Kamerabasierte Fahrbahnerkennung zur automatisierten Fahrbahnführung eines Modellauto

Bahri Enis Demirtel

Master Thesis – 02. November 2017

Betreuer: Dr. -Ing. Eric Lenz

TECHNISCHE UNIVERSITÄT DARMSTADT

REGELUNGSTECHNIK UND MECHATRONIK rtm

Aufgabenstellung

Für schriftliche Arbeiten (Pro-/Projektseminar, Studien-, Bachelor-, Master-, Diplomarbeiten, etc.) soll Studierenden ein LaTeX-Dokument zur Verfügung gestellt werden, das die Vorgaben aus den *Richtlinien zur Anfertigung von Studien- und Diplomarbeiten* [**?**] umsetzt. Die Dokumentation soll die Funktionen des Dokumentes beschreiben und Hinweise zu ihrer Anwendung geben.

Grundlage ist die `tudreport`-Klasse. Die damit erstellten Arbeiten müssen sowohl zum Ausdrucken geeignet sein als auch für die Bildschirmdarstellung und die elektronische Archivierung als PDF-Datei.

Beginn:	02. May 2017
Ende:	02. November 2017
Seminar:	15. November 2017

Prof. Dr.-Ing. Ulrich Konigorski

Dr. -Ing. Eric Lenz

Technische Universität Darmstadt
Institut für Automatisierungstechnik und Mechatronik
Fachgebiet Regelungstechnik und Mechatronik
Prof. Dr.-Ing. Ulrich Konigorski

Landgraf-Georg-Straße 4
64283 Darmstadt
Telefon 06151/16-25200
www.rtm.tu-darmstadt.de

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 02. November 2017

Bahri Enis Demirtel

Kurzfassung

Das LaTeX-Dokument `sada_tudreport` ist eine Vorlage für schriftliche Arbeiten (Proseminar-, Projektseminar-, Studien-, Bachelor-, Master- und Diplomarbeiten, etc.) am Institut für Automatisierungstechnik der TU Darmstadt. Das Layout ist an die *Richtlinien zur Anfertigung von Studien- und Diplomarbeiten* [**?**] angepasst und durch Modifikation der Klasse `tudreport` realisiert, so dass in der Arbeit die gewohnten LaTeX-Befehle benutzt werden können. Die vorliegende Anleitung beschreibt die Klasse und gibt grundlegende Hinweise zum Verfassen wissenschaftlicher Arbeiten. Sie ist außerdem ein Beispiel für den Aufbau einer Studien-, Bachelor-, Master- bzw. Diplomarbeit.

Schlüsselwörter: Studienarbeit, Bachelorarbeit, Masterarbeit, Diplomarbeit, Vorlage, LaTeX-Klasse

Abstract

The LaTeX document `sada_tudreport` provides a template for student's research reports and diploma theses (" Proseminar-, Projektseminar-, Studien-, Bachelor-, Master- und Diplomarbeiten") at the Institute of Automatic Control, Technische Universität Darmstadt. The layout is adapted to the *"Richtlinien zur Anfertigung von Studien- und Diplomarbeiten"* [**?**] and is implemented by modification of the standard `tudreport` class, so that common LaTeX commands can be used in the text. This manual describes the class and dwells on general considerations on how to write scientific reports. Additionally, it is an example for the structure of a thesis.

Keywords: Research reports, diploma theses, template, LaTeX class

Inhaltsverzeichnis

Symbole und Abkürzungen

Lateinische Symbole und Formelzeichen

Symbol	Beschreibung	Einheit
I	Strom	A
R	Widerstand	Ω
U	Spannung	V

Griechische Symbole und Formelzeichen

Symbol	Beschreibung	Einheit
$\mathbf{\Psi}$	Datenmatrix	
σ	Standardabweichung	
ω	Kreisfrequenz	s^{-1}

Abkürzungen

Kürzel	vollständige Bezeichnung
Dgl.	Differentialgleichung
LS	Kleinste Quadrate (*Least Squares*)
PRBS	Pseudo-Rausch-Binär-Signal (*Pseudo Random Binary Signal*)
ZVF	Zustandsvariablenfilter

1 Introduction

1.1 Introduction

As in all industries, technology in the automotive industry is continuing to develop day by day. For example, the number of sensors, and their corresponding features, is increasing exponentially. One such sensor is the color camera. To begin with, in the automotive industry, cameras were used only to assist drivers in parking and reversing.

Nowadays, however, one of the main functions of color cameras is lane detection, in both autonomous cars and in cars equipped with a lane departure warning system. In this master's thesis, the lanes will be detected and then formulated mathematically.

The results of this master's thesis will be utilized and expanded upon by the students who will participated in the Echtzeitsysteme Projektseminar at the Technical University of Darmstadt. One of the aims of this seminar is to attend the Carolo-Cup organized annually by the Technical University of Braunschweig. Because of that, the width, the curvature, and the changes of the curvature of the track used in this master's thesis are the same as those belonging to the track used in the Carolo-Cup. In a real-life situation, there are of course oftentimes more factors that can hinder lane detection, including shadows cast by trees, buildings, and other structures; sunlight directly entering the lens of the camera and similarly less-than-ideal lighting conditions; dirt and debris on the road surface; and so on.

Therefore, the lanes of the track must be detected in a sufficiently short amount of time and there should be no dead time between lane detection and mathematical formulation. Lane detection must also be sufficiently robust, so that it should not be disrupted by less-than-ideal lighting conditions.

1.2 Problem Statement and Objective Target

Autonomous driving is a topic currently being actively researched. Research on autonomous driving can be conducted in two fundamental areas: lane detection and lane guidance. With regard to lane detection, there are different scientific techniques that can be utilized, according to the literature, all with their own advantages and disadvantages under different conditions. For example, some techniques are suitable for straight lines, but not for curves. Others are suitable for curves as well but do not function well under certain light conditions. Others still are quite robust and suitable for curves, yet are computationally intensive (resulting in a video

feed with significant gaps). In this master's thesis, my aim is to research and implement the most appropriate and effective method for use in the Carola-Cup.

1.3 Structure of Thesis

In Chapter 2, the fundamentals of lane detection are explained. All methods utilized in this thesis, along with their respective justifications, are also explained in this chapter. Some methods are also compared with regard to their advantages and disadvantages.

In Chapter 3, the steps of implementation are explained. The components can be divided broadly into the properties of the track, the hardware of the model car, and the software libraries and programs to be utilized. In this chapter, the program flow will also be explained in detail.

In Chapter 4, the results of the methods utilized will be compared. The computing time of all phases in this thesis will be presented and discussed. Also, all parameters utilized and their effects on this thesis will be also presented and discussed. In this chapter, the researcher will attempt to find an answer to the question, 'How can computing time be reduced?'.

In Chapter 5, the state of the art will be discussed. The other possible solutions for lane detection will also be explored here and their advantages and disadvantages will be compared.

In Chapter 6, all results of this master thesis will be presented and possible improvements and/or enhancements will be discussed.

2 Fundamentals

2.1 Properties of Truck at Carola-Cup

The Carolo-Cup is an annual competition at the Technical University of Braunschweig which are attended by student. Every year the truck and some properties of the competition is changing. For example, in the competitions until 2017 there was no traffic sign but from 2017 there are also some traffic signs, speed limit zones, blocked areas and crosswalks with pedestrian. Because of this reason, in the competitions until 2017, there was only one way to understand who has the right of way. If there is a stop line in the way which in front of intersection, it means, the car has to wait until the intersection is free. In the competitions from 2017, the intersections are in different parts: They are 'Intersections with stop lines', 'Intersections with give-way lines', 'Intersections with priority to right', 'Enforced crossing direction - give-way condition', 'Enforced crossing direction - right of way condition'. Except 'Intersections with priority to right', they all have traffic signs, which signs who has priority. If there is a no traffic sign, it means,right side always have priority.

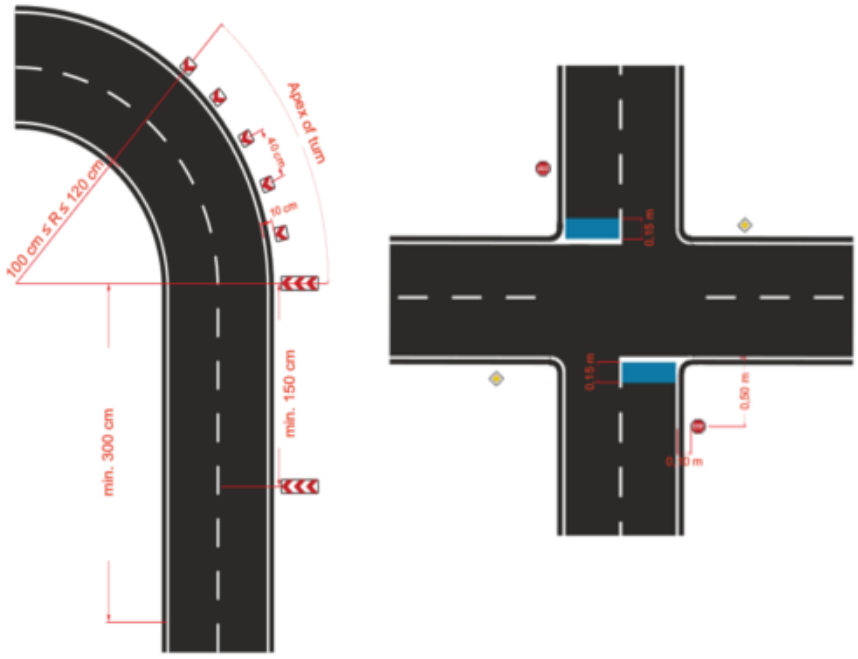


Abbildung 2.1: Left: Markings for sharp turns at Carolo-Cup. Right: Intersections with stop lines at Carolo-Cup

2.2 Inverse Perspective Mapping

Inverse Perspective Mapping(IPM) is an algorithm which is able to obtain accurate bird's-eye view images from the sequential of forward looking cameras. With the IPM algorithm, each image pixel is remapped, and a new array of pixels is created where the lines in perspective are transformed into straight lines and objects are distorted. IPM is one of the most used methods in lane detection. In lane detection, IPM ensures that the lanes are shown vertical and parallel to each other. On the other hand, because of the re-mapping of pixels, IPM is a computationally expensive method. Because of this reason, in some cases in this master's thesis, rather than remapping all pixels of the images, only the pixels relevant to the lane and accordingly, the fitted curve, were remapped. Thanks to this, in some cases, a lot of computing time was saved.

In order to use the IPM method, the intrinsic and extrinsic parameters of camera are necessary to process images for coordinate transformation and calibration.

- **Intrinsic Parameters :** Intrinsic parameters are camera-specific. It includes information of the focal length (f_x, f_y) and optical centers (c_x, c_y). It is also called a camera matrix. It depends on the camera only, so once calibrated, it can be stored for future purposes. It is expressed as a 3x3 matrix:

$$\text{camera matrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- **Extrinsic Parameters :** Extrinsic parameters are dependent on the camera position. The parameters are H and θ. H is the distance between the camera and ground.. θ is the camera tilt angle.

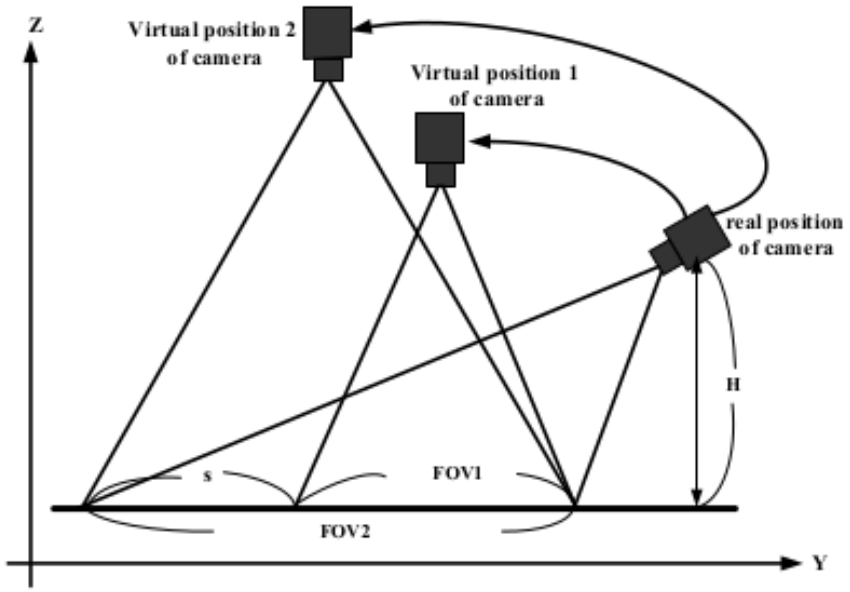


Abbildung 2.2: Related Positions of the Camera [1]

As seen at **??**, the camera on the car has field of view 2(FOV2) at the real position of the camera but in this case, the view is not a bird's-eye view, so if the same FOV is to be observed from a bird's-eye view, IPM will virtually change the position to Virtual Position 2 of the camera. In this case, although the camera is at its real position, it will appear as though it is at Virtual Position 2. For that, the image coordinates must be also changed. Below, steps of IPM calculations from the paper of [1] will be detailed.

In the formula, the original image coordinates will be defined as (x,y), the destination image coordinates will be defined as (x^*,y^*), the distance between the ground and the camera will be defined as H, the focal length of camera will be defined as f, and the tilt angle of camera will be defined as θ.

$$x^* = H\frac{xsin\theta + fcos\theta}{-ycos\theta + fsin\theta} \; ; \; y^* = H\frac{ysin\theta + fcos\theta}{-ycos\theta + fsin\theta}$$

In this equation , the transformed component values of x^* and y^* may be less than or equal to zero. Because of this reason, a constant d is defined as $\left|H(sin\theta + cos\theta)/(fsin\theta - cos\theta)\right| + 1$. This means that the coordinate point in the original source image has been mapped into the point of the destination image coordinate system. Below there is the proposed equation :

$$x^* = H\frac{x\sin\theta + f\cos\theta}{-y\cos\theta + f\sin\theta} + \mathrm{d}\ ,\ y^* = H\frac{y\sin\theta + f\cos\theta}{-y\cos\theta + f\sin\theta} + \mathrm{d}\ ,\ \text{where d} = \left|\frac{H(\sin\theta + f\cos\theta)}{f\sin\theta - \cos\theta}\right| + 1$$

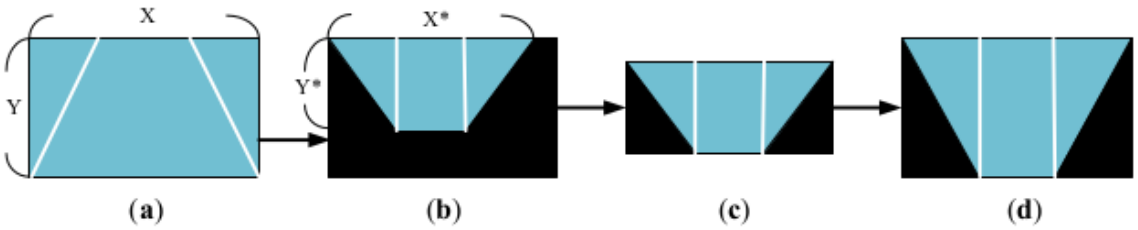


Abbildung 2.3: Procedures of IPM

2.3 Edge Detection

Edge detectors are essential parts of most of computer vision systems. Edge detectors dramatically decrease the amount of data to be processed and extract the useful parts of images. They work by detecting discontinuities in brightness. In this project, the edge detector was used in order to detect the lanes and to exclude unnecessary information from images. There are different methods for edge detection, but they can be grouped into two categories. They are :

- **Gradient method :** This method searches for the maximum and minimum in the first derivative of the image and within can find the edges. For this method, the first order derivative filter must be used. For example : Sobel-Operator.
- **Laplacian method :** This method searches for the zero crossing in the second derivative of the image and with it, can find the edges. For this method, the second order derivative filter must be used. For example : Laplacian Filter.

According to [2],here are three steps of the edge detection algorithm. They are :

- **Filtering :** For edge detection, it is required to use a suitable smoothing filter. The filters sharpen the edges and ignore the unnecessary information. It is often utilized to improve the functioning of an edge detector against noise. The more filtering is applied, however, the greater the loss of edge strength.
- **Enhancement :** To be able to better detect edges, changes in the intensity in the area surrounding a point must be determined. Pixels in which a significant change in intensity occurs are emphasized by enhancement, which is usually applied by calculating the gradient magnitude.
- **Detection :** Though many points in an image have a nonzero value for the gradient, not all of these points are actually edges. Because only points with strong edge content are desired, a method must be applied to determine which points are actual edge points. Thresholding is often utilized to do so.

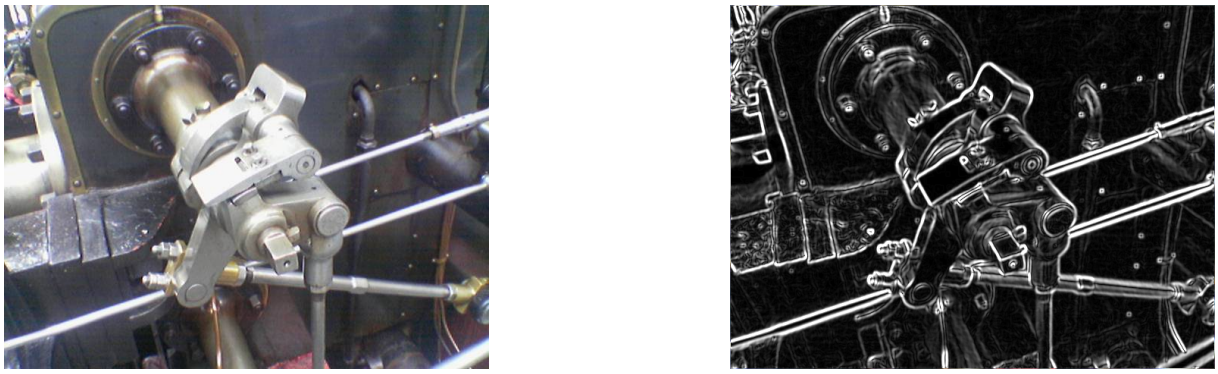
Well known smoothing filters are :

- Sobel-Operator
- Canny Edge Detector
- Laplacian-Filter
- Prewitt-Operator

In this master thesis, the Sobel Operator was utilized, so it will be described in more detail.

2.3.1 Sobel Operator

One of the most used edge detectors at image processing and computer vision is Sobel Operator. The Sobel Operator uses vertical and horizantel masks. These masks are used odd-sized square matrices and they are generally 3x3 matrices.



(a) Original Image **(b)** Sobel Operator applied Image

Abbildung 2.4: Sobel Operator[3]

2.4 Hough-Transformation

2.4.1 Standart Hough-Transformation

2.4.2 Probabilistic Hough-Transformation

2.5 K-Nearest Neighbors Algorithm

K-Nearest Neighbor(KNN) is an non-parametric lazy learning algorithm. Non-parametric technique means, that it doesn't make any assumptions on the underlaying data distribution. In the definition of KNN was used the term of 'lazy learning algorithm'. It means, it doesn't use the data training points to do any generalization. In other words, there is no explicit training phase or it is very minimal. It also means that the training phase is pretty fast. Most of the lazy

algorithms - especially KNN - makes decision based on the entire training data set. On the other hand, KNN is one of the top 10 data mining algorithms[4].

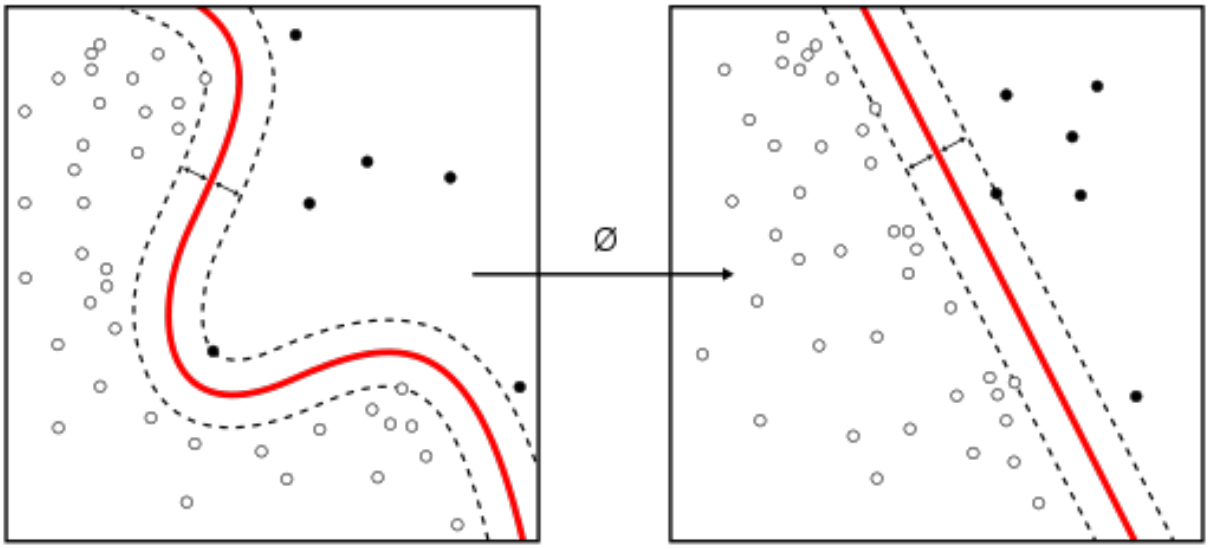


Abbildung 2.5: K-Nearest-Neighbors Algorithm[5]

K-Nearest Neighbors Algorithm has advantages and disadvantages. According to [6],the main advantages of KNN are simplicity, effectiveness, intuitiveness and competitive classification performance in many domains. On the other hand, KNN can have poor run-time performance when the training set is large. It is very sensitive to irrelevant or redundant features because all features contribute to the similarity and thus to the classification. The computation cost is also quite high because we need to compute distance of each query instance to all training samples.

2.6 Curve Fitting

Curve fitting is used to find the 'best fit' line or curve for a series of data points. Curve fitting produces mostly mathematical equation that can be used to find points anywhere along the curve.[7] They are several different types of curve fitting. Some of them are; linear, exponential, polynomial, exponential, power, logarithmic, etc. In this master thesis, polynomial curve fitting was used. Polynomial curve fitting differs from order of the polynomial. Polynomial curve fittings are called with different names with depending on their orders. For first order polynomial curve fitting is called as linear regression, for second order quadratic regression, for third order cubic regression.

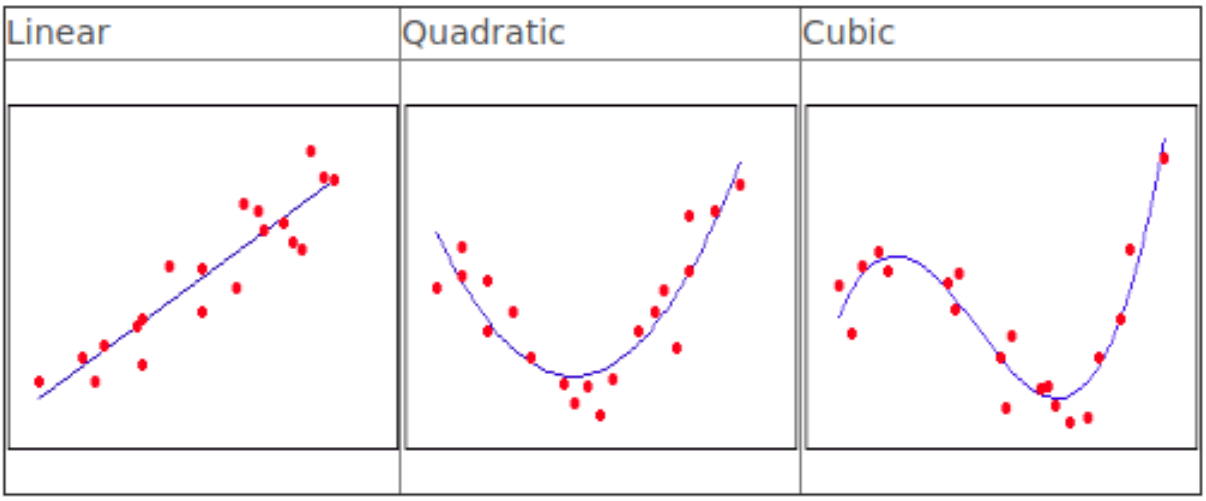


Abbildung 2.6: Types of Polynomial Curve Fitting[8]

3 Implementation

3.1 Test Track

As previously mentioned, the medium-term goal of this master thesis is attending the Carola-Cup at Braunschweig University, so the test truck was prepared according to the Carola-Cup criteria by Nicolas Acero Sepulveda, who also did his bachelor thesis with this model automobile. For this test truck, two black PVC floor carpets were used and on these floor carpets, the lanes of the track were made by using white electrical tape. The straight part of the track was made on one of these PVC floor carpets and the curved part of track was made on the second PVC floor carpet. The straight part of the track is approximately 2 meters long and the curve radius of the curved part of test track is approximately 1 meter. This curve is the tightest curve at Carola-Cup, so with this, the test track can be tested in the worst case scenario. In the Carola-Cup competition, the track is much larger; however, for the purposes of this master thesis, a larger test track in not needed.

Abbildung 3.1: Test Truck

3.2 Hardware

3.2.1 Model Auto

During the course of this master thesis, a model automobile was being used which was prepared for the Projectseminar Echtzeitsysteme at Technical University of Darmstadt. The chassis, steering mechanism, power train, and engine control were derived from the model-building of a Japanese company, Tamiya. The maximum velocity of the model automobile is approximately 1 m/s and the minimum steering radius is around 90 cm.

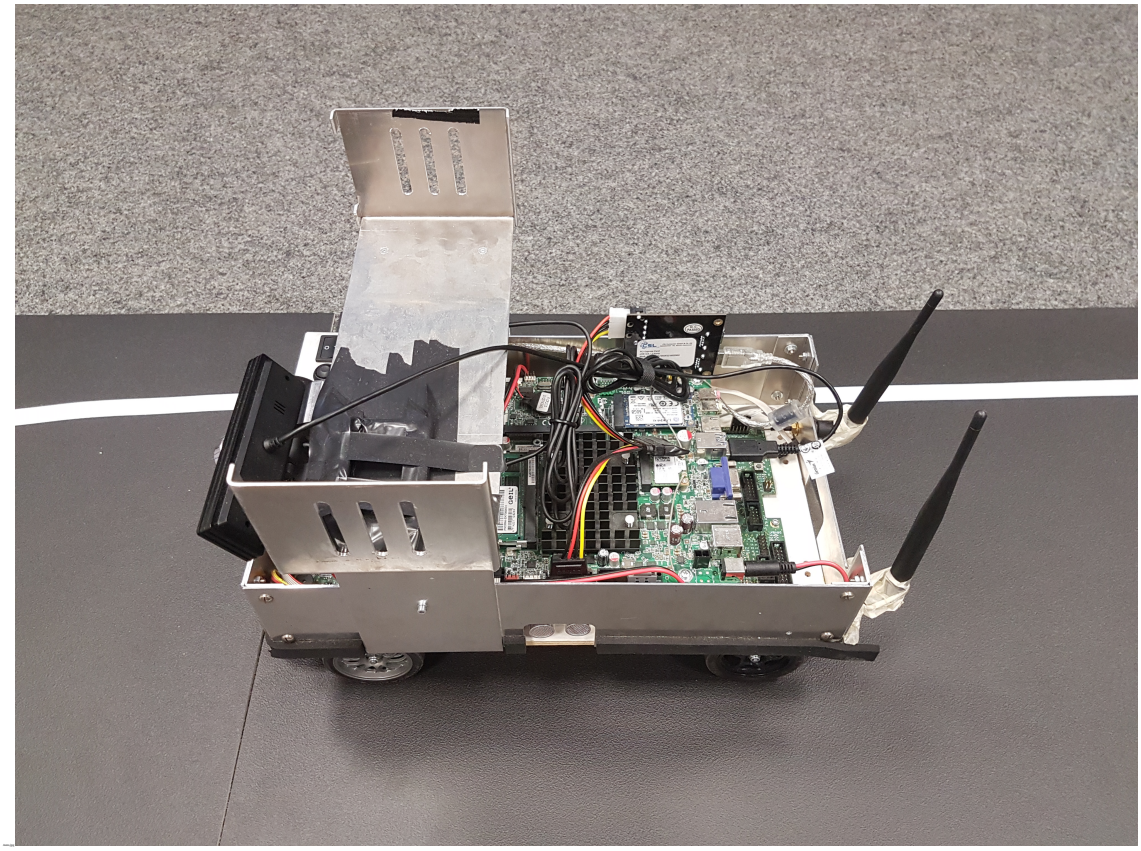


Abbildung 3.2: Model Auto

3.2.2 Microcontroller and Main Board

In this model auto, there is a microcontroller and a main board. The microcontroller is used for controlling steering and receiving the measurements from ultrasonic sensors and hall effect sensors. The 16-bit microcontroller is from MB96300 series from Fujitsu company.

The main board on the model car is from PD10BI-MT ThinMini-ITX series from MiTAC company. This main board communicate with the microcontroller over via UART interface through USB connection. On this main board, there is an Intel Quadcore-Processor and an Intel HD Graphics card. Furthermore, there is a 8 GB DDR3-1600 RAM and 1Gbit/s Ethernet, VGA, HDMI,

USB 2.0/3.0, SATA ports and an Intel Dual Band Wireless AC 7260 Network adapter, which is connected to two external WLAN antennas. A 60 GB Kingston SSD-Harddisk is connected over an integrated PCI-Express Port. A 3200 mAh Li-Fe battery is used as power supply.

3.2.3 Camera

The camera is one of the main components of lane detection and accordingly, autonomous driving. For this thesis, I had to research the most suitable camera because all cameras have different properties.

At the beginning of the Projectseminar Echtzeitsysteme, the Logitech C270 HD Webcam was being used. The resolution of the camera is 1280x960 pixels and the Frame per Second (FPS) value is 30 Hertz (Hz) at a 640x480 pixel resolution. The field of View (FOV) is just 60 degrees. The problem with this camera is that if there is a curve, the camera cannot see all of the lanes, and thus is not very suitable for lane detection. When I started my master thesis, there was a Kinect v2 camera on the model car. The Kinect v2 camera was developed by Microsoft and released in 2013. This camera has a depth sensor with a resolution of 512x424 pixels and its FOV is 70x60 degrees. The FPS value is 30 Hz at a 512x424 pixel resolution. This camera also has a color camera with resolution of 1920x1080 pixels and a FOV of 84.1x53.8 degrees. The FPS value is 30 Hz at a 1920x1080 pixel resolution. This camera had two main disadvantages for this master thesis. The first disadvantage is the FOV value of camera. This value is better than the value of Logitech C270 camera but it is still not enough for curve lane detection. The second main disadvantage is the location of the color camera. The color camera of this camera is not in the middle of camera, but rather, on the right. This is a disadvantage for us because when there are curves going left as opposed to right, the camera is unable to see the left and even perhaps the middle lane of the truck. Thus, this is problematic for lane detection.

Due to these reasons, I had to choose a camera which has a sufficiently high FOV value. After doing research, I decided that the Genius Widecam F100 camera is the best choice for this master thesis because this camera has a FOV value of 120 degrees and it can also be used with the Linux Operating System. The resolution of this camera is 1920x1080 pixels and the FOV value is 120 degrees. The FPS is 30 Hz at a 1920x1080 pixel resolution. With this camera, it is possible to detect most if not all lanes, including when there are curves.

Abbildung 3.3: Genius 120-degree Ultra Wide Angle Full HD Conference Webcam(WideCam F100)

3.3 Software

In this chapter, the software algorithms will be focused, which are defined in this master thesis. Through program flow charts and explaination of all steps of program flow charts, it will be tried to explain algorithms better. For finding the best solution, 5 different source codes versions(?variants/?methods) were generated. For all these source codes,the computing times were calculated and compared which solution can detect the lanes better. In next pages, there are detailed explanations of used versions(?variants/?methods). The development environment and the used softwares will be also described , which are used in master thesis.

3.3.1 Development Environment and Related Softwares

As also mentioned at subsection 3.2.2, in this project the introduced main board was used. One of the compact and fast version of Linux 16.04 operating system, *Lubuntu* was installed in this main board.

The version *Kinetic* of ROS was used for implementation of this master thesis. ROS is the abbreviation of **R**obotic **O**perating **S**ystem, which is a robotics middleware (i.e. collection of software frameworks for robot software development). At ROS wiki page[9], ROS is defined that, ROS is an open-source, meta-operating system for your robot. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package

management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

For using prepaid image processing functions, an open source computer vision and machine learning software library was used, which is called as OpenCV. According to OpenCV website[**?**], there are more than 2500 optimized algorithms in OpenCV library and OpenCV has more than 47 thousand people of user community.

ROS can be programmed by programming languages Python, C++ or Lisp and OpenCV can be programmed by programming languages Python or C++. In this master thesis, C++ was used.

3.3.2 Preprocessing :

There are so different possibilities for lane detection algorithms. Of course, each of them has some advantages and disadvantages. In this master thesis, some different methods were defined and in this chapter, these methods will be explained in detail.

In all these methods, some processes are common, which is called as preprocessing phase. At the beginning,*the frames are gotten from the camera via ROS-Topic.* ROS uses different image formats but OpenCV uses as another image format Matrix(Mat) object. This taken frame via ROS-Topic must be converted from ROS image data type to Mat object. For that converting, a ROS-Package were used, which is called as *cvbridge*[10]. cvbridge converts to ROS image format to Mat Object which is the OpenCV Image Format.

Mat is a class which has two parts. The parts are a matrix header and a pointer to the matrix containing the pixel values. The matrix header contains the informations like the size of matrix, storing method and etc. It has always fixed size but the size of matrix is variable from image to image.

There are so many methods, which can store the pixel values to the Mat object. In this case, the color space and used data type can be choosed. For gray images, it is easy to choose the color space because there are just two colors : black and white. With changing the density of colors(black and white), it is possible to create many shades of gray. There are more methods for colorful images. Colorful images have generally three or four channels. These three channels are used for RGB color values. The RGB colors are based on red, green and blues colors which can be detected all colors by human eyes. For transparency of a color can be used a fouth channel which is calles as alpha(A).

There are also another color formats, which have some advantages[11].

- RGB format is so similar to human eye use, but at OpenCV display system uses BGR format which has another row of colors than RGB format
- HSV and HLS formats are more natural way to describe colors. They decompose colors into their hue, saturation and value/luminance components. Another advantage of HSV and HLS make less sensitive to the light conditions of the input image.

- At JPEG image formats is used YCrCb format.
- if the distance of a given color to another color want to be measured, CIE L*a*b* format is more suitable than others.

After the frame from Camera via ROS-Topic was received, the colorful frame has to be converted to grayscale frame. For detection lanes, Hough Transformation is used and for Hough Transformation, grayscale format of input image is needed. To convert a frame from BGR format to grayscale format brings some advantages, the main advantage is the processing time. Normally colorful frame matrix content has three or four channels but at grayscale frame matrix content has just one channel so grayscale frame matrix size is much smaller compare to colorful frame matrix size. Because of this reason the image processing time is much more smaller at grayscale format compare to BGR format. For converting BGR formatted frame to grayscale formatted frame, *cvtColor* function from OpenCV is used.

For stabile lane detection, the light conditions must be considered. Because of this reason, after converting the frame with BGR format to grayscale format, the lightest and darkest pixels were searched. After finding the lightest and darkest pixels, light conditions can be estimated approximately. These values will be used in the next step. After this processing, a filter is applied to all frames which tranforms an image into a binary image by transforming each pixel according to whether it is inside or outside a specified range. The user chooses a threshold value to process. If a pixel is greater than this value, it is assigned an 'inside' value. Otherwise it is assigned an 'outside' value. Depending on the lightest and darkest pixel values, the threshold value changes. Through this dynamical parameter, the lanes are more clear detectable and noises can be cancelled more successful.

After threshold filter, an edge detection filter must be applied. In this master thesis, the Sobel operator is used which explained at Chapter 2 in detail.

This preprocessing part is common for all cases but after this process, there are differences at cases.

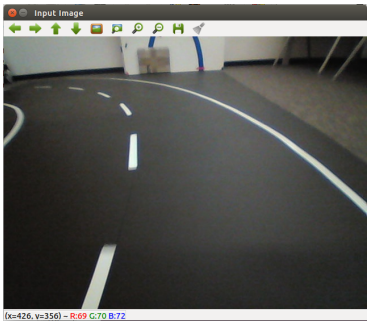
3.3.3 Method 1/Case 1 : Hough Transformation + Rectangle + Curve Fitting + IPM (v0.9.2)

When the preprocessing part is over, the lanes must be detected. For the finding the position of lanes, the Standart Hough Transformation is used. But the Standart Hough Transformation didn't used to all of the frame, the frame were cropped. This cropping has a reason and there is an advantage thanks of cropping. It will be explained soon in detail.

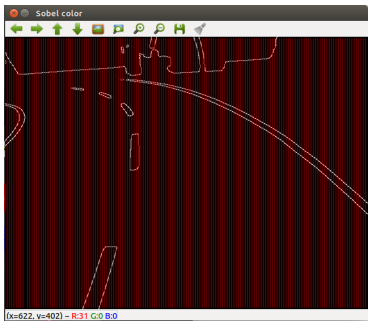
We get frames from camera 640x480 pixels resolution. It was the default value at that camera. The resolution of the camera can be increased by its settings, but it is not possible to decrease. For decreasing the resolution of the camera, there is another method and it is used in another case so it will be explained when it is used in another case. In this case, the minimum resolution

is used, which the camera allows us. There are some reasons. The most important reasons is, if the resolution is increased, there are more pixels so the computing time increases too. High computing time is something, which we don't want to have.

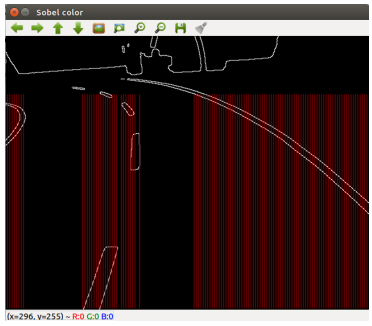
Height of the frame has 640 pixels, but as before mentioned, the frame was cropped. As seen at Figure 3.4b, if there is a curve, far points from camera won't be the part of truck. There can be some stuffs which are not relevant with truck and lanes so these stuffs caused to produce Standart Hough Transformation lines. In this thesis, the Standart Hough Transformation were used, if there is an edge which could detected by Sobel Operator, produces a red line so the first 100 pixels of frame height were took off and the last 540 pixels were used. As seen at Figure 3.4c, the red lines are shown just last 540 pixels of frame height so they don't cover the irrelevant informations at the frame. If there is a lane, the red lines are so close each other but if there is no lane, there are so many distance(at least 50 pixels) between red lines. At Figure 3.4c, there are three different groups of red lines. All these group include a lane. Thanks of Standart Hough Transformation, we know that, between which vertically pixels stand the lanes.



(a) Original Image



(b) Hough Transformation without Cropping Image

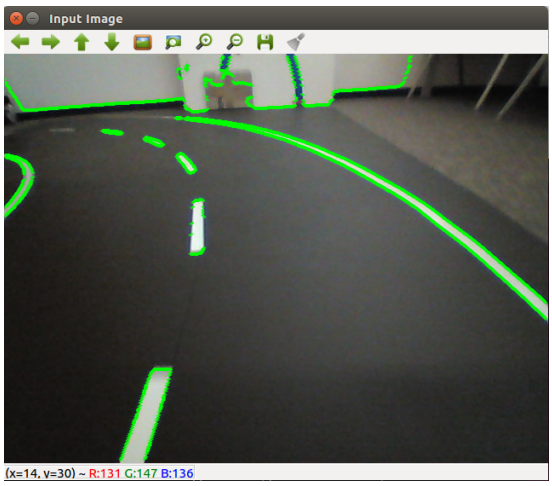


(c) Hough Transformation with Cropping Image

Abbildung 3.4: Detecting Lane Positions

After this processing, we have to find the starting points of lanes and the all pixels which are on the lanes. For that, we have to use Probabilistic Hough Transformation. Probabilistic Hough Transformation is a bit different than Standart Hough Transformation. Standart Hough Transformation is more suitable for straight lanes but if there is a curve, Standart Hough Transformation doesn't work enough good. As before mentioned, a line can be represented as $y = mx+c$ or in parametric form, as $\rho = x\cos\theta + y\sin\theta$ where ρ is the perpendicular distance from origin to the line, and θ is the angle formed by this perpendicular line and horizontal axis measured in counter-clockwise. But it is different at Probabilistic Hough Transformation. A line is represented by two or more points. If Probabilistic Hough Transformation finds at least two points from the same lane, it represents beginning and ending points of these lines.

In this master thesis, the lines are not shown because we don't see the Probabilistic Hough Lines. We need just these points(pixels), which are on the lanes.



(a) Original Image with Probabilistic Hough Transformation

(b) A Image with Sobel Operator and Probabilistic Hough Transformation

Abbildung 3.5: Probabilistic Hough Transformation Points

At Figure 3.5a and Figure 3.5b, Probabilistic Hough Transformation Points(green pixels) are shown. They are so many pixels, which are found by Probabilistic Hough Transformation. There are some parameters at this function in OpenCV. With changing the parameters of Probabilistic Hough Transformation in OpenCV, less Hough Points on the lanes can be found but finding too few Hough points can cause some problems while detecting the lanes. On the other hand, finding so many Hough points need also more computing time, this is also a situation which we don't want to have. So in this case, the parameters of Probabilistic Hough Transformation fuction in OpenCV must be optimized. Thanks of optimization, the best solution (less computing time and good lane detection) is found.

Now, we know, that between which pixel columns stand lanes so we can find the start pixels of lanes. For that, we have to find the Hough pixels for all lanes(right, middle and left lane) at the lowest part of the frame. Each Hough points which are on a lane, can be compared with each other so the Hough point of the lane, which is the lowest part of the frame can be found. It is the starting point of the lane. This process must be done for all lanes which can be seen on the frame.

Next part of the project is getting the Hough points which are relevant the lanes. For each lane, the Hough points must be grouped. For getting Hough points which are relevant with lanes, 'rectangle' method is used, which is named by me. At rectangle method, a rectangle is drawn and save coordinates of all Hough points in that rectangle. All Hough points are saved in a vector and the uppest Hough point at the rectangle must be found. From that point, another rectangle must be drawn but the size of rectangles are going to progressively smaller. Because the objects are going to seem smaller when they are far away from the camera. There is an exception at middle lane. Middle lane has dashed lines so the rectangles at middle lane must be bigger than at the left and right lanes.

Here must be a picture of rectangles

Last part of this case is curve fitting. Curve Fitting is an algorithm which gives a mathematical description and this mathematical description has the best fit to a series of data points. But in this master thesis, usage of curve fitting is changed a bit. At curve fitting, the coordinates of Hough points are used but we changed the x and y axises of these Hough points because y-axises of these coordinates have more range than x-axises so this swapping raises the stability of the curve fitting.

After using the Hough points as input, three different mathematical equation are produced. One of these equation is for the left lane, the other one is for the middle lane and the last equation is for the right lane. While producing of these equations, of course just relevant Hough points are used. For example, for left lane curve equation, the Hough points from the left lane were used.

Next part of this case is plotting the curves which are produced by curve fitting function. We start from 0th pixel to 480th pixel vertically and the output values of equation for each vertically pixels are found. For each lane, the curves are plotted in different color.

At the last part of this case is Inverse Perspective Mapping. End of the project, we want to get mathematical description of lanes from bird's-eye view. So the perspective of lanes must be changed from the camera side to the top of the truck side. For that, we have to use 'findHomography' function from OpenCV. Thanks this function, all curve points can be converted to perspective of the top of the truck side. All pixels could also be converted from camera perspective to the top of the truck perspective but in that case, we had to convert 640x480 pixels, respectively 307200 pixels. But in this case, we convert just 480 pixels for each lane, it means, we convert totally 1440 pixels. Because of this reason, this case is efficienter that to convert all pixels.

Here must be a picture of curve fitting

all of the pixels in the 100th column

3.3.4 Method 2/Case 2 : IPM + Hough Transformation + KNN + Curve Fitting(v0.4.2)

As we before mentioned, preprocessing part is common for all cases. In this case, frames which are taken from camera, are convert directly from camera perspective to top of the truck perspective.At Section 3.3.3 (Case 1), we convert the just Curve Fitting pixels from the camera perspective to the top of the truck perspective. To convert 640x480 pixels, respectively 307200 pixels take a bit longer time compare to the Case 1. Original frame which can be seen at Figure 3.6a is converted to the frame which can be see at Figure 3.6b by OpenCV 'findHomography' function.

(a) Original Image **(b)** A Image with Inverse Perspective Mapping

Abbildung 3.6: Inverse Perspective Mapping

Next step of this case is that finding between which vertically pixels stand the lanes. At Case 1(Section 3.3.3 we had to apply Standart Hough Transformation to the frame whose the first 100 pixels of frame vertically were took off(380x640 pixels). After IPM implementation, we don't need to apply Standart Hough Transformation to the so big part of the frame. Just last 160 vertically pixels (160x640 pixels) is enough to find all positions of lanes.

After finding between which vertically pixels stand the lanes, the Probabilistic Hough Transformation was used.

4 Evaulation and Discussion

5 Related Works

6 Conclusion

Literaturverzeichnis

[1] Lin, Chien-Chuan und Ming-Shi Wang: *A Vision Based Top-View Transformation Model for a Vehicle Parking Assistant*. Sensors, Department of Engineering Science, National Cheng Kung University Taiwan, No.1, University Road,Tainan City 701, Taiwan, 2012.

[2] Ramesh Jain, Rangachar Kasturi, Brian G. Schunck: *Machine Vision*. McGraw-Hill, 1995.

[3] Wikipedia: *Sobel operator*, October 2017.

[4] Xindong Wu, Vipin Kumar, J. Ross Quinlan Joydeep Ghosh Qiang Yang Hiroshi Motoda Geoffrey J. McLachlan Angus Ng Bing Liu Philip S. Yu Zhi-Hua Zhou Michael Steinbach David J. Hand Dan Steinberg: *Top 10 algorithms in data mining*. Springer-Verlag London Limited 2007, 2007.

[5] Wikipedia: *k-nearest neighbors algorithm*, August 2017.

[6] Sadegh Bafandeh Imandoust, Mohammad Bolandraftar: *Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background*. S B Imandoust et al. Int. Journal of Engineering Research and Applications, 3:605–610, October 2013.

[7] *The KaleidaGraph Guide to Curve Fitting*.

[8] Frost, Jim: *Curve Fitting with Linear and Nonlinear Regression*, August 2013.

[9] Dirk, Thomas: *What is ROS*, May 2014.

[10] Wiki, ROS: *cv bridge ROS Wiki*, December 2013.

[11] team opencv dev: *Mat The Basic Image Container*, October 2017.