Hochschule Esslingen
University of Applied Sciences

Object Detection and Tracking Systems with Stereo Vision for Autonomous Driving

Burak Erdogan

Master of Science Thesis in Automotive System Management-Car Electronics Department of Hochschule Esslingen

Supervisor: **Prof. Thao Dang**

Acknowledgments

First of all, I would like to thanks to my family to give me this opportunity of doing master in Hochschule Esslingen in Germany. Despite all the obstacles, their endless support is one of the biggest motivation for me. I also want to thank my friend Gokce Demir and all my other friends who supported me during this master program.

Last but not least, special thanks to my supervisor in Hochschule Esslingen Prof. Thao Dang, not only for his technical feedback but also his precious suggestion for my career. I will appreciate and take his ideas into account all my life.

Esslingen, August 2019

Burak Erdogan

Abstract

Object detection and tracking systems have become one of the most important topics for zero-emission electric cars as autonomous driving turned into reality from a futuristic idea. First, We are going to examine related works. After that, we are going to have a look at STIXEL method which is used as an object detection algorithm in this master thesis. Next step is going to be examining and comparing different tracking algorithm each other in OpenCV library. At the last step, we are going to configure the CSRT tracking algorithm by adding depth information which is obtained from STIXEL method as a third channel.

Throughout all these processes KITTI dataset is going to be used as a sample video sequence. In order to compare different tracking algorithm each other, we are going to use tracklets information in KITTI dataset

Table of Contents

List of Figures

List of Tables

Nomenclature

LIDAR : Light Detection and Ranging

SONAR: Sound navigation ranging

DARPA: The Defence Advanced Research Projects Agency

RADAR: Radio Detection and Ranging

AEB: Autonomous Emergency Braking

ACC: Adaptive Cruise Control

SAD: Sum of Absolute Differences

CNN: Convolutional Neural Network

ReLU: Rectified Linear Units

RCNN: Region-Convolutional Neural Network

SVM: Support Vector Machine

RPN: Regional Proposal Network

YOLO: You Only Look Once

SGM: Semi-Global Matching

ADAS: Advanced Driver-Assistance Systems

CPU: Central Processing Unit

FPGA: Field-Programmable Gate Array

DP: Dynamic Programming

RGB: Red-Green-Blue

MOSSE: Minimum Output Sum of Squared Error

FFT: Fast Fourier Transform

KCF: Kernelized Correlation Filter

TLD: Tracking-Learning-Detection

MIL: Multiple Instance Learning

GOTURN: Generic Object Tracking Using Regression Network

CSRT: Discriminative Correlation Filter with Channel and Spatial Reliability

ROS: Robot-Operating System

1.Introduction

1.1 Overview

In this master thesis, there will be 5 sections which are Introduction, Fundamental, Object Detection Stixel, Object Tracking in Video Sequence, Evaluation, Conclusion, and Outlook.

In Introduction, the purpose of the thesis and research questions will be mentioned.

In Fundamental section, computer vision and the stereo image will be introduced. Because those two topics are basic concepts of the background of the project.

In the third section, the object detection systems will be introduced. STIXEL method was used as the object detection algorithm, therefore STIXEL method will be explained in detail.

In the Object Tracking section, different object tracking algorithms will be investigated. CSRT method is the method that was changed and improved in the project. Therefore CSRT will be presented longer than other tracker methods.

In Evaluation part, the different results of the tracking methods for the same video sequence will be compared. Especially improvements in CSRT method will be presented in this section with graphs.

In the last part, we are going to discuss of results that are obtained by using different methods, image formats. There are some steps that are not completed because of time limitation. At the end of this section, those steps and possible future studies on this thesis will be presented.

1.2 Purpose

The purpose of this project is examining the object tracking systems by using depth information is obtained by a stereo camera. Although, there are many studies on object tracking systems, in this thesis the effect of depth information to object tracking systems will be investigated.

1.3 Research Questions

1. What is computer vision and what are the different methods for computer vision?

2. What is the stereo vision? How is STIXEL method implemented?

3. How to use KITTI dataset and how can a result data be compared?

4. What is Open-CV? What is the different object tracking algorithms in Open-CV?

5. What is the CSRT object tracking algorithm?

6. How to get feedback from the CSRT algorithm to modify the tracking system?

7. How to use depth information with CSRT method in order to get a better result?

8. What is Robot Operating System(ROS) and how to transfer 2 dimension objects information to 3D ROS system

In the following chapters, those questions are going to be answered step-by-step. The results of the experiments are going to be discussed.

Because of the time limitation, some steps that are decided at the beginning could not be completed. In the last chapter, those steps are going to be explained and the possible future studies on this thesis are going to be represented

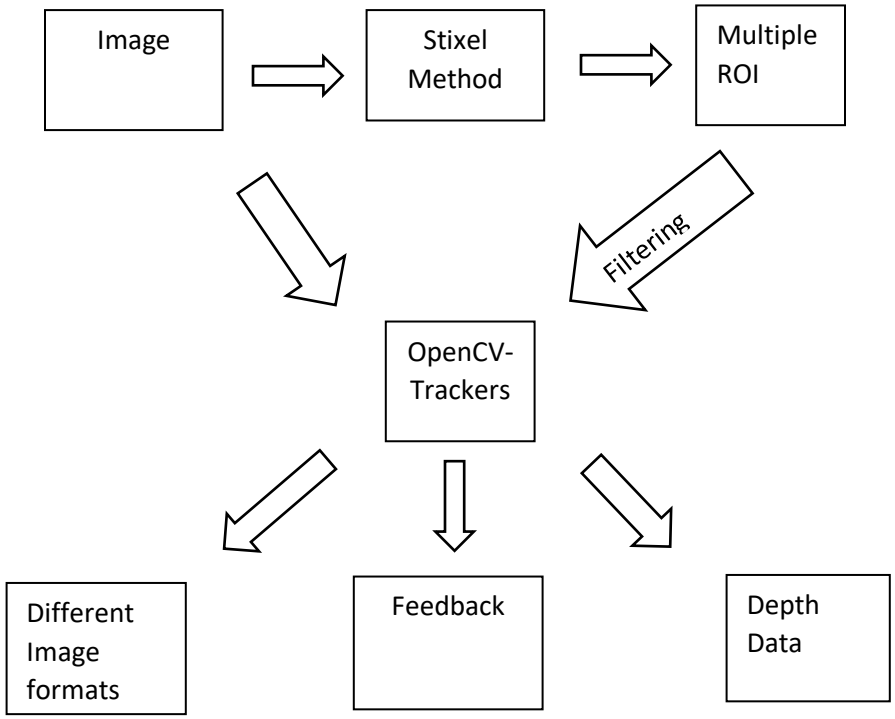


Figure 1: Project Diagram

2. Fundamentals

2.1 Computer Vision

The concept of computer vision came out in the late 1950s. One of the most important milestones was research that was published by two neurophysiologists David Hubel and Torsten Wiesel [1]. Those two scientists were making an experiment in order to observe cats neural reactions for a visual experience. They accidentally or luckily realized that the neuron gives a reaction to a sharp edge.

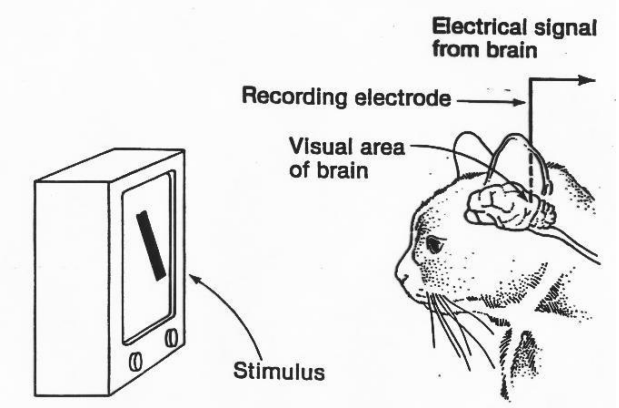


Figure 2 [2]: Neuron reaction experiment

In 1959, Russel Kirsch achieved to transform images into matrix values with the machine was developed by himself and his colleagues. With this step, images have been processed by a computer in its own language. One of the first digitally scanned images was Russel's son was in a 5cm by 5cm format.

Figure 3 [3]: The photo of Russel Kirsch's son

Transferring a photo into grids of numbers was the beginning of technic that we call today 'Digital Signal Processing'. By the time with technological developments, the interaction between machines and real-world has reached state-of-art level.

Especially, with the new software algorithms like machine learning method, computer vision almost allows AI to have a better vision capability than a human being has. This improvement, of course, has changed the driving technology.

Buick Centurion of General Motors Motorama's concept car was the first car that rear camera system was used in 1956. Until 1991 camera systems were used in different concept models but the first production automobile with camera system was 1991 Toyota Soarer. Developing in sensor technology has made those systems cheaper in order to implement in an automobile. However, today simple cameras are not the only tools are used in the ADAS system but RADAR, LIDAR, Ultrasonic Camera are also different tools that are being used to develop new technology for autonomous driving.

Figure 4 [4]: Buick Centurion Concept Car

2.2 SENSORS

2.2.1 LIDAR

LIDAR "Light Detection and Ranging" sensor works like SONAR but it uses beams of laser light instead of sound waves. The system fires off millions of beams of light per second and then measures the time that for the beams of light turn back to the sensor. With this process, a LIDAR system can create a 3D map and detect any object around the system.

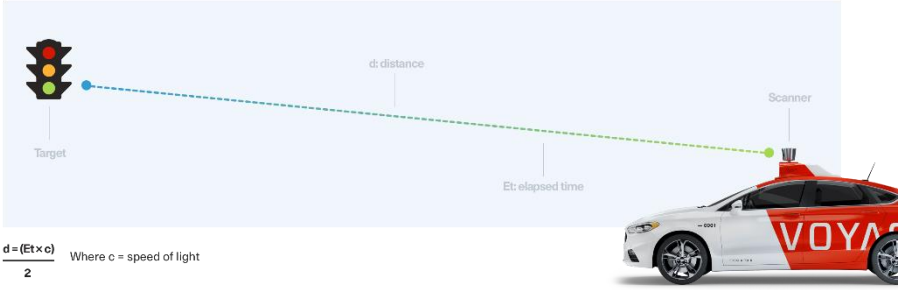


Figure 5 [5]: Basic of LIDAR System

LIDAR was invented in the 1960s, right after the invention of the laser. It was used in the Apollo 15 mission in 1971. Than, LIDAR was mostly used in archeology. The system was providing enormous mapping data to a scientist.

LIDAR was not considered to use in the autonomous system until the 2000s. When Stanley [6] won the 2005 Grand DARPA Challenge, LIDAR started to become more popular for autonomous driving. In 2007 DARPA challenge most of the teams used LIDAR as the basis of their mapping system.

Despite its success in 3D mapping, LIDAR is not used in a production automobile because of its cost. The market leader of electric car TESLA does not prefer to use LIDAR but the companies like Google, Uber which are investing to research on autonomous driving use LIDAR in their concept cars. However, by the new startups on LIDAR systems, it is expected that the cost of a LIDAR system will decrease enough to use in a car.

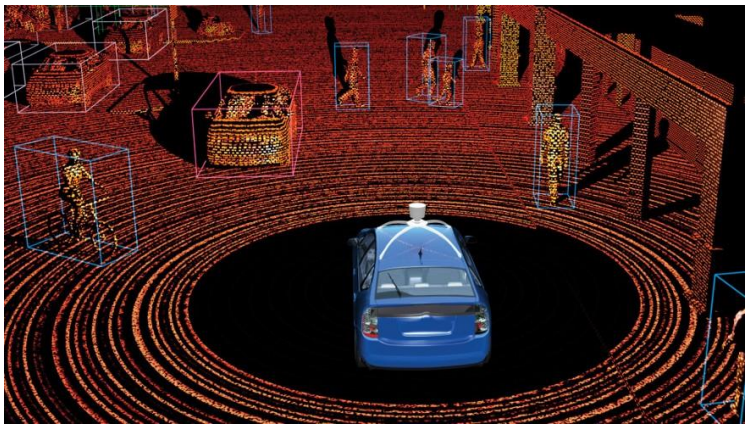


Figure 6 [5]: Google's car Waymo with LIDAR

2.2.2 RADAR

Radar works in the same principle as LIDAR. It uses radio wave to detect objects. However, Radio waves have less absorption than light waves. Therefore, accuracy in RADAR is worse than LIDAR system's accuracy.

Radar systems are already being used in many ordinary cars in assistance systems such as Autonomous Emergency Braking (AEB) and Adaptive Cruise Control(ACC). Radars are used mostly for detecting metallic objects. It's limited ability to define objects makes it less usable for mapping in autonomous driving

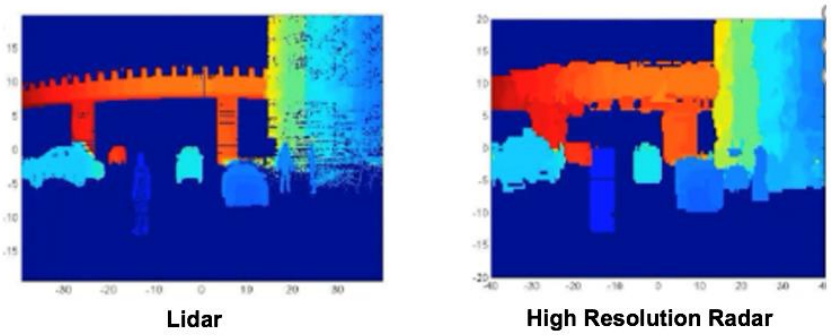


Figure 7 [7]: LIDAR vs RADAR

2.2.3 Ultrasonic Sensors

Ultrasonic Sensors have been used since the 1990s as assistance parking sensors. They use sound waves, therefore, their reaction times are considerably low. Their range is also too limited but thanks to their low price, they are considered as an additional sensing accuracy.

2.2.4 Camera

The camera system has been used in automobiles in many assistant systems for years. Its low price makes it affordable easily for mass-production automobiles. For example in 2018 in USA backup camera system has become mandatory for all new vehicles [8]. However, it has become the main system instead of an assistant system by autonomous driving era.

Unlike LIDAR, RADAR or Ultrasonic Sensor, the camera is a passive sensor that gathers light from the environment. Although it provides high-quality data, this data needs to be processed by some computer vision algorithms like Deep Learning in order to detect and track the objects.

In our project, stereo vision dataset from KITTI was used. Therefore in the next chapter, the idea of stereo vision will be explained and in further chapter KITTI dataset also will be explained in detail.

2.3 Stereo Vision

2.3.1 Basics of Stereo Vision

As it is mentioned in the last chapter 2.2.4, Camera is a passive sensor. That means it collects light from the environment as raw data. Some specific cameras can provide direct information like thermal cameras can provide heat information of objects.

In order to have the distance information of any object, one camera is not enough.

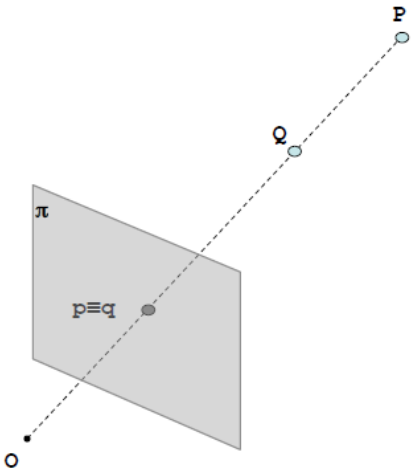


Figure 8 [9]: Two image in the same point on the camera lens

o = optical center

π = image plane

P= object , p = object image in camera's image plane

Q= object , q = object image in camera's image plane

Z= distance from an object

Xr, Xt= horizontal reference frames of the left and right imagers

f= focal length

b= Baseline

As it was shown in Figure 8, two objects in different distances from the camera might have the same image point on the camera lens. Therefore, we need at least two cameras to detect the position of an object.

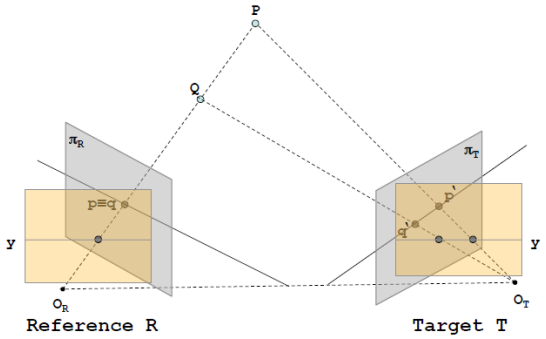


Figure 9 [9]: Two camera with two objects

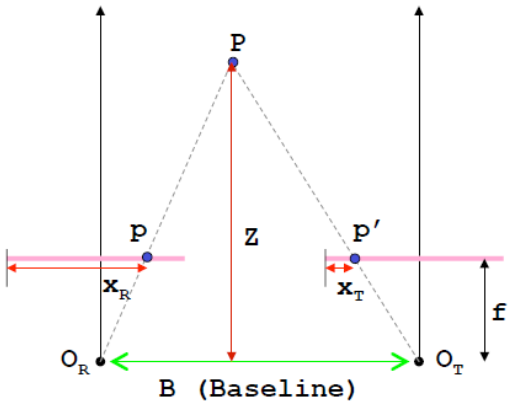


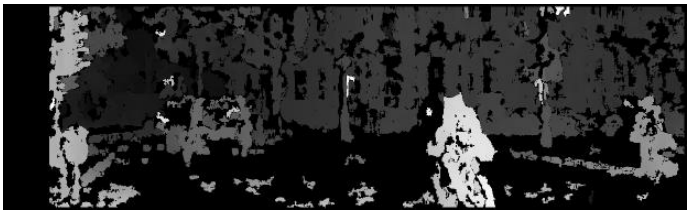
Figure 10 [9]: Object P and it's image positions on both camera lenses

With perfectly aligned two cameras, we can calculate the depth of an object by applying eq (1) for similar triangles.

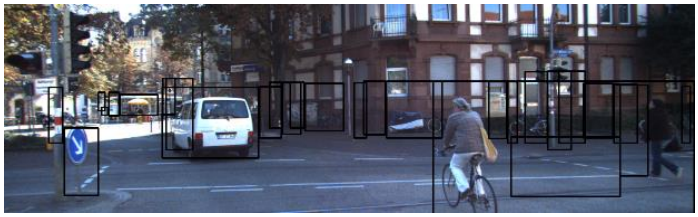
$$\frac{b}{Z} = \frac{(b+Xt)-Xr}{Z-f} \quad (1)$$

$$Z = \frac{b*f}{Xr-Xt} \quad (2)$$

Xr – Xt is known as disparity (d) that shows differences in horizontal axes of the image of an object on both lenses. According to equation 2, there is an inverse ratio between disparity and Z. That means, disparity value gets higher as the object gets closer to the stereo camera system.



Disparity Image



Stereo Image

Figure 11: Disparity Image and Stereo Image

2.3.2 Stereo Calibration

In the last chapter 2.3.1. the basic equation of stereo vision was explained. However, this equation can be used directly in case of having perfectly aligned ideal stereo geometry. Unfortunately, this kind of perfect systems only exists theoretically. Therefore, real stereo cameras need to be calibrated.

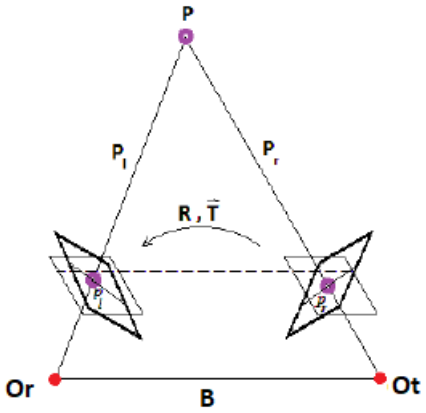


Figure 12: Realistic Stereo Camera Geometry

The first step of the calibration is finding geometric positions of the left and right cameras against each other. In order to get cameras aligned, we need to find Rotation Matrix R and Translation Vector T.

When each camera is looking at the same object, the object has an image in the camera planes. Those images can be represented with respect to the rotation matrix and translation vector of each camera.

$$P_l = R_l P + T_l \quad (3)$$

$$P_r = R_r P + T_r \quad (4)$$

P_l and P_r also can be found with rotation matrix and translation vector between the cameras

$$P_l = R^T (P_r - T) \quad (5)$$

When these three equations are solved together. We can get the expression of R and T

$$R = R_l R_r \quad (6)$$

$$T = T_r - (R\, T_l) \quad (7)$$

After having R and T values , next step is calculating Essential and Fundamental Matrices E and F. Essential Matrices indicates the coordinate relation between P_l and P_r

$$P_r^T E P_l = 0$$

$$\text{where } E = RS \quad (8)$$

S is the cross product of T and P_l

$$S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}$$

F also indicates the coordinate relation but in term of pixels

$$q_r F q_l = 0$$

$$\text{Where } F = (M_r^{-1})^T E M_l^{-1} \quad (9)$$

q_r and q_l are the pixel coordinates of the object images on both camera lenses M_l and M_r express the inside matrix of each camera.

With E and F values, information of stereo camera geometry can be processed. In the next step, stereo cameras will be aligned

Each camera is set as their epipolar lines are horizontal against each other. It allows stereo camera calibration algorithm to search for one pixel point in one camera, along with the same point in another camera. This process is called **Rectification.**

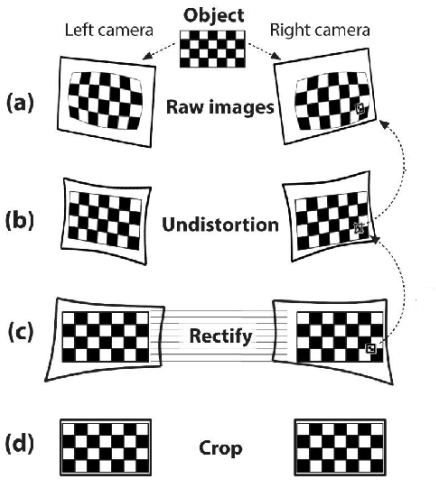


Figure 13: Stereo Rectification with a chessboard

When raw images are processed in the Rectification step to have a perfectly aligned image, now Depth information can be obtained by using eq. (2). However, first, **Correspondence** step has to be completed to have better results.

In Correspondence step, rectified images are checked again in case of any problem occurs due to the movement of the camera, object or elapsed time. This process can be done by applying an algorithm like Sum of Absolute Differences(SAD) to observe that if a part of the image in one camera locates in the same coordinate in the other camera. This process is also repeated for other cases like the movement of objects in the image or changing medium conditions.

As the last step, we can obtain depth information from the stereo image. However, the stereo image is a 2D image and we need to transfer 2D coordinates to 3D coordinate. As

we discussed in chapter 2.3.1, the difference of object images on both camera lenses gives us the disparity value (d). We have seen in eq. (2) , the distance between the object and camera pair can be calculated with this disparity value. We can adapt that equation to calculate other dimensions in 3D space.

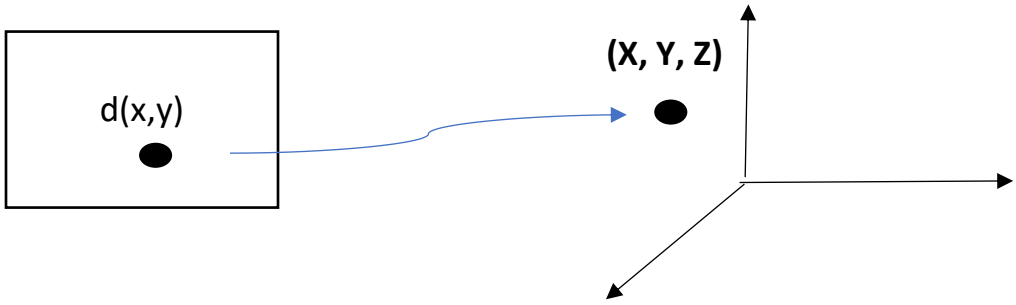


Figure 14: Coordinate transfer from 2D to 3D

$$X = Z\frac{X_R}{f} \qquad (10)$$

$$Y = Z\frac{Y_R}{f} \qquad (11)$$

With eq. (10) and (11) 2D coordinates can be transferred to 3D space and depth map can be obtained from the disparity map.

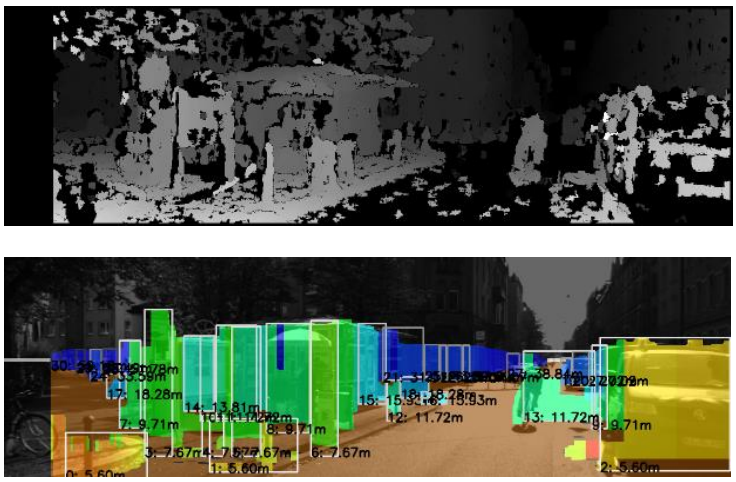


Figure 15: Disparity Map and Depth Map(with
(Bounding Box around objects)

3 Object Detection

Basically, Object Detection is trying to find specific objects of interest (pedestrian, car, animal, traffic lights)within the image or video sequences. It has become more popular and important with the fully autonomous driving idea.

In this project, STIXEL method was used [10] as the basis of the project. However, object detection with Deep Learning is another popular and efficient method, therefore in the next chapter different deep learning methods will be explained as a related work briefly.

3.1 Object Detection with Deep Learning

Deep learning is a sub-class of Machine Learning which an algorithm that can educate itself. It was first introduced by Rina Dechter in 1986 [11]. The algorithm consists of different layers which input layer, hidden layer and output layer. There might be more than one hidden layer to process input data. The main purpose of deep learning is to estimate the outcome of input data of specific interests like defining objects, weather forecast, ticket prices.

Many different deep learning algorithms have been developed by the years for object detection and tracking. Most of these algorithms are Convolutional Neural Network-based algorithms.

3.1.1 Convolutional Neural Network

CNN(Convolutional Neural Network) is one of the main algorithms for computer vision. CNN takes the input image and classifies the image by giving a decision of what kind of object the image has. CNN gives this decision by passing the image through its layers. It has 5 different layers and at the end of the process CNN gives a decision on a scale from 0 to 1

Layers of CNN

-Convolutional Layer

-Non-Linear Layer

-Pooling(Downsampling) Layer

-Flattering Layer

-Fully-Connected Layer

In Figure 16 [12] the complete CNN architecture can be seen

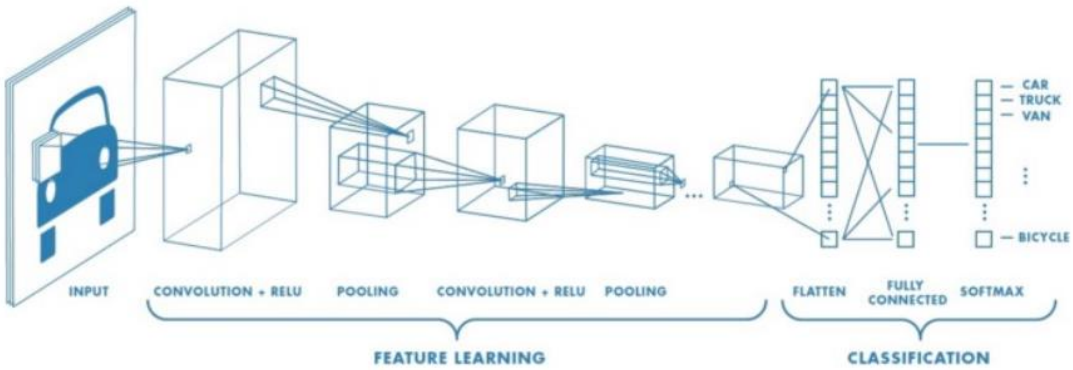


Figure 16: Architecture of CNN

3.1.1.1 Convolutional Layer

Convolution is the first layer of CNN. In this layer, the features of the image are determined by multiplying input image and a filter matrix.

In computer science, each image is formed of matrices and layers. For example, 5x5x3 resolution of an image expresses the dimensions of an image as height x width x dimension. In our example, the height and width of the image are 5 which means our image is 5x5 matrix. The dimension expresses the channel of the image. 3 dimension means the example image has 3 channel as RGB. When an image is in grayscale then the dimension will be 1.

In convolution, an image is multiplied by a smaller filter in order to have featured map.

5x5 Input Image=

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

3x3 Filter Matrix =

1	0	1
0	1	0
1	0	1

Multiplying Input and Filter Matrix =

1	1	1	0	0
0	1	1	1	0
0	0	1_{x1}	1_{x0}	1_{x1}
0	0	1_{x0}	1_{x1}	0_{x0}
0	1	1_{x1}	0_{x0}	0_{x1}

Convolved Feature map =

4	3	4
2	4	3
2	3	4

There are many types of filters like edge detection, gaussian blur. Those filter can also be applied to the same image in series. That means there might be more than the convolutional layer in a CNN architecture. However as it can be noticed above, the dimension of the convolved feature map is smaller than the dimension of the input image. In order to keep the original dimension of the input image, zeros will be added to the feature map. This process is called **Padding**

Padding =

0	0	0	0	0
0	4	3	4	0
0	2	4	3	0
0	2	3	4	0
0	0	0	0	0

3.1.1.2 Non-Linearity Layer

After Convolutional Layer, Non-Linear Layer comes as a second step. In this step, An activation function takes the padded feature map and creates an activation map.

ReLU(Rectified Linear Units) is the most [13] efficient activation function. The simple of ReLU is

$$Y_i^{(l)} = \max(0, Y_i^{(l-1)}) \qquad (12)$$

Figure 17: Original Image, Feature Map and Activation Map

After ReLU is applied to Feature Map, black pixels are changed with 0 in Activation Map

3.1.1.3 Pooling Layer

Pooling Layer is put between consecutive convolutional layers in order to decrease the number of calculation steps. It basically picks the highest value from a small matrix in the whole image. By this process, a small image can be obtained from the original image as the features of the original image are protected. There are many types of Pooling process but the most popular one is Max Pooling

-Max Pooling

-Average Pooling

-Sum Pooling

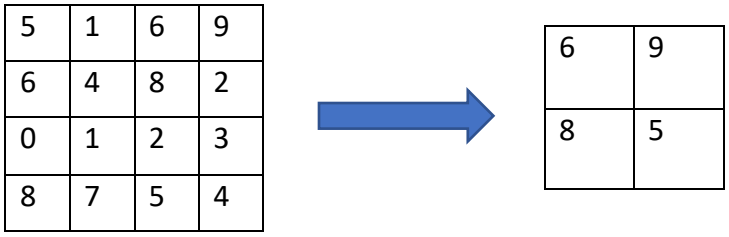


Figure 18: Basic Example of Max Pooling

3.1.1.4 Flattering Layer

The purpose of Flattering Layer is preparing the data the comes from Convolutional Layer for Fully Connected Layer. Neural Network process the data in 1D. Therefore, the matrixes in Convolutional Later is changed to a 1D vector in Flattering Layer

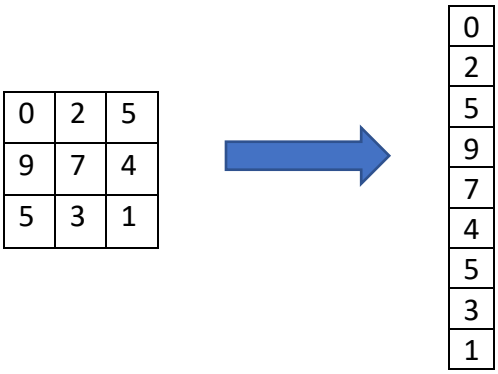


Figure 19: Flattering

3.1.1.5 Fully-Connected Layer

Fully Connected Layer is the layer that performs classification. Fully connected means connection all every node that comes to this layer at the second layer. Then, based on the training of the algorithm, at the end of Fully-Connected later CNN predicts the feature of the image by giving it a value in a scale 0-1

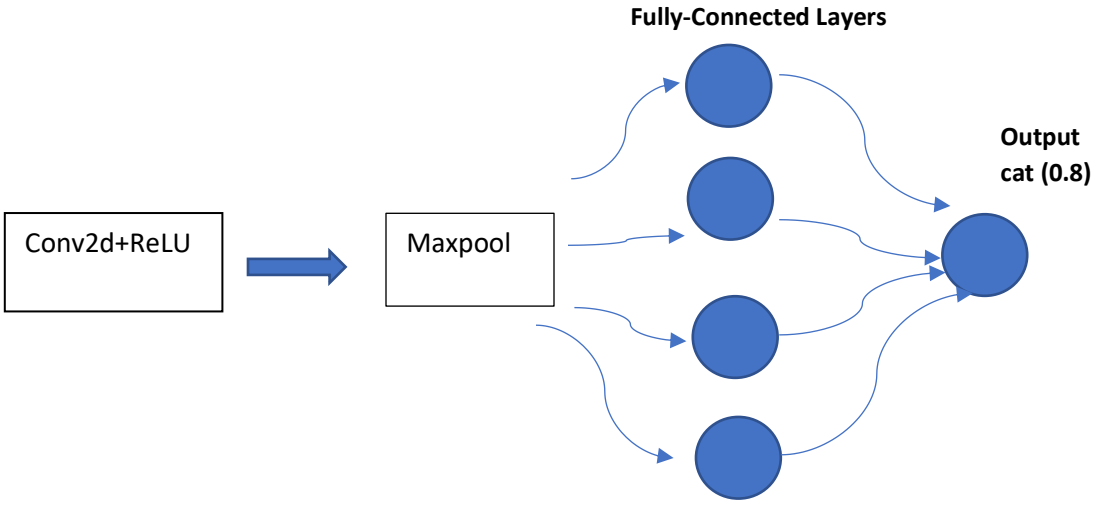


Figure 20: Basic Fully-Connected Layer schema

3.1.2 Region-CNN

In object detection, the purpose is drawing a bounding box around the object that we want to detect. However, there might be multiple objects in the image with bound boxes around them which you can not know before. In this case, the length of the output layer of CNN will not be constant. We can solve that problem by taking different regions from the image and then applying CNN for each region to detect the presence of the desired object. However, the location and the dimension of the desired objects in the image can not be known beforehand so, we might have to select a huge number of regions in the image. This process takes too much time even for a basic object detection operation.

Region-CNN(RCNN) was introduced in 2014 by Ross Girschick [14] to solve this problem. His approach is to select just 2000 regions from the image. However, these regions are selected by using a special algorithm is called **Selective Search.** He also called the regions selected by Selective Search as **Region Proposals**

These 2000 region proposals are sent to CNN and CNN generates a 4,096 element vector which shows the contents of the image. In this step, CNN is used as a feature producer and the output of the CNN is sent to Support Vector Machine(SVM) for classification. Each SVM is trained for one class which means there must be SVM as many as a class number.

The main problem with RCNN is the process is still too slow and It can not be used in a real-time system. Furthermore, Selective Search algorithm is a fixed algorithm which means there is not a learning step in that stage

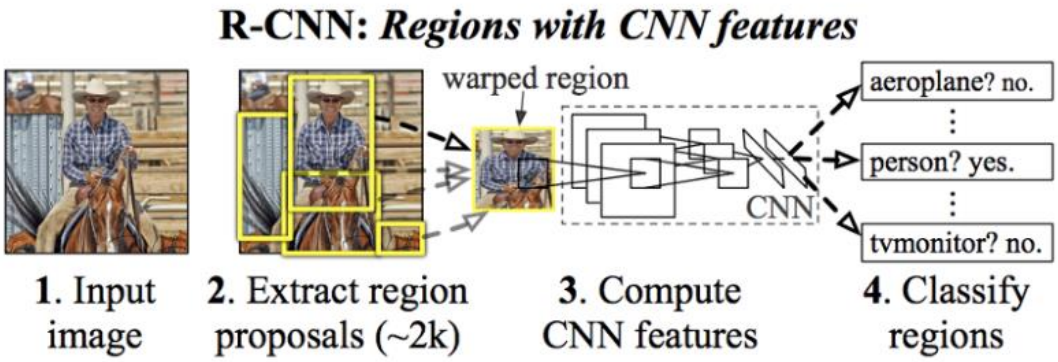


Figure 21 [14]: RCNN

3.1.3 FAST R-CNN

Ross Girsch has improved his own project R-CNN and introduced Fast R-CNN in 2015 [15]. In Fast R-CNN, an input image is sent to CNN process directly instead of creating 2000 region proposals. Region proposals are generated after Convolutional Layer when feature map is obtained and then, by using RoI pooling layer the size of the region proposals get fixed and region proposals are sent to Fully-Connected Layer. As the last step, the softmax layer in RoI feature vector is used to classify the proposed region.

Fast R-CNN is faster than R-CNN because convolutional layer step is completed only one time for each image.

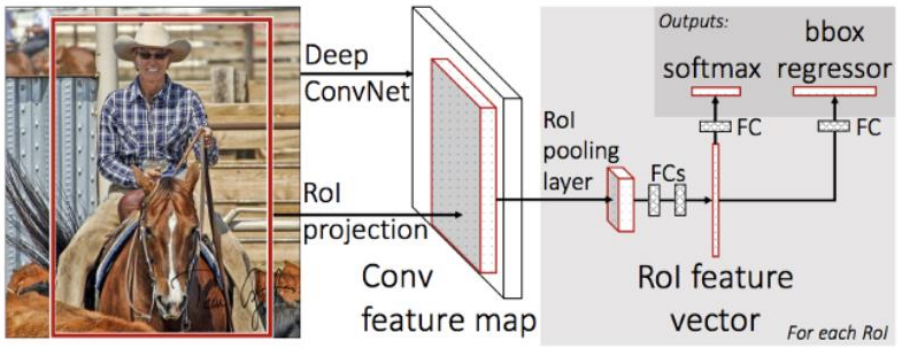


Figure 22 [15]: Fast R-CNN

3.1.4 FASTER R-CNN

Although R-CNN and Fast R-CNN are creating region proposals in different steps, both of them still use selective search. Selective search as has been introduced in chapter 3.1.2 is a slow and fixed algorithm. Instead of selective search, Shaoging Ren developed Regional Proposal Network(RPN) and introduce his research at 2015 [16]. Unlike selective search, Regional Proposal Network is part of the training process in the architecture of Faster R-CNN. This improvement decreased the region proposal numbers and accelerated testing time in order to use the system almost in real-time.

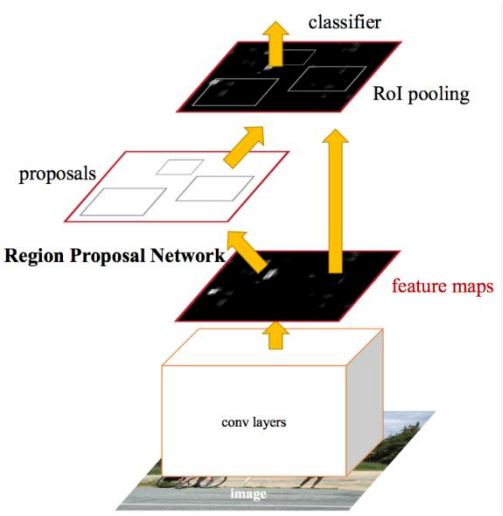


Figure 23 [16]: Faster R-CNN

3.1.5 YOLO

Yolo was introduced in 2015 by Joseph Redmon [17]. Previous CNN based object detection algorithms look at regional proposal areas but not all the image.

YOLO splits input image into bounding boxes and with single neural network predicts the class of bounding boxes. Yolo is less successful in terms of accuracy (localization error) but faster than other object detection algorithms that were discussed in previous chapters.

In 2016 Joseph Redmon improved his technic and release his research “YOLO9000: Better, Faster,Stronger” [18]. In 2018 Joseph Redmon updated his research again and introduced [19] “YOLOv3: An increment Improvement”

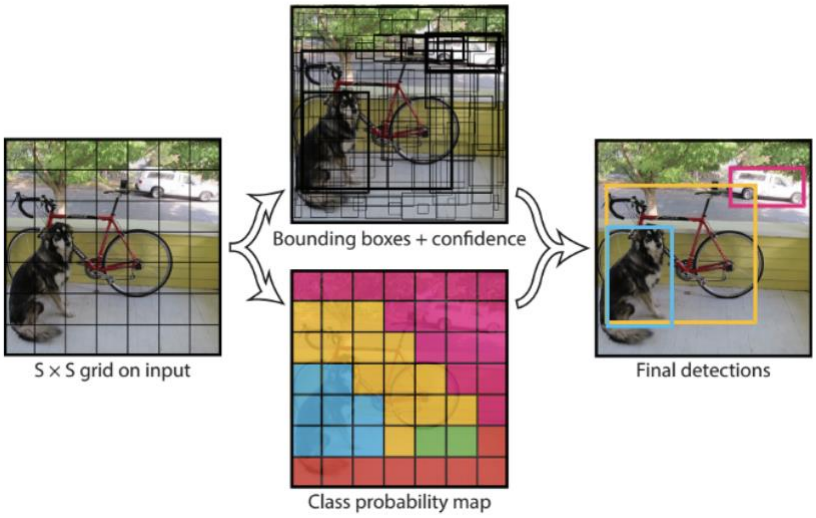


Figure 24 [17]: YOLO

3.2 STIXEL

Stereo vision has started to play an essential role in automotive industry in advanced ADAS systems and automotive driving. In 2005, H. Hirschmuller published his research “Accurate and efficient stereo processing by semi-global matching and mutual information” [20]. SGM (semi-global matching) algorithm provides accurate object boundaries. Today, most of the stereo vision algorithms are SGM based algorithms.However, SGM is really complex algorithm and needs a really big memory in CPU. Furthermore, the resolution quality of cameras in ADAS systems are getting higher every day. This causes more calculation in a simple object detection process. For example, in order to obtain disparity map from 1024x440 image pair, over 400,000 single disparity calculation is needed.

As a solution to this problem, David Pfeiffer, Hernan Badiono and Uwe Franke released their research “The Stixel World – A Compact Medium Level Representation of the 3D-World” [21]. Basically stixel means, rectangular sticks are made of pixels. If we assume

one stixel is made of 5 pixels then , we have 5 times less number disparity calculation than number of disparity calculation for each pixel.

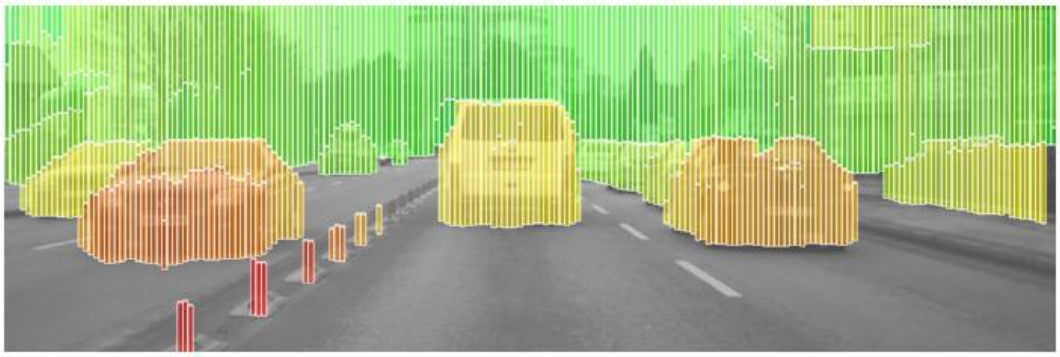


Figure 25 [21]: Stixel -World

3.2.1 Building Stixel World

As it can be seen in Figure 25. A stixel has two important parameters, its height and the distance from the car. In order to build such a stixel system as in Fig. 25. There is cascade system includes 5 steps. These steps are

-Dense Stereo

-Occupancy Grid

-Free Space Computation

-Height Segmentation

-Stixel Extraction.

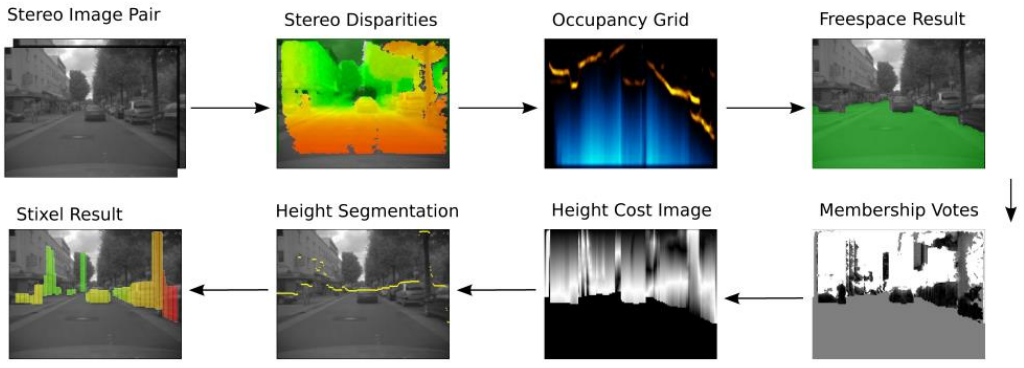


Figure 26 [21]: Stixel Steps

3.2.1.1 Dense Stereo

SGM provides dense stereo scheme from disparity map. SGM can be run on a normal CPU within a few seconds. Also "Gravitational Constraint" has been introduced at 2007 [22] in order to increase accuracy of disparities. However, for working in real-time, these implementation must be run in an FPGA system. A dense stereo can be seen in Figure 27

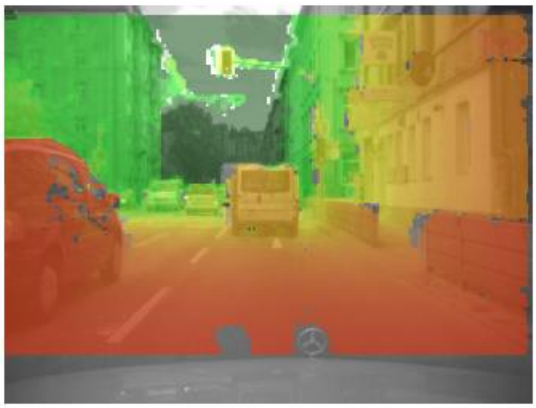


Figure 27 [10]: Dense Disparity Image

3.2.1.2 Occupancy Grid

Occupancy Grid is a 2D array shows the potential obstacle in the environment. It was first introduced in 1987 by A. Elfes [23]. Occupancy Grid is used to extract free-space data in Stixel process.

For occupancy grid, our measurement model is [24]

$$p(z_t|m, l_t) \tag{13}$$

Which means, probability of observing z_t in the map m and the exact location l_t.

For the first step to solving mapping problem, we split our image into cells. Each cell might have two states 1:occupied or 0:empty. Vector of the cells is m and each cell is named as m_i. The standard approach to occupancy grid based on multiple cells

$$p(m_i|z_{1:t}, x_{1:t}) \quad (14)$$

There are 2^m possible state. We can decrease the complexity by filter each cell assuming each filter is independent

$$p(m|z_{1:t}, x_{1:t}) = \prod_i p(m_i|z_{1:t}, x_{1:t}) \quad (15)$$

The equation above can be solved with a binary Bayes filter to predict occupancy grid

3.2.1.3 Free Space Computation

The purpose of free space computation is finding the first obstacle in searching direction. Searching starts from the bottom of the image and it goes in a vertical direction until an obstacle is detected. The area between the start point and obstacle is called free space. In stixel-world research [10] DP(dynamic programming) was used instead of determining a threshold for each column independently.

Sometimes, there might be two different obstacles in the same direction. In this case only first obstacle in that column is taken into account and second obstacle is not recognized.

The output of DP gives a vector coordinates (u, d_u). "u" is a column of the image and d_u is the disparity from the image to the first obstacle. For every vector coordinates (u, d_u), a triangulated coordinate (x_u, z_u) is determined. Collection of these triangulated coordinates (x_u, z_u) with the origin (0,0) gives the total free space area from the camera point of view.

3.2.1.4 Height Segmentation

In height segmentation, the upper boundary of obstacles is obtained. When free space is eliminated from the image, we look at two different disparity in the same stixel column for finding obstacles upper limit.

We discussed in the last chapter vector coordinates of obstacle (u, d_u) and its triangle coordinate (x_u, z_u). The purpose in this step is finding row position v_t which means upper boundary of the obstacle at (x_u, z_u).

In stixel-world approach [10], every disparity d(u,v) votes for its membership to the object. If this vote is positive, that means the disparity belongs to foreground object, If it is negative then object of that disparity at background.

If we use a threshold to decide where the object is, we might make a mistake in case of a sensitive situation.

Instead of setting a constant threshold, a Boolean membership in a continuous variation with an exponential function is better solution

$$M_{u,v}(d) = 2^{(1-(\frac{d-d_u}{\Delta D_u})^2)} - 1 \qquad (16)$$

ΔD_u is computed parameter, d_u is disparity and u is the column where we make our calculation. Calculate ΔD_u for each independent column

$$\Delta D_u = d_u - f_d(z_u + \Delta Z_u)$$

Where,

$$f_d(z) = b * f_x/z \qquad (17)$$

$f_d(z)$ is the disparity according to depth z. b corresponds to baseline, fx is focal length and ΔZ_u is a parameter. This equation gives us a chance to define the membership in meters instead of pixels.

Cost of the image is computed according to the membership values

$$C(u,v) = \sum_{i=0}^{i=v-1} M_{u,v}\big(d(u,i)\big) - \sum_{i=v}^{i=v_f} M_{u,v}(d(u,i)) \qquad (18)$$

v_f is the row position at the above equation. For the computation of the optimal path, $G_{hs}(V_{hs}, E_{hs})$ is created. V_{hs} presents vertices and E_{hs} presents edges.

An example of a data and a smoothness term according to minimized cost

$$c_{u,v_0,v_1} = C(u, v_0) + S(u, v_0, v_1) \tag{19}$$

Eq (19) is the cost of the edge connection of vertices V_{v,v_0} and V_{u+1,v_1} and C(u,v) is a data term as in eq (18).

$S(u, v_0, v_1)$ is for smoothness and jump cost in the vertical direction. Its define is

$$S(u, v_0, v_1) = C_s|v_0 - v_1| \cdot \max(0.1 - \frac{|z_u - z_{u+1}|}{N_Z}) \tag{20}$$

C_S is the cost of the jump. C_S becomes 0 when the difference in depth between columns is equal or larger than N_Z. It reaches a maximum when the difference in depth between columns is 0.

3.2.1.5 Stixel Extraction

When we have free space computation and height segmentation, obtaining the stixel is so easy. At free space computation, we have erased the area that has no obstacle or any object. In height segmentation we have found the upper boundaries of every object in the image. As the last step, we cut the stixel from the point that the depth changes in the same pixel stick with the information we have obtained in height segmentation. This process provides us the top point v_T of the object. For the bottom point v_B , we choose the first point that intersects free space in the ground.

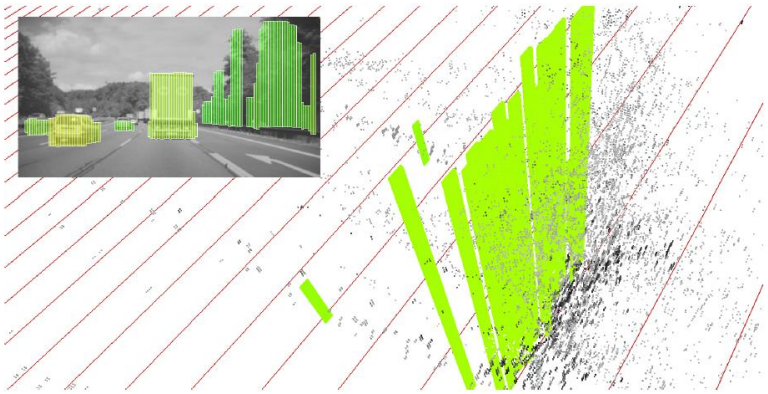


Figure 28 [10]: Stixel

3.2.2 Stixel in the project

In this master thesis, open-source stixel code of Inha University was used [25]. The code has 5 different steps. The first step is stereo matching. In this step, the code creates disparity map of the stereo image. After creating the disparity map, the ground is removed by applying v-disparity method. Then, in the second step which is stixel estimation upper boundaries of all objects are determined and the information of sky is removed from the object and stixel is created. Stixel segmentation is the third part. When we eliminate the ground and the sky and also determine the object dimension, in third step bounding boxes are drawn around every object. After this step, in fourth step and fifth step we bundle our camera information and stixel image to have depth information.

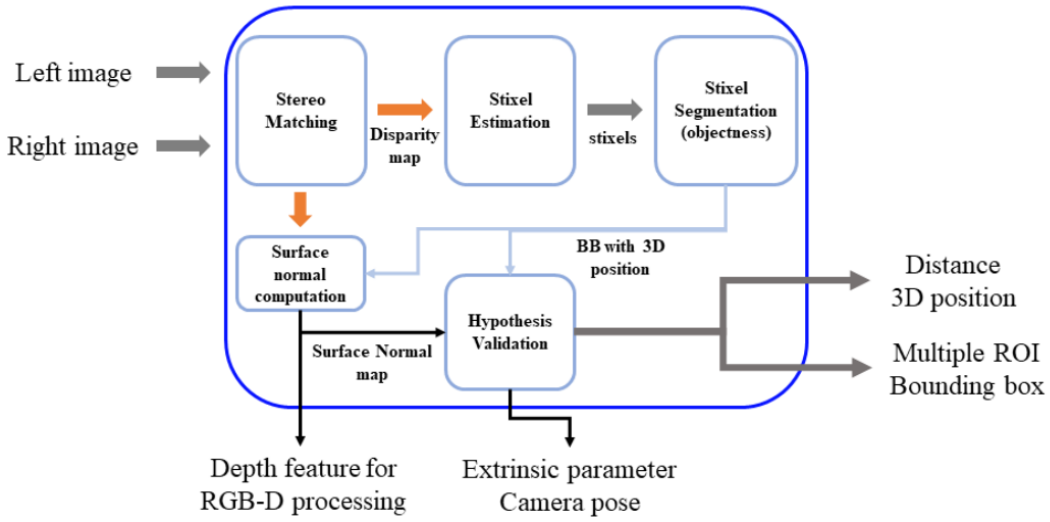


Figure 29 [25].: Stixel Code system design

4 Object Tracking

In computer science, each pixel in a frame is represented by a number depending on its color and each image which is made of pixels is represented by multiple matrixes. In basic understanding, trying to find those matrixes of an image in each frame is the fundamental of the object tracking algorithm.

OpenCV is a widely-used open source library for computer vision and it was used in this project. OpenCV has 8 different tracking algorithm in its own library as a function.

-BOOSTING

-MIL

-KCF

-CSRT

-MedianFlow

-TLD

-MOSSE

-GOTURN

CSRT was used in the project in order to use depth information in the tracking process since CSRT is able to track 3 channel objects(RGB images).

4.1 MOSSE

Minimum Output Sum of Squared Error (MOSSE) was introduced in 2010 by David S. Bolme [26]. The method uses correlation filter for tracking. MOSSE is robust to changes in lighting,pose, and scale. It also detects occasion-based the peak-to-sidelobe ratio which allows tracker to stop and resume when the object disappears and comes back to frame again.

For creating a faster tracker, correlation is calculated in the Fourier domain Fast Fourier Transform(FFT). $F = F(f)$ is 2D Fourier transform of the input image and H = F(h) is 2D Fourier transform of the filter.

$$G = F \odot H^* \quad (21)$$

The eq. (21) is the element-wise multiplication in FFT. According to the Convolution Therom, correlation is element-wise multiplication in the Fourier Domain. Inverse FFT is used to transform the correlation output G back to the spatial domain.

To start training MOSSE we need a set of training image f_i and training output g_i. Training is computed in Fourier domain. As we have seen in eq. (21) F_i and G_i is the input and output images counterparts in Fourier domain

$$H_i^* = G_i / F_i \quad (22)$$

To find a filter H, MOSSE calculates the minimum sum of error between the desired output and actual output that is obtained by applying filter H.

$$\min H^* \sum_i |F_i \odot H^* - G_i|^2 \quad (23)$$

The problem in the equation is the optimization. It is assumed that target is always centered in f_i and output g_i is fixed always. In order to solve this problem we need to calculate each element in H (w,v) independently. Therefore, the equation is rewritten as includes H indexes w,v.

$$0 = \frac{\partial}{\partial H_{wv}^*} \sum_i |F_{iwv} H_{wv}^* - G_{iwv}|^2 \quad (24)$$

The closed-form of MOSSE filter is found by solving this equation for H^*.

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^*} \quad (25)$$

During tracking the object can change its position scale and also lightning condition can change often. Therefore the filter needs to adapt itself to new conditions. Taking average is used to overcome this problem.

$$H_i^* = \frac{A_i}{B_i} \quad (26)$$

$$A_i = \eta G_i \odot F_i^* + (1 + \eta) A_{i-1} \quad (27)$$

$$B_i = \eta F_i \odot F_i^* + (1 + \eta) B_{i-1} \quad (28)$$

η is learning rate and it is used to make filter adopt to changing in objects appearance

4.2 KCF

Kernelized Correlation Filter (KCF) was published by F. Henriques [27]. F. Henriques improved linear correlation filter by adding kernel method. The first focus of the algorithm is Ridge Regession.

$$\min w \sum_i (f(x_i) - y_i)^2 + \lambda ||w||^2 \quad (29)$$

The eq. (29) is the equation for minimizing the squared error over samples x_i. λ is a regularization parameter.

$$w = (X^T X + \lambda \mathrm{I})^{-1} X^T y \quad (30)$$

Eq. (30) is the close form of the eq. (29). The Fourier transform of the eq. (30) is

$$w = (X^H X + \lambda \mathrm{I})^{-1} X^H y \quad (31)$$

X^H is the Hermitian transpose. And according to Representer Theorem [28], the solution can be shown as the sum of a training set

$$w = \sum_i a_i \varphi(x_i) \tag{32}$$

w is linear combination of the samples, a_i is scalar coefficient and φ is the inner products that can be expressed kernelized function. According to fast kernel regression step in the F. Henriques's research, the final description of the detection [28] is

$$\hat{y}' = \hat{f}(x') = \hat{a} \cdot \hat{k}_{xz} = \frac{\hat{y}}{\hat{k}_{xz} + \lambda} \hat{k}_{xz} \tag{33}$$

Where $\hat{k}_{xz}$ is the row the kernel matrix and $\hat{a}$ is a vector of coefficients that represent the solution in the dual space.

4.3 TLD

Tracking-Learning-Detection(TLD) was introduced by Z. Kalal at 2010 [29]. Tracker algorithms estimate the object position in the next frame in video sequences. They only need an initialization the detect object but in further processes generally, they fail at tracking when the object disappears. Detection-based algorithms find the objects in each frame independently.Actually they do not track any object but estimate the object location again and again. Therefore they do not fail when an object disappears but this kind of algorithms requires offline training for each object to detect and classify.For a better result in long-term object tracking issue, TLD brings three different tasks together. The tracker part fallows the objects in each frame, the detection part localizes every object and corrects the tracker part in case of an object disappears. As last the learning part corrects the detection part and update it during all tracking process.

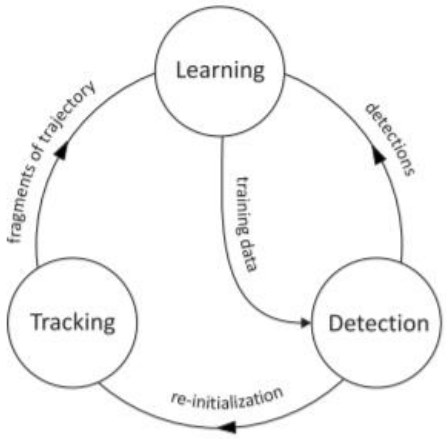


Figure 30 [29]: Block Diagram of TLD

The detector scans the video sequence frame-by-frame and decides if an object exists in the frame. The tracker component of TLD is actually a Median-Flow tracker but the tracker of TLD also has a failure detection. The tracker estimates the displacements of the points within the bounding boxes. It uses median to predict the motion of the object in a grid of 10x10 points.

For failure detection, the algorithm checks if the object still exists in the frame. d_i represents the displacement of a point and d_m is the median displacement so, the single displacement is

$$|d_i - d_m| \tag{34}$$

And the tracker failures if

$$medium|d_i - d_m| > 10\ pixels \tag{35}$$

When failures detected, the tracker stops to return the bounding box to the frame.

The learning part of TLD uses PN learning framework. PN learning has two experts P and N. P-experts represents false-negative and N-experts represents false positive.

x is an example from feature-space X and y is a label of space of labels $Y = \{-1, 1\}$. X is called unlabeled set and Y is labeled set and $L = \{(x, y)\}$ is a labeled set. The inputs of PN-Learning is labeled set L_i and unlabeled set X_u. The purpose of PN-learning is to learn a classifier f. f is a function of family F which is parameterized by Θ.

PN-learning 4 different blocks.

-Classifier

-Training Set

-Supervised Training

-P-N experts

The training process starts by adding labeled set L to the training set. The training set is moved to supervised learning. Then, the next step is iterative bootstrapping. The Classifier classify unlabeled set in previous steps

$$y_u^k = f(x_u|\Theta^{k-1}) \qquad (36)$$

k is an iterative number at above eq. (36). P-N experts estimate incorrect examples by analyzing the classification. The process keeps going by retraining the classifier until reaching convergence.

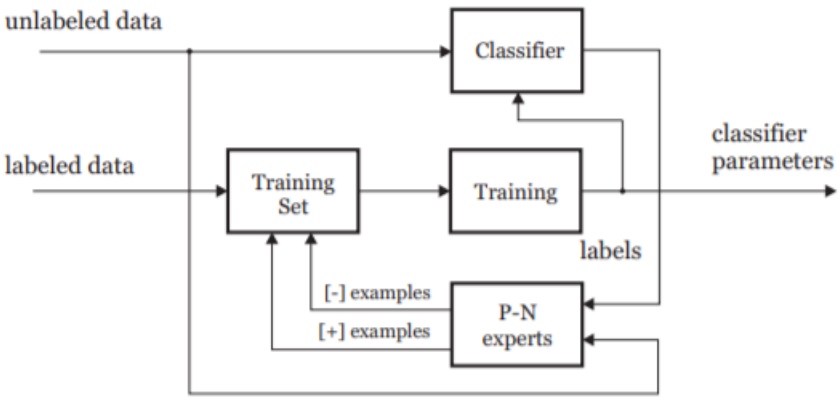


Figure 31 [29]: PN learning block diagram

4.4 Median-Flow

Median-flow was introduced by Z. Kalal, Mikolajczyk K. [30]. The method is based on error calculation algorithm is called forward-backward consistency. The idea behind the algorithm is tracking should be in both directions of time-flow. Initially, the tracker produces a trajectory forward in time and then a validation trajectory is created by backward tracking from last frame to the first frame. When these two trajectories are compared with each other, in the case of there is big difference, the first trajectory which is forward in time is considered the wrong trajectory.

$S = (I_t, I_{t+1}, I_{t+2}, \dots., I_{t+k})$ is an image sequence and x_t is a point in the image in time t. When x is tracked by k steps forward in time, the forward trajectory is $T_f^k = (x_t, x_{t+1}, x_{t+2}, \dots., x_{t+k})$. In order to create validation trajectory, x_{t+k} is tracked backward in time and validation trajectory occurs as $T_b^k = (\widehat{x_t}, \widehat{x_{t+1}}, \widehat{x_{t+2}}, \dots., \widehat{x_{t+k}})$. The differences between two trajectories are

$$FB(T_f^k|S) = distance(T_f^k, T_b^k) \tag{37}$$

In median-flow tracker, each bounding box has a rectangular point grid. Each point is tracked by Lucas-Kanade tracker. With forward-backward consistency method, these points are predicted and %50 of the worst predictions are eliminated. The remain %50 of prediction is used to estimate for localization of object in next frame. This method is referred as Median-Flow Tracker.

4.5 Boosting

Boosting method was introduced in the research of H. Grabner [31] in 2006. The tracker is based on AdaBoost tracking algorithm. The objects are separated from the background to be tracked. The main advantage of the method is it is able to on-line training. This capability gives the method a chance to adapt itself for classification during the tracking. Therefore the difficulties like changing in environment like or object visibility are handled easily.

The desired object is considered as a positive sample. In order to create negative sample, random regions of the same size as the desired object are chosen from the background in the image. These samples are used to make iterations in on-line boosting algorithm.

On line boosting algorithm has three different elements weak classifier, selector and strong classifier. The weak classifier is almost better than random estimate. h^{weak} is generated by a weak classifier. The selector selects a weak classifier from the hypothesis $H^{weak} = \{h_1^{weak}, h_2^{weak}, \ldots, h_m^{weak}\}$.

$$h^{sel}(x) = h_m^{weak}(x) \tag{38}$$

With a linear combination of selectors, a strong classifier is generated from a set of weak classifiers.

$$hStrong(x) = sign(conf(x)) \tag{39}$$

$$conf(x) = \sum_{n=1}^{N} \alpha_n \cdot h_n^{sel}(x) \tag{40}$$

Where $conf(x)$ is a confidence measure of the strong classifier. The main step of on-line boosting is generating selectors. With each weak classifier, selectors are updated again and again. The best selector is the one detects the error of weak classifier from samples seen so far.

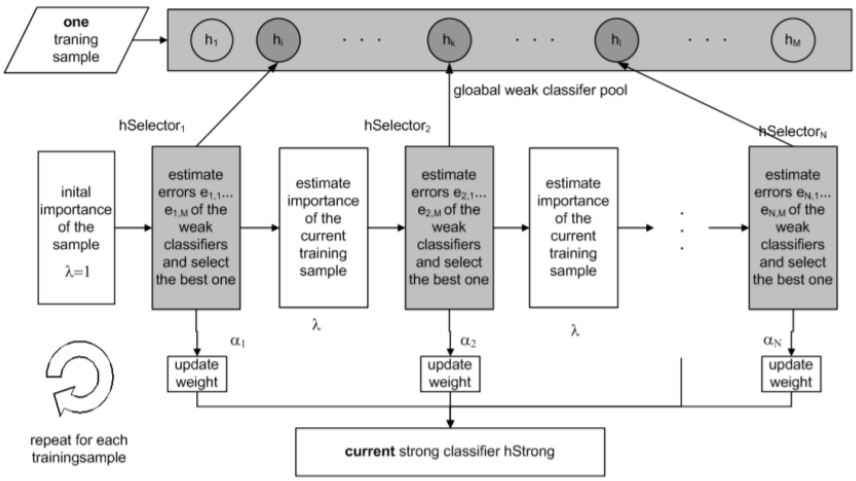


Figure 32 [31]: On-line boosting diagram

4.6 MIL

Multiple Instance Learning(MIL) was published by B.Bobenko in 2009 [32]. It can be said that MIL is an improved version Boosting method. In boosting method, the algorithm splits the object from the background by labeling the object as positive and labeling the rest of the image as negative. Instead of having exact positive and negative regions. there is also potential positive regions. The neighborhood regions around the object are labeled also positive. This might cause a decrement in detection exact object center however it increases the area that tracker tries to detect the object in it.Furthermore, in MIL algorithm, the object is searched not in one area but in a bag that includes many areas. The algorithm labels these bags as positive or negative not bounding boxes one-by-one.

4.7 GOTURN

Generic Object Tracking Using Regression Network (GOTURN) was introduced D. Held in 2016 [33]. GOTURN is the only one tracker which is CNN based on this list. The main difference between GOTURN and other CNN based tracker is GOTURN is capable of making offline training. Offline training provides GOTURN to teach to handle changes in viewpoint, lighting changes, etc... On the other hand, GOTURN is quite faster than other neural network-based object tracking algorithms. GOTURN is able to track objects at 100 fps... In order to reach that speed, GOTURN uses a regression-based approach instead of classification-based approach.

In GOTURN method, the target object is fed to convolutional layers, the output of these convolutional layers are sent to fully connected layers. The purpose of fully connected layer is to detect objects movement by comparing the features from target object and the features in the current frame. During the image sequence environmental factors may change, the function is learned by fully connected layers is learned through examples to handle these changes.

$$c'_x = c_x + w \cdot \Delta x \quad (41)$$

$$c'_y = c_y + h \cdot \Delta y \quad (42)$$

(c'_x, c'_y) is the center of the bounding box in current frame and (c_x, c_y) is the center of bounding box in the previous frame. The eq. (41) and (42) is for concretizing the idea of

motion smoothness. w is the width and h is height, respectively of the previous frame. Δx and Δy are random variables that represent the change in position of the bounding boxes. These variables can be modeled with a Laplace distribution with a mean of 0.

Another model for size changes

$$w' = w \cdot \gamma_w \quad (43)$$

$$h' = h \cdot \gamma_h \quad (44)$$

Where,w' and h' are the current width and height of the current bounding box and w and h are the previous width and height of the bounding box. γ_w and γ_h represent the change in size of the bounding boxes. These variables can be modeled by a Laplace distribution with a mean of 1.

Cross-validation is used to determine the scale of parameters for the Laplace distributions which are $b_x = 1/5$ (motion of bounding box center) $b_s = 1/15$ (change in bounding box size)

4.8 CSRT

Discriminative Correlation Filter with Channel and Spatial Reliability was introduced by A.Lukezic et al in 2017 [34] . The spatial Reliability map makes filter pick a part of object for tracking. Thanks to this method, two problems are handled which circular shift enabling an arbitrary search and the restriction to the rectangular shape of objects. The spatial reliability map is generated by solving the graph labeling problem in each frame. CSRT uses 2 standard features of HoGs and Color names. It is considerably slow(25 fps) but accurate.

For Filter Learning,$f = \{f_d\}_{d=1:N_d}$ is a set of the feature of a N_d channel set and $h = \{h_d\}_{d=1:N_d}$ is filter where

$f_d \in \mathcal{R}^{d_w \, x \, d_h}$, $h_d \in \mathcal{R}^{d_w \, x \, d_h}$, the maximum possibility of object position is

$$p(x|h) = \sum_{d=1}^{N_d} p(x|f_d)p(f_d) \quad (45)$$

$p(x|f_d) = [f_d * f_h](x)$ is a convolution of feature map with learned template evaluated at x and $p(f_d)$ is a prior reflecting channel reliability.

Minimizing the sum of squared differences between the channel-wise correlation outputs and the desired output $g \epsilon \mathcal{R}^{d_w \, x \, d_h}$,

$$arg\ min_h \sum_{d=1}^{N_d} ||f_d * h_d - g||^2 + \lambda \sum_{d=1}^{N_d} ||h_d||^2 \qquad (46)$$

The eq. (46) is used for obtaining optimal filter at the learning stage. Minimizing Eq. (46) by equating the complex gradient of w.r.t each channel to zero suffers from boundary defects. In order to solve this problem, spatial reliability map is proposed.

Spatial reliability map $m \epsilon [0,1]^{d_w \, x \, d_h}$ with elements, $m \epsilon \{0,1\}$ represents the learning ability of each pixel

$$p(m = 1|y,x) \propto p(y|m = 1,x)p(x|m = 1)p(m = 1) \qquad (47)$$

The eq. (47) is the probability of pixel x depending on the appearance of y. The probability of appearance of y $p(y|m = 1,x)$ is calculated by Bayes rule from the objects color model which is stable during tracking as color histograms $c = \{c^f, c^b\}$.

Pixels away from the center might have object or background information. For this probability, a weak spatial prior is defined as

$$p(x|m = 1) = k(x;\boldsymbol{\sigma}) \qquad (48)$$

where $k(x;\boldsymbol{\sigma})$ is a modified Epanechnikov kernel and $\boldsymbol{\sigma}$ is minor bounding box axis.

Spatial reliability map determines pixel that must ignored in the filter. A constraint

$h \equiv m \odot h$ is closed for solution of eq. (46) as optimized contrained.

h_c is a dual variable and the constraint is $h_c - m \odot h \equiv 0$

$$\mathcal{L}(\widehat{h_c}, h, \hat{I}|m) = ||\widehat{h_c^H}, diag(\hat{f}) - \hat{g}||^2 + \frac{\lambda}{2}||h_m||^2 + [\widehat{I^H}(\widehat{h_c} - \widehat{h_m}) + \overline{\widehat{I^H}(\widehat{h_c} - \widehat{h_m})}] + \mu||\widehat{h_c} - \widehat{h_m}||^2 \quad (49)$$

Eq. (49) is the augmented Lagrangian equation relates to the constraint. $\hat{I}$ is a Lagrange multiplier. Eq (49) can be minimized by the alternating method, that solves following problems at each iteration

$$\widehat{h_c^{i+1}} = \underset{h_c}{arg\,min}\ \mathcal{L}(\widehat{h_c}, h^i, \widehat{I^i}|m) \quad (50)$$

$$h^{i+1} = \underset{h}{arg\,min}\ \mathcal{L}(\widehat{h_c^{i+1}}, h, \widehat{I^i}|m) \quad (51)$$

The Lagrange multiplier is updated as

$$\widehat{I^{i+1}} = \widehat{I^i} + \mu(\widehat{h_c^{i+1}} - h^{i+1}) \quad (52)$$

The minimized eq. of eq. (50) is

$$\widehat{h_c^{i+1}} = (\hat{f} \odot \bar{\hat{g}} + (\mu\widehat{h_m^i} - \widehat{I^i})) \odot^{-1} (\hat{f} \odot \bar{\hat{f}} + \mu^i) \quad (53)$$

$$h^{i+1} = m \odot \mathcal{F}^{-1}[\widehat{I^i} + \mu^i \widehat{h_c^{i+1}}]/(\frac{\lambda}{2D} + \mu^i) \quad (54)$$

Eq. (52) and (53) can are calculated in the frequency domain but for eq. (54) inverse FFT is needed. Another inverse FFT is used for computing $\widehat{h^{i+1}}$. With two inverse FFT, a very fast optimization for Filter learning can be achieved.

The product of learning channel reliability measure and a detection reliability measure gives the channel reliability.

$$f_d * h_d \cong g \qquad (55)$$

f_d is a discriminative feature channel and h_d is a filter. g is an ideal response.However this response has noise on the channel.Therefore, the channel learning reliability $p(f_d)$ in eq. (45) is the maximum response of a learned channel filter.

The detection reliability in CSRT is based on the ratio between the second and first major mode in the response map which is p_{max2}/p_{max1}. However, this ratio has disadvantage when multiple objects appear around the target object. In order to overcome this disadvantage, the ratio is clamped by 0.5. Thus, the per-channel detection reliability is determined as

$$w_d^{(det)} = 1 - min(p_{max2}/p_{max1}, \frac{1}{2}) \qquad (56)$$

The object is detected by summing the responses of the learned correlation filter h_{t-1} weighted by the estimated channel reliability scores. In order to upgrade the tracking, foreground and background histograms $\tilde{c}$ are extracted at the estimated location and updated by an autoregressive scheme with learning rate η_c.

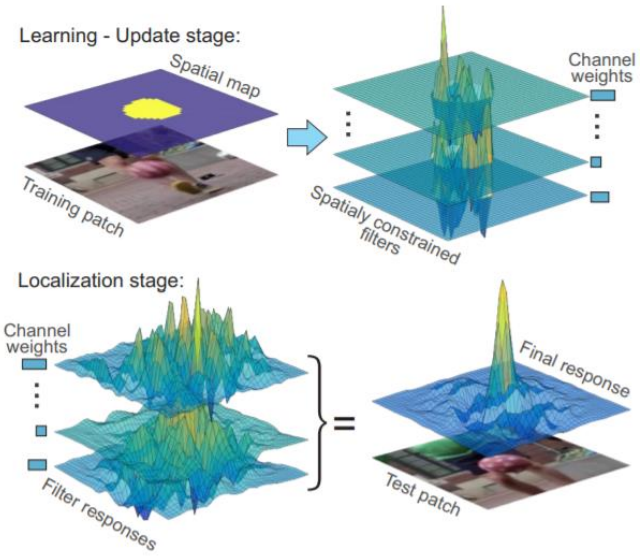


Figure 33 [34]: CSRT Approach

5 Evaluation

5.1 Project Tools

In this master thesis, we used OpenCV open-source library in Ubuntu 16.04 operating system. It is really practical and easy to use a library, especially for Computer Vision. C++ was mostly used as a programming language.

Test videos were taken from KITTI dataset [35]. One of the important features of KITTI dataset is it provides tracklets data of the videos. Tracklets data contains each object positions and classifications frame-by-frame. This allows us to compare our result with real data. MATLAB was used to extract tracklets data.

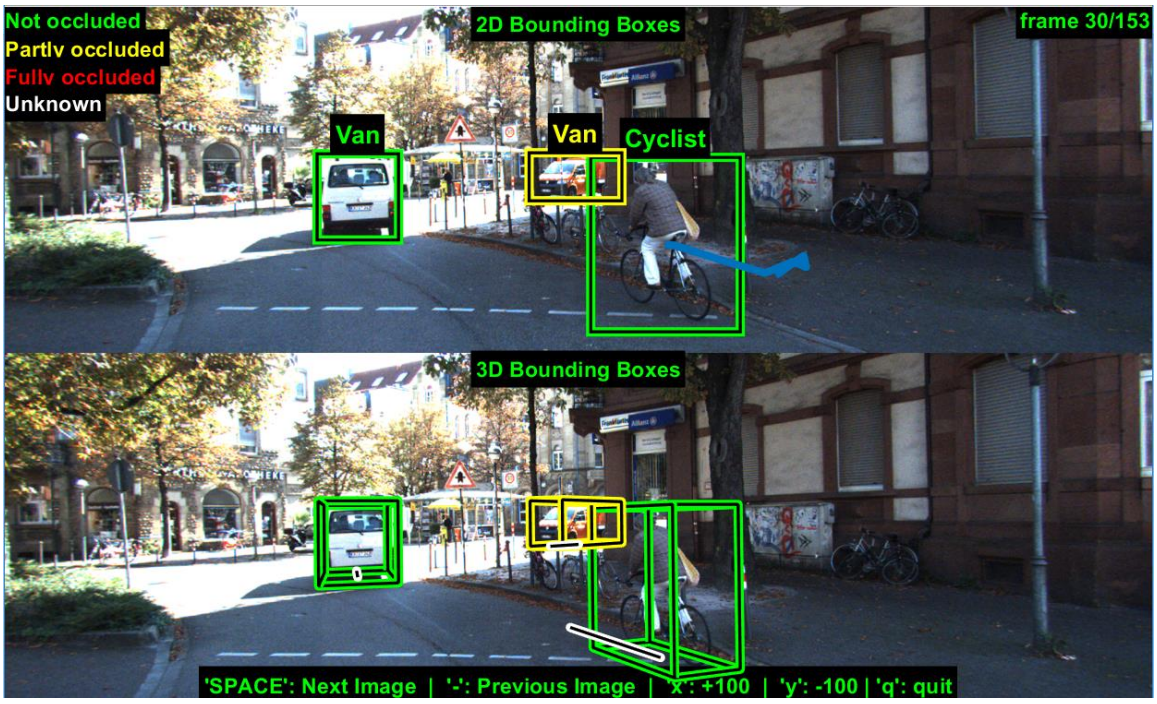


Figure 34: Tracklets Data

5.2 Object Filtering

Stixel method detects not only pedestrians or vehicles on the road but also almost every object in the frame. Since the algorithm takes bundled stixels in same distance as an object, we have lots of bounding box whether they are real objects or not

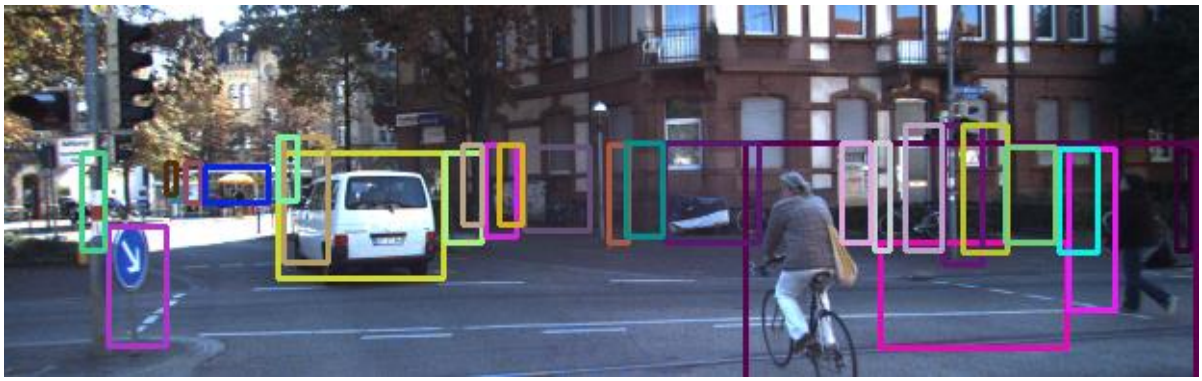


Figure 35: Unfiltered Frame

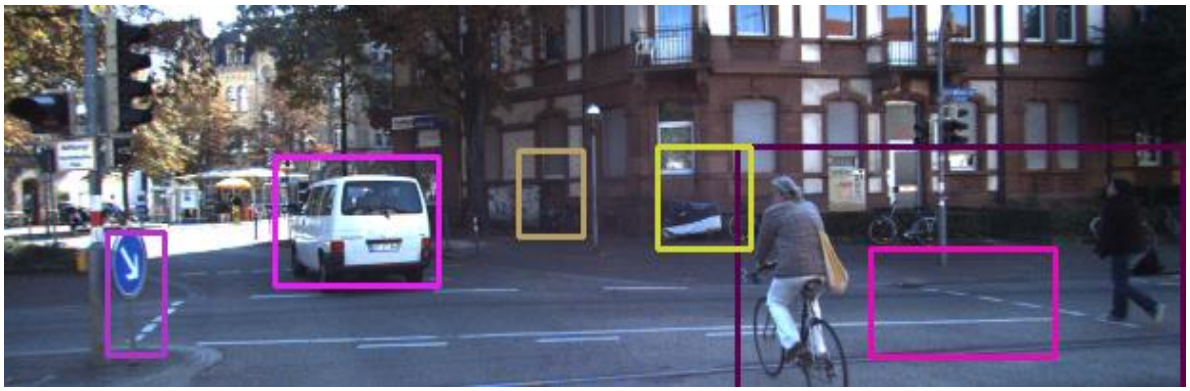


Figure 36: Filtered Frame

5.3 Tracking with Grayscale Image

We started tracking the video sequence in grayscale format. Because we wanted to see the performance of different tracking algorithm for the same object. The reason why grayscale format was selected was not every algorithm is capable of tracking in RGB format.

Figure 37: Tracking with Grayscale Format

In the graphs below, blue lines are the cyclist's paths during the video sequence according to tracklets data and red lines are the paths according to tracking algorithm. The distance graphs of the algorithms show the differences between tracklets data and our data in 2D.

Evaluation time of each tracking data:

BOOSTING = 12006,972 ms

CSRT=11859,612 ms

KCF = 11202,856 ms

MIL = 12069,473 ms

MOSSE=12028,710 ms

TLD = 11986,54 ms

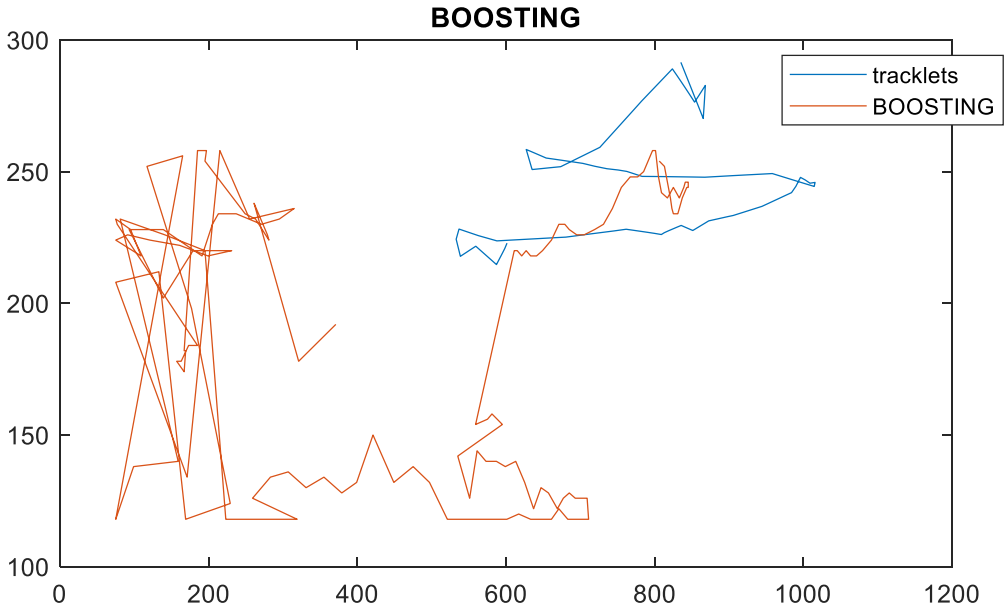


Figure 38: BOOSTING

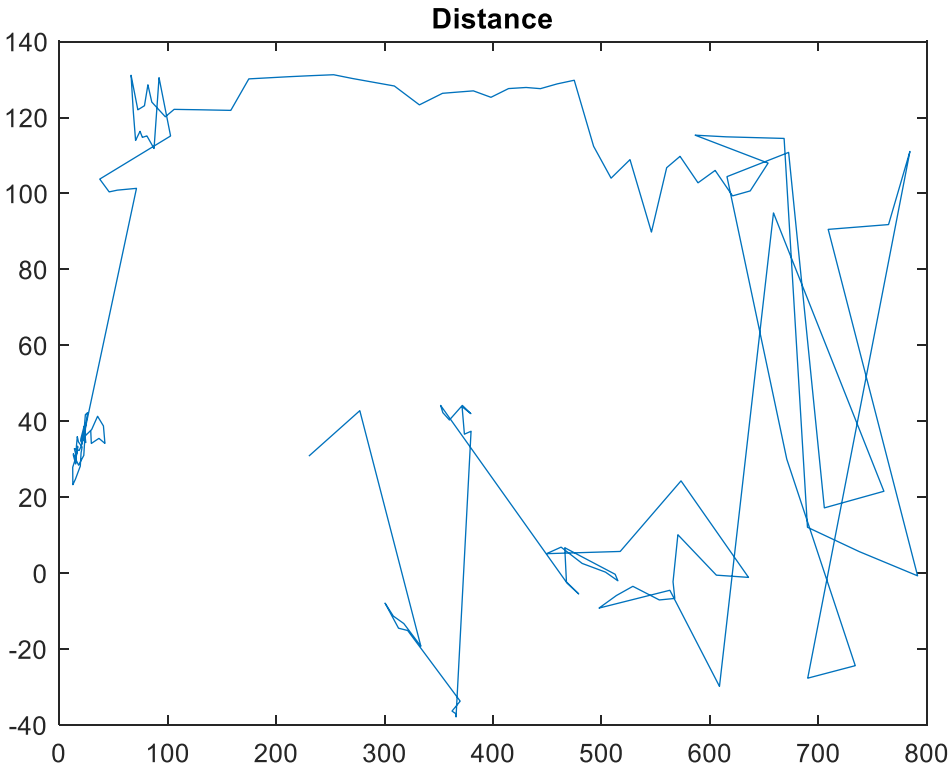


Figure 39: BOOSTING distance

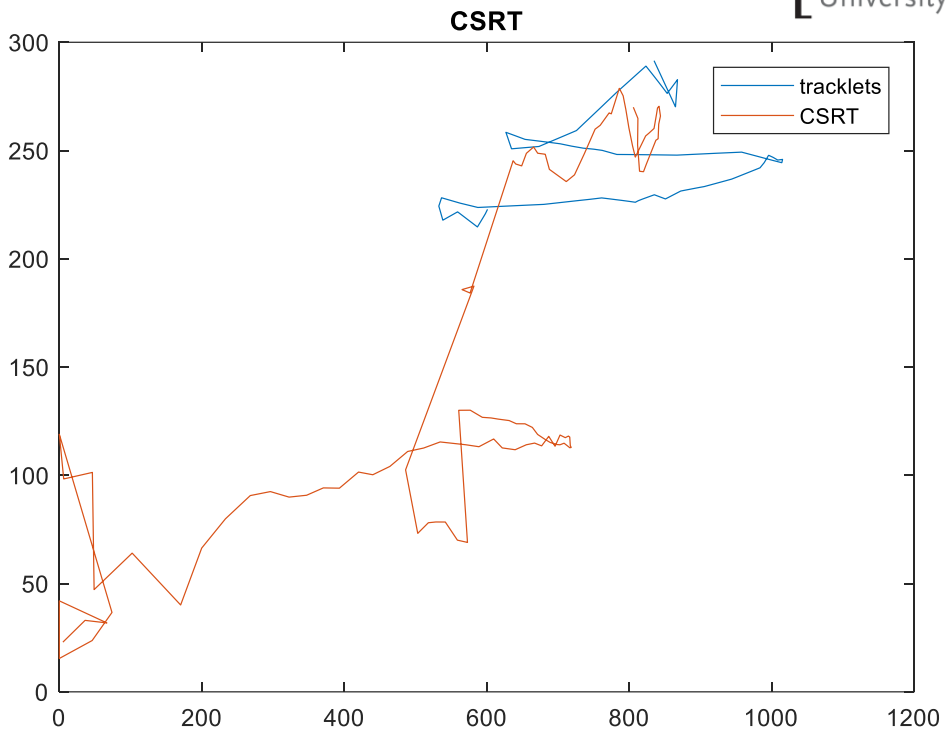


Figure 40: CSRT

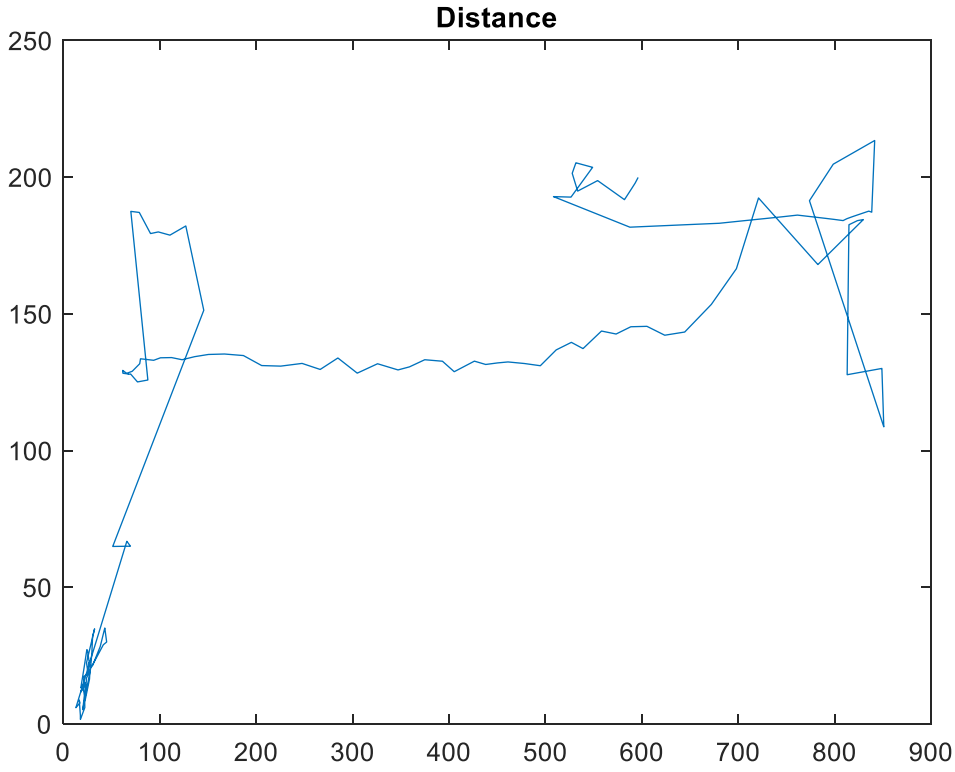


Figure 41; CSRT distance

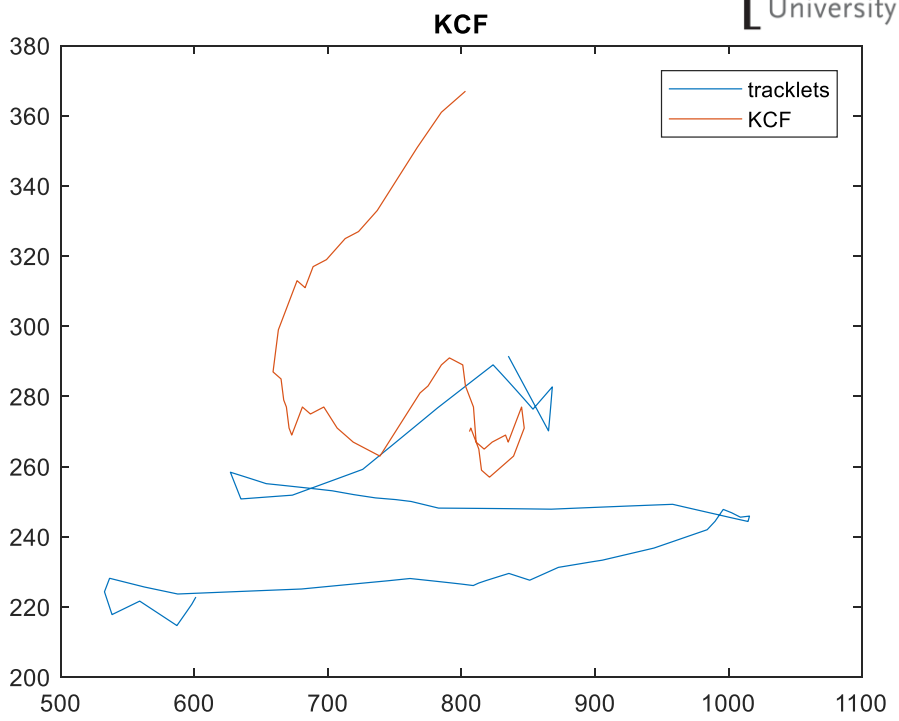


Figure 42: KCF

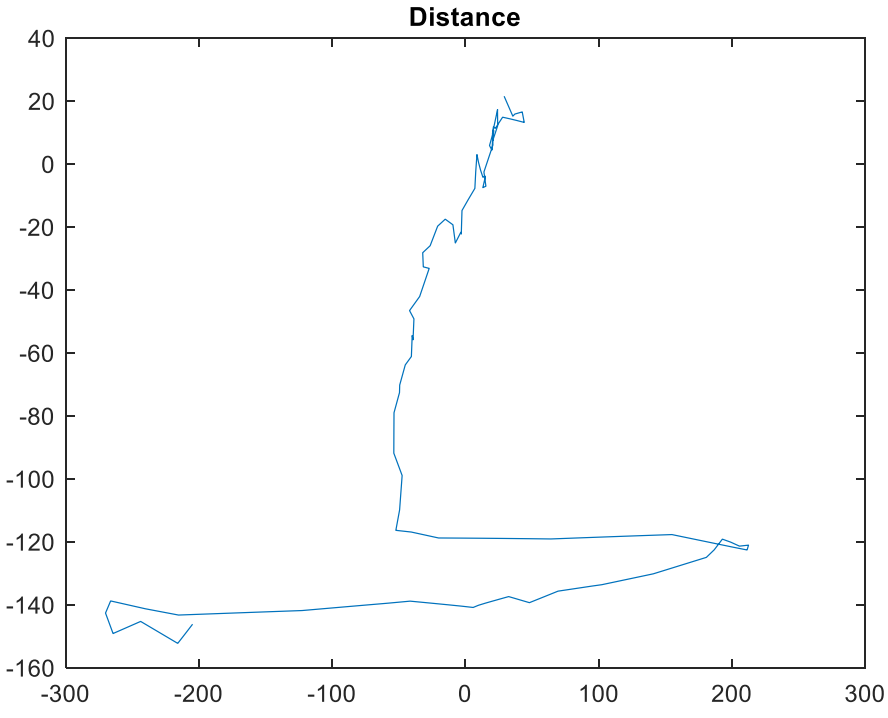


Figure 43: KCF distance

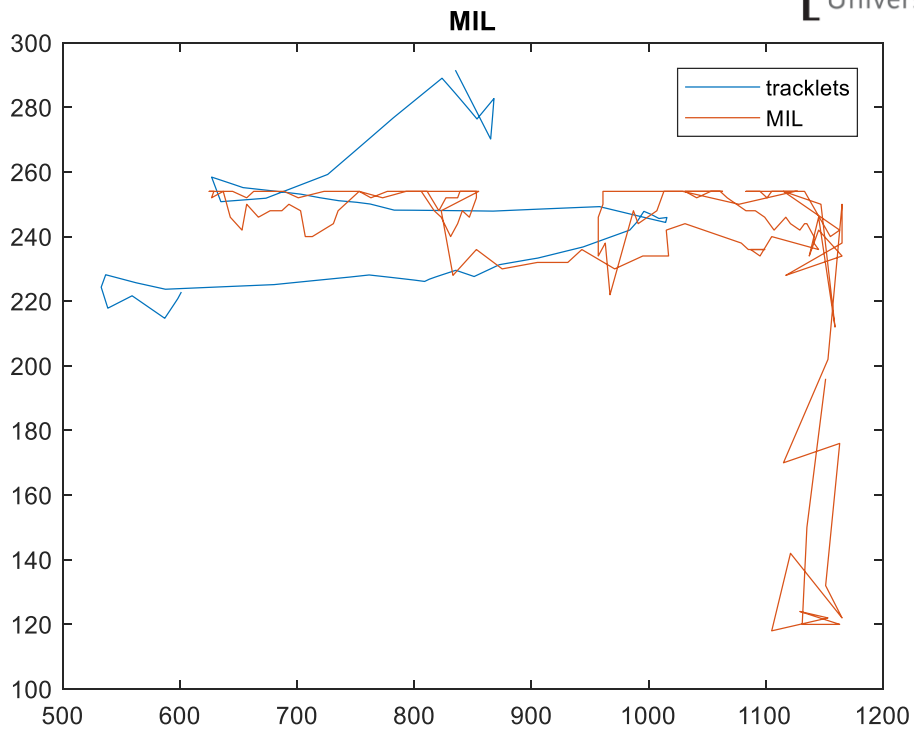


Figure 44: MIL

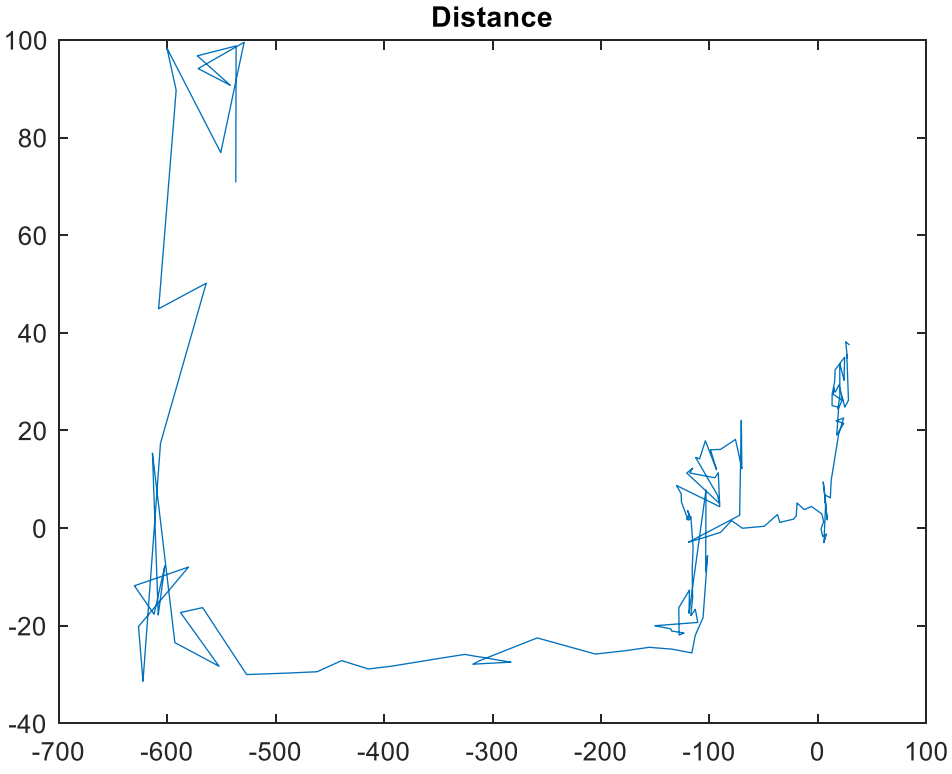


Figure 45: MIL distance

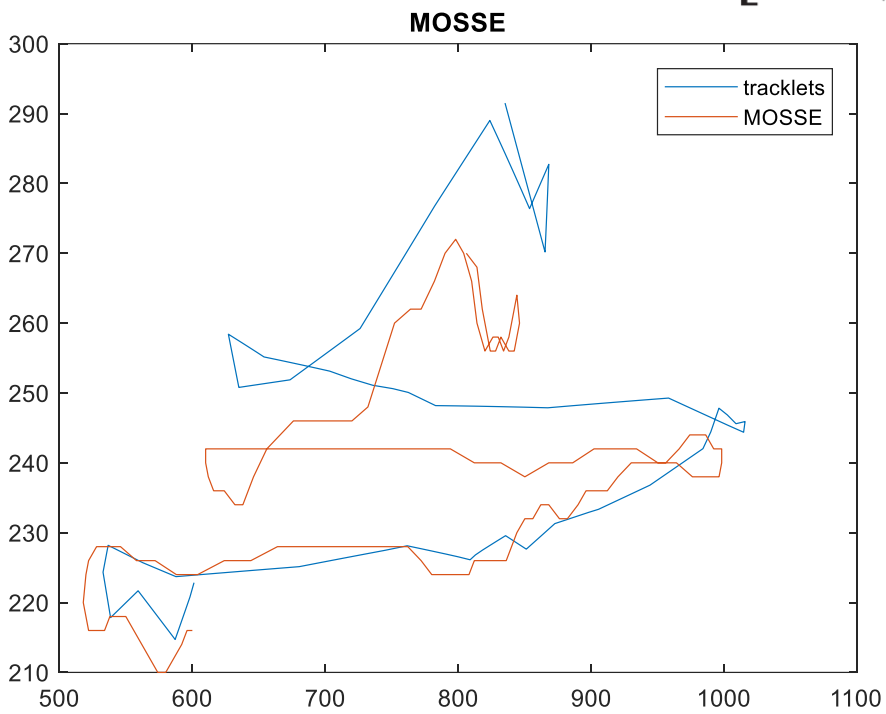


Figure 46: MOSSE

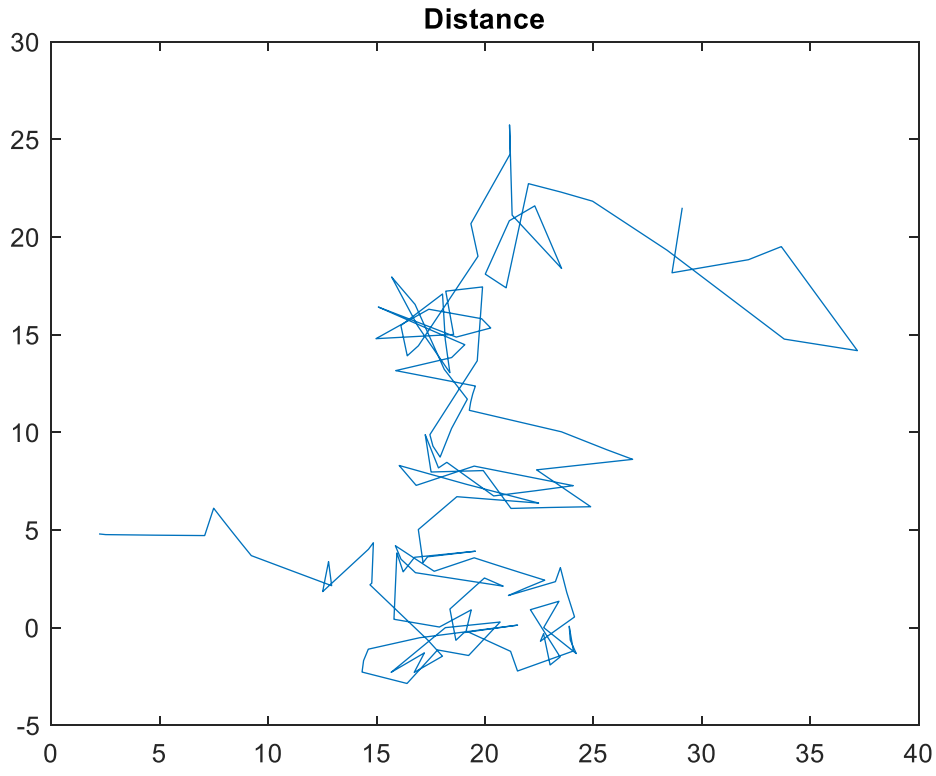


Figure 47: MOSSE distance

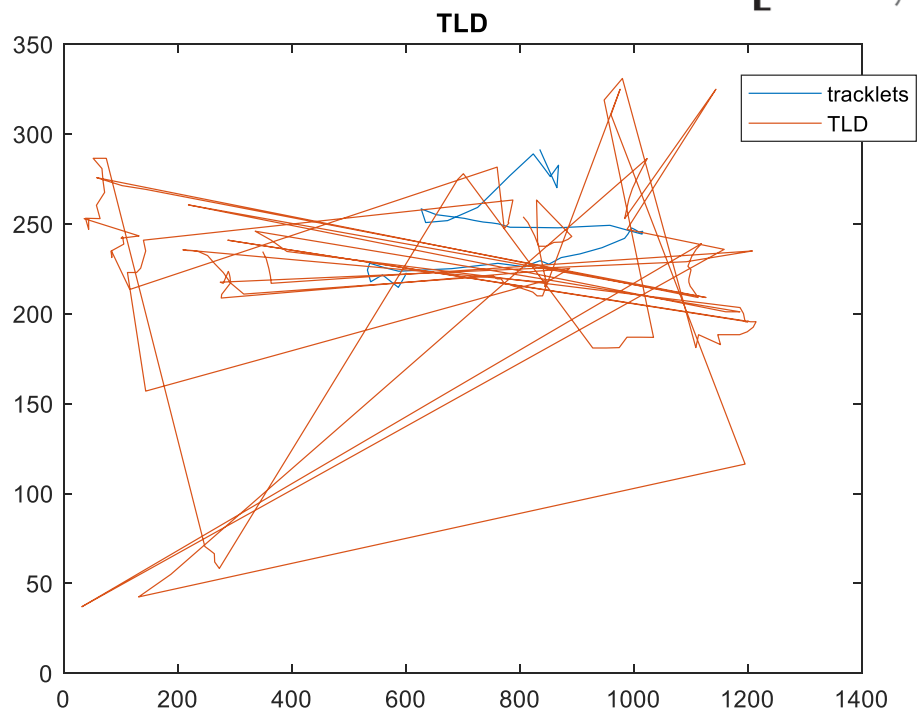


Figure 48: TLD

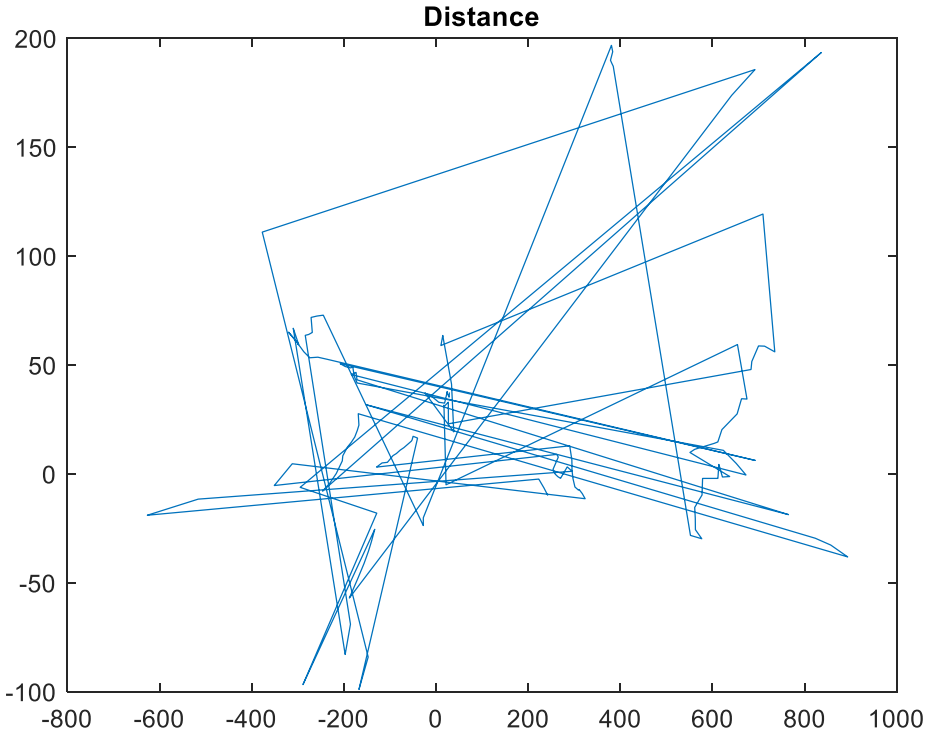


Figure 49: TLD distance

MOSSE was the only one tracking algorithm that was able to track the cyclist till the end of the sequence. All other algorithms lost the cyclist at different frames of the video sequence. On the other hand, when TLD lost the cyclist, it tried to adapt itself by its own learning algorithm, however it could not find the cyclist again and drew a bounding box randomly at different point in the frame.

In the table (1) all average distances are shown, however they are not trustable because in some algorithms, despite tracker lost the object it kept tracking another object or stuck in a point outside of the object. Therefore it may seem like more accurate but it was just coincidence.

	BOOSTING	CSRT	MIL	MOSSE	TLD	KCF
Average X	299,0496	362,7225	173,2249	18,9451	118,7733	-8,5059
Average Y	56,7859	125,7253	6,3806	7,6355	24,0369	-91,2664

Table 1: Average Distance of all algorithms

5.4 Robustness in Grayscale Format

Robustness was checked for MOSSE and CSRT by shifting the bounding boxes 10 pixels. Both of the trackers are not robust.

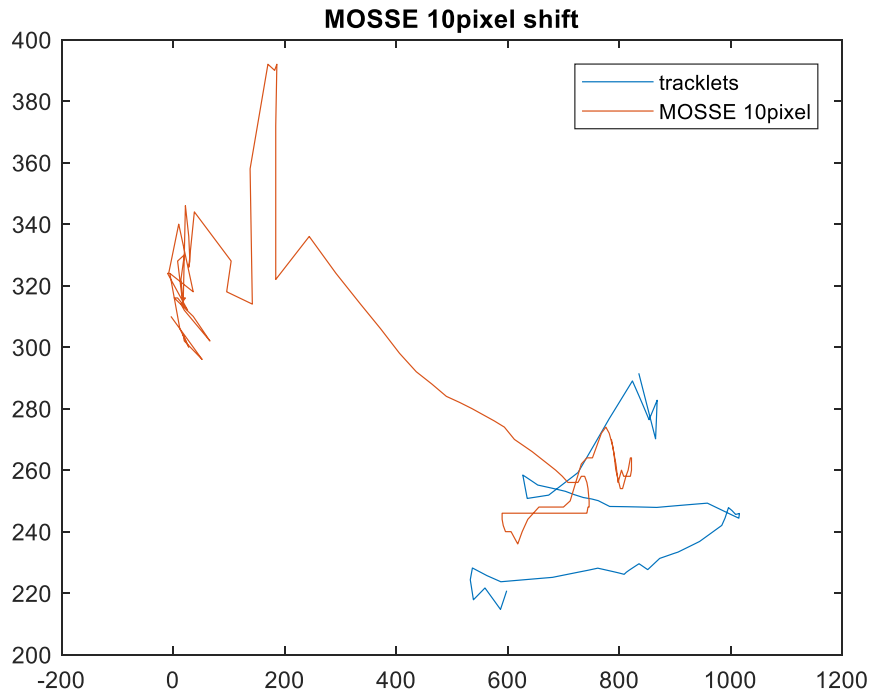


Figure 50: MOSSE with 10 pixels shift

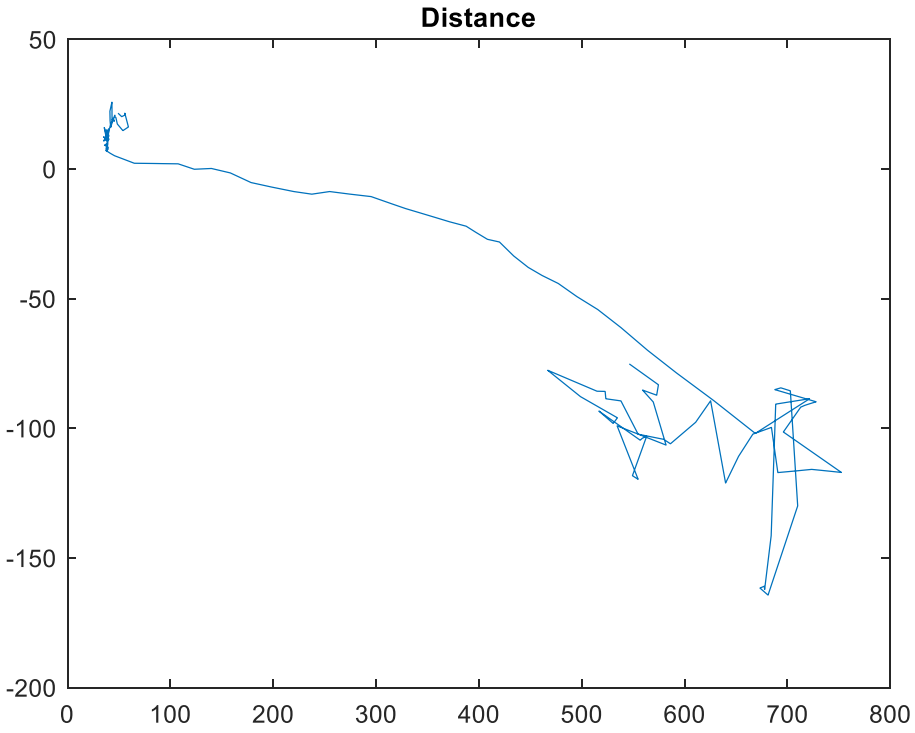


Figure 51:Distance graph of MOSSE with 10 pixel shift

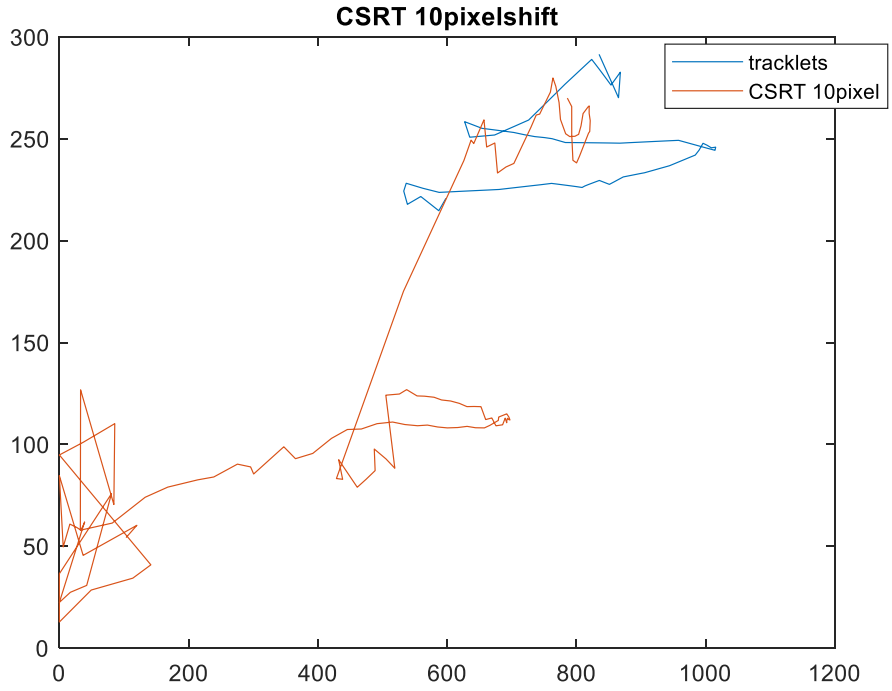


Figure 52: CSRT with 10 pixels shift

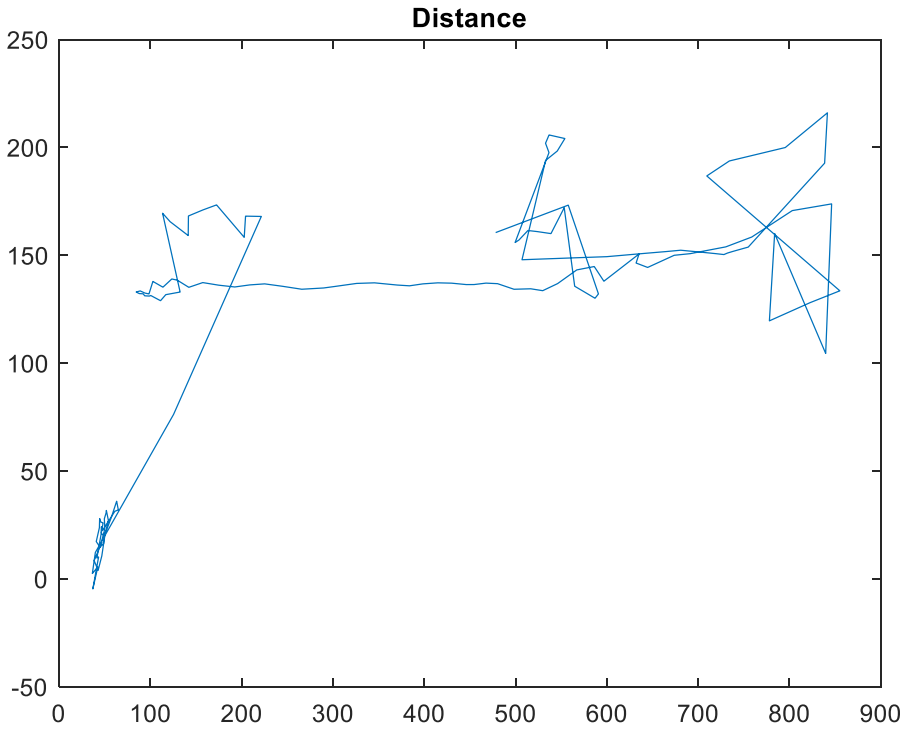


Figure 53:Distance graph CSRT with 10 pixel shift

	Shifted MOSSE	Shifted CSRT
Average distance of X	335,8470	367,8179
Average distance of Y	-40,3383	119,9271

Table 2: Average Differences in Robustness test

5.5 Tracking with RGB Format

We used CSRT to track the video sequence. In order to improve CSRT, we used feedback information in CSRT and add depth data from STIXEL method. In RGB format, Van in the video sequence was chosen as the desired object to track.

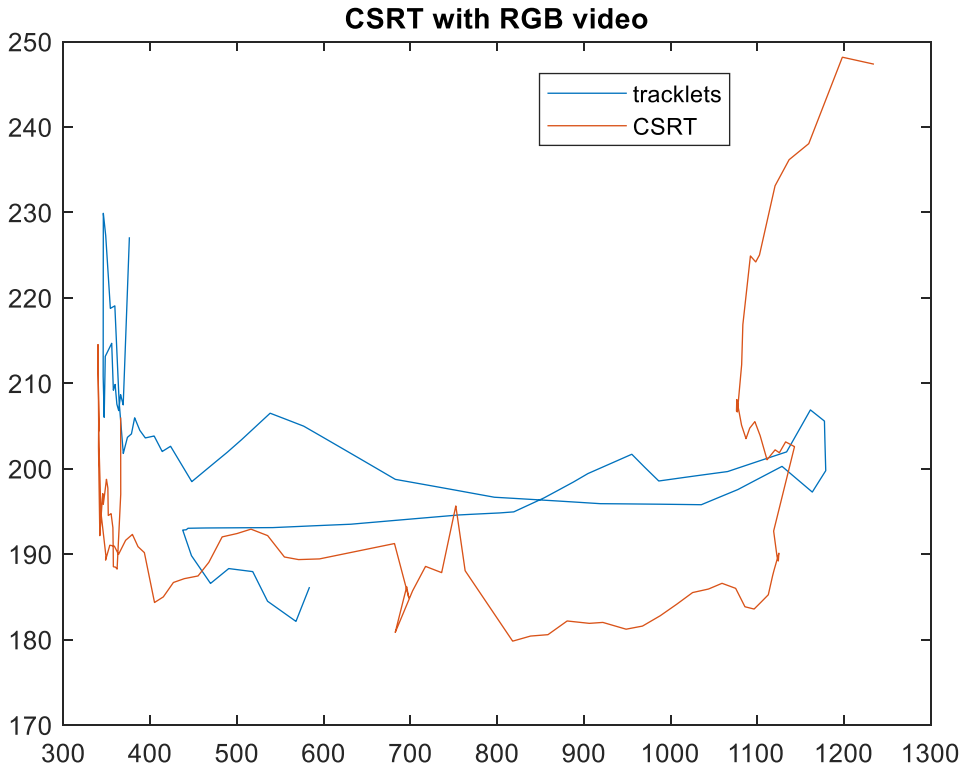


Figure 55: CSRT with RGB video

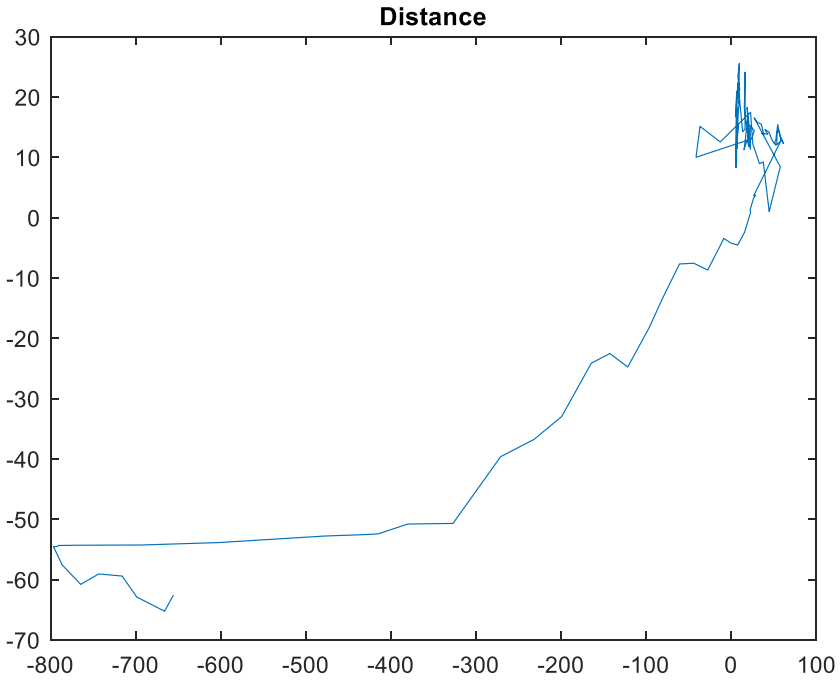


Figure 56: The Distance data of CSRT with RGB video

5.5.1 Using Depth Data in Tracking

For STIXEL method disparity map is already created. In RGB image, we changed R channel of the image with disparity information and thus the image with depth information is obtained.

Figure 57: Image with depth

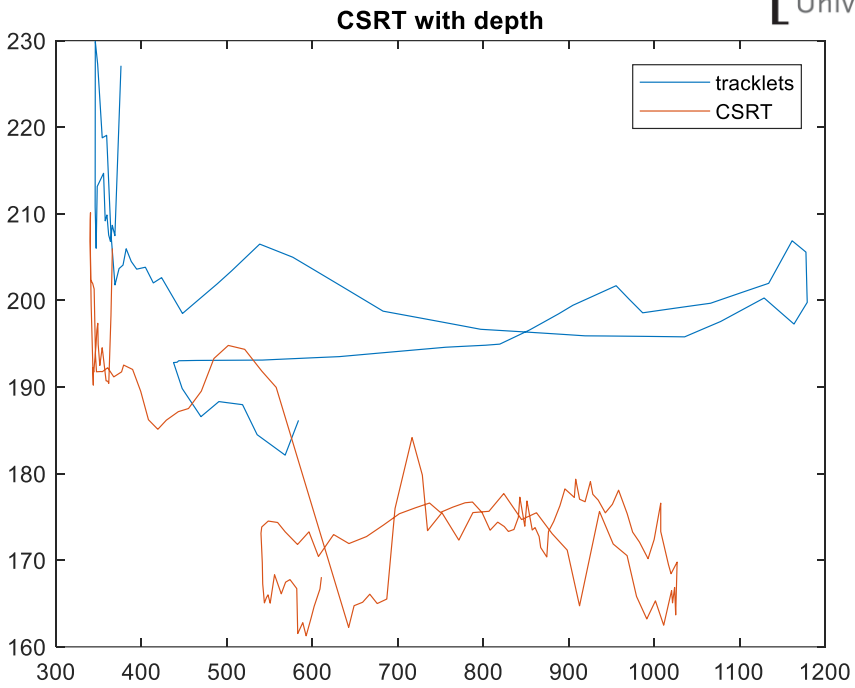


Figure 58: CSRT with depth data

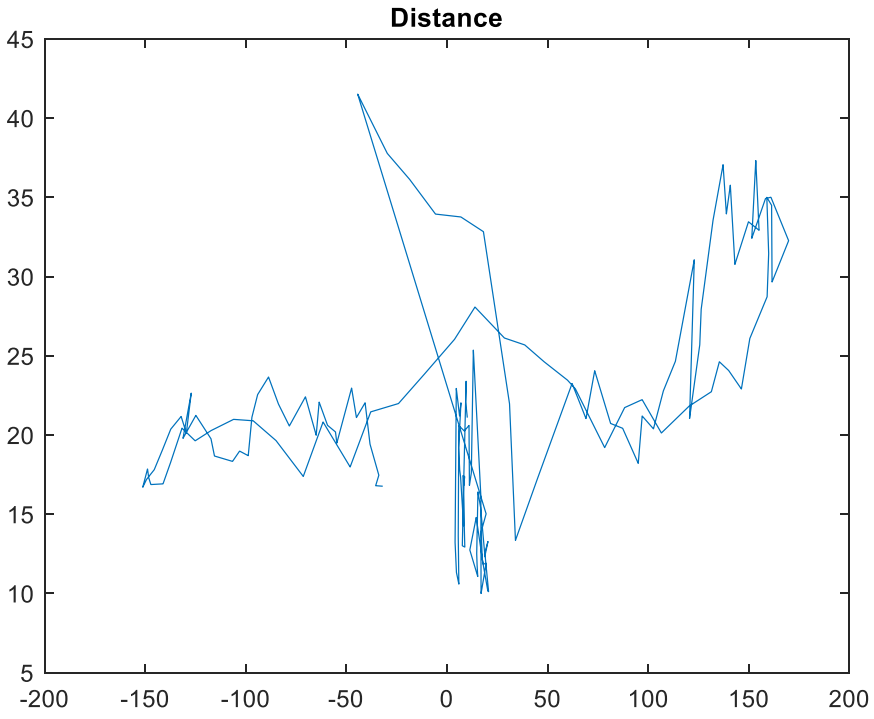


Figure 59: The distance graph of CSRT with depth data

	CSRT-RGB	CSRT-Depth
Average distance in X axis	-212,2441	9,0140
Average distance in Y axis	-12,6936	21,4357

Table 3: Average distances of CSRT-RGB and CSRT-Depth

6 Conclusion

In this master thesis, we have examined different tracking algorithms in Open-CV. The main purpose of the project was to improve CSRT tracking algorithm. However, the first obstacle was finding a way to compare the tracking results. The first approach was calculating object position differences between tracklets data(provides real data) and tracker data. On the other hand we also calculated average differences to make a better comparison. However the biggest problem with taking position differences into account is it does not provide accurate data. The reason, as it was explained before, one method may lose its first object but keep tracking another object around the desired object. In that case, we might have a similar position graph with real position graph but the graph does not provide any trustable result. For example, when we compare CSRT-RGB result and CSRT-depth result, it can be said that CSRT-depth result is better. However, during the CSRT-depth test tracker lost the van and started tracking the cyclist. Since the cyclist and the van have similar path in the video sequence, our result seems close the real path.

In order to make a comparison in our project, positive tracking time can be taken into account. For example for CSRT tracker, the frame number was determined when tracker lost the desired object.

Van1 is desired object	CSRT-Grayscale	CSRT-RGB	CSRT-Depth
Frame number	52	88	57

Table 4: Frame numbers when trackers lose the object

As it can be seen in the table (4). CSRT works better with RGB than Grayscale format. Depth data was added to RGB format in order to improve CSRT's accuracy but it decreased the accuracy.

Another attempt to improve CSRT algorithm was using feedback data. CSRT has its own control system to determine if object is being tracked. This system is a binary system in original source code. We changed the source code to have that feedback value. The feedback value was used to eliminate some false bounding boxes in the frame. Furthermore, we wanted to use that value to predict the time when tracker loses the object. The problem with using the feedback value is there is not constant value or constant percentage for all objects to set a boundary limit. When video sequence starts, each bounding box has its own evaluation value. We set 70% limit to decide performance of tracking. However in 37. frame we lost many bounding boxes and at another attempt could not find an exact common percentage for all bounding boxes.

```
32, Time elapsed: 83.110ms ,Total Time : 2772.713ms , Average Time : 84.022ms
id:0  quality:0.0999326   feedback:0.101151  check:0  status:1
id:1  quality:0.0438758   feedback:0.0438758  check:0  status:1
id:2  quality:0.0357246   feedback:0.0388129  check:0  status:0
id:3  quality:0.0732008   feedback:0.0932181  check:0  status:0
id:4  quality:0.037412   feedback:0.0399741  check:0  status:1
id:5  quality:0.152985   feedback:0.165611  check:0  status:0
33, Time elapsed: 81.765ms ,Total Time : 2854.502ms , Average Time : 83.956ms
id:0  quality:0.101571   feedback:0.0999326  check:0  status:1
id:1  quality:0.0438758   feedback:0.0438758  check:0  status:1
id:2  quality:0.0371571   feedback:0.0357246  check:0  status:0
id:3  quality:0.0749849   feedback:0.0732008  check:0  status:0
id:4  quality:0.0364761   feedback:0.037412  check:0  status:1
id:5  quality:0.137284   feedback:0.152985  check:0  status:0
34, Time elapsed: 81.768ms ,Total Time : 2936.276ms , Average Time : 83.894ms
id:0  quality:0.0872663   feedback:0.101571  check:0  status:1
id:1  quality:0.0438758   feedback:0.0438758  check:0  status:1
id:2  quality:0.0391144   feedback:0.0371571  check:0  status:0
id:3  quality:0.0980659   feedback:0.0749849  check:0  status:0
id:4  quality:0.041328   feedback:0.0364761  check:0  status:1
id:5  quality:0.126845   feedback:0.137284  check:0  status:0
35, Time elapsed: 79.906ms ,Total Time : 3016.188ms , Average Time : 83.783ms
id:0  quality:0.0921565   feedback:0.0872663  check:0  status:1
id:1  quality:0.0438758   feedback:0.0438758  check:0  status:1
id:2  quality:0.0407827   feedback:0.0391144  check:0  status:0
id:3  quality:0.0893537   feedback:0.0980659  check:0  status:0
id:4  quality:0.0370654   feedback:0.041328  check:0  status:1
id:5  quality:0.129479   feedback:0.126845  check:0  status:0
36, Time elapsed: 78.662ms ,Total Time : 3094.855ms , Average Time : 83.645ms
id:0  quality:0.109365   feedback:0.0921565  check:0  status:1
id:1  quality:0.0365615   feedback:0.0438758  check:0  status:1
id:2  quality:0.0368271   feedback:0.0407827  check:0  status:0
id:3  quality:0.0956912   feedback:0.0893537  check:0  status:0
id:4  quality:0.043742   feedback:0.0370654  check:0  status:1
id:5  quality:0.122292   feedback:0.129479  check:0  status:0
37, Time elapsed: 76.167ms ,Total Time : 3171.027ms , Average Time : 83.448ms
id:0  quality:0.113005   feedback:0.109365  check:0  status:1
id:1  quality:0.0428361   feedback:0.0365615  check:0  status:1
id:2  quality:0.0370306   feedback:0.0368271  check:0  status:0
id:3  quality:0.102401   feedback:0.0956912  check:0  status:0
id:4  quality:0.0403964   feedback:0.043742  check:0  status:1
id:5  quality:0.131965   feedback:0.122292  check:0  status:0
```

Figure 60: Feedback Test(if feedback of a bounding box drops to %30 of its initial value in a consecutive 10 frame, the bounding box is deleted. Check represents the consecutive number and status represents the deleted bounding box)

In conclusion, we have seen measuring distance for testing accuracy is not a safe method. Besides, using depth data in an RGB frame instead of one of its channel does not increase the accuracy of the CSRT tracker. And as last, feedback value in CSRT source code is not usable for filtering bounding boxes.

7 Future Prospects

Some steps that were decided at the beginning of the thesis could not be completed because of time limitation. As a future work of this thesis, those steps can be implemented.

The first step is about how to use depth data. In our project depth data was added instead of one of the RGB channel. Actually,depth data can be added as a fourth channel into the frame and RGB-D four-channel image format can be created. However the reason why we did not make it is CSRT algorithm in Open-CV does not work with fourth channel images. CSRT algorithm can be improved to use four-channel data.

The second step that we could not complete was running this project with ROS(Robot-Operating-System) in order to make simulation in 3D environment. The data was transferred to ROS in this thesis but determining the 2D object position in 3D environment could not be completed. The project may be implemented in ROS as future work.

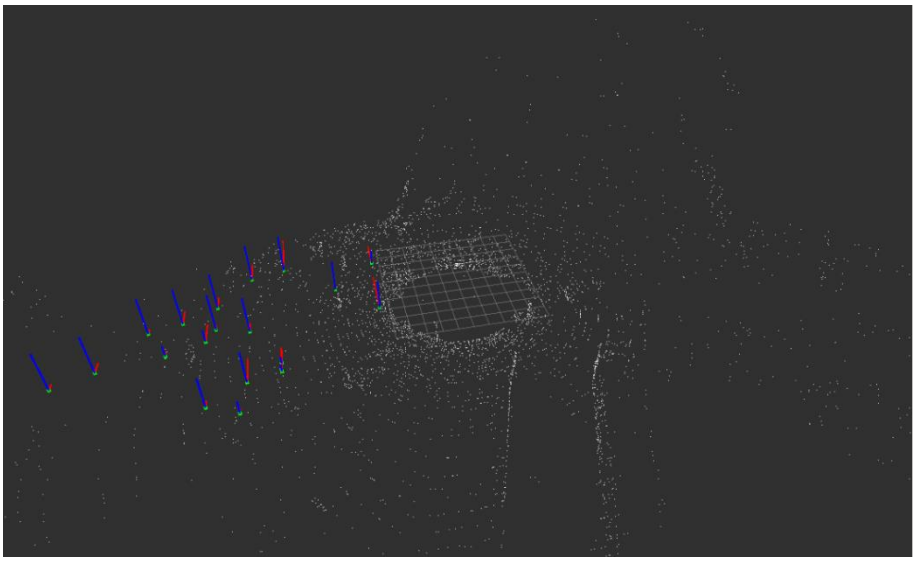


Figure 61: The image from ROS rviz.

And as last, STIXEL method that was used in this thesis does not classify and filter the objects in the video sequence. It causes a problem like many irrelevant bounding boxes. The project can be implemented again with another STIXEL code to have more accurate results.

Bibliography

[1] D. H. a. T. Wiesel , "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology,* pp. 574-591, 1959.

[2] "https://goodpsychology.wordpress.com," 13 3 2013. [Online]. Available: https://goodpsychology.wordpress.com/2013/03/13/235/.

[3] W. Kirsch, 23 5 2007. [Online]. Available: https://commons.wikimedia.org/wiki/File:NBSFirstScanImage.jpg.

[4] B. Centurion. [Online]. Available: https://www.reddit.com/r/RetroFuturism/comments/a6en7s/1956_buick_centurion_concept_car_of_the_future/.

[5] O. Cameron. [Online]. Available: https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff.

[6] S. T. M. M. H. D. D. Stavens, "Stanley: The robot that won the DARPA Grand Challenge," *FIELD ROBOTICS,* vol. 23, no. 9, 25 9 2006.

[7] M. LAPEDUS. [Online]. Available: https://semiengineering.com/radar-versus-lidar/.

[8] N. H. T. S. Administration. [Online]. Available: https://www.nhtsa.gov/driver-assistance-technologies/driver-assistance-technologies.

[9] S. Mattoccia, Lecture Notes.

[10] U. F. D. P. H Badino, "The stixel world-a compact medium level representation of the 3d-world.," *Joint Pattern Recognition Symposium,* pp. 51-60, 2009.

[11] R. Dechter, Learning while searching in constraint-satisfaction problems, California, 1986.

[12] T. Ergin. [Online]. Available: https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad.

[13] G. E. Hinton, "Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines vinod nair," 2010.

[14] J. D. T. D. J. M. Ross Girshick, "Rich feature hierarchies for accurate object detection and semantic segmentation," *In Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 580-587, 2014.

[15] R. Girshick, "Fast r-cnn," *In Proceedings of the IEEE international conference on computer vision,* pp. 1440-1448, 2015.

[16] R. &. H. K. G. R. Shaoqing, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems,* pp. 91-99, 2015.

[17] J. Redmon, " You only look once: Unified, real-time object detection," *In Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 779-788, 2015.

[18] A. F. a. J. Redmon, "YOLO9000: better, faster, stronger," *In Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 7263-7271, 2016.

[19] A. Redmon J. & Farhadi, "Yolov3: An incremental improvement," 2018.

[20] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," *IEEE,* pp. 807-814, 2005.

[21] U. F. a. D. P. Hernan Badino, "The stixel world-a compact medium level representation of the 3d-world," *Joint Pattern Recognition Symposium,* pp. 51-60, 2009.

[22] S. K. Gehrig and U. Franke, "Improving stereo sub-pixel accuracy for long range stereo," *IEEE 11th International Conference on Computer Vision,* pp. 1-7, 2007.

[23] A. Elfes, " Sonar-based real-world mapping and navigation.," *IEEE Journal on Robotics and Automation,* pp. 249-265, 1987.

[24] D. Bagnell, "Occupancy Maps," *Statistical Techniques in Robotics,* pp. 16-831.

[25] tkwoo. [Online]. Available: https://github.com/tkwoo/StereoVisionforADAS.

[26] D. S. ,. B. J. Bolme, "Visual object tracking using adaptive correlation filters," *In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. 2544-2550, 2010.

[27] R. C. Joao F. Henriques, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence,* pp. 583-596, 2014.

[28] C. Dehlin, "Visual Tracking Using Stereo Images," 2019.

[29] Z. Kalal, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence,* pp. 1409-1422, 2010.

[30] K. M. Z. Kalal, "Forward-backward error: Automatic detection of tracking failures," *2010 20th International Conference on Pattern Recognition,* pp. 2756-2759, 2010.

[31] M. G. H. B. Helmut Grabner, "Real-time tracking via on-line boosting," *Bmvc,* vol. 1, pp. 1-6, 2006.

[32] M.-H. Y. B. Boris Babenko, " Visual tracking with online multiple instance learning," *IEEE Conference on computer vision and Pattern Recognition,* pp. 983-990, 2009.

[33] S. T. David Held, "Learning to track at 100 fps with deep regression networks," *European Conference on Computer Vision,* pp. 749-765, 2016.

[34] T. V. ,. L. C. Z. Alan Lukezic, "Discriminative Correlation Filter with Channel and Spatial Reliability," *CVPR,* pp. 6309-6318, 2017.

[35] U. &. A. G. &. P. L. Raquel, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," *Conference on Computer Vision and Pattern Recognition(CVPR),* 2012.

[36] A. Z. R Hartley, Multiple View Geometry in Computer Vision, 2003.

[37] [Online]. Available: https://medium.com/@tuncerergin/convolutional-neural-network-convnet-yada-cnn-nedir-nasil-calisir-97a0f5d34cad.

[38] T. K. Bruce D. Lukas, "An Iterative Image Registration Technique with an Application to Stereo Vision," *DARPA,* pp. 121-130, 1981.