Hochschule Ravensburg-Weingarten

Master thesis

Outdoor robot:

Characteristics and calculation of the mathematical model

Pablo Díaz Martínez

Supervisor in HS: Prof. Ralf Stetter.

Supervisor in UC3M: Prof. Mª Elisa Ruiz Navas

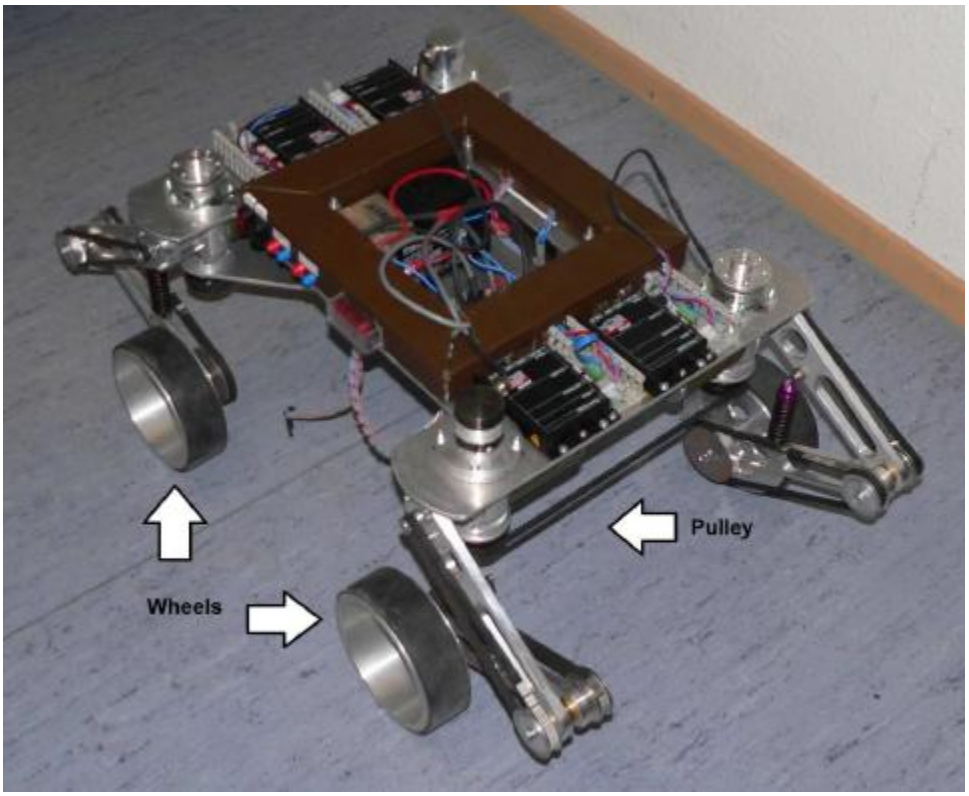
Duration time: Oct. 2010-Feb. 2011

Resumen

El siguiente proyecto fin de carrera denominado: Robot exterior, Características y análisis del modelo matemático. Ha sido realizado en la universidad Ravensburg-Weingarten de Alemania, en el departamento de mecatrónica. El proyecto ha sido resultado de 5 meses de trabajo con el profesor Ralf Stetter, poniendo a mi disposición un robot que llevan desarrollando un año, para que continuara con su desarrollo.

Un robot de exteriores está pensado para trabajar en condiciones de riesgo de repetibilidad y en malas condiciones atmosféricas. Lo impredecible de estas tareas hace que su desarrollo sea más complejo. Para el desarrollo de un robot tendrá que ser formado un grupo de trabajo con gente de diferentes areas, mecánica, electrónica y automática. Y una vez formado se diseñará entorno a las funciones que va a realizar, donde las va a hacer y el grado de automatización que necesita.

En el robot utilizado para el desarrollo de este proyecto fin de carrera, ya se han implementado la parte mecánica y la parte electrónica, por lo que se procederá al análisis de sus características y una vez conocidas estas al desarrollo de las posibles trayectorias que este podrá realizar. A continuación se muestra la imagen del robot objeto del proyecto.



En primer lugar se lleva a cabo un análisis de la parte mecánica del robot. Este está formado por 4 ruedas, cada una de ellas está unida al robot mediante un sistema de suspensión formado por dos brazos y un muelle, para así poder absorber mejor las dificultades del terreno. Tanto las ruedas traseras como las delanteras, van unidas en paralelo por una polea para llevar la misma dirección en todo momento. El robot está compuesto también por dos frenos. Cada uno está colocado sobre el eje vertical de giro de cada uno de los pares de ruedas, con la función de bloquear la dirección de las ruedas y así no perder el rumbo.

Cada una de las 4 ruedas está impulsada por un motor EC o brushless, estos motores son de la casa Maxon. Los motores están formados por tres partes, el motor propiamente dicho, una caja reductora para poder aumentar el par del motor, ya que el robot necesitará de pares altos para poder desplazarse, y un encoder con el que se puede saber la posición del motor en cada instante.

En el análisis de hardware la parte principal a analizar son los microcontroladores que lleva incorporados el robot. Estos son, al igual que el motor, de la compañía Maxon, y se llaman EPOS. Se puede ver en la figura a continuación.

Se ha implementado uno por cada rueda. Los EPOS están diseñados para controlar motores CC mediante encoders, así como motores EC mediante encoders o sensores de efecto hall. Un EPOS como entradas tiene la alimentación, el encoder, los sensores de efecto hall y algún pin más para cualquier utilidad, como salida tiene una para el motor, que controlando la cantidad de corriente podremos variar su velocidad, y algún pin más para cualquier otro uso. Los EPOS se pueden comunicar de dos maneras, una es integrando los en una red CAN, mediante el protocolo CANopen, para lo que dispone de dos puertos. Y así poder comunicarse todos los controladores del robot. Y otra manera de comunicación es el uso de puertos RS-232, pero este solo se puede utilizar para la comunicación directa con un ordenador o solo con otro EPOS.

El robot está formado también por dos encoders con los que podremos conocer el valor de la dirección de giro de los pares de ruedas con respecto del robot. Como maestro del robot es utilizado un pequeño ordenador llamado PC-ALIX. Este tiene 2 puertos USB y un disco duro de 4GB.

EL robot utiliza tres baterías de 12V, un de ellas para alimentar el ordenador y las otras dos, unidas en paralelo, son usadas para alimentar todos los motores y los encoders.

Por último hay que analizar el hardware del robot. Primero decir que el ordenador tiene instalado una versión reducida de windows xp ya que es el sistema operativo más compatible con los programas a utilizar en el futuro en el robot.

Para la comunicación primero es utilizado el protocolo RS-232, con este se comunica uno de los EPOS con el PC-ALIX. Cuando el EPOS tiene la información comprueba si es para él. En caso contrario usando la función “CAN gateway”, traslada la información a la red CAN que forman el resto de controladores y los dos encoders. La red CANopen sigue una estructura maestro esclavo y está orientada a nodos. Por lo que el EPOS que se comunica con el ordenador está programado como maestro y el resto como esclavos.

Un vez hecho todo el análisis del robot, se procede al análisis y desarrollo de las posibles trayectorias. Un robot de 4 ruedas puede hacer 3 posibles trayectorias mostradas en las siguientes figuras.

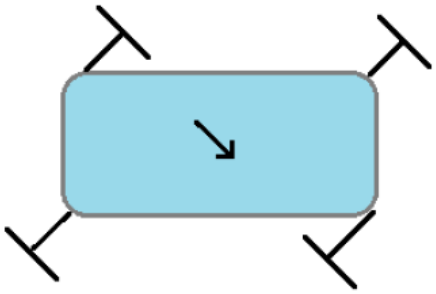


Figure 4.1: Parallel movement.

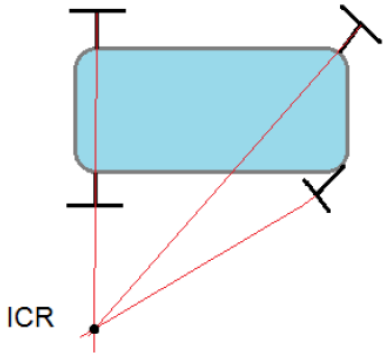


Figure 4.2: turning with a radius of curvature.

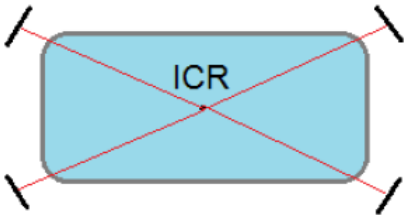


Figure 4.3: rotational motion of the robot.

El primero es el movimiento en paralelo, todas las ruedas son orientadas en la misma dirección y después este se mueve hacia delante o hacia atrás.

El segundo es el giro con un radio de curvatura. Los vectores angulares de la velocidad de cada rueda son orientados hacia un mismo centro instantáneo de rotación.

Es el giro del robot sobre sí mismo, se coloca el centro instantáneo de rotación en el centro del robot y este gira sobre si mismo. En este robot el último movimiento no se puede realizar ya que las ruedas están unidas 2 a 2, y esta restricción impide que todas las ruedas estén orientadas hacia al centro del robot, ya que siempre tienen que llevar la misma dirección 2 a 2.

Primero analizaremos y desarrollaremos el movimiento en paralelo. El robot se moverá llevando una trayectoria lineal en la dirección elegida. Para llevar a cabo este movimiento hay que seguir algunos pasos. Como el robot no tiene un motor dirigido a cambiar la dirección de las ruedas, debemos actuar primero sobre un par de ruedas bloqueando con el freno la dirección del otro. Lo que se debe de hacer es imprimir las velocidades con sentido opuesto a cada una de las ruedas del par y entonces estas girarán. Se hace primero en uno y luego en otro para asegurarse que se hace correctamente. Un vez giradas solo se debe dar a todas la misma velocidad ya sea para que se mueva hacia delante o hacia atrás. Dado que las ruedas están unidas por una polea la dirección de giro nunca puede llegar a ser de 90 grados ya que la suspensión de las ruedas golpea con esta impidiendo realizar correctamente el movimiento.

Después se realizó un programa en Matlab, para la simulación de la trayectoria del robot. Y en la siguiente figura se muestra una de las simulaciones.

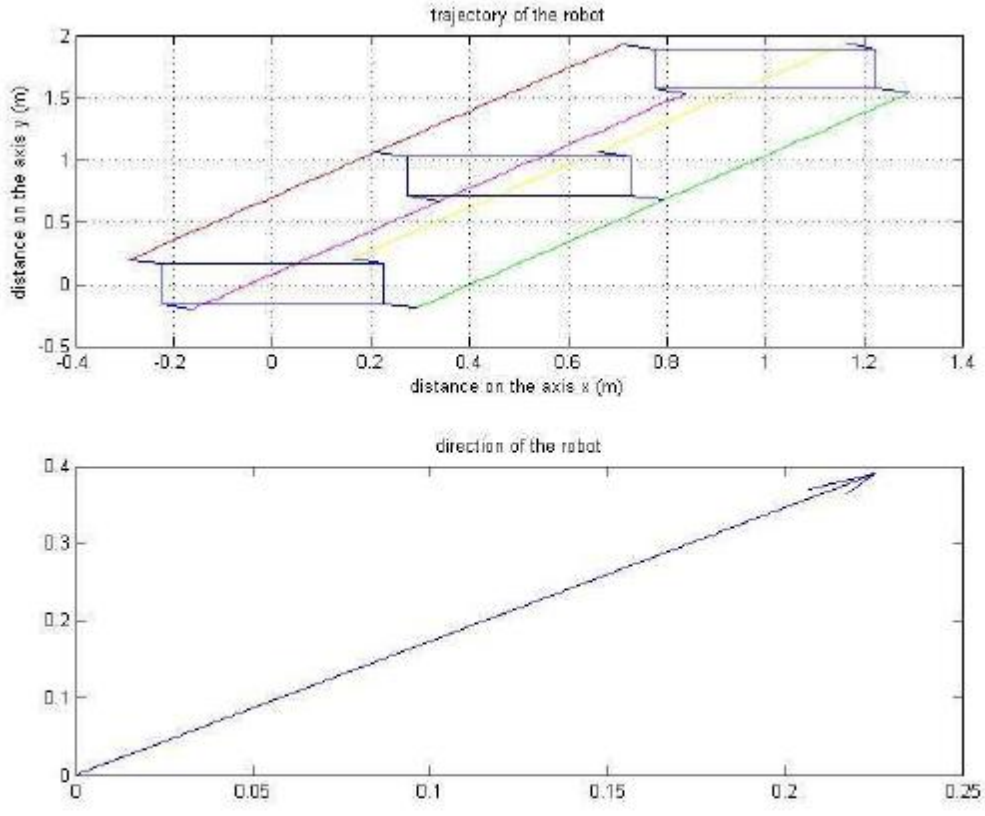
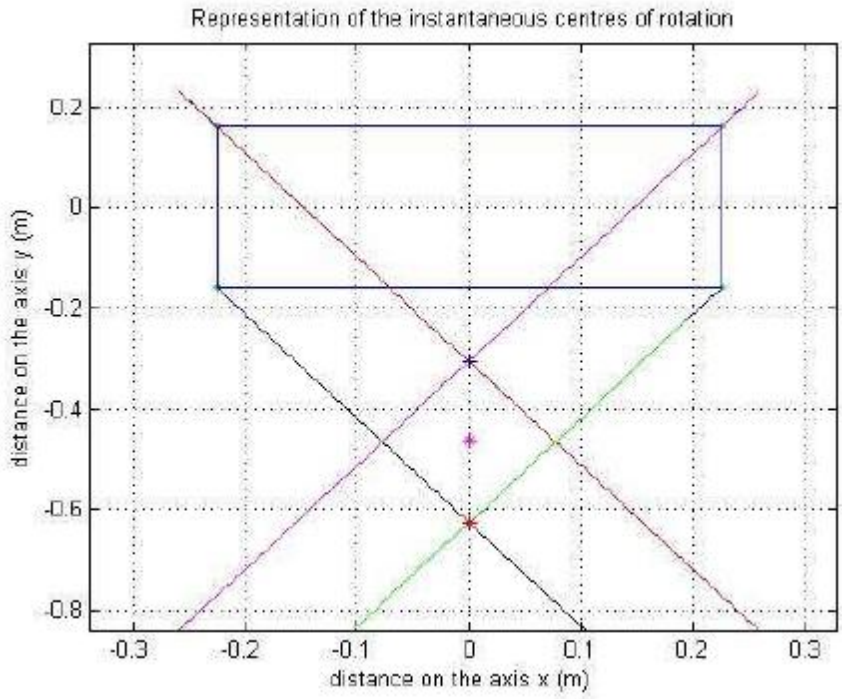


Figure 4.7: simulation of parallelmovement (60, 0.5, 4)

El siguiente tipo de movimiento a analizar es el giro con un radio de curvatura. El diseño del robot provoca algunas restricciones que puede causar problemas para seguir la trayectoria correcta. Para una correcta trayectoria todas las ruedas tienen que estar orientadas a un mismo centro instantáneo de rotación. Como están unidas en paralelo dos de ellas siempre tienen el mismo ángulo de de giro de modo que nunca pueden tener el mismo centro instantáneo de rotación. Por lo que se usa el programa Matlab para la desarrollar una trayectoria aproximada del robot y compararla con la trayectoria real que tendería a hacer el robot.

En el caso de este robot siempre se encuentran 2 centros instantáneos de rotación, uno para las ruedas del lado izquierdo y otro para las ruedas del lado derecho. Po lo que hay que elegir un centro instantáneo de rotación para todo el robot, la mejor manera es el uso de la media de los otros dos ya que estos siempre se encuentran a la misma distancia sea cual sea el ángulo de giro. En la siguiente figura se puede observar los dos centros y el nuevo calculado.



Una vez obtenido el centro de rotación para todo el robot, con la distancia al centro del robot y la velocidad lineal requerida para el robot la velocidad angular del robot es calculada. Y con esta el nuevo centro de rotación son calculadas las velocidades aproximadas en cada una de las ruedas del robot. En la figura siguiente podemos observar una la simulación de una de las trayectorias del robot.

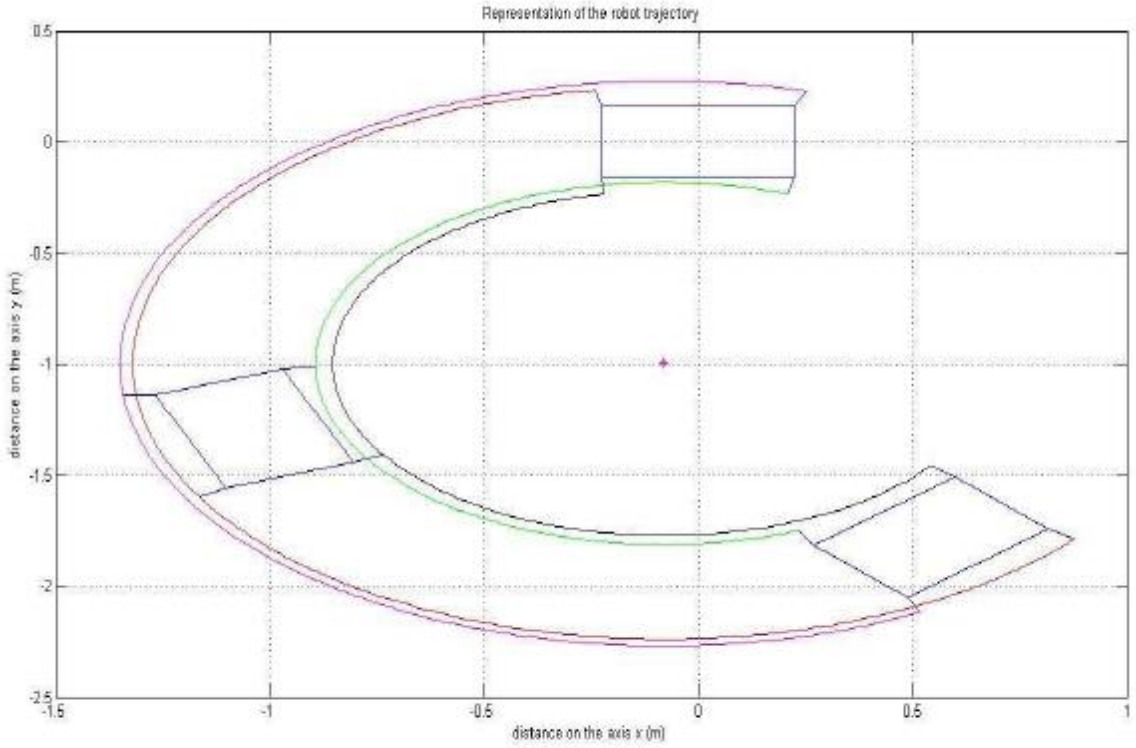
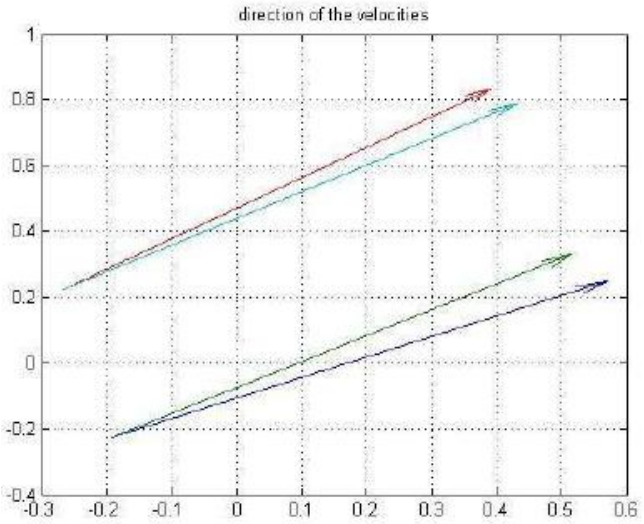
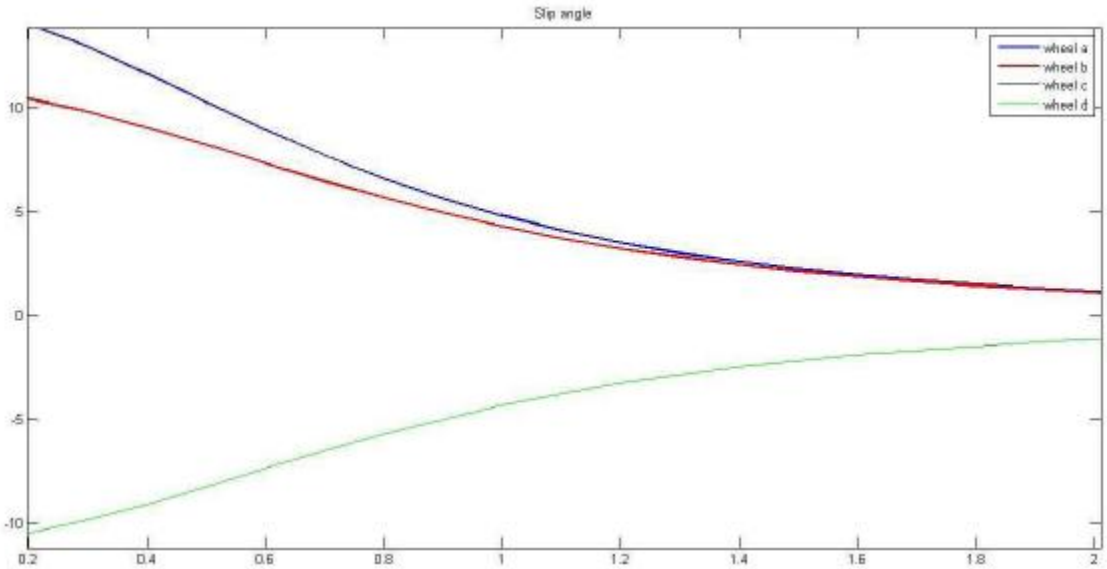


Figure 4.16: Trajectory for α12=0.15 and α34=0.3 Rad.

El problema es que esta trayectoria es la que el robot seguiría en un caso ideal, pero dado que las ruedas están girando con respecto a su centro de rotación real, y no con respecto al aproximado, la velocidad real y la velocidad aproximada tendrán direcciones ligeramente diferentes. Esta diferencia la podemos observar en la gráfica de a continuación.



En la gráfica se puede observar que el ángulo de diferencia entre ambas direcciones no es muy alto. Realizando varias simulaciones observamos que los valores de estos ángulos son siempre de alrededor de 7 a 8 grados. Pero los valores de estos ángulos pueden ser corregidos. En la simulación no fue considerado el deslizamiento de las ruedas. El deslizamiento es producido debido a las fuerzas laterales que soportan las ruedas, estas son perpendiculares al movimiento y son producidas por la deformación del neumático durante el giro. Produciendo una variación en la dirección de la velocidad. El ángulo entre estas dos velocidades es denominado ángulo de deslizamiento. Utilizando un programa desarrollado por una compañera de la universidad e introduciendo las características de nuestro robot, comprobamos los posibles valores de los ángulos de deslizamiento. Estos están mostrados en la siguiente gráfica.



Como se puede observar los valores de los ángulos de deslizamiento tienen similares valores a los a los ángulos debidos a las diferencias en la direccion de la velocidad de las trayectorias reales y las trayectorias aproximada. Por lo que la diferencia entre ambas se verá corregida por este ángulo de deslizamiento.

Acknowledgements

I want to thank my supervisor of the Hoschule Ravensburg-Weingarten, Prof. Ralph Stetter, for his helpful advices during the semester.

The most special thanks to my parents, for their advices, their support, their affection and for giving me the opportunity of starting my own future. Thanks to my sister, who was always waiting for me at home with the biggest hug. Also thanks to the rest of my family for making me feel like I am part of something special.

Thanks to my friends since childhood, “los sotillo”, for helping me to disconnect from the studies and for showing me that life has different points of views. Thanks also to my friends of the University for make me feel like at home at the university. And also thanks to my friends from Germany, for giving me the last push to finish my degree.

And at last, special thanks to Maider for all her support and being there for me at every moment.

Nomenclature

α	alpha, angle turn of the wheels [rad]
r	distance between the contact point and the turn axis [m]
θ	the initial rotation angle of the wheels frames [Rad]
a	distance in the axis y between the wheel and the turn axis [m]
b	distance in the axis y between the wheel and the turn axis [m]
α12	the rotation angle of the wheels 1 and 2. [Rad]
l	distance between the vertical turn axis of the rear and the front wheels. [m]
h	distance between the vertical turn axis of the right and the left wheels. [m]
x11	horizontal position of the wheel 1 frame axis. [m]
y11	vertical position of the wheel 1 frame axis. [m]
T1	represents the rotation matrix of the wheel 1 frame.
x21	horizontal position of the wheel 2 frame axis. [m]
y21	vertical position of the wheel 2 frame axis. [m]
T2	represents the rotation matrix of the wheel 1 frame.
α34	the rotation angle of the wheels 3 and 4. [Rad]
x31	horizontal position of the wheel 3 frame axis. [m]
y31	vertical position of the wheel 3 frame axis. [m]
T3	represents the rotation matrix of the wheel 3 frame.
x41	horizontal position of the wheel 4 frame axis. [m]
y41	vertical position of the wheel 4 frame axis. [m]
T4	represents the rotation matrix of the wheel 4 frame.
vl	the lineal velocity of the robot. [m]
R	global radio of the robot to the ICR. [m]
av	angular velocity of the robot. [Rad/s]

Content

1. Objectives

The objective of the following project is to work on an outdoor robot. It has been designed to explore and work in open areas or indoor places as big pipes or caves. The main idea of the project is to develop the movements of the robot. For which the program MATLAB is used to simulate the trajectories.

The robot can make some type of movements; they are evaluated acording to the characteristics and the restrictions of the robot. First all the components of the robot are analysed. Once finished the analysis, it is used to develop the possible trajectories of the robot.

The robot is controlled by a small PC that it is connected to 4 EPOS controllers, one per wheel. For the communication between the controllers it is used the CANopen protocol. It has also two encoders connected which read the angle of the directions from the rear and the front wheels. The front wheels are moved in parallel, due to the fact that they are connected by a pulley. The rear wheels are connected in the same way as the front wheels.

2. Introduction to the outdoor robots

The new challenges of robotics are the design and development of outdoor robots, making them autonomous or semiautonomous. They should be able of working in situation of high risk, repeatability and difficult weather conditions. The unpredictable of these situations make this task more difficult. Some of the areas more interesting in these robots are the mining, the agriculture and those related with the exploration permanent frozen zones, in desert zones or in planetary.

To develop an outdoor robot we need a team work with people from different areas, knowledge as mechanic, electronic, and artificial intelligence is required. To design an outdoor robot we have to think in the mission that it has, because we need to know the task to develop; the terrain where it works; the environmental conditions where it works; the grade of automation that it needs.

Depending on the terrain, it should be implemented a different transmission of the movement between the robot and the environment; like wheels, legs or chains. Then it has to be chosen the way to transmit this movement, it has different electrical motors or it can implement a combustion engine too.

Once made this; the connection between all the parts of the robots, sensors, actuators, motors and controllers must be made. As well it has to be chosen the protocol to communicate them.

When the robot is assembled, the programming work takes place. It is different if the robot is automatic or semiautomatic. It has to be considered the way that it is going to be controlled, and if the work area is known.

3. Analysis of the robot.

The robot is made to work in opening areas. The structure of it is thought to work in different grounds. We can divide the analysis of the robot in three parts: mechanical, hardware and software.

The structure of the robot weight 28 kg. In the figure 3.1 we can see the dimension of the robot.

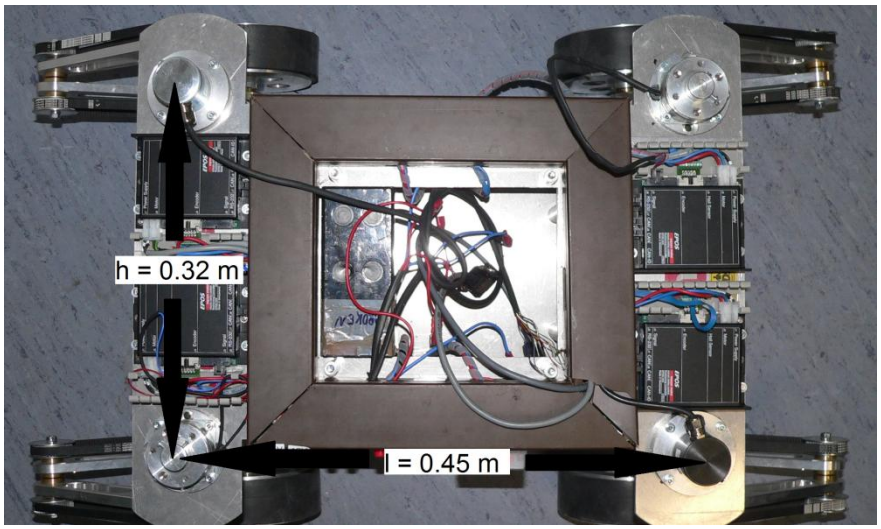


Figure 3.1 with the dimensions of the robot

3.1 Mechanical analysis.

The robot is moved by 4 wheels. The wheels are located in each corner of the robot, as it may be seen in the *figure 3.2*. The wheel has a big radio; it permits the robot to solve better the difficulties of the terrain. The surface of the wheel is made of rubber; it provides better traction to the robot.

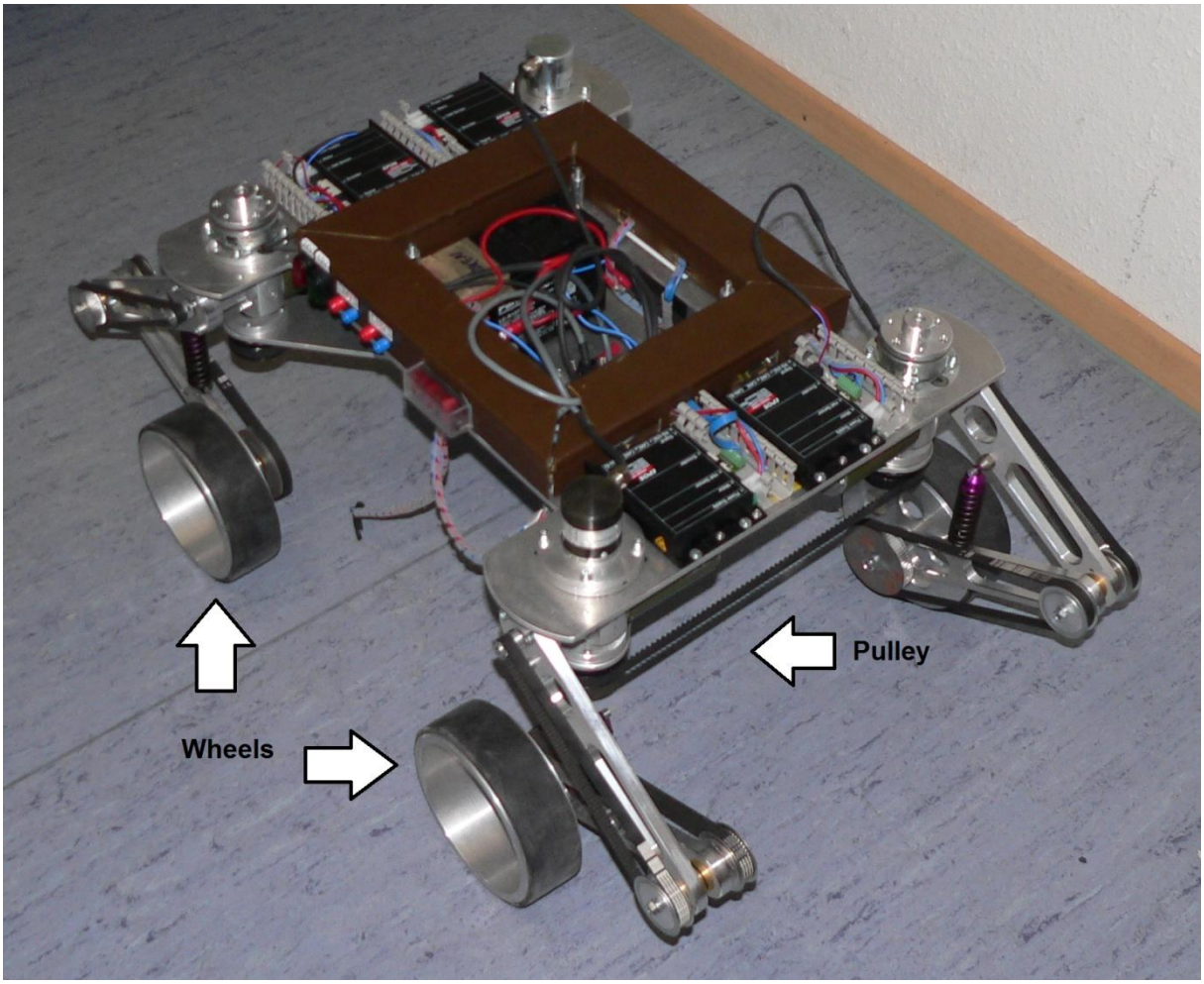


Figure 3.2: Collocation of the wheels and the pulley.

In the *figure 3.3* it can be seen that the wheels are joined to the robot by a traction system made of two arms and one suspension. This system permits the robot to absorb the different heights of the terrain and the impacts against the ground elements, as stones, sticks, holes in the ground. There are two pulleys to transmit the movement from the motor to the wheel. This transmission is a reduction system, because the robot needs more torque than the motor can give.

Figure 3.3: Traction system of the robot.

The front wheels are joined in pairs by a pulley as well as the rear wheels. The pair of wheels has the same direction at every moment. When the wheels are aligned in a straight line, they can turn almost 90 degrees in each direction. The pulley can be seen in the *figure 3.2.*

The robot has two brakes which can be seen in the *figure 3.4*. They are over one of the two wheels, from the pair, to hold them; when it has been decided to continue with the same direction.

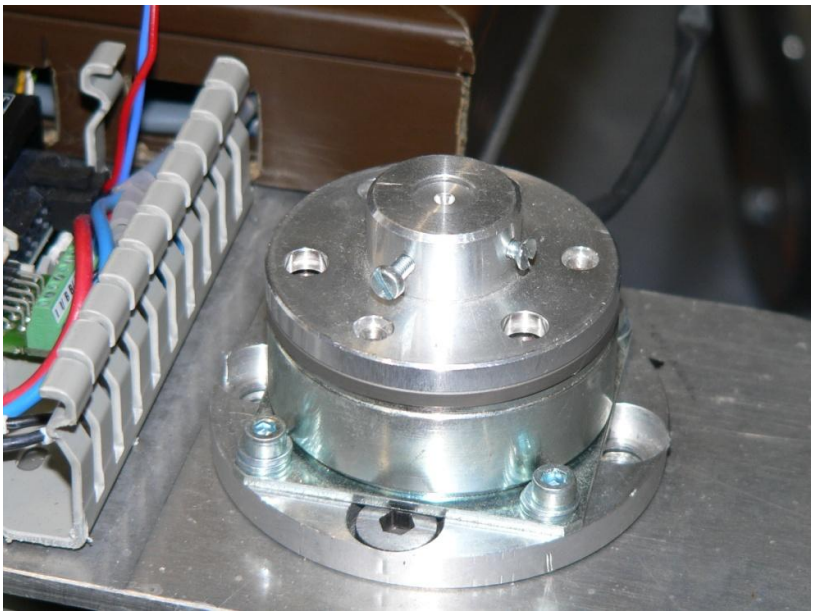


Figure 4: brakes.

The robot had installed four brushed motors, which had been changed due to the fact that they were damaged. The students, who worked before in the robot,

made wrong the calculation of the torque needed. They tested the motor in the robot, as the motors didn't have enough power to move the mass of the robot, they were broken. Other student, who worked in the project, looked for new motors to fulfil specifications in the robot.

The motors are electrical from the Maxon Company. The motor is composed by three parts: a motor, a planetary gearhead and an encoder. The motor is an electronically commutated motor or also called brushless. They have better properties and characteristics than the brushed motors to control the angular position. The robot requires stopping the motor in definite positions; therefore the bought brushless motors are the best choice. It has hall sensors which can control the position of the rotor. The proper motors have been selected considering the power supply provided by the batteries, the necessary torque and the velocity required. The characteristics of the motor can be seen in the *table 3.1*.

Table 3.1: Motor characteristics

Characteristics of EC 40 I		
Nominal voltage	V	24
Nominal speed	rpm	10400
Nominal torque	mNm	48,2
Nominal current	A	2,64
Max. Efficiency	%	84

A planetary gearhead is used to achieve a higher torque. It has a reduction ratio of 14:1. This ratio manages to transform velocity to torque which is needed in the developed robot. The efficiency of the gearhead is of 75%.

The last part is the encoder; which job is to precisely control at every moment the position of the motor. The resolution is 500 counts per turn.

The future students have to apply and run the new motors on the robot. To do this work, they have to design a new piece to fix the motor to the structure; because the dimensions of the new pieces are different from the older ones.

3.2 Hardware analysis.

The motors are controlled by EPOS controllers; there is one EPOS per motor. The EPOS controller is from the Maxon Company. The EPOS is a small-sized full digital smart motion controller [1]. Due to the flexible and high efficient power stage, the EPOS 24/5 drives brushed DC motors with digital encoder as well as brushless EC motors with digital Hall sensors and encoder.

The EPOS is specially designed to be commanded and controlled as a slave node in a CANopen network. And it can be operated trough a RS-232 communication port also.

The outputs and inputs can be seen in the *figure 3.5.* It has as output the motor control (J2) and some pins to send outputs signals (J5). As inputs, it has the encoder (J4), the power supply (J1), the hall sensor (J3) and some pins for the inputs. Also it has one port RS-232 (J6) and two ports CAN (J7 & J8). It has 8 pins to give it the CAN-ID.

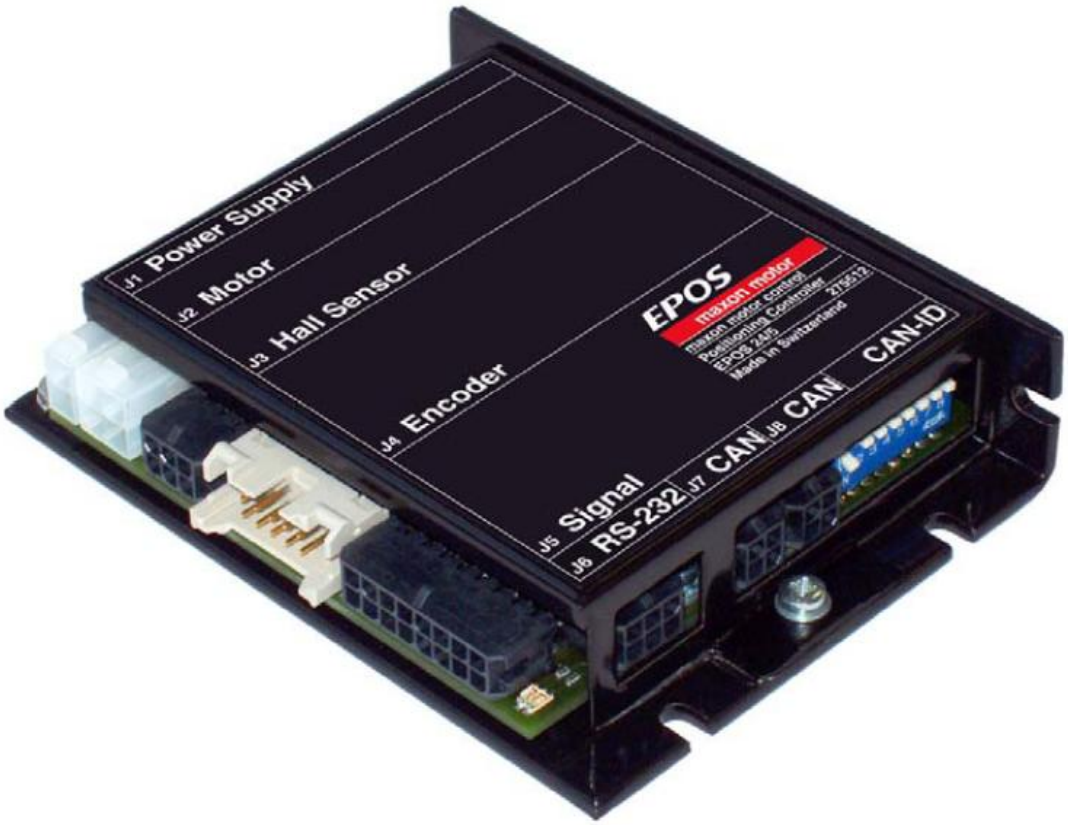


Figure 3.5: EPOS controller [1]

In the robot, the motor output is used to send the current necessary to move the wheels. The encoder input is used to know the position of the motor. Two

output signals are used to control the brakes that keep the trajectory. Only in one of them the RS-232 is used for communication between the EPOS and the PC-ALIX. And the CAN ports for the communication between the EPOS and other two encoders.

The robot has other two additional encoders. These encoders are used to know the angle between the pair of wheels and the robot. As it can be seen in the *figure 3.6,* they are allocated in the vertical axis of one wheel from the pairs of wheels.

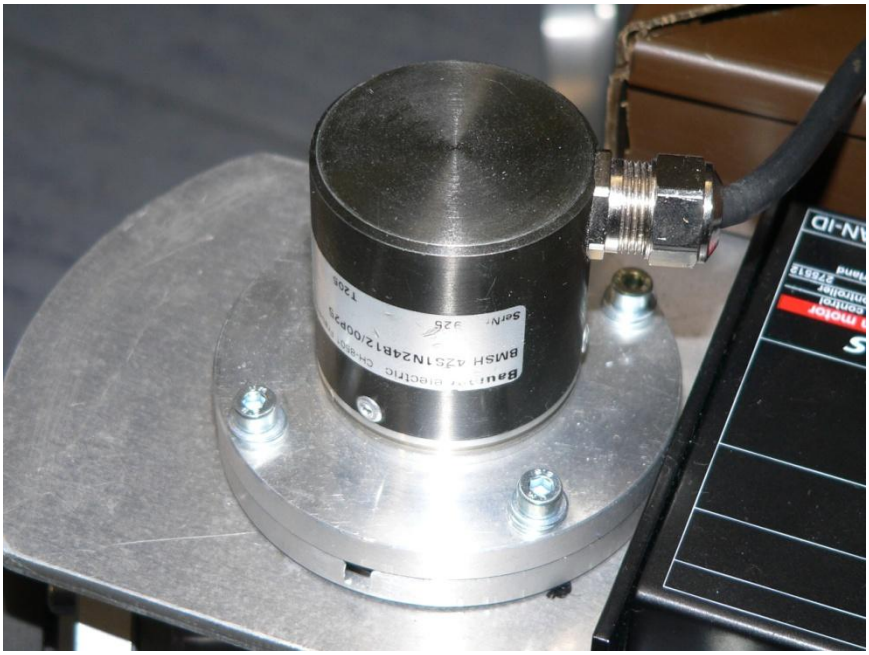


Figure 3.6: Encoder

The master of the robot is a small PC-ALIX. It has two USB ports and a VGA output. It has as a hard disk a compact flash memory of 4 GB. It presented problems at the beginning; because this capacity is a limitation to install the programs used in the computer.

The robot uses three batteries of 12V each. Two of them are connected in parallel giving 24V. They are used to give the power supply to the EPOS, the encoders and the motors. The last battery is used two give the supply to the PC-ALIX. The robot has two buttons to connect the batteries and has a fuse box to keep safety the electrical installation.

The cable connection is showed in the *figure 3.7.* It has two types of cables one for the CAN communication another type for the power supply.

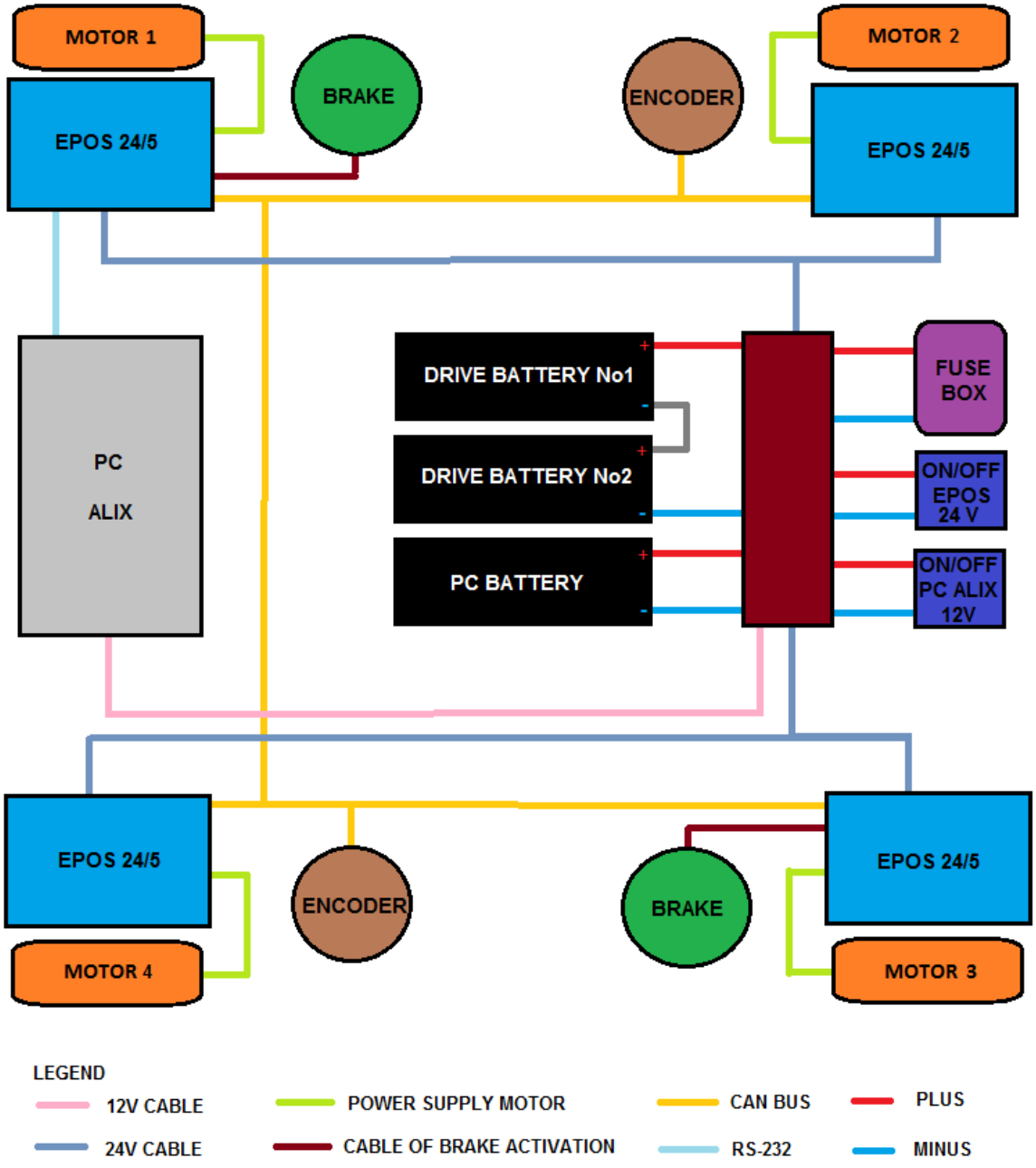


Figure 3.7: Cables connections.

3.3 Software analysis.

The PC had installed the operating system Debian 5. It was changed for the windows xp because it is the most compatible operating system with the programs that are used for programming and communication. The version installed has the basic things of the operating system.

It has been used the protocol RS-232 for the communication between the PC and one of the EPOS. When the EPOS has the information, it can transmit this to the others controllers by the CAN gateway functionality.

It has been used the protocol CANopen for the communication between the EPOS and the encoders. The communication by CANopen between the EPOS is the fastest way of communication, since it can be used in different operating modes as: position, speed and current regulation. The CANopen interface allows the networking to multiple axis drives and online dominated by a CAN bus master. For the transmission of the data, it is used the Process Data Objects (PDOs)[2]. They are unconfirmed services, they don't have protocol overhead. It makes a faster and flexible way to send data from one controller to any other controller. All the PDOs have a unique identifier, it can be sent by only one node, but it can be received by more than one. The CANopen management network follows a structure master/slave and is node orientated. So it has to be a network management master and the rest of the nodes are network management slaves. The network can be seen in the *figure 3.8*.

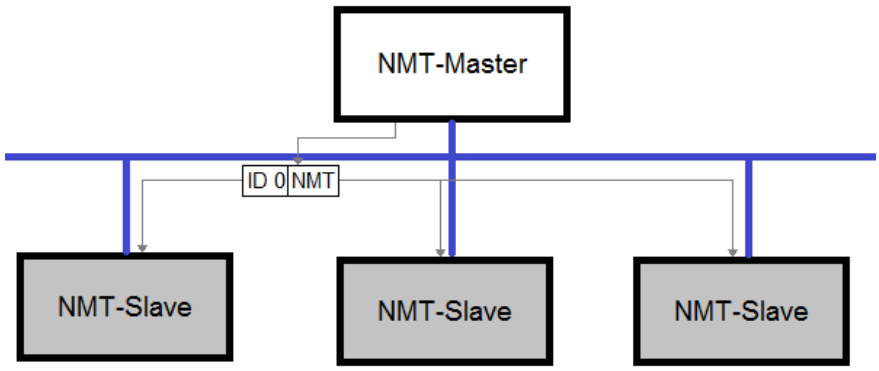


Figure 3.8: Network Management (NMT)

4. Trajectories of the robot.

A robot formed by four wheels can have some different movements. This can be divided in three groups [3]. The first one is a parallel movement; the wheels are put in the same direction and then it goes straight forward or backwards. The *figure 4.1* shows the movement. One useful use is to avoid obstacles with fewer manoeuvres.

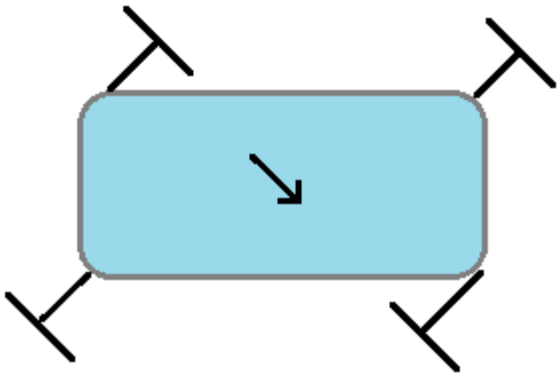


Figure 4.1: Parallel movement.

The second one is turning with a radius of curvature; the wheels are orientated in the way which they have a common instantaneous centre of rotation (ICR). The radius and the position of the ICR can be changed. This allows easily changing the trajectory. An example is shown in the *figure 4.2.*

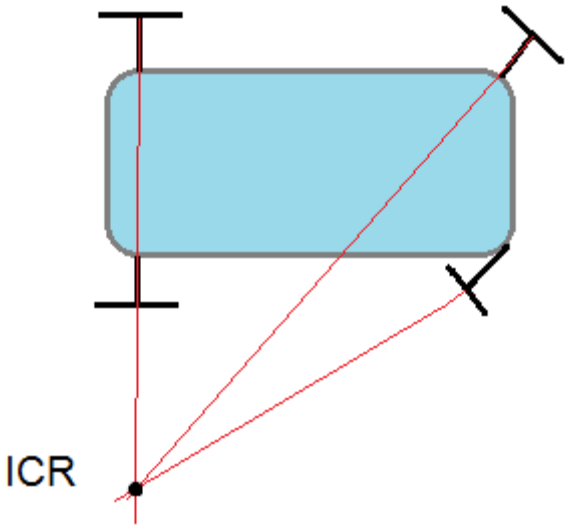


Figure 4.2: turning with a radius of curvature.

The last one is the rotational motion of the robot. It can be seen the movement in the *figure 4.3*. It has the ICR in the centre point of the robot. It can turn in much reduced spaces.

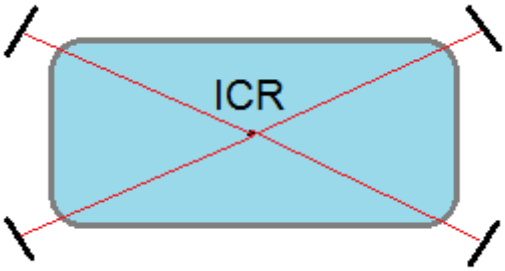


Figure 4.3: rotational motion of the robot.

The robot has the wheels joined in pairs. This means that the rear wheels and the front wheels are moved in parallel. This is a restriction in the degrees of freedom of the robot. From the group of movements shown before, the robot can only do the first two. It can´t do the last one, because the wheels can´t be orientate to the centre of the robot.

Below it is described in detail the two possible trajectories and shown some simulations made with the mathematical program Matlab.

4.1 Parallel movement

The robot follows a linear trajectory in an angle chosen. To carry out this movement, some steps must be followed. First the wheels have to be turned in the direction chosen. The robot doesn´t have a motor to change the direction of the wheels. To make the turn, the same velocity but with different direction must be given to each wheel of the pair. For safety reasons each pair must turn in different times, blocking the other with the brake. After the turn is made, all wheels get the velocity wanted to move forwards or backwards. In the *figure 4.4*, it can be seen the position of the wheel in the robot for this movement.

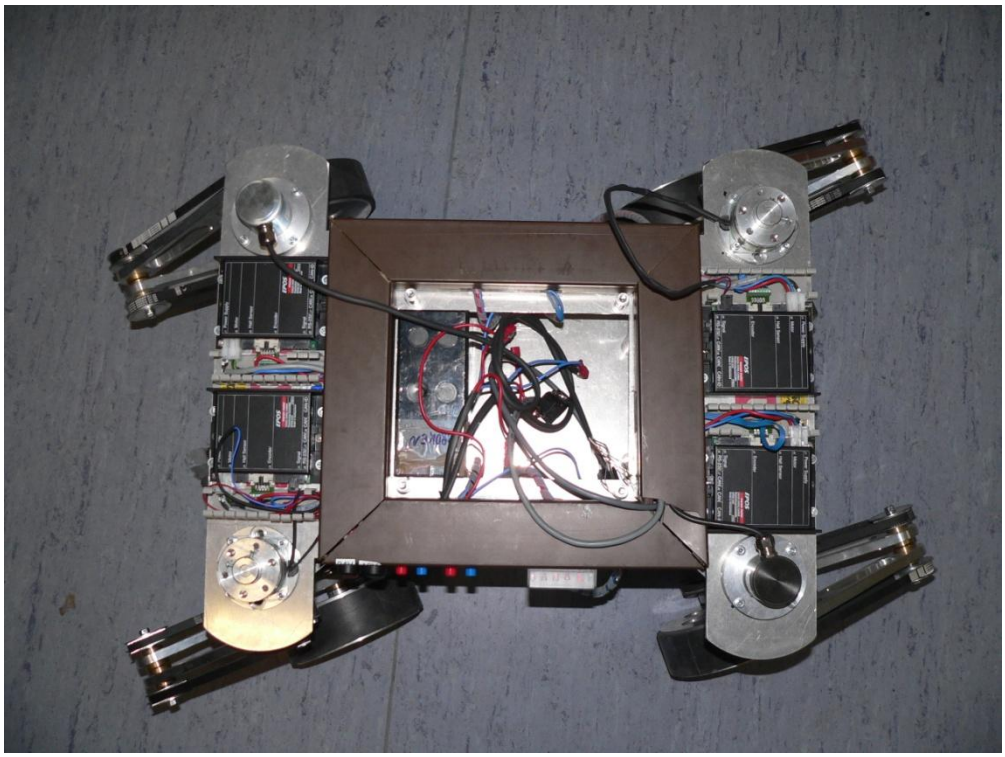


Figure 4.4: The robot in the parallel movement.

The wheels can turn almost 90 degrees to each direction. They can´t turn more because the arms, that join the wheel with the motor, hit the pulley between the two wheels. It can be chosen the forward direction or backward direction, by change the direction velocity of the wheels.

The contact point of the wheel with the ground is not in the same vertical axis that the turn axis of the wheel with respect to the robot. It should be considered if whiling to simulate the trajectory. First considered that in the initial position the wheels are aligned in the straight direction, the turn angle is 0. This angle is called alpha (α). Calculations for the centre of the wheel are made using trigonometry. The contact point is always at the same distance to the turn axis. For the calculation of this distance the next equations are used, it is calculated the distance r between the contact point and the turn axis and the angle θ that makes with the axis x; as we can see in the *figure 4.5*:

[4.1]

[4.2] –

Where:

a= distance in the axis y between the wheel and the turn axis.

b= distance in the axis x between the wheel and the turn axis.

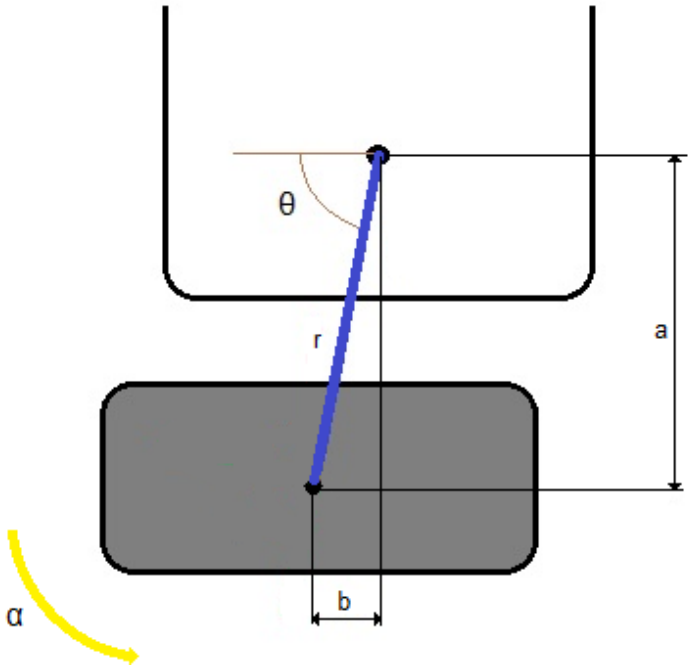


Figure 4.5: distances of the wheel.

This calculation is valid for the four wheels. The positive direction of the angle α is the opposite of the clockwise. It has been supposed that it turns this angle respecting to the centre of the robot.

For the simulation it was made a program in Matlab called **parallelmovement** (appendix 11.1). The inputs parameters of the program are three: the angle α, the lineal velocity of the robot in m/s, and the time of the simulation. The program plots two graphics: one with the trajectory, and other with the direction of the joystick for this trajectory. Below are showed some examples of the program.

It can be seen in the *figure 4.6* a straight movement backwards, with 0 degrees and the robot has a velocity of 1(m/s).

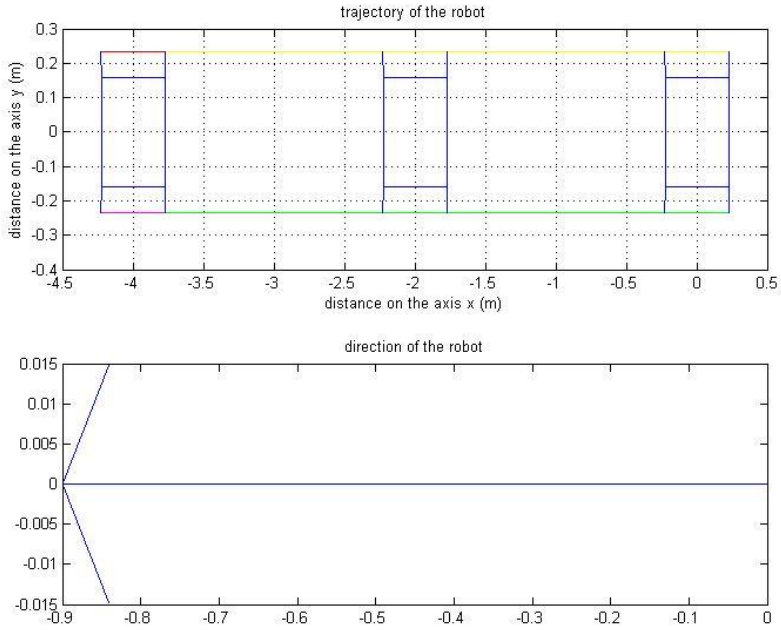


Figure 4.6: simulation of parallelmovement (0,-1, 4)

In the *figure 4.7* a linear movement with 60 degrees as the value of α for and a lineal velocity of 0.5 (m/s) can be seen.

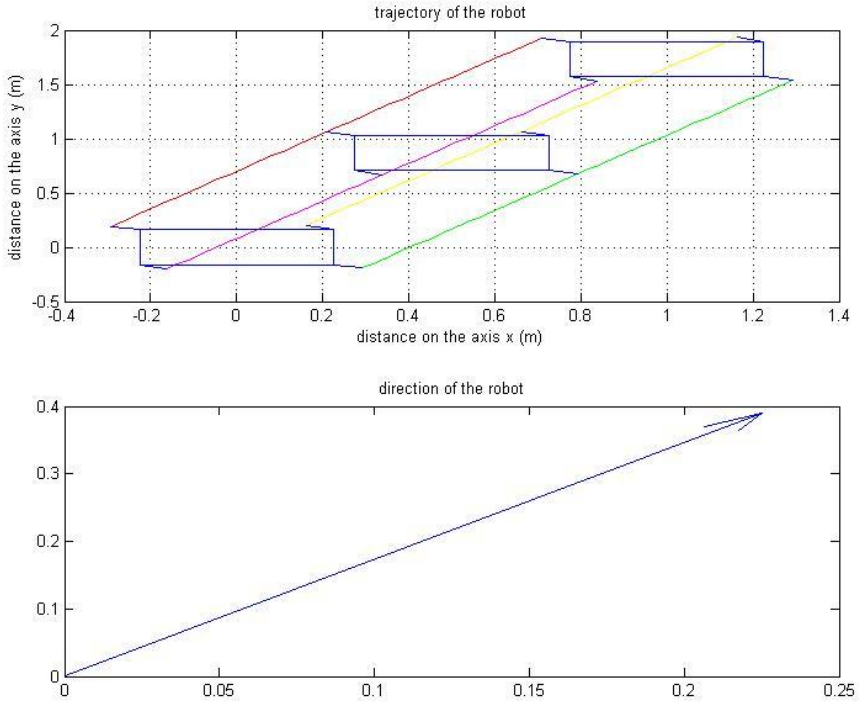


Figure 4.7: simulation of parallelmovement (60, 0.5, 4)

4.2. Turning with a radius of curvature.

The robot uses turning with a radius of curvature to make a turn with a determinate angle of curvature and a radius of curvature. The design of the robot gives some restrictions that can cause problem to follow the correct trajectory. For a correct trajectory all the wheels have to be orientated to the same instantaneous centre of rotation. As the wheels are join in pairs, two have always the same turn angle, so they can´t have a common instantaneous centre of rotation. It is used the program Matlab to simulate an approximate trajectory and to compare it with the real trajectory. This program made is called **turning** (Appendix 11.2)**.**

The centre of the robot is determinate as the global frame. After, it is chosen four new frames in each wheel. These frames are used to calculate the direction of the angular velocity. In the figure 4.8 the collocation of the frames can be seen.

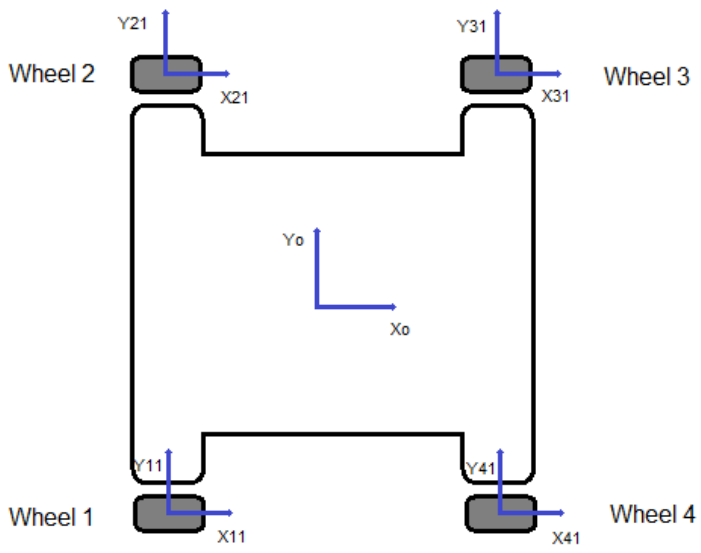


Figure 4.8: Positions of the frames

Below, the translation of the wheels frames is presented. The rotation matrix to change the coordinates from the wheel frame to the robot frame, and the correspondent figure to each system.

Wheel 1

The *equation 4.3* and the *equation 4.4* represent the translation of the wheel 1 frame, and the *equation 4.5* represents the rotation of the wheel 1 frame. It can

be seen the parameters of the equations represented in the *figure 4.9.* The direction of the angular velocity is represented in red.

[4.3] –

[4.4] –

[4.5]

Where:

θ=the initial rotation angle of the wheel frame. [Rad]

α12= the rotation angle of the wheels 1 and 2. [Rad]

l=distance between the vertical turn axis of the rear and the front wheels. [m]

h=distance between the vertical turn axis of the right and the left wheels. [m]

x11=horizontal position of the wheel 1 frame axis. [m]

y11=vertical position of the wheel 1 frame axis. [m]

T1=represents the rotation matrix of the wheel 1 frame.

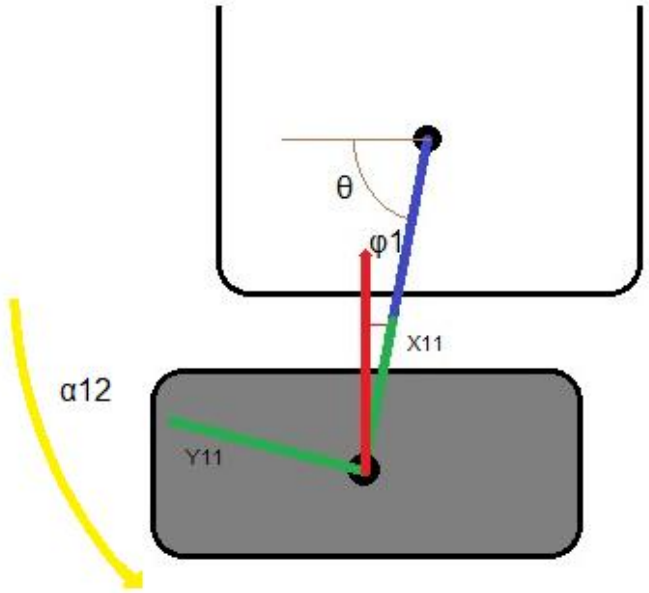


Figure 4.9: Wheel 1.

Wheel 2

The *equation 4.6* and the *equation 4.7* represent the translation of the wheel 2 frame, and the *equation 4.8* represents the rotation of the wheel 2 frame. It can be seen the parameters of the equations represented in the *figure 4.10.* The direction of the angular velocity is represented in red.

[4.6] –

[4.7] –

[4.8]

Where:

θ=the initial rotation angle of the wheel frame. [Rad]

α12= the rotation angle of the wheels 1 and 2. [Rad]

l=distance between the vertical turn axis of the rear and the front wheels. [m]

h=distance between the vertical turn axis of the right and the left wheels. [m]

x21=horizontal position of the wheel 2 frame axis. [m]

y21=vertical position of the wheel 2 frame axis. [m]

T2=represents the rotation matrix of the wheel 2 frame.

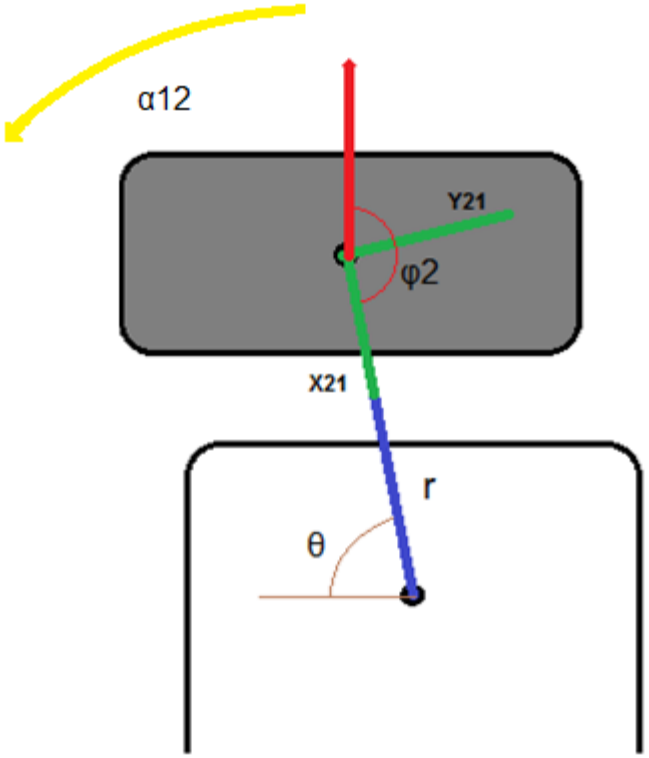


Figure 4.10: Wheel 2.

Wheel 3

The *equation 4.9* and the *equation 4.10* represent the translation of the wheel 3 frame, and the *equation 4.11* represents the rotation of the wheel 3 frame. The parameters of the equations are represented in the *figure 4.11.* The direction of the angular velocity is represented in red.

[4.9] –

[4.10] –

[4.11]

Where:

θ=the initial rotation angle of the wheel frame. [Rad]

α34= the rotation angle of the wheels 3 and 4. [Rad]

l=distance between the vertical turn axis of the rear and the front wheels. [m]

h=distance between the vertical turn axis of the right and the left wheels. [m]

x31=horizontal position of the wheel 3 frame axis. [m]

y31=vertical position of the wheel 3 frame axis. [m]

T3=represents the rotation matrix of the wheel 3 frame.

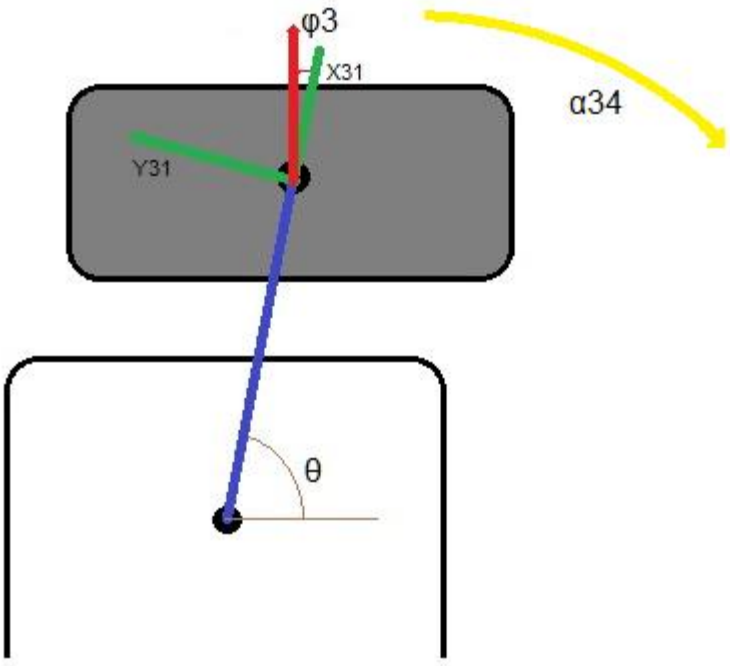


Figure 4.11: Wheel 3.

Wheel 4

The *equation 4.12* and the *equation 4.13* represent the translation of the wheel 4 frame, and the *equation 4.14* represents the rotation of the wheel 4 frame. The parameters of the equations are shown in the *figure 4.12.* The direction of the angular velocity is represented in red.

[4.12]

[4.13]

[4.14]

Where:

θ=the initial rotation angle of the wheel frame. [Rad]

α34= the rotation angle of the wheels 3 and 4. [Rad]

l=distance between the vertical turn axis of the rear and the front wheels. [m]

h=distance between the vertical turn axis of the right and the left wheels. [m]

x41=horizontal position of the wheel 4 frame axis. [m]

y41=vertical position of the wheel 4 frame axis. [m]

T4=represents the rotation matrix of the wheel 4 frame.

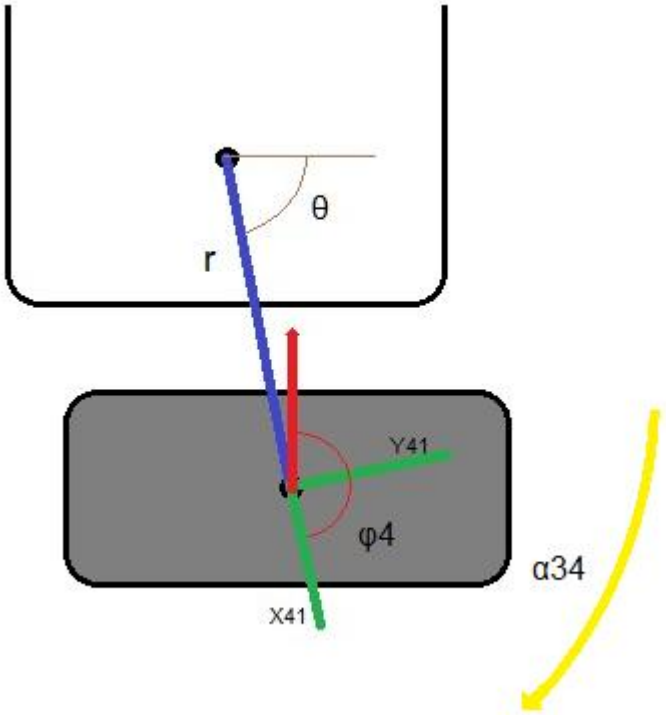


Figure 4.12: Wheel 4.

For the angles of curvature (α12, α34) the instantaneous centre of rotation has to be searched. In the case of this robot it´s always found two instantaneous centres of rotation, one for the right wheels (wheel 1 and wheel 4) and other for the left wheels (wheel 2 and wheel 3). In this step, it was decided to make an approximation. A new instantaneous centre of rotation for the entire robot is chosen. This point is given by getting the media of the other two points. Using the program **turning** it can be seen clearly some examples of this situation. In the figures are represented the line of the angular velocity and the instantaneous centres of rotation, and the new one calculated by the media. In the *figure 4.13* the ICR is shown, when both pairs of wheels have the same angle of rotation. In pink is showed the approximate ICR, in blue the real ICR for the left wheels in red the ICR for the right wheels.

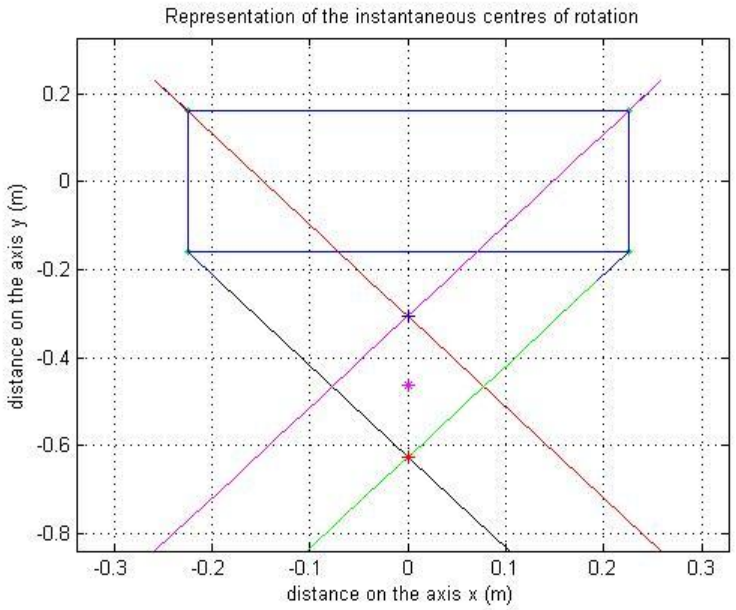


Figure 4.13: Representation of ICR for α12=α34= -0.45 Rad.

In the figure 4.14 ICR is shown, when α12 and α34 have a different value.

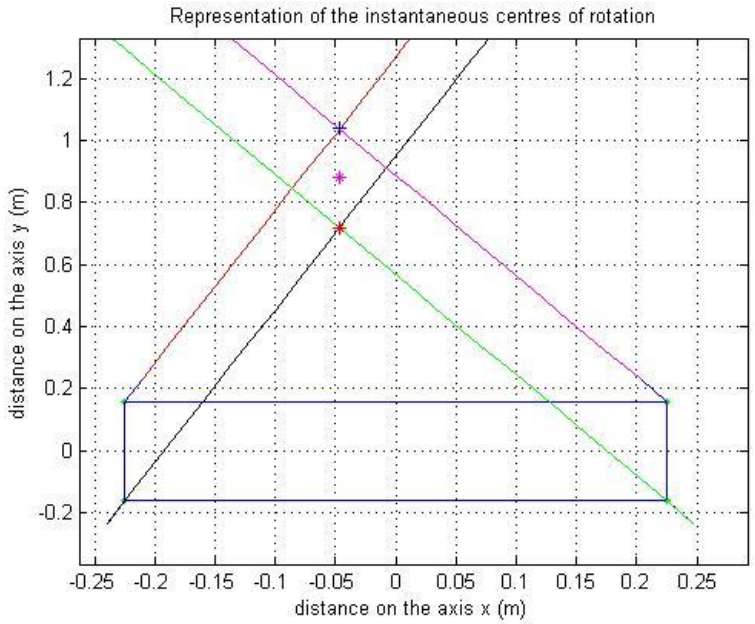


Figure 4.14: Representation of ICR for α12=0.2 and α34=0.3 Rad.

It is calculated a radius of rotation by the media of the radius from the 4 wheels to the new instantaneous centre of rotation. With this radius it can be calculated the angular velocity of the robot using the *equation 4.15*. After, this angular velocity is used to know the linear velocity of each wheel. For this calculation it is used the distance to the new instantaneous centre of rotation.

[4.15]

Where:

vl= the lineal velocity of the robot. [m]

R= global radio of the robot to the ICR. [m]

av= angular velocity of the robot. [Rad/s]

When all the velocities are calculated, the approximate trajectory of the robot can be represented. For these representations Matlab program **turning** has been used. In the *figure 4.15*, the trajectory for the same angles of rotation in each pairs of wheels is shown. The robot has also a lineal velocity of 0.5m/s and the simulation time is 4 seconds. In the *figure 4.16*, the trajectory for different values of the angles of rotation α12 and α34 is pictured. The robot has also a lineal velocity of 1 m/s and the simulation time is 4 seconds. The radius of the robot in this trajectory is R=0.6699 m.

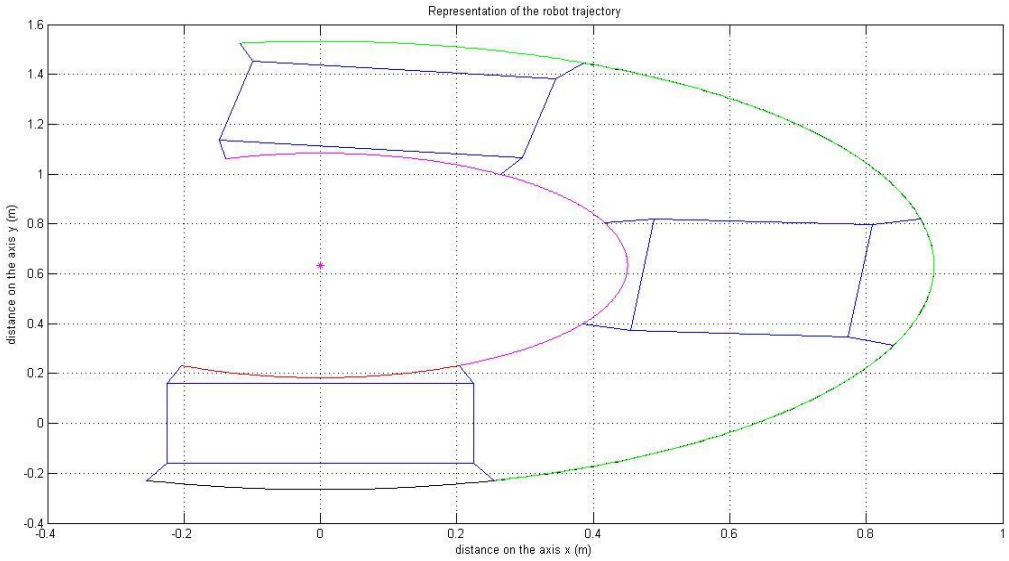


Figure 4.15: Trajectory for α12=α34=-0.35 Rad.

In the *figure 4.16*, the trajectory for different values of the angles of rotation α12 and α34 is plotted. The robot has also a lineal velocity of 1 m/s and the simulation time is 4 seconds. The radio of the robot in this trajectory is R=1.0275m.

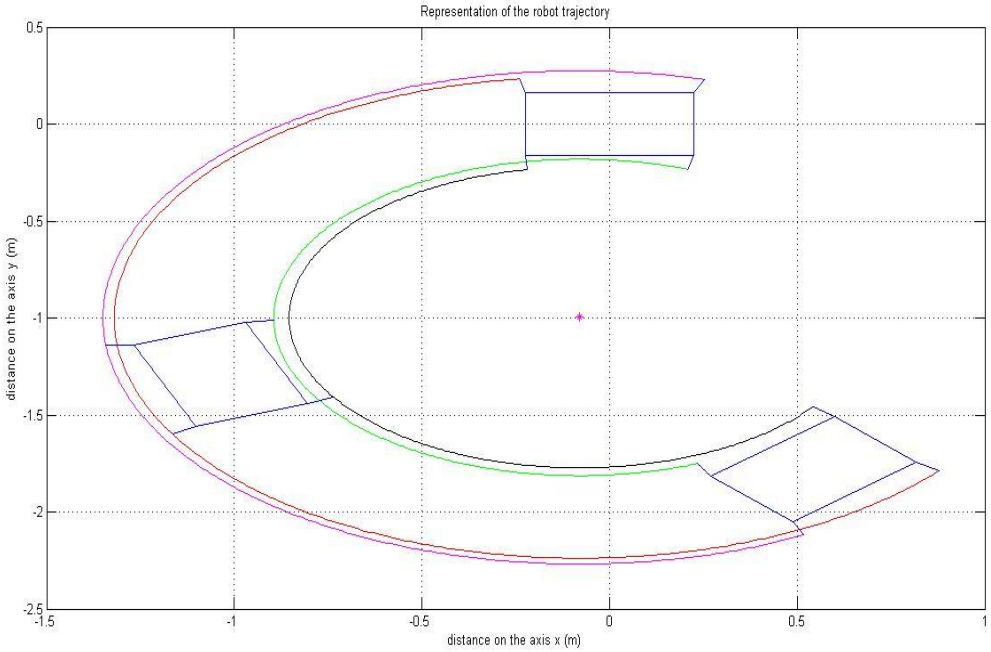


Figure 4.16: Trajectory for α12=0.15 and α34=0.3 Rad.

The previous figures represent approximate trajectories of the robot. Since the wheels don´t have the exactly same instantaneous centre of rotation, they will try to do their real trajectory. Using the program **turning** is testing the difference of the two velocity vectors. An example in the *figure 4.17* is seen.

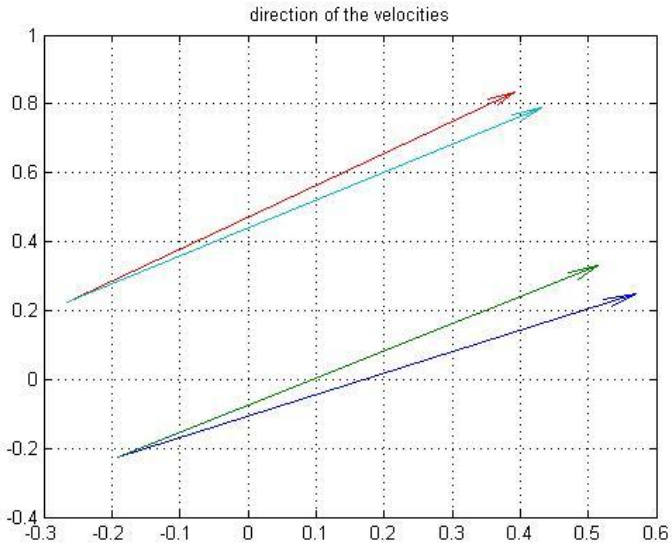


Figure 4.17: Velocity vectors for α12=α34=0.3 Rad.

The values of the angles between the two velocities are 6.38 and 3.8. But these angles can be corrected. In this simulation it wasn´t considered the slip of the wheels. The slip is produced due to lateral forced. These forces are perpendicular to the movement. They are caused by the deformation of the tire during the cornering. They produce a variation in the direction of the velocity. The angle between the two velocities is called slip angle. Using a study of the slip side it can be checked some approximate slip angles [4]. Using the Matlab program **friction_and_pacejka.m** [4] with the parameters of the outdoor robot, it shows that the slip angles have similar values to the angles between the real trajectory and the approximate. In the figure 4.18 the slip angle with respect to the radio of the trajectory is pictured. And so the angle will be adjusted trough the slip angle.

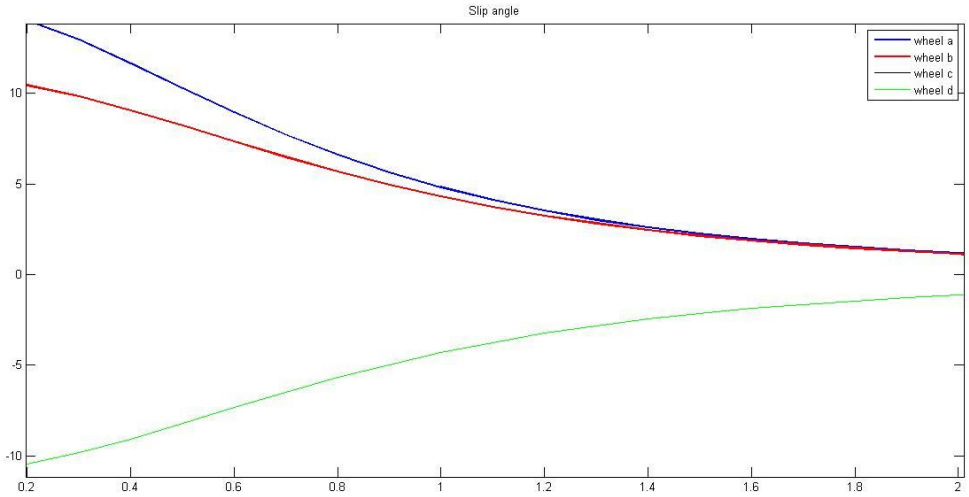


Figure 4.18: Slip angle [4]

6. Conclusion

The objective of this study is to know the possible trajectories of the outdoor robot. Due to the restrictions of the robot the movements are limited. First the movements were reduced from three to two since one restriction is that the wheels are join in parallel, so the robot can´t turn from its gravity centre.

To test better the trajectories, the mathematical program Matlab was used. Using this program it can be drawn all the possible trajectories.

A linear movement was calculated and described. The direction of the movement has to be chosen and given to the wheels. Then the robot follows forwards or backwards these directions. The robot can follow almost all the direction, it can´t turn the wheels 90 degrees, because the arms of the suspension hit the pulley that join the two wheels.

The robot can follow turning trajectory. To follow this trajectory it has to turn around the instantaneous centre of rotation. Due that the wheels are join in parallel it has two instantaneous centres of rotation, then it has to be calculated a new one for the robot. It is used the media between the two centres to make this calculation. The velocity is applied to the wheels from this new instantaneous centre of rotation. As the wheels will want to turn around their own instantaneous centre of rotation, the real velocity and the approximate velocity have a different direction. But for this approximate it wasn't consider the lateral forces. These forces will produce a deviation in the direction of the real velocity making it closer to the approximate velocity.

The robot will turn, but to make this movement has to slip. The slip will produced in the robot some vibrations but they will be absorbed by the suspension of the robot.

7. Next steps

The robot couldn´t be more developed due to some technical problems. New motors had to be ordered, because the old motors were damage. They didn´t arrive on time. The next students have to replace the motors and design a new piece to fix the motor to the robot, because the new motors have different size.

It can be chosen two ways of communication between the pc and all the controllers. One way is connect the port RS-232 between the pc and one EPOS controller, and from this EPOS controller pass the information to the CANopen to communicate the rest of the controller to the PC. The second way is to use a converter USB-CAN and connect all of them in a CANopen network. The second option is the best. In the first option the communication has to be executed with the worse characteristics of the CANopen and the RS-232 option. Then the velocity of the CANopen is lost, because the bit rate of the CANopen is 1Mbit/s and the bit rate of the RS-232 is 115.2 Kbit/s. Actually is being used the first option, it would be advantageous if the communication is changed to use only CANopen protocol.

Other new step is that they have to carry out the programming of the robot. Communication and the EPOS controllers should be programmed. The controllers can be set up in different modes; I recommend setting up in the velocity mode, because the development of the trajectories is made by the velocity. Then it has to be programming the trajectories developed in this project.

8. Bibliography

[1] EPOS 24-5 Hardware reference http://www.maxonmotor.ch (accessed on 25/10/2010)

[2] EPOS Application Note: CANopen Basic Information http://www.maxonmotor.ch (accessed on 25/10/2010)

[3] Zajac M., Stetter R., Paczynski A., Ucinski D.: Concept of control system for an innovative mobile robot chassis.

[4] Sara Yeni Gonzalez Endrinal: Modelling the wheels of the Robot MAX2D and surfacing.

9. List of the figures

10. List of the tables

11. Appendix

11.1 Program of the parallel movement

```
function parallelmovement(al,vl,t)
%linear velocity --> vl
%angle of the trajectory in degrees --> al
%time supposed in the simulation --> t
clc
% function that drawn the trajectory of the parallel movement.
%These parameters define the distance between the center of the wheel
%and the center of rotation with respect to the robot
a=0.075;
b=0.01;
r=(a^2+b^2)^(1/2);
th=atan(a/b);
%these are the lenghths of the robot
l=0.45;
h=0.32;
%we change the degrees to radians
al=al/180*pi;
%this bucle is used to restrict the angle of the trajectory to the
posible
%that the robot cans turn
if (al< -1.5)

    al=-1.5;

elseif (al>1.5)
    al=1.5;
end

%we allocate the frame in the center of the robot, then we use the
%lengths of the robot to allocate the wheels frames

%------------  wheel 1 ----------------%

vl1=[cos(al) ; sin(al)] ;   %direction of linear velocity

x1=-r*cos(th+al)-l/2;  %translation from the robot frame to the wheel
y1=-r*sin(th+al)-h/2;

%------------  wheel 2 ----------------%


vl2=[cos(al) ; sin(al)];   %direction of linear velocity

x2=-r*cos(th-al)-l/2;   %translation from the robot frame to the wheel
y2=r*sin(th-al)+h/2;

%------------  wheel 3 ----------------%
vl3=[cos(al);sin(al)];   %direction of linear velocity

x3=r*cos(th+al)+l/2; %translation from the robot frame to the wheel
y3=r*sin(th+al)+h/2;
```

```
%------------  wheel 4 ----------------%

vl4=[cos(al);sin(al)];   %direction of linear velocity

x4=r*cos(th-al)+l/2; %translation from the robot frame to the wheel
y4=-r*sin(th-al)-h/2;


T=0:0.1:t;          % counter to paint the trajectory


%plot of the trajectories
subplot(2,1,1)
%trajectory of the wheel 1
vlx1(1:1:(t*10+1))=T.*vl1(1)*vl+x1;
vly1(1:1:(t*10+1))=T.*vl1(2)*vl+y1;

plot(vlx1,vly1,'m')
hold on
grid on
title('trajectory of the robot')
xlabel('distance on the axis x (m)')
ylabel('distance on the axis y (m)')

%trajectory of the wheel 2
vlx2(1:1:(t*10+1))=T.*vl2(1)*vl+x2;
vly2(1:1:(t*10+1))=T.*vl2(2)*vl+y2;

plot(vlx2,vly2,'r')

%trajectory of the wheel 3
vlx3(1:1:(t*10+1))=T.*vl3(1)*vl+x3;
vly3(1:1:(t*10+1))=T.*vl3(2)*vl+y3;

plot(vlx3,vly3,'y')

%trajectory of the wheel 4
vlx4(1:1:(t*10+1))=T.*vl4(1)*vl+x4;
vly4(1:1:(t*10+1))=T.*vl4(2)*vl+y4;

plot(vlx4,vly4,'g')

%this bucle is used to paint the robot in different times of the
trajectory
for i=0:2:t
    line([-l/2+cos(al)*i*vl,-l/2+cos(al)*i*vl],[-
h/2+sin(al)*vl*i,h/2+sin(al)*i*vl])
    line([l/2+cos(al)*i*vl,-
l/2+cos(al)*i*vl],[h/2+sin(al)*vl*i,h/2+sin(al)*i*vl])
    line([l/2+cos(al)*i*vl,l/2+cos(al)*i*vl],[h/2+sin(al)*vl*i,-
h/2+sin(al)*i*vl])
    line([-l/2+cos(al)*i*vl,l/2+cos(al)*i*vl],[-h/2+sin(al)*vl*i,-
h/2+sin(al)*i*vl])
    line([-l/2+cos(al)*i*vl,x1+cos(al)*i*vl],[-
h/2+sin(al)*vl*i,y1+sin(al)*i*vl])
```

```
    line([-
l/2+cos(al)*i*vl,x2+cos(al)*i*vl],[h/2+sin(al)*vl*i,y2+sin(al)*i*vl])

line([l/2+cos(al)*i*vl,x3+cos(al)*i*vl],[h/2+sin(al)*vl*i,y3+sin(al)*i*vl
])
    line([l/2+cos(al)*i*vl,x4+cos(al)*i*vl],[-
h/2+sin(al)*vl*i,y4+sin(al)*i*vl])
 end

hold off



%this plot is used to paint a vector with the direction of the robot
%trajectory
subplot(2,1,2)
quiver(0,0,vl*cos(al),vl*sin(al))
title('direction of the robot')
hold off

% k=(0:0.05:vl);
% l=length(k);
% for i=1:l
%     tetha(i)=th;
% end
% polar(tetha,k)
```

11.2 Program turning

```
function turning(al12,al34,q,vl)
clc
%al12 is the angle of rotation for the wheels 1 and 2 in radians
%al34 is the angle of rotation for the wheels 3 and 4 in radians
%q is the simulation time in seconds
%vl is the lineal velocity of the robot in m/s
%These parameters define the distance between the center of the wheel
%and the center of rotation with respect to the robot
a=0.075;
b=0.005;

r=(a^2+b^2)^(1/2);
th=atan(a/b);

%these are the lengths of the robot
l=0.45;
h=0.32;

%variables used for the iterations
n=31;
n1=(n-1)/10;
T=0:0.1:n1;
```

```
%------------  wheel 1 ----------------%
phi1=pi/2-th;   %phi represents the angle of the direction of angular
                %acceleration

T1=[cos(th+al12) -sin(th+al12); sin(th+al12) cos(th+al12)]; %Base change
matrix
                                                    %robot frame and the
                                                    %wheel frame

va11=[cos(phi1) ; sin(phi1)] ; %direction of angular acceleration in the
                               %frame of the wheel


va10=T1*va11;       %direction of angular acceleration in the robot frame
x11=-r*cos(th+al12)-l/2;  %translation from the robot frame to the wheel
frame
y11=-r*sin(th+al12)-h/2;


%------------  wheel 2 ----------------%

phi2=pi/2+th;   %phi represents the angle of the direction of angular
                %acceleration

T2=[cos(-th+al12) -sin(-th+al12);sin(-th+al12) cos(-th+al12)]; %Base
change matrix
                                                      %robot frame and
the
                                                      %wheel frame

va21=[cos(phi2) ; sin(phi2)];   %direction of angular acceleration in the
                                %frame of the wheel

va20=T2*va21;        %direction of angular acceleration in the robot
frame
x21=-r*cos(th-al12)-l/2;   %translation from the robot frame to the wheel
frame
y21=r*sin(th-al12)+h/2;


%------------  wheel 3 ----------------%
phi3=pi/2-th;   %phi represents the angle of the direction of angular
                %acceleration

T3=[cos(th-al34) -sin(th-al34); sin(th-al34) cos(th-al34)]; %Base change
matrix
                                                    %robot frame and the
                                                    %wheel frame

va31=[cos(phi3);sin(phi3)];   %direction of angular acceleration in the
                              %frame of the wheel

va30=T3*va31;          %direction of angular acceleration in the robot
frame
x31=l/2+r*cos(th-al34); %translation from the robot frame to the wheel
frame
y31=h/2+r*sin(th-al34);
```

```
%------------  wheel 4 ----------------%
phi4=pi/2+th;   %phi represents the angle of the direction of angular
                %acceleration

T4=[cos(-th-al34) -sin(-th-al34); sin(-th-al34) cos(-th-al34)]; %Base
change matrix
                                                        %robot frame and
the
                                                         %wheel frame

va41=[cos(phi4);sin(phi4)];   %direction of angular acceleration in the
                              %frame of the wheel

va40=T4*va41;          %direction of angular acceleration in the robot
frame
x41=r*cos(th+al34)+l/2; %translation from the robot frame to the wheel
frame
y41=-r*sin(th+al34)-h/2;


%calculation instantaneous axis of rotation between the wheels 1 and 4
A1=[va10(2), -va10(1); va40(2), -va40(1)];
B1=[(va10(2)*x11-va10(1)*y11);(va40(2)*x41-va40(1)*y41)];

X1=(A1^-1)*B1;   %instantaneous axis of rotation 1-4


%calculation instantaneous axis of rotation between the wheels 2 and 3
A2=[va30(2), -va30(1); va20(2), -va20(1)];
B2=[(va30(2)*x31-va30(1)*y31);(va20(2)*x21-va20(1)*y21)];

X2=(A2^-1)*B2;   %instantaneous axis of rotation 2-3



% calculation of the new instantaneous centre of rotation
Rr=((X2(1)-x31)^2+(X2(2)-y31)^2)^(1/2);
Rl=((X1(1)-x11)^2+(X1(2)-y11)^2)^(1/2);

R=(Rr+Rl)/2                          %turn radio of the robot

ICR=[X1(1),X1(2)+0.1625]; %robot instantaneous centre of rotation


%this sentence change the direction of the angular velocity in case that
%the robot turn to the other side, it is used only in the graphics

if(al12>0 | al34>0)
   T=-T;
end


%This graphic represents the instantaneous center of rotation and the
%direction of the angular velocity.
```

```
subplot(2,1,1)

plot(l/2,h/2,'g .') %rotation point of the wheels with the robot
title('Representation of the instantaneous centre of rotation')
xlabel('distance on the axis x (m)')
ylabel('distance on the axis y (m)')

hold on
plot(-l/2,h/2,'g .')

plot(l/2,-h/2,'g .')

plot(-l/2,-h/2,'g .')


%representation of the robot
line([-l/2,-l/2],[-h/2,h/2])
line([-l/2,l/2],[h/2,h/2])
line([l/2,l/2],[h/2,-h/2])
line([l/2,-l/2],[-h/2,-h/2])
line([-l/2,x11],[-h/2,y11])
line([-l/2,x21],[h/2,y21])
line([l/2,x31],[h/2,y31])
line([l/2,x41],[-h/2,y41])


%Instantaneous centre of rotation
plot(ICR(1),ICR(2),'m *')
grid on

%representation of the angular velocity direction
vax1(1:1:n)=T.*va10(1)+x11;
vay1(1:1:n)=T.*va10(2)+y11;
plot(vax1,vay1,'k')

vax2(1:1:n)=T.*va20(1)+x21;
vay2(1:1:n)=T.*va20(2)+y21;

plot(vax2,vay2,'r')


vax3(1:1:n)=T.*va30(1)+x31;
vay3(1:1:n)=T.*va30(2)+y31;

plot(vax3,vay3,'m')

vax4(1:1:n)=T.*va40(1)+x41;
vay4(1:1:n)=T.*va40(2)+y41;

plot(vax4,vay4,'g')


plot(X1(1),X1(2),'r *')
plot(X2(1),X2(2),'b +')


hold off
```

```
R1=((ICR(1)-x11)^2+(ICR(2)-y11)^2)^(1/2);%new radio of the wheel 1
R2=((ICR(1)-x21)^2+(ICR(2)-y21)^2)^(1/2);%new radio of the wheel 2
R3=((ICR(1)-x31)^2+(ICR(2)-y31)^2)^(1/2);%new radio of the wheel 3
R4=((ICR(1)-x41)^2+(ICR(2)-y41)^2)^(1/2);%new radio of the wheel 4

%Radios calculated to draw the robot in different positions of the
%trajectory
R5=((ICR(1)+l/2)^2+(ICR(2)+h/2)^2)^(1/2);
R6=((ICR(1)+l/2)^2+(ICR(2)-h/2)^2)^(1/2);
R7=((ICR(1)-l/2)^2+(ICR(2)-h/2)^2)^(1/2);
R8=((ICR(1)-l/2)^2+(ICR(2)+h/2)^2)^(1/2);

%calculation of the angular velocity of the robot
av=vl/R;

%lineal velocities of the four wheels
vl1=av*R1;
vl2=av*R2;
vl3=av*R3;
vl4=av*R4;

%Velocities calculated to draw the robot in different positions of the
%trajectory
vl5=av*R5;
vl6=av*R6;
vl7=av*R7;
vl8=av*R8;



%In this sentence is calculated the angles that form the wheels with the
%ICR to  in the time 0
if(al12>0 | al34>0)

w1=pi-atan(abs(y11-ICR(2))/abs(x11-ICR(1)));
w2=pi-atan(abs(y21-ICR(2))/abs(x21-ICR(1)));
w3=atan(abs(y31-ICR(2))/abs(x31-ICR(1)));
w4=atan(abs(y41-ICR(2))/abs(x41-ICR(1)));
w5=pi-atan(abs(-h/2-ICR(2))/abs(-l/2-ICR(1)));
w6=pi-atan(abs(h/2-ICR(2))/abs(-l/2-ICR(1)));
w7=atan(abs(h/2-ICR(2))/abs(l/2-ICR(1)));
w8=atan(abs(-h/2-ICR(2))/abs(l/2-ICR(1)));

else
    w1=atan(abs(y11-ICR(2))/abs(x11-ICR(1)))+pi;
    w2=atan(abs(y21-ICR(2))/abs(x21-ICR(1)))+pi;
    w3=2*pi-atan(abs(y31-ICR(2))/abs(x31-ICR(1)));
    w4=2*pi-atan(abs(y41-ICR(2))/abs(x41-ICR(1)));
    w5=atan(abs(-h/2-ICR(2))/abs(-l/2-ICR(1)))+pi;
    w6=atan(abs(h/2-ICR(2))/abs(-l/2-ICR(1)))+pi;
    w7=2*pi-atan(abs(h/2-ICR(2))/abs(l/2-ICR(1)));
    w8=2*pi-atan(abs(-h/2-ICR(2))/abs(l/2-ICR(1)));
end
```

```
t=(0:0.01:q);  %This count represent the time of the simulation


subplot(2,1,2)

%these are the plots to draw the trajectories of the wheels
plot(ICR(1)+R1*cos(w1+av*t),ICR(2)+R1*sin(w1+av*t),'k')
hold on

plot(ICR(1)+R2*cos(w2+av*t),ICR(2)+R2*sin(w2+av*t),'r')
plot(ICR(1)+R3*cos(w3+av*t),ICR(2)+R3*sin(w3+av*t),'m')
plot(ICR(1)+R4*cos(w4+av*t),ICR(2)+R4*sin(w4+av*t),'g')
plot(ICR(1),ICR(2),'m *')

title('Representation of the robot trajectory')
xlabel('distance on the axis x (m)')
ylabel('distance on the axis y (m)')
grid on

%this bucle is used to draw the robot during the trajectory in different
%times

for i=0:2:q

    g1=[ICR(1)+R1*cos(w1+av*i),ICR(2)+R1*sin(w1+av*i)];
    g2=[ICR(1)+R2*cos(w2+av*i),ICR(2)+R2*sin(w2+av*i)];
    g3=[ICR(1)+R3*cos(w3+av*i),ICR(2)+R3*sin(w3+av*i)];
    g4=[ICR(1)+R4*cos(w4+av*i),ICR(2)+R4*sin(w4+av*i)];
    g5=[ICR(1)+R5*cos(w5+av*i),ICR(2)+R5*sin(w5+av*i)];
    g6=[ICR(1)+R6*cos(w6+av*i),ICR(2)+R6*sin(w6+av*i)];
    g7=[ICR(1)+R7*cos(w7+av*i),ICR(2)+R7*sin(w7+av*i)];
    g8=[ICR(1)+R8*cos(w8+av*i),ICR(2)+R8*sin(w8+av*i)];
    line([g1(1),g5(1)],[g1(2),g5(2)])
    line([g2(1),g6(1)],[g2(2),g6(2)])
    line([g3(1),g7(1)],[g3(2),g7(2)])
    line([g4(1),g8(1)],[g4(2),g8(2)])
    line([g5(1),g6(1)],[g5(2),g6(2)])
    line([g6(1),g7(1)],[g6(2),g7(2)])
    line([g7(1),g8(1)],[g7(2),g8(2)])
    line([g8(1),g5(1)],[g8(2),g5(2)])

end

hold off



%these are the calculation of the velocity vectors

cp1=[ICR(1)+R1*cos(w1),ICR(2)+R1*sin(w1)];

dir11=atan((X1(2)-pc(2)/(X1(1)-pc(1))));
dir12=atan((ICR(2)-pc(2)/(ICR(1)-pc(1))));
slipangle1=(dir1-dir2)*180/pi

cp2=[ICR(1)+R2*cos(w2),ICR(2)+R2*sin(w2)]
```

```
dir21=atan((X2(2)-pc2(2)/(X2(1)-pc2(1))))+pi/2;
dir22=atan((ICR(2)-pc2(2)/(ICR(1)-pc2(1))))+pi/2;
slipangle2=(dir12-dir22)*180/pi

figure(2)
%this plot show the velocity vectors of the right and the left wheels
subplot(1,1,1)
quiver(pc(1),pc(2),cos(dir1),sin(dir1))
hold on
grid on
title('direction of the velocities')
quiver(pc(1),pc(2),cos(dir2),sin(dir2))
quiver(pc2(1),pc2(2),cos(dir12),sin(dir12))
quiver(pc2(1),pc2(2),cos(dir22),sin(dir22))
```