



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

RADAR AND VIDEO FUSION

FÚZE RADARU A VIDEO

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR

AUTOR PRÁCE

ONDŘEJ GALETA

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. LUKÁŠ MARŠÍK

BRNO 2023

Bachelor's Thesis Assignment



140523

Institut: Department of Computer Graphics and Multimedia (UPGM)
Student: **Galeta Ondřej**
Programme: Information Technology
Specialization: Information Technology
Title: **Radar and Video Fusion**
Category: Signal Processing
Academic year: 2022/23

Assignment:

1. Study available literature related to fast-ramp FMCW radar signal processing, objects tracking, and radar-video fusion.
2. Select an algorithm suitable for extraction of information about objects such a speed, distance, and position. As well, select an algorithm for detection in video.
3. Design or select a method allowing, based on extracted information from radar and video, object tracking within the scene.
4. Implement the designed method.
5. Collect data using a suitable HW in real environment. Carry out a series of tests to evaluate performance and reliability of the system.
6. Discuss achieved results and suggest possible future work as well.

Literature:

- TI IWR6843: Single-chip 60-GHz to 64-GHz intelligent mmWave sensor integrating processing capability (<https://www.ti.com/product/IWR6843>)

Detailed formal requirements can be found at <https://www.fit.vut.cz/study/theses/>

Supervisor: **Maršík Lukáš, Ing.**
Head of Department: Černocký Jan, prof. Dr. Ing.
Beginning of work: 1.11.2022
Submission deadline: 10.5.2023
Approval date: 3.11.2022

Abstract

The main goal of this work is to increase the quality of object tracking in three-dimensional space using the fusion of radar and video data obtained from traffic monitoring scenes. The thesis presents several methods dealing with collecting point pairs from both sensor coordinate systems for spatial calibration of sensors, with an emphasis on automation. It solves the association problem of multiple detected objects at once using a modified Hungarian algorithm. The work also shows some methods of predicting the distance of objects detected by video using a dataset obtained from radar. Output from the fusion model provides spatially extended tracks and more accurate counting of tracked objects than radar or video models alone, which can be used for further analysis of traffic in the observed scene.

Abstrakt

Hlavním cílem této práce je zvýšení kvality sledování objektů v třírozměrném prostoru pomocí fúze radarových a video dat získaných ze scén sledujících dopravu. Práce uvádí několik metod zabývajících se sbíráním párů bodů ze souřadnicových systémů obou senzorů pro prostorovou kalibraci s důrazem na automatizaci. Práce dále řeší problém asociace více detekovaných objektů naráz za pomoci modifikovaného Maďarského algoritmu. Práce rovněž ukazuje některé způsoby predikce vzdálenosti objektů detekovaných videem pomocí datasetu získaného z radarových dat. Výstup z fúzního modelu poskytuje prostorově rozšířené sledování dráhy a přesnější počítání sledovaných objektů než radarové a video modely samotné, což může být využito pro další analýzu dopravy v zkoumané oblasti.

Keywords

radar, millimeter-wave radar, video, fusion, object detection, object tracking, spacial calibration, time synchronization, depth prediction, Hungarian algorithm, deep learning

Klíčová slova

radar, milimetrový radar, video, fúze, detekce objektů, sledování objektů, prostorová kalibrace, časová synchronizace, predikce hloubky, Maďarský algoritmus, hluboké učení

Reference

GALETA, Ondřej. *Radar and Video Fusion*. Brno, 2023. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Lukáš Maršík

Rozšířený abstrakt

Hlavním cílem této práce je zvýšení kvality sledování objektů v třírozměrném prostoru pomocí pozdní fúze radarových a obrazových dat získaných ze scén sledujících dopravu. Myšlenkou této práce je zvolit vhodnou metodu pro prostorovou kalibraci, asociaci objektů detekovaných radarem a videem současně a získat ze scény takovou množinu informací, která by umožnila zjistit pozici objektu v třírozměrném prostoru i za předpokladu, že by byl objekt detekován pouze na videu.

Metody u prostorové kalibrace jsou zaměřeny na automatizaci sbírání párů objektů detekovaných videem a radarem, kterými se následně pomocí přímé lineární transformace počítá transformační matice. Jednou z metod je aproximovat transformační matici pomocí vnitřních a vnějších parametrů a následně provést sběr párů objektů pomocí asociace se znalostí této aproximované transformační matice. Tato metoda se ukázala jako potenciálně přesná při dobře zvolených parametrech. Zvolení vhodných parametrů je ovšem i přes navržený nástroj usnadňující práci poměrně náročné. Dalším z použitých přístupů je rozdělení zkoumané scény do dvou podoblastí pomocí parametrů jak pro objekty detekované na videu, tak pro objekty detekované na radaru. Parametry pro radar jsou zvoleny jako směr pohybu objektu a pro video vertikální čára dělicí obraz na dvě části dle podoby vozovky v obraze. Rozdělením scény se problém mění do podoby, kdy pouze v dané podoblasti musí být pouze jeden objekt detekovaný videem a radarem a tím pádem je asociace jednoznačná. Přístup byl ještě rozšířen na čtyři podoblasti, kde se scéna rozdělila pomocí horizontální dělicí čáry a odhadu rozsahu vzdálenosti, která tato čára reprezentuje na vozovce. Tím byl počet podoblastí rozšířen na čtyři. Tato metoda prokázala větší stabilitu než metoda sběru párů pomocí aproximované matice a dosáhla větší přesnosti než metoda dvou podoblastí na testovacím vzorku. Přesnost nejlepší metody je přibližně 12 pixelů při testovacím vzorku o rozlišení 640×480 pixelů.

Další část práce se zabývá predikcí hloubky. V případě této práce je tím primárně myšlena predikce pozice objektů detekovaných videem v souřadnicovém systému radaru. Pro predikci hloubky byli implementovány dva přístupy. První přístup se snažil hloubku dopočítat pomocí transformační matice a rovnice popisující aproximovaný povrch vozovky, která je vytvořena s pomocí objektů detekovaných radarem. Tímto přístupem totiž vzniká vypočitatelná soustava rovnic, pomocí které lze pozici vypočítat. Současným problémem této metody je současná nepřesnost detekce radarových objektů. Ty se totiž i přes to, že se pohybují po vozovce, detekují v různých výškách a tím pádem je predikce pozice poměrně nepřesná. Tento problém se dá omezeně eliminovat predikcí dvou ploch, kdy jedna je pro menší vozidla a druhá pro větší, protože výška detekce je dle všeho ovlivňována reálnou velikostí vozidla. Další použitý přístup je predikce vzdálenosti za použití hluboké neuronové sítě, kdy vstupem byli souřadnice objektů ze soustavy souřadnic používané videem a výstupem souřadnice ze soustavy souřadnic používané radarem. Jako dataset posloužila množina asociovaných párů souřadnic radarových a video objektů. Při dobře zvoleném modelu a parametrech se na zkoumaném testovacím vzorku dosáhlo průměrné přesnosti přibližně 2 metrů, což je lepší výsledek než u předchozího přístupu.

Asociace je prováděna pomocí modifikovaného Maďarské algoritmu ve dvou fázích. Byla zvolena modifikace algoritmu, která se dokáže vypořádat i s rozdílným počtem objektů detekovaných na radaru a videu. První fáze se zabývá asociací objektů detekovaných v jeden moment. Skóre v první fázi je vzdálenost detekovaných objektů v souřadnicovém systému videa, přičemž je algoritmus rozšířen o maximální hodnotu skóre, které je přípustné, aby byla zajištěna robustnost fúzního systému. Druhá fáze se zabývá asociací celých drah. Skóre ve druhé fázi je implementováno pomocí počtu dílčích asociací jednotlivých drah.

Použitá metoda se dokázala více přiblížit k reálnému počtu vozidel v testovacím vzorku než dokázaly jednotlivé senzory samostatně. Průměrná délka detekovaných drah se navýšila přibližně o čtvrtinu oproti průměrné délce drah z videa. Ve srovnání s průměrnou délkou radarových drah je průměrná délka z fúze skoro dvaapůlkrát větší.

Radar and Video Fusion

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of Ing. Lukáš Maršík. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Ondřej Galeta
May 8, 2023

Acknowledgements

Studying at the faculty gave me a lot of knowledge. Personally, I most value the way of thinking that some lecturers tried to convey to us. Technical thinking is knowledge that cannot simply be read from an encyclopedia. So my thanks go to the lecturers who did not see the lectures as something they had to do, but rather as something meaningful. My thanks also go to the supervisor, who was able to advise me in some situations that happened during this work. Thanks also go to all the friends I made thanks to studying at the faculty and who helped me with group projects. Last but not least, thanks go to my entire family, who supported me in all aspects of my studies.

Contents

1	Introduction	2
2	Current state of radar processing	3
2.1	Principle of a frequency modulated continuous wave radar	3
2.2	Intermediate frequency signal processing	4
2.3	Radar data representation	7
2.4	Group tracker	8
3	Current state of video processing	10
3.1	Video object detection and tracking	10
3.2	Intrinsic and extrinsic transformation matrix	12
3.3	Object depth prediction	13
4	Current state of radar and video fusion algorithms	15
4.1	Radar and video spacial calibration	16
4.2	Radar and video time synchronization	17
4.3	Object association	18
5	Fusion architecture design	20
5.1	Video and radar branch	20
5.2	Fusion module	21
6	Implementation	23
6.1	Radar and video detection and tracking	23
6.2	Spacial calibration	25
6.3	Predicting depth from video	27
6.4	Video and radar object and track association	29
7	Testing and evaluation	31
7.1	Testing spacial calibration methods	31
7.2	Testing association	35
7.3	Testing depth prediction methods	37
8	Conclusion	40
	Bibliography	42

Chapter 1

Introduction

Video systems can detect various objects and track them. These systems are used on highways, where there is a need to oversee objects or collect data about them for further analysis. Unfortunately, the performance of video systems alone is limited. Their detection, classification, and tracking accuracy are good only under certain conditions. On the other hand, there is a wave of new radar chips, which are still in development and hopefully await their peak in popularity. Nowadays, these systems are usually not as accurate as video systems, so they cannot fully replace video in most cases. However, their data brings a new view to the observed scene, because of their nature of collecting.

One of the possible answers is the fusion of these systems into one. Combined systems have much more information about the investigated environment. Both sensors collect different data about the same objects, which could lead to better performance and robustness of the whole system. Unfortunately, fusion brings several problems that need to be solved. To profit from the fusion, data from both sensors must be time synchronized, spatially transformed to a uniform coordinate system, and associated if there is more than one detection at a time. These issues have not been solved in every possible environment. Despite the great interest, many works still bring new ideas on how to proceed in fusion. The current chance to bring something new to this field is also a reason to select this topic for a thesis. The second reason is the prediction made by the author that with the increased popularity of radar and video fusion systems, there will be increased demand for people who understand this topic.

One of the main goals of this work is to present an overview focused on radar, video, and fusion processing mechanisms. Based on the overview, select algorithms that are able to track objects using radar data and detect objects in video and test them on data collected in a real-world environment. Considering previous paragraphs, there is also an idea that suggests that fusion tracking should perform better than video or radar alone.

Chapter 2 introduces the fundamentals of FMCW radar signal processing. Chapter 3 is focused on video processing and describes some of the current capabilities of video systems. Chapter 4 theoretically describes some radar and video fusion algorithms. Chapter 5 presents the proposed design and explains its key features. It is focused on proposed solutions to current state problems. Chapter 6 describes the implementation of the designed system. It presents implemented approaches and mechanisms used to compute or predict the required output. Chapter 7 deals with testing and evaluation. It describes testing procedures, input parameters for each experiment, and the evaluation of the achieved results.

Chapter 2

Current state of radar processing

This chapter is focused on FMCW radar technology. The main task is to present a brief overview of how radar data can be processed from the synthesizer to a structure containing detected objects and tracking their path.

2.1 Principle of a frequency modulated continuous wave radar

A Frequency modulated continuous wave radar or FMCW radar is a special case of general radar. FMCW radar operation scheme is described in Figure 2.1. A signal generator, called a synthesizer, generates an electromagnetic signal, called chirp. The chirp is then transmitted by a transmitting antenna (TX). Signals emitted to space are then reflected from objects and captured back by a receive antenna (RX). The signal from the receive antenna is combined with the transmitting signal in a component called a mixer to produce an intermediate frequency (IF) signal. [6]

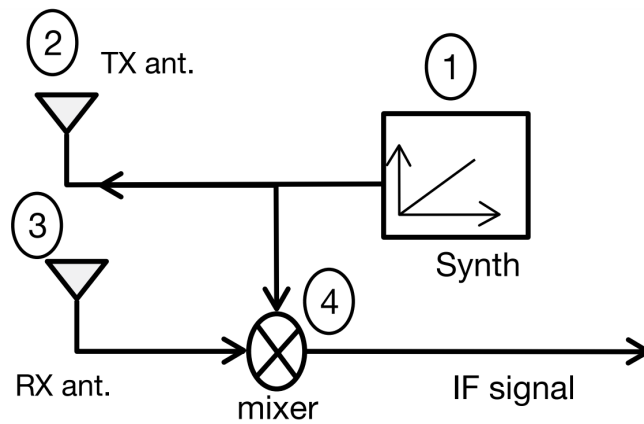


Figure 2.1: FMCW radar block diagram (taken from [6])

Meaning of chirp

Chirp is a signal with an increase in linear frequency over time. Its features can be described by its bandwidth, carrier frequency, and duration. Duration describes the transmission time of the signal. The bandwidth is the difference between the initial and final frequencies of a chirp. Carrier frequency, also known as starting frequency, is the initial frequency of

a chirp. With recent progress in this field, the maximum bandwidth range is ordinarily around 4 GHz a carrier frequency is up to 79 GHz. Both of these frequencies lead to better radar performance. The higher bandwidth theoretically increases the possible range resolution. [1]

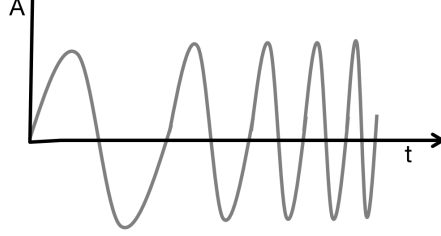


Figure 2.2: The amplitude of a chirp signal varies in time (taken from [6])

2.2 Intermediate frequency signal processing

Unless otherwise stated, this section is mainly based on the article *The fundamentals of millimeter wave radar sensors* [6]. The signal leaving the mixer is known as an intermediate frequency (IF) signal. The signal is still in analog form. For computational use, it has to be sampled to a discrete form.

Range measurement

The range could be computed from time and speed. In this case, the time is meant as a time which signal needs to travel from a transmit antenna to a receive antenna, and the speed is meant as the speed of light, which is equal to the speed of the emitted signal. To be specific, the mixer on Figure 2.1 receives a signal described as Equation 2.1 directly from the synthesizer and a signal described as Equation 2.2 from the RX antenna. Then both signals are combined to form a signal described by Equation 2.3.

$$x_1 = \sin(\omega_1 t + \phi_1) \quad (2.1)$$

$$x_2 = \sin(\omega_2 t + \phi_2) \quad (2.2)$$

$$x_{out} = \sin[(\omega_1 - \omega_2)t + (\phi_1 - \phi_2)] \quad (2.3)$$

The problem could also be shown graphically in the upper part of Figure 2.3. In this Figure, τ represents the time delay between the transmitted (TX) and received (RX) chirps. Variable τ could be derived as Equation 2.4. The bottom part in Figure 2.3 represents the subtraction of chirps above. It also defines the validity of the IF signal, which corresponds to τ . IF signal is valid only when the TX and RX signals overlap. IF signal is constant because the size of the gap $S\tau$ between both signals does not change over time. This assumption is true only if the object causing the reflection is not moving. However, the error made by this error is neglected or corrected when Doppler-FFT is processed.

$$\tau = \frac{2d}{c} \quad (2.4)$$

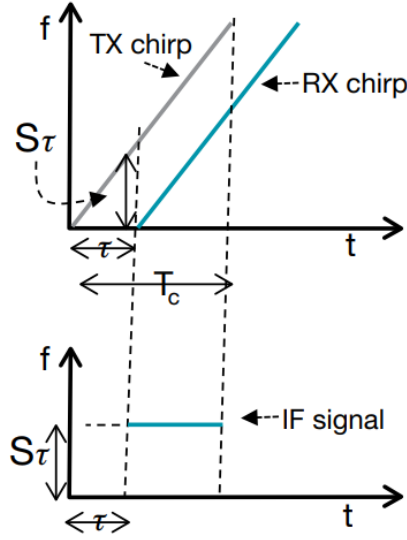


Figure 2.3: IF frequency reflection graph (taken from [6])

In summary, the final equation that describes the reflection signal is shown in Equation 2.5, where a is the amplitude of the signal, and variables f_0 and ϕ_0 are further described in Equation 2.6, where λ is the wavelength.

$$x = A \sin(2\pi f_0 t + \phi_0) \quad (2.5)$$

$$f_0 = \frac{S2d}{c}, \phi_0 = \frac{4\pi d}{\lambda} \quad (2.6)$$

When multiple reflections occur, they will have different size of $S\tau$, because they vary in distance and therefore in time τ . This feature allows to detect multiple reflections with a single chirp sent by the TX antenna. In this case, the Fourier transform must be performed to obtain individual signal components.

Velocity measurement

For velocity measurement, a frame of at least two chirps has to be transmitted from the transmit antenna with a time delay T_c between the start of the next one. After receiving, all chirps are processed through FFT to detect object range. Range FFT corresponding to each chirp will have a peak on the same location, but they will differ in phase $\Delta\phi$ as shown in Figure 2.4. Using this feature, velocity v could be calculated from Equation 2.7, where λ is the wavelength of the signal.

$$v = \frac{\lambda \Delta\phi}{4\pi T_c} \quad (2.7)$$

This equation shows that the velocity can be measured to some level or there will be ambiguity. The measurement is valid only if $|\Delta\phi| < \pi$. Maximum velocity v_{max} can be computed as Equation 2.8 [7]. Velocity resolution v_{res} can be computed by Equation 2.9, where L is the number of chirps in the frame [1].

$$v_{max} = \frac{\lambda}{4T_c} \quad (2.8)$$

$$v_{res} = \frac{\lambda}{4LT_c} \quad (2.9)$$

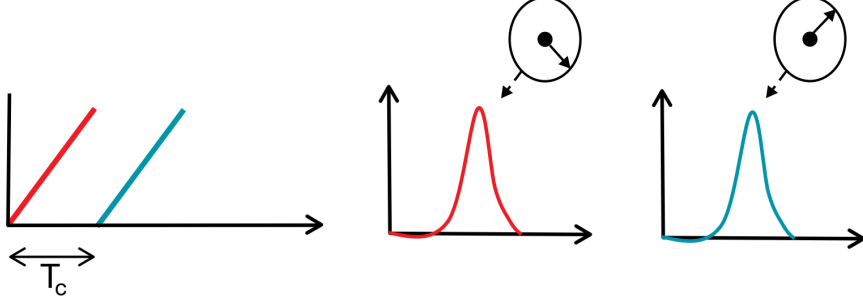


Figure 2.4: Phase shift made by doppler effect (taken from [6])

The minimum number of chirps in the frame is two. However, if multiple objects reflect at the same distance, more chirps are needed. The peak of every chirp incorporates the phase contribution of every object at the same distance. The second FFT, called Doppler FFT, is performed on every phasor of the frame to resolve the phase difference of each individual object.

Angle estimation

FMCW has the ability to estimate the angle between the horizontal plane and the reflected signal. This function is possible only when FMCW has multiple antennas. For example, the radar evaluation module for the IWR6843 chip used in this work has 3 antennas, which allows to detect angles in 2 axis [25]. The main idea is described in Figure 2.5. The signal traveled distance from the same reflection point to the receive antenna varies for each receive antenna.

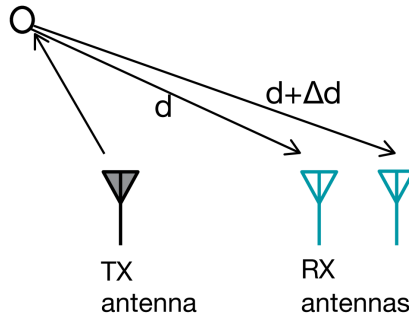


Figure 2.5: IF frequency reflection graph (taken from [6])

In the configuration in Figure 2.5, the phase change can be mathematically expressed as Equation 2.10, where Δd is the distance difference between the receive antennas and the reflected object and λ is the wavelength.

$$\Delta\phi = \frac{2\pi\Delta d}{\lambda} \quad (2.10)$$

With basic knowledge of the right triangle from Figure 2.5, it could be derived Equation 2.11, where l is the distance between the receive antennas and θ is the angle of arrival.

$$\Delta d = l \sin(\theta) \quad (2.11)$$

by combining both equations above is created Equation 2.12. The equation also shows that the angle of arrival has a nonlinear dependency with the sine function. This means that the accuracy of angle estimation is higher when the angle is closer to zero and lower when the angle is closer to the maximum angular field of view.

$$\theta = \sin^{-1}\left(\frac{\lambda}{2\pi l}\right) \quad (2.12)$$

2.3 Radar data representation

The previous section described how to determine the range and angle of an object that caused reflection. Results could be used in different applications. This leads to the issue of how to format the output. It is important to mention that the position represented by any format can be false because there is no guarantee that signal was reflected only once, which is ordinary, especially in confined spaces. Mainly there are point clouds and heat maps. An example of their visual form is presented in Figure 2.6.

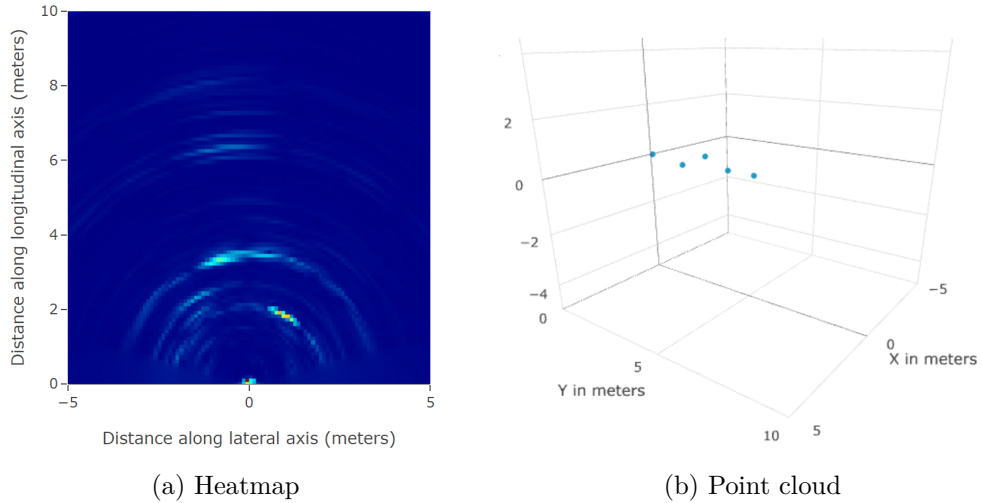


Figure 2.6: Different types of radar data representation

Point Cloud is a set of points in the three-dimensional coordinate system. The standard point cloud represents each point by x , y , and z axis coordinates. The radar point cloud also has information about the radial Doppler velocity of each point [7]. To produce a point cloud, the IF signal has to be processed by an algorithm such as Constant false alarm rate or threshold-based methods [30]. Unfortunately, this leads to data loss, because it has to present the most probable point position with fine accuracy and without any circumstances like reflection results from nearby area of the point.

Heat map is another representation of output. Ordinary is visualized as a range map, range-doppler map, or range-doppler-azimuth map. The range map is a two-dimensional array with time and range on x and y axis. The main difference between a heat map and a point cloud is that a heat map conserves more information than a point cloud because there is skipped part, where point cloud data processing loses information, and the whole outcome of FFT is presented. [1]

2.4 Group tracker

Group tracker or Gtrack is an algorithm developed by Texas Instruments. It is able to analyze the point cloud. It detects and tracks objects in the scene [24]. Big objects such as people, cyclists, cars, etc. in point clouds are almost every time represented by several points. The number of points representing one object can vary, but it can be estimated by its scale, reflectivity, or surrounding environment, which comes from the physical essence of the radar. One of the roles of the Gtrack is to create sets of points representing one object also known as clustering. Then these objects are tracked using the Kalman filter more accurately described in the next subsection. A verbally edited high level description from *Tracking radar targets with multiple reflection points* [24] is presented below.

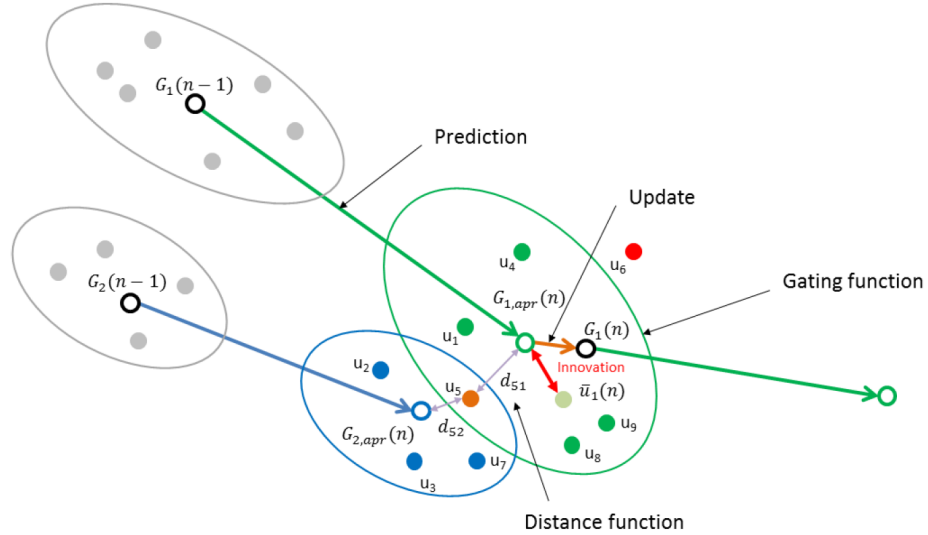


Figure 2.7: Group tracker algorithm (taken from [24])

- In the tracker prediction step, the track centroids are predicted for the current timestamp n based on the tracker motion model and the previous time stamp $n - 1$. All predicted track centroids are labelled. In Figure 2.7 centroids are $G_{1,apr}(n)$ and $G_{2,apr}(n)$
- A measurement boundary is constructed around each of the predicted centroids. The shape of this boundary is defined by the gating function which represents the amount of innovations that would be accepted
- A set of the points captured in current time $[u_1, u_2, u_3, \dots, u_9]$ received from the radar has to be associated with tracks. Distance between every point and every predicted

track centroid is computed. Then points are assigned to the closest centroid, which also has them inside its boundary box. In Figure 2.7 point u_5 is inside the boundary boxes of both $G_{1,apr}$ and $G_{2,apr}$ predicted track centroids. However, distance d_{51} is smaller than d_{52} which means that the point is assigned to $G_{1,apr}(n)$

- For each track, the mean is calculated from a set of points assigned to a specific track. In Figure 2.7 is for track 1 with set of associated points $[u_1, u_4, u_5, u_8, u_9]$ and computed mean $u_1(n)$
- Between the predicted centroid $G_{1,apr}(n)$ and mean of the measurements is difference called innovation. The innovation is used as a measure to determine how much is prediction different from the observed measurement
- Using the predicted centroid $G_{1,apr}(n)$ and innovation, each track centroid is updated. In Figure 2.7 it is updated to position $G_1(n)$
- The whole sequence of operation is repeated for the next time stamp

The object detected by Gtrack has several parameters. The most important is probably the position represented in the three-dimensional coordinate system. The position is computed as the center of a set of points and it should represent an object from the real world [26]. Thanks to the nature of radar, detected object position is definitely not an accurate estimation of a center of a real world object. The point cloud is not dense enough and it has problems with detecting the edges of objects.

Kalman filter

The Kalman filter is one of the most essential filters in navigation. The problem could be described as a difference between measured and estimated models of investigated quantities. The goal is to find a value that would most accurately correspond to ground truth. The main idea of the filter is that every model has some kind of uncertainty, which could be made by measurement accuracy (noise) or neglecting influences. [27]

It could also be explained by an example. Let's imagine that there is a car. The car is moving at a certain speed and has some position on the road. The goal is to compute its position in the next time step. The measurement model uses, for example, GPS for position and direction and the wheel speed sensor for velocity. All described sensors will measure with an error. The estimation model uses only values measured by the measurement model and does not take into account acceleration, change of direction, and many more aspects, which could change the outcome. Both models have some errors that can be described. The goal is to find the most probable position based on both models using the rotation of prediction and measurement steps.

Chapter 3

Current state of video processing

This chapter is dedicated to processing video images and the basic principles of a digital camera. The main task is to present a short overview of several current possibilities for image processing that are connected to fusion.

A digital camera is a device that captures light and transfers it into digital form. The invention of the digital camera was conditioned by the invention of an image sensor. An image sensor is an electrical component capable of capturing light and converting it to two-dimensional digital information called an image. An image is a two-dimensional array of pixels. Pixel is the smallest addressable element in an image. Each pixel has a value. According to color format, pixel value represents its color. [4]

It is important to mention that most image sensors use a rolling shutter. This means that each row or column of the image is captured at a slightly different time, which can have, in some cases, an effect on the determination of an accurate timestamp. On the other hand, cameras with a rolling shutter combined with flash can be used for the synchronization of multiple cameras. [22]

Before light hits the image sensor, it is usually focused by the lens. This also refers to the field of view (FOV). FOV is the angle observable by the camera at one moment. One of the parameters used to compute the FOV is the focal length. Focal length is the distance from the view plane to the center of the projection. A small field of view is unable to monitor nearby objects. A large field of view lens brings deformation to the image. The field of view also affects the pixel resolution of objects at certain distances in the image plane. [4]

Nowadays, it is popular to use a dual camera system. The dual camera system unlocks several abilities. If they are well synchronized, they can not only increase FOV but also compute image depth [9], which is the imaginary third dimension of the video image.

3.1 Video object detection and tracking

Object detection can be divided into object localization and classification. The goal of object localization is to detect the presence of objects in an image and create a bounding box around them. The input for object localization is an image and the output is a set of bounding boxes. The goal of object classification is to determine the object class with some degree of certainty. Input is an image with a single object or an image with additional information about the positions of objects because classification itself can not localize ob-

jects. This is also the reason why classification makes sense only after the localization of an object. The output of object classification is a class label. [5]

Considering video object detection models, there are mainly R-CNN, Fast R-CNN, Faster R-CNN, and YOLO. However, for the purpose of this work, only YOLO seems relevant as the fastest detection model. Traffic monitoring brings many data. The camera at a low frame rate brings ten frames per second. Although real-time processing may not be a necessary condition, it is important to process this load of data in a reasonable time. [5]

Model You Look Only Once (YOLO)

The model was first described in 2015 by Redmon et al. [18]. The name YOLO is an abbreviation of You Look Only Once. At first, YOLO creates a grid on the input image. The most probable class is then determined for each cell in the grid. At the same time, the bounding boxes are approximated. The center of each bounding box is then computed. According to the grid cell to which the center belongs, each bounding box receives is assigned a class. The process is visualized in Figure 3.1. [18]

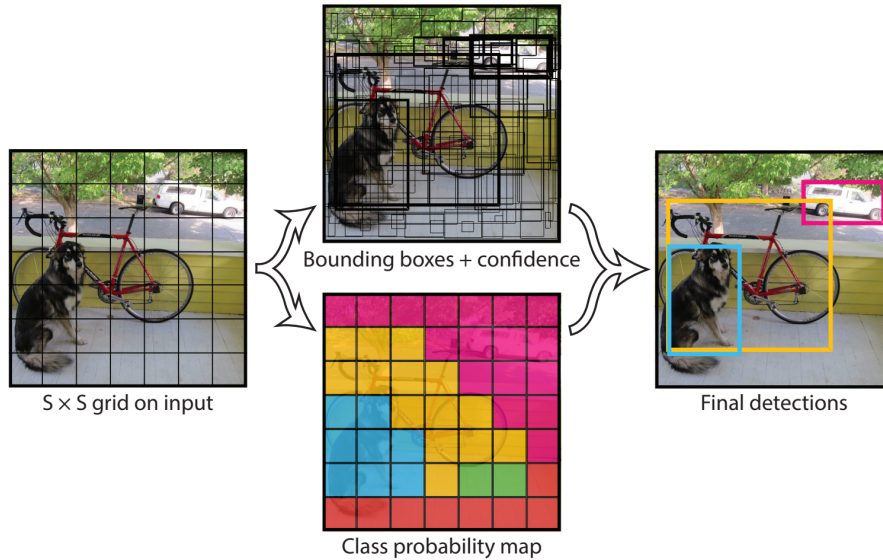


Figure 3.1: YOLO model (taken from [18])

Since 2015, the model has been upgraded many times. At the time of writing this work the latest official version created by the original authors is YOLOv8. [10]

Video object tracking

Object tracking is a process in which the algorithm tracks the position of an object within the scene. The main task of the object tracker is to predict the position of the object and attach the detected objects to the track. [29]

This work uses ByteTrack as a tracking algorithm. The ByteTrack was introduced by Zhang et al. [31] in 2022 and they presented impressive results in comparison with the vast majority of other tracking algorithms. Its input is objects detected by the object detector and detection threshold score. Its output is found tracks. The ByteTrack divides objects into two groups. One group is filled with objects with a lower score than the input threshold and the other is filled with objects with a higher score than the threshold.

After that prediction of position for the current frame is computed using the Kalman filter. The first association is performed with track predictions and high score detections using the Hungarian algorithm. Objects without associated track create new tracks. The second association is performed on objects with a lower score than the threshold. Objects from this group and without association are deleted. Tracks without associated objects are placed in the set of lost tracks. If tracks remain in the set longer than the number of video frames specified before the initialization of the whole algorithm, they are considered as ended. [31]

3.2 Intrinsic and extrinsic transformation matrix

One of the main differences between radar and video data is the dimension. Radar data are set into three dimensional Cartesian coordinate system. On the other hand, video data are only in two dimensional Cartesian coordinate system. The fusion algorithm described in this work requires the possibility of transforming points from one coordinate system to another. In literature [4, 23], a transformation of a point from the three-dimensional coordinate system of the world or any related object to the image plane of the camera is performed using the intrinsic and extrinsic matrix. Multiplying these two matrices creates the transformation matrix. [4]

Extrinsic Matrix

The goal of the extrinsic matrix is to transform the point into the 3D coordinate system of the camera. This includes operations such as rotation and translation. A point in the space could be translated to a new position by shifting with a displacement vector t . Translation can be described as Equations 3.1, where variables with null index $p_0 = (x_0, y_0, z_0)$ are initial coordinates, $t = (t_x, t_y, t_z)$ is the translation vector and variables with an apostrophe (x', y', z') are the final result. The form of the translation matrix is visible in Equation 3.2. [4]

$$x' = t_x + x_0, y' = t_y + y_0, z' = t_z + z_0 \quad (3.1)$$

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

Elemental rotation can be performed on one axis of the coordinate system. These rotations are also called pitch, roll, and yaw. Depending on naming axis, they are represented by Equations 3.3, 3.4 and 3.5, where θ is the angle of rotation. [23]

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

Multiplying translation and rotation matrices creates an extrinsic matrix H_e as shown in Equation 3.6. [23]

$$H_e = R_x R_y R_z T \quad (3.6)$$

Intrinsic matrix

Intrinsic matrix transforms the point from the three-dimensional coordinate system of the camera to the two-dimensional coordinate system also known as the image plane. It includes operations such as shearing and scaling. Scaling increases the distance of the point from the origin of the coordinate system, if the point is not exactly in the origin. Scaling is usually represented as shown in Equations 3.7 and 3.9, where S_x and S_y are coefficients representing the amount of enlargement in the dedicated axis. [4]

$$S_c = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.7)$$

If the scaling is not uniform, it is called shearing. For this work, the most important shearing is parallel to the x axis. It can be described as Equation 3.8 or 3.9, where k is the coefficient defining the amount of shear. [4]

$$x' = x + ky, y' = y \quad (3.8)$$

$$S_h = \begin{bmatrix} 1 & k & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.9)$$

Combining these two operations creates an intrinsic matrix. In some cases, a more sensible approach can be selected as shown in Equation 3.10, where f is the focal length, s_x and s_y are dimensions of one pixel in the image plane, o_x and o_y refers to center coordinates or half of the pixel resolution for each axis. [4, 16]

$$H_i = \begin{bmatrix} f/s_x & 0 & o_x \\ 0 & d/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \quad (3.10)$$

3.3 Object depth prediction

In the beginning of this chapter, it is described that the camera output is two-dimensional. However, the third dimension could be estimated. One of the commonly used ways to estimate depth is using multiple images of the same observed area from different camera positions. If objects in space are moving, images should be captured at one moment or shortly after the previous one. In that case, the small time difference between individual captures results in better depth estimations. Capturing multiple images at once could be

performed using a dual camera system. In [9] they have extracted features from each image and used image matching based on ORB algorithm. They have declared 2.5 % accuracy. However, they used a laser to increase the density of extracted features and focused on very close-range objects.

Another way is to capture image using the monocular camera. Abdulwahab et al. [2] presented in their work depth prediction from a single monocular image using the feature extraction and their semantic context. Their method seems to preserve object boundaries and with significant extension of their work, it can be potentially used in traffic monitoring scenes.

Another approach more relevant to this work is to estimate the depth of the detected object with some additional information from the observed area. In the case of this work, it is the coordinates of objects detected by radar. One of the common approaches is to use the DNN.

Sengupta et al. [21] presented work, where they were testing the video depth approximation of human at close range. They have cut the maximum radar depth to 5 meters and created a grid of $0.1\text{ m} \times 0.1\text{ m}$ cells. Then they collected an image of a person in each cell with its ground truth of the actual position as a training set. Their model consists of three hidden DNN layers with 256, 128 and 64 nodes. They used Adam optimizer with a batch size of 20. They trained the model for 25 epochs. With this model, they have been able to get to centimeter accuracy under certain conditions.

Ming Liu et al. [14] presented work, where they were developing an online intelligent method to calibrate radar and camera. Their case was more related to this work than the previously described method. Their depth prediction model was mapping objects detected on the video plane to the plane defined as a road surface. Their model consists of 5 hidden DNN layers with 5, 10, 20, 10, and 5 nodes. They have collected a set of 10000 radar and video pairs for training. They used ReLU as the activation function, MSE as the loss function, and stochastic gradient descent as the optimizer. They have trained the model with batch size 128 for roughly 100 epochs.

Chapter 4

Current state of radar and video fusion algorithms

This chapter is dedicated to describing several methods and algorithms used in radar and video fusion. The main identified issues are time synchronization, spacial calibration, and association. One of the most telling pieces of information about the model is its level of fusion. According to Abdu et al. [1], fusion can be divided into three levels.

The first level is called data level fusion. The main idea of data level fusion is to combine the radar and camera output as soon as possible. Radar and video data are fused in a raw structure, which contains more information than the processed one. The second level is called feature level fusion. Feature level fusion could be described as mid-level between data and decision level fusion. All sensor fusion branches are processed at the beginning separately. Models produce as outputs from each branch feature map [13]. After features in a certain level are obtained, a fusion of branches takes place to get the final result. Last level is called decision level fusion. In the decision level fusion model is fusion performed on the highest level of abstraction. Object detection, classification, and tracking are performed on radar and video data separately. Results from both branches are then compared and possibly processed to the final result of fusion. The difference between fusion is also visualized in Figure 4.1. [1]

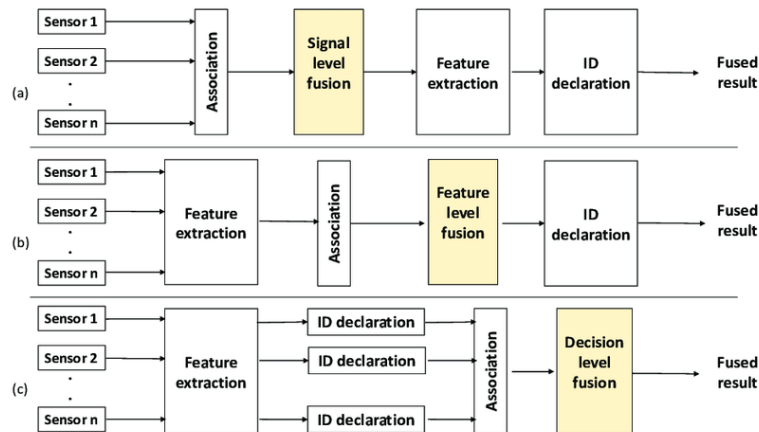


Figure 4.1: Different level of fusion: (a) signal (or data) level fusion, (b) feature level fusion, and (c) decision level fusion (taken from [11])

4.1 Radar and video spacial calibration

The basic idea is that there is a need to transform three-dimensional data into a two-dimensional image plane to correctly visualize detected radar points or objects in the camera image plane. There are several methods to perform this task. Probably the most straightforward method is to compute the transformation matrix using intrinsic and extrinsic parameters as shown in Section 3.2. This approach was chosen for example by Pan et al. [16].

Another approach is to collect point pairs from both coordinate systems that correspond to each other. According to *A Comparative Study on Camera-Radar Calibration Methods* [15], Direct linear transformation with some modifications is one of the best ones stated if the initial set of point pairs is large enough.

The problem that direct linear transformation (DLT) solves is described as Equation 4.1, where $\tilde{p}$ are the homogeneous coordinates of $p = [x, y]$ related to the radar plane, $\tilde{q}$ are the homogeneous coordinates of $q = [u, v]$ related to the video image plane and H is the desired transformation matrix described in Equation 4.2. Equation 4.2 also emerges the least number of known input of points. Direct linear transformation needs at least four known point pairs to compute the transformation matrix. However, to achieve accurate results, more pairs are needed. Points also should not be on the same plane. [15]

$$\tilde{q} = H\tilde{p} \quad (4.1)$$

$$H = \begin{bmatrix} h_{11} & h_{21} & h_{31} \\ h_{12} & h_{22} & h_{32} \\ h_{13} & h_{23} & h_{33} \end{bmatrix} \quad (4.2)$$

The coordinates of the image plane can be rewritten as Equations 4.3 and 4.4. [15]

$$u = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \quad (4.3)$$

$$v = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \quad (4.4)$$

Considering this, matrix coefficients can be computed as shown in Equation 4.5. This equation represents one pair. Matrices from different point pairs can be placed one under the other to represent multiple point pairs matrix and subsidized as shown in Equation 4.6, where A represents coefficient computation matrix and b represents the vector of relevant u and v coordinates. After creating A and b matrices, transformation matrix can be computed as Equation 4.7. [28]

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -u_i x_i & -u_i y_i & -u_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -v_i x_i & -v_i y_i & -v_i \end{bmatrix} H = \begin{bmatrix} u_i \\ v_i \end{bmatrix} \quad (4.5)$$

$$AH = b \quad (4.6)$$

$$H = (A^T A)^{-1} A^T b \quad (4.7)$$

To achieve the best result, Oh et al. [15] have used Levenberg-Marquard optimizer and normalized data. They stated that after normalization, the set of radar points has a zero mean and an average distance from point to mean is $\sqrt{2}$. DLT can be also extended to accept as input three dimensions.

4.2 Radar and video time synchronization

It is important to synchronize data from both sensors. The fusion of data captured from the scene is relevant only if they are taken at the same time or very soon after. There are several methods on how to achieve synchronization.

To put the importance of accurate synchronization in perspective. Let's have a car travelling at a velocity of $v = 72 \text{ km/h} = 20 \text{ m/s}$. Taking into account equation $d = vt$, where d is distance and t is time, the car will travel every millisecond 2 centimeters, every 10 milliseconds 20 centimeters, and every 100 milliseconds 2 meters.

Hardware synchronization

Hardware synchronization is also known as hard synchronization. The principle of hardware synchronization is to send a physical signal to the sensor through hardware triggers to initiate frame capturing. Basically, there is one main fusion device sending signals to the hardware trigger of each sensor to collect data at a particular time. The main fusion device could then be used to collect data from all sensors and synchronize them with knowledge about the exact relative capture time. [8]

Software synchronization

Software synchronization is also known as soft synchronization. The principle of software synchronization is to provide a time source for each sensor and attach a timestamp to captured data. One approach is to use timestamps. Both radar and video data frames have to have an accurate timestamp [8]. This results in several needs:

- Every sensor dedicated to data collection has to be able to timestamp every data frame with fine accuracy.
- All clocks must be synchronized with fine accuracy.

The true meaning of fine accuracy is depending on the fusion use case. In the case of people tracking by et al. [21], it was 25 milliseconds. In the case of traffic monitoring, fine accuracy could be considered as within ten milliseconds (see example at the beginning of this section). Clocks can be synchronized with several methods. One of them is GPS. Atomic clocks in GPS satellites can keep the time accuracy within nanoseconds. Despite the long transmission distance, the device receiving the GPS signal could still achieve microsecond accuracy due to synchronized device motion, atmospheric and relativistic effects [3].

Despite accurate timestamps, there are still delays between different sensor data frames, if each sensor started collecting data at a different time and their frequency does not change as shown in Figure 4.2, where on the upper time axis are video frame capture events and on bottom time axis are radar frame capture events. This issue could also be significant in hardware-synchronized fusion systems, where there is a significant delay between receiving radar and video data frame. If radar data are already tracked objects with position and velocity estimates, their position can be extrapolated to the time when video frames were taken based on motion model. [32]

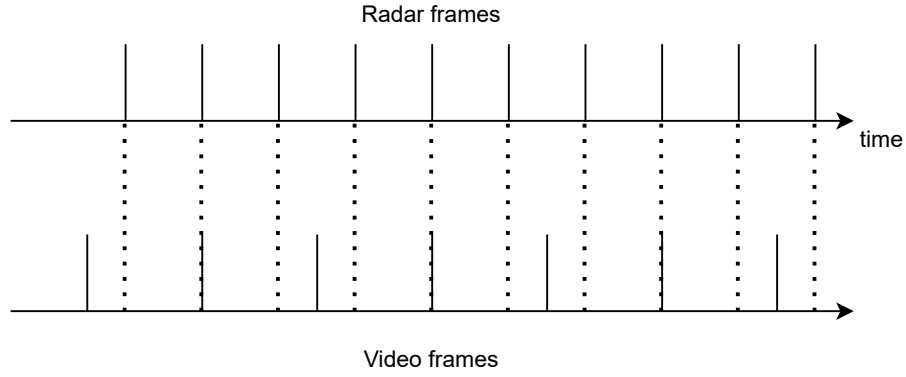


Figure 4.2: Difference between receive time of radar and video frames

4.3 Object association

Object association is a system used in many decision-level and feature-level fusions to create pairs of objects detected by radar and video if more than one detection occurs in one sensor branch. Associating could be performed based on the detected object position difference from sensors [20]. Position differences could be considered in the two-dimensional camera image plane or in the three-dimensional radar coordinate system, depending on the transformation capabilities of the fusion system. All the mentioned methods result in nonnegative $n \times m$ matrix, where n is the number of objects detected by radar and m is the number of objects detected by video. The matrix values represent the score or probability between every possible pair. This problem could be solved using the Hungarian algorithm [20].

Hungarian algorithm

Hungarian algorithm also known as the Kuhn–Munkres algorithm or Munkres assignment algorithm is a combinatorial optimization algorithm. It was developed in 1955 by the American mathematician Harold Kuhn. However, his work was based on Dénes Kőnig and Jenő Egerváry, and therefore Kuhn named the method Hungarian. Hungarian algorithm problem could be described as nonnegative $m \times m$ matrix, where each row and column must have exactly one value selected so that the sum of these values will give the smallest possible value. In other words, there are m workers and m jobs. Every worker can do only one job. The cost of a same job could vary for each worker. The goal is to cover all the jobs at a minimal cost. The original algorithm was developed only for square matrices. [12]

Unfortunately, in this work solving rectangular matrices is needed. For this original algorithm was modified. A verbally edited step-by-step description of the modified Hungarian algorithm from Pilgrim [17] is presented below.

- Create the cost matrix with n and m dimensions in which each element represents the cost of assigning one the n th worker to the m th job. The number of rows has to be smaller or equal to the number of columns and let $k = \min(m, n)$. Rotate the matrix if needed.
- Find the element with minimal value for each row and subtract it from each element in that row.

- Find zero value in the matrix. If there is no starred zero in the same column or the same row, mark it with the star. Try to mark all zeros in the matrix.
- Count the number of columns, which contain starred zero. These rows are covered. If the number of covered columns is equal to k , the positions of starred zeros represent the solution. If it is not so, continue to the next step.
- Find a noncovered zero and prime it. If the row containing primed zero has no starred zero, continue to the next step. Otherwise, cover the row containing the primed zero and uncover the column of mentioned starred zero. Continue with this process until there are no uncovered zeros. Find and save the smallest uncovered value in the matrix. Skip to the next step.
- Create a list of primed and starred zeros as follows. Mark the uncovered primed zero found in the previous step as Z_0 . If there is a starred zero in a column of Z_0 , mark it as Z_1 . In a row of Z_1 is always the primed zero. Mark it as Z_2 . Loop this sequence until a primed zero has no starred zero in its column. Add all found Z s to the list. Unstar each starred zero from the list. Change each primed zero from the list to the starred zero. Uncover every line in the matrix and return two steps back.
- Add the value found in the previous step to each element of each covered row and subtract it from each element of each uncovered column. Go 2 steps back.

Chapter 5

Fusion architecture design

This chapter is dedicated to the fusion architecture designed in this thesis. Input for this work is radar and video frames with some additional information about the scene, depending on the selected spacial calibration and association method. Output is a set of fused tracks with their timestamped positions. The designed method belongs to decision level fusion. The data flow is described in Figure 5.1. In the beginning, there are two separate branches, which is typical for this level of fusion.

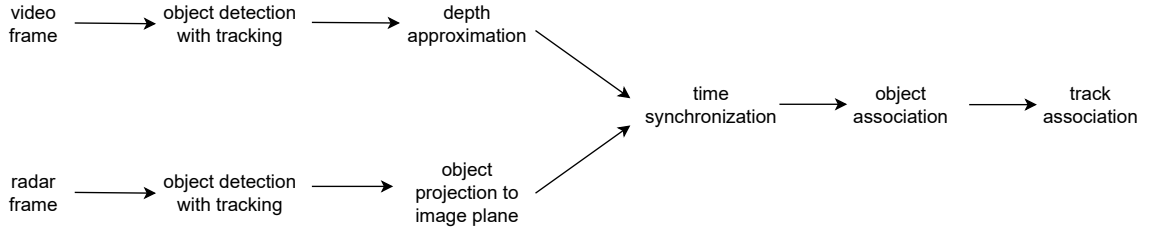


Figure 5.1: Fusion model diagram

5.1 Video and radar branch

The model could be categorized as decision-level fusion. Fusion on this level has several advantages. One of the biggest advantages is the possibility of reusing current models and algorithms to process data from sensors. Pre-trained video models are much more accurate than pre-trained models for radar so far. With decision level fusion there is the possibility of using only video detection and classification models alone, which would be complicated for any other levels of fusion because they are unable to use whole model. Considering algorithms, Texas Instruments implemented in some of their mmWave chips a tracking algorithm called Gtrack, which is able to detect objects, measure their speed, acceleration and track them. Due to this feature, there could be a lowered computational load on fusion unit, because part of the work can be accelerated on the radar chip. Another advantage is robustness of whole system. If system loses confidence in the results of one sensor, it can rely just on data from the other sensor, because both sensors could try to get same features as result.

Input for the model is video data obtained from the camera and pre-processed radar data from the radar. Video data are then processed using the YOLO [10] detection model in

combination with Bytetrack¹ tracking model repository implemented by Zhang et al. [31]. Objects detected on video are then processed by the depth prediction neural network to obtain coordinates from a three-dimensional radar coordinate system. Radar data are processed in a more suitable format. Centers of tracked objects from the radar are transformed into the video image plane.

5.2 Fusion module

The fusion module is the core of the whole work. From this point, radar and video data are processed together. One of its tasks is to perform time synchronization. Data from radar and video have to be time synchronized when fusion is performed because, without synchronization, there are very limited ways how to benefit from fusion. In the case of this work, it means that data frames are connected together when their timestamps are as close to each other as possible.

Spatial calibration

Spatial calibration can be performed by collecting several accurate points with physical object clearly visible by both sensors. As an example can be mentioned corner reflector used by Reich [19]. The problem is mathematically described in the previous chapter and could be relatively easily computed by fusion device. Unfortunately, using this method can be costly, because it includes manual point collection. Ming et al. [14] proposed an association method relying on a line where object detected by video enters the field of view of radar, but this method needs very fine radar detection accuracy that is not easily achievable in a real environment.

If the radar and video sensor are attached to a solid construction of a physical fusion device so that there is a fixed distance and angle between the individual sensors, calibration can be performed on only one device of that type. Unfortunately, achieving this kind of accuracy in the installation is challenging, especially if the sensors are far from each other.

One of the goals of this work is to discover ways how to simplify the whole spatial calibration process. The part that consumes the most resources is data collection, because commonly used methods require collecting data necessary for computing manually. There are some methods describing automatic pair collection, but they require accurate and reliable data in combination with very small amount of detected objects at one time [14]. This work presents several ways to perform pairs collection to obtain a suitable data set for spatial calibration even if the data are not entirely reliable or if the observed area has a dense traffic of observed objects.

Depth prediction

Depth prediction for video frames can be performed by various methods. This work relies on two types of them, which can use the benefits of radar data. The main idea is to train a model that can predict the depth of an object detected in video using the dataset that obtains radar and video object pairs, where radar objects contain information about depth.

The first approach is to use to compute it by a set of equations given by the transformation matrix and plane approximation. The transformation matrix itself does not support the reversal operation. However, vehicles in space are not moving with absolute freedom.

¹<https://github.com/ifzhang/ByteTrack>

They are bounded to the ground that can be approximated by the equation. With the approximation of road surface and transformation matrix, there can be performed depth prediction of detected objects.

The second approach is to use DNN from other works [14, 21] and try to adjust them for specific demands of this work based on the training data set and the output coordinate system, which is one dimension higher than originally. The goal is to implement designated methods and develop a collecting algorithm for training and testing dataset. Testing includes comparing these methods by their accuracy.

Association of detected objects

The association of detected objects is ordinary performed by the Hungarian algorithm. Unfortunately, the basic version of the Hungarian algorithm is not suitable for this work. It relies on the same amount of elements of each type. In other words, square score matrix. It also does not take into account the fact that some objects are detected only by one sensor.

For this reason, basic algorithm is modified based on own modifications or modifications presented in other works. It accepts rectangular matrices as input and also contains a threshold function, which determines the maximal score value accepted for the association. Hungarian algorithm also requires the creation of a scoring function, which should describe the probability of association for different elements.

The association can be performed in two steps. The first association can be performed on objects detected at once. The second association can be performed upon tracks with knowledge of the number of associations between different detected objects from the previous steps to achieve a better association of whole tracks.

Chapter 6

Implementation

This chapter is dedicated to implementation of the designed solution. Input for this work is radar and video frames with some additional information about the scene, depending on the selected spacial calibration and association method. Output is a set of fused tracks with their timestamped positions. The key part of this work are spacial calibration methods, depth prediction methods and association method that can be divided into partial association to associate objects detected at one moment and track association to perform object tracking in fusion level. Spacial calibration and depth prediction are necessary for fusion model to perform object tracking. Some depth prediction models also need more significant dataset than spacial calibration ones or even need the transformation matrix. The recommended procedure is at first calculate transformation matrix using one of spacial calibration methods. Then using the transformation matrix to create model for depth prediction and after that activate the tracks fusing method.

6.1 Radar and video detection and tracking

Fusion is at its start divided into radar and video branches. Both branches receive different data structures describing the same scene from the same moment and therefore they use different methods to detect and track objects ready for fusion. After each branch end its processing they are synchronized according to their timestamps. The fusion module relies on the timestamps provided with the frames. To achieve even more time synchronization accuracy, positions of radar objects are approximated to the exact time when the video frame was taken. This is possible because the velocity of each radar object is known.

Radar branch

This work uses radar data from a real system based on IWR6843 radar chip introduced by Texas instruments. Input data structure works on TLV principle as a standardized encoding scheme [26]. TLV stands for type, length, and value, which also partially defines the structure. Every frame starts with the header. Frame header contains for fusion several essential information like timestamp, length of packet, number of TLV following frame header, and more. The timestamp represents time in milliseconds. The input data from the radar are already processed by the Gtrack algorithm. This approach is also the most probable in real deployment because IWR6843 can compute Gtrack with its hardware acceleration. The radar coordinate system is presented in Figure 6.1.

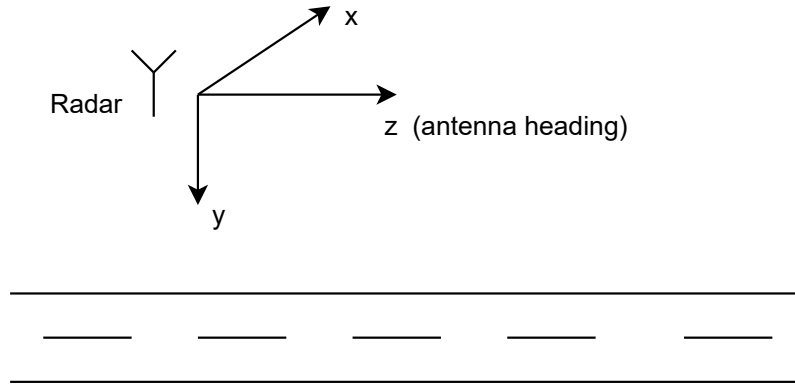


Figure 6.1: Radar coordinate system

Video branch

Input for the video processing branch are video frames taken from the camera. The origin of the coordinate system in the image plane is situated in top left corner increasing value in down and right direction as shown on Figure 6.2. Output of video branch is structure containing detected objects image plane coordinates, class, confidence and their track id, if tracking algorithm is successful. For video recognition and tracking is used pre-trained model YOLOv8s [10] with the ByteTrack¹ repository implemented by Zhang et al. [31]. YOLOv8 has several different model subversions according to their size. The model size interferes with the model performance. Smaller models are faster than bigger, but less accurate. For the need of this work, the version called YOLOv8s was chosen. Since the ByteTrack combined with YOLOv8 can be considered partially compatible, ByteTrack does not associate with detected objects, but just creates a bounding box prediction about the current position for each track detected, an association has to be made. Because of this, for every possible pair is computed the intersection over Union (IoU) of their bounding boxes. The detected object with the largest non-zero match is then associated with the chosen track id.



Figure 6.2: Video coordinate system

¹<https://github.com/ifzhang/ByteTrack>

6.2 Spacial calibration

Extracting calibration pairs with division to areas method

Points are represented as centers of detected objects in both coordinate systems. The main problem is to find objects from both systems, which represent the same object in the real world. This could be performed only under certain circumstances defined by the rules below.

Firstly, both systems have to detect after filtering any probable false positive detection only one object at a time within the scene. To increase the number of extracted pairs, some scenes can be divided into areas. The necessity of only one detected object is then transferred to created areas. One of the approaches is to divide scene by a vertical line in the image plane. The division has to be performed in a way that detected objects in the left area are moving in a different direction than objects in the right area. The collecting algorithm is then able to determine which radar detected object belongs to the observed subarea. Another approach can be division into areas by a horizontal line. If there is a dependence between the distance of the sensor from the object and the vertical position of the video object center. This approach requires at least a rough estimate of the distance in one place. The distance can be described by an interval to increase confidence. However, the greater this interval is, the greater the undefined hole with no collected pairs will be. For this reason, there will be only experiments with only one line of this type. To simplify the process of creating the areas, a simple line calculator was developed as is shown in Figure 6.3.

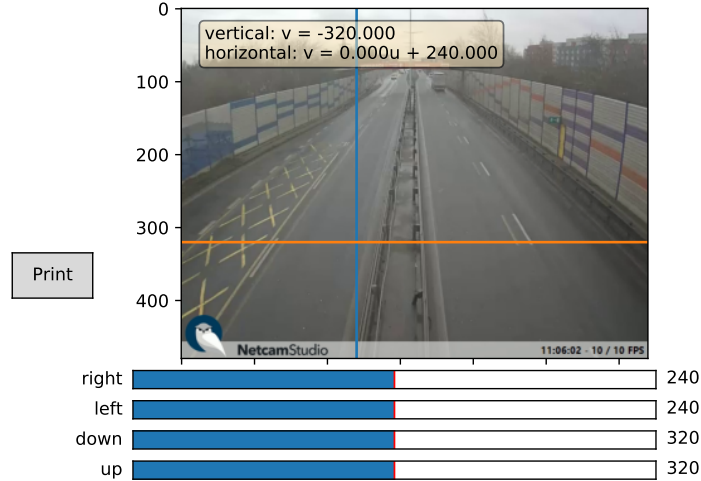


Figure 6.3: Screenshot of dividing lines calculator

Secondly, there has to be a trust that there are no false negative or positive errors or that both sensors have these errors for the same objects. This parameter sounds tricky, but trust in pairs could be gained if a few elementary conditions are met. Considering the camera, it has great performance in good visibility. Sunny days are much better than foggy nights. If all detection for potential point pair collection is performed under good visibility conditions, data could be potentially trusted. Considering radar, it uses a radar detection

algorithm that is ineffective in very dense traffic where objects are close to each other. This issue was thrown out by the first condition. Another case is when objects in the real world are stationary to the radar. The radar tracking algorithm described is much more accurate for moving objects, because of the information about the Doppler effect for each point. The last issue is the field of view. Points can be extracted only from areas, which are completely inside FOV of both sensors.

It is important to mention that some errors in pairs could be tolerated if they can be recognized from valid ones by their big error. Transformation error is computed as Euclidean distance between reference and computed image plane coordinates.

Extracting calibration pairs with approximated intrinsic and extrinsic transformation matrix

Another approach to the problem is using as help an approximated intrinsic and extrinsic transformation matrix (AIEM), which can be corrected. In this method, objects detected by radar are filtered. Filter discards any objects, which are not alone in the area defined by the filter to increase the probability of correct association.

For this method, the inaccurate matrix can be estimated using the intrinsic and extrinsic parameters of the camera. If they are known, the transformation matrix can be easily computed as shown in Section 3.2. Parameters such as coordinate shift between the radar and camera, pan, or roll do not have to have a significant effect. On the other hand, pitch, focal length, and sensor chip size are significant. To increase performance, the pitch between radar and video sensor differs by purpose and it is not only by error made by physical manufacturing or installation. Focal length and tilt are both creating matrix coefficients, which are used to scale projected points in the image plane. This varies for the type of camera and it can be time-consuming to estimate. For this reason, a calibration visualizer was implemented to show real-time results of change values as shown in Figure 6.4.

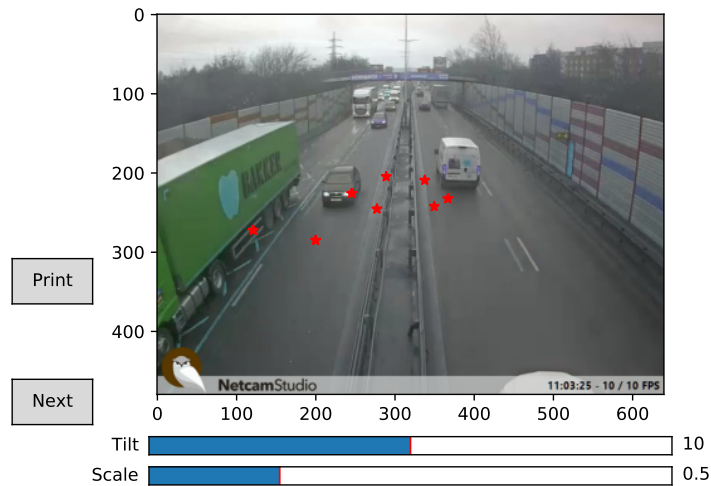


Figure 6.4: Screenshot of approximated intrinsic and extrinsic matrix (AIEM) calculator

Objects are then transformed into an image plane of a video sensor. Objects there are then scored among each other to get a score matrix and associated using modified Hungarian algorithm that is more described in Section 4.3.

Computing transformation matrix using collected pairs

For the computation of the transformation matrix, the discrete linear transformation method known as DLT is selected as the best method presented in [15] and described in Section 4.1. To increase the accuracy of the transformation matrix, the least squares optimizer is used. The whole method is also extended to transform points from the three-dimensional coordinate system that radar has to the two-dimensional system of the image plane.

The DLT requires that pairs in the collected set have to be scattered in the observed area uniformly. For this reason, the density filter is implemented. It guarantees that the closest distance between two pairs in a three-dimensional observed area is bigger or equal to the selected input value. This erases all hot spots, which can lead to overlearning the matrix.

Another aspect that can strongly decrease the quality of the final computed matrix is the number of error pairs in the set. Some of these errors are visible in the result when the same set which is used to compute the matrix is used to test it. If the number of pairs is sufficient and there are pairs with errors significantly bigger than the mean, there is an increased probability that these pairs are associated incorrectly.

6.3 Predicting depth from video

Computing depth in case of this work is basically transforming point from a two-dimensional coordinate system of video to a three-dimensional coordinate system of radar.

For computing depth, two different approaches are relevant for this work. Computing depth using the transformation matrix and approximated plane is not a commonly used solution. On the other hand, predicting depth using DNN is used in several works mentioned in Section 3.3.

Computing using transformation matrix and plane

The first approach is to seek a mathematical solution to this problem. For transforming the point from the radar to the video image plane, there is a transformation matrix. Unfortunately, this system of equations does not support the desired reversal operation. For such a transformation, an additional equation has to be found.

The use case for this work is primary traffic monitoring. Vehicles are moving on a road, which means that they are moving on a surface. The ruggedness of the terrain surface can be simplified to a plane. Plane general equation is presented in 6.1, where x , y and z are the radar coordinates and a , b and c are wanted coefficients. The plane could be approximated through the center positions of objects detected by radar. If there is a sufficient set of these objects, this approximation could be performed quite accurately and could be used as the fourth equation necessary for computing three-dimensional coordinates.

$$ax + by + c = z \tag{6.1}$$

Equation 6.1 has to be adjusted to homogeneous coordinates as is in Equation 6.2, where i is a homogeneous coordinate of the radar coordinate system. With the addition of

Equations 6.3, 6.4 and 6.5, where subscript for each h is the index of row and column of known transformation matrix and u and v represents coordinates on the video image plane, there is a system of four equations with four unknowns, which is calculable.

$$\frac{ax + by - z}{i} = c \quad (6.2)$$

$$h_{00}x + h_{01}y + h_{02}z + h_{03}i = u \quad (6.3)$$

$$h_{10}x + h_{11}y + h_{12}z + h_{13}i = v \quad (6.4)$$

$$h_{20}x + h_{21}y + h_{22}z + h_{23}i = 1 \quad (6.5)$$

One of the problems this approach brings is the plane approximation. Plane can be approximated with several detected radar targets, where their center coordinates are used as a point, which belongs to the plane. Theoretically, three points are the minimum number. Practically, more points are necessary to create a good approximation. The main reason is that objects detected by the radar using the Gtrack algorithm are not on a same plane. This issue results in error, because center of radar detected object is not exactly the center of ground truth object. One of the solutions can be an approximation of two planes based on the class of detected objects because there is a possibility that objects higher in reality also have higher center coordinates when detected by Gtrack than lower objects. Results of this experiment are presented in Section 7.3.

Predicting using DNN

The second approach is to estimate depth using deep neural network. These methods require the collection of radar and video detected objects associated in pairs to learn the neural network. Because of this, it is advantageous to find out the form of the transformation matrix of the fusion system and use it as a booster to find more associated pairs in shorter frame sequence. If the fusion does not have to be performed in real-time, there is a possibility to calibrate sensors first and after that associate pairs on the whole learning dataset. For the purpose of this work, two DNN models were implemented. The output for both models is the same. It is (x, y, z) , where x , y and z represent the predicted coordinates of the video object in the coordinates of the radar sensor.

The first model is based on Sengupta et al. mentioned in Section 3.3 with some changes. The input of the DNN remains. It is the square $(u_{min}, v_{min}, u_{max}, v_{max})$, where u_{min} and v_{min} are coordinates of the top left corner of the detected video object and u_{max} and v_{max} are bottom right coordinates of the detected video object bounding box. Output is changed to three-dimensional coordinates and the number of epochs is increased to 50. There is also added an additional hidden layer of 32 nodes before the final output layer to increase accuracy. The final model consists of hidden layers with 256, 128, 64, and 32 nodes. The activation function is not mentioned by Sengupta so ReLU is selected.

The second model is based on Ming et al. also mentioned in Section 3.3 with several changes. The output model layer is extended into three that represent dimensions. Hidden layers of the model are kept the same because the size of this model can be an advantage in comparison with the first model. The input of the DNN remains. It is the pair (u, v) , where u and v are the center coordinates of the detected object. They claimed that they

used Stochastic gradient descent (SGD) as an optimizer. The usage of this optimizer is not suitable for the case of this work. Initial tests indicated that the success of SGD strictly depends on input parameters of the training process that are not the same for different training datasets. These properties are highly unfavorable for a work focused on the automatic collection of training datasets. For this reason, SGD is replaced by Adam optimizer. Batch size is changed to 8 and a number of epochs remain 100. The activation function is kept as ReLU as in the original model.

6.4 Video and radar object and track association

In the video and radar object association section of the design, detected objects from each sensor are put in pairs. Both objects should respond to the same object from the real world. In this work, the spatial association is implemented using a Hungarian algorithm based on Pilgrim’s [17] work with some modifications. The score matrix for the Hungarian algorithm can be computed in both coordinate systems. However, this work is focused only method, which computes the score in the video image plane. The basic Hungarian algorithm is modified to accept as input any rectangular score matrix and threshold.

The main reason, why the threshold is implemented, is the robustness of the system. The system does not rely that every frame from every sensor will contain all real objects present on the scene. This means that even if the number of detected objects is same for both sensors, there is no guarantee that all objects will have its opponent detected in the second sensor. The threshold is set to a level which determines the maximal transformation error. The threshold level is determined experimentally with respect to the shape of the scene, calibration, and synchronization accuracy.

For computing association on plane, the score is computed as the Euclidean distance between the center of video detected object and the center of radar detected object projected in the image plane. Threshold is implemented as a linear function. The reason for this is the perspective. Objects closer to the camera are larger in the image plane, so larger center deviation can be tolerated. It was determined to adjust the linear threshold function in a way that the threshold value will be approximately half of the road lane. To increase association accuracy even more, a track direction filter is implemented. In traffic monitoring scenes is usual that vehicles that are getting closer to the position of the sensor are getting down in the image plane and vehicles that are getting away from the position of the sensor are getting up in the image plane.

Associating tracks

Despite the threshold used in the modified Hungarian algorithm, some objects can be possibly associated falsely. To increase the accuracy of the track association, partial associations from one moment are put into context with others. This approach does not necessarily lead to the loss of the possibility of real-time processing if only captures from the past are used. However, the best results are achieved with a delay when the track is lost because it left the observed scene. Track associating algorithm works with object ids. It counts the number of collected partial associations for each object id. The number of collected matches for each possible id pair is then transformed into a score matrix in a way that a lower score means more partial matches for that particular pair. The association is then decided by the same modified Hungarian algorithm as for the partial association solving the association for one moment.

To simplify the evaluation of the selected track association method. A simple visualizer is implemented. This visualizer reads tracks obtained from the track association and projects them into provided video. This visualization is beneficial especially to determine which vehicles in video are tracked properly. The screenshot of the visualizer is presented in Figure 6.5, where every red dot resembles the current position of one track.



Figure 6.5: Screenshot of visualizer of fused tracks

Chapter 7

Testing and evaluation

This chapter is focused on testing the methods implemented in this work. The set of frames used for the tests was collected from the traffic monitoring system with the help of this thesis supervisor. The system is placed on a four-lane highway. The hardware component responsible for radar detection is similar to the IWR6843 evaluation module with slight changes. The camera responsible for the video detection can be considered a standard single-lens industrial camera. The video frames from it have a resolution of 640×480 pixels.

7.1 Testing spacial calibration methods

The testing set of frames for spacial calibration is almost 5 minutes long and with 10 frames per second, it contains 2974 frames total, all with synchronized radar frames. The observed area is a road with 2 lanes in both directions. In the middle of a road is a road barrier that diverts traffic by direction of travel and also creates space between them on the video image plane. Data frames are processed according to the process described in Chapter 6.1 so each radar target has its three-dimensional coordinates and vector referring speed and each video target has its coordinates on the image plane.

The most straightforward method is unmoderated. It collects pairs only when one radar target and one video target are detected. Using this procedure, 32 pairs were collected and they are visualized in Figure 7.1, where each red dot represents the position of one pair in the image plane.



Figure 7.1: Radar and video pairs collected without any acceleration

Testing data collecting with division to areas method

For dividing areas method lines has to be defined. For line definition, there was used created tool mentioned in Section 6.2. In Figure 7.2 are visible manually adjusted lines. The orange line is defined by Equation 7.1. It was estimated that objects detected under this line should have a distance between the fusion device and the detected object at least 20 meters and above this line 30 meters. The blue line is defined by Equation 7.2.

$$v = -0.026u + 194 \quad (7.1)$$

$$v = -17.333u + 5560 \quad (7.2)$$

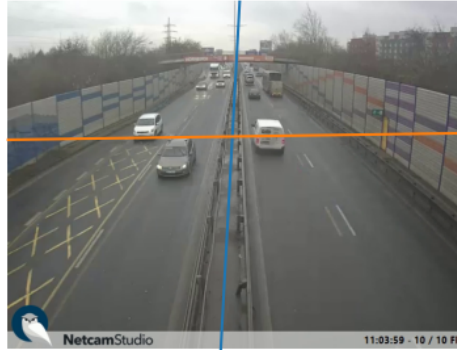


Figure 7.2: Dividing lines projected in video image plane

This method is divided into two experiments. The first experiment is focused only on a vertical line referring to the direction of vehicle travel and divides the observed area into two sub-areas. The second use both lines and divides the observed scene into four subareas. After all available frames were processed, 144 pairs were collected with the two areas method, and 1241 pairs were collected with four areas method. Video image plane coordinates of each pair are visualized in Figure 7.3.



(a) Two areas method

(b) Four areas method

Figure 7.3: Radar and video pairs collected by the two areas method and four areas method

Testing data collecting with approximated intrinsic and extrinsic matrix method

The first thing to do is to create an approximated intrinsic and extrinsic transformation matrix. The radar is placed close to the camera. This means that translation is not significant for calibration. Speaking of rotation, there is a need to take into account pitch, which is estimated with a calibration matrix visualizer presented in Section 6.2 at 19.42 degrees in direction to the ground. Roll and yaw are also probably present, but their effect seems difficult to discern by a human eye. That is why it is considered as irrelevant. The most difficult thing is to determine the size of the chip in combination with the focal length. For this reason, a calibration matrix visualizer is used to speed up the whole process. The scale coefficient, which resembles the distance in the image plane, was estimated at 0.827. The transformation matrix H_t based on parameters below is presented in Equation 7.3

$$H = \begin{bmatrix} 529.231 & 106.413 & 301.788 & 0 \\ 0 & 454.144 & 94.3481 & 0 \\ 0 & 0.332541 & 0.943089 & 0 \end{bmatrix} \quad (7.3)$$

After creating the initial matrix, every center of every radar target is transformed to the video image plane with time-corresponding video targets. The pairs are then created using the modified Hungarian algorithm. Since the created matrix is inaccurate, there is a need to filter out any targets for which it could be difficult to determine their belonging to a pair. For this additional filter is used to filter out any targets of the same sensor that are close to each other. For objects detected by radar, the close distance is determined at 3 meters, and for objects detected by video, it is described in Equation 7.4, where v is the vertical position in the image plane and d is minimal distance. The reason for using a function instead of a fixed value is for perspective, which is explained in further detail in Section 6.4.

With this method, 3255 pairs were collected. The result of collecting pairs is visualized in Figure 7.4, where each red dot refers to the image plane position of the collected radar and video pair.

$$d = 0.4375v - 25 \quad (7.4)$$



Figure 7.4: Radar and video pairs collected by the estimated intrinsic and extrinsic matrix method (AIEM)

Overall comparison of data collecting calibration methods

For comparative testing, a special dataset of pairs was created. All pairs used in this part were excluded from data collection, if they were used. The dataset has 2500 pairs of radar and video target coordinates.

Each collected set of pairs was then filtered with a density filter. All points are further than 0.75 meters from each other in the radar coordinate system. Then it was performed a discrete linear transformation with the least square optimizer and pairs which were significantly inaccurate (more than 100 pixel error in comparison with dataset value), were excluded from the next final iteration of DLT. Final results are presented in Table 7.1. The second column represents the number of pairs after filtering. The third and the fourth represent a mean error and its standard deviation in pixels. The fifth column represents maximal error. This can be misleading because they are proportional to video input resolution. That is why the sixth column is introduced to present errors relative to the resolution with the equation $E_r = E_m / (imageWidth + imageHeight)$, where E_r is the relative maximal error and E_m is the maximal error.

Table 7.1: DLT with normalized values, least squares optimization and after filtering of probable error

Collecting method	Filtered pairs	Mean error [pixel]	Standard deviation [pixel]	Maximal error [pixel]	Relative maximal error
Unmoderated	28	119.5	602.3	24681.5	22.04
Two areas	103	13.3	12.2	162.2	0.14
Four areas	435	11.7	8.8	89.5	0.08
AIEM	981	11.6	9.0	90.1	0.08

In Table 7.1 is visible that the most successful methods are dividing the observed area into four subareas, which includes creating two dividing lines, and approximated intrinsic and extrinsic matrix (AIEM). However, AIEM is not reliable. Parameters for this accurate AIEM were obtained after several attempts. Most of them were worse than the four areas method. For this reason, the most successful method should be the four areas, which is more reliable. The method that fails the most is the unmoderated, waiting for only one detected radar and video target. This method is not able to collect sufficient amounts of points from the presented dataset and fails to compute the transformation matrix.

Initially, there was a idea that the more points are collected, the more accurate the matrix is. Unfortunately, this statement is only half the truth, because, despite all the filtering, each method can still collect invalid pairs. This means that there is an ideal relation between the number of pairs collected and the number of invalid pairs.

The success of all the collection methods can vary depending on the values chosen in the filtering. The matrix collecting method's success depends on the initial computed intrinsic and extrinsic matrices and association algorithm.

One of the filters, which can strongly improve accuracy, is the density filter. The density filtering discards several pairs from the collected set. This means that density filtering is losing the accuracy of the transformation matrix at some point. If the collected pairs are scattered uniformly, there is basically no need for density filtering. If there is a dense cluster of pairs in space and areas around are rare for them, density filtering is needed. For the

presented testing dataset, the density filter was set to create a minimal distance between radar targets in set to 0.75 meters.

Another filtering method used is the threshold and number of iterations for filtering probable errors. For the presented testing dataset, the error threshold was set to 100 pixels with one iteration. This means that after the first DLT was computed, points with an error greater than 100 pixels were excluded from the second final DLT.

Speaking about the complexity of collecting methods for a user, it is hard to determine, which method is the most complex to carry out. The intrinsic and extrinsic matrix method require user expertise in the calibration to approximate the matrix, if all needed camera and radar parameters are not known. Its performance is strongly associated with the accuracy of the intrinsic and extrinsic matrix created. The divided areas method requires less expertise but requires additional information about the observed area in the form of an approximation of the distance of the dividing line from the sensor.

In Figure 7.5 it is visible that the error is decreasing with increasing distance of detected objects from the positions of sensors. This effect is due to the perspective of the image and it is important for the next section dedicated to testing association.

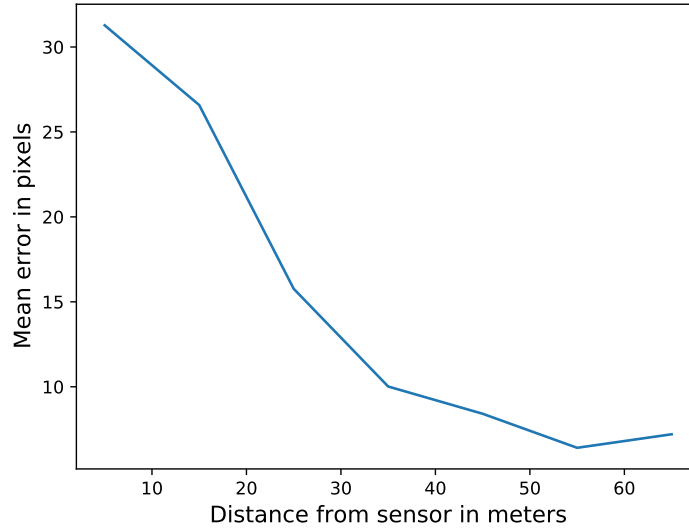


Figure 7.5: Relation between mean error and object distance

In Equation 7.5 is presented transformation matrix, which is used in other parts of testing if a transformation matrix is needed.

$$H = \begin{bmatrix} -15.5978 & 6.27496 & -8.44150 & -41.0124 \\ 1.62089 & -4.49439 & -1.68948 & -80.9574 \\ 0.000520114 & 0.0195183 & -0.0263375 & -0.186418 \end{bmatrix} \quad (7.5)$$

7.2 Testing association

The testing dataset is the same as the one presented in Section 7.1. It is an almost 5 minute long capture of video and radar frames. The setup for testing the partial association consists of a transformation matrix and adjusting the threshold. The provided transformation

matrix is the best transformation matrix computed in the previous section shown in Equation 7.5. The threshold function was roughly approximated by eye with two points. The first point was estimated to be 100 pixels at the 450 vertical pixel line, which corresponds to approximately 5 meters distance. The second point was estimated to be 10 pixels at 120 pixel line which corresponds to approximately to 70 metres distance from sensors in radar coordinate system. This was made in respect of the recommendation mentioned in the implementation chapter, which considers a half of the road lane as a good threshold distance. The used threshold function is in Equation 7.6, where v is the vertical position of the video object detected in the image coordinates and t is the threshold.

$$t = 0.3v - 20 \quad (7.6)$$

The track association was conducted after the whole partial association was performed. This lead to the loss of ability to process frames in real-time or at least the most trusted results are not presented at the end of the linked dataset, but after several frames pass through the system and objects making tracks leave the observed area.

Table 7.2: Results of track association

Metrics	Fusion	Radar	Video
Number of tracks	366	720	484
Mean track length	41.58	17.15	32.77
Standard deviation of track length	11.76	9.71	17.88
Small time gaps	66	6	120
Big time gaps	81	6	106

The result of track association is presented in Table 7.2. The track fusion algorithm received 720 radar tracks and 484 video tracks. Both values are quite out of context because manually in the testing set, there were counted approximately 375 vehicles. The fusion algorithm was able to associate 366 tracks, where the track contain at least one piece of information from each sensor. The rest of the tracks were unable to associate using the selected method, probably because they were not properly detected by radar and video at the same time. However, this result better matches to the real number of vehicles than the video or radar tracking algorithm alone. Especially the radar detection algorithm seems to be unreliable because it was detecting several objects that were, according to it, traveling in the opposite direction than they were supposed to, which is definitely untrue according to time-synchronized video images. Several larger vehicles, such as trucks and buses, were also sometimes detected as two or more targets by the Gtrack algorithm. According to the visualizer presented in section 6.4, this issue was at least partially solved by fusion.

The track association algorithm was also able to extend tracks. In comparison with radar, the fusion model has on average 142 % longer tracks, and in comparison with video, it has on average 27 % longer tracks. This effect is also made by the nature of the association itself. The track association favors long paths over short ones. Some video tracks are too short and so they cannot get enough score in the score matrix for the Hungarian association.

Considering time continuity, there are also changes. Before track association, there were 120 time gaps when video tracks had been lost for a one frame and 106 time gaps when video tracks had been lost for more than one frame. Radar targets were almost perfectly continuous. After track association, there are 66 time gaps when tracks have been lost for a one frame and 81 time gaps when tracks have been lost for more than one frame. Relative

to the number of detected tracks, there are 27 % fewer small gaps and 1 % more big gaps than video tracking alone. This can be considered a slight improvement, but exact values can probably change drastically in other possible experiments. The main point of this evaluation technique is to prove that fusion serves more continuous tracks because when video tracks are lost, radar tracks can replace them, and conversely.

7.3 Testing depth prediction methods

For testing the computing depth methods, a pair set was collected using the association method from the previous section. Input parameters are set as in these sections (equations 7.6 and 7.5). Thanks to this method, from the same set of frames as is presented in Section 7.1 it collected 7355 pairs. The dataset is split into training and testing sets. The training set is first 6500 pairs and the rest is a testing set. In Figure 7.6 is the dataset visualized in the camera image plane.



Figure 7.6: Video pairs collected by the two areas method and four areas method

In plane and transformation matrix methods, the training part of dataset is used only for approximating the plane. The calculated plane for the one plane method is presented in Equation 7.7, where x , y , and z are coordinates from the radar system.

$$6.94x - 140.23y + 578.28 = z \quad (7.7)$$

The computed planes for the two planes method requires also a class of each object. The class was collected in pairs from the object detected by video. Results are presented in Equation 7.8 for cars and 7.9 for trucks and buses.

$$3.97x - 78.83y + 367.35 = z \quad (7.8)$$

$$-15.35x + 498.54y - 1653.37 = z \quad (7.9)$$

Considering DNN models, their parameters are according to Section 6.3.

Overall comparison of computing depth

Results of tested methods are presented in Table 7.3. Values used in this table refer to the Euclidean distance between the approximated point and the base truth point. However, the error distribution function is not the same on every axis as is presented in Figure 7.7. This Figure shows that in absolute values, an error has a more significant effect on z axis representing the distance of the detected object from sensors or depth for the video sensor.

Table 7.3: Results of computing depth

Method	Mean error [m]	Standard deviation [m]	Maximal error [m]
One plane	5.32	3.42	18.05
Two planes	4.62	3.17	17.20
DNN Sengupta	2.00	1.48	9.93
DNN Ming	5.29	4.26	23.78

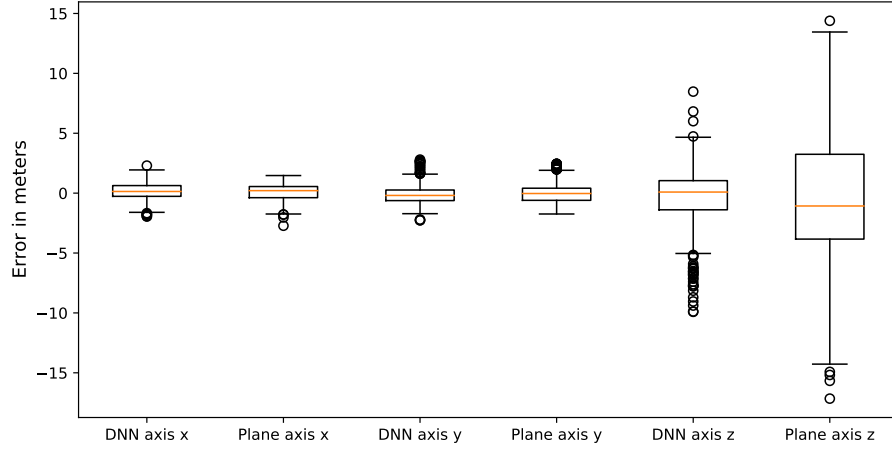


Figure 7.7: Error of depth prediction on different axis for the best method of each type

From Table 7.8 is visible that DNN method of depth prediction model based on Sengupta is the best method implemented. The failure of DNN method based on Ming is mostly because of model input. Ming et al. used this model to predict the distance of objects that were further from the camera and it seems that most of the detected vehicles had approximately the same size. For this reason, it was possible to reduce input to only the center coordinates of objects detected by video. In this work, the size of detected vehicles in the video is changing drastically. There are represented both large trucks and small city cars. To prove this fact, DNN based on Ming was also trained with the same input format as DNN based on Sengupta and it has achieved almost the same accuracy as it. This claim is also supported by Figure 7.8, where DNN based on Ming is slightly improving at a distance interval from 45 meters to 60 meters.

It is important to mention that accuracy can be disrupted by the transformation matrix. Both types of methods are using it, but each type for a different purpose. DNN methods are using it only to associate detected objects from sensors so if the association is performed correctly, there is no error transferred. On the other hand, plane computation methods are

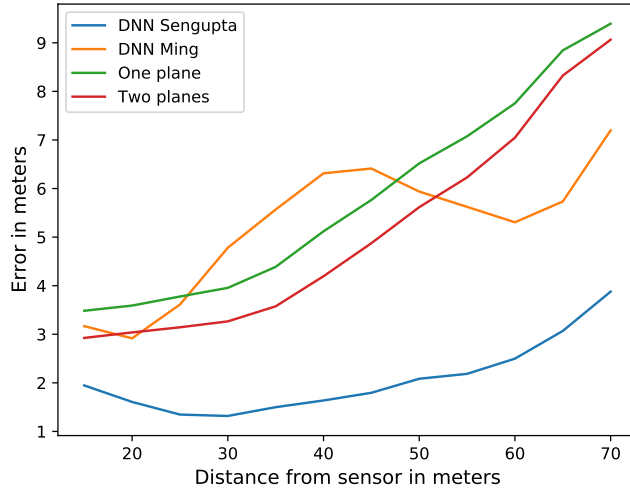


Figure 7.8: Relation between mean prediction error and real object distance

using it directly in the equations. This also means that any change in the transformation matrix has a much more significant effect on the result than in DNN methods. Another reason is the position of radar targets on the vertical axis. Their position seems too scattered within the space and their approximation using a plane is therefore inaccurate. This problem is partially overcome with the two planes method as shown in Table 7.3 and Figure 7.8, but not solved entirely. DNN methods also have some shortcomings. Their results depend on initialization values that can possibly lead to worse results than presented. Despite this, there is a still high chance that they will be more accurate than plane methods.

In Figure 7.8 is visible that the mean error is increasing in distance from sensors. This effect is made by perspective, which results in a very small difference between the coordinates of objects in the image plane at such a distance and therefore very small difference of input parameters for each depth prediction method.

Despite the result of plane methods, they should not have been considered a dead end. In comparison with the DNN methods used, plane or in the future more advanced surface methods can be naturally more mathematically straightforward than DNN and so they should have a smaller number of operations needed to compute a result, if not minimal. DNN methods require except transformation matrix also an additional dataset to train the neural network, which has to be usually bigger than the minimum required dataset for plane methods. Dataset for DNN also needs associated objects which are much harder to obtain than just a set of radar object centers to approximate the plane of road used in the first type of methods. DNN methods are also not that reliable because of the initial weights of the model. Despite a sufficient dataset, bad initial weights can result in poor performance of this methods. If in the future radar detection algorithms will serve more information about the real position of objects and their dimensions, there will be ways to improve their performance to the same level as the DNN methods has nowadays.

To sum it up, the most successful depth prediction method is a deep neural network that is based on a modified model presented by Sengupta et al. This model is also used in the track association algorithm. Every video prediction made in the final output is predicted using this model.

Chapter 8

Conclusion

The main goal of this work was to increase the quality of object tracking in three-dimensional space using the fusion of radar and video data obtained from traffic monitoring scenes. The topic was studied in several chapters to outline the current state of methods and their issues. Then several methods were modified and implemented to outline the likely best approach from this set of methods. Methods were also tested on a dataset collected from a real-world environment and evaluated.

The solution includes semi-automatic methods of collecting radar-video pairs of detected objects to compute the transformation matrix using the DLT method with a least squares optimizer. There were tested methods, which included creating a rough approximation of the intrinsic and extrinsic transformation matrix to create possible pair associations. There were also tested methods that included dividing the scene into areas to determine association without knowledge of the transformation matrix. The most successful was the method, which divided the scene into four subareas. On a less than 5-minute-long set of frames with approximately 375 vehicles and a resolution of 480×640 pixels, the four subareas method was able to collect a sufficient number of points to cover the whole scene with radar-video object pairs. Combining with the small error of pairs falsely associated, this method was able to create a transformation matrix, which has a mean error of approximately 12 pixels.

Another important part was the depth prediction. To complete this task, two types of methods were implemented. The first type of methods were approximating depth using a set of equations, including the transformation matrix and plane approximation. The second type of method was to use a deep neural network. Using the transformation matrix and partial association algorithm, the fusion module was able to automatically collect the dataset for training. Despite surprisingly good results of plane approximation methods, the most successful method was one based on DNN. The best implemented DNN reached a mean error of around 2 meters.

The last part was the association. The association was performed on two stages. The first stage was to associate objects detected at one point in time. The issue was solved using the Hungarian algorithm with several modifications. One of the modifications was to implement a threshold for the score matrix. This had a significant impact on the result because the model does not rely on the fact that all objects detected by the first sensor are also detected by the second, so it gives the system desired durability. The second stage was focused on the association of tracks using the already mentioned Hungarian algorithm. The score in the score matrix was computed as the number of affirmative associations from the previous step for each possible pair. This approach was able to get closer to the real number of tracks in the testing set than video or radar data alone. It also extended radar

and video tracks. The mean track length of the fusion system is around 25 % longer than the mean length from the video tracker alone and almost two and half times longer than the mean length from the radar tracker alone.

Future work can be focused on approximating the real position of objects in a scene based on Bayesian models with respect to radar and video position estimation accuracy. This will also probably include creating a dataset, which will have the positions of real-world objects measured by more accurate technology than video or FMCW radar. Another future work can be focused on a different direction of training than that implemented or training the radar on a dataset obtained from video data. In the future, there is a possibility of experimenting with different scoring functions that will possibly include velocity or class probability that can request transforming velocity vector of objects detected by video from the image plane to the radar coordinate system or the ability to classify objects detected by radar. Further, it is possible to implement mechanisms that will determine the amount of trust in radar or video data based on conditions affecting the accuracy of individual sensors, such as traffic density, visibility, or weather. This work can also be used as a reference for solving the task at a different level of data fusion than the stated decision level.

Bibliography

- [1] ABDU, F. J., ZHANG, Y., FU, M., LI, Y. and DENG, Z. Application of Deep Learning on Millimeter-Wave Radar Signals: a Review. *Sensors*. Xiamen, China: MDPI Open Access Journals. 2021, vol. 21, no. 6. DOI: 10.3390/s21061951. ISSN 1424-8220. Available at: <https://www.mdpi.com/1424-8220/21/6/1951>.
- [2] ABDULWAHAB, S., RASHWAN, H. A., SHARAF, N., KHALID, S. and PUIG, D. Deep Monocular Depth Estimation Based on Content and Contextual Features. *Sensors*. 2023, vol. 23, no. 6. DOI: 10.3390/s23062919. ISSN 1424-8220. Available at: <https://www.mdpi.com/1424-8220/23/6/2919>.
- [3] B. W. PARKINSON AND J. J. SPILKER. *Global positioning system : theory and applications*. Washington: American Institute of Aeronautics and Astronautics, 1996. Progress in astronautics and aeronautics ; vol. 163-164. ISBN 1-56347-106-3.
- [4] BHUYAN, M. K. *Computer vision and image processing : fundamentals and applications*. Boca Raton: CRC Press, 2020. ISBN 978-0-8153-7084-0.
- [5] BROWNLEE, J. *A Gentle Introduction to Object Recognition With Deep Learning* [online]. Machine Learning Mastery, 22. may 2019. Revised 27.1. 2021 [cit. 2023-03-02]. Available at: <https://machinelearningmastery.com/object-recognition-with-deep-learning/>.
- [6] CESAR IOVESCU, S. R. The fundamentals of millimeter wave radar sensors. Dallas: Texas Instruments. july 2020, p. 1–8. Available at: <https://www.ti.com/lit/wp/spyy005a/spyy005a.pdf>.
- [7] CHENG, Y., SU, J., JIANG, M. and LIU, Y. A Novel Radar Point Cloud Generation Method for Robot Environment Perception. *IEEE Transactions on Robotics*. IEEE. 2022, vol. 38, no. 6, p. 3754–3773. DOI: 10.1109/TRO.2022.3185831. ISSN 1552-3098.
- [8] GAO, B., ZHENG, K., ZHANG, F., SU, R., ZHANG, J. et al. Research on multi-target tracking method based on multi-sensor fusion. *Smart and Resilient Transportation*. Emerald Publishing Limited. Jan 2022, vol. 4, no. 2, p. 46–65. DOI: 10.1108/SRT-05-2022-0010. Available at: <https://doi.org/10.1108/SRT-05-2022-0010>.
- [9] HU, J.-J., CHEN, W.-Y. and HUANG, S.-Y. The Study of the Object Depth Estimation Based on Point Pair Features on Images. In: *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul 2019, p. 217–220. DOI: 10.1109/SNPD.2019.8935808. ISBN 978-1-7281-1652-5.

- [10] JOCHER, G., CHAURASIA, A. and QIU, J. *YOLO by Ultralytics*. January 2023 [cit. 2023-05-01]. Available at: <https://github.com/ultralytics/ultralytics>.
- [11] KHAN, M. and ANWAR, S. Paradox Elimination in Dempster–Shafer Combination Rule with Novel Entropy Function: Application in Decision-Level Multi-Sensor Fusion. *Sensors*. Indianapolis: [b.n.]. november 2019, vol. 19, p. 4810. DOI: 10.3390/s19214810.
- [12] KUHN, H. W. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*. march 1955, vol. 2, 1-2, p. 83–97. DOI: <https://doi.org/10.1002/nav.3800020109>.
- [13] LIM, T.-Y., ANSARI, A., MAJOR, B., FONTIJNE, D., HAMILTON, M. et al. Radar and Camera Early Fusion for Vehicle Detection in Advanced Driver Assistance Systems. *NeurIPS Machine Learning for Autonomous Driving Workshop*. 2019.
- [14] LIU, M., LI, D., LI, Q., LU, W. and YIN, J. An Online Intelligent Method to Calibrate Radar and Camera Sensors for Data Fusing. *Journal of Physics: Conference Series*. IOP Publishing. sep 2020, vol. 1631, no. 1, p. 012183. DOI: 10.1088/1742-6596/1631/1/012183. Available at: <https://dx.doi.org/10.1088/1742-6596/1631/1/012183>.
- [15] OH, J., KIM, K.-S., PARK, M. and KIM, S. A Comparative Study on Camera-Radar Calibration Methods. In: *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2018, p. 1057–1062. DOI: 10.1109/ICARCV.2018.8581329. ISBN 978-1-5386-9583-8.
- [16] PAN, G., GUO, Z., NIU, Z. and ZHAO, J. Radar and video fusion to identify vehicle tracks. In: *2021 6th International Conference on Smart Grid and Electrical Automation (ICSGEA)*. Kunming, China: [b.n.], 2021, p. 235–237. DOI: 10.1109/ICSGEA53208.2021.00058. ISBN 978-1-6654-3264-1.
- [17] PILGRIM, R. Tutorial on Implementation of Munkres’ Assignment Algorithm. Murray: Murray State University. august 1995, p. 1–13. DOI: 10.13140/RG.2.1.3572.3287.
- [18] REDMON, J., DIVVALA, S., GIRSHICK, R. and FARHADI, A. You Only Look Once: Unified, Real-Time Object Detection. Washington: arXiv. 2015. DOI: 10.48550/ARXIV.1506.02640. Available at: <https://arxiv.org/abs/1506.02640>.
- [19] REICH, B. *Exploitation of Neural Networks for Fusion of Image and Non-Image Data*. Brno, CZ, 2022. Master Thesis. brno university of technology, Faculty of Information Technology. Available at: <https://www.fit.vut.cz/study/thesis/22713/>.
- [20] SENGUPTA, A., CHENG, L. and CAO, S. Robust Multiobject Tracking Using Mmwave Radar-Camera Sensor Fusion. *IEEE Sensors Letters*. IEEE. 2022, vol. 6, no. 10, p. 1–4. DOI: 10.1109/LSSENS.2022.3213529. ISSN 2475-1472.
- [21] SENGUPTA, A., JIN, F. and CAO, S. A DNN-LSTM based Target Tracking Approach using mmWave Radar and Camera Sensor Fusion. In: *2019 IEEE National Aerospace and Electronics Conference (NAECON)*. IEEE, 2019, p. 688–693. DOI: 10.1109/NAECON46414.2019.9058168. ISSN 2379-2027.

- [22] SMID, M. and MATAS, J. Rolling Shutter Camera Synchronization with Sub-millisecond Accuracy. In: Porto, Portugal: VISAPP - 12th International Conference on Computer Vision Theory and Applications, 2017. Available at: https://cmp.felk.cvut.cz/~smidm/download/visapp2017_camera_ready_web.pdf.
- [23] SZELISKI, R. *Computer Vision : Algorithms and Applications*. London: Springer, 2010. Texts in computer science. ISBN 978-1-84882-934-3.
- [24] TEXAS INSTRUMENTS. *Tracking radar targets with multiple reflection points*. 1st ed. Dallas: Texas Instruments, april 2017 [cit. 2023-02-03].
- [25] TEXAS INSTRUMENTS. *60GHz mmWave Sensor EVMs*. 1st ed. Dallas: Texas Instruments, october 2018 [cit. 2023-02-03]. 1–97 p. Available at: <https://www.ti.com/lit/ug/swru546e/swru546e.pdf?ts=1683021315497>.
- [26] TEXAS INSTRUMENTS. *Understanding the UART Data Output Format* [online]. 2021, Revised 7.8. 2022 [cit. 2023-05-02]. Available at: https://dev.ti.com/tirex/explore/node?a=VLyFKFf__4.9.0&devtools=IWR6843ISK&node=A__ADnbI7zK9bSRgZqeAxpvrQ__com.ti.mmwave_industrial_toolbox__VLyFKFf__LATEST.
- [27] WELCH, G. and BISHOP, G. An Introduction to the Kalman Filter. *Proc. Siggraph Course*. Chapel Hill: The University of North Carolina at Chapel Hill. january 2006, vol. 8, p. 16.
- [28] XU, G., LI, D., CHEN, X., LI, Q. and YIN, J. A Modified Online Intelligent Method to Calibrate Radar and Camera Sensors for Data Fusing. *Journal of Physics: Conference Series*. Beijing: IOP Publishing. sep 2021, vol. 2025, no. 1, p. 012007. DOI: 10.1088/1742-6596/2025/1/012007. Available at: <https://dx.doi.org/10.1088/1742-6596/2025/1/012007>.
- [29] YILMAZ, A., JAVED, O. and SHAH, M. Object tracking: A survey. *ACM computing surveys*. Baltimore: ACM. 2006, vol. 38, no. 4, p. 13–es, [cit. 2023-03-04]. ISSN 0360-0300.
- [30] ZHANG, G., GENG, X. and LIN, Y.-J. Comprehensive mPoint: a Method for 3D Point Cloud Generation of Human Bodies Utilizing FMCW MIMO mm-Wave Radar. *Sensors*. 2021, vol. 21, no. 19. DOI: 10.3390/s21196455. ISSN 1424-8220. Available at: <https://www.mdpi.com/1424-8220/21/19/6455>.
- [31] ZHANG, Y., SUN, P., JIANG, Y., YU, D., WENG, F. et al. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. *Proceedings of the European Conference on Computer Vision (ECCV)*. Tel Aviv: Springer. 2022. ISSN 978-3-031-19769-7.
- [32] ZHOU, Y., DONG, Y., HOU, F. and WU, J. Review on Millimeter-Wave Radar and Camera Fusion Technology. *Sustainability*. 2022, vol. 14, no. 9. DOI: 10.3390/su14095114. ISSN 2071-1050. Available at: <https://www.mdpi.com/2071-1050/14/9/5114>.