AIVERO AS Internship

Group 18gr942

Project Report

Fall Semester 2018

Aalborg University
Vision, Graphics and Interactive Systems

Copyright © Group 18gr942, Vision, Graphics and Interactive Systems, Aalborg University 2019

This report is compiled in LaTeX. Additionally is Python, Adobe Illustrator, and Inkscape used to code, draw figures, and charts.

Vision, Graphics and Interactive Systems
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
AIVERO AS Internship

Theme:
Computer Vision

Project Period:
Fall Semester 2018

Project Group:
Group 18gr942

Participants:
Niclas Hjorth Stjernholm

Supervisor:
Thomas B. Moeslund
Raphael Düershceid

Number of Pages: 48

Date of Completion:
January 9, 2019

Abstract:

The goal of Aivero, AS is to develop robot skills in several different settings. The first step of this, is to enable the robot to pick groceries straight from the shelves of a general supermarket. This has been the main task in the internship.
During the internship several tasks were assigned, leading to a decreased amount of time for the main objective. The tasks have varied in size and subject, but has lead to a development of a new set of skills. Playing a part in a company has also given an insight and experience in working in a company and participating in product development and work planning.
The main obejctive changed to real time object detection of groceries, which was implmented training You Only Look Once (YOLO) on a 60 classes grocery dataset. The network achieved an accuracy of 48.17%, which should be increased to prove viable.

The content of this report is freely available, but publication may only be pursued with reference.

Preface

This report is composed by Niclas Stjernholm during the third semester of the master's programme in Vision, Graphics and Interactive Systems at Aalborg University. The third semester is used as a company internship.

For citation the report employs the Harvard method. If citation is not present in tables or figures, they are produced by the author.

This project is implemented in Python 3.6.

Aalborg University, January 8, 2019

Niclas Hjorth Stjernholm
<nstjer14@student.aau.dk>

Contents

Glossary

CNN	Convolutional Neural Network.
D2S	MVTec D2S: Densely Segmented Supermarket Dataset.
DOF	Degrees of Freedom.
FC	Fully Connected.
FPS	Frames per Second.
GQ-CNN	Grasp Quality Convolutional Neural Network.
GStAPP	GStreamer application.
MOT	Multiple Object Tracking benchmark.
RGB-D	Red, Green, Blue, Depth.
SVM	support vector machines.
YOLO	You Only Look Once.

Chapter 1

Introduction

Several warehouses around the world are starting to use robots for picking groceries of different sorts [Olsen, 2018; Perez, 2018; Vincent, 2018]. This is done to automate the process of grocery picking and make the entire process faster, to be able to deliver packages faster to the consumer. Olsen [2018]; Perez [2018]; Vincent [2018] present robots picking from boxes and not free standing products like in a general supermarket.

The goal of Aivero is to enable robots in a general supermarket set-up, to pick groceries straight from the shelves with only the information of which section of a shelf the desired grocery is. The aim is to do this using a video stream directly from the robot and process this data in real time. Using the video stream a processing unit must identify the desired grocery and find the optimal picking point on the item. For a potential higher accuracy the goal is to use both a regular colour video stream, RGB, and depth video. This will enable the robot to see the groceries in physical shape better from the depth video, but also use th RGB video stream to recognise e.g. labels on a product.

Chapter 2

Aivero AS

Aivero is small company with nine employees including the intern. The aim of the company is to develop skills for robotics via 3D vision and AI. The first implementation done towards the goal was to be able to stream and store Red, Green, Blue, Depth (RGB-D) video data. For the storing to be viable, because of the amount of space RGB-D video can take up, a compression algorithm is needed, which has been developed and works on PC and is implementable on an Nvidia TX2.

The company has offices in Stavanger, Norway and Aalborg, Denmark. Three people in Norway, where the CEO, board director and marketing are located. Four people in Aalborg, with the CTO, two student developers and myself doing full time as an intern. Two other developers are part of the team. One software developer in Aarhus, Denmark, and another part time developer located in Delhi, India. The company was started in December 2017. My responsibility area is AI development and planning, as the only one in the company with a bit more extensive knowledge in deep learning and computer vision.

Chapter 3

Background Research

As a part of the implementation of a viable grasping point of objects in an environmental setting a research of state of the solutions of both object tracking and detection is necessary as well as researching full existing solutions.

3.1 Object Detection and Tracking

In order to get some understanding of the state of the art technologies used the research is split into multiple sections for different areas in the process of teaching a robot to pick groceries from a shelf.

3.1.1 Object Detection

The aim of generic object detection is to locate and classify an object in an image, sometimes including a bounding box on objects with a confidence of existence in the image. In general there are two approaches to object detection namely region proposal, which follows the traditional object detection pipeline and then classifying each proposal into different object categories. The other approach is the regression or classification based approach by adopting a unified framework to achieve final results [Zhao et al., 2018]. Zhao et al. [2018] presents several solutions of both approaches and have also made a small roadmap of different popular solutions up until 2017. This is shown in Figure 3.1.

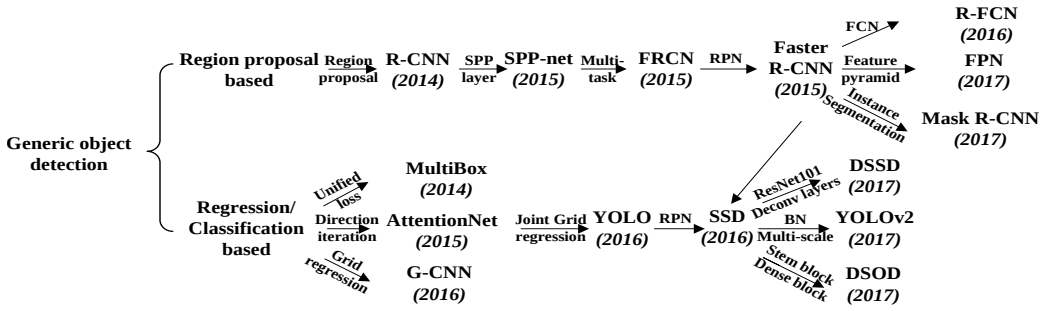


Figure 3.1: Small roadmap of different popular solutions up until 2017 [Zhao et al., 2018]

Another way of doing object detection is using decision trees instead of deep learning as proposed by Gall et al. [2012].

Region Proposal

The region proposal framework is a two step process. The framework firstly scans an image and then focuses on any regions in the image of interest. As shown in Figure 3.1 the R-CNN solution is one of the first in the region proposal approach. The design has three stages in the process of object detection. Firstly a region proposal generation, generating 2000 region proposals for each image. Afterwards a Convolutional Neural Network (CNN) is applied to extract features from warped or cropped regions, extracting a 4096-dimensional feature vector. From there classification and categorisation is done with pre-trained support vector machines (SVM) from multiple classes. The final bounding boxes are produced from adjusted scored regions using bounding box regression.

Regression or Classification framework

To compete with the time consumption of regional proposal the one-step framework based on global regression/classification is utilised by mapping straight from image pixels to bounding box coordinates and class probabilities. This is the technique the framework You Only Look Once (YOLO) applies. YOLO divides an input image into an $S \times S$ grid, each cell is responsible for predicting the object centred in the cell. The first YOLO framework has 24 convolutional layers and 2 Fully Connected (FC) layers [Zhao et al., 2018]. The second version, referred to as both YOLOv2 and YOLO9000, uses the Darknet-19 model, which has 19 convolutional layers and 5 maxpooling layers. It is based on the Googlenet architecture [Redmon and Farhadi, 2016]. At the point of deployment this framework has state of the art performance. The third iteration of the framework is YOLOv3 with an update of the Darknet architecture increasing the size from 19 convolutional layers to 53. The newest YOLO framework is from early 2018 and is still one of the best in its category. Redmon and Farhadi [2018] states the framework does not perform as well as RetinaNet in precision, but the speed of the framework is faster. YOLO is fast and lightweight

enough to run in real time. It is mostly trained on the COCO and VOC datasets [Redmon and Farhadi, 2018].

Decision Trees

A decision tree consists of split nodes and leaves. The split nodes evaluate each image patch based on a predefined decision metric and pass the patch to either left or right in the tree. The leaf is the end of a tree and stores the statistics of the image patches which arrived during training of the tree. This is illustrated in Figure 3.2.

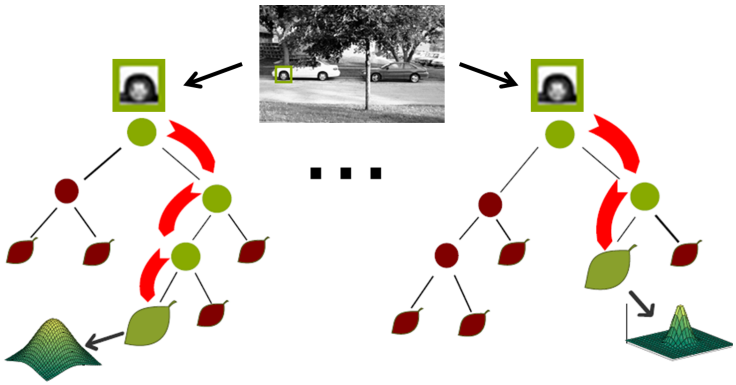


Figure 3.2: Example of decision tree evaluation on image patch from bounding box [Gall et al., 2012]

Gall et al. [2012] uses a random forest for multi class object detection. A random forest consist of a set amount of decision trees. The detection problem becomes a distribution estimation problem, where the random forests allow to learn features and descriptors that are optimal for estimating the distributions with low uncertainty. These distributions may be desirable to restrict to Gaussians or Gaussian Mixture Models [Gall et al., 2012].

The amount of training data is the most crucial parameter for detection accuracy, as random forests are designed to handle large amounts of data. If it is small training sets extra verification steps may be needed [Gall et al., 2012].

3.1.2 Tracking

General object tracking is tracking an object in a video or a sequence of images, in form of e.g. the problem of estimating the trajectory of an object in the image plane as the object moves in the image sequence. This means, that the tracker consistently labels the tracked objects in the different frames of a video [Acm Reference Format: Yilmaz et al., 2006]. Situations often present tracking of more than one object at the time where a multi object tracking framework is needed. Leal-Taixé et al. [2017] has evaluated on 32 different state of the art trackers from no sooner than March 2017 on multiple objects. The evaluation is performed on the Multiple Object Tracking

benchmark (MOT)15 and MOT16 datasets. This is done to introduce a standardised benchmark and analyse the performance of the trackers [Leal-Taixé et al., 2017]. Leal-Taixé et al. [2017] states there are six top-performing trackers with a multiple object tracking accuracy higher than 40%. Of the top six trackers, two of them are using deep learning, one using Recurrent neural networks to encode appearance of pedestrians and the other is using deep matching. The common component of the top performers are strong affinity models [Leal-Taixé et al., 2017].

Three Dimensional Object Tracking

In order to utilise three dimensional video, another dimension needs to be added by introducing depth in the data. This is done by using a depth camera to record and colouring the video in accordance to depth in either greyscale or a heatmap.

Tan and Ilic [2014] propose a solution for both two dimensional template tracking an three dimensional object tracking using multi forest decision trees. This is based on another objective function which relates the image intensities of a template and transformation parameters by the pseudo-inverse of a Jacobian matrix, but with the use of random forests instead of the Jacobian, making the method generalised to any input function and not constrained to two dimensional intensity images [Tan and Ilic, 2014]. The random forests are regression forests to learn how different values of the transformation function affect the input function. The training of forests begins by creating a training dataset with n_ω random values to transform the input to affect the computation of the sample points. They randomly select n_r points from n_s sample points of the template for constructing a tree to impose randomness. Tan and Ilic [2014] uses 100 trees for each 6 parameters with a grid enclosed on the model seen from the depth image, only the points which lie on the model are used as sample points. In training 50 000 depth images are rendered with different kinds of distortions, such as rotation and translation. This is done for each camera view, of which there are 42.

Object Pose

For a robot to be able to interact with an object, more information than the position in an image is needed. The orientation of the object is also important. Xiang et al. [2017] seeks to estimate the six degrees object pose with the network, PoseCNN. PoseCNN, is trained using RGB-D scans of 21 objects and evaluated on both the YCB dataset and the OccludedLINEMOD dataset, to evaluate the robustness towards occlusions. The network has three different tasks done in order to estimate a six dimensional pose. Firstly a semantic labelling, secondly a 3D translation estimation, and thirdly a 3D rotation regression [Xiang et al., 2017]. This is illustrated in Figure 3.3.

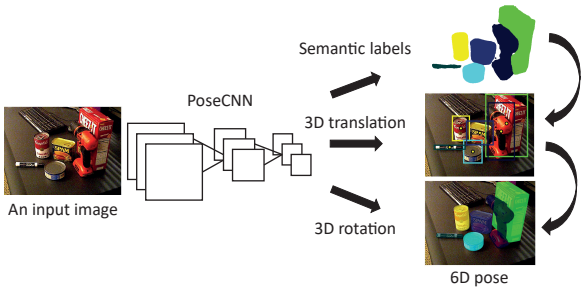


Figure 3.3: Overview of the three steps performed in by the PoseCNN [Xiang et al., 2017]

3.1.3 Grasping Planning

For a robot to interact with an object, a grasping point on the tracked object is necessary. Mahler et al. [2017a] proposes a solution of grasping planning in two different solutions based on the same deep learning model called Grasp Quality Convolutional Neural Network (GQ-CNN). They have created their own dataset with 1 500 3D object models for training the network. For the second iteration Dex-Net 2.0, Mahler et al. [2017a] analyse a dataset of 6.7 million point clouds, with grasping options, which has been generated from 3D models in randomised poses on a table, for training. For the third iteration, Dex-Net 3.0 is generated. This dataset is 2.8 million point clouds with suction grasps, and grasp robustness labels made from 1 500 3D objects models. The datasets are used to train the GQ-CNN to classify suction grasp robustness. This is done with a model of the two different grippers used with the robot when testing [Mahler et al., 2017b].

Dex-Net 2.0 is tested on 3D printed objects designed to challenge the parallel gripper used in this iteration and 10 household objects [Mahler et al., 2017a]. Dex-Net 3.0 is tested on objects divided into three different categories of basic, typical, and adversarial. Both the object categories and the performance in the different groups are shown in Figure 3.4 fro Dex-Net 3.0.

	Basic		Typical		Adversarial		
	AP (%)	Success Rate (%)	AP (%)	Success Rate (%)	AP (%)	Success Rate (	
Planarity	81	74	69	67	48	47	
Centroid	89	92	80	78	47	38	%)
Planarity-Centroid	98	94	94	**86**	64	62	
GQ-CNN (ADV)	83	77	75	67	**86**	**81**	
GQ-CNN (DN3)	**99**	**98**	**97**	82	61	58	

Figure 3.4: Dex-Net 3.0 object categories and performance. Basic (e.g. prismatic objects), Typical, and Adversarial [Mahler et al., 2017b]

Chapter 4

Project Specification

This chapter specifies the scope of the project. It outlines and delimits the goals for the work conducted, as well as setting the requirements for the solutions implemented during the project work.

4.1 Problem Statement

The two objectives, first one set as an open objective the second closed, given by the company were originally:

- Improve depth quality of commodity RGB-D cameras
- 6 Degrees of Freedom (DOF) object pose tracking of known objects using 3D cameras (for robotic picking of groceries)

The primary objective was the second and closed objective, to implement a 6 DOF object pose tracking. As a part of smaller company other tasks also needed work, which took time from the amount of time for the primary objective.

4.2 Secondary Objectives

Other tasks were:

- Train a You Only Look Once (YOLO)v3 network for object detection in a Tensorflow implementation
- Create small test setup for matrix storage in C++
- GStreamer command testing and configuration
- Setting up a docker for testing

4.3 Task specification

Specification of the tasks at hand is necessary to be able to understand what is needed to complete the assignments given.

4.3.1 Main Task

The main task, performing 6 DOF object pose tracking, is to develop skills for a robot to be able to work in a regular supermarket setting picking from shelves set up as in a supermarket. This requires a vision application able to do object recognition, pose estimation, and be able to determine the most viable grasping point on the desired object. This is to be done with a 3D video stream, combining RGB and depth video.

The dataset will in the end depend on the application, as it must be of the exact groceries offered in the supermarket at hand, and should be created by the developer. But as a proof of concept, a fulfilling dataset online can be used.

Working on the other tasks meant the main task got scaled down due to time constraints. Instead, the first steps of the object pose tracking became essential for showing potential in the idea. This mainly focused on finding an already working solution of object detection and grasping point estimation and training a real time object detection solution for groceries.

4.3.2 Secondary Tasks

The secondary tasks are smaller tasks assigned by the company. As the company is of smaller size, help is often needed on other tasks to finish a sprint, or meet a deadline where all help is needed.

YOLOv3 Tensorflow Implementation

As part of participation in Robot Union, a robotics acceleration program, the company asked for a small real time object detection solution with YOLO working on Tensorflow. The solution is supposed to work on an industrial objects database containing screws or fuses and extension boxes and objects similar to these.

The task had a strict deadline as the idea was to show the object detection at the first Robot Union presentation, which was rather close to the time of requesting the implementation.

Matrix Storage

The description of the task reads: *As the developer I want to automate the testing of the basic performance of the camera compression algorithm. This shall be repeatable across multiple platforms.*

This is the description of the entire task and the description was added after the task was given, as the task evolved to include more work than first described. The task given, was to be able to store raw video data as matrices in a txt or csv file.

GStreamer Pipeline Configuration

As a part of the compression software being sold with an Nvidia TX2, a streaming solution for both storage and preview of the video data is necessary. This is done with GStreamer. The task given, was to store the received RTP packages as compressed video data, be it RGB or depth video. The video needed to be played back at, at least 30 Frames per Second (FPS).

The task also included the option to play the video directly from the stream, which meant decompression and then playback, also at 30 FPS. Lastly the implementation also needed the option to open the stored compressed file and displaying it to the user.

Docker Set Up

The task was to implement the GStreamer pipeline from the previous task in a docker container. This should work on a barebone ROS Balena image, as ROS was used connect the different modules using ROS nodes.

As Docker was new territory this task also included learning to set up a docker service, which meant, the task took longer than for an experienced Docker programmer.

Chapter 5

Implementation

Based on chapter 4, Project Specification, a number of tasks has been set. This chapter clarifies the work done and solutions found to complete the tasks.

5.1 Object Detection and Tracking

As stated in section 4.3.1 the main task got scaled down due to the amount of other work needed done. Instead of being able to do 6 Degrees of Freedom (DOF) object pose tracking of known objects using 3D cameras (for robotic picking of groceries), the first steps of the object pose tracking became essential for showing potential in the idea.

Instead finding an already working solution of object detection and grasping point estimation and showing a real time object detection solution for groceries became the goals.

5.1.1 Grasping Point Estimation

Mahler et al. [2017b] has developed a grasping planning network called Dex-Net. This project was chosen as a viable solution to object grasp planning. This is done over several generations and has two different grasping methods. Both a parallel gripper and suction based end-effectors.

Both implementations of Dex-Net 2.0 and 3.0 are trained using a synthetic dataset of point clods which include grasps and some metrics for grasping quality, either for a parallel gripper or a pneumatic suction cup. This has been modelled for the end-effector in each generation.

The network trained is the Grasp Quality Convolutional Neural Network (GQ-CNN) also designed by Mahler et al. [2017a]. The network is able to estimate the highest quality grasping point for both end-effectors. The Dex-Net 2.0 is 93% successful on objects in training and achieves 99% on a dataset of 40 household objects [Mahler et al., 2017a]. The Dex-Net 3.0 has three different categories of objects; Basic (prismatic or cylindrical), Typical (more complex geometry), and Adversarial

(with few available suction-grasp points) and achieves success rates of 98%, 82%, and 58% respectively. When training with the adversarial objects only, they reach a success rate of 81%. These objects are shown in Figure 3.4 with the performances of on the different objects.

5.1.2 Real Time Object Detection

The solution needs to be real time, to show the potential of the implementation, for that reason You Only Look Once (YOLO) is chosen as the model to implement, since it is able to run real time. Opposite to section 5.2 there is no requirement to use Tensorflow, instead a GitHub repository of the YOLOv3 network is used. YOLO uses C and CUDA for training neural networks [Redmon and Farhadi, 2018]. With CUDA the program is able to utilise an Nvidia GPU to drastically lower the training opposed to using just the CPU.

You Only Look Once

The implementation of YOLOv3 used is a forked repository of the Darknet created by Joseph Redmon but with an extended Readme easing insight into training other datasets than the ones included in the repository. Darknet originally uses another annotation than both the COCO and VOC dataset, by using a `txt` file for each image in the chosen dataset. An annotation file consists of a line for each new object in the image, with four floats afterwards between 0 and 1 relative to width and height of image in the order: `<x_center> <y_center> <width> <height>`.

Database

The database used for this is "MVTec D2S: Densely Segmented Supermarket Dataset (D2S)" by Follmann et al. [2018]. This dataset contains 60 different groceries with pixel-wise labels of all object instances. Objects are placed in a setting to resemble a real world setting of an automatic checkout. Each object is featured in several positions and three different lightings of each angle. The setup, lightings and different backgrounds are shown in Figure 5.1. The different backgrounds are only used for testing.

Figure 5.1: D2S image acquisition setup and different lighting settings and the different backgrounds used

Training

Before training, the data and network cfg need preparation to fit the new data properly. This is due to the network design towards another dataset and annotations for the dataset are not in the right format.

The annotations from the D2S dataset are written in a json file and needs conversion. This conversion is done using a python script to analyse the json file and write the new files. The script is shown in section B.1 in the appendix.

Besides the annotations, the network needs a file path to every training and test image from the root of the executable program.

In the cfg for the network it is important to change the amount of classes to the amount in the custom dataset i.e. 60. When changing the amount of classes, recalculating the anchors can boost performance. A script for this is included in the repository and only needs the image paths file. The filters in the cfg also need changing as these are set based on the amount of classes in the dataset, these should be: filters = $(60_{\text{classes}} + 5) \cdot 3 = \underline{195}$.

To increase accuracy the network is trained with pre-trained weights from the Darknet website for YOLOv3. The network is trained with a batch size of 64 and a learning rate at 10^{-3} for 83 000 iterations.

5.1.3 Results

After 83 000 iterations the network started to be unstable and was stopped at an average loss of 0.7171. The loss curve for the training is shown in Figure 5.2.

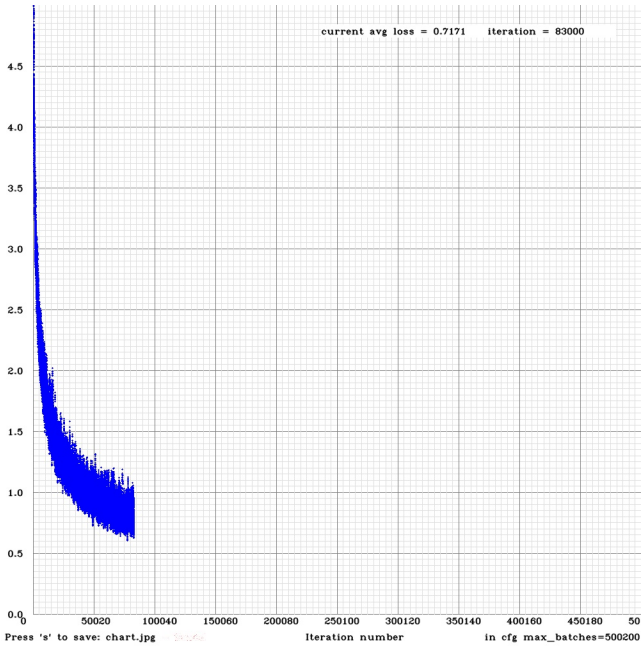
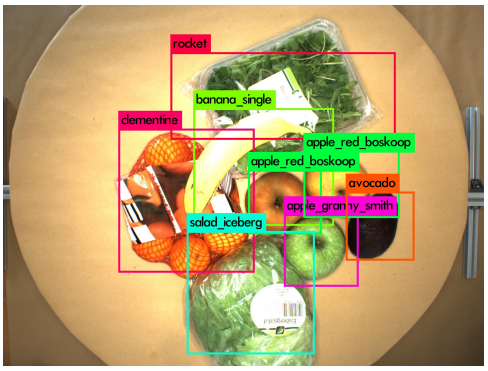


Figure 5.2: Loss curve for training Darknet for the D2S database

The trained network is then tested on both single testing images to see if it is able to detect objects in those. An example of this is shown in Figure 5.3.

(a) Two objects from the D2S dataset recognised by the network



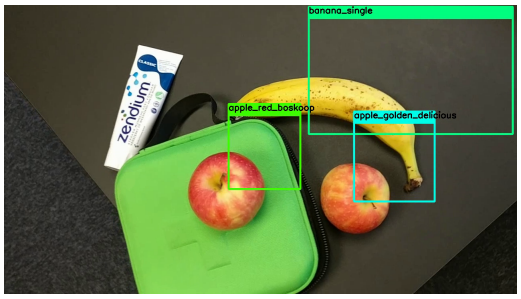
(b) Multiple objects from the D2S dataset recognised by the network

Figure 5.3: Single tested images from the D2S dataset

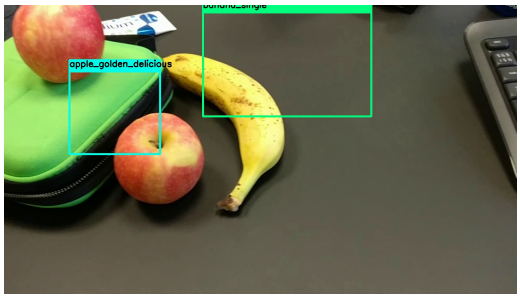
When testing on a home recorded video of different objects the network is having issues recognising objects in the scene. The recorded scene consists of two apples and a banana, which are the only objects resembling objects from the database. It does somewhat recognise the apples and the banana at points, but the certainty varies. It also has some false positives during the test video. While analysing the video, the

certainties are output in the terminal running the detection, a dump of some of this output is found in an attached zip file with the report with the name `d2s_dump.txt`.

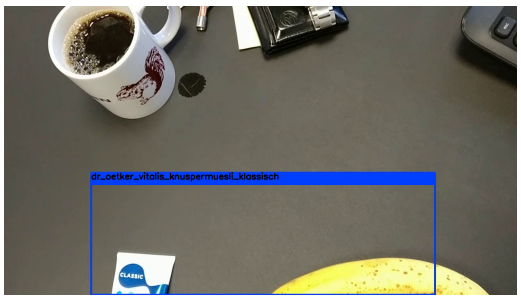
Screenshots of the video with both false positives and true positives are shown in Figure 5.4. Due to an offset of the bounding boxes, the corner of a box is marking the centre of the object. This issue only occurs in the video test and not in the single images. As the terminal output is limited to a certain amount of lines, not all frames are included in the certainty dump file.



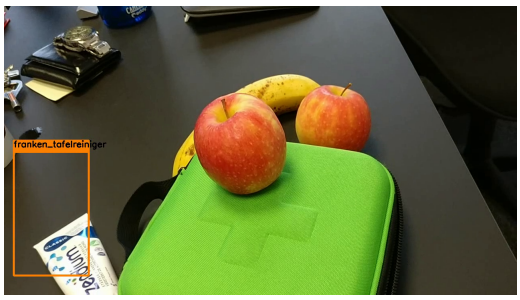
(a) True positives with certainties at:
banana_single: 80%
apple_golden_delicious: 28%
apple_red_boskoop: 37%



(b) True positives with certainties:
banana_single: 47%
apple_golden_delicious: 85%



(c) False positive with certainty:
dr_oetker_vitalis_knuspermuesli_klassisch:
32%



(d) False positive with certainty:
franken_tafelreiniger: 55%

Figure 5.4: Screenshots of the video with both false positives and true positives

An overview of the precision on each object class is shown in Appendix C. This also shown the mean average precision is 48.17%.

The test video is also found in the attached zip file with the report, under the name `darknet_test_video.avi`.

5.2 YOLOv3 Tensorflow Implementation

As the solution is supposed to work real time, YOLO was chosen as the desired solution, but also because the CTO had had YOLO running before using the standard setup of the Darknet running in C trained on the COCO dataset.

Because of the desire to have it running on Tensorflow another version of Darknet is needed, as Tensorflow works with Python and not C. Given previous experience with Keras, a solution build upon this is preferred as this will ease the understanding of the implementation and make potential changes to the network easier to do.

5.2.1 Keras YOLO3

The Keras implementation is found on GitHub and is made for YOLOv3. It is based on another repository made with the YOLOv2 Darknet version, which is a smaller network, but also with a general lower precision [Redmon and Farhadi, 2016, 2018]. The repository is found at: https://github.com/qqwweee/keras-yolo3.

This is a conversion of the network design to a Keras implementation. It employs the annotation layout of one row for each image in a text file with every bounding box in the image separated with spaces:

Row format: `image_file_path box1 box2 ... boxN`, and
Box format: `x_min,y_min,x_max,y_max,class_id`.

5.2.2 Dataset

It was requested, that the object detection was trained to recognise some industrial objects such as screws, electric fuses for a fuse box and extension boxes. The dataset chosen is "T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects" made by Hodaň et al. [2017].

As stated in the title of the article for the dataset, the dataset is a an Red, Green, Blue, Depth (RGB-D) dataset. As YOLO only works with one type of images only the RGB images are used.

5.2.3 Training

The training is done by running a Python script with Tensorflow activated, to enable GPU processing, which launches the training. Before training, the data needs to be prepared, as the annotation of the dataset differs from the on the network uses.

Data Preparation

As the annotations from the T-Less dataset is written in an XML file format with a file for each image of each object, a conversion is necessary. This is done with a small Python script converting to the annotation style mentioned in section 5.2.1. The script is shown in section B.2 in the appendix.

Training Setup

The setup of the network is including a `ReduceLROnPlateau` function, which reduces the learning rate when a metric has stopped improving. Another callback function

included is `EarlyStopping` which stops the training when a certain set improvement is not met for a specified amount of iterations.

YOLOv3 uses the Darknet-53 Convolutional Neural Network (CNN) which has 53 convolutional layers. This is also shown in Figure 5.5. The last Fully Connected (FC) layer has been changed from 1000 classes to 30, as this is the amount of classes included in the T-Less dataset. The training is done with pre-trained weights from the Darknet training with the COCO dataset.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		30	
	Softmax			

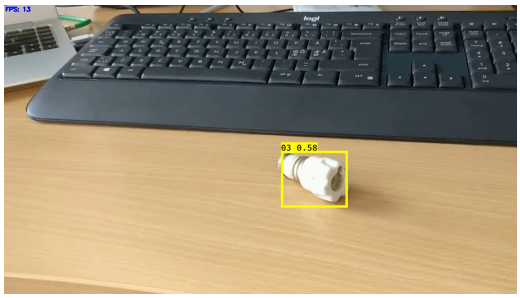
Figure 5.5: Overview of Darknet-53, the CNN used with YOLOv3 [Redmon and Farhadi, 2018]

For training, the optimiser Adam is used, first with a learning rate of 10^{-3} if more training is needed after 50 epochs and the learning rate has not been lowered, it is lowered to 10^{-4}, and with a batch size of 32.

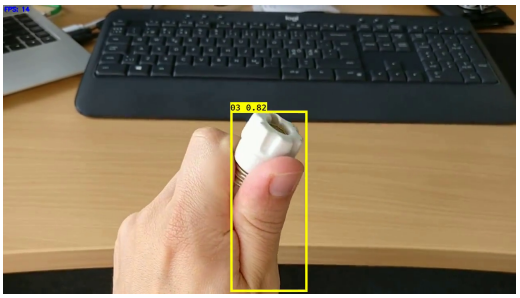
5.2.4 Results

Due to a limited amount of time and objects resembling the objects in the dataset, the testing video used for this network only included one item from the T-Less dataset, an electric fuse. The fuse is filmed in an office setting with a keyboard, laptop, monitor visible in the video as well. This object was chosen as it was easy to find in many situations and places, which meant it might also be possible to find one at the Robot Union event, which the demo was for.

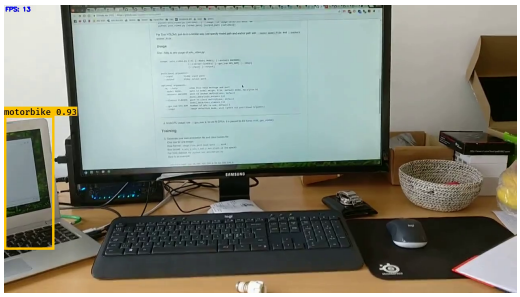
Unfortunately, the precision was not high enough to show a demo at the event. As shown in Figure 5.6, there was a lot of false positives in the testing and low accuracy when detecting the desired object. In the video, the object *03* is the fuse.



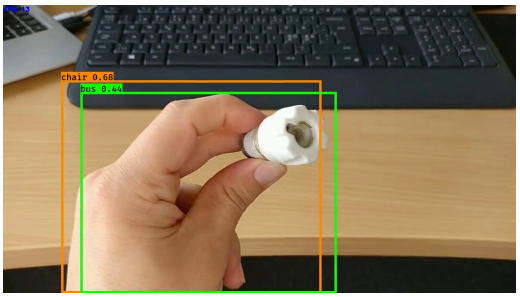
(a) True positive, fuse detected with 58% certainty



(b) True positive with occlusion, fuse detected with 82% certainty



(c) False positive detection of a motorbike



(d) False positive detection of chair and bus

Figure 5.6: Video test results for the YOLOv3 network trained on the T-Less dataset

5.3 Matrix Storage

This task was intended as an on stream process, storing the video data before compressing as matrices to a file, to automate testing of the camera compression algorithm. At the point of development, the core solution of the pipeline was written in C++ and not the final GStreamer solution implemented for the video streaming solution.

This task required some insight into the meta solution of the pipeline already written and understanding of the code. The code had several dependencies and required some extra functionality in order to generate the data as wanted.

Before reaching a solution to the task, another solution for the pipeline was made, and therefore made the task obsolete.

5.4 GStreamer Pipeline Configuration

The pipeline configuration consists of several smaller tasks within the same GStreamer application (GStAPP). In total it has three functions:

- Receive RTP packages and store them in the desired destination
- Receive RTP packages, decompress, and play them back to the user

- Decompress the video data and play it for the user

The source for the RTP packages is a UDP source, which means an IP and a port needs to be set to receive the packages. When testing, the IP is localhost i.e. `127.0.0.1`, and the port depends on which video stream the user wants to receive, as there are both an id stream, a depth video stream and a colour video stream. These are either `9000`, `9001`, or `9002`. It is important to set the same ports on the sender side in order to receive data.

The data received is payloaded and has h265 encoding which needs de-payloading and decoding before storing to file or playing back.

When differing between storing and streaming to a video player, it is only one command in difference. It is either `filesink` and a path to destination for storing or `fpsdisplaysink` to play back the video.

Besides these settings, it is important to match *caps* which are video settings set from the sender side, and must also be et on the receiver side. These depend on the file type. An example of a GStreamer command to receive and store a video file is:

```
gst-launch-1.0 videotestsrc num-buffers=100 ! x265enc tune=zerolatency ! video/x-h265,
stream-format=byte-stream,alignment=au ! filesink location=h265.pay
```

5.5 Docker Set Up

The docker set up was to be able to test the GStAPP described in section 5.4, by having it running in a docker container on an Nvidia TX2.

Due to Docker being a new area, learning of Docker composing was necessary, which was also a goal in the task more than getting the test scenario running. Training and learning was done using the website https://training.play-with-docker.com/.

The set up was to be build upon a barebone ROS Balena image, which had been build before. This meant the docker still needed some Catkin set up and enabling for building with Meson and Conan, and then calling the build command in the docker, and the GStAPP would be ready to launch.

With a sender in one docker and the receiver in another the testing of the communication is done. The majority of the time spent on the task was learning how Docker works and not the design of the docker file written.

Chapter 6

Evaluation

This chapter is split in two evaluations. One is of the results shown in chapter 5, the other is the internship and working in the company.

6.1 Result Evaluation

There are two different object detection results presented in chapter 5, both are made using YOLO but with two different implementations and datasets, and will be evaluated separately.

6.1.1 Real Time Object Detection

The results show, an average loss of 0.7171 which can prove to be too high, as seen in the resulting video test screenshots. Although the results show the ability to do object detection on images from the dataset used for training.

With a mean average precision of 48.17% the network needs more training even though the loss curve is flattening. This can be done both by doing more iterations or by adding more data. Often, by adding more data for the network to train on, accuracy can be increased. This can be done using data augmentation e.g. rotating, flipping, occluding, and cutting each image. by doing this the data can be increased ten fold or more. Another reason to add more data and not train longer is the amount of instability observed at 83 000 iterations, which could lead to worse results.

6.1.2 YOLOv3 Tensorflow Implementation

When being close enough to the desired object, detection is successful, but with a somewhat low certainty. This can show some pitfalls towards smaller objects in the trained network. The false positives show issues with training, but they also show issues with labels. The network is trained with the YOLO weights trained on the COCO dataset, which might have introduced the label issues with the way the weights are used in this implementation of YOLO.

Another issue can, once again, be the amount of data available and could be increased by doing data augmentation.

6.2 Internship Evaluation

Working in Aivero I was the only one on the team working with computer vision and deep learning. Other employees had some experience in the area, but had other areas to mainly focus on. This meant that I sometimes was not sure on what approach to choose and could not always seek help from a more experienced developer. But sparring with co-workers often lead to ideas on solutions.

As a part of the team, I also participated in daily SCRUM stand up meetings to summarise the work done and the planned work to be done. Once a week a company wide stand up was held to get insight into what everyone were doing.

From September through December, we were mainly two people in the office. With a total of four people working in the office with two being students, only working part time. Being this few helps build a more friendly relationship to each other, which can make it easier to ask for help or solve potential issues.

My role in the company was mainly to develop the 'robotic skills' which was the computer vision of doing object pose estimation and tracking. Besides this I helped with smaller tasks when needed. Due to a deadline in a product sale, all hands were needed for that project to deliver on time.

In November, from the 18th to the 20th, four of the five Danish employees, including myself, went to Stavanger, Norway, to meet the Norwegian part of the company. This was both to do some team building and getting to know each other, but also planning a roadmap for the company in the nearest future. This was a really nice experience, which definitely made it feel like being a part of a team.

The internship has shown me how it is to be a part of a company and how a deadline can influence your own work. It has helped building a larger skill set and introduced me to contacts, other students, and given an idea of what to do when graduating.

Chapter 7

Conclusion

During the internship the main task has been to develop a system with;

6 Degrees of Freedom (DOF) object pose tracking of known objects using 3D cameras (for robotic picking of groceries).

Due to time constraints this main task was decreased to show the possibility of implementing real time object detection with a grocery dataset and showing a solution of grasping estimation on objects with deep learning.

The Dex-Net implementations show that it is possible to estimate grasping points of objects when the right model is applied to a big dataset, even with synthetic data. The robot is able to evaluate the Red, Green, Blue, Depth (RGB-D) input data after training and can estimate a successful grasping point with high precision.

The real time object detection is made using You Only Look Once (YOLO)v3, as this network is known for being a fast, lightweight network. The dataset used is a grocery dataset with 60 different object classes including annotations. The mean average precision achieved with the network is 48.17% on the test images. This precision has proven too low to handle the test video, showing a lot of false positives and in general low precision on each object class, as seen in Appendix C. A proposed solution to increase precision, is to add more data by augmenting the training images with several different methods.

Being in a company meant several tasks instead of just one project. The other tasks lead to learning new skills in programming and IT knowledge. The tasks was assigned due to a deadline, which meant they had highest priority, taking away time from the main project.

Due to being the sole employee working with computer vision, discussing solutions was not always possible because of the lack of insight in the subject.

The internship has shown how it is being a part of a company and has given some experience in working with people outside of the university programme. It has also been testing the various skills learned and made for improvement in several subjects. It has furthermore given inspiration to future plans after the university.

Bibliography

Acm Reference Format: Yilmaz, A., Javed, O., and Shah, M., 2006. Object tracking: A survey. *ACM Comput. Surv*, 38, p. 45. doi: 10.1145/1177352.1177355. Available at: <http://doi.acm.org/10.1145/1177352.1177355>.

Follmann, P., Böttger, T., Härtinger, P., König, R., and Ulrich, M., 2018. MVTec D2S: Densely Segmented Supermarket Dataset. *European Conference on Computer Vision (ECCV)*, pp. 569–585. Available at: <www.mvtec.com>.

Gall, J., Razavi, N., and Van Gool, L. An introduction to random forests for multi-class object detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7474 LNCS, pp. 243–263, 2012. ISBN 9783642340901. doi: 10.1007/978-3-642-34091-8_11. Available at: <http://link.springer.com/10.1007/978-3-642-34091-8{_}11>.

Hodaň, T., Haluza, P., Obdrzalek, Š., Matas, J., Lourakis, M., and Zabulis, X. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, pp. 880–888, 2017. ISBN 9781509048229. doi: 10.1109/WACV.2017.103. Available at: <http://cmp.felk.cvut.cz/{~}hodanto2/data/hodan2017tless.pdf>.

Leal-Taixé, L., Milan, A., Schindler, K., Cremers, D., Reid, I., and Roth, S., 2017. Tracking the Trackers: An Analysis of the State of the Art in Multiple Object Tracking. apr. Available at: <https://arxiv.org/abs/1704.02781>.

Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Ojea, J. A., and Goldberg, K., 2017. Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics. mar. ISSN 0929-5593. doi: 10.15607/RSS.2017.XIII.058. Available at: <http://arxiv.org/abs/1703.09312>.

Mahler, J., Matl, M., Liu, X., Li, A., Gealy, D., and Goldberg, K., 2017. Dex-Net 3.0: Computing Robust Robot Vacuum Suction Grasp Targets in Point Clouds using a New Analytic Model and Deep Learning. sep. doi: 10.1109/ICRA.2018.8460887. Available at: <http://arxiv.org/abs/1709.06670>.

Olsen, L., nov 2018. *Fuldautomatisk robotlager: Nu tager det syv minutter at pakke en pakke*, Horsens. Available at: <https://hsfo.dk/erhverv/Fuldautomatisk-robotlager-Nu-tager-det-syv-minutter-at-pakke-en-pakke/artikel/199254>.

Perez, S., 2018. *Walmart pilots a grocery-picking robot to fulfill customers' online orders*. Available at: <https://techcrunch.com/2018/08/03/walmart-pilots-a-grocery-picking-robot-to-fulfill-customers-online-orders/?guccounter=1>.

Redmon, J. and Farhadi, A., 2016. YOLO9000: Better, Faster, Stronger. dec. Available at: <http://arxiv.org/abs/1612.08242>.

Redmon, J. and Farhadi, A., 2018. YOLOv3: An Incremental Improvement. apr. Available at: <http://arxiv.org/abs/1804.02767>.

Tan, D. J. and Ilic, S. Multi-forest tracker: A Chameleon in tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1202–1209. IEEE, jun 2014. ISBN 9781479951178. doi: 10.1109/CVPR.2014.157. Available at: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909553>.

Vincent, J., 2018. *WELCOME TO THE AUTOMATED WAREHOUSE OF THE FUTURE*. Available at: <https://www.theverge.com/2018/5/8/17331250/automated-warehouses-jobs-ocado-andover-amazon>.

Xiang, Y., Schmidt, T., Narayanan, V., and Fox, D., 2017. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. nov. Available at: <https://arxiv.org/abs/1711.00199>.

Zhao, Z.-Q., Zheng, P., Xu, S.-T., and Wu, X., 2018. Object Detection with Deep Learning: A Review. Available at: <https://arxiv.org/pdf/1807.05511.pdf>.

Appendix A

Internship Journal

The journal starts on the next page, due to pdf pages size.

Issue / Worklog	Worked
PP-54 - Initial Planning and Setup	**30.00**
2018-08-22 - Architecture walkthorugh with Raphael (Niclas Stjernholm)	2.75
2018-08-30 - Sprint planning for Robot Union event. (Niclas Stjernholm)	1.50
2018-08-30 - Setting up in the new office. (Niclas Stjernholm)	1.75
2018-08-30 - Setting up new PC, getting essentials installed and up and running. (Niclas Stjernholm)	2.00
2018-08-31 - Got tensorflow working on new pc (Niclas Stjernholm)	2.00
2018-09-17 - Kickoff Planning Gstreamer Development (Niclas Stjernholm)	1.00
2018-09-26 - Setup of the new office room moving everything into the new room (Niclas Stjernholm)	2.00
2018-10-04 - Planning next steps in object recognition development. (Niclas Stjernholm)	1.00
2018-11-19 - Visiting Norway office meeting robotNorge and such. (Niclas Stjernholm)	8.00
2018-11-20 - Visiting Norway office meeting robotNorge and such. (Niclas Stjernholm)	8.00
PP-205 - Investigate and learn about Azure services	**2.50**
2018-10-29 - Checking pricing on the azure services and estimating which services are needed for machine learning. (Niclas Stjernholm)	1.75
2018-10-30 - Checking pricing on the azure services and estimating which services are needed for machine learning. And checking pricing on AKS (Niclas Stjernholm)	0.75
PP-257 - Deprecated - Detection of known objects using RGB and / or D	**114.00**
2018-10-04 - Adding new tasks to jira. (Niclas Stjernholm)	0.25
2018-10-05 - Checking up on Nvidia Deepstream. Also sat up the Gitlab runner on pc (Niclas Stjernholm)	1.50
2018-10-05 - Checking up on Nvidia Deepstream (Niclas Stjernholm)	3.25
2018-12-03 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	4.00
2018-12-03 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	3.75
2018-12-04 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	4.00
2018-12-04 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	3.75
2018-12-05 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	4.00
2018-12-05 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	4.00
2018-12-06 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	4.00
2018-12-06 - Getting yolo running the the MVTech Supermarket data set. Yolo is a bit tricky with its labels. A lot of preparation is needed. More than expected. (Niclas Stjernholm)	3.75
2018-12-07 - Report writing (Niclas Stjernholm)	3.00
2018-12-07 - Report writing (Niclas Stjernholm)	2.00
2018-12-10 - Training YoloV3 network on MvTech Grocery Dataset and writing on report (Niclas Stjernholm)	4.00

Issue / Worklog	Worked
2018-12-10 - Training YoloV3 network on MvTech Grocery Dataset and writing on report (Niclas Stjernholm)	3.75
2018-12-11 - Training YoloV3 network on MvTech Grocery Dataset and writing on report (Niclas Stjernholm)	4.00
2018-12-11 - Training YoloV3 network on MvTech Grocery Dataset and writing on report (Niclas Stjernholm)	3.50
2018-12-12 - YoloV3 has reached 26000 iterations with a loss of 0.1. Need some test data collected. Writign report (Niclas Stjernholm)	4.00
2018-12-12 - YoloV3 has reached 26000 iterations with a loss of 0.1. Need some test data collected. Writign report (Niclas Stjernholm)	3.50
2018-12-13 - Training with Tiny Yolo.v3 and writing report. (Niclas Stjernholm)	4.00
2018-12-13 - Training with Tiny Yolo.v3 and writing report. (Niclas Stjernholm)	3.75
2018-12-14 - Tiny Yolo.v3 done training, trying to fix bounding box drawing, as it was offset with a corner in the center found in the object. The network trained is doing nice recognition of own images but doesn' t fair too well on new information with objects which differ a little from the ones in the dataset. Writing report. (Niclas Stjernholm)	4.00
2018-12-14 - Tiny Yolo.v3 done training, trying to fix bounding box drawing, as it was offset with a corner in the center found in the object. The network trained is doing nice recognition of own images but doesn' t fair too well on new information with objects which differ a little from the ones in the dataset. Writing report. (Niclas Stjernholm)	3.50
2018-12-17 - Checking trained network on MvTech data looking a bit more into bounding boxes, creating an issue on Github. Writing report. (Niclas Stjernholm)	3.00
2018-12-17 - Checking trained network on MvTech data looking a bit more into bounding boxes, creating an issue on Github. Writing report. (Niclas Stjernholm)	4.50
2018-12-18 - Checking trained network on MvTech data looking a bit more into bounding boxes, creating an issue on Github. Writing report. (Niclas Stjernholm)	2.42
2018-12-18 - Checking trained network on MvTech data looking a bit more into bounding boxes, creating an issue on Github. Writing report. (Niclas Stjernholm)	5.33
2018-12-19 - Writing report. Short day, due to plans (Niclas Stjernholm)	4.00
2018-12-19 - Report writing (Niclas Stjernholm)	3.75
2018-12-20 - Report writing (Niclas Stjernholm)	4.00
2018-12-20 - Report writing (Niclas Stjernholm)	3.75
2018-12-21 - Report writing (Niclas Stjernholm)	4.00
PP-270 - Deprecated Automate basic testing for video compression.....	**14.50**
2018-08-27 - Installation of Realsense software and flashing of Jetson. Had a bit of trouble getting through installation regarding realsense, but figured it out. (Niclas Stjernholm)	3.75
2018-08-28 - Installing realsense and getting realsense sample program running. (Niclas Stjernholm)	1.50
2018-08-31 - Building examples (Niclas Stjernholm)	2.00
2018-09-03 - Looking into different solutions as to store video matrix data (Niclas Stjernholm)	3.75
2018-09-03 - Understanding the 3dq meta solution and creating a store function before the encoding (Niclas Stjernholm)	3.50
PP-365 - Evaluate Aetina TX2 prototype enclosure.....	**1.50**
2018-10-30 - Went to Syndikatet to get 3D printet backplate. (Niclas Stjernholm)	1.50
PP-371 - Daily standup and meetings.....	**1.00**

Issue / Worklog	Worked
2018-10-30 - standup meeting (Niclas Stjernholm)	1.00
PP-387 - Create overview of state of the art on object detection and pose tracking.....	**95.50**
2018-08-21 - Introduction to Random Forests understanding. (Niclas Stjernholm)	2.75
2018-08-21 - An Introduction to Random Forests for Multi-class Object Detection read through and understood (Niclas Stjernholm)	3.75
2018-08-22 - PoseCNN readthrough and understandment, looking into 6D Object pose estimation using Deep learning and RGB-D. (Niclas Stjernholm)	0.50
2018-08-22 - PoseCNN readthrough and understandment, looking into 6D Object pose estimation using Deep learning and RGB-D. (Niclas Stjernholm)	4.25
2018-08-23 - Looking into RGBD-Fusion and depth recovery. (Niclas Stjernholm)	3.75
2018-08-23 - Looking into RGBD-Fusion and depth recovery. (Niclas Stjernholm)	3.25
2018-08-24 - Looking into RGBD-fusion and looking for relevant litterature (Niclas Stjernholm)	3.25
2018-08-24 - Research and documentation of state of the art solutions (Niclas Stjernholm)	2.50
2018-08-27 - Documentation of state of the art object detection articles research. (Niclas Stjernholm)	3.50
2018-08-28 - Standup meeting, looking into datasets for industrial RGB(-D) objects to use for training. (Niclas Stjernholm)	2.00
2018-08-28 - Standup meeting, looking into datasets for industrial RGB(-D) objects to use for training. (Niclas Stjernholm)	4.00
2018-08-29 - Database research and documentation. Also found a possible solution for T-LESS 6D Pose Estimation. (Niclas Stjernholm)	3.50
2018-08-29 - Looking further into solution regarding T-LESS database and other databases. Setting up new pc. (Niclas Stjernholm)	4.00
2018-08-30 - Looking for solutions fro implementing a quick object detection using grayscale images (Niclas Stjernholm)	0.75
2018-08-30 - Looking into database while trying to install tensorflow, CUDA and cudnn (Niclas Stjernholm)	1.75
2018-09-04 - Looking into solutions of yolo for training on the industrial objects. Trying to get Yolo to run. Need an older GCC compiler. (Niclas Stjernholm)	3.50
2018-09-28 - Searching for network design overview including performances etc. (Niclas Stjernholm)	4.00
2018-10-02 - Looking for object tracking state of the art soltutions. The newest Dexnet 3 solution with vacuum suction grasping may be a good solution (Niclas Stjernholm)	3.00
2018-10-03 - Looking into object tracking solutions. Most are made for person tracking e.g. a path in video or pose tracking for human poses. These exist for both regular ML and deep learning (Niclas Stjernholm)	4.00
2018-10-03 - Looking into object tracking solutions. Most are made for person tracking e.g. a path in video or pose tracking for human poses. These exist for both regular ML and deep learning (Niclas Stjernholm)	3.50
2018-10-04 - Looking into Dex-Net and other solutions and databases. (Niclas Stjernholm)	3.50
2018-11-27 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	2.75
2018-11-27 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	5.25
2018-11-28 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	3.50
2018-11-28 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	4.00
2018-11-29 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	3.50
2018-11-29 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	4.00

Issue / Worklog	Worked
2018-11-30 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	3.50
2018-11-30 - Planning rest of internship and working on miniproject for course (Niclas Stjernholm)	4.00
PP-414 - Basic tracking for detected objects.....	**133.50**
2018-09-04 - Looking into training YoLO network with new classes and trainingset and using it on tensorflow instead of darknet. (Niclas Stjernholm)	3.75
2018-09-05 - Getting yolo to run on tensorflow (Niclas Stjernholm)	3.75
2018-09-05 - Working with different implementations of Tensorflow and Yolo. Using darkflow introduces a lot of different errors regarding weights and labels. Looking into using ROS and tensorflow instead. (Niclas Stjernholm)	3.75
2018-09-06 - Trying to resolve issues with different network solutions. (Niclas Stjernholm)	3.50
2018-09-06 - Trying to make a network train on our own dataset. For this we need annotations and not just labels for each image. Or a complete different network layout. (Niclas Stjernholm)	3.75
2018-09-07 - Parsing yml annotations instead of xml for t-less dataset working with darkflow (Niclas Stjernholm)	3.75
2018-09-07 - Parsing yml annotations instead of xml for t-less dataset working with darkflow (Niclas Stjernholm)	0.50
2018-09-10 - Yolo network solutions researched Found a Keras solution, now annotation conversion is next, then training should be possible. (Niclas Stjernholm)	4.00
2018-09-10 - Implementing keras solution of yolo, annotations needs are being converted. (Niclas Stjernholm)	3.75
2018-09-11 - Creating annotation txt file for t-less dataset (Niclas Stjernholm)	4.00
2018-09-11 - Finished annotations script, and now training a yolo network. (Niclas Stjernholm)	3.75
2018-09-12 - Training of the yolo network using keras and the T-LESS dataset. Diving deeper in the keras implementation. (Niclas Stjernholm)	4.25
2018-09-12 - Still training the yolo network on keras with T-LESS. Looking into usin gonly a few of the objects from the dataset together with the VOC dataset. (Niclas Stjernholm)	3.50
2018-09-13 - Training the Yolo network using the bottleneck version of training. (Niclas Stjernholm)	4.00
2018-09-13 - Trying the get the keras yolo model to analyse a small video and training the bottleneck network. Started a normal training on the combined dataset. (Niclas Stjernholm)	2.50
2018-09-14 - Training of network and testing on small video. Now using Tiny Yolo for training as this is a smaller network, and might be able to complete more training before resource exhaustion occurs. (Niclas Stjernholm)	3.00
2018-09-14 - Further training of the tiny network. Looking into the model meanwhile. (Niclas Stjernholm)	1.75
2018-09-17 - Looking for a solution to overfitting and too high a loss for the YOLO3 keras network. Trying a new training, now using the normal yolo3 network with pretrained darknet weights. (Niclas Stjernholm)	3.00
2018-09-17 - Looking for a solution to overfitting and too high a loss for the YOLO3 keras network (Niclas Stjernholm)	3.50
2018-09-17 - Looking for a solution to overfitting and too a loss for the YOLO3 keras network (Niclas Stjernholm)	0.50
2018-09-18 - Finished training started last afternoon. Results are better, recognisition of the fuse is achieved. Tweaking and research in tweaking today. (Niclas Stjernholm)	4.00
2018-09-18 - Finished training started last afternoon. Results are better, recognisition of the fuse is achieved. Tweaking and research in tweaking today. (Niclas Stjernholm)	3.75
2018-09-19 - Training from yesterday still running. Still looking into network tweaks and practicing python programming in general. (Niclas Stjernholm)	4.00

Issue / Worklog	Worked
2018-09-19 - Training from yesterday still running. Still looking into network tweaks and practicing python programming in general. (Niclas Stjernholm)	3.50
2018-09-20 - Previous training didn't yield nice results. Labeling needs to be fixed, as this may cause problems. Checking this by training only the picked t-less objects. (Niclas Stjernholm)	4.00
2018-09-20 - Previous training didn't yield nice results. Labeling needs to be fixed, as this may cause problems. Checking this by training only the picked t-less objects. (Niclas Stjernholm)	3.50
2018-09-21 - Training froze during the evening, started it over. It will not finish until sometime during the night to Saturday. (Niclas Stjernholm)	3.25
2018-09-24 - Still too high a loss. Classes for t-less set needed a fix, which is done. Training is started again with proper classes and another type of weights. (Niclas Stjernholm)	4.00
2018-09-24 - Still training from the session before lunch. Looking further into the general keras setup and .py files (Niclas Stjernholm)	3.50
2018-09-25 - Evaluating result from the newest keras yoloV3 training. (Niclas Stjernholm)	1.00
2018-09-25 - Looking into decision trees, how they are programmed in python. (Niclas Stjernholm)	3.00
2018-09-25 - Looking at implementations of random decision trees and forests similar to Multi-forest Tracker: A Chameleon in Tracking. (Niclas Stjernholm)	3.50
2018-09-26 - Researching the design of random forests. (Niclas Stjernholm)	3.00
2018-09-26 - Researching the design of random forests. (Niclas Stjernholm)	2.50
2018-09-27 - Lecture/Meeting with Ivan from Globus AI (Niclas Stjernholm)	1.25
2018-09-27 - Researching the design of random forests. (Niclas Stjernholm)	0.50
2018-09-27 - Planning workload and focus with Raphael (Niclas Stjernholm)	0.50
2018-09-27 - Researching the design of random forests. (Niclas Stjernholm)	4.00
2018-09-27 - Researching the design of random forests. (Niclas Stjernholm)	1.50
2018-11-15 - Mini project work for course (Niclas Stjernholm)	7.50
2018-11-16 - Mini project work for course. (Niclas Stjernholm)	7.50
PP-436 - Realsense best practices for depth data.....	**4.25**
2018-10-08 - Found Intel's own recipe for best practices with the Realsense cameras. The document is uploaded to the assignment in Jira and a summary is in Confluence. (Niclas Stjernholm)	4.25
PP-437 - find general dataset for default use in training.....	**12.00**
2018-10-01 - Looking for different datasets for default use case. The Dexnet proposal by Raphael seems very viable given the argumentation of the article. (Niclas Stjernholm)	3.00
2018-10-01 - Looking for different datasets for default use case. The Dexnet proposal by Raphael seems very viable given the argumentation of the article. (Niclas Stjernholm)	4.50
2018-10-02 - As the Dexnet dataset isn't RGB-D but point clouds, I have been looking for regular RGB-D databases. (Niclas Stjernholm)	4.50
PP-439 - Check license on GQ-CNN data.....	**0.50**
2018-10-05 - There seems to be no further licensing on the dataset, than what is already on the entire solution. (Niclas Stjernholm)	0.50
PP-440 - Research Dex-Net Object Classification.....	**101.25**
2018-10-04 - In general looking deeper into Dex-Net and how the implementation is done. (Niclas Stjernholm)	2.75

Issue / Worklog	Worked
2018-10-09 - Researching the dexnet classification process. It was a bit harder than expected to find their documentation on this matter. (Niclas Stjernholm)	2.75
2018-10-09 - Researching the dexnet classification process. It was a bit harder than expected to find their documentation on this matter. (Niclas Stjernholm)	5.00
2018-10-10 - Researching the network build and how the data is prepared and fed to the network. (Niclas Stjernholm)	4.00
2018-10-10 - Researching the network build and how the data is prepared and fed to the network. (Niclas Stjernholm)	3.50
2018-10-11 - Documentation of the dex-net network building (Niclas Stjernholm)	2.00
2018-10-22 - Looking further into the dex net solution (Niclas Stjernholm)	3.00
2018-10-22 - Looking further into the dex net solution (Niclas Stjernholm)	2.75
2018-11-02 - Looking into dexnet setup, what is needed and such. (Niclas Stjernholm)	3.00
2018-11-05 - Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	2.75
2018-11-05 - Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	5.00
2018-11-06 - Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	4.00
2018-11-06 - Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	3.75
2018-11-07 - Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	2.00
2018-11-07 - As I had a janitor coming to fix something in my apartment I had to take a break at 10:00, so I took a break there and ate lunch while working. Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	5.50
2018-11-08 - Got sick, so I went home at noon. Didn't meet in on Friday. Researching the Dexnet network build and training setup. Looking into the data fed to the network, refreshing previously gained knowledge to verify. (Niclas Stjernholm)	4.50
2018-11-12 - Getting the dexnet example running. They have an argument passer which messes up. Will look at the dexnet 2.0 instead to see if the Github example is better supported for open source usage. (Niclas Stjernholm)	2.50
2018-11-12 - Getting the dexnet example running. They have an argument passer which messes up. Will look at the dexnet 2.0 instead to see if the Github example is better supported for open source usage. (Niclas Stjernholm)	5.00
2018-11-13 - Getting dexnet 2.0 running and helping prepare the Jetsons for dispatch. (Niclas Stjernholm)	2.50
2018-11-13 - Getting dexnet 2.0 running and helping prepare the Jetsons for dispatch. (Niclas Stjernholm)	5.25
2018-11-14 - Getting the Dexnet dependencies installed and setup. Figuring out how to get it running. (Niclas Stjernholm)	7.50
2018-11-21 - Handling project report setting up git repo and preparing for documentation (Niclas Stjernholm)	5.00
2018-11-22 - Figuring out how dexnet2 is working. Trying to get it running, having some issues with the eGPU. (Niclas Stjernholm)	7.75
2018-11-23 - Solving issues with dexnet2. Supervisor visit - I need to do some planning on the rest of the intership and figure out what the headline should be. Planning a bit afterwards and documenting. (Niclas Stjernholm)	7.50
2018-11-26 - Working a bit more on Dexnet fix, and planning of rest of the internship. (Niclas Stjernholm)	2.00

Issue / Worklog	Worked
PP-444 - GQ-CNN Object and training-set process.....	**3.25**
2018-10-08 - Looking into the setup and training of the network reading through the guide made for it. (Niclas Stjernholm)	3.25
PP-459 - Create GSt client that receivees RTP and stores to file.....	**27.25**
2018-10-11 - Looking for solutions on storing a file and understanding the different commands for gst (Niclas Stjernholm)	2.50
2018-10-11 - Understanding the receiver and sender in the RPT GST client (Niclas Stjernholm)	3.00
2018-10-12 - UNderstainding Gstreamer, and figuring out what needs to be done to use filesink (Niclas Stjernholm)	4.75
2018-10-15 - Trying to read the file, decompress it and show the video saved, but the files format saved isn't able to open. (Niclas Stjernholm)	1.50
2018-10-31 - Looking into the newest proposed pipeline for storing the received files and implementing this in the receiver.c. Having issues with building as the combiner.h is missing, combiner.hpp is giving a few issues. (Niclas Stjernholm)	4.00
2018-10-31 - Looking into the newest proposed pipeline for storing the received files and implementing this in the receiver.c. Having issues with building as the combiner.h is missing, combiner.hpp is giving a few issues. (Niclas Stjernholm)	3.75
2018-11-01 - battling with issues with the file storing and playback pipeline. The commands work, but building the execs is troublesome because of warnings and the conversion to C++. (Niclas Stjernholm)	2.50
2018-11-01 - battling with issues with the file storing and playback pipeline. The commands work, but building the execs is troublesome because of warnings and the conversion to C++. (Niclas Stjernholm)	5.25
PP-461 - Create GStapp that opens files cont. RTP packages, uncompresses and displays them	**15.50**
2018-10-15 - Looking into opening the file, decompressing and displaying a video (Niclas Stjernholm)	1.75
2018-10-24 - Looking into the displaying of the stored file. It seems like the problem may be the sender and not the receiver. The missing link is the stored file is missing a TIME-format. Trying to resolve this. (Niclas Stjernholm)	4.00
2018-10-24 - Looking into the displaying of the stored file. It seems like the problem may be the sender and not the receiver. The missing link is the stored file is missing a TIME-format. Trying to resolve this. (Niclas Stjernholm)	3.75
2018-10-25 - Still trying to find a solution to the issue. Tried to checkout another branch, but the playback of the video recorded on the webcam only showed a black screen. (Niclas Stjernholm)	1.25
2018-10-25 - Still trying to find a solution to the issue. Tried to checkout another branch, but the playback of the video recorded on the webcam only showed a black screen. (Niclas Stjernholm)	4.75
PP-462 - Create GStapp that receives RTP (via UDP) uncompresses and displays them.....	**20.75**
2018-10-16 - Getting the new receiver built and understanding the new code made. (Niclas Stjernholm)	1.00
2018-10-17 - Looking into uncompressing the stored files (Niclas Stjernholm)	1.00
2018-10-17 - Deciphering the new code, exploring the features including already and looking for a way to open the files. We need some other kind of sample than numbers to experiment properly. (Niclas Stjernholm)	1.00
2018-10-18 - Finding a solution for opening the video and setting up a test sender, sending videotestsrc as payload. Since it's a long time since I last coded C, and the lack of experience with gstreamer, this takes longer than expected. (Niclas Stjernholm)	3.50
2018-10-18 - Finding a solution for opening the video and setting up a test sender, sending videotestsrc as payload. (Niclas Stjernholm)	2.50
2018-10-19 - Trying to get gstreamer to first save a fiel recieved from rtp and the play it. (Niclas Stjernholm)	1.75

Issue / Worklog	Worked
2018-10-19 - Trying to get gstreamer to first save a fiel recieved from rtp and the play it. (Niclas Stjernholm)	4.00
2018-10-22 - Trying to resolve the opening of a filesinked file. (Niclas Stjernholm)	2.00
2018-11-02 - Still having issues, with the building of the receiver. Passing it on, on monday. (Niclas Stjernholm)	4.00
PP-463 - Balenarise dynamic_configure frontend.....	**19.50**
2018-10-25 - Looking into docker and learning it. (Niclas Stjernholm)	1.75
2018-10-26 - Looking into docker and learning it. (Niclas Stjernholm)	4.00
2018-10-26 - Looking into docker and learning it. Trying to acces Gitlab docker, denied acces started looking in the dockerfile 20 minutes before leaving for the day. (Niclas Stjernholm)	2.50
2018-10-29 - Looking into the docker and building the resinOS image for melodic. (Niclas Stjernholm)	0.75
2018-10-29 - Looking into the docker and building the resinOS image for melodic. (Niclas Stjernholm)	5.00
2018-10-30 - Looking into the docker and building the resinOS image for melodic. Building the web_dynamic_configure on the new image. Pushed as web_dyn_reconf:resinOS (Niclas Stjernholm)	3.00
2018-10-30 - Looking into the docker and building the resinOS image for melodic. (Niclas Stjernholm)	1.75
2018-10-31 - Trying to build, running into errors. Raphael made the image during the evening. (Niclas Stjernholm)	0.75
PP-533 - SCRUM Kickoff and Planning.....	**6.00**
2018-11-26 - Planning and refining of SCRUM implementation (Niclas Stjernholm)	3.00
2018-11-26 - Planning and refining of SCRUM implementation (Niclas Stjernholm)	3.00

Appendix B

Python scripts

B.1 Json Conversion to txt

```
import os
import json
from os import listdir, getcwd
from os.path import join

classes = ["adelholzener_alpenquelle_classic_075",
    "adelholzener_alpenquelle_naturell_075",
    "adelholzener_classic_bio_apfelschorle_02",
    "adelholzener_classic_naturell_02",
    "adelholzener_gourmet_mineralwasser_02",
    "augustiner_lagerbraeu_hell_05", "augustiner_weissbier_05",
    "coca_cola_05", "coca_cola_light_05", "suntory_gokuri_lemonade",
    "tegernseer_hell_03", "corny_nussvoll", "corny_nussvoll_single",
    "corny_schoko_banane", "corny_schoko_banane_single",
    "dr_oetker_vitalis_knuspermuesli_klassisch",
    "koelln_muesli_fruechte", "koelln_muesli_schoko", "caona_cocoa",
    "cocoba_cocoa", "cafe_wunderbar_espresso",
    "douwe_egberts_professional_ground_coffee",
    "gepa_bio_caffe_crema", "gepa_italienischer_bio_espresso",
    "apple_braeburn_bundle", "apple_golden_delicious",
    "apple_granny_smith", "apple_red_boskoop", "avocado",
    "banana_bundle", "banana_single", "clementine",
    "clementine_single", "grapes_green_sugraone_seedless",
    "grapes_sweet_celebration_seedless", "kiwi", "orange_single",
    "oranges", "pear", "pasta_reggia_elicoidali",
    "pasta_reggia_fusilli", "pasta_reggia_spaghetti",
    "franken_tafelreiniger", "pelikan_tintenpatrone_canon",
    "ethiquable_gruener_tee_ceylon", "gepa_bio_und_fair_fencheltee",
    "gepa_bio_und_fair_kamillentee",
    "gepa_bio_und_fair_kraeuterteemischung",
    "gepa_bio_und_fair_pfefferminztee",
    "gepa_bio_und_fair_rooibostee", "kilimanjaro_tea_earl_grey",
    "cucumber", "carrot", "corn_salad", "lettuce", "vine_tomatoes",
    "roma_vine_tomatoes", "rocket", "salad_iceberg", "zucchini"]
```

```
7
8  label_dir = '/home/nicstar/Documents/aivero/d2s/d2s_annotations_v1.1/
       annotations/D2S_validation_wo_occlusion'
9
10 #box form[x,y,w,h]
11 def convert(size,box):
12     dw = 1./size[0]
13     dh = 1./size[1]
14     x = box[0]*dw
15     y = box[1]*dh
16     w = box[2]*dw
17     h = box[3]*dh
18     return (x,y,w,h)
19
20 def convert_annotation():
21     if not os.path.exists(label_dir):
22         os.makedirs(label_dir)
23     with
           open('/home/nicstar/Documents/aivero/d2s/d2s_annotations_v1.1/
           annotations/D2S_validation_wo_occlusion.json','r') as f:
24         data = json.load(f)
25     for item in data['images']:
26         image_id = item['id']
27         file_name = item['file_name']
28         width = item['width']
29         height = item['height']
30         value = filter(lambda item1: item1['image_id'] ==
               image_id,data['annotations'])
31         outfile =
               open('/home/nicstar/Documents/aivero/d2s/d2s_annotations_v1.1/
               annotations/D2S_validation_wo_occlusion/%s.txt'%(file_name[:-4]),
               'a+')
32         for item2 in value:
33             category_id = item2['category_id']
34             value1 = filter(lambda item3: item3['id'] ==
                   category_id,data['categories'])
35             name = value1[0]['name']
36             class_id = classes.index(name)
37             box = item2['bbox']
38             bb = convert((width,height),box)
39             outfile.write(str(class_id)+" "+" ".join([str(a) for a in
                   bb]) + '\n')
40         outfile.close()
41
42 if __name__ == '__main__':
43     convert_annotation()
```

B.2 T-Less Annotation Conversion to txt

```
1  #conversion of annotations from yml to txt
2
3  import numpy as np
4  import os
```

```
import ruamel.yaml as yaml

#range for the entire dataset, sets for the picked ones
#obj_ids = range(1, 31)
obj_ids = range(0,5)

cwd = os.getcwd()
data_path = os.path.join(cwd, 'data/t-less_v2')

#choose entire set or just the picked objects
#obj_gt_path_mask = os.path.join(data_path, 'test_primesense',
    '{:02d}', 'gt.yml')
#folder_path =  os.path.join(data_path, 'test_primesense',
    '{:02d}','rgb','{:04d}.png')
obj_gt_path_mask = os.path.join(data_path, 'train_primesense_pick',
    '{:02d}', 'gt.yml')
folder_path =  os.path.join(data_path, 'train_primesense_pick',
    '{:02d}','rgb','{:04d}.png')


def load_gt(path):
    with open(path, 'r') as f:
        gts = yaml.load(f, Loader=yaml.CLoader)
        for im_id, gts_im in gts.items():
            for gt in gts_im:
                if 'obj_bb' in gt.keys():
                    gt['obj_bb'] = [int(x) for x in gt['obj_bb']]
                    gt['obj_bb'] = [gt['obj_bb'][0], gt['obj_bb'][1],
                        gt['obj_bb'][0] + gt['obj_bb'][2],
                        gt['obj_bb'][1]+gt['obj_bb'][3]]

                    gts[im_id] = gt['obj_bb']

    return gts

for obj_id in obj_ids:

    #loading info of images
    obj_gt_path = obj_gt_path_mask.format(obj_id)
    object_gt = load_gt(obj_gt_path)

    for i in object_gt:
        #choose one of the text files based on the dataset.
        text_file = open("t-less-annotations_pick.txt", "a+")
        #text_file = open("t-less-annotations-single.txt", "a+")
        text_file.write(', '.join([folder_path.format(obj_id,i) + ' '
            + ','.join(map(str, object_gt[i]))+','+str(obj_id)]) +
            '\n')
        text_file.close()
        #print(str(folder_path.format(obj_id)) + '
            ,'.join(map(str,object_gt[i])) + i)
```

Appendix C

Object Class Precision of D2S database

```
class_id = 0, name = adelholzener_alpenquelle_classic_075,       ap = 45.09 %
class_id = 1, name = adelholzener_alpenquelle_naturell_075,      ap = 63.64 %
class_id = 2, name = adelholzener_classic_bio_apfelschorle_02,   ap = 53.34 %
class_id = 3, name = adelholzener_classic_naturell_02,   ap = 53.53 %
class_id = 4, name = adelholzener_gourmet_mineralwasser_02,      ap = 52.99 %
class_id = 5, name = augustiner_lagerbraeu_hell_05,      ap = 62.81 %
class_id = 6, name = augustiner_weissbier_05,    ap = 53.61 %
class_id = 7, name = coca_cola_05,       ap = 53.49 %
class_id = 8, name = coca_cola_light_05,         ap = 44.93 %
class_id = 9, name = suntory_gokuri_lemonade,    ap = 54.38 %
class_id = 10, name = tegernseer_hell_03,        ap = 53.03 %
class_id = 11, name = corny_nussvoll,    ap = 44.84 %
class_id = 12, name = corny_nussvoll_single,     ap = 45.24 %
class_id = 13, name = corny_schoko_banane,       ap = 44.30 %
class_id = 14, name = corny_schoko_banane_single,        ap = 54.49 %
class_id = 15, name = dr_oetker_vitalis_knuspermuesli_klassisch,
            ap = 39.73 %
class_id = 16, name = koelln_muesli_fruechte,    ap = 36.93 %
class_id = 17, name = koelln_muesli_schoko,      ap = 51.51 %
class_id = 18, name = caona_cocoa,       ap = 20.09 %
class_id = 19, name = cocoba_cocoa,      ap = 44.42 %
class_id = 20, name = cafe_wunderbar_espresso,   ap = 33.58 %
class_id = 21, name = douwe_egberts_professional_ground_coffee,
             ap = 43.44 %
class_id = 22, name = gepa_bio_caffe_crema,      ap = 33.86 %
class_id = 23, name = gepa_italienischer_bio_espresso,   ap = 43.60 %
class_id = 24, name = apple_braeburn_bundle,     ap = 44.48 %
class_id = 25, name = apple_golden_delicious,    ap = 54.40 %
class_id = 26, name = apple_granny_smith,        ap = 54.33 %
class_id = 27, name = apple_red_boskoop,         ap = 54.44 %
```

```
class_id = 28, name = avocado,   ap = 54.46 %
class_id = 29, name = banana_bundle,     ap = 44.71 %
class_id = 30, name = banana_single,     ap = 54.24 %
class_id = 31, name = clementine,        ap = 53.37 %
class_id = 32, name = clementine_single,         ap = 54.25 %
class_id = 33, name = grapes_green_sugraone_seedless,    ap = 61.98 %
class_id = 34, name = grapes_sweet_celebration_seedless,         ap =
    53.65 %
class_id = 35, name = kiwi,      ap = 54.52 %
class_id = 36, name = orange_single,     ap = 63.40 %
class_id = 37, name = oranges,   ap = 47.80 %
class_id = 38, name = pear,      ap = 54.55 %
class_id = 39, name = pasta_reggia_elicoidali,   ap = 39.05 %
class_id = 40, name = pasta_reggia_fusilli,      ap = 37.67 %
class_id = 41, name = pasta_reggia_spaghetti,    ap = 38.51 %
class_id = 42, name = franken_tafelreiniger,     ap = 45.17 %
class_id = 43, name = pelikan_tintenpatrone_canon,       ap = 54.30 %
class_id = 44, name = ethiquable_gruener_tee_ceylon,     ap = 44.71 %
class_id = 45, name = gepa_bio_und_fair_fencheltee,      ap = 44.53 %
class_id = 46, name = gepa_bio_und_fair_kamillentee,     ap = 54.55 %
class_id = 47, name = gepa_bio_und_fair_kraeuterteemischung,     ap =
    53.48 %
class_id = 48, name = gepa_bio_und_fair_pfefferminztee,          ap =
    53.61 %
class_id = 49, name = gepa_bio_und_fair_rooibostee,      ap = 53.80 %
class_id = 50, name = kilimanjaro_tea_earl_grey,         ap = 42.89 %
class_id = 51, name = cucumber,          ap = 42.79 %
class_id = 52, name = carrot,    ap = 54.17 %
class_id = 53, name = corn_salad,        ap = 38.45 %
class_id = 54, name = lettuce,   ap = 36.87 %
class_id = 55, name = vine_tomatoes,     ap = 34.25 %
class_id = 56, name = roma_vine_tomatoes,        ap = 45.22 %
class_id = 57, name = rocket,    ap = 38.59 %
class_id = 58, name = salad_iceberg,     ap = 52.09 %
class_id = 59, name = zucchini,,         ap = 54.17 %

for thresh = 0.25, precision = 0.86, recall = 0.51, F1-score = 0.64
for thresh = 0.25, TP = 19429, FP = 3039, FN = 18837, average IoU =
    68.63 %

mean average precision (mAP) = 0.481719, or 48.17 %
```