Budapest University of Technology and Economics (BME)

FINAL THESIS TRANSPORTATION ENGINEERING MASTER

DEVELOPMENT OF A VISION-BASED FEATURE FOR AUTOMATIC GUIDED VEHICLES (AGV'S)

Author: Jesús Abellán López

Tutor: Dr. Bohács Gábor

Academic Year: 2018-19

Table of contents

Table of Figures

1. Introduction to Industry 4.0

In the past, manufacturing processes were limited to only one factory. However, complexity of production networks is growing enormously in the next years. Boundaries between individual manufacturing plants will no longer exist in the future. Instead, they will be removed in order to interconnect a huge number of factories or even geographical regions. It is interconnection the key fact that represents the beginning of a new industrial era. Until arriving here there have been a series of revolutions in the past:

The First Industrial Revolution took place between the years 1760 and 1840. It began in Great Britain and the core aspect was the usage of steam to power the factories in addition with the mechanization of the production process going from hand-based production methods to machines.

With the Second Industrial Revolution (1870-1914), also known as the Technological Revolution, the mass production model was implemented which meant an automatization of the production process. In addition, electricity started to be used in the assembly lines. It was a period of rapid industrial development specially in Britain, Germany and the United States.

Finally, in 1960 the emergence of computer driven systems meant the beginning of the Third Industrial Revolution or the Digital Revolution It symbolized the advancement of technology from mechanical devices and analog electronic to the digital technology.

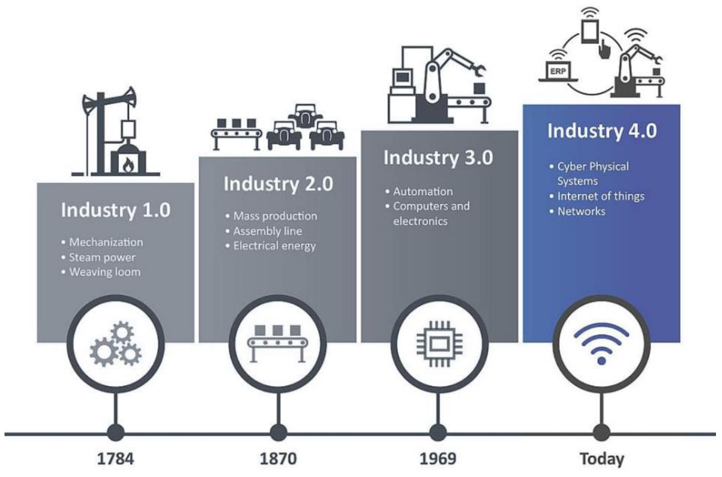


***Figure 1**: Industry evolution*[1]

[1] Consulted in *https://dzone.com/articles/industry-40-the-top-9-trends-for-2*

Nowadays, Industry 4.0 is considered as the Fourth Industrial Revolution and it has recently started. Its development will continue towards the second and third decades of the 21st century. It is about interconnection and it implies a radical change in the way industry responds to society’s needs.

The term "Industry 4.0", I4.0 or simply I4, originates in 2011 from a German project whose aim was to promotes the computerization of manufacturing processes. Later, it was spread around all Europe until nowadays when it is strongly established in most European countries.

If previous industrial revolutions were guided by innovations in manufacturing processes, the advancement of Industry 4.0 will be driven by a smart and interconnected environment. It means a new era for industrial production.

Industry 4.0 is a complete restructuring of production processes, transforming analog and centralized workflows into digital and decentralized production processes. With these new methods it will be possible to predict, control, plan and produce in a more intelligent way. This implies a higher level of automation and the digitalization of the different machines and factories.

This new industrial revolution will be a reality thanks to both cyber physical systems and the Internet of Things:

Cyber physical systems are production and logistics units controlled by algorithms based on computing and integrated with the internet. In this way, these systems have the possibility of controlling physical objects making use of virtual networks. Production will involve machines and objects been equipped with sensors that continuously collect data about status, location and usage patterns When combined and analysed accordingly these data will lead to much more efficient processes and to optimal and preventive maintenance of machinery and equipment.

The other key element of this new revolution is the Internet, more specifically, the Internet of Things (or its acronym IoT). IoT is a concept that refers to a digital interconnection of everyday objects with the internet.

It is, in short, the linking of the internet more with objects than with people. With it, products and devices are automatically identified by intelligent sensors and linked to each other. In this way, valuable information is obtained and analysed.

All things considered; it asis worth pointing out that the Fourth Industrial Revolution is essentially the Internet of Things applied to manufacturing processes. The physical systems communicate and cooperate both with each other and with human workers remotely via the wireless web.

By using Industry 4.0 technologies like cyber-physical systems, big data analytics and cloud computing, products will autonomously decide where and how they are produced. It will also help in the detection of production failures, enabling their prevention and increasing productivity and the company’s benefits.

1.1 Artificial intelligence and its applications

The main goal of Industry 4.0 is to achieve the correct functioning of a big number of the so named "smart factories". These new manufacturing plants will be able to meet production necessities in a much more efficient way. As part of the Industry 4.0 ecosystem, they are characterized by the interconnection between the different machines of the production process and by the continuous information exchange with their external environment: market levels of offer and demand, external competitors, potential buyers…

Artificial intelligence (AI) is considered as the key element of this transformation and it is strongly related with the storage of huge amounts of data. (big data), the usage of algorithms to process them and the interconnection. AI is an area of computer science that emphasizes the creation of intelligent machines that work and react like humans. These processes include learning (the acquisition of information and rules for using the information), reasoning (using rules to reach approximate or definite conclusions) and self-correction.

In brief, the use of artificial intelligence is essential in order to strengthen Industry 4.0 processes.

The application range of artificial intelligence and Industry 4.0 is very wide. Below, just a few of its possible applications are explained[2]:

- Construction: one of the most significant examples of how Industry 4.0 is changing the construction industry is the Semi-Automated Mason or SAM. It is a robot capable of laying 3000 bricks per day, with no need of break and no mistakes. Its usage in construction will both increase productivity while reducing overall labour costs.

- Public transportation: Industry 4.0 is being introduced in public transport networks with the aim of creating a sustainable model for cities and preserve life quality of their citizens. Thanks to artificial intelligence most of the public

[2] Consulted in *https://medium.com/@viarbox/a-few-real-world-examples-of-industry-4-0-8e2de4f4f23e*

transport media will no longer need drivers thus increasing the productivity and efficiency of the service.

- Food industry: this industry is also experiencing a radical change in its production processes thanks to automation. Farmers need to produce more food more sustainably and from the same amount of land. Autonomous tractors will be able to work for 24 hours in a highly efficient and secure way. Also, thanks to the flexibility of new technologies, farmers have access to real time data wherever they are. One clear example can be seen in the milking area. This natural process has been completely automatized in the recent years thanks to robotics. In addition, big data analytics allows the industry to have a much more precise control of the milking process and how each cow is responding to it.

- Factory Logistics: AI has been introduced in production plants and distribution centres where all the necessary transport operations are made in a completely autonomous way thanks to robots and remoting control. This implies the decrease of human labour necessity and the rise of Automatic Guided Vehicles (AGV). This new technology for vehicles will be explained in more detail in the following points.

1.2 Autonomous Guided Vehicles (AGV's)

Industry 4.0 development cannot be understood without Automatic guided vehicles (AGV). Nowadays industry is facing a more and more complex scenario. This situation implies that it is not enough to work in the conventional way, processes are being pushed to automation.

In addition, safety and speed are crucial factors for the success of material flow and can only be achieved by adapting warehouse transportation processes. During transportation most accidents and damages are caused by human errors so this adaptation to the new environment must go in the way of automation and the decrease of human intervention in the transportation process, from the production plant until the warehouse where the selected product is stored.

As a result, the vehicle automation for storage and handling equipment becomes necessary in warehouses implementing Industry 4.0. This requirement is fulfilled with the use of AGV's.

Automatic guided vehicles (AGV)[3] are mobile robots that follow floor markers, or use vision, magnets, or lasers to move following a certain pathway. The pathway must be preprogramed and must be simple and unobstructed for the AGVs. These self-driving vehicles are commonly used in production plants and distribution centres in order to transport materials, in special, pallet handling is one of its most used applications. The idea of using an automated guided vehicle system is based on the concept of optimization.

Their use results in time, energy and space savings. They are characterized by high efficiency, reliability and complete flexibility in terms of route configuration and planning. Problems caused by human error are solved: no accidents, no stress and no wasted journey. In addition, another advantage of AGVs are the low maintenance costs.

[3] Consulted in *https://www.jungheinrich.ie/products/automated-solutions-agv/*

1.2.1 Already existing pallet trucks

As mentioned before, pallet handling within a warehouse is one of the most popular applications of AGVs.

Traditionally, in industry this function has been implemented making use of pallet jacks. There are two main types:

- Manual pallet truck**:** also called hydraulic pallet truck since the lifting of the forks is achieved with a hydraulic piston. This kind of trucks are powered by the oscillating movement that the operator performs when he or she wants to raise the load. On the other hand, descending movement is done by releasing a lever that acts on a discharge valve that returns the fluid to the tank, emptying the piston by gravity.

Manual pallet trucks are recommended for sporadic use, when the load to be transported is not very heavy and when the distances of displacement are not very large.

They can lift a weight of up to 2500 kg and up to a height of 8-10 cm.

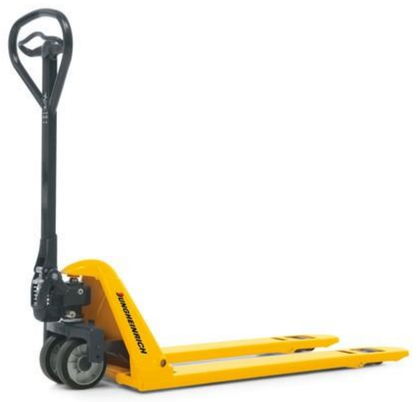


Figure 2*: Manual pallet truck from Jungheinrich*[4]

[4] Consulted in: *https://www.jungheinrich.co.uk/products/new-forklifts/pallet-trucks/hand-pallet-trucks*

- Electric pallet truck**:** it is the evolution of manual pallet trucks. These devices are equipped with batteries and an electric motor that performs the functions of displacement and elevation. They are used when loads or distances surpassed the limits of manual manipulation or when the frequency of use is too high.

Electric pallet trucks work with batteries that provide a range of autonomy of 8 to 10 hours.

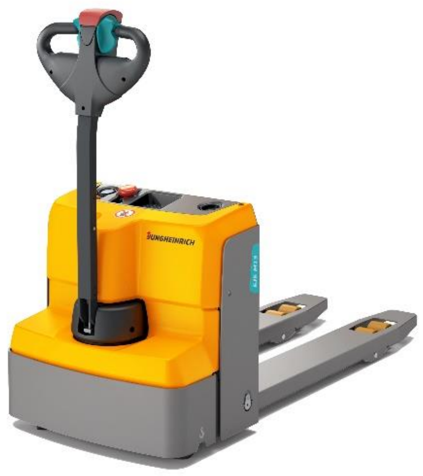


Figure 3*: Electric pallet truck from Jungheinrich*[5]

[5] Consulted in: *https://www.jungheinrich.ie/shop/jungheinrich-shop/electric-pallet-truck-eje-m1315/*

1.2.2 AGV Automatization

The aim of this project is to automatize a forklift, another type of auto pallet mover, very used in warehouses and logistic centres which is the evolution of the traditional pallet trucks by making them more compact and handled. By means of a laser sensor, a camera and a software the proposed forklift will be completely automated and so there will be no need of driver, increasing the company's productivity and efficiency.

A forklift is a vehicle used to transport pallets, by means of two flat parallel bars located on the front called "forks". These forks are inserted into the pallet in order to move it. They are mounted on a support that slides vertically by a mast allowing the movement of lifting and lowering the load.

The lifting of the forks and the inclination of the mast are made by hydraulic pistons driven by a pump to which a motor supplies the power. This type of vehicle keeps the weight of the load out of the plane formed by its four wheels. Therefore, in order to maintain it balanced, they are equipped with a counterweight on the back.

***Figure 4**: Forklifts from Jungheinrich*[6]

Nowadays, forklifts are agile and highly efficient vehicles in terms of performance and energy savings. Within a company they save hours of work since they allow the movement of very heavy loads at once and with no need of human strength.

[6] Consulted in *https://www.gwequip.com/jungheinrich-forklifts/*

On the other hand, they present as a disadvantage the fact that their use requires some specific previous training by the driver.

With this project it is intended to get rid of that disadvantage, automating not the forklift itself but the driver and turning it into an automated guided vehicle.

The proposed machine to be automated is a forklift that belongs to the Continental company[7], a German group specialized in automotive manufacturing and transport industries. In the following picture it is shown the specified forklift:

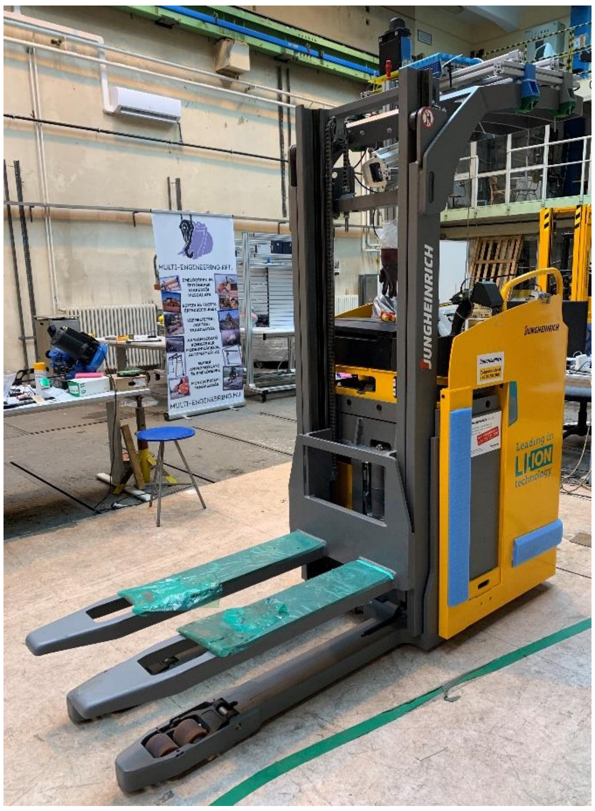


Figure 5: *Jungheinrich ESC 214/216*

It is a model from Jungheinrich AG, specifically, Jungheinrich ESC 214/216. These electric sideways are suitable for longer transport distances during stacking and retrieval. With a width of just 820 mm and a short working aisle width, the ESC model shows a strength in applications where space is limited and in block storage warehouses.

7 https://www.continental-corporation.com/hu-hu

Some of its most important features are[8]:

• Powerful, innovative 2.8 kW three-phase drive motor for high acceleration and performance

• Safe speed regulation by electronic Speed-Control.

• High residual capacities and powerful lift motor.

• Extremely durable frame for very high operational demands.

• Display with steering position display and travel program selection (optional).

• Electric steering for easy manoeuvring with a few turns of the steering wheel

• Excellent energy management for low operating costs.

•Narrow working aisle width and well protected operator position for out-standing operation in narrow working aisles.

• Ergonomic layout of all controls.

• Comfortable seating position with height-adjustable footplate and adjustment for body weight.

The initial lift of the ESC 214z/216z allows it to cope extremely easily with uneven floor conditions. All travel and lift functions are performed using the multifunction lever which is within easy reach. In addition, it is an electric truck using a lithium battery.

From the data sheet, its basic features are known:

Rated load capacity	1400 kg
Lift	2900 mm
Travel speed	9,1 km/h
Battery voltage	24 V
Drive power	2,8 kW

[8] Consulted in *https://www.jungheinrich.es/productos/vista-rapida-a-nuestras-carretillas/apiladoras/esc-214214z216216z/*

In addition, all the technical data can be found in the complete data sheet of the model attached in the annexes chapter.

With the aim of its automation, this forklift, apart from the forks used to handle the pallet, the mast and the overhead guard, is equipped with a camera located on the top of the guard and a laser sensor. The laser bean transmitter and receiver is used to determine the relative position and orientation of the pallet.

Its basic performance is as follow: the forklift beans a red line through an integrated laser sensor. When there is a certain distance to the pallet, as the forklift is approaching to it, the projected line is shifted, what indicates that the forklift is facing a certain pallet. And from this layout the forklift can determine the relative position of the pallet.

In the following chapters, a more detailed explanation about its functioning will be given.

1.2.3 Alternative automated solutions

Looking for similar automated solutions in the market, different alternatives can be found.

Jungheinrich is an international company based in Hamburg, Germany, dedicated to material handling equipment, warehousing and material flow engineering sectors. This German company offers a wide range of automated guided vehicles (AGVs), storage and conveyor systems. Referred to forklifts, there are three main different models with different capabilities each of them:

AGV EKS 210a[9]

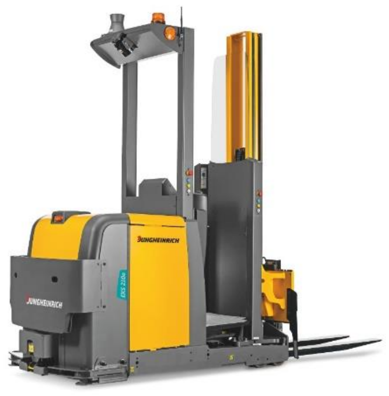


Figure 6: *AGV EKS 210a*

The EKS 210a model ensures the highest possible level of reliability and safety. It can be used as an entirely automatic system, but also in mixed operation with manual trucks and persons.

Using a laser navigation system and controlled by the transport manager, the truck moves independently using reflectors within the logistic centre. The EKS 210a uses reflectors to orient itself automatically within the warehouse. The truck can be integrated into the existing structures without floor or conversion work. Consequently, flexible changes in the travel route can be easily and rapidly executed at any time.

[9] Consulted in *https://www.jungheinrich.ie/products/automated-solutions-agv/automated-guided-vehicles-agv/agv-eks-210a/*

It uses a lithium-ion technology regarding the battery which ensures lasting performance, fast charging times and zero maintenance.

Basic characteristics:

Lift (Standard mast)	From 800 to 3000 mm
Capacity/load	Up to 1,5 ton

AGV ERC 215a[10]

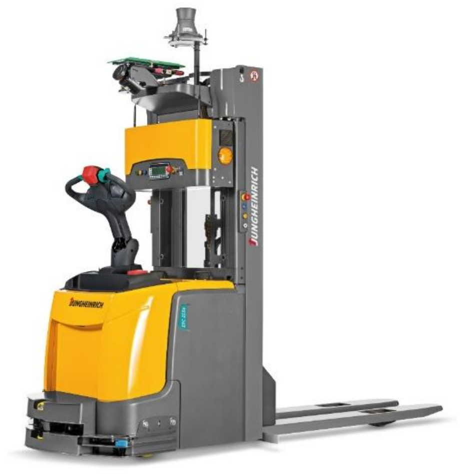


Figure 7*: AGV ERC 215a*

The ERC215a model is an AGV that combines advanced engineering with precision navigation technology and safety components, thus ensuring a high level of reliability and safety. It can be used in mixed operations mode with manual trucks and pedestrians.

Navigation is via laser sensor, so no floor work is required. To do this, reflectors are attached to suitable objects along the travel route such as walls, columns or natural landmarks.

The basis of the ERC 215a is formed by an electric pedestrian pallet truck, a tried and tested standard truck combined with appropriate safety technology as well as automation and navigation components.

[10] Consulted in https://www.jungheinrich.ie/products/automated-solutions-agv/automated-guided-vehicles-agv/agv-erc-215a/

In addition, the lithium-ion technologies used in the batteries guarantee a high degree of availability thanks to extremely short charging times.

Basic characteristics:

Lift (Standard mast)	From 3100 to 4000 mm
Capacity/load	From 1,3 ton to 1,5 ton

AGV ERE 225a[11]

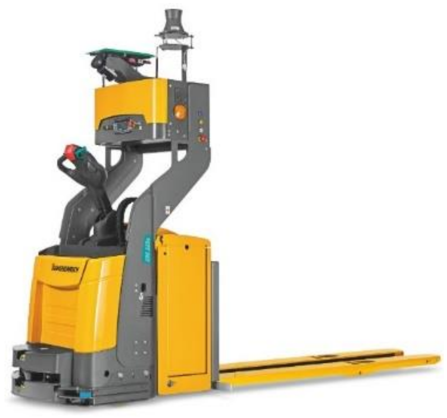


Figure 8: *AGV ERE 225a*

The ERE 225a can be used in mixed operations mode with manual trucks and pedestrians. Its high capacity combined with the variable fork lengths enables this model to be used for the transport of multiple pallets at floor lever.

To navigate the warehouse, reflectors are attached to suitable objects along the travel route such as walls, columns or natural landmarks.

The basis of the ERC 215a is formed by an electric pedestrian pallet truck, a tried and tested standard truck combined with appropriate safety technology as well as automation and navigation components. With a maximum capacity of 2.5 ton, the transport of heavy

[11] Consulted in *https://www.jungheinrich.ie/products/automated-solutions-agv/automated-guided-vehicles-agv/agv-ere-225a/*

loads is possible at floor level. It is also possible to equip the ERE 225a with long forks for the transport of multiple pallets.

It uses a lithium-ion technology regarding the battery which ensures lasting performance, fast charging times and zero maintenance.

Basic characteristics:

Lift (Standard mast)	Up to 122 mm
Capacity/load	Up to 2,5 ton

1.3 Current sensors used in AGVs

The range of available technology for automatizing vehicles is enormous regarding the track guidance technology used; below are shown the main sensors used for the automation:[12]

1. Guide Wire:

Guide wires technology have proved to be reliable and good value in many long-term applications. Track guidance systems allow vehicles without driver to move along a current carrying conductor.

Inductive systems typically consist of a frequency generator, a steering antenna and the guide wire in the ground. Reference frequency and current as well as lateral separation and height from guide conductor can be selected from a large scope of variants.

This inductive method has the advantage that it is not sensitive to dirty so their use has gained acceptance in industrial plants.

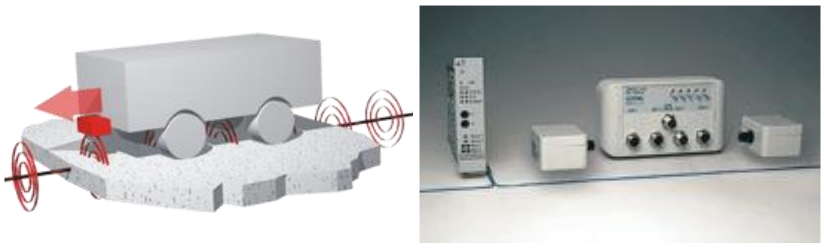


Figure 9: *Guided Wire sensor*

[12] Consulted in *https://www.goetting-agv.com/*

2. <u>Optical track guidance along lines:</u>

This technology basically consists on applying lines to the roadway to guide the vehicles. Track recognition is carried out with modern cameras and image processing systems.

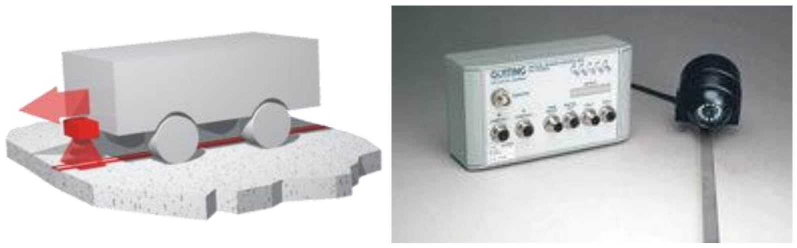


***Figure 10**: Optical sensors*

3. <u>Ground markers: transponders:</u>

Using the transponders, either specific positions are determined along a line or track guided system, or virtual tracks are defined. The transponders are activated by the transponder reader/antenna, transmit their identification signal and their position is then detected with an accuracy of millimetres or centimetres. So, the vehicle can identify the position lengthways and/or crossways.

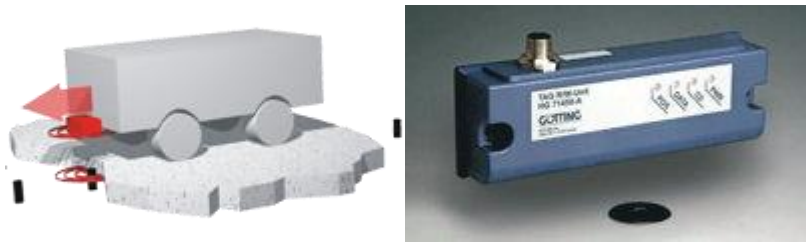


***Figure 11:** Transponder sensor*

4. <u>Laser Scanner:</u>

With laser scanners the vehicle can identify its position and the direction it is going in with accuracy making use of reflection markers at the edge of the roadway, e.g. on the walls. This technology is used if a high level of flexibility in track guidance is required. In connection with sensors for obstacle detection (ultrasonic or optical systems), obstacles, when emerging, can be avoided and the vehicle is guided to its destination using alternative routes.

With this technology, in the AGV’s memory it is recorded the map of the reflector layout. Comparing it with the information obtained by the laser transmitter and receiver, allows the navigation system to triangulate the current position of the AGV. Direction is adjusted in order to keep the AGV on track by using the constantly updating position.

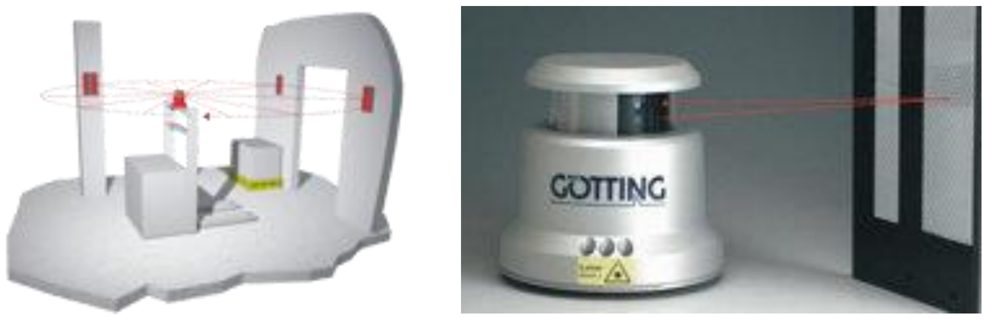


***Figure 12**: Laser sensor*

1.4 Typical application of vision systems in vehicles

As an alternative to only using laser scanners, Auto Guided Vehicles can also offer camera- guided processes, improving the pallet position determination procedures. This last option is the one chosen for the studied forklift of the Continental Company and will be explained in detail in the following paragraphs.

Before, sensors used in the different Auto Guided Vehicles with the aim of detecting several variables of the environment were pretty simple. However, all that changed, when it was realised that replacing human sensory input (eyes) and image processing (brain) by using cyber physical systems was a possible task.[13]

Research is on to develop a vision sensor as good as the human eye so it should reunite most of the following features:

- Must provide *360°* horizontal coverage around the vehicle
- Must resolve objects in *3D* very close to, and far from, the vehicle
- Must resolve/identify multiple static/moving objects up to maximum range
- Must do all the above in all lighting and weather conditions
- Finally, provide all this data in ‘real-time’.

As answers to this request, science has come up with different sensors to replace the human eye in Autonomous Guided Vehicles: digital video cameras, lidar (Light Detection and Ranging) scanners or simply radars.

[13] Consulted in *https://www.rs-online.com/designspark/lidar-radar-digital-cameras-the-eyes-of-autonomous-vehicles*

Digital Video Cameras are an obvious candidate as an artificial eye which in many ways offer superior performance to their natural counterpart as they are able to maintain high-resolution in pixels and colour across the full width of its field of view; they also maintain constant ‘frame-rate’ across the field of view and there is the possibility of using two cameras and providing stereoscopic 3D vision.

In addition, it is remarkable that digital cameras maintain performance over time since these objects do not suffer from hay-fever or macular degeneration.

However, despite all these advantages of digital cameras over the human eye, it is not enough with the camera itself. Digital video cameras require incredibly powerful computers to turn their output signals into a 3D map of the world around the car with static objects identified, and moving ones identified and tracked in real-time.

The trivial conclusion is that digital video cameras, lidar, radar and even ultrasonic (Sonar) will all be needed to achieve an acceptable safe and precise level of autodriving, at least for the short term.

2. Concepting of a camera-based image processing function

As said before, automating the proposed forklift from the Continental Company is the major aim of this project. Getting rid of the driver will improve the forklift performance and will increase the factory's productivity, since all possible human errors will be removed.

In order to achieve the desired level of automation a laser sensor has been implemented to the forklift. As shown in the pictures below, it is located on the top of the overhead guard pointing to the front of the AGV.

Specifically, the one which has been used for this application is the *Laser SICK NAV 350*[14], a 2D laser scanner.

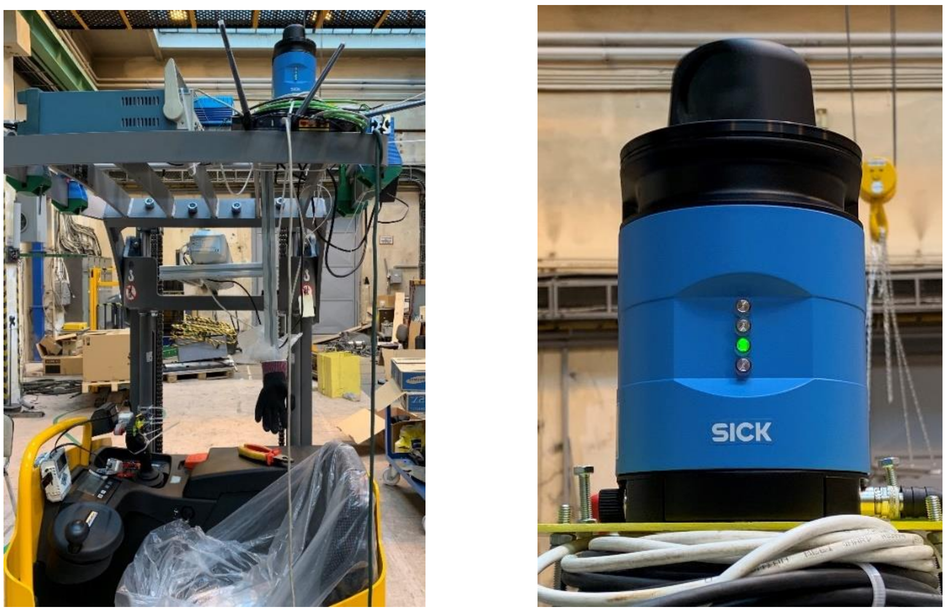


***Figure 13**: Laser SICK NAV 350*

This kind of sensors are suitable for performing detection and ranging tasks on surfaces. Regardless of the angle of installation, SICK 2D LiDAR sensors operate with consistent reliability and accuracy and can be used both indoors and outdoors. For navigation, detection, or measurement: 2D LiDAR sensors supply reliable measurement data for a whole host of tasks.

[14] Consulted in *https://www.sick.com/ag/en/detection-and-ranging-solutions/2d-lidar-sensors/c/g91900*

The NAV operates on a principle like that of optical radar. With its all-round view, it detects reflector marks within its operating area and very accurately measures their distance and direction.

The laser sensor uses these values to calculate its own absolute position and reports it to the on-board computer. This informs the AGV about the new storage location and thus defines the route to be taken.

The model chosen, NAV350-3232 presents the following main features which can be found in its data sheet:

Application	Indoor
Light source	905 nm
Aperture angle (horizontal)	360º
Scanning frequency	8 Hz
Angular resolution	0.25º
Working range	0.5 m – 250 m
Scanning range	35 m – 100 m

Thanks to this laser sensor a 3D map of the forklift surroundings is created and the global position of the pallet with respect to the camera is known.

However, the pallet position obtained with the laser sensor is only an approximation; that is the reason why this sensor needs to be supplemented with the digital video camera in order to have an accurate recognition of the actual position of the pallet.

The camera supplier chosen for this application is Hitachi, a Japanese multinational conglomerate company headquartered in Chiyoda, Tokyo, Japan.

In the picture below it is presented the camera used in this case:

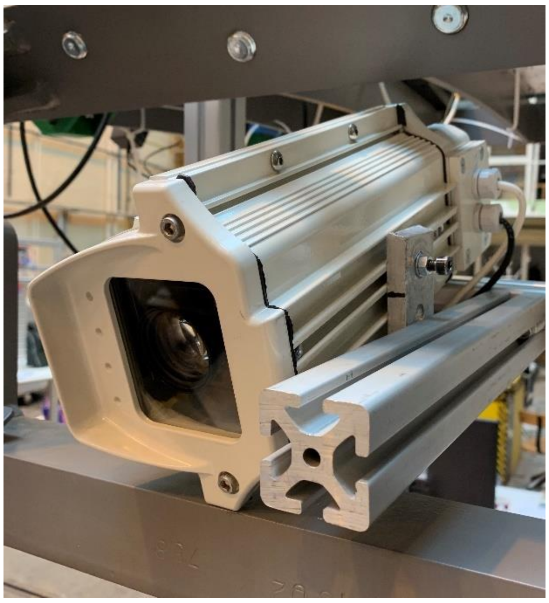


Figure 14*: Digital camera VK-S654R/S654ER*

The camera model used is *VK-S654R/S654ER*[15]. It can be installed in cars and other means of transport, and by automatically reduce blur, image jump to get clear and stable images. Also, through progressive video frames it eliminates grainy images and annoying lines.

In the following figures it is shown how the camera is mounted into the forklift and which is its range of action, both in the front view and the top view.

[15] Consulted in *http://www.accessories-shops.com/hitachi-camera-series/hitachi-vk-s654er-35x-integration-color-sony-ccd-camera.html*

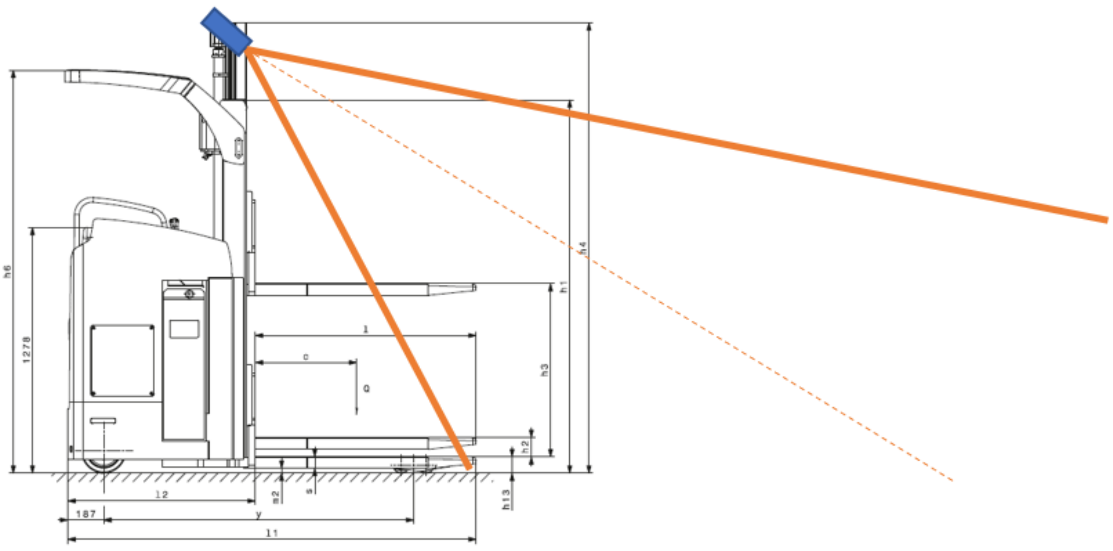


***Figure 15**: Camera mounted into the forklift. Front view*

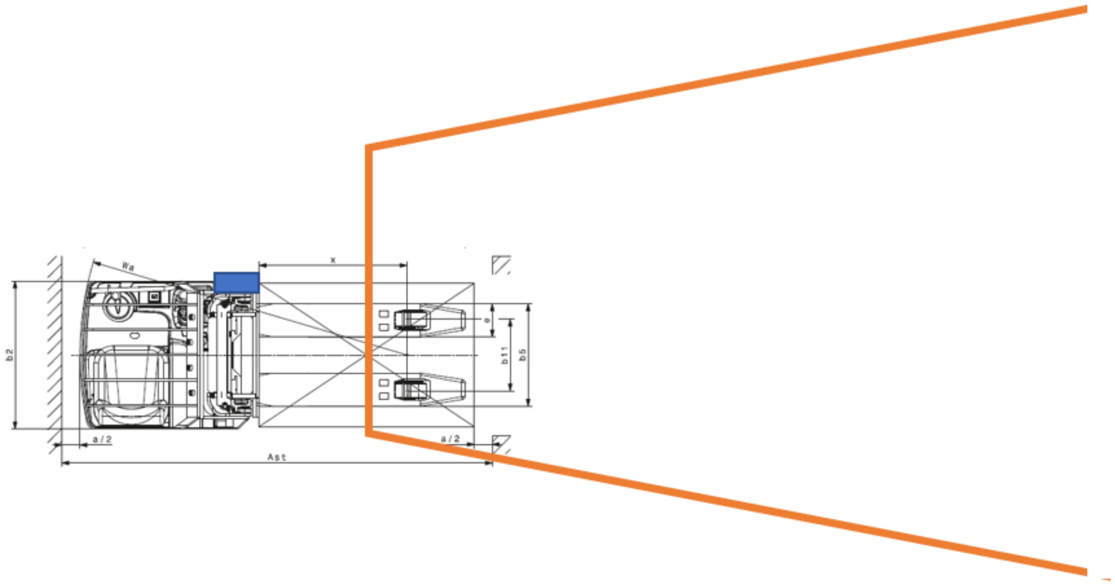


***Figure 16:** Camera mounted into the forklift. Top view*

Nevertheless, and as said before it is not enough with the camera itself to completely automatize the desired vehicle. Powerful computers are needed to turn the camera output signal into a 3D map of the pallet surroundings.

In this project, an industrial PC known as the Blue Tank PC is used. It runs two different operating systems: Windows and Linux.

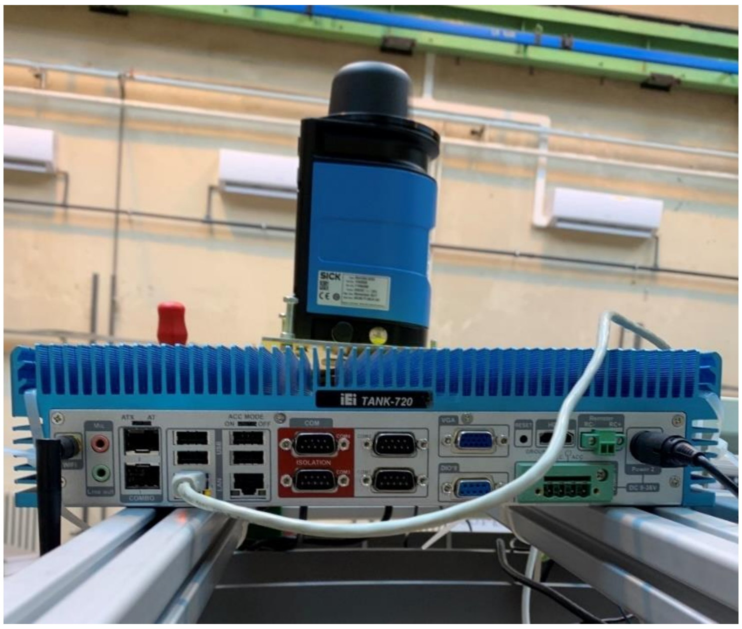


***Figure 17**: Blue Tank PC*

With the first one the software to control the camera is operated. The used software, named VCamControl allows the computer to obtain the pictures taken by the digital camera which is installed into the forklift. On the other hand, the Linux operating system is the one in charge of the programming tasks executed in order to detect the pallet in the picture and of giving the different instructions to the vehicle.

These operations are made in the environment of the software Visual Studio. Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio supports 36 different programming languages and allows the code editor and debugger to support nearly any programming language, provided a language-specific service exists. The built-in language used to program the forklift is C++.

The different source codes implemented in visual studio will be obtained from OpenCV. Open source Computer Vision is a library of programming functions mainly aimed at real-time computer vision. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface.

In addition, it runs on the following desktop operating systems: Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD.

3. Implementing the necessary functionality

Combining laser sensor, digital video camera and the Blue Tank PC, the forklift automatization is completely done. Thanks to these devices, the relative position of the pallet with respect the forklift is known at every moment and so, the vehicle can manage it in order to get through the pallet and move it all along the warehouse or logistic centre.

Basically, laser sensor obtains the global position of the pallet; then, with the aid of the implemented camera, the image coordinates of the two pallet holes are also known. In order to get the final coordinates of the two holes so that the forklift forks can be introduced into them, there are several necessary intermediate steps that are going to be explained in the following paragraphs:

3.1 Camera calibration

First, it is necessary the correct calibration of the digital video camera. Calibration consists in comparing the values obtained with a measuring instrument with the corresponding measure of a reference pattern.

Images obtained using a camera present a significant distortion. With the correct calibration and some remapping, we can correct this. In addition, calibration allows to determine the relation between the cameras natural units (pixels) and the real-world units (millimetres, for example).

For the distortion OpenCV considers the radial and tangential factors.

For the radial factor it uses the following formulas:

$$x_{corrected} = x\,(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$y_{corrected} = y\,(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

So, for an old pixel point at (x, y) coordinates in the input image, its position on the corrected output image will be $(x_{corrected,} y_{corrected})$.

The presence of the radial distortion manifests in form of the “barrel” or “fish-eye” effect.

On the other hand, tangential distortion occurs because the image taking lenses are not perfectly parallel to the imaging plane. It can be corrected via the formulas:

$$x_{corrected} = x\,(2p_1 xy + p_2(r^2 + 2x^2))$$

$$y_{corrected} = y\,(2p_2 xy + p_2(r^2 + 2y^2))$$

So, we have five distortion parameters which in OpenCV are presented as one row matrix with 5 columns:

$$Distortion_{coefficients} = (k_1\; k_2\; p_1\; p_2\; k_3)$$

Now for the unit conversion we use the following formula:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

The unknown parameters are f_x and f_y (camera focal lengths) and c_x, c_y which are the optical centres expressed in pixels coordinates.

The matrix containing these four parameters is referred to as the *camera matrix*. While the distortion coefficients are the same regardless of the camera resolution used, these should be scaled along with the current resolution from the calibrated resolution. The process of determining these two matrices is the calibration.

Calculation of these parameters is done through basic geometrical equations. The equations used depend on the chosen calibrating objects. Currently OpenCV supports three types of objects for calibration: classical black-white chessboard, symmetrical circle pattern and asymmetrical circle pattern

Basically, snapshots of these patterns must be taken with the camera and let OpenCV find them. Each found pattern results in a new equation. To solve the equation, at least a predetermined number of pattern snapshots are needed to form a well-posed equation system. This number is higher for the chessboard pattern and less for the circle ones. For example, in theory the chessboard pattern requires at least two snapshots. However, in practice, for good results they will be needed at least 10 good snapshots of the input pattern in different positions.

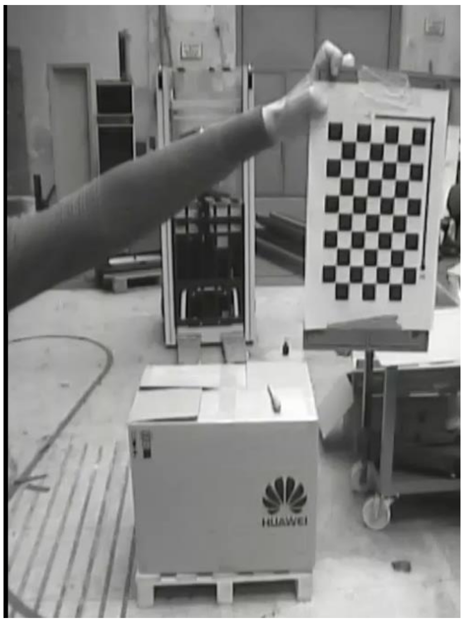


Figure 18*: Snapshot sample for chessboard calibration*

Below, it can be found the source code in C++ used in order to obtain the camera matrix unknown parameters[16]. In this case, chessboard calibration has been used:

1. Read the settings:

```
Settings s;
const string inputSettingsFile = argc > 1 ? argv[1] : "default.xml";
FileStorage  fs(inputSettingsFile,  FileStorage::READ);  //  Read  the
settings
if (!fs.isOpened())
{
      cout  <<  "Could  not  open  the  configuration  file:  \""  <<
inputSettingsFile << "\"" << endl;
      return -1;
}
fs["Settings"] >> s;
fs.release();                                          // close Settings
file

if (!s.goodInput)
{
      cout << "Invalid input detected. Application stopping. " << endl;
      return -1;
}
```

2. Get next input, if it fails or we have enough of them – calibrate:

Now there is a big loop where the following operations are done:

- Get the next image from the image list, camera or video file.

- If this fails or we have enough images then we run the calibration process. In case of image we step out of the loop and otherwise the remaining frames will be undistorted (if the option is set) via changing from *DETECTION* mode to the *CALIBRATED*one.

```
for(int i = 0;;++i)
{
  Mat view;
  bool blinkOutput = false;

  view = s.nextImage();

  //-----  If no more image, or got enough, then stop calibration and
show result -------------
  if( mode == CAPTURING && imagePoints.size() >= (unsigned)s.nrFrames )
  {
        if(   runCalibrationAndSave(s,   imageSize,     cameraMatrix,
distCoeffs, imagePoints))
```

[16] Consulted in *https://docs.opencv.org/2.4.13.7/doc/tutorials/calib3d/camera_calibration/camera_calibration.html*

```
              mode = CALIBRATED;
        else
              mode = DETECTION;
  }
  if(view.empty())           // If no more images then run calibration,
save and stop loop.
  {
            if( imagePoints.size() > 0 )
                  runCalibrationAndSave(s,  imageSize,    cameraMatrix,
distCoeffs, imagePoints);
            break;
  imageSize = view.size();  // Format input image.
  if( s.flipVertical )     flip( view, view, 0 );
```

3. Find the pattern in the current input:

The formation of the equations mentioned above aims to finding major patterns in the input: in case of the chessboard this are corners of the squares. The position of these will form the result which will be written into the *pointBuf* vector.

```
vector<Point2f> pointBuf;

bool found;
switch( s.calibrationPattern ) // Find feature points on the input format
{
case Settings::CHESSBOARD:
  found = findChessboardCorners( view, s.boardSize, pointBuf,
  CV_CALIB_CB_ADAPTIVE_THRESH       |       CV_CALIB_CB_FAST_CHECK       |
CV_CALIB_CB_NORMALIZE_IMAGE);
  break;
case Settings::CIRCLES_GRID:
  found = findCirclesGrid( view, s.boardSize, pointBuf );
  break;
case Settings::ASYMMETRIC_CIRCLES_GRID:
  found     =     findCirclesGrid(    view,     s.boardSize,    pointBuf,
CALIB_CB_ASYMMETRIC_GRID );
  break;
}
```

Both the current image and the size of the board are passed, and the positions of the patterns is known. Furthermore, they return a Boolean variable which states if the pattern was found in the input (we only need to consider those images where this is true!).

Then again in case of cameras we only take camera images when an input delay time is passed. This is done in order to allow user moving the chessboard around and getting different images. Similar images result in similar equations, and similar equations at the calibration step will form an ill-posed problem, so the calibration will fail. For square images the positions of the corners are only approximate.

This may be improved by calling the cornerSubPix function. It will produce better calibration result. After this a valid input result is added to the *imagePoints* vector to collect all the equations into a single container. Finally, for visualization feedback purposes the found points on the input image are drawn using findChessboardCorners function.

```
if ( found)                    // If done with success,
  {
      // improve the found corners' coordinate accuracy for chessboard
        if( s.calibrationPattern == Settings::CHESSBOARD)
        {
            Mat viewGray;
            cvtColor(view, viewGray, CV_BGR2GRAY);
            cornerSubPix( viewGray, pointBuf, Size(11,11),
              Size(-1,-1),                               TermCriteria(
CV_TERMCRIT_EPS+CV_TERMCRIT_ITER, 30, 0.1 ));
        }

        if( mode == CAPTURING &&  // For camera only take new samples
after delay time
            (!s.inputCapture.isOpened() || clock() - prevTimestamp >
s.delay*1e-3*CLOCKS_PER_SEC) )
        {
            imagePoints.push_back(pointBuf);
            prevTimestamp = clock();
            blinkOutput = s.inputCapture.isOpened();
        }

        // Draw the corners.
        drawChessboardCorners( view, s.boardSize, Mat(pointBuf), found
);
  }
```

4. Show state and result to the user, plus command line control of the application:

```
//------------------------------ Output Text --------------------
----------------------------
string msg = (mode == CAPTURING) ? "100/100" :
          mode == CALIBRATED ? "Calibrated" : "Press 'g' to
start";
int baseLine = 0;
Size textSize = getTextSize(msg, 1, 1, 1, &baseLine);
Point textOrigin(view.cols - 2*textSize.width - 10, view.rows -
2*baseLine - 10);

if( mode == CAPTURING )
{
  if(s.showUndistorsed)
```

```
        msg  =  format(  "%d/%d  Undist",  (int)imagePoints.size(),
    s.nrFrames );
    else
        msg = format( "%d/%d",  (int)imagePoints.size(),  s.nrFrames
    );
 }

 putText( view, msg, textOrigin, 1, 1, mode == CALIBRATED ?  GREEN
    : RED);

 if( blinkOutput )
       bitwise_not(view, view);
```

In addition, the image may be corrected using undistort function:

```
//------------------------- Video capture  output  undistorted --------
----------------------
if( mode == CALIBRATED && s.showUndistorsed )
{
  Mat temp = view.clone();
  undistort(temp, view, cameraMatrix, distCoeffs);
}
//------------------------------ Show image and check for input commands
-------------------
imshow("Image View", view);
```

Then an input key is given and if this is *u* the distortion removal is changed, if it is *g* detection process is started again, and finally for the *ESC*key the application is quitted:

```
char key =  waitKey(s.inputCapture.isOpened() ? 50 : s.delay);
if( key  == ESC_KEY )
      break;

if( key == 'u' && mode == CALIBRATED )
   s.showUndistorsed = !s.showUndistorsed;

if( s.inputCapture.isOpened() && key == 'g' )
{
  mode = CAPTURING;
  imagePoints.clear();
}
```

Because the calibration needs to be done only once per camera, it makes sense to save it after a successful calibration. Due to this first, calibration is made and if it succeeds, result is saved into an OpenCV style XML or YAML file.

```
bool runCalibrationAndSave(Settings& s, Size imageSize, Mat&
cameraMatrix, Mat& distCoeffs,vector<vector<Point2f> > imagePoints )
{
 vector<Mat> rvecs, tvecs;
 vector<float> reprojErrs;
 double totalAvgErr = 0;

 bool ok = runCalibration(s,imageSize, cameraMatrix, distCoeffs,
imagePoints, rvecs, tvecs,
                            reprojErrs, totalAvgErr);
 cout << (ok ? "Calibration succeeded" : "Calibration failed")
     << ". avg re projection error = "  << totalAvgErr ;

 if( ok )   // save only if the calibration was done with success
     saveCameraParams( s, imageSize, cameraMatrix, distCoeffs, rvecs
,tvecs, reprojErrs,
                           imagePoints, totalAvgErr);
 return ok;
}
```

Calibration is done with the help of the calibrateCamera function. It has the following parameters:

- The object points. This is a vector of *Point3* vector that is a collection of the points where these important points are present. Corner points are calculated with the *calcBoardCornerPositions* function as:

```
▪ void calcBoardCornerPositions(Size boardSize, float squareSize,
  vector<Point3f>& corners,
▪                   Settings::Pattern       patternType        /*=
  Settings::CHESSBOARD*/)
▪ {
▪ corners.clear();
▪
▪ switch(patternType)
▪ {
▪ case Settings::CHESSBOARD:
▪ case Settings::CIRCLES_GRID:
▪   for( int i = 0; i < boardSize.height; ++i )
▪     for( int j = 0; j < boardSize.width; ++j )
▪         corners.push_back(Point3f(float( j*squareSize ), float(
  i*squareSize ), 0));
▪   break;
▪
▪ case Settings::ASYMMETRIC_CIRCLES_GRID:
▪   for( int i = 0; i < boardSize.height; i++ )
▪      for( int j = 0; j < boardSize.width; j++ )
▪         corners.push_back(Point3f(float((2*j      +      i      %
  2)*squareSize), float(i*squareSize), 0));
▪   break;
▪ }
▪ }
```

And then multiply it as:

```
vector<vector<Point3f> > objectPoints(1);
calcBoardCornerPositions(s.boardSize,  s.squareSize,  objectPoints[0],
s.calibrationPattern);
objectPoints.resize(imagePoints.size(),objectPoints[0]);
```

- The image points. This is a vector of *Point2f* vector which for each input image contains coordinates of the corners for chessboard.
- The size of the image acquired from the camera, video file or the images.
- The camera matrix.

```
▪   cameraMatrix = Mat::eye(3, 3, CV_64F);
▪   if( s.flag & CV_CALIB_FIX_ASPECT_RATIO )
▪       cameraMatrix.at<double>(0,0) = 1.0;
```

- The distortion coefficient matrix. Initialize with zero.

```
distCoeffs = Mat::zeros(8, 1, CV_64F);
```

Finally, the function returns the average re-projection error. This number gives a good estimation of precision of the found parameters. This should be as close to zero as possible. The error is calculated by using the projectPoints to first transform the object point to image point. Then the absolute norm between the patterns obtained with the transformation and the corner finding algorithm is calculated. To find the average error the arithmetical mean of the errors calculated for all the calibration images is calculated.

```
double  computeReprojectionErrors(  const  vector<vector<Point3f>  >&
objectPoints,
                          const vector<vector<Point2f> >& imagePoints,
                          const vector<Mat>& rvecs, const vector<Mat>&
tvecs,
                          const  Mat&  cameraMatrix  ,  const  Mat&
distCoeffs,
                          vector<float>& perViewErrors)
{
vector<Point2f> imagePoints2;
int i, totalPoints = 0;
double totalErr = 0, err;
perViewErrors.resize(objectPoints.size());

for( i = 0; i < (int)objectPoints.size(); ++i )
{
  projectPoints(     Mat(objectPoints[i]),     rvecs[i],      tvecs[i],
cameraMatrix,  // project
                                              distCoeffs, imagePoints2);
  err    =    norm(Mat(imagePoints[i]),    Mat(imagePoints2),    CV_L2);
// difference

  int n = (int)objectPoints[i].size();
  perViewErrors[i] = (float) std::sqrt(err*err/n);                    //
save for this view
  totalErr       += err*err;                                          //
sum it up
  totalPoints     += n;
}

return std::sqrt(totalErr/totalPoints);                  // calculate the
arithmetical mean
}
```

3.2 Blob detector function

Once the camera is correctly calibrated it is possible to obtain the real coordinates of the two pallet holes x, y from its image coordinates u, v taken out from the pictures captured with the digital camera.

The software implemented in the Blue Tank PC detects these holes and capture their coordinates with the aid of the function 'blob detector'. The source code for this function has been obtain from OpenCV library and it is written using C++ language.[17]

This function detects the blobs present in a certain image and returns their coordinates in pixels unit system. Below it is presented this function more in detail.

A blob is a group of connected pixels in an image that share some common property (E.g. grayscale value).

OpenCV provides a convenient way to detect blobs and filter them based on different characteristics. SimpleBlobDetector, as the name implies, is based on a rather simple algorithm described below. The algorithm has the following steps.

1. Thresholding: Convert the source images to several binary images by thresholding the source image with thresholds starting at *minThreshold*. These thresholds are incremented by *thresholdStep* until *maxThreshold*. So, the first threshold is minThreshold, the second is *minThreshold* + *thresholdStep*, the third is *minThreshold + 2 x thresholdStep*, and so on.

2. Grouping: In each binary image, connected white pixels are grouped together.

3. Merging: The centres of the binary blobs in the binary images are computed, and blobs located closer than *minDistBetweenBlobs* are merged.

4. Centre & Radius Calculation: The centres and radii of the new merged blobs are computed and returned.

[17] Consulted in: *https://www.learnopencv.com/blob-detection-using-opencv-python-c/*

The parameters for SimpleBlobDetector can be set to filter the type of blobs we want. This filtering can be made by colour or size:

1. <u>By Colour</u>: First it is needed to set *filterByColor* = 1. Set *blobColor* = 0 to select darker blobs, and *blobColor* = 255 for lighter blobs.

2. <u>By Size</u>: It is possible to filter the blobs based on size by setting the parameters *filterByArea* = 1, and appropriate values for *minArea* and *maxArea*. E.g. setting *minArea* = 100 will filter out all the blobs that have less than 100 pixels.

Below, it can be found the source code in C++ where the blob detector function has been implemented

```
// ConsoleApplication1.cpp : This file contains the 'main' function.
Program execution begins and ends there.
//

#include "pch.h"
#include <iostream>
#include <opencv2/core/core.hpp>
#include <stdio.h>
#include <string.h>
#include "Windows.h"
                       //SDK include
#include "PlayM4.h"
                               // SDK include
#include <opencv2/opencv.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <atlstr.h>
                               //Nincs benne az SDK-ban, de kell a
CStringhez
// #include <afx.h>
                               //Ez csak bezvar, de kellhet
#include <atltrace.h>
#include <chrono>    //sleep-hez kell elvileg, de semmit nem csinál
#include <thread>
#include <limits>

using namespace std;
using namespace cv;


void main()
{

        // Read image
        Mat im = imread("blob.jpg", IMREAD_GRAYSCALE);
        Mat output;
        //edge det
        //Canny(im, output, 50, 150, 3);
        //convert the image
        inRange(im, Scalar(10, 10, 10), Scalar(90, 90, 90), output);
        bitwise_not(output,output);
        // Setup SimpleBlobDetector parameters.
        SimpleBlobDetector::Params params;

        // Change thresholds
        params.minThreshold = 10;
        params.maxThreshold = 255;

        // Filter by Area.
        params.filterByArea = true;
        params.minArea = 1000;
        params.maxArea = 10000;

        // Filter by Circularity
        params.filterByCircularity = false;
        params.minCircularity = 0.1;

        // Filter by Convexity
        params.filterByConvexity = false;
        params.minConvexity = 0.67;

        // Filter by Inertia
        params.filterByInertia = false;
        params.minInertiaRatio = 0.01;


        // Storage for blobs
        vector<KeyPoint> keypoints;


#if CV_MAJOR_VERSION < 3   // If you are using OpenCV 2

        // Set up detector with params
        SimpleBlobDetector detector(params);

        // Detect blobs
        detector.detect(im, keypoints);
#else

        // Set up detector with params
        Ptr<SimpleBlobDetector>               detector                 =
SimpleBlobDetector::create(params);

        // Detect blobs
        detector->detect(output, keypoints);
#endif

        // Draw detected blobs as red circles.
        // DrawMatchesFlags::DRAW_RICH_KEYPOINTS flag ensures
        // the size of the circle corresponds to the size of blob

        Mat im_with_keypoints;
        drawKeypoints(output, keypoints, im_with_keypoints, Scalar(0, 0,
255), DrawMatchesFlags::DRAW_RICH_KEYPOINTS);

        // Show blobs
        imshow("keypoints", im_with_keypoints);
        waitKey(0);
```

Image is read from the camera in a grayscale and then filtered by color with inRange function in order to clearly identify the two blobs corresponding to the two pallet holes.

***Figure 19**: Image taken with the camera*

After using bitwise_not function with the aim of inverting the black and white colours, the image is ready for blob detector. By making use of the function the two blobs are clearly identified as shown in the following image:

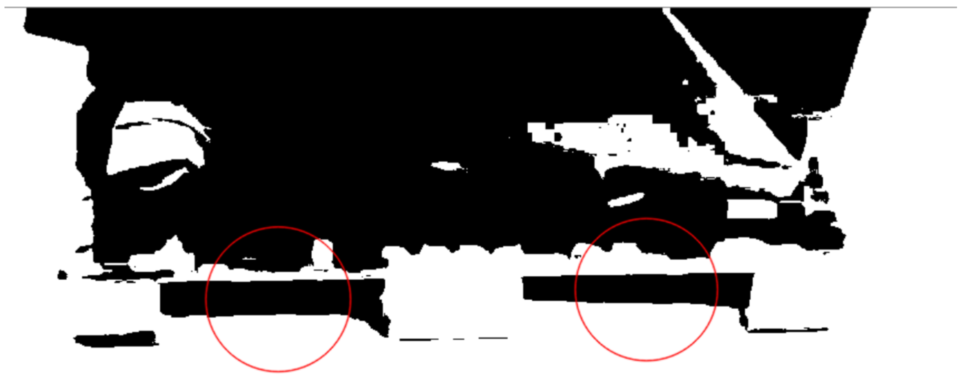


***Figure 20**: Pallet holes detected using blob detector*

3.3 Determination of the vehicle angle and distance from the real blob coordinates

With the implementation of blob detector function, coordinates of the two pallet holes are known parameters. They are given in the image coordinate systems whose unit system is in pixels, $u_{\ i}(pixels), v_{\ i}(pixels)$.

It is necessary to translate these coordinates to the camera coordinate system in millimetres $x_i'(mm), y_i'\ (mm)$ and finally to the reality coordinate system which is also expressed in millimetres $x_i\ (mm),\ y_i\ (mm)$.

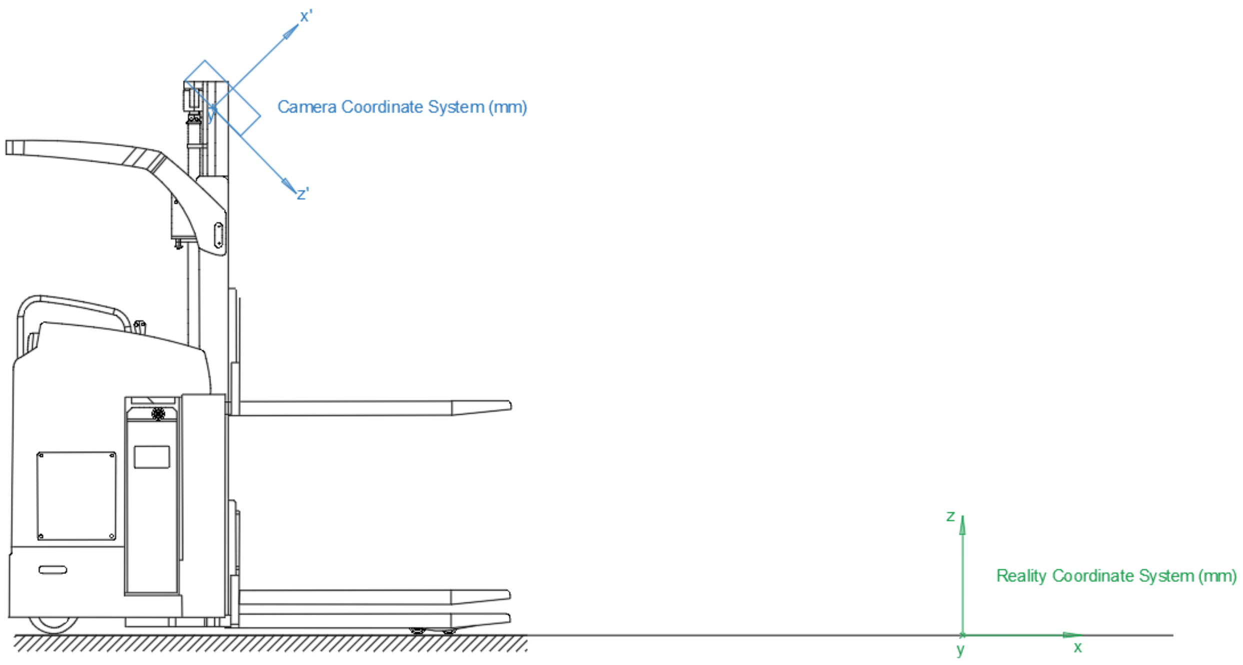


***Figure 21**: Camera coordinate system and Reality coordinate system*

From the holes coordinates it would be possible to determine the pallet distance and its relative angle with respect to the forklift. Whit this information, Blue Tank PC will know which are the necessary instructions for the forklift in order to reach the pallet and transport it all along the warehouse or logistic centre.

3.3.1 Transformation from camera coordinate system to reality coordinate system

Initial data are the blob coordinates $u_{\,i}(pixels), v_{\,i}(pixels)$expressed in the camera coordinate system.

From these coordinates the blob coordinates in millimetres expressed in the camera coordinate system $x'_i(mm), y'_i\,(mm), z'_i\,(mm)$ are deduced by solving tedious equations making use of the Matlab software.

After camera calibration applying the chessboard method, the parameters from the camera matrix are known: f_x , f_y, c_x and c_y.

If these parameters are known, the following equations (derived from the transformation of an internal matrix) can be written:

$$\frac{f_x \times x_1}{z_1} + \frac{0 \times y_1}{z_1} + c_x = u_1$$
$$\frac{0 \times x_1}{z_1} + \frac{f_y \times y_1}{z_1} + c_y = v_1$$
$$\frac{f_x \times x_2}{z_2} + \frac{0 \times y_2}{z_2} + c_x = u_2$$
$$\frac{0 \times x_2}{z_2} + \frac{f_y \times y_2}{z_2} + c_y = v_2$$

Where $x_i(mm), y_i\,(mm), z_i\,(mm)$ are the true coordinates of the given point in the reality coordinate system.

This is so far 4 equations but 6 unknowns (x_1, y_1, z_1, x_2, y_2, z_2 x_3, y_3, z_3) so the system cannot be solved yet.

Coming back to reality, distance between the centre of the two holes of the pallet, whose coordinates are known both in the camera coordination system and in the global coordinate system, is a known parameter. From it, another equation can be added to the system:

$$((x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2)^{\frac{1}{2}} = d_{12}$$

Where d_{12} is the known distance. So, this is already a system of 5 equations and 6 unknowns.

We could solve it by 2 points because we know that the height of these two points (in the global coordinate system) is the same, we can use it for a sixth equation:

$$z_1 \times \sin\alpha + x_1 \times \cos\alpha = z_2 \times \sin\alpha + x_2 \times \cos\alpha$$

Notice how in this last equation the parameter α,which represents the inclination of the camera with respect to the floor, is introduced. Thanks to this parameter, transformation between camera coordinate system in millimetres $x_i{}'(mm), y_i{}'\ (mm)$ and reality coordinate system $x_i\ (mm),\ y_i\ (mm)$ is possible.

Now, this system of 6 equations and 6 unknowns can be solved.

3.3.2 Obtaining the relative angle and distance from the forklift

From now on, (x_1, y_1, z_1, x_2, y_2, z_2 x_3, y_3, z_3) are known variables. The next step is to determine the rotation angle of the pallet respect to the forklift, β, and the distance between the pallet and the forklift, d.

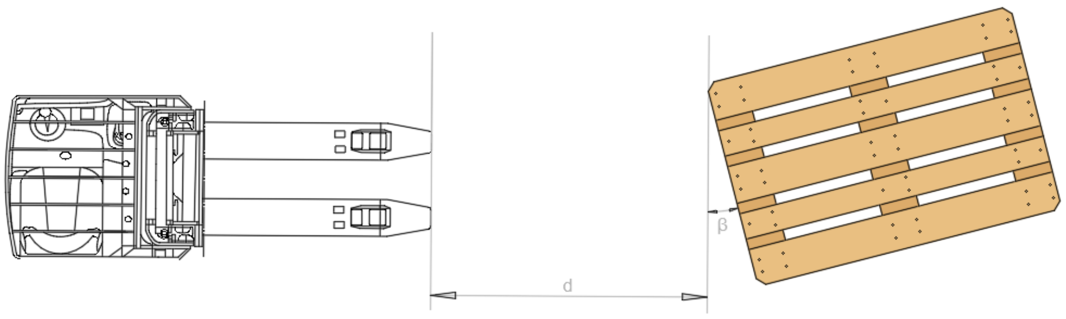


Figure 22*: Angle of the pallet respect to the forklift, β, and distance between the pallet and the forklift, d*

Since the coordinates of the two detected blobs are known, the distance between the pallet and these two holes can also be calculated as follows:

$$tx_1 = z_1 \times cos\,\alpha - y_1 \times sin\,\alpha$$

$$tx_2 = z_2 \times cos\,\alpha - y_2 \times sin\,\alpha$$

Notice how the distance in the y axes is not calculated since it remains unchanged after the coordinate system transformation. On the other hand, z coordinate is not relevant as long as it is considered floor level, so it is not calculated neither.

The middle point corresponding to the pallet centre is calculated as:

$$tx = \frac{tx_1 + tx_2}{2}$$

$$ty = \frac{ty_1 + ty_2}{2}$$

And finally, the absolute distance between the pallet and the forklift can be obtained as:

$$d = \left({t_x}^2 + {t_y}^2\right)^{\frac{1}{2}}$$

To conclude, the rotation angle of the pallet respect to the forklift, β, is obtained geometrically as:

$$\beta = \tan^{-1} \frac{tx_2 - tx_1}{ty_2 - ty_1}$$

Finally, rotation angle of the pallet respect to the forklift, β, and the distance between the pallet and the forklift, d are not unknown parameters anymore.

3.4 Instructions for AGV’s controlling

Once the pallet distance and its relative angle with respect to the forklift are determined, several different scenarios can emerge. In each case, instructions given to the forklift in order to reach the pallet are different. In the following lines the different situations are presented in addition to the necessary AGV’s instructions for each of them.

First, notice that once the forklift is facing the selected pallet, a 0,5 metres distance must be left between them so that the AGV can move forward and introduce its forks in the two holes. Besides, the pallet can be reached from two sides, either on the right or on the left. In the following pictures it is going to be considered it is reached from the left side; analogously the same instructions are valid for the right side. By last, it is considered a tolerance zone for each of the sides (marked in green colour in the following picture) of 2 cm, in which the forklift should move at every moment.

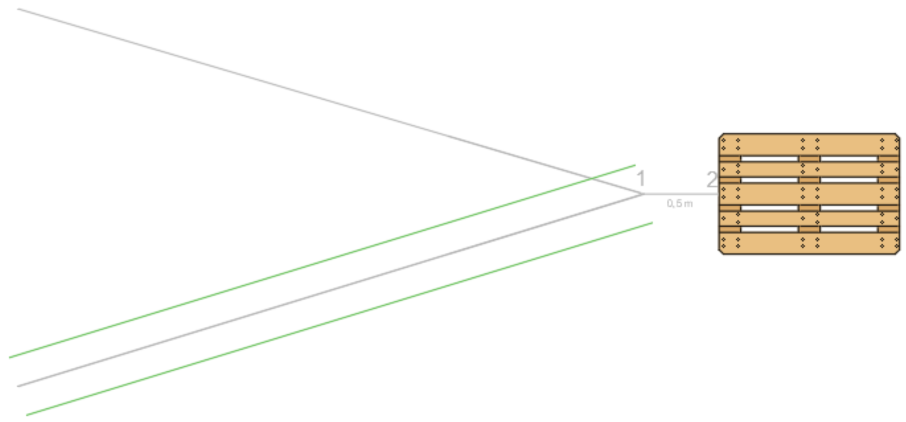


***Figure 23**: Tolerance zone in the right-side path*

With these conditions and taking as reference mark for the forklift the middle point between the end of its forks (shown in the picture below), different situations can be exposed.

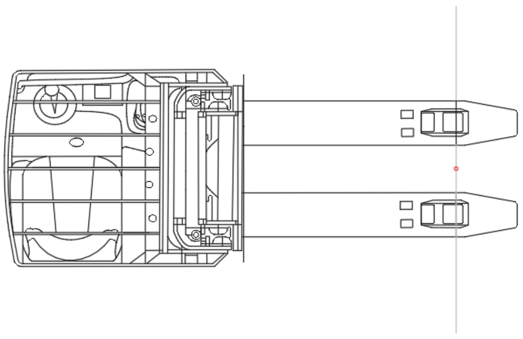


***Figure 24**: Reference point for the forklift*

Control is mainly done in 3 steps: reaching reference point 1; facing the pallet and move forward until its border; and finally introducing the forks into the holes by doing a final forward movement. The two first stages are controlled by the digital camera, while the third one is managed by the laser sensor.

- **STEP I**:

The final goal is to match forklift reference mark with point 1, corresponding to the 0,5 metres distance from the pallet.

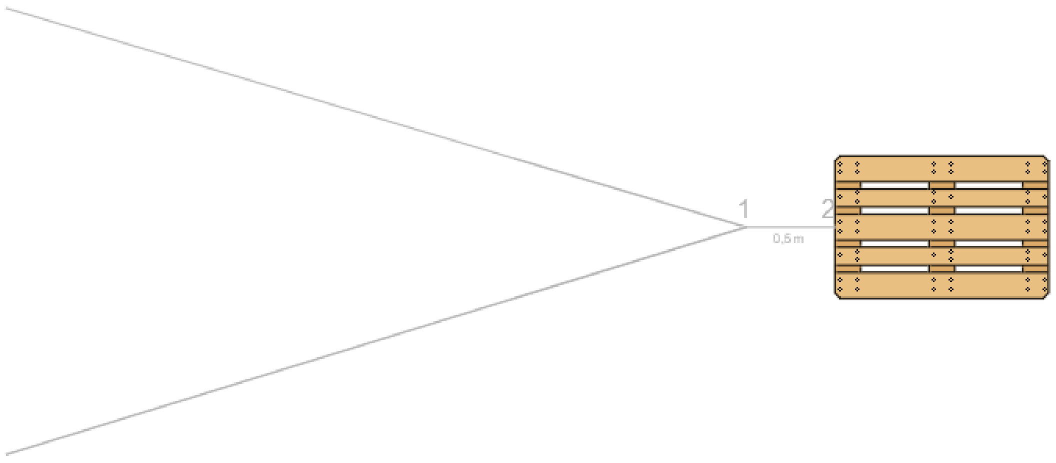


***Figure 25**: Illustration of points 1 and 2*

In total, 9 different cases are considered, and they are divided into three main groups depending on the position of the forklift with respect to the tolerance zone: AGV is on the left from the tolerance zone, AGV is inside the tolerance zone or AGV is on the right from the tolerance zone.

1. AGV ON THE LEFT FROM THE TOLERANCE

When the forklift is located on the left from the tolerance zone of the path, three different scenarios can be given.

1.1 AGV is going away from the path

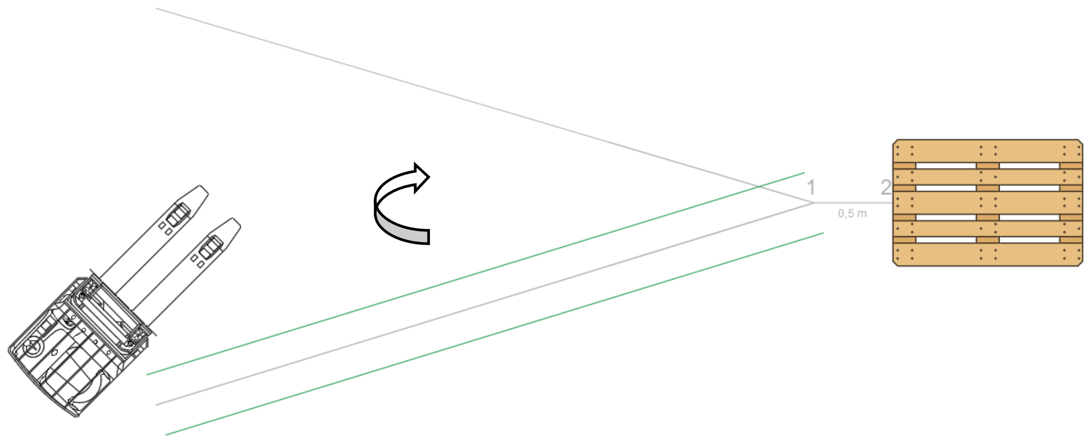


Figure 26*: Step I. Case 1.1*

The forklift must TURN RIGHT until it crosses the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

1.2 AGV is going towards the path

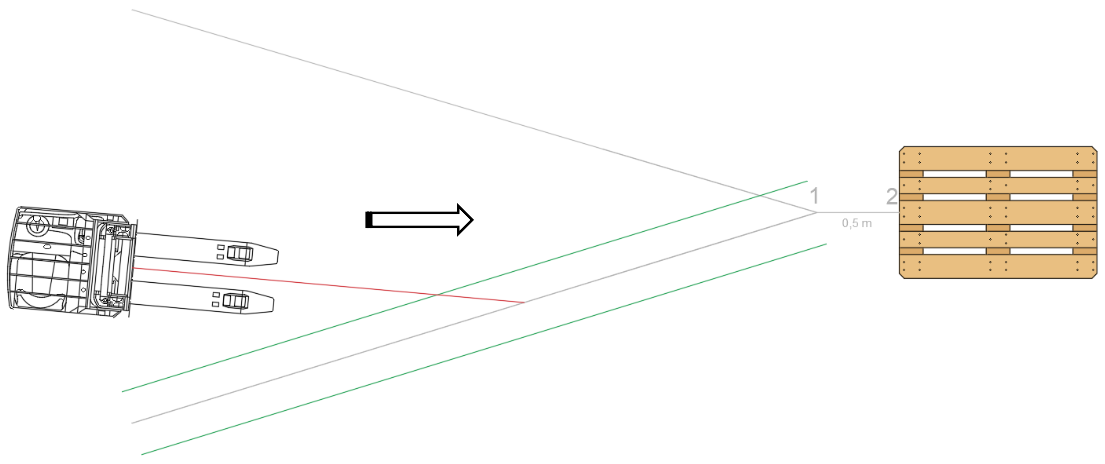


Figure 27: *Step I. Case 1.2*

The forklift must GO AHEAD until it crosses the stablished path and then continue through it until its reference mark reaches point 1, located 0,5 metres from the selected pallet.

1.3 AGV is going away from the path, crossing its continuity line after the 0,5 metres necessary distance

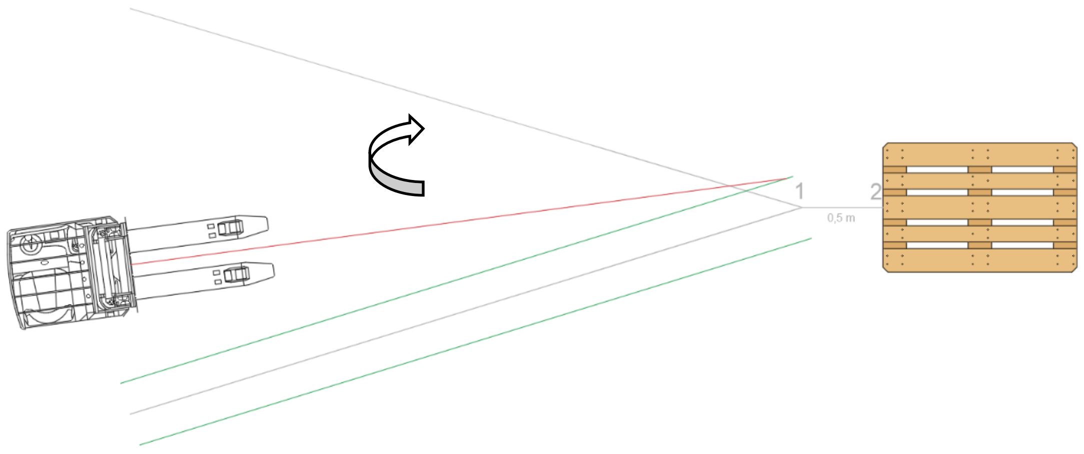


***Figure 28**: Step I. Case 1.3*

The forklift must TURN RIGHT until it crosses the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

2. AGV INSIDE THE TOLERANCE

If the forklift is located inside the tolerance zone of the path, three different scenarios can also be given.

2.1 AGV is rotated to the right

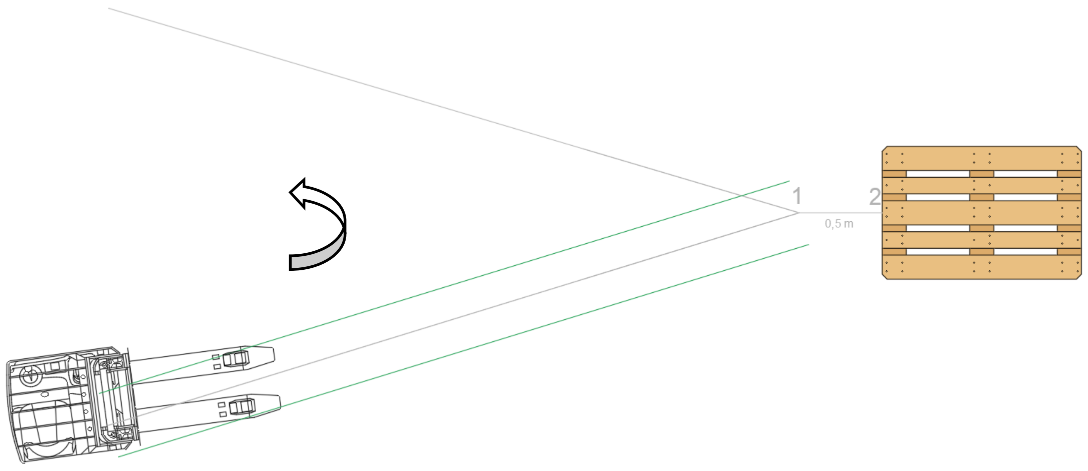


***Figure 29**: Step I. Case 2.1*

The forklift must TURN LEFT until it moves parallel to the stablished path and then continue through it until its reference mark reaches point 1, located 0,5 metres from the selected pallet.

2.2 AGV is going towards the path

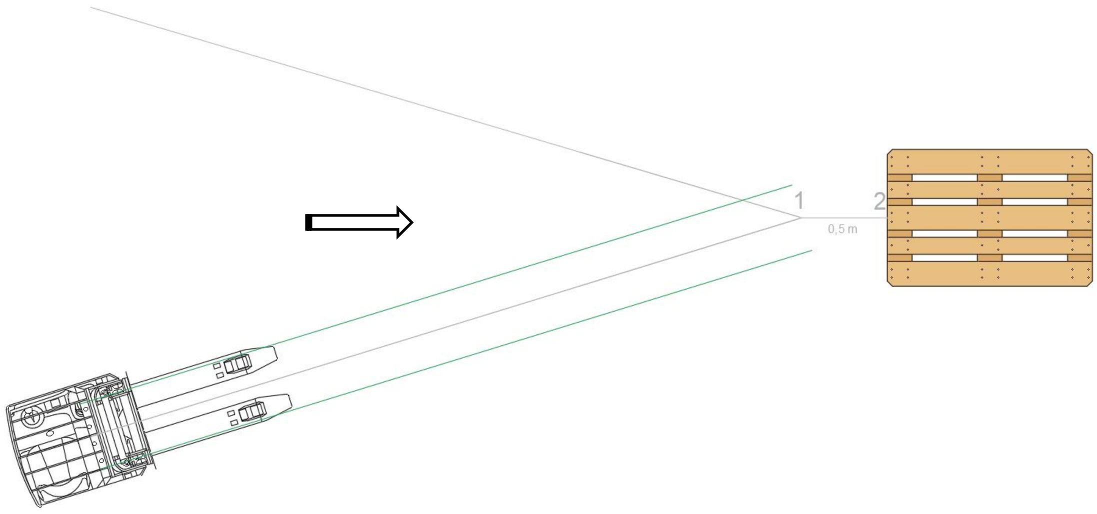


***Figure 30**: Step I. Case 2.2*

The forklift must GO AHEAD the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

2.3 AGV is going away from the path, crossing its continuity line after the 0,5 metres necessary distance

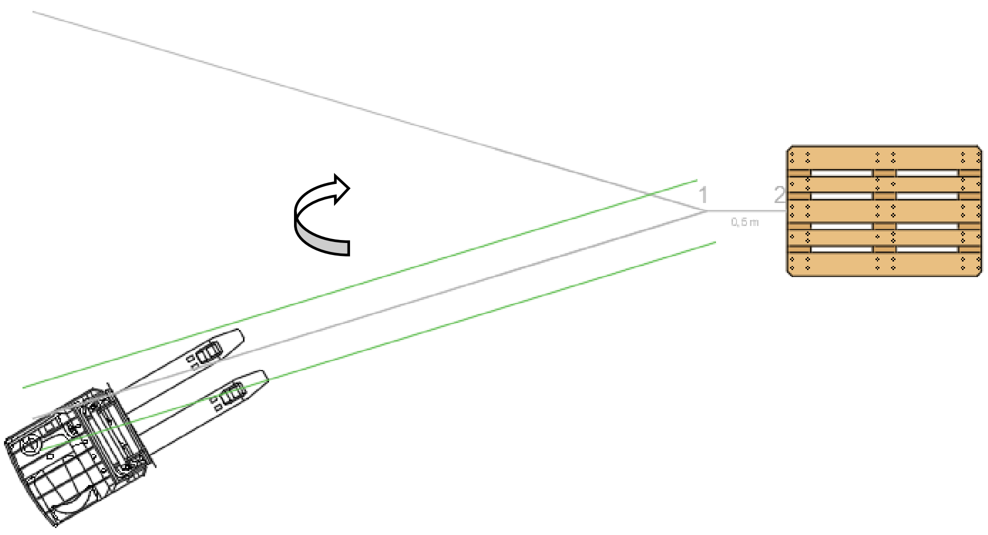


Figure 31: *Step I. Case 2.3*

The forklift must TURN RIGHT until it moves parallel to the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

3. AGV ON THE RIGHT FROM THE TOLERANCE

When the forklift is located on the right from the tolerance zone of the path, three different scenarios can be given.

3.1 AGV is going away from the path

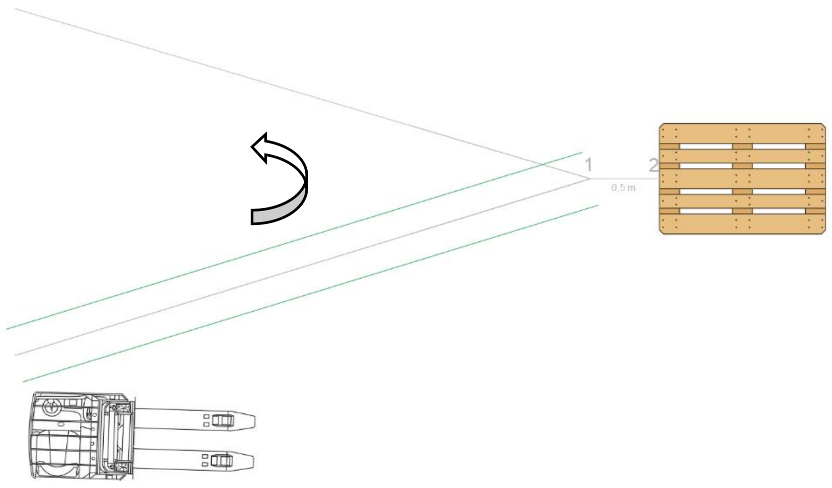


***Figure 32**: Step I. Case 3.1*

The forklift must TURN LEFT until it crosses the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

3.2 AGV is going towards the path

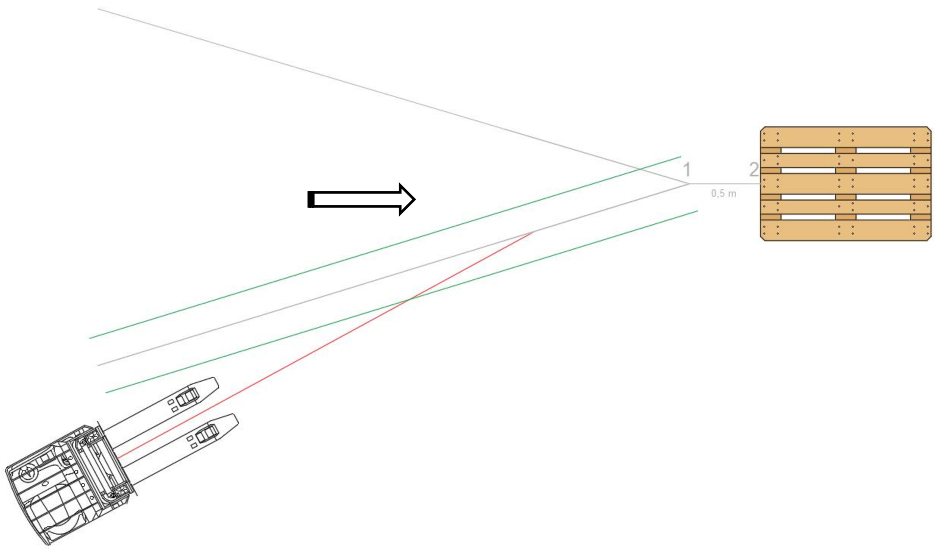


***Figure 33**: Step I. Case 3.2*

The forklift must GO AHEAD until it crosses the stablished path and then continue through it until its reference mark reaches point 1, located 0,5 metres from the selected pallet.

3.3 AGV is going away from the path, crossing its continuity line after the 0,5 metres necessary distance

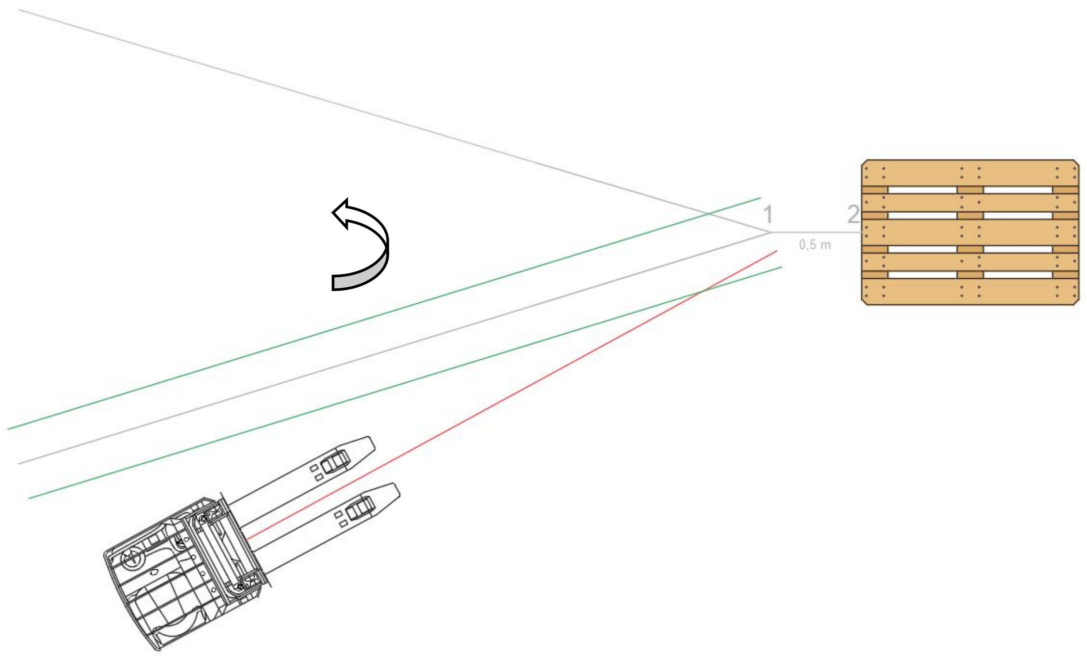


***Figure 34**: Step I. Case 3.3*

The forklift must TURN LEFT until it crosses the stablished path and then continue through it until its reference mark reaches point 1, which is located 0,5 metres from the selected pallet.

- **STEP II:**

Once the forklift comes to point 1, in this second stage it is rotated until it faces the pallet and then it goes forward until it gets to point 2. These two sub steps are shown in the following figures:

II.I Turn the forklift until it faces the pallet

As said before, it is considered that the pallet is reached from the left side so in this case it is necessary to TURN RIGHT; if it was reached from the right side the instruction would be to turn left.

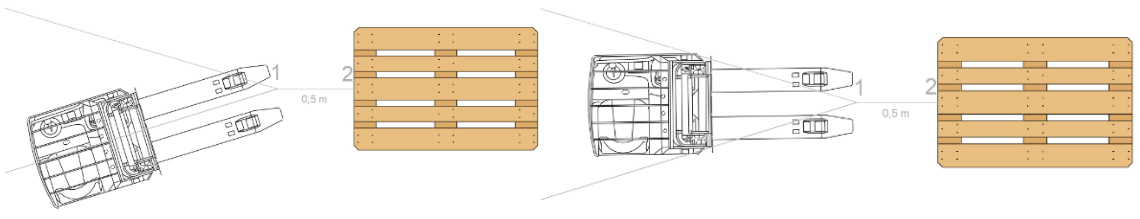


Figure 35*: Step II.I*

II.II Go forward until point *2*

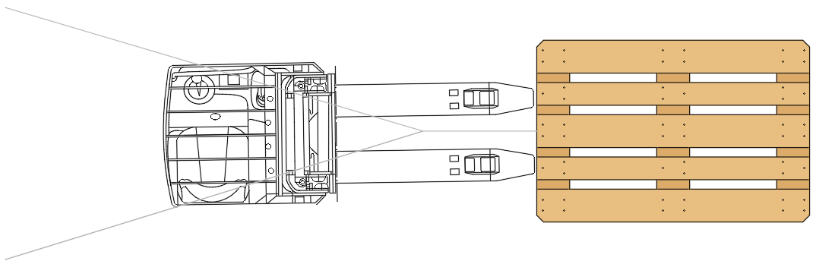


Figure 36*: Step II.II*

- **STEP III:**

The final step is to introduce the forks in the pallet holes and for that it is only necessary one last movement forward.

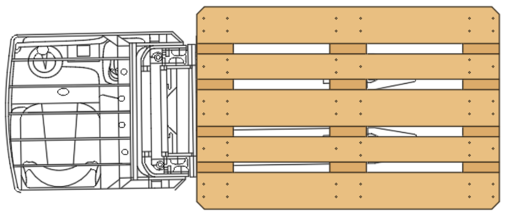


***Figure 37**: Step III*

It is worth pointing out that motion function is not going to be implemented until the end of my stay at the university. That is the reason why test results regarding the functioning of the proposed system for automating the forklift have not been included in this document.

4. Summary and outlooks

Industry 4.0 represents a new industrial era whose development is driven by a smart and interconnected environment thanks to cyber physical systems and the Internet of Things. Artificial Intelligence has been introduced in production plants and distribution centres where all the necessary transport operations are made in a completely autonomous way thanks to remoting control.

The goal of this project is to automatize a forklift belonging to the Continental Company, specifically the model Jungheinrich ESC 214/216, turning it into an Autonomous Guided Vehicle (AGV).

By means of the laser sensor SICK NAV 350, the digital video camera VK-S654R and a software run in the industrial PC known as the Blue Tank PC, the proposed forklift will be completely automatized.

The laser sensor can determine the global relative position of the pallet making use of reflector markers. Thanks to the digital video camera and the programming function blob_detector the image coordinates in the camera coordinate systems of the two pallet holes are known. Making use of complex equations solved with Matlab, these coordinates area translated into the reality coordinate systems. From them it is possible to obtain the pallet distance and its relative angle which will be the input data for the AGV to establish which are the necessary instructions to reach the pallet and transport it all along the warehouse.

To conclude, considering the complex scenario industry is facing these days, it can be said that it is not enough to work in the conventional way. Processes are pushed to automation. In my view, the proposed automation for the Continental Company is a trivial but very effective method that will decrease human labour necessity reducing human errors and thus, increasing the company’s productivity.

5. Annexes

Truck width of just 820 mm

Narrow aisle width for use in confined warehouse areas and block storage

Ergonomic sideways seated position for fatigue-free travel

3-phase AC technology for dynamic movement operations

ESC 214/216/214z/216z

Electric sideways-seated truck (1,400/1,600 kg)

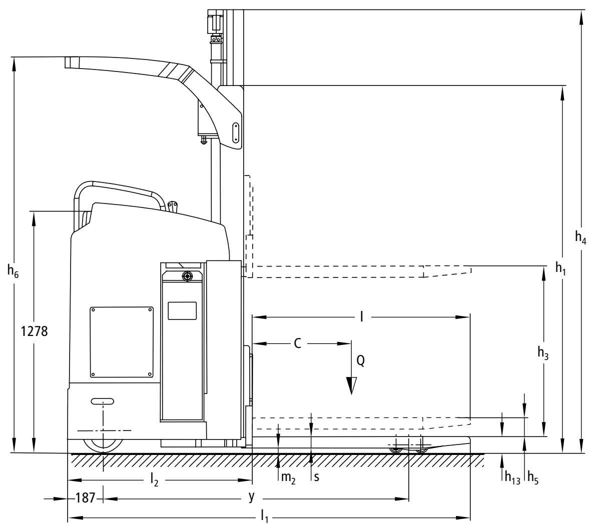
The Jungheinrich electric sideways seat stackers ESC 214/216 are ideally suitable for longer transport distances during stacking and retrieval. With a width of just 820 mm and a short working aisle width, the ESC shows particular strength in applications where space is limited and in block storage warehouses.
The comfortable sideways seating position provides the operator with excellent visibility. An advantage which comes into its own during frequent changes of direction. Individual height adjustment of the foot well provides a relaxed work environment for operators of varied heights.
Other features include:

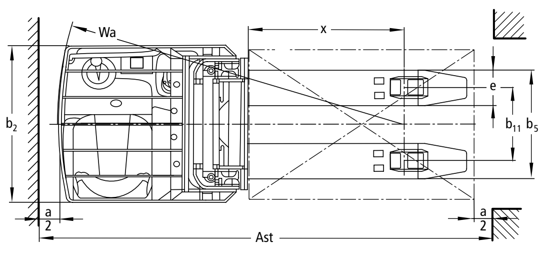
- Powerful, innovative 2.8 kW three-phase drive motor for high acceleration and performance
- High residual capacities and powerful lift motor.
- Extremely durable frame for very high operational demands.
- Display with steering position display and travel program selection (optional).
- Electric steering for easy manoeuvring with a few turns of the steering wheel
- Excellent energy management for low operating costs

The initial lift of the ESC 214z/216z allows it to cope extremely easily with uneven floor conditions. All travel and lift functions are performed using the multifunction lever which is within easy reach.
Thanks to the compact design and excellent ergonomics, the ESC is the ideal stacker for working in restricted spaces.

JUNGHEINRICH
Machines. Ideas. Solutions.

ESC 214/216/214z/216z





Standard mast designs ESC 214/216/214z/216z

	Lift h_3 (mm)	Lowered mast height h_1 (mm)		Free lift h_2 (mm)		Extended mast height h_4 (mm)		Height of overhead guard h_6 (mm)	
		ESC 214 / 214z	ESC 216 / 216z	ESC 214 / 214z	ESC 216 / 216z	ESC 214 / 214z	ESC 216 / 216z	ESC 214 / 214z	ESC 216 / 216z
Duplex ZT	2800	-	1995	-	100	-	3372	-	2093
	2900	1995	-	100	-	3422	-	2093	-
	3100	-	2145	-	100	-	3672	-	2243
	3200	2145	-	100	-	3722	-	2243	-
	3500	-	2345	-	100	-	4072	-	2288
	3600	2345	-	100	-	4122	-	2288	-
	4000	-	2595	-	100	-	4572	-	2288
	4100	2595	-	100	-	4622	-	2288	-
	4200	-	2695	-	100	-	4772	-	2288
	4300	2695	-	100	-	4822	-	2288	-
Duplex ZZ	2800	-	1945	-	1373	-	3372	-	2093
	2900	1945	-	1420	-	3422	-	2093	-
	3100	-	2095	-	1523	-	3672	-	2243
	3200	2095	-	1570	-	3722	-	2243	-
	3500	-	2295	-	1723	-	4072	-	2288
	3600	2295	-	1770	-	4122	-	2288	-
	4000	-	2545	-	1973	-	4572	-	2288
	4100	2545	-	2020	-	4622	-	2288	-
	4200	-	2645	-	2073	-	4772	-	2288
	4300	2645	-	2120	-	4822	-	2288	-
Triplex DZ	4200	-	1945	-	1376	-	4770	-	2093
	4300	1945	-	1426	-	4830	-	2093	-
	5250	-	2295	-	1726	-	5820	-	2288
	5350	2295	-	1776	-	5880	-	2288	-
	6200	-	2615	-	2046	-	6790	-	2288

Issue: 06/2010

Technical data in line with VDI 2198

Identification	1.1	Manufacturer (abbreviation)			Jungheinrich			
	1.2	Model			**ESC 214**	**ESC 216**	**ESC 214z**	**ESC 216z**
	1.3	Drive			Electric			
	1.4	Manual, pedestrian, stand-on, seated, order picker operation			seat			
	1.5	Load capacity/rated load	Q	t	1.4	1.6	1.4	1.6
	1.6	Load centre distance	c	mm	600			
	1.8	Load distance	x	mm	860	860	874[2]	874[2]
	1.9	Wheelbase	y	mm	1,648	1,648	1,677[2]	1,677[2]
Weights	2.1.1	Net weight incl. battery (see row 6.5)		kg	1,590	1,590	1,660	1,660
	2.2	Axle load with load front/rear		kg	1,316 / 1,674	1,340 / 1,850	1,285 / 1,775	1,370 / 1,890
	2.3	Axle load without load front/rear		kg	1,113 / 477	1,113 / 477	1,162 / 498	1,162 / 498
Wheels / frame	3.1	Tyres			Polyurethane			
	3.2	Tyre size, front		mm	Ø 230 x 77			
	3.3	Tyre size, rear		mm	Ø 85 x 85			
	3.4	Additional wheels (dimensions)		mm	Ø 140 x 126			
	3.5	Wheels, number front/rear (× = driven wheels)			2-1x/4			
	3.6	Tread width, front	b_{10}	mm	544			
	3.7	Tread width, rear	b_{11}	mm	385			
Basic dimensions	4.2	Mast height (lowered)	h_1	mm	1,995			
	4.3	Free lift	h_2	mm	100			
	4.4	Lift	h_3	mm	2,900	2,800	2,900	2,800
	4.5	Extended mast height	h_4	mm	3,422			
	4.6	Initial lift	h_5	mm	0	0	125	125
	4.7	Height of overhead guard	h_6	mm	2,095[3]			
	4.8	Seat height/stand height	h_7	mm	950			
	4.15	Height, lowered	h_{13}	mm	90			
	4.19.4	Total length including fork length	l_1	mm	2,125	2,125	2,140	2,140
	4.20	Length to face of forks	l_2	mm	975	975	990	990
	4.21	Overall width	b_1/b_2	mm	820			
	4.22	Fork dimensions	s/e/l	mm	60 / 185 / 1,150	60 / 185 / 1,150	60 / 185 / 1,150	65 / 185 / 1,150
	4.25	Width across forks	b_5	mm	570			
	4.32	Ground clearance, centre of wheelbase	m_2	mm	30	30	25	25
	4.34	Aisle width for pallets 800 × 1200 lengthways	Ast	mm	2,383[1]	2,383[1]	2,398[1]	2,398[1]
	4.35	Turning radius	W_a	mm	1,843	1,843	1,872[2]	1,872[2]
Performance data	5.1	Travel speed, laden/unladen		km/h	9.1 / 9.1			
	5.2	Lift speed, laden/unladen		m/s	0.15 / 0.24	0.13 / 0.24	0.14 / 0.23	0.14 / 0.23
	5.3	Lowering speed, laden/unladen		m/s	0.42 / 0.42	0.42 / 0.42	0.42 / 0.38	0.42 / 0.38
	5.7	Gradeability laden/unladen		%	7 / 12			
	5.10	Service brake			electric			
Electrics	6.1	Drive motor, output S2 60 min.		kW	2.8			
	6.2	Lift motor rating at S3 10%		kW	3			
	6.3	Battery as per DIN 43531 /35/36 A, B, C, no			no			
	6.4	Battery voltage/nominal capacity K5		V/Ah	24 / 465			
	6.5	Battery weight		kg	380			
Misc.	8.1	Type of drive control			AC SpeedControl			

[1] Diagonal in accordance with VDI: + 190 mm
[2] Load section raised: - 78 mm
[3] See mast table

In accordance with VDI Guideline 2198 this specification sheet provides details of the standard truck only. Non-standard tyres, different masts, optional equipment, etc. may result in different values.

Benefit from the advantages

Grab handle for safe access and for the fitting of various options

Comfortable seating position with height-adjustable footplate

Ergonomic layout of all controls

Excellent visibility through the mast

Innovative 3-phase AC technology
Better performance simultaneously coupled with reduced operating costs. Make the most of these advantages:
- High level of efficiency with excellent energy management.
- Powerful acceleration.
- Rapid change in direction of travel
- No carbon brushes - maintenance-free drive motor.
- Two-year warranty on the drive motor.

Productivity-enhancing ergonomics
The arrangement of the operator work-station, sideways on to the direction of travel, ensures optimum visibility and reduces potential neck complaints due to frequent changes in direction. A safety switch in the foot -well ensures maximum safety.
- Comfortable seat with adjustment for body weight.
- Padded armrest.
- Padded knee area.
- Grab handle for safe entry (also serves as the mounting point for storage or for a radio data terminal).

Travel comfortably and safely
Safe speed regulation by electronic Speed-Control. Speed-Control ensures maximum safety and effortless control of speed in all situations.
- Roll-back protection on gradients by self-acting brake.
- Constant travel speed both ways on gradients.
- Energy recovery by regenerative braking.
- Particularly smooth stepless operation.

Easy operation and good all-round visibility
- Operation of all drives and lift functions by multifunction lever within easy reach.
- Proportional hydraulics for precise load positioning.
- Smooth electric steering.
- Protected seating position within the enclosed truck profile.
- Free all-round view with no obstructive struts; good visibility through the mast and overhead guard.

Narrow truck with high manoeuvrability
- With a truck width of only 820 mm, the ESC is ideal for use in narrow aisles and block storage.
- Narrow working aisle width and well protected operator position for outstanding operation in narrow working aisles.

Robustness and ease of servicing
- Easy access to the controller and electronics.
- Side opening to the drive compartment.
- Robust, enclosed frame.

Powerful battery
- 3 PzS 465 Ah
- Lateral battery exchange on rollers.

Additional equipment
- Various storage options.
- Heated seat.
- Load guard.
- Display (travel direction, steering setting, battery status, operating hours, time of day, travel programs).

Jungheinrich Lift Truck India Pvt. Ltd

502 A & B, 5th Floor, Alpha Building,
Hiranandani Gardens, Powai,
Mumbai - 400076
India
Tel: +91 (0)22 2572-7900
Fax: +91 (0)22 2572-7945

info@jungheinrich.in
www.jungheinrich.in

T_ESC 214/216/214z/216z_122015_en_IN_000