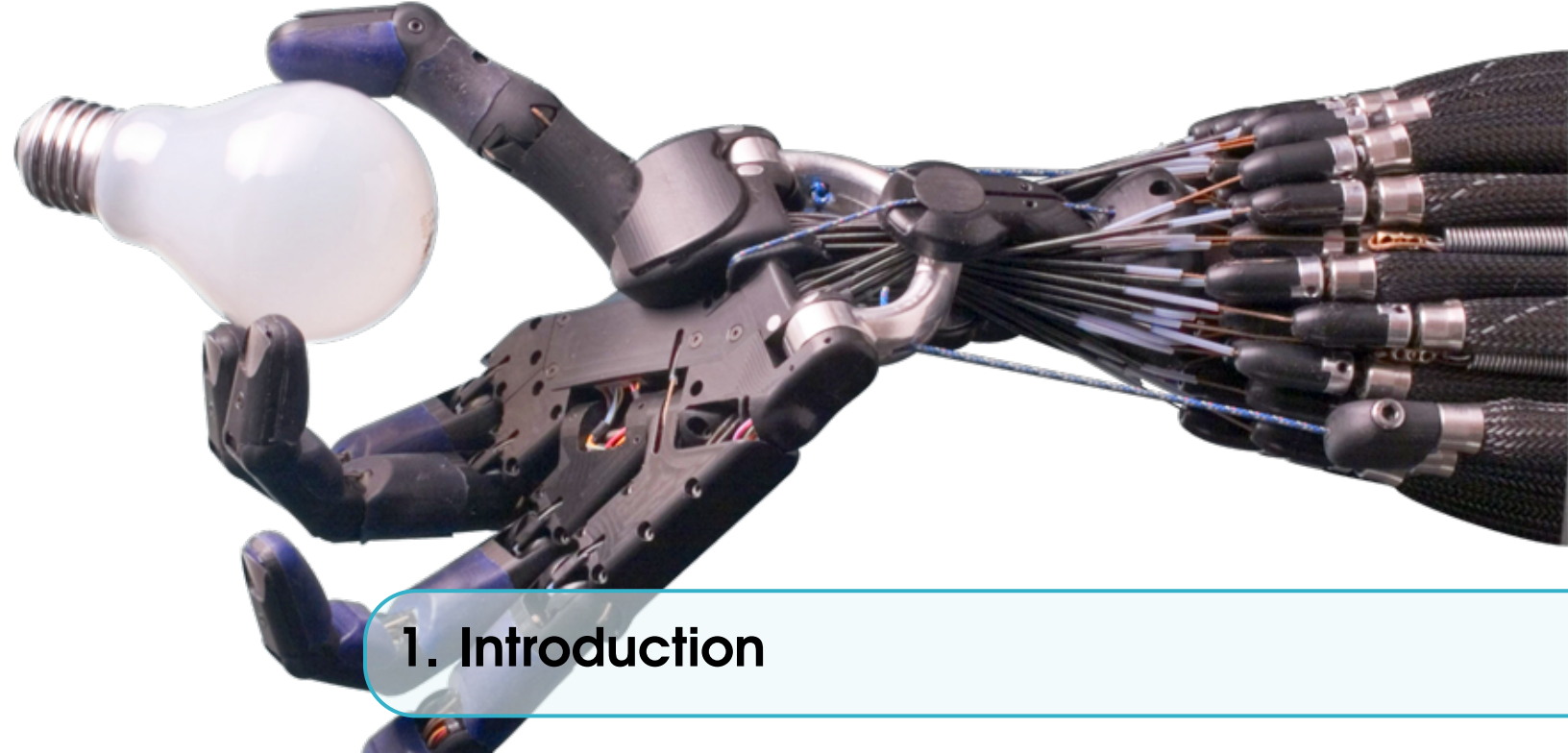


Abstract

In the light of current advancement in deep learning, robot vision is not an exception. Many popular machine learning algorithms has already been proposed and implemented to solve intricate computer vision problems. The same has not been in the case of robot vision. Due to real time constraints and dynamic nature of environment such as illumination and processing power, very few algorithms are able to solve the object recognition problem at large.

The primary objective of the thesis project is to converge into an accurate working algorithm for object recognition in a cluttered scene and subsequently helping the BAXTER robot to pick up the correct object among the clutter. Feature matching algorithms usually fail to identify most of the object having no texture, hence deep learning has been employed for better performance. The next step is to look for the object and localize it within the image frame. Although basic shallow Convolutional Neural Network easily identifies the presence of an object within a frame, it is very difficult to localize the object location within the frame. This work primarily focuses on finding a solution for an accurate localization. The first solution which comes to mind is to produce a bounding box surrounding the object. In literature, YOLO is found to be providing a very robust result on existing datasets. But this was not the case when it was tried on new objects belonging to the current thesis project work. Due to high inaccuracy and presence of a huge redundant area within the bounding box, an algorithm was needed which will segment the object accurately and make the picking task easier. This was done through semantic segmentation using deep CNNs.

Although time consuming, RESNET has been found to be very efficient as its post processed output helps to identify items in a significantly difficult task environment. This work has been done in light of upcoming AMAZON robotic challenge where the robot successfully classified and distinguished everyday items from a cluttered scenario. In addition to this, a performance analysis study has also been done comparing YOLO and RESNET justifying the usage of the later algorithm with the help of performance metrics such IOU and ViG.



1. Introduction

Robot vision is the ability of a robot to perceive its surroundings and interact according to the visual feedback in context of its environment. Along the years major goals for robot vision has been detection, recognition, servoing and so on. Robot vision extends the scope of computer vision in terms that it helps to fulfill the robot tasks in dynamic and real time conditions. Major goal of this project is to distinguish items in a cluttered scenario and produce a region with the recognizing item for better gripping or isolating it to its target position. It is to be noted that a significant amount image post processing has also been performed in order to produce the gripping region data in the item.

1.1 Motivation

With the rapid advancement in technology, robot vision is playing a key role in revolutionizing the industrial environment. There are many open challenges in the industrial sector where robot vision plays a major part and one many such instances is AMAZON Robotics Challenge. The ongoing thesis also serves as a proposition to the vision module of the robotics challenge which will be discussed in the further section. According to the overview section of the AMAZON Robotics Challenge :

Amazon is able to quickly package and ship millions of items to customers from a network of fulfillment centers all over the globe. This wouldn't be possible without leveraging cutting-edge advances in technology. Amazon's automated warehouses are successful at removing much of the walking and searching for items within a warehouse. However, commercially viable automated picking in unstructured environments still remains a difficult challenge. It is our goal to strengthen the ties between the industrial and academic robotic communities and promote shared and open solutions to some of the big problems in unstructured automation.

In this context to perform automated picking in a dynamic environment, a robust vision system is required in order to keep track of the products, detect it correctly with a satisfactory confidence

and subsequently followed by gripping or picking action by a robot. This has been the primary motivation for developing a vision module for the BAXTER Robot so that it can use the visual information as a cue to correctly identify and classify a list of items and arrange it to the required target position.

1.2 Objective

Prior to diving into the details of the current objectives of the problem, several other open robot vision challenges needs a special mention.

- IMAGENET Large Scale Visual Recognition Challenge: Includes different sections for detection, localization and video feature detection.
- KAGGLE CIFAR 100 challenge: Ongoing classification challenge for testing individual's machine learning algorithm.
- MS COCO (Common Objects in Context): Organised by Microsoft, involves financial prize. Mainly involves detection within given dataset.
- Multi-modal Brain Tumor Segmentation Challenge: Bio-medical image analysis, involves detection or pre-detection of brain tumor through image segmentation.
- ARC Amazon Robotics Challenge : Involves classification of everyday items within a cluttered scenario and picking and classifying different objects. Vision is considered as a major sub-module and is hosted annually, involving cash prize.

We will discuss our goals separately which will cover the objectives of the AMAZON picking challenge and it's correlation with this current work.

Objectives of the AMAZON robotics challenge involve picking of objects from a shelf where around 40 items are distributed more or less uniformly in different bins within the shelf. In this project, an automated movable shelf is prepared to assist BAXTER. In the first challenge (Picking), the BAXTER robot is supposed to recognize a set of items from a clutter of objects assisted by the vision system and will put the items in the target bins. In the second challenge (Stowing), the BAXTER robot is supposed to recognize a set of target items within a very large clutter of objects and put it back inside the movable shelf. Although overall objective seems different for the challenge, vision objective remains the same- recognizing and localizing a target item within the 2D pixel-space. The rest of the objectives are accomplished by the motion, control and gripper module. It needs to be mentioned that the challenge involves distinguish 16 prior known items (out of the previously mentioned 40 items) and 16 unknown items. It is to be mentioned that the primary scope of this project is to detect the 16 known items, the unknown items have been solved with normal feature matching technique due to faster implementation and lesser efficiency. In short the objective of the vision module is to distinguish known items within a dynamic cluttered scenario.

The major focus of this thesis work is to highlight the improvements made in the vision module. The vision module receives a set of target objects and the task of the vision module is to detect whether the objects are present in the current frame of the robot camera. If the object(s) are present, the vision module will localize the detected items and preferably provide a metric of confidence. This work primarily focuses on finding out a concrete deep learning based algorithm which will provide a pixel level accurate localized output which will be assisting the gripper to grasp an item and putting it on the target location. This work tries to explore different types of localization methods including bounding box output and semantic segmentation method. Apart from this, the objective also includes the proper integration of the deep learning module to work congruently with BAXTER robo-picker system.

1.3 Research Background

Ideal robot vision setup should be immune to dynamic changes from real time constraints such as illumination change, blurring and unexpected noise due to sudden and unexpected movement of the robotic system in action and its surrounding environment. In order to distinguish different objects, one needs to extract the features of the provided object followed by feature matching in the given environment. There have been many different feature extractors such as SURF (Bay et al. (2008)) and SIFT(Lowe (2004)) . Although qualitatively superior, they take a considerable amount of time for real time detection. There have been solution for them as well such as the ORB (Rublee et al. (2011)) extractor which provides sufficient keypoints for matching but is functionally inferior than SURF. For a large image database of different object, one needs to record all the keypoints and for each object and try to match it as per given problem.

Apart from this, another direct approach is to take the assistance of Machine Learning. In the past few years, several machine learning algorithms has been instrumental in recognizing and localizing objects in the pixel space. These include graphically drawing a rectangular grid indicating an object such as YOLO (Redmon et al. (2015)) or curving out the entire mask through semantic segmentation. Several such similar algorithms algorithms have been instrumental for the past few years. Special mentions include single shot detectors such as YOLO (Redmon and Farhadi (2016))(Santos (2016d)) and (Liu et al. (2015))(Santos (2016c)) as mentioned above. Apart from Deep residual learning (He et al. (2016),Santos (2016a)) Faster RCNN(Ren et al. (2015)) has also been proposed for similar application for image recognition and classification (Zhang et al. (2016)). Further bounding box detection scenarios has also been immensely implemented (LeCun et al. (2013),Zeiler and Fergus (2013),Xie and Tu (2015)). Image segmentation algorithms primarily include GoogleNet (Szegedy et al. (2014)) and (He et al. (2016)). In this literature, we are more interested in finding a solution which provides a segmented mask corresponding to different objects. These are primary works on Convolutional Neural Networks (CNN) such as the end-to-end trained Fully Convolutional network (Long et al. (2015)) for image segmentation. Adding to this, DeepLab (Chen et al. (2016)) is also a major contender for semantic image segmentation using deep convolution nets and fully connected CRFs. Adding to the list of end-to-end learning, DeconvNet (Noh et al. (2015)) also receives a special mention. Although precise in segmentation, it has a time constraint even when trained with fast GPU based machines. Apart from this, several recent works involves the application of convnets to dense prediction problems, which also included semantic segmentation by Silberman et al. (2012). The next section will focus on clarifying several popular terms in the field of deep learning based vision.

Application of semantic segmentation for robotics and autonomous system is not uncommon. We have seen some previous works on segnet where it is applied on vision for autonomous cars (Vijay et al. (2015)). Object recognition for robopicker has been employed before(Jonschkowski et al. (2016)) via multi-class segmentation as well. This work uses a much more efficient algorithm of RESNET in order to identify and localize the objects. Apart from this, YOLO has also been applied for the first time to enhance the vision module for a robo-picker.

1.4 Useful definitions

- *Computer and Robotic Vision:* According to Kragic and Vincze (2009) Computer vision targets the understanding of a scene usually from a single images or from a fixed position of camera. In this area, methods are usually focused on a particular problem or algorithms. On the other hand, one should look at robot vision in terms of system level perspective. Vision is

only a part of a very large system in action, and coordination between several sub-modules lead to the completion of a task. Here, visual processing is a part of a very complex entity.

- *Object recognition and object detection*: Simply putting, Object recognition is basically object detection of different classes of objects and subsequent localization and labeling within image pixel space. Object detection involves the same principle, but in this case it only detects a single class of object within a test image.
- *Object localization*: Roughly speaking, Object Localization is object recognition and subsequent indication of the particular area (bounding box, contour) and labeling in the image space.
- *Segmentation and semantic segmentation* : In image segmentation, regions of a particular image is segmented, without any label attached to it. Additionally, region of an image which is consistent with each other should be in same segment. In semantic segmentation each pixel is labeled and classified. So roughly speaking, each segmented region of the image will belong to particular class, including non-objects.



2. System Description

2.1 Hardware

In our current setup, we have employed a BAXTER (by *rethink robotics*) robot with specifically designed grippers for the picking and stowing task. The mounting for BAXTER has also been modified for the given task. Apart from this, the vision system is specifically maintained by input from Microsoft Kinect. There is another movable setup which serves the purpose of an automated shelf. This automated shelf extracts bins and places it in front of the BAXTER. Each bin contains a particular number of items. The Kinect is placed vertically above the table of the automated shelf. The object recognition is performed in this platform. The platform is also movable in 3 directions. Based on the position of the bins, the platform moves and takes out the required shelf and places it in front of the BAXTER for easy grasping and placement. The detection operation done by a nearby high power GPU enabled workstation. Apart from the PC screen, the segmented object can also be viewed on BAXTER screen.

Adding to this, there is a temporary artificial roof for preventing unwanted illumination for object recognition. Moreover, artificial lightning has also been introduced in order to provide a fixed illumination. Since only a 2D image recognition is required, input from only one KINECT is considered.

It's worth mentioning that the workstation is Desktop which runs CUDA enabled NVIDIA GTX 1080 for high speed evaluation from the checkpoints provided from training.



Figure 2.1: BAXTER is one of the most popular robots used by the research community. This is a representative image of BAXTER, and the grippers were custom made for the modified Picking and stowing task.



Figure 2.2: Front view of the setup. The small black box hanging above is the KINECT. The shelf can be seen on the right side of the image.

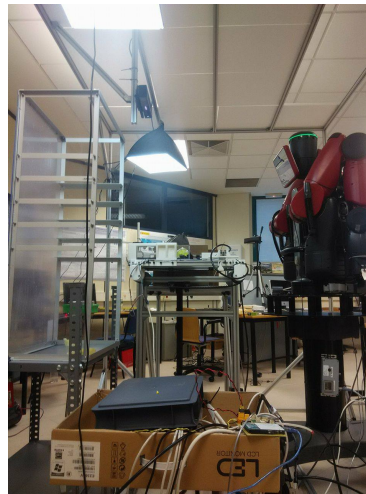


Figure 2.3: Side view of the setup showing the shelf control box and the BAXTER.

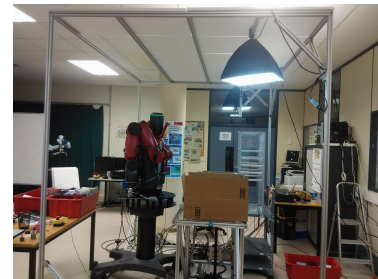


Figure 2.4: Side view of the setup showing the entire platform. The illumination setup and the temporary ceiling can be seen which obstructs light passing through it.

2.2 Software

The BAXTER is operated by ROS (Py) module from the desktop workstation. Apart from this, OpenCV libraries and Python Image Libraries has also been extensively used for image post processing.

Primarily, Tensorflow libraries has been extensively used for the implementation of YOLO and RESNET in Python. The training algorithm creates a checkpoint file after sufficient epoch run of the program and is transferred to the BAXTER workstation. The checkpoint file is nothing but the trained variables of the deep neural network.

The movable platform is controlled by ARDUINO based module and extensive C++ libraries have been used for moving the setup.

2.3 Work Pipeline

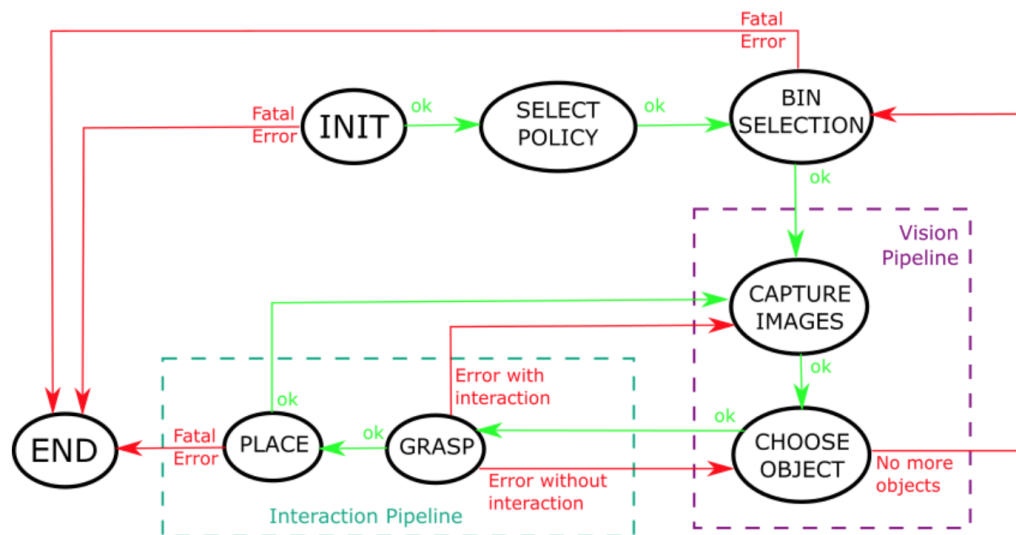
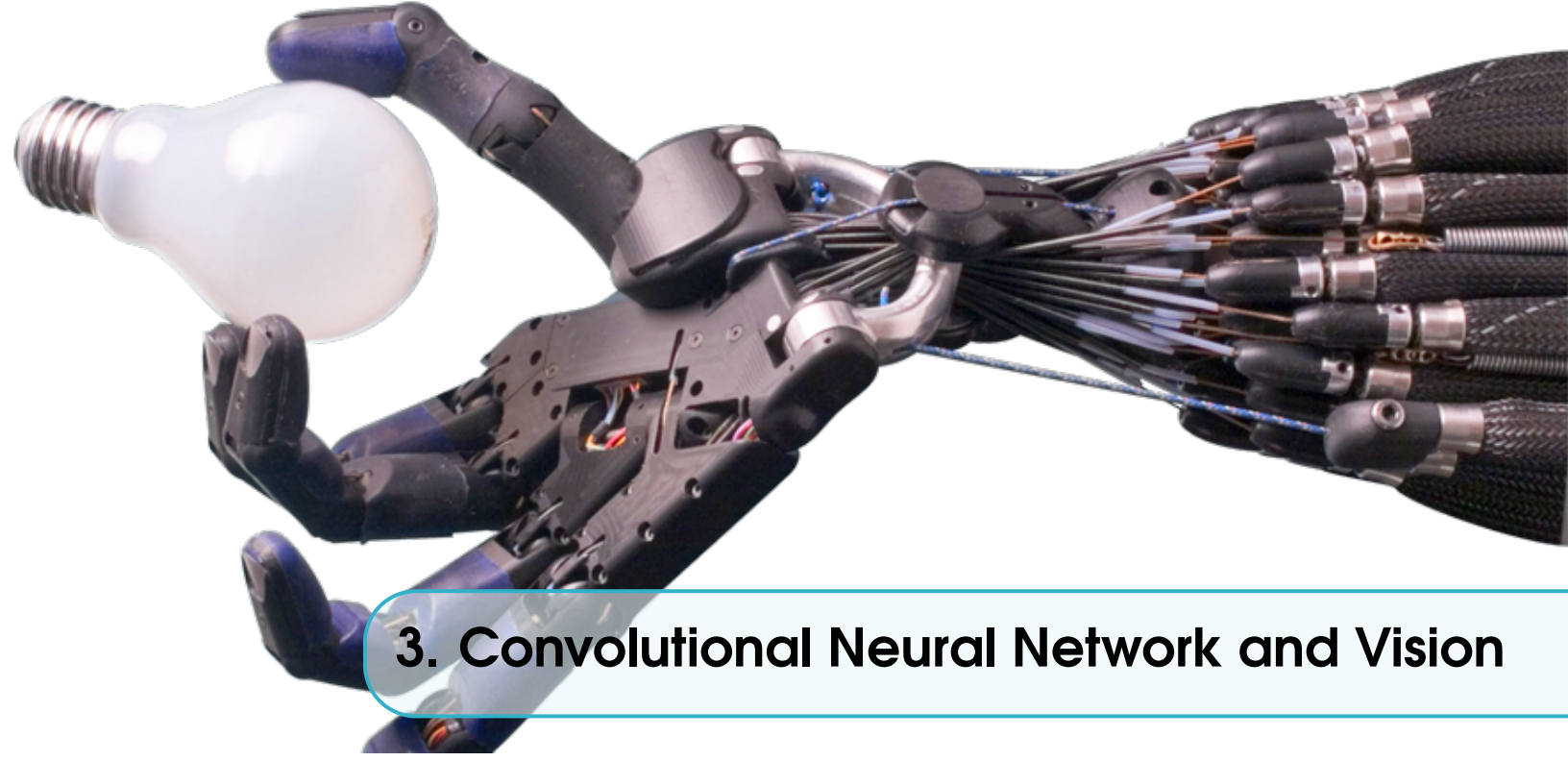


Figure 2.5: Total working pipeline for the BAXTER for object recognition, decision and picking Duran (2017).



3. Convolutional Neural Network and Vision

Convolutional neural networks are considered as a convenient medium to train image dataset rather than the usual perceptron based network. This is a comparatively newer area of research and is most widely used for object recognition, denoising and localization. CNNs are becoming very popular in research community and in conjunction with the development of very powerful GPUs, humongous number of layers can be added at a go and computed at a reasonable amount of time. In this section we will develop a brief focus on the initial stages of the CNN based algorithm and how the segmentation based method was selected after trying out several methods.

3.1 Brief introduction to Machine Learning

A short introduction to neural networks and its implications are provided here in brief. For further detailed reading, detailed explanation is suggested(Nielsen (2016)).Neural networks are biologically-inspired programming architecture which enables a computer to learn from observational data. The basic unit of a neural network is the simple perceptron. A perceptron contains several parameters which are variables and changes with respect to the given training data minimizing the loss function with respect to the expected output in the training set. A neural network consists of primarily three different type of layers -input layer, hidden layer(s) and output layer. The input layer is always equal to the number of different classes in a dataset. The total number of hidden layer(s) are a variable and depending on the number of classes, compute capability and the type of output and vary from one layer to hundreds or more number of layers. The functional unit of each layer is the perceptrons. As the training data is fed through the network, the hidden layers take in the data and changes its variables according to the expected output and minimizing loss function. So basically, by training it is meant that the variables of the perceptron are being fixed with the help of training dataset. These structures do not only contain a high number of layers but can also be structurally complex. Since the discussion of complexity is not the primary concern of this report, it is suggested to go through the aforesaid reference for further detailed analysis.

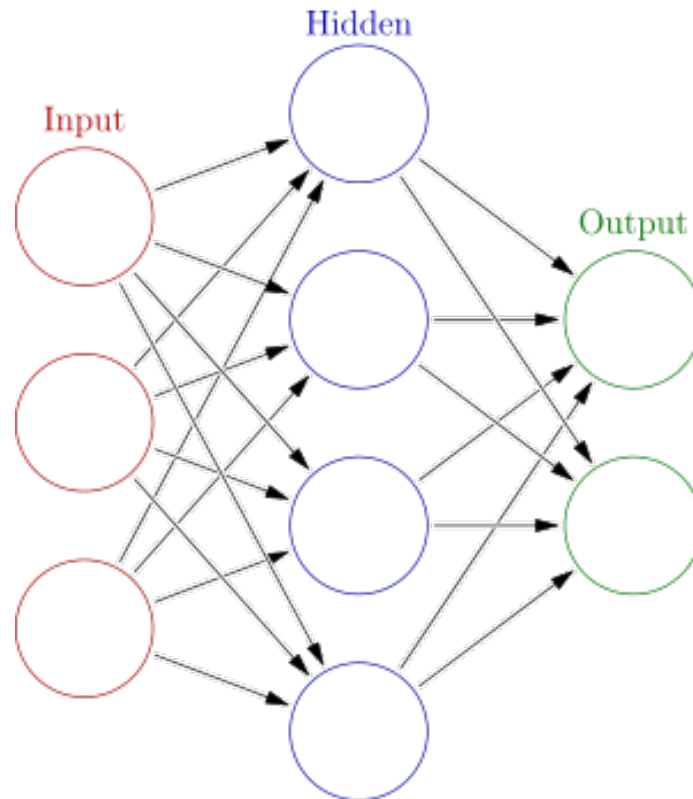


Figure 3.1: In this image it can be observed the input layer, hidden layer and the output layer of a basic neural network. Source: Wikipedia

3.2 Deep Learning

Deep learning is nothing but a powerful set of techniques for efficient learning in neural networks. Deep neural networks are often much harder to train than shallow neural networks (Nielsen (2016)). In this section, we will briefly forage into the popular deep convolutional networks. To give a perspective, if we have an image of dimensions h,w ; the number of input neurons for the network would have been a value of h multiplied by w . The next step would have been to train the biases and variables of the network so that it can correctly classify a given test data with the help of those variables. These approaches have been tried before for the classical datasets and has given average to satisfactory results. After the onset of deep learning in 1998(Lecun et al. (1998)), these results seemed to achieve near perfection with deep neural network.

3.2.1 Convolutional Neural Network (CNN)

To start with, it is a little impractical to use networks having fully connected layers for the classification of images. It is to be noted that the fully connected layers don't take into account the spatial nature of the image, which constitutes its pattern and structure. As rightly mentioned by Nielsen, it treats input pixels which are far apart and close together on exactly the same footing (Nielsen (2016)). Hence, a network is required which is suitable and well adapted for classification of images.

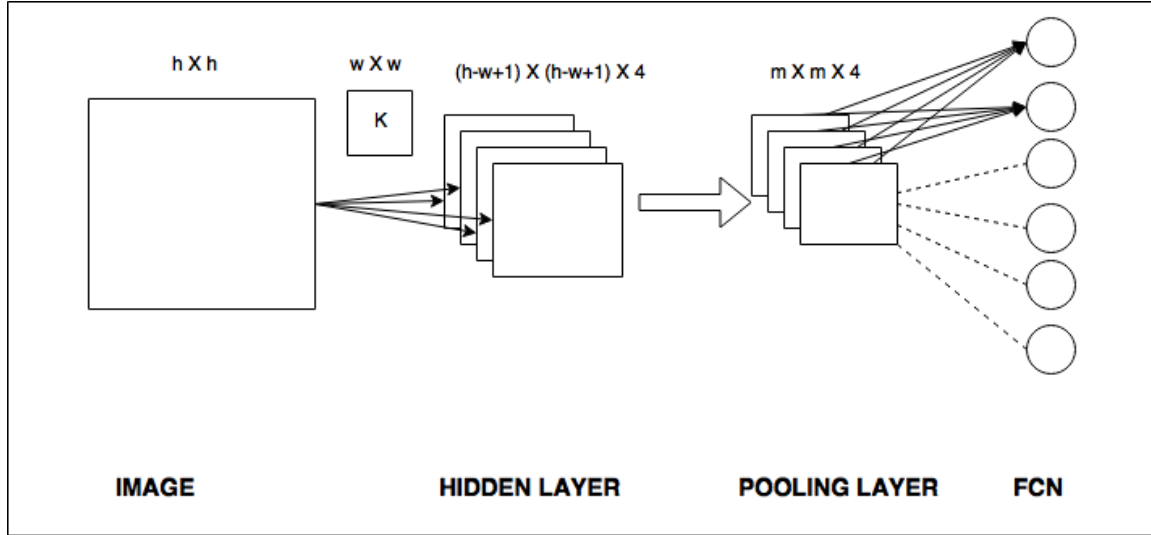


Figure 3.2: K denotes the convolution matrix kernel which slides over the Image input. Several of these kernels having different weight perform convolution and produce a set of hidden layer matrices, which are basically hidden neurons. Each of them detect a particular feature. Then it is followed by max pooling, and at the end it consists of a fully connected layer which finally provides the verdict for classification.

This is exactly where CNNs come into action.

Unlike a single line of neurons, the CNN treats the images with a series of neurons in a rectangular shape, having dimensions same as the image (or it might vary depending on the depth of the layer). The pixel intensities are the input to the first layer of the CNN. Next, this is connected to a layer of hidden neurons. But before connecting them, a smaller input for the next layer is prepared. This hidden region is called local receptive field. Each connection to the next layer corresponds to a weight, which it learns with every loop run. Basically, a convolutional operation is done on the image having a particular dimension and the resultant matrix has a dimension lower than the input image matrix. Convolutional operation is performed by a smaller window and the movement can have different stride lengths.

Each hidden neuron corresponds a bias and a matrix of weights associated with it. So for the entire single hidden neuron matrix, same weights and biases are used. For j, k th hidden neuron, the following equation provides the output (Nielsen (2016)):

$$\sigma \left(B + \sum_{x=0}^C \sum_{y=0}^C w_{x,y} a_{j+x,k+y} \right) \quad (3.1)$$

σ denotes the activation function, B is the bias, w denotes the weight matrix and a denotes the input activation at a particular position of the image. For this particular property, the first hidden layer detects the same feature. It will detect that particular feature irrespective of the position of the feature point present in the image. This is the reason if a particular object is present in a different orientation, the network is supposed to detect it anyways. Hence, it shows a translational invariance feature. This matrix which includes weights and biases are also called kernels. The resultant map or matrix is called feature map. For better recognition through feature, several feature maps are required.

Theoretically, more the number of maps, more the number of features the particular network can detect. Of course, the most prodigal advantage for this kind of network is that it requires much lesser number of parameters than its shallow neural network counterpart.

Next, we have pooling layers. The pooling layer prepares a condensed feature map. It literally takes a small collection of pixels and summarizes it's values. The result is a smaller resultant matrix. The pooling layers are situated just after the convolutional layer. Basically, in broader sense it throws away the location information for the feature and more importantly, decreases the number of parameters eventually. There are many techniques of pooling such as max-pooling and L2 pooling. At the end, there is fully connected layer (FC). It connects all the neurons from the last layer to each of the neuron in the FC. Basically, depending on the cumulative score, the particular pattern or object found is eventually classified.

3.3 Problem of Localization and initial approaches

As of now, only classification has been discussed. Basically, a primitive CNN can only tell whether a particular object is present within the entire image or not. This is not what we are eventually looking for, but can be used as an initial step. So, the last layer basically provides us a score or probability of a particular object being present in the entire frame. Our final output should be exact co-ordinates of the location of a particular target object. It maybe in the form of a bounding rectangle localizing the object within the image or a segmentation mask showing precisely where the object is, on a pixel level accuracy. Before delving into that, a sliding window approach has been developed initially. The main idea is simple. Take a particular target object, slide the window throughout image and the frame providing the highest score should theoretically be giving us the exact location of the object. Due to time constraints for multiple objects and due to very high rate of false positives, this idea has been found to be non-feasible in case of our application ¹.

3.4 Single shot detector : YOLO

Single shot detectors usually provides a bounding box around the target object within the image frame. There are many such earlier works such as RCNN, Fast RCNN(Ren et al. (2015)) , SSD(Liu et al. (2015)) and YOLO(Redmon et al. (2015)). In our context, we will be exploring the possibilities presented by YOLO (You only look once) detector.

There are many reasons for selecting this particular algorithm for exploring bounding box localization algorithms. In literature, YOLO is considered to be extremely faster than other contemporary algorithms which makes it perfectly compatible with real time systems. Apart from this, the results shown by YOLO with respect to SSD is highly impressive when used against the test images of VOC 2007-2012 dataset and COCO dataset. YOLO showed a much higher accurate prediction percentages than the other single shot detectors². This is the reason why we will explore our possibilities of using YOLO in the context of the Robo-picker.

Before going into details of YOLO, we will develop a general idea about how single shot detectors work. These methods usually localizes objects with regression. In short, instead of returning a class, it returns a bounding box (such as x, y, h, w). We train this system with an image having ground truth box as the label, and calculate the loss between ground truth and the predicted

¹A real time demonstration can be seen here: <https://www.youtube.com/watch?v=ncdjBpDF6RQ>

²Performance list of different bounding box detectors can be seen here <https://pjreddie.com/darknet/yolo/>



Figure 3.3: YOLO trial on CMU kitchen dataset after 8000 iterations. This is a sample run on cluttered dataset example. This is to check the validity before the original test on AMAZON dataset.

bounding box. In short, an FC is attached at the last convolution layer. The prediction is usually the coordinate of the bounding box.

To give a general idea, single shot detectors usually convert an image after a series of convolution and pooling into a very small matrix, but having a very deep tensor. The main input image is basically squeezed into a smaller matrix after layers of convolution and usually each tensor component at the end represents a particular block of the image. This last layer is connected to a Fully connected layer (FC). In the FC, loss is calculated for each box with respect to the ground truth and the bounding box is eventually determined.

The original version of YOLO is a little less precise, but it has been improved in version 2. The main advantage of this detector is that it is much faster than its counterpart. The input image is generally converted into a 448 x 448 matrix. It is subsequently fed into the network and the output is filtered

by NMS (Non-max suppression algorithm algorithm). YOLO consists of 9 convolution layers and after the 6th maxpool layer, the input image is gradually resized into 7×7 matrix. The final layer tensor dimension is determined as:

$$\left(S * S * (B * 5 + C) \right) \quad (3.2)$$

In short, the image after 9 layers of convolution and pooling, is resized into a dimension of $S \times S$ grid having B bounding boxes and C number of classes. So if we want to classify 40 objects and take $B = 5$ and $S = 7$ (Redmon et al. (2015)), then the dimension of the final tensor will be $7 \times 7 \times 50$.

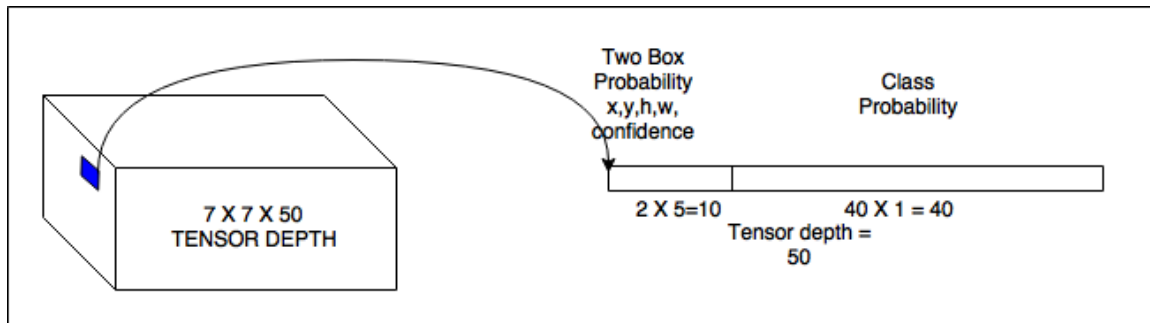


Figure 3.4: Final layer of YOLO. The image is squeezed into smaller matrix, but due to the depth of the tensor matrix, a significant amount of information is held intact.

The 7×7 tensor will now have two (variable C) box information (usually x,y,h,w) and confidence for each box. Apart from this, the rest of the 40 cells represents the class score per box. In short, each cell belong to a particular class and holds the probability information so that it can belong to a particular class. But in order to qualify to a particular class, it has to cross some pre-determined threshold. At the end, with the help of thresholding and NMS, we can filter out the invalid bounding boxes.

3.5 Semantic segmentation and RESNET

Segmentation involves detecting whether a particular pixel has any label associated with it or not. By evaluating that, one can easily determine a map corresponding to the area covering the particular object within the scene. The most basic explanation can be found from the explanation provided from the concept of Fully convolutional Neural Network. In this case, the last fully connected neural network layer is replaced by a convolutional neural network. This actually gives us the general idea of the contents within the image. It is to be noted that the last layer is converted into a convolutional layer with the help of upsampling. Suppose an input image is eventually compressed into a single row vector CNN having a very deep tensor. At the last layer, denconvolution or transposed convolution is used to recover the activation positions which indicates the meaningful data residing within the given image. In other words, it is just scaling up of the activation layers into a meaningful image having the region of interest around the location of the object. The upsampling step has some learnable parameters as well. Other precise methods include 'extreme segmentation'(Santos (2016b)) which is highly time consuming and it uses a method of unpooling for upsampling of image.

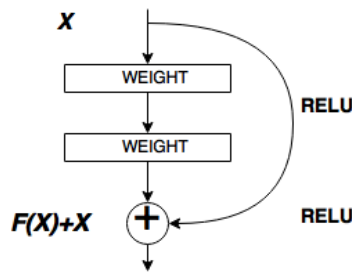


Figure 3.5: Basic building block of RESNET function

In this work, we will work on RESNET, a CNN based segmentation algorithm. Although having a deep architecture, it is much more efficient in terms of IOU with respect to its other semantic segmentation counterparts. It shows a very high efficiency on VOC 2007-2012 dataset (Shuai et al. (2016)) and this is the reason the RESNET has been chosen to be implemented on the current problem scenario.

The RESNET is basically a 150 layer deep network consisting of deep residual blocks. It has been seen that more than 30 layer deep network is not suitable for deep end-to-end learning. This is in contrast with the notion that deeper network should produce better results. In case of RESNET, it has been shown that a 1000 layer depth network can have improved results (He et al. (2016)). The actual idea is that it re-routes the input and adds the concept learned in the previous network. The idea is that the next layer would learn from both the inputs of the previous layer and the previous layer itself. It is supposed to further enhance the training method unlike other networks.

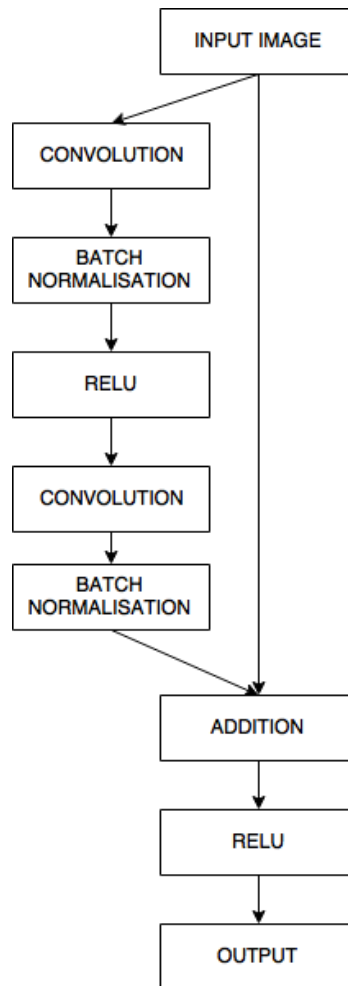


Figure 3.6: Basic building block of RESNET network with n -layers

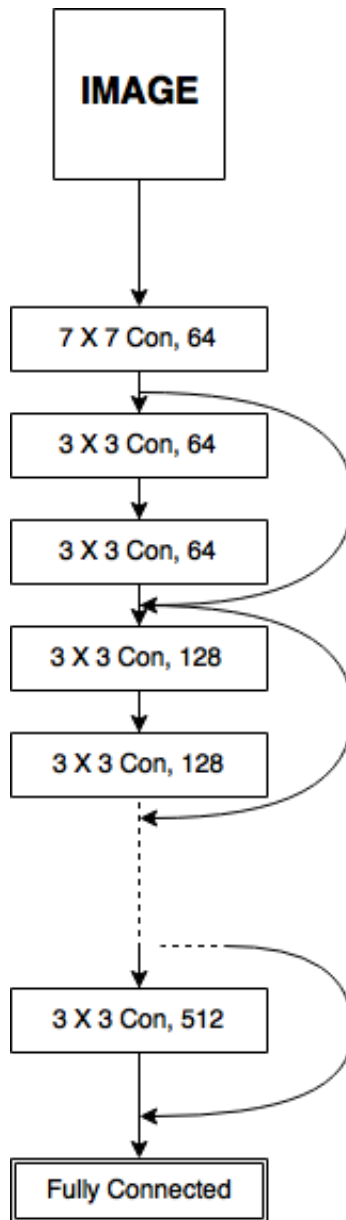
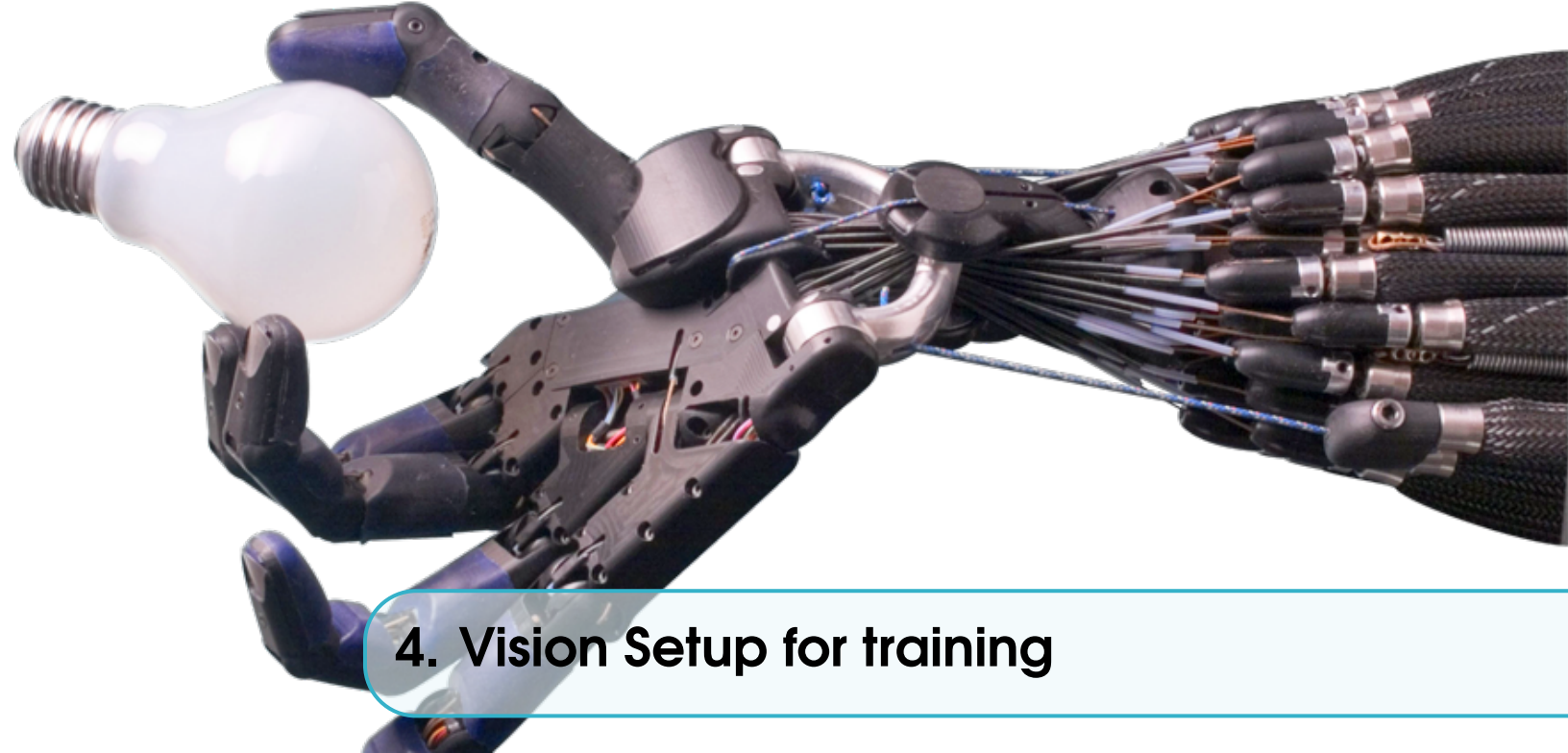


Figure 3.7: Detailed layer of RESNET structure



Figure 3.8: RESNET trial on VOC2012 segmentation test dataset. It should be noted that the different classes are denoted by different colors.



4. Vision Setup for training

Preparation of dataset is one of the most prime aspect for training and testing the deep learning algorithm. It is to be noted here that one should be very careful about the different lightning conditions and resolution of particular items. In this particular case, the ARC primarily faces two challenges- creating a dataset from scratch for the known items category and prepare a work model for acquiring the dataset for the unknown items. The following section will be dedicated to the organization of the Robot vision setup and dataset acquisition module.

4.1 Calibration and setup

In this section, a short highlight on the hardware and it's calibration will be mentioned briefly. For our current module, we have chosen KINECT 2.0 as the image acquisition module. The KINECT 2.0 has the following specifications:

- *Depth Sensing*: 512 x 424 ,30 Hz ,FOV: 70 x 60, One mode: .5–4.5 meters
- *1080p color camera*: 30 Hz (15 Hz in Low light)
- *New active infrared capabilities*: 512 x 424 ,30 Hz
- *Sensor dimensions, LxWxH*: 9.8" x 2.6" x 2.63" (+/- 1/8"), 24.9 cm x 6.6 cm x 6.7 cm, Length: The KINECT cable is approximately 9.5 feet (2.9 m) long, Weight: approximately 3.1 lbs (1.4 kg), Sensor FOV: 70 x 60

These specifications provide enough resources for acquisition and training of a set of image dataset. But before that, The device needs to be calibrate in terms of RGB color, IR and depth measurements. This particular task is performed by the ROS enabled toolbox for calibration of KINECT 2.0, known as the IAI toolbox.

Next a series of more than 150 images of a checkerboard for known dimensions were taken in succession and saved in a particular directory specifically built for Kinect2bridge (IAI) module. After the toolbox has succesfully calibrated the same, one can test the distance and make a comparative visualisation to check approximately whether the device has been successfully calibrated or not (Wiedemeyer (2015))¹.

¹Kinect 2 calibration guide: [://github.com/code-iai/iai_kinect2](https://github.com/code-iai/iai_kinect2)



Figure 4.1: The latest KINECT v2, used in our current application. Source: Microsoft XBOX

4.2 DATASET specification

The datasets consists of primarily two parts- the main RGB image and its corresponding label. The primary focus in this section will be the datasets used for running YOLO and RESNET. In case of YOLO, the label information came in the form of a four point coordinate within an image, and the object name. This information is stored in an XML file. The format for this dataset is exactly similar to PASCAL VOC dataset for localisation (Everingham et al. (2012)). It is to be noted that the first attempt to recognize an object with low number of CNN layers had no localisation data. Hence in order to overcome that, a manual localization algorithm was used which involves the sliding window concept. Due to significant performance issues, we switched on to better localisation algorithms such as YOLO and RESNET. It is to be noted that in case of the simple CNN, there was no label information.

RESNET uses a labeled mask for object segmentation. The label is denoted by another RGB image having only single unique intensity shade around the segment space. The neural network takes in data in conjunction to the original RGB image.

The dataset were prepared manually for YOLO. The different images were localized with the help of software and the corresponding XML files were produced. In case of RESNET, the dataset was artificially generated². Different objects were segmented out in a particular resolution and artificially placed in an expected background with known orientation, brightness and illumination. The corresponding label masks were also produced in sync with the RGB images.

4.3 Training time and practical constraints

Usually, there are different recommended epoch runs for each and every localisation and segmentation module. In case of YOLO, it is recommended that the algorithm be run for atleast 100000 times for 20 classes (Redmon et al. (2015)). In case of RESNET, the recommended epoch run is 20000 times for 20 classes classification. The operations were commenced without the use of GPU cards and and it took immense computation time (3-4 days for the minimum epoch runs). With the integration of the systems with the GPU (NVIDIA GTX1080). Approximately the execution speed for RESNET algorithm is 90 seconds per step; while with GPU it is 0.9 seconds per step. In this context, a checkpoint file of the trained variables has been used after being trained for 4500 steps.

²Our dataset generator: https://github.com/vasilya93/arc_image_generator

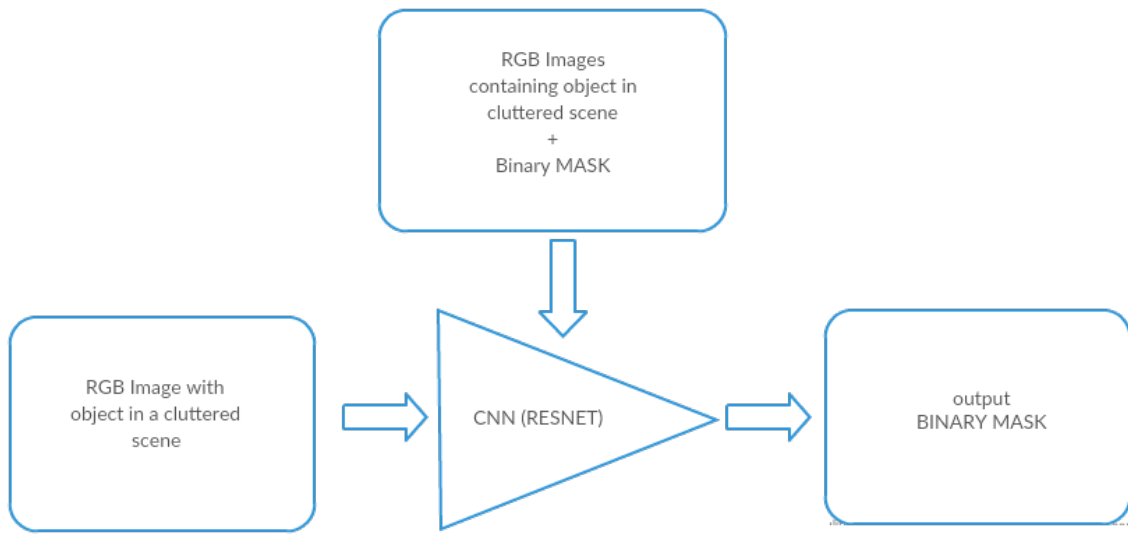


Figure 4.2: Graphical structure of the working dataset. It is to be noted that the Binary masks are labeled with respect to their intensity value and corresponding assigned item number.

4.4 Dataset input and labels

The input data includes the target item in a cluttered scene, and a corresponding mask. The corresponding mask is a binary image having a blob around the particular item. The CNN is trained and it takes a cluttered scene as an input and produces a corresponding mask indicating the number of objects it has eventually found, in different colored shades. The dataset is produced by a repetitive algorithm which places different objects in different positions in the clutter. There is exactly 39 items in the list and the total number of images used in this scenario is 10000. Correspondingly, 10000 binary mask images has also been produced for the purpose of giving the position to the CNN. The images of the objects were taken in separately, segmented and a transparent alpha channel was added as a background. It was then placed artificially in the cluttered scene, and a series of images were produced with multiple objects in them, as shown in Figure 4.3. In short, items were placed artificially in different cluttered and non-cluttered backgrounds.

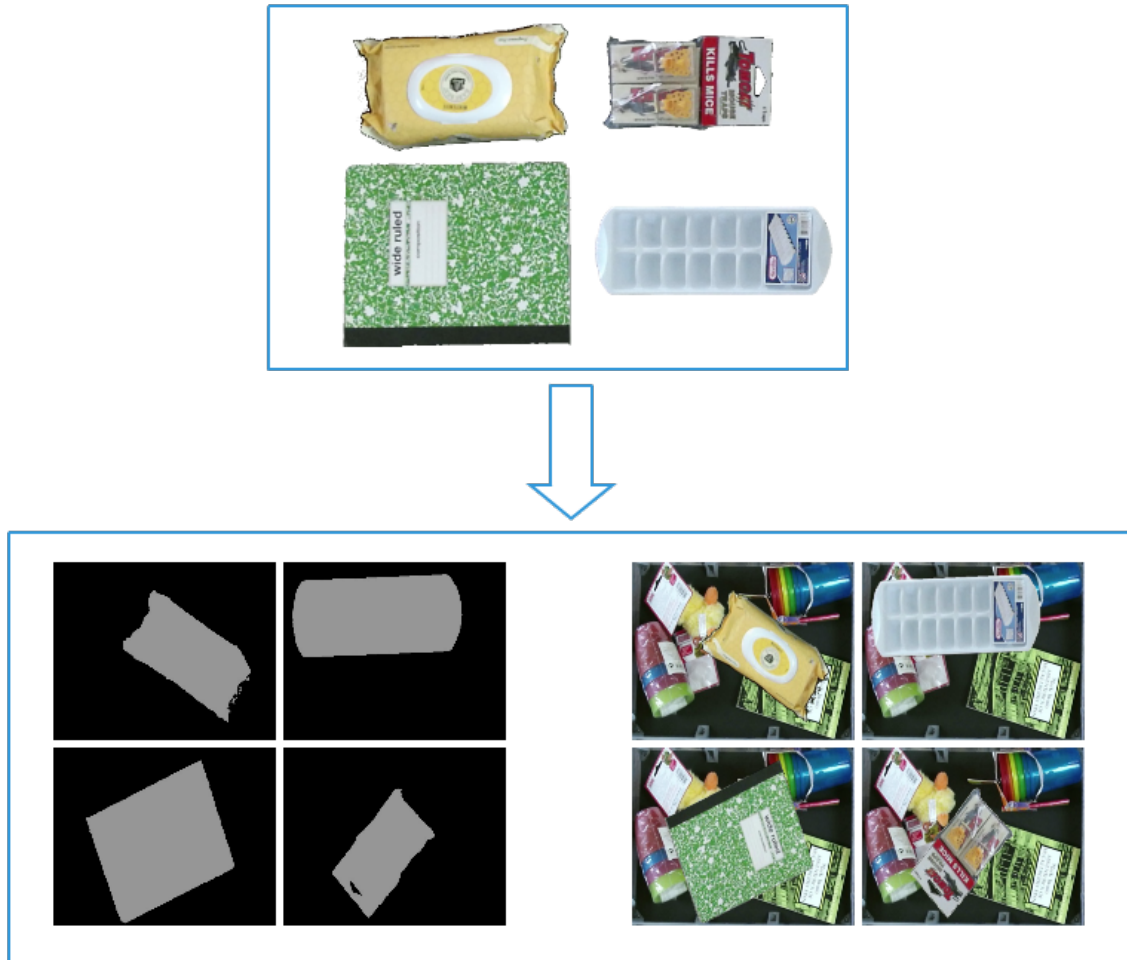


Figure 4.3: At first, only the object images are separated, by adding a transparent alpha channel. Then they are placed in artificially in cluttered scene as it's background. They also have a corresponding binary mask. The binary shades have different values depending on the item. This particular color value is the label of the image. Here it has not been shown for better comprehension. The last block is used as an input to the RESNET.



5. Object Recognition for picking

As the dataset module has been taken care of, it is time to test the algorithms in the first place. As of now, this work gave a brief introduction with multilayer perceptron based sliding window localization, YOLO based bounding box localization and finally the RESNET based segmentation. In this current segment, major focus will be on the evaluation of the aforesaid algorithms in terms of robot vision application in grasping. Although efficiency metrics exists which generally prescribes the efficiency in the light of computer vision, in our case we will be evaluating it in terms of grasping. In the following sections we will begin with some results of YOLO and RESNET. Later, some evaluation metrics will be discussed for final evaluation and justification of the validity of given algorithms in context of robot vision.

5.1 The challenge and the plan: revisited

We revisit the vision module of the AMAZON picking challenge as a brief overview. The challenge is to detect objects from a very cluttered scenario and the sole objective of the vision module is to present a very accurate mask or a filter indicating the presence of the object within the frame. This mask will filter out the point-clouds which should be belonging to the objects only and this will be subsequently used by the grasping module to detect and grasp an object. The target is not to extract a region which will have the entire object in it, but to extract a region of interest which solely belongs to the object, without any redundant region surrounding it. Otherwise, in spite of detecting the correct item, the gripper might pick up the wrong object from the redundant region due to minute positional error of the gripper itself (see Figure 5.1).

5.2 YOLO: Observations

A list of fixed items were collected from the object list of the AMAZON picking challenge. The tensor parameter was fixed according to the number of classes in the object list. The results are being depicted in Figure 5.2 and Figure 5.3. An alarming number of objects has been missed by the algorithm as shown by the results. It is to be noted that with a few exception, there is a huge



Figure 5.1: YOLO output of book. Although detection is done with a high accuracy, there is a lot of redundant space inside the ROI (Region of Interest). The purple region denotes the redundant space and any wrong item, if present might be picked up by the gripper due to detection of point-clouds not belonging to the item in the ROI.

redundant space within the bounding box which does not belong to the object, as cited in the previous section.

5.3 RESNET: Observations

A target object was chosen and the algorithm was run on an image frame containing the items. The resultant images show very promising results. The object space has been accurately extracted through semantic segmentation (see Figure 5.4 and 5.5). The grasping module now has a very accurate set of cloudpoints to choose from.

5.4 Results and Evaluation

In the following sections we will discuss the results and the evaluation with respect to each algorithm. This will help us in concluding the practicability of using an algorithm for picking task by a robot.

5.4.1 Evaluation metric

In order to evaluate the two algorithms, we will be using Intersection Over Union (IoU) and Visual Grasping score (ViG). Usually IoU has been used conventionally to evaluate localization capability of an algorithm. But our target is to evaluate whether the information provided by the vision module is useful for grasp module or not. For this, we need to know what percent of the total detected pixels are originally within the actual object space? However small the space is, if the detected area lies within the ground truth; it should theoretically provide us full grasp-ability. Hence, keeping this particular condition in mind another metric has been developed which will provide a more accurate prediction of vision based grasping and subsequent picking. The ViG is nothing but the fraction of the detected area within the ground truth. Refer to Figure 5.6 for visual depiction of the metrics. Usually, it is expected that an algorithm must have an IOU greater than 0.5.

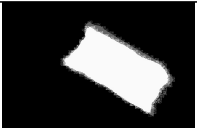




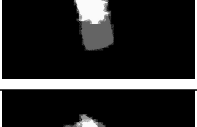

5.4.2 Evaluation of YOLO



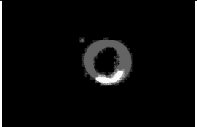


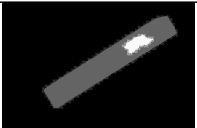





The YOLO algorithm has been tested with respect to the labeled ground truth. Refer Figure 5.7 for sample examples on images. The mean IOU after evaluating all of the items was found out to be **0.255**. Additionally, The ViG score was found out to be **0.405**.


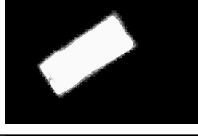








5.4.3 Evaluation of RESNET

The RESNET algorithm has been tested in accordance to the labeled ground truth mask. Refer Table 5.1 for sample examples on images and the score. The mean IOU after evaluating all of the items was found out to be **0.49**. Additionally, The ViG score was found out to be **0.94**.

Table 5.1: IOU and ViG score of RESNET segmented mask for different objects. The gray part denotes the part of ground truth which is not detected. The white part shows the detected area by RESNET.

Item	IOU score	ViG score	Figure
Baby wipes	0.85	0.98	
Notebook	0.56	0.99	
Marble bag	0.59	0.98	
Hinged rulebook	0.74	0.99	
Hanes socks	0.42	0.99	
Water bottle	0.46	0.91	
White bath sponge	0.42	0.96	

Toilet Brush	0.17	0.95	
Mousetrap	0.39	0.99	
Duct tape	0.14	0.97	
Glue Stick	0.75	0.99	
Wine glass	0.69	0.97	
Reynolds wrap	0.08	1.00	
Scissors	0.22	0.94	
Salt pack	0.74	0.83	
Irish-spring pack	0.32	0.99	
Speedstick	0.09	1.00	
Face-cloth	0.36	1.0	

Red Joke book	0.89	0.99	
Ticonderoga pencil pack	0.86	0.99	
Scotch bright	0.47	0.99	
Spoon	0.02	1.00	
Kleenex	0.74	0.99	
Pie plates	0.83	0.98	
Tennis ball	0.80	0.99	
Robot DVD	0.90	0.99	
Crayons	0.60	0.99	
Balloon Pack	0.21	0.89	

Algorithm	IOU	ViG
YOLO	0.255	0.405
RESNET	0.49	0.94

Table 5.2: Mean IOU and ViG score of RESNET and YOLO algorithm

5.5 Discussions

As discussed earlier, the results procured from RESNET algorithm satisfies our requirement quite nicely. It is to be noticed that the primary focus of this work is not to make a comparative analysis of the given algorithms, but to highlight the way how we reached a solution for our own requirement and functionality. The target of this work was to produce a very accurate map of the point-clouds which completely belongs to the particular object. In our case, we have utilized RESNET in order to do that as it doesn't produce a redundant area within a bounding box like YOLO. This information has been of supreme importance for the grasp module. The APPENDIX section shows the functionality of the robot while utilizing the RESNET algorithm.

We can see a significant jump in performance when we apply a semantic segmentation network for vision for picking task (see Table 5.2). In this case, it is to be noted that we are more interested in the pixels that belong to the original object and it is important that the pixels belong to the original object only. This parameter is shown precisely by ViG metric. The IOU, although a very significant metric, doesn't show the full picture. The point is, we are not proposing to choose a metric over the other, but use the metrics in congruence with the other as they convey different application information. The IOU displays information about the vision performance, whereas ViG displays theoretical ability of the grasping module to use the visual information for efficient grasping. Both the metrics are interrelated, where ViG can be seen as specialization of the IOU metric itself. In any case, referring to the overall performance, it can be concluded that semantic segmentation based CNNs provide a significant improvement in providing visual information for grasping, in this case it is RESNET.



(a) Robot DVD : uncluttered background



(b) Robot DVD: cluttered background



(c) Gum Set : uncluttered background



(d) Gum Set : cluttered background



(e) Icetray : uncluttered background



(f) Icetray : cluttered background



(g) Red Joke Book : uncluttered background



(h) Red Joke Book : cluttered background

Figure 5.2: The YOLO algorithm produces a bounding box surrounding the box in an uncluttered background. But in case of cluttered background, this method fails in a significant manner.



(a) Green color Pencil : uncluttered background



(b) Green color Pencil : cluttered background



(c) Salt bag : uncluttered background



(d) Salt bag : cluttered background



(e) Socks bag : uncluttered background



(f) Socks bag : cluttered background



(g) Baby wipes : uncluttered background



(h) Baby wipes : cluttered background

Figure 5.3: YOLO produces detection result which is mostly unreliable for object detection. In some cases, as shown above, it detects the wrong object and sometimes outside the coordinates of the frames. It is to be noted that this result has been produced after 57000 iterations.



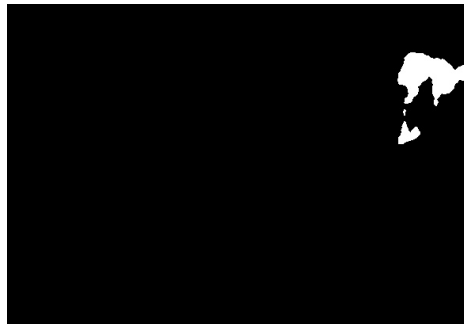
(a) Input target: Ball container



(b) Output: Detected



(c) Input target: Irish Spring bar



(d) Output: Detected



(e) Input target: Yellow Baby wipes



(f) Output: Detected



(g) Input target: Toothbrush

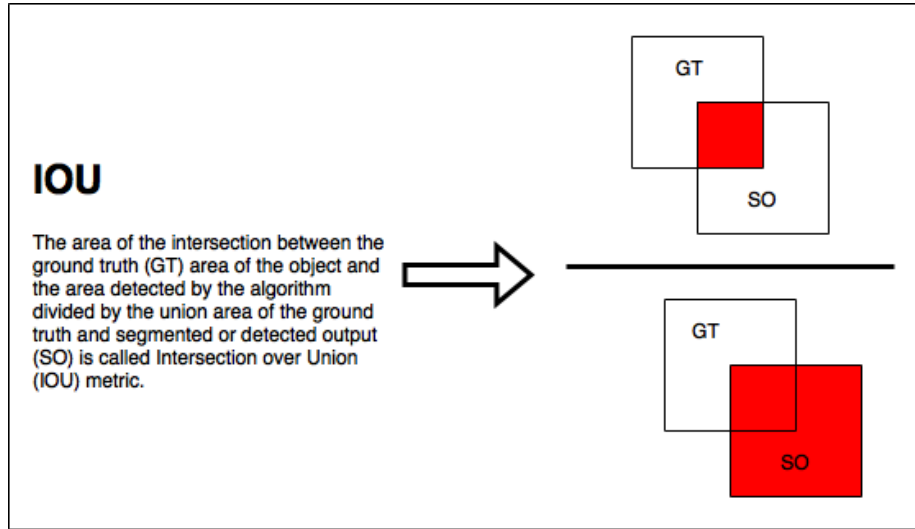


(h) Output: Detected

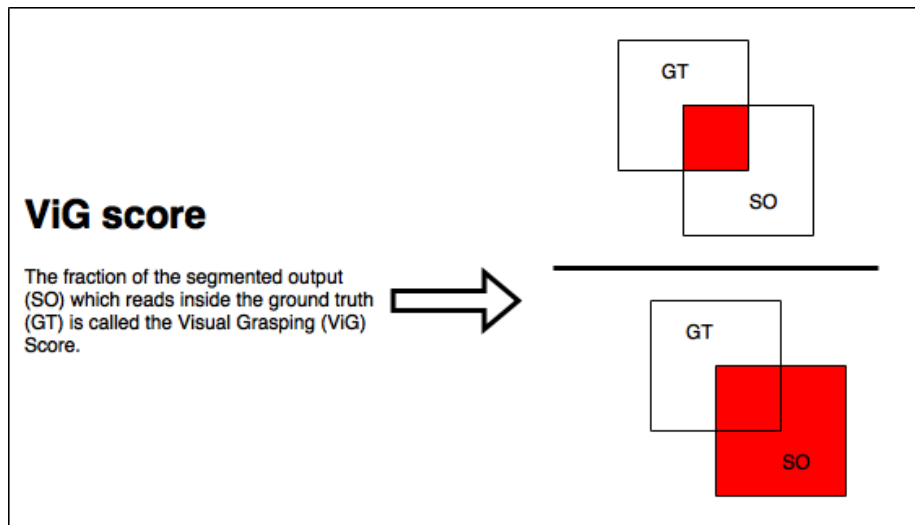
Figure 5.4: RESNET mask produced for the target items.



Figure 5.5: Accurate depiction of the target items through semantic segmentation (RESNET). Total iterations in this case is 4500. This seems to be a concrete option for object recognition for picking task.



(a) Intersection Over Union

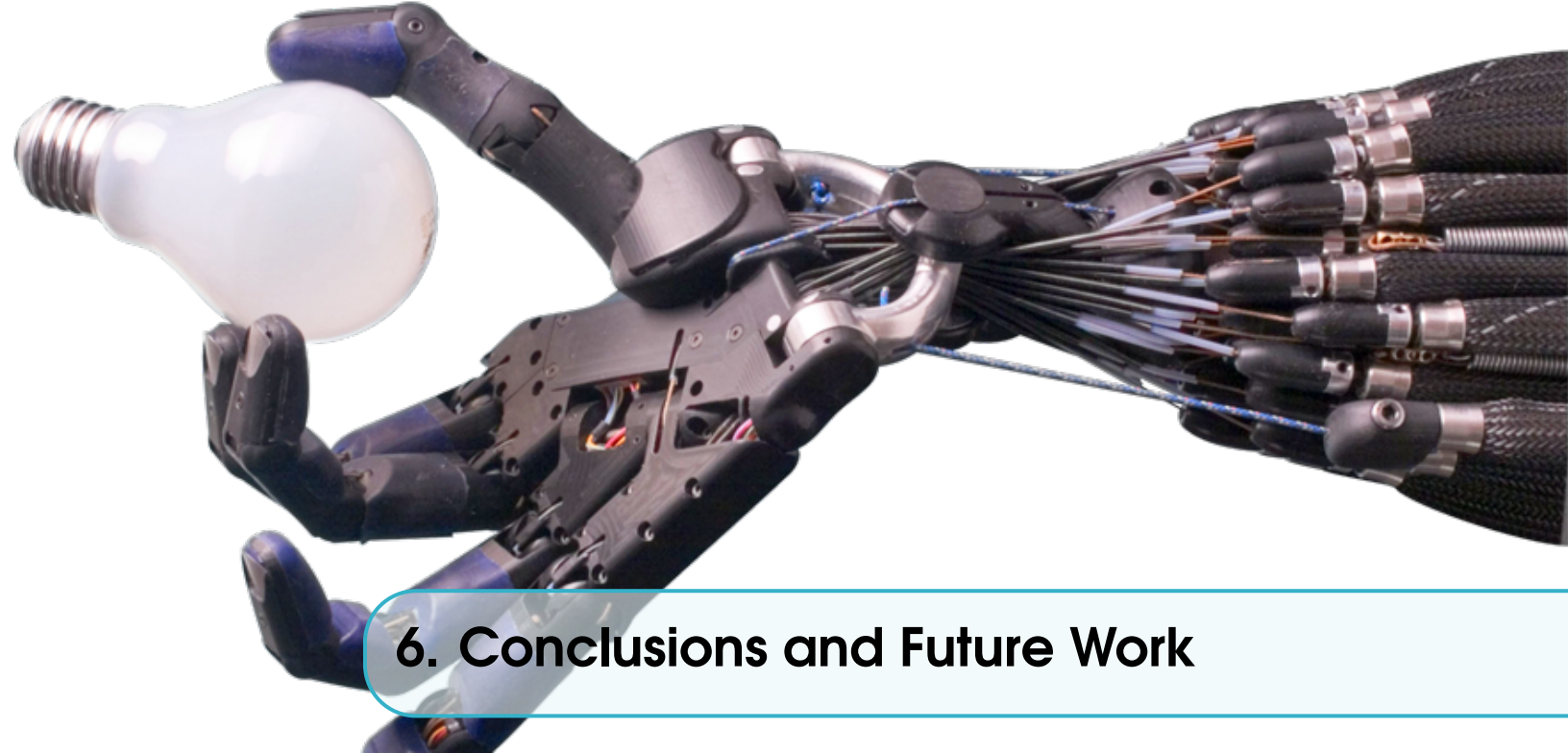


(b) Visual Grasping score

Figure 5.6: Aforesaid figures show metrics used for the evaluation of the algorithms. It is to be noted that our primary concern is the ViG score. Higher score means that more number of pixels are inside the particular object in the image, meaning easy grasp probability by the gripper module.



Figure 5.7: YOLO produces detection result which is mostly unreliable for object detection. In some cases, as shown above, it detects the wrong object and sometimes outside the coordinates of the frames. It is to be noted that this result has been produced after 57000 iterations.



6. Conclusions and Future Work

In this section, we will discuss the conclusions derived from the results and the direction in which we can proceed and enhance the mechanism of object recognition and other related functionality as well.

6.1 General Conclusions

It can be safely concluded from the experimentation that using a semantic segmentation based algorithm is a better proposition than to use a localization method which produces bounding box as an output. As shown before, apart from the detection error we have a redundancy error. In the later case, unwanted pixels are also selected which do not belong to the object at all. This problem has been solved by semantic segmentation. In RESNET based semantic segmentation, the objects have been segmented out very carefully with pixel level accuracy and is free of rectangle shape related error which previously produced the redundancy. Apart from the the detection accuracy increased almost twice than the single shot detector algorithm (YOLO).

Hence in an application like this, where it is needed to select precise 3D point-clouds of objects mostly having irregular shapes, semantic segmentation based methods provide a better filter for segmenting out point-clouds belonging to the objects at large, especially in a cluttered background. The aforesaid work also shows segmentation output of most of the items graphically. In most of the cases the segmentation mask do not cover the entire object, but the segmentation mask almost solely belongs to the object itself which is shown by the ViG score. This gives the correct cloud-points to the grasp module.

6.2 Future Work

There is a huge scope for future work for improving the application base. It is to be mentioned that the primary focus for the future work is not only to improve the algorithm itself, but to improve upon the usability and application itself. We can simply provide a very short list of probable and promising future aspect of continuing this work:

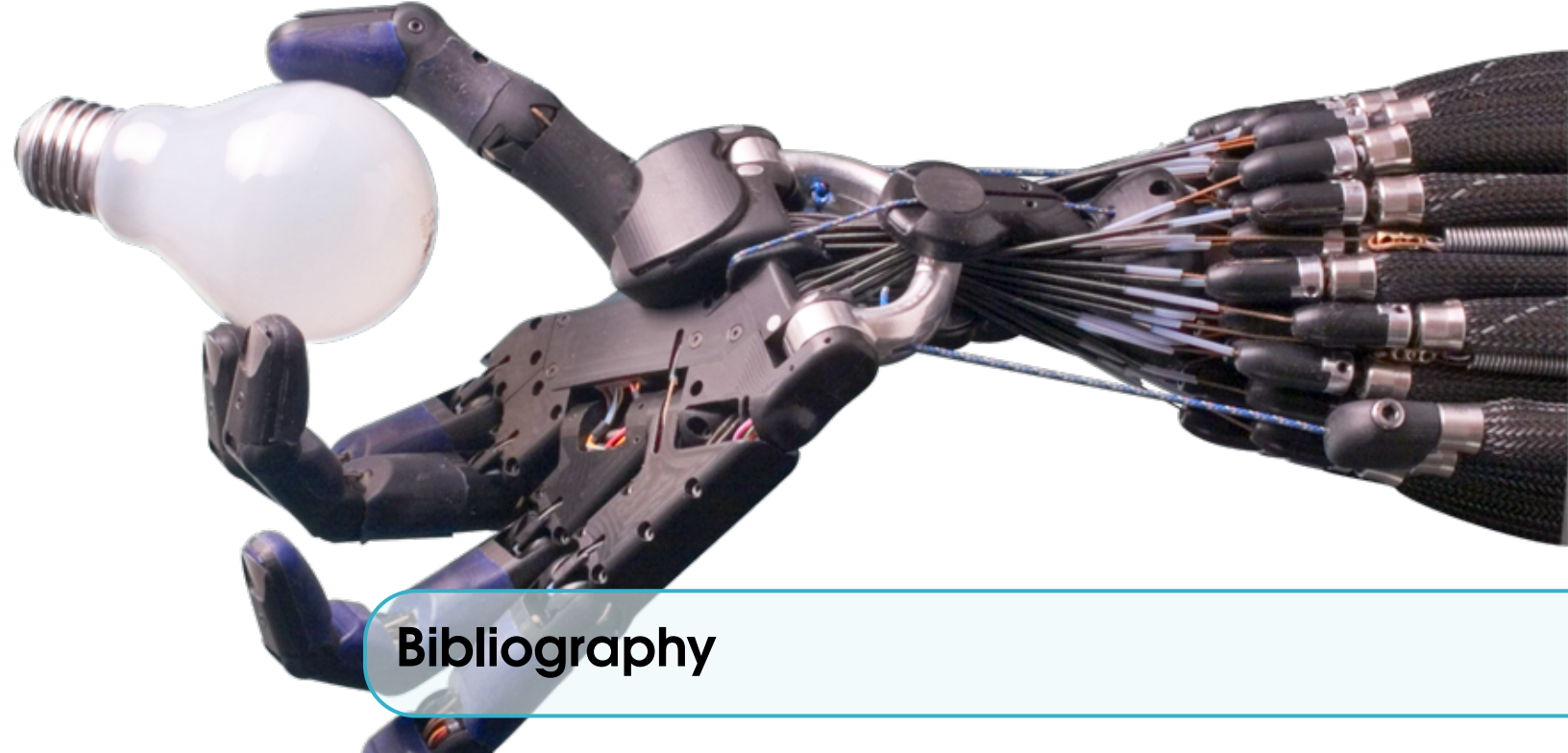
- Fast data acquisition and training
- Compatible with portable devices
- Grasp enabled multi-labeling
- Implementation in other common application

With the advent of faster GPUs, RESNET can be implemented in a very fast manner. In case of unknown items, an advanced dataset acquisition module needs to be developed which will automatically extract the alpha channel based information, subsequently producing the artificial dataset and the masks, train it on the algorithm via transfer learning and implemented on-the-go. Apart from making the algorithm faster, focus should be made on implementing it on unknown items and implementation in a quick manner. This constitutes not only the algorithm itself, but the peripheral dataset acquisition module as well.

Nowadays, application of detection modules are not confined in research modules only, but in commercial space as well. That includes portable devices and embedded devices. To implement this on hand held devices, memory problem needs to be solved. Object recognition has a huge application in commercial space and works are already underway in order to enhance this sector.

Additionally, a very important aspect of picking is to identify separately the regions which can be graspable. It is not enough to extract the information about the location of the object, but it is also important to produce a grasp map out of the detected frame. For this, multi-labeling of the section of the items are needed to be done. This multi-labeling is a near impossible task for an artificially generated dataset like the current scenario. A much deeper knowledge of the point-clouds are needed so that they can be associated with their corresponding labels. This will enormously ease out the task for the grasp module as it can then easily re-position itself easily in accordance with the position of the graspable space within the object.

There are numerous application for semantic segmentation based analysis. Segmentation of scene for self driving cars, human pose estimation (Varol et al. (2017)) are one of the very few sections where multi-labeling and semantic segmentation based task is utilized for practical application based scenarios.

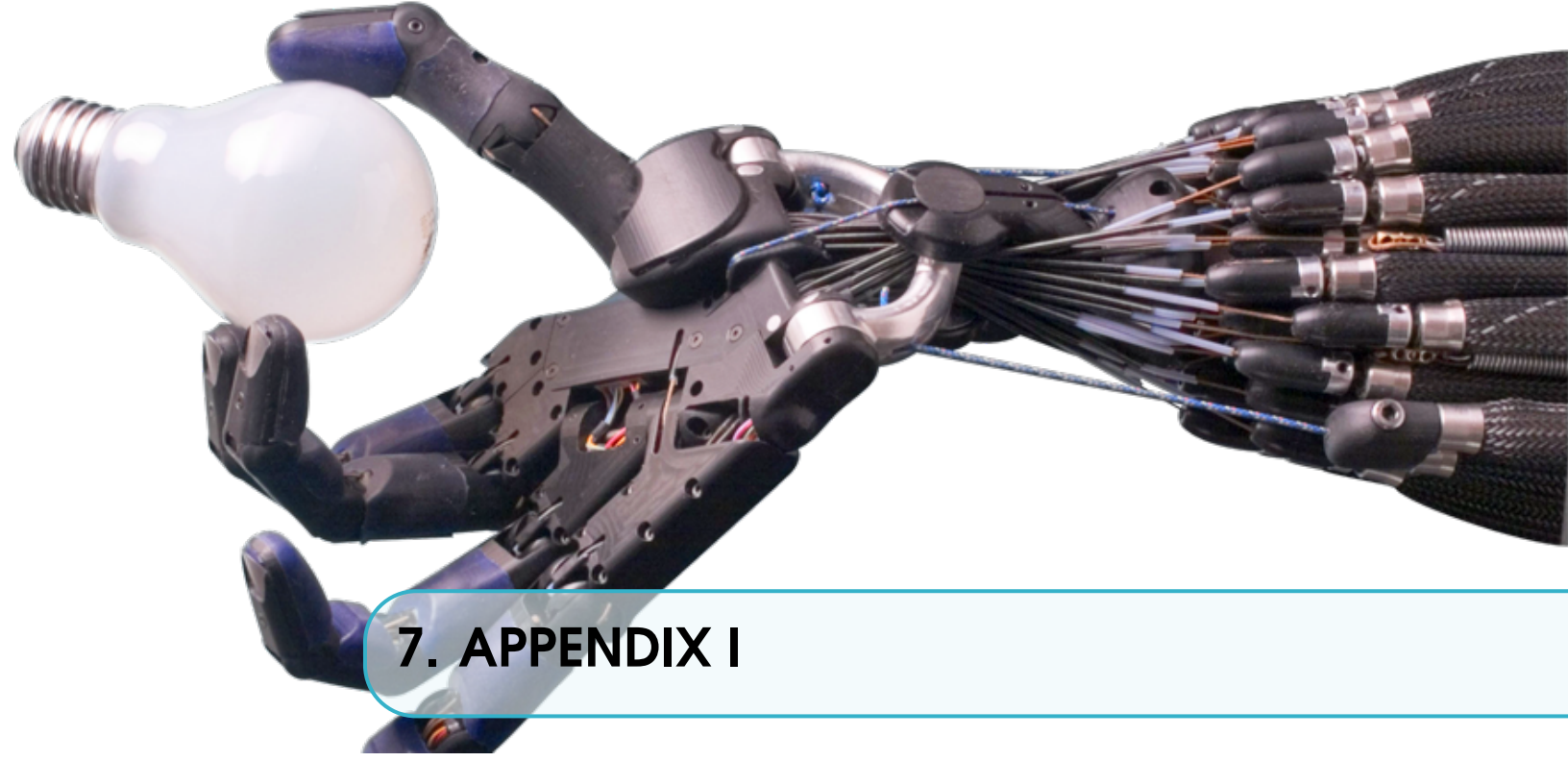


Bibliography

- Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359.
- Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2016). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915.
- Duran, A. (2017). Working pipeline for baxter robot in amazon picking challenge 2017. <https://sourceforge.net/p/robinlab-arc2017/wiki/System%20requirements/>.
- Everingham, M., Van-Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2012). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Jonschkowski, R., Eppner, C., Höfer, S., Martín-Martín, R., and Brock, O. (2016). Probabilistic multi-class segmentation for the amazon picking challenge. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–7.
- Kragic, D. and Vincze, M. (2009). Vision for robotics. *Found. Trends Robot*, 1(1):1–78.
- Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- LeCun, Y., Faret, C., Couprie, C., and Najman, L. (2013). Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1915–1929.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. E., Fu, C., and Berg, A. C. (2015). SSD: single shot multibox detector. *CoRR*, abs/1512.02325.

- Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110.
- Nielsen, M. (2016). Neural network and deep learning. <https://neuralnetworksanddeeplearning.com>.
- Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. *CoRR*, abs/1505.04366.
- Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640.
- Redmon, J. and Farhadi, A. (2016). YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242.
- Ren, S., He, K., Girshick, R. B., and Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497.
- Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision, ICCV '11*, pages 2564–2571, Washington, DC, USA. IEEE Computer Society.
- Santos, L. (2016a). Resnet for image segmentation. https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/residual_net.html.
- Santos, L. (2016b). Semantic segmentation : an overview. https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/image_segmentation.html.
- Santos, L. (2016c). Single shot detector for image localisation. <https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/single-shot-detectors/ssd.html>.
- Santos, L. (2016d). You only look once. <https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/single-shot-detectors/yolo.html>.
- Shuai, B., Liu, T., and Wang, G. (2016). Improving fully convolution network for semantic segmentation. *CoRR*, abs/1611.08986.
- Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgb-d images. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part V, ECCV'12*, pages 746–760, Berlin, Heidelberg. Springer-Verlag.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, abs/1409.4842.
- Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., and Schmid, C. (2017). Learning from synthetic humans. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

-
- Vijay, B., Ankur, H., and Roberto, C. (2015). Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293*.
- Wiedemeyer, T. (2014 – 2015). IAI Kinect2. https://github.com/code-iai/iai_kinect2. Accessed June 12, 2015.
- Xie, S. and Tu, Z. (2015). Holistically-nested edge detection. *CoRR*, abs/1504.06375.
- Zeiler, M. D. and Fergus, R. (2013). Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901.
- Zhang, L., Lin, L., Liang, X., and He, K. (2016). Is faster R-CNN doing well for pedestrian detection? *CoRR*, abs/1607.07032.



7. APPENDIX I

One can have a look at the implementation of the thesis work in BAXTER robot in the following videos:

- **BAXTER picking real time using RESNET and feature matching algorithm:** https://www.youtube.com/edit?o=U&video_id=LZ4DEKmXGb0
- **BAXTER picking in simulation environment:** <https://www.youtube.com/watch?v=cW1BU4sg1A>
- **BAXTER stowing in real time:** <https://www.youtube.com/watch?v=lu654np4IY0>