ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Σχεδιασμός Κινηματικού Μοντέλου με Ολίσθηση για Ρομπότ με 4 Τροχούς Mecanum και Διάγνωση Βλάβης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΕΛΕΝΑΣ Α. ΟΙΚΟΝΟΜΟΥ

Επιβλέπων:

Κωνσταντίνος Ι. Κυριακόπουλος
Καθηγητής Ε.Μ.Π.

ΕΡΓΑΣΤΗΡΙΟ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Αθήνα, Οκτώβριος 2019

This page is intentionally left blank.

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ
ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Σχεδιασμός Κινηματικού Μοντέλου με Ολίσθηση για Ρομπότ με 4 Τροχούς Mecanum και Διάγνωση Βλάβης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΕΛΕΝΑΣ Α. ΟΙΚΟΝΟΜΟΥ

Επιβλέπων: Κωνσταντίνος Ι. Κυριακόπουλος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 2^{α} Οκτωβρίου 2019.

……	……	……
Κωνσταντίνος Ι. Κυριακόπουλος Καθηγητής Ε.Μ.Π.	Ευάγγελος Παπαδόπουλος Καθηγητής Ε.Μ.Π.	Γεώργιος-Χ. Βοσνιάκος Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

…….

ΕΛΕΝΑ Α. ΟΙΚΟΝΟΜΟΥ

Διπλωματούχος Μηχανολόγος Μηχανικός Ε.Μ.Π.

NATIONAL TECHNICAL UNIVERSITY OF ATHENS

SCHOOL OF MECHANICAL ENGINEERING

MECHANICAL DESIGN & CONTROL SYSTEMS DIVISION
CONTROL SYSTEMS LAB

DIPLOMA THESIS

Design of a kinematic model with slip for an omni-directional mobile robot with 4 mecanum wheels along with a fault detection scheme

Author

ELENA A. OIKONOMOU

Supervisor:

Konstantinos J. Kyriakopoulos
Professor of Robotics, Mechatronics & Control

October 2019

This page is intentionally left blank.

Περίληψη

Σκοπός της παρούσας εργασίας είναι η ανάλυση του προβλήματος της ολίσθησης που παρουσιάζουν τα τροχοφόρα ρομπότ παντός διεύθυνσης κίνησης τα οποία διαθέτουν τροχούς mecanum. Η πιο διαδεδομένη μέθοδος εκτίμησης της θέσης μίας τροχοφόρας ρομποτικής διάταξης περιλαμβάνει τη χρήση encoders. Η ολίσθηση όμως των τροχών μπορεί να επηρεάσει σημαντικά την επίδοση των encoders που είναι τοποθετημένοι στους άξονες των τροχών των οχημάτων αυτών με αποτέλεσμα να επηρεάζεται και η συνολική αξιοπιστία του ρομπότ, μια και αυτό δεν θα μπορεί να υπολογίσει με ακρίβεια την θέση του στον χώρο. Επομένως, προκειμένου να αντιμετωπιστεί το πρόβλημα αυτό, αναπτύξαμε ένα κινηματικό μοντέλο το οποίο λαμβάνει υπόψιν του την ολίσθηση των τροχών με το έδαφος.

Αναπτύξαμε επίσης ένα σύστημα ανίχνευσης και διάγνωσης βλαβών, βασισμένο στο μαθηματικό μοντέλο του συστήματος, για τροχοφόρα ρομπότ παντός διεύθυνσης κίνησης τα οποία διαθέτουν τέσσερις τροχούς mecanum. Η βασική ιδέα πίσω από το προτεινόμενο σύστημα είναι η χρήση τεχνικών οι οποίες είναι βασισμένες στην δομική ανάλυση του συστήματος προκειμένου να δημιουργήσουμε υπόλοιπα, τα οποία ορίζονται ως η διαφορά της πραγματικής τιμής μίας μεταβλητής από την αναμενόμενη, η οποία αναμενόμενη τιμή έχει προκύψει από το μαθηματικό μοντέλο του συστήματος. Το πλεονέκτημα της μεθοδολογίας αυτής, έγκειται στο γεγονός ότι αποτελεί μία απλή μέθοδο, η οποία μπορεί να εφαρμοστεί για τη δημιουργία υπολοίπων σε μη-γραμμικά συστήματα, χρησιμοποιώντας μετρήσεις μόνο της θέσης και του προσανατολισμού της ρομποτικής πλατφόρμας. Με την μεθοδολογία αυτή, το ρομπότ είναι σε θέση να ανιχνεύσει την ύπαρξη βλαβών στους κινητήρες των τροχών του αλλά και να αναγνωρίσει ποιος ή ποιοι από τους τέσσερις τροχούς έχουν υποστεί τη βλάβη.

Στη συνέχεια, σχεδιάσαμε ένα σχήμα ελέγχου ανεκτικό σε βλάβες, το οποίο είναι βασισμένο στη φιλοσοφία των Συναρτήσεων Πλοήγησης. Με το σχήμα αυτό, η ρομποτική διάταξη είναι σε θέση να επιτύχει οποιαδήποτε επιθυμητική θέση και προσανατολισμό ορίσουμε, αποφεύγοντας ταυτόχρονα συγκρούσεις με στατικά εμπόδια που υπάρχουν στον χώρο εργασίας. Ο ελεγκτής αυτός είναι ικανός να οδηγήσει το ρομπότ αποτελεσματικά ακόμα και στην περίπτωση όπου υπάρχει μία ή και δύο βλάβες στους τροχούς του.

Τέλος, πραγματοποιήθηκαν εκτενείς προσομοιώσεις αλλά και πειράματα στο εργαστήριο με την ρομποτική πλατφόρμα παντός διεύθυνσης κίνησης youBot της εταιρίας KUKA, τα οποία επιβεβαιώνουν τα αποτελέσματα της παρούσας διπλωματικής εργασίας.

Λέξεις-κλειδιά: *κινηματικό μοντέλο με ολίσθηση, διάγνωση βλάβης, ανίχνευση και αναγνώριση βλάβης, δομική ανάλυση, έλεγχος ανεκτικός σε βλάβες, πλατφόρμα παντός διεύθυνσης*

This page is intentionally left blank.

Abstract

In the present thesis, we investigate the problem of wheel slippage on omni-directional mobile robots. Wheel slip affects the performance of the rotary shaft encoders that are mounted on the vehicle's wheels and as a result the accuracy of the robot since encoders are the most popular dead-reckoning method that is used heavily in almost any application involving mobile robots. Therefore, in order to overcome this limitation, we implement a kinematic model with slip, which takes into account the slippage of the wheels.

We also introduce a model-based actuator fault diagnosis system for omni-directional mobile robots with 4 mecanum wheels. The idea behind the proposed method is to use structural analysis-based techniques in order to generate residuals. The advantage of the proposed method is that it is a simple method that can offer feasible solutions to residual generation for nonlinear systems by only utilizing measurements of the robot's position and orientation. Through this proposed scheme, we are able to detect actuator faults in one or two wheels and identify the wheels that the fault has occurred on.

Subsequently, we implemented a fault tolerant control scheme which incorporates Navigation Functions in order for the robotic platform to achieve any desired configuration while avoiding collisions with static obstacles. This control scheme is capable of compensating up to two faulty wheels.

To validate our work, an extensive simulation and experimental procedure was carried out using the omni-directional mobile platform youBot by KUKA.

Keywords: *kinematic modeling with slip, fault diagnosis, fault detection and isolation, structural analysis, fault-tolerant control, omni-directional mobile robots*

This page is intentionally left blank.

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Κωνσταντίνο Κυριακόπουλο που μου πρόσφερε την ευκαιρία να αποτελέσω μέλος της ερευνητικής του ομάδας και να εργαστώ υπό την επίβλεψή του σε ένα τόσο καινοτόμο εργαστήριο όπως το Control Systems Lab (CSL). Οι γνώσεις και η εμπειρία που απέκτησα κατά την εκπόνηση αυτής της εργασίας είναι εξαιρετικά διδακτικές και χρήσιμες.

Ακόμη, θα ήθελα να ευχαριστήσω θερμότατα τον μετα-διδακτορικό ερευνητή Χαράλαμπο Μπεχλιούλη για την συνολική επίβλεψη της παρούσας διπλωματικής, την καθοριστική βοήθειά του αλλά και τις πολύτιμες συμβουλές του. Ευχαριστώ πολύ.

Θα ήθελα επίσης να εκφράσω τις ευχαριστίες μου προς όλα τα μέλη του εργαστηρίου CSL για το ευχάριστο και φιλικό κλίμα που δημιούργησαν αλλά για και την προθυμία τους να βοηθήσουν οποιαδήποτε στιγμή.

Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου για όλα όσα μου έχουν προσφέρει αλλά και όλους εκείνους που έχουν συντελέσει στο να γίνω καλύτερος άνθρωπος.

This page is intentionally left blank.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Omni-directional Mobile Robots

Robotic vehicles are extensively used in industrial, domestic and technical applications and they are continuously gaining in importance. They are used for surveillance, inspection and transportation tasks. Mobile robotic platforms can also be found in service robots, wheel chairs, other mobile devices that aim at facilitating human physical disabilities, vehicles used in hazardous environments or in large automated warehouses for supervision, inspection and handling of materials.

One of the main requirements of an autonomous mobile robot is its ability to move through the operational space, avoiding obstacles and finding its way to the next location, in order to perform its task. However, it appears that in order to be competent in fulfilling fruitful work in the limited and possibly occluded space of the aforementioned fields, such vehicular robots should be versatile enough to move quickly and accurately in any direction. In the past few years, the academic community has extensively explored the development of such mobile platforms.

In order to move in tight areas and to avoid obstacles mobile robots should have good mobility and maneuverability. These capabilities mainly depend on the wheels design. Wheeled mobile robots employ either conventional wheels, like the unicycle wheel (Figure 1-1), that do not allow sideways sliding, or wheels that allow sideways sliding through the use of rollers around the rim of the wheel, such as the mecanum wheel (Figure 1-2).

The Mecanum wheel was invented in 1973 by a Swedish engineer, named Bengt Erland Ilon while working at the Swedish company Mecanum AB. This is why this type of wheel is also called Mecanum, Ilon or Swedish wheel. Using four of these wheels provides omni-directional movement for a vehicle without needing a conventional steering. The first mobile robot with Mecanum wheels, named “Uranus”, was designed and constructed in Carnegie Mellon University in 1985 (Figure 1-3).

A mecanum wheel is a conventional wheel with a series of rollers attached to its circumference. These rollers typically each have an axis of rotation at 45° to the plane of the wheel and at 45° to a line through the center of the roller parallel to the axis of rotation of the wheel.

Fig. 1-1 Unicycle wheel

Fig. 1-2 Mecanum wheel

Fig. 1-3 "Uranus" mobile robot

By alternating wheels with left and right-handed rollers, in such a way that each wheel applies force roughly at right angles to the wheelbase diagonal the wheel is on, the vehicle is stable and can be made to move in any direction and turn by varying the speed and direction of rotation of each wheel. Moving all four wheels in the same direction causes forward or backward movement. Running the wheels on one side in the opposite direction to those on the other side causes rotation of the vehicle, and running the wheels on one diagonal in the opposite direction to those on the other diagonal causes sideways movement. Combinations of these wheel motions allow for vehicle motion in any direction with any vehicle rotation, including no rotation at all.

For the past few years, omni-directional mobile robots have gained popularity and are now widely used in industry due to their enhanced motion capabilities and they are expected to be deployed even more widely in the future (Figure 1-4).

The term of omni-directional is used to describe the ability of a system to move instantaneously in any direction from any configuration. Robotic vehicles are often designed for planar motion. In such a two-dimensional space, a body has three degrees of freedom. It is capable of translating in both directions and rotating about its center of gravity. Most conventional vehicles however do not have the capability to control every degree of freedom independently. Conventional wheels are not capable of moving in a direction parallel to their axis. This so called non-holonomic constraint of the wheel prevents vehicles using skid-steering, like a car, from moving perpendicular to its drive direction. While it can generally reach every location and orientation in a 2D space, it can require complicated maneuvers and complex path planning to do so. On the contrary, mobile robots with mecanum wheels can perform rotation and side-translation simultaneously.

Fig. 1-4 Omni-directional mobile robots

1.2 Objective

Torque is the twisting force that the motors on the wheels of a vehicle produce. In low-traction situations, where traction is defined as the maximum amount of force that the wheel can apply against the ground, the maximum amount of torque that can be created is only determined by the amount of traction and not by the motors. If the wheels won't stick to the ground there is simply no way to harness enough power. Below are the three factors that affect traction:

- *The weight on the wheels*. The more weight on a wheel, the more traction it has.
- *The coefficient of friction*. This factor relates the amount of friction force between two surfaces to the force holding the two surfaces together. In our case, it relates the amount of traction between the wheels and the ground, to the weight resting on each wheel. The coefficient of friction is mostly a function of the material of the wheel and the type of surface the vehicle is driving on.
- *Wheel slip*. Wheel slip occurs when the force applied to a wheel exceeds the traction available to that wheel.

Slip is the relative motion between a wheel and the ground surface it is moving on. Wheel slip is inevitable when a Wheeled Mobile Robot (WMR) is moving at a high speed or on a slippery surface. Research on omni-directional mobile robots has shown that a vehicle with mecanum wheels is susceptible to slippage, and as a result, the most popular dead-reckoning method, which includes the usage of rotary shaft encoders, is prevented from being performed well on a vehicle with mecanum wheels. This phenomenon however, can greatly affect the accuracy of mobile robots, since they might not be able to estimate their position efficiently, which could result in a failure to accomplish their missions.

This study aims to address the impact of wheel slip on omni-directional mobile robots and a kinematic modeling that takes wheel slip into consideration has been proposed. Additionally, a fault diagnosis system has been developed in order to detect and identify actuator faults on the four wheels of the omni-directional platform. To surmount the problems caused by wheel slip and actuator faults, a fault tolerant control scheme has been developed and implemented in order to navigate the robot successfully towards the goal configuration even in the presence of slip and faulty wheels.

1.3 Literature review

Several previous studies in the broader literature have examined the problem of wheel slip in mobile robots [37], [38]. However, most of them have almost exclusively focused on unicycle-type wheels. Their findings, although significant, do not apply on mecanum wheels that omni-directional mobile robots employ.

Some authors have suggested the use of position rectification controllers in order to overcome the problems of wheel slip in omni-directional robots. The authors of [36], have implemented a control scheme which uses symptomatic and preventive rectification methods with multi-sampling periods. Despite the fact that position rectification controllers are an efficient solution, they do not address the modeling of slip and thus introduce a more complicated and computationally expensive methodology.

A closer look to the literature on position errors due to wheel slip, reveals some studies that introduced wheel parameters adjustments in order to reduce those errors. In [39] and [40], the authors propose some simple models by deriving adjustment equations to improve the performance of the robot and eradicate position errors. These approaches though, constitute a calibration method and as such they cannot be consider for online usage in unknown environments.

Significant research related with the slip modeling of wheels appears in the works of [2]. In particular, they present a kinematic modeling method, through a generic wheel equation, which is based on recursive kinematics formulations. This generic equation is later particularized for fixed, orientable, castor and mecanum types of wheels. They produce a sliding velocity vector for the mecanum wheel which depends on some geometric data that are readily available for any robotic platform as well as the robot's body, wheel and roller velocities.

In [1], the same authors propose a kinematic modeling method with slip for wheeled mobile robots, including omni-directional robots, based on physical principles. In order to provide a proper physical sense to the kinematic models with slip, they derived the dynamics of the mobile robots and applied three successively approximations consisting of the quasi-static motion model, the slip kinematic model and the weighted least-squares solution. In their first model, the quasi-static motion model, they regard the dynamics of the system with null accelerations. In the second one, they neglect the centripetal force, as is possible for smooth maneuvering, where the robot's linear and angular velocities do not have high values at the same time. In their third model, the weighted least-squares solution, they apply a weighted matrix of friction coefficients and consider the friction forces linearly dependent on the sliding velocities while also neglecting the centripetal forces.

These three models however, make a lot of approximations that are not valid in a random motion that typically wheeled mobile robots execute. In this thesis, we propose a kinematic model with slip which is based on the work of the authors in [1], where we consider and develop the full model with slip with no approximations.

Significant prior research has also been made in the field of structural analysis [3], [4], [5], [6]. In [5] and [6] the authors discuss how structural analysis techniques are applied to an unmanned ground vehicle for residual generation. In [4], the authors have implemented a structural analysis technique on a four-wheel skid steering mobile robot in order to identify actuator faults. However, no previous research has investigated the fault detection and identification of actuators of an omni-directional mobile robot with 4 mecanum wheels. The research efforts in [5], [6] and [11], [12], [14] are primarily intended to detect faults in the sensors of a wheeled robot or they do not specify the matching used to identify the faults.

In [7], a fault detection technique has been efficiently employed in an omni-directional mobile robot by examining the current measurements of the motors in order to diagnose a fault. There are some cases though, where we are unable to obtain these measurements (such as an absence of a sensor or a malfunction).

In this work, we propose a model-based actuator fault diagnosis for an omni-directional mobile robot with 4 mecanum wheels. We consider actuation faults in which the wheels are not able to receive commands, but still can rotate freely due to the friction with the ground. We assume that only the measurements of the position and the heading angle of the robot are available.

1.4 Thesis Outline

The thesis is structured as follows:

Chapter 2 presents the kinematic modeling with slip for an omni-directional mobile robot with 4 mecanum wheels. We derive the mathematical equations for the kinematics and the dynamics of the platform as well as the equations for the slipping conditions.

Chapter 3 introduces a model-based actuator fault diagnosis system for an omni-directional mobile robot that is capable of identifying actuator faults in one or two wheels by employing structural analysis techniques.

Chapter 4 consists of a fault tolerant control scheme that incorporates Navigation Functions in order for the robotic platform to achieve any desired configuration while avoiding collisions with static obstacles. The control scheme is capable of compensating up to two faulty wheels.

Chapter 5 presents the simulation results and showcases the experimental studies performed on the mobile robot youBot by KUKA that support and validate our research.

Chapter 6 concludes the work and suggests possible future extensions.

Chapter 2

Kinematic Modeling with Slip

2.1 Conventional kinematics

Kinematics is a branch of classical mechanics that describes the motion of objects without considering the forces that cause them to move. The conventional kinematic equations of a 4 mecanum wheeled mobile robot are presented in this section.

First, we will introduce some terminology. Assuming horizontal movement, the position of the Wheeled Mobile Robot's (WMR) body is completely specified by three scalar variables (x, y, θ), usually referred to as WMR posture, in vector form. We define two reference frames:

- Global (Inertial) frame G
- Robot body-fixed frame R

The state vector/WMR posture with respect to the global frame is:

$$P = [x \quad y \quad \theta]^T \tag{2.1}$$

Its first-order time derivative $\dot{P}$ is called WMR velocity vector. The velocities of the vehicle's center of mass expressed in the body frame {R} attached on it are:

$$\dot{P} = [u_x \quad u_y \quad \omega]^T \tag{2.2}$$

Similarly, for each wheel, wheel velocity vector and wheel velocities are defined. We denote the rotational velocity of the wheels by $\dot{\varphi}_i$, i = 1, …, 4, the radius of the wheel by r, the longitudinal and lateral distance of the wheels to the center of mass by L, l respectively.

The geometry of the platform we study, illustrating an omni-directional mobile robot, is shown in Figure 2-1.

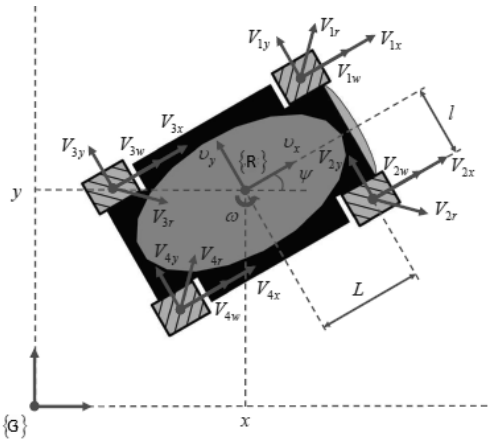

Fig. 2-1 Geometry of the omni-directional mobile platform with 4 mecanum wheels

The forward kinematics of the platform is given by:

$$\begin{bmatrix} u_x \\ u_y \\ \omega \end{bmatrix} = J_V \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{bmatrix} \tag{2.3}$$

where J_v is the Jacobian matrix:

$$J_V = \frac{r}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\frac{1}{L+l} & \frac{1}{L+l} & -\frac{1}{L+l} & \frac{1}{L+l} \end{bmatrix} \tag{2.4}$$

The robot's body velocities in the global frame are given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = J_I \begin{bmatrix} u_x \\ u_y \\ \omega \end{bmatrix} \tag{2.5}$$

where the rotation matrix J_I is:

$$J_I = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

Even though these kinematic equations are widely used, there is one major drawback that could possibly impact performance in certain occasions. This drawback is that there is an underlying assumption that there is not wheel slip present. Therefore, in order to overcome this limitation, we need to develop a model that takes wheel slippage into account.

2.2 Wheel kinematic equations

Toward this direction, of developing a model that takes wheel slippage into account, we define the wheel kinematic equations that correspond to the sliding linear velocity between the rollers and the ground in the coordinate frame Ei.

But before we begin constructing the wheel kinematic equations, we will define the following variables:

- L_i Frame attached to wheel i with the x-axis coincident with the wheel rotation axle
- E_i Frame attached to roller of wheel i with the x-axis coincident with its rotation axle
- ${}^R\dot{P}$ WMR velocity vector in coordinate frame R
- l_i Distance between center of wheel and origin of robot frame R
- $\dot{\beta}_i$ Angular velocity of the steering link with respect to the WMR
- α_i Angle between distance vector and x-axis of robot frame R
- γ_i Roller angle with respect to the axis of rotation of the wheel ($\pm$45°)
- $(\dot{\varphi}_i,\ \dot{\varphi}_{ri})$ Rotation velocity of the wheel and the rollers in coordinate x of frames Li and Ei
- $(r_i,\ r_{ri})$ Wheel radius and roller radius

The frames attached to the mecanum wheel and the robot body along with some of the defined variables are illustrated in Figure 2-2:

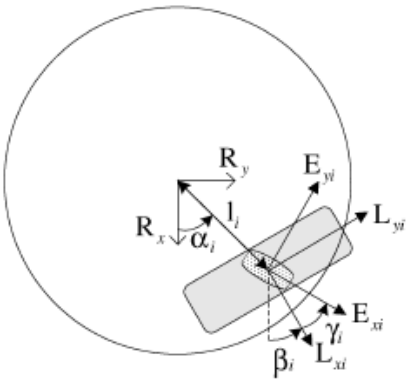

Fig. 2-2 Mecanum wheel attached to robot body

The wheel kinematic equations that depict the sliding velocity of the wheel in the coordinate frame Ei are:

$$v_{slip\,i} = \begin{pmatrix} \cos(\beta_i+\gamma_i) & \sin(\beta_i+\gamma_i) & l_i\sin(\beta_i+\gamma_i-\alpha_i) & r_i\sin(\gamma_i) & 0 \\ -\sin(\beta_i+\gamma_i) & \cos(\beta_i+\gamma_i) & l_i\cos(\beta_i+\gamma_i-\alpha_i) & r_i\cos(\gamma_i) & r_{ri} \end{pmatrix} \times \begin{pmatrix} {}^R\dot{P} \\ \dot{\varphi}_i \\ \dot{\varphi}_{ri} \end{pmatrix} \qquad (2.7)$$

Once the wheel equations are established, a compound global kinematic equation for the WMR, regarding its sliding velocities, may be defined. The compound global WMR kinematic equation for an omni-directional mobile robot with four mecanum wheels is given by:

$$v_{slip} = \begin{pmatrix} v_{slip\ 1} \\ \vdots \\ v_{slip\ N} \end{pmatrix} = \begin{pmatrix} A_{p1} & A_{w1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{pN} & 0 & \dots & A_{wN} \end{pmatrix} \begin{pmatrix} {}^{R}\dot{P} \\ \dot{q}_{w1} \\ \vdots \\ \dot{q}_{wN} \end{pmatrix} = \begin{pmatrix} A_p & A_w \end{pmatrix} \begin{pmatrix} \dot{P} \\ \dot{q}_w \end{pmatrix} = A \cdot {}^{R}\dot{q} \qquad (2.8)$$

where:

- N Number of wheels of the WMR
- $\dot{q}_{wi}$ Vector with all the velocities of the wheel i
- $\dot{q}_w$ Vector of all the wheel velocities
- ${}^{R}\dot{q}$ Vector of all velocities

The slip kinematic equation for the WMR result in:

$$v_{slip} = \begin{pmatrix} v_{slip\ 1} \\ v_{slip\ 2} \\ v_{slip\ 3} \\ v_{slip\ 4} \end{pmatrix} =$$

$$= \begin{bmatrix} \cos(\beta_1+\gamma_1) & \sin(\beta_1+\gamma_1) & l_1\sin(\beta_1+\gamma_1-\alpha_1) & r_1\sin(\gamma_1) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\sin(\beta_1+\gamma_1) & \cos(\beta_1+\gamma_1) & l_1\cos(\beta_1+\gamma_1-\alpha_1) & r_1\cos(\gamma_1) & r_{r1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \cos(\beta_2+\gamma_2) & \sin(\beta_2+\gamma_2) & l_2\sin(\beta_2+\gamma_2-\alpha_2) & 0 & 0 & r_2\sin(\gamma_2) & 0 & 0 & 0 & 0 & 0 \\ -\sin(\beta_2+\gamma_2) & \cos(\beta_2+\gamma_2) & l_2\cos(\beta_2+\gamma_2-\alpha_2) & 0 & 0 & r_2\cos(\gamma_2) & r_{r2} & 0 & 0 & 0 & 0 \\ \cos(\beta_3+\gamma_3) & \sin(\beta_3+\gamma_3) & l_3\sin(\beta_3+\gamma_3-\alpha_3) & 0 & 0 & 0 & 0 & r_3\sin(\gamma_3) & 0 & 0 & 0 \\ -\sin(\beta_3+\gamma_3) & \cos(\beta_3+\gamma_3) & l_3\cos(\beta_3+\gamma_3-\alpha_3) & 0 & 0 & 0 & 0 & r_3\cos(\gamma_3) & r_{r3} & 0 & 0 \\ \cos(\beta_4+\gamma_4) & \sin(\beta_4+\gamma_4) & l_4\sin(\beta_4+\gamma_4-\alpha_4) & 0 & 0 & 0 & 0 & 0 & 0 & r_4\sin(\gamma_4) & 0 \\ -\sin(\beta_4+\gamma_4) & \cos(\beta_4+\gamma_4) & l_4\cos(\beta_4+\gamma_4-\alpha_4) & 0 & 0 & 0 & 0 & 0 & 0 & r_4\cos(\gamma_4) & r_{r4} \end{bmatrix} \begin{pmatrix} {}^{\bar{R}}\dot{P} \\ \dot{\varphi}_1 \\ \dot{\varphi}_{r1} \\ \dot{\varphi}_2 \\ \dot{\varphi}_{r2} \\ \dot{\varphi}_3 \\ \dot{\varphi}_{r3} \\ \dot{\varphi}_4 \\ \dot{\varphi}_{r4} \end{pmatrix}$$

$$= A \cdot {}^{R}\dot{q} \qquad (2.9)$$

where:

$$\beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$$

since the mecanum wheels are fixed to the body and not steerable.

2.3 Dynamics with Lagrange formulation

The main inconvenience of the slip kinematic modeling methods described in the previous subsection is the lack of a proper physical sense. In order to overcome this, here we will consider as a starting point the WMR dynamics with Lagrange formulation. Afterwards, this dynamic approach will be combined with the kinematics to form a system of equations that will help tackle our main problem.

The Lagrangian method utilizes the energies in a system. The central quantity of Lagrangian mechanics is the Lagrangian, a function which summarizes the dynamics. This function makes use of generalized coordinates, their time derivatives, and time, and contains the information about the dynamics of the entire system.

The Lagrangian as a combination of the kinetic and potential energy is defined by:

$$L_{agrangian} = T - U \tag{2.10}$$

where:

- T is the kinetic energy and
- U is the potential energy of the system

The general Lagrange equations for a finite-dimensional dynamic system are given by:

$$\frac{d}{dt}\left(\frac{\vartheta L_{agrangian}}{\vartheta \dot{q}}\right) = \frac{\vartheta L_{agrangian}}{\vartheta \dot{q}} \tag{2.11}$$

Substituting the Lagrangian in the above equation for our system, we get:

$$\frac{d}{dt}\left(\frac{\vartheta T}{\vartheta \dot{q}}\right) - \frac{\vartheta T}{\vartheta q} = Q \tag{2.12}$$

where:

- q is the generalized coordinate vector which in our case is equivalent to ^{R}q
- Q is the generalized force vector that includes the conservative forces

The generalized force vector is:

$$Q = \sum_{i=1}^{N} \tau_i \frac{\vartheta \dot{\varphi}_i}{\vartheta \dot{q}} + \sum_{i=1}^{N} F_{fric\,i}^{T} \frac{\vartheta v_{slip\,i}}{\vartheta \dot{q}} - \sum_{i=1}^{N} D_r \dot{\varphi}_i \tag{2.13}$$

where:

- τ_i rotation torque of wheel i
- $F_{fric\,i}$ frictional force on wheel i in coordinate frame Ei $F_{fric\,i} = [F_{fric\,xi} \quad F_{fric\,yi}]^T$
- D_r friction coefficient

Note that a frictional force $F_{fric\,i}$ in the opposite direction of the sliding velocity $v_{slip\,i}$ produces a negative sign of the second term of equation (2.13), what indicates a dissipative work. Expression (2.13), using (2.9), becomes:

$$Q = \tau + F_{fric}^{T} A - D_r \dot{\varphi} = \tau + A^T F_{fric} - D_r \dot{\varphi} \tag{2.14}$$

where τ is a global torque vector and $F_{fric}, \dot{\varphi}$ are the grouped frictional forces and wheel velocities respectively.

The kinetic energy of the WMR is:

$$T = \frac{1}{2}\left(M_T\left(u_x^2 + u_y^2\right) + I_T \omega^2 + \sum_{i=1}^{N} I_{ri}\dot{\varphi}_i + \sum_{i=1}^{N} I_{rri}\dot{\varphi}_{ri} \right) \tag{2.15}$$

where:

- M_T mass of the WMR
- I_T moment of inertia of the WMR with respect to a Z axle crossing its center of mass
- I_{ri} moment of inertia of wheel i with respect to its rotation axle
- I_{rri} moment of inertia of the roller of Swedish wheel i

The mobile robot dynamics (2.12), taking into account (2.9), (2.13), (2.14), (2.15) result in:

$$M_T \dot{v} = \sum_{i=1}^{N} Rot({}^{R}\theta_{Ei}) F_{fric\,i} \tag{2.16}$$

$$I_T \dot{\omega} = \sum_{i=1}^{N} \left({}^{R}d_{Ei} \times \begin{pmatrix} Rot({}^{R}\theta_{Ei}) F_{fric\,i} \\ 0 \end{pmatrix} \right) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{2.17}$$

$$I_{rj}\ddot{\varphi}_j = \tau_j + r_j [\sin(\gamma_j) \quad cos(\gamma_j)] \cdot F_{fric\,j} - D_r \cdot \dot{\varphi}_j \tag{2.18}$$

$$I_{rrj}\ddot{\varphi}_{rj} = r_{rj} [0 \quad 1] \cdot F_{fric\,j} \tag{2.19}$$

with j from 1 to N, where:

- $\dot{v} = [u_x \quad u_y]^T$ is the robot's body linear velocities
- ${}^{R}d_{Ei} = [\pm l \quad \pm L \quad 0]^T$ is the distance vector from origin of robot frame R to frame Ei

Considering the equations for each individual wheel and the robot's body velocities, we can further analyze the above expressions and get the final equations regarding the WMR's dynamics:

$$M_T\dot{v} = \left(Rot({}^R\theta_{E1})F_{fric\,1} + Rot({}^R\theta_{E2})F_{fric\,2} + Rot({}^R\theta_{E3})F_{fric\,3} + Rot({}^R\theta_{E4})F_{fric\,4}\right) \quad (2.20)$$

$$I_T\dot{\omega} = \sum_{i=1}^{N}\left({}^R d_{Ei} \times \begin{pmatrix} Rot({}^R\theta_{Ei})F_{fric\,i} \\ 0 \end{pmatrix}\right) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad (2.21)$$

$$I_{r1}\ddot{\varphi}_1 = \tau_1 + r_1[\sin(\gamma_1) \quad cos(\gamma_1)] \cdot F_{fric\,1} - D_r \cdot \dot{\varphi}_1 \quad (2.22)$$

$$I_{r2}\ddot{\varphi}_2 = \tau_2 + r_2[\sin(\gamma_2) \quad cos(\gamma_2)] \cdot F_{fric\,2} - D_r \cdot \dot{\varphi}_2 \quad (2.23)$$

$$I_{r3}\ddot{\varphi}_3 = \tau_3 + r_3[\sin(\gamma_3) \quad cos(\gamma_3)] \cdot F_{fric\,3} - D_r \cdot \dot{\varphi}_3 \quad (2.24)$$

$$I_{r4}\ddot{\varphi}_4 = \tau_4 + r_4[\sin(\gamma_4) \quad cos(\gamma_4)] \cdot F_{fric\,4} - D_r \cdot \dot{\varphi}_4 \quad (2.25)$$

$$I_{rr1}\ddot{\varphi}_{r1} = r_{r1}[0 \quad 1] \cdot F_{fric\,1} \quad (2.26)$$

$$I_{rr2}\ddot{\varphi}_{r2} = r_{r2}[0 \quad 1] \cdot F_{fric\,2} \quad (2.27)$$

$$I_{rr3}\ddot{\varphi}_{r3} = r_{r3}[0 \quad 1] \cdot F_{fric\,3} \quad (2.28)$$

$$I_{rr4}\ddot{\varphi}_{r4} = r_{r4}[0 \quad 1] \cdot F_{fric\,4} \quad (2.29)$$

2.4 Kinematic Modeling with Slip

As we have previously stated, wheel slippage could have a considerable impact on the accuracy of the robot and therefore cause it to fail to perform its missions and achieve its objectives.

This section introduces the kinematic modeling with slip for an omni-directional mobile platform with four mecanum wheels that has been proposed in order to surmount the problems caused by wheel slippage.

We consider the full dynamic model along with the no-slip kinematic equations, $v_{slip} = 0$, for each wheel for the nominal case where there is no wheel slip present. In the event that the no-slip kinematic equations are not satisfied, thus there is wheel slip present, we make a change in our model and we replace the no-slip equation with the condition that the frictional force on the slipping wheel is equal to the Coulomb friction force.

Friction is the force resisting the relative motion of surfaces sliding against each other. There are several types of friction, but in classical mechanics we are mostly interested in dry friction. Dry friction is a force that opposes the relative lateral motion of two solid surfaces in contact. It can be subdivided into:

- static friction between non-moving surfaces and
- kinetic or dynamic friction between moving surfaces

With the exception of atomic or molecular friction, dry friction generally arises from the interaction of surface features, known as asperities. When surfaces in contact move relative to each other, the friction between the two surfaces converts kinetic energy into thermal energy, that is, it converts work to heat.

Friction is desirable and important in supplying traction to facilitate motion on land. Most land vehicles rely on friction for acceleration, deceleration and changing direction. Sudden reductions in traction can cause loss of control and accidents. Friction is not itself a fundamental force. Dry friction arises from a combination of inter-surface adhesion, surface roughness, surface deformation, and surface contamination. The complexity of these interactions makes the calculation of friction from first principles impractical and necessitates the use of empirical methods for analysis and the development of theory.

The Coulomb model of friction, named after Charles-Augustin de Coulomb, refers to an approximate model used to calculate the force of dry friction and is fairly accurate empirically. Specifically, it is a model that helps predict the direction and magnitude of the friction force between two bodies with dry surfaces in contact.

It is governed by the model:

$$F \leq \mu \cdot F_N \tag{2.30}$$

where

- F is the force of friction exerted by each surface on the other. It is parallel to the surface, in a direction opposite to the net applied force
- μ is the coefficient of friction, which is an empirical property of the contacting materials
- F_N is the normal force exerted by each surface on the other, directed perpendicular (normal) to the surface

Static friction occurs when there is no slipping between the two surfaces of contact. This can occur both when the two bodies are in rest relative to each other, and when the surfaces are rolling against each other. There exists a constant μ_s, termed the limiting coefficient of static friction, dependent on the nature of the two surfaces in contact, such that the static friction is determined by the following rules:

- Static friction is equal in magnitude and opposite in direction to the net external force acting along the plane of contact that is causing a tendency for the surfaces to slip against each other. In other words, it acts to precisely cancel out the external force that would otherwise have caused slipping, thus producing no slipping.
- The maximum value of static friction that can be experienced through the surface of contact equals $\mu_s F_N$, where μ_s is the limiting coefficient of static friction. In the limiting case, the two surfaces are said to just start slipping against each other.

Kinetic friction or dynamic friction, occurs when there is slipping between the two surfaces of contact. There exists a constant μ_k, termed the coefficient of kinetic friction, dependent on the nature of the two surfaces in contact, such that, for sufficiently small speeds of slipping:

- Kinetic friction is opposite in direction to the direction of actual slipping. Its direction does not depend on the direction of other external forces.
- The magnitude of kinetic friction is $\mu_k F_N$, where μ_k is the coefficient of kinetic friction, and F_N is the normal force between the bodies.

The coefficient μ_k for a given pair of surfaces is typically somewhat less than μ_s, indicating that once two bodies start slipping against each other, it is easier for them to continue slipping.

If one of the surfaces of contact is also undergoing rotational motion, then the relevant slipping is not the relative motion of the bodies but the relative motion of the surfaces, which takes into account both the sliding and the rotation.

A special case is the case of rolling, which satisfies the no slip condition: there is no slipping at the surface of contact. This happens because the relative translational speed at the point of contact cancels the rotational speed at the point of contact. Thus, we note that:

- In the case of rolling, where the no slip condition is satisfied, we apply the model for static friction. In particular, the friction force is bounded from above in magnitude by $\mu_s F_N$, but need not equal $\mu_s F_N$. The direction is determined by looking at the force and torque equations.

- In any case other than rolling, we have kinetic friction, so the model for kinetic friction applies. The magnitude of friction force is $\mu_k F_N$, and its direction is opposite the direction of relative slipping of the surfaces, which is determined by combining the translational motion of the bodies and the rotational motion at the region of contact.

In our approach, we will utilize the notion of static friction and its limiting coefficient μ_s. When the two surfaces of interest, the wheel roller and the ground, begin slipping against each other, the frictional force on the wheel reaches its upper bound $F_c = \mu_s F_N$. By examining the magnitudes of the frictional forces on the four wheels of the vehicle, we are able to determine whether a wheel is slipping.

The main idea behind this approach is described in the flow diagram below (Figure 2-3):

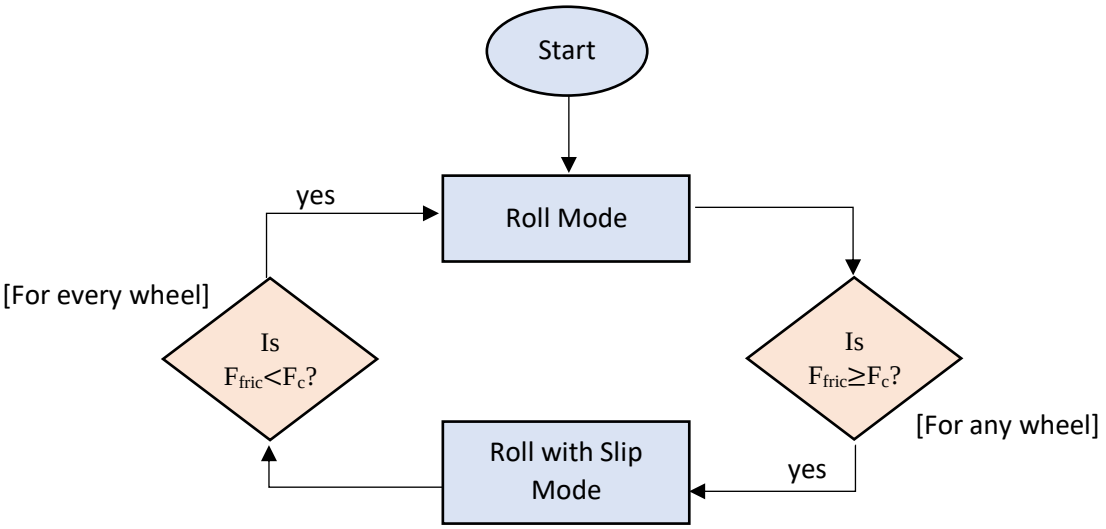

Fig. 2-3 Flow diagram of slip conditions

The proper selection of the friction coefficient plays a vital role in the accuracy of the system and therefore should be chosen wisely.

The Kinematic Modeling with slip is governed by the following Non-Linear Differential equations:

$$M_T\dot{v} = \left(Rot({}^R\theta_{E1})F_{fric\,1} + Rot({}^R\theta_{E2})F_{fric\,2} + Rot({}^R\theta_{E3})F_{fric\,3} + Rot({}^R\theta_{E4})F_{fric\,4}\right) \quad (2.31)$$

$$I_T\dot{\omega} = \sum_{i=1}^{N}\left({}^R d_{Ei} \times \begin{pmatrix} Rot({}^R\theta_{Ei})F_{fric\,i} \\ 0 \end{pmatrix}\right) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad (2.32)$$

$$I_{r1}\ddot{\varphi}_1 = \tau_1 + r_1[\sin(\gamma_1) \quad cos(\gamma_1)] \cdot F_{fric\,1} - D_r \cdot \dot{\varphi}_1 \quad (2.33)$$

$$I_{r2}\ddot{\varphi}_2 = \tau_2 + r_2[\sin(\gamma_2) \quad cos(\gamma_2)] \cdot F_{fric\,2} - D_r \cdot \dot{\varphi}_2 \quad (2.34)$$

$$I_{r3}\ddot{\varphi}_3 = \tau_3 + r_3[\sin(\gamma_3) \quad cos(\gamma_3)] \cdot F_{fric\,3} - D_r \cdot \dot{\varphi}_3 \quad (2.35)$$

$$I_{r4}\ddot{\varphi}_4 = \tau_4 + r_4[\sin(\gamma_4) \quad cos(\gamma_4)] \cdot F_{fric\,4} - D_r \cdot \dot{\varphi}_4 \quad (2.36)$$

$$I_{rr1}\ddot{\varphi}_{r1} = r_{r1}[0 \quad 1] \cdot F_{fric\,1} \quad (2.37)$$

$$I_{rr2}\ddot{\varphi}_{r2} = r_{r2}[0 \quad 1] \cdot F_{fric\,2} \quad (2.38)$$

$$I_{rr3}\ddot{\varphi}_{r3} = r_{r3}[0 \quad 1] \cdot F_{fric\,3} \quad (2.39)$$

$$I_{rr4}\ddot{\varphi}_{r4} = r_{r4}[0 \quad 1] \cdot F_{fric\,4} \quad (2.40)$$

$$v_{slip}(1) = 0 \quad (2.41)$$

$$v_{slip}(2) = 0 \quad (2.42)$$

$$v_{slip}(3) = 0 \quad (2.43)$$

$$v_{slip}(4) = 0 \quad (2.44)$$

$$v_{slip}(5) = 0 \quad (2.45)$$

$$v_{slip}(6) = 0 \quad (2.46)$$

$$v_{slip}(7) = 0 \quad (2.47)$$

$$v_{slip}(8) = 0 \quad (2.48)$$

where

$$v_{slip} = \begin{pmatrix} v_{slip\,x1} \\ v_{slip\,y1} \\ v_{slip\,x2} \\ v_{slip\,y2} \\ v_{slip\,x3} \\ v_{slip\,y3} \\ v_{slip\,x4} \\ v_{slip\,y4} \end{pmatrix} = A \cdot \dot{q} =$$

$$= \begin{bmatrix} \cos(\gamma_1) & \sin(\gamma_1) & l_1\sin(\gamma_1-\alpha_1) & r_1\sin(\gamma_1) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\sin(\gamma_1) & \cos(\gamma_1) & l_1\cos(\gamma_1-\alpha_1) & r_1\cos(\gamma_1) & r_{r1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \cos(\gamma_2) & \sin(\gamma_2) & l_2\sin(\gamma_2-\alpha_2) & 0 & 0 & r_2\sin(\gamma_2) & 0 & 0 & 0 & 0 & 0 \\ -\sin(\gamma_2) & \cos(\gamma_2) & l_2\cos(\gamma_2-\alpha_2) & 0 & 0 & r_2\cos(\gamma_2) & r_{r2} & 0 & 0 & 0 & 0 \\ \cos(\gamma_3) & \sin(\gamma_3) & l_3\sin(\gamma_3-\alpha_3) & 0 & 0 & 0 & 0 & r_3\sin(\gamma_3) & 0 & 0 & 0 \\ -\sin(\gamma_3) & \cos(\gamma_3) & l_3\cos(\gamma_3-\alpha_3) & 0 & 0 & 0 & 0 & r_3\cos(\gamma_3) & r_{r3} & 0 & 0 \\ \cos(\gamma_4) & \sin(\gamma_4) & l_4\sin(\gamma_4-\alpha_4) & 0 & 0 & 0 & 0 & 0 & 0 & r_4\sin(\gamma_4) & 0 \\ -\sin(\gamma_4) & \cos(\gamma_4) & l_4\cos(\gamma_4-\alpha_4) & 0 & 0 & 0 & 0 & 0 & 0 & r_4\cos(\gamma_4) & r_{r4} \end{bmatrix} \begin{pmatrix} {}^{\bar{R}}\dot{P} \\ \dot{\varphi}_1 \\ \dot{\varphi}_{r1} \\ \dot{\varphi}_2 \\ \dot{\varphi}_{r2} \\ \dot{\varphi}_3 \\ \dot{\varphi}_{r3} \\ \dot{\varphi}_4 \\ \dot{\varphi}_{r4} \end{pmatrix} \quad (2.49)$$

In order to be able to solve this system and acquire the robot's position and velocities, we will follow the succeeding procedure. Our aim is to solve for the frictional forces on the wheels and substitute them in our kinematic and dynamic equations. First, we will consider the modeling equations in a matrix form. Then, by differentiating the no-slip equations, we will get an expression with respect to the state vector $\ddot{q}$. By combining the previously described equations, we will manage to form an expression for the frictional forces.

If we express the above system of equations (eq. 2.31-2.40) in a more compact form, we obtain the following:

$$M\ddot{q} = f\left(F_{fric\,i}, \tau_i, \dot{\varphi}_i\right) \quad (2.50)$$

which denotes that $\ddot{q}$ is a function of the frictional forces, torques and wheel velocities of the robot, where

$$M = \begin{bmatrix} M_T & 0 & & & & \dots & & & & & 0 \\ 0 & M_T & & & & & & & & & \\ & & I_T & & & & & & & & \\ & & & I_{r1} & & & & & & & \\ & & & & I_{rr1} & & & & & & \\ \vdots & & & & & I_{r2} & & & & & \vdots \\ & & & & & & I_{rr2} & & & & \\ & & & & & & & I_{r3} & & & \\ & & & & & & & & I_{rr3} & & \\ & & & & & & & & & I_{r4} & 0 \\ 0 & & & & & \dots & & & & 0 & I_{rr4} \end{bmatrix} \quad (2.51)$$

Then by solving for $\ddot{q}$, we have:

$$\ddot{q} = M^{-1} \cdot f\left(F_{fric\,i}, \tau_i, \dot{\varphi}_i\right) \tag{2.52}$$

The no-slip condition indicates that the sliding velocities of the wheels equal to zero:

$$A \cdot \dot{q} = 0 \tag{2.53}$$

This means that, under the no-slip condition, the following equation is also satisfied:

$$A \cdot \ddot{q} = 0 \tag{2.54}$$

Substituting (2.52) into (2.54) results in:

$$A \cdot \ddot{q} = A \cdot M^{-1} \cdot f\left(F_{fric\,i}, \tau_i, \dot{\varphi}_i\right) = 0 \tag{2.55}$$

Now, we can solve (2.55) with respect to the frictional forces F_{fric}. This produces the subsequent output:

$$A' \cdot F_{fric} + B' = 0 \tag{2.56}$$

$$\Rightarrow F_{fric} = -A'^{-1} \cdot B' \tag{2.57}$$

The frictional forces F_{fric} are a function of the torques on the wheels and the wheel velocities.

$$F_{fric} = f_1(\tau_i, \dot{\varphi}_i) \tag{2.58}$$

We can utilize this relation to identify slipping wheels and also solve the system of equations (eq. 2.31-2.40) in order to acquire the robot's position and velocities at every time step.

Chapter 3

Fault Diagnosis

3.1 Introduction

In order for robots to operate safely and efficiently, it is essential to monitor their behavior so that possible faults can be addressed before they result in catastrophic failures.

Model based Fault Detection and Isolation (FDI) is a method that performs fault diagnosis using mathematical models by identifying significant deviations of the system's actual response compared to the one of the theoretical model. The mathematical model of the process runs in parallel to the real system and is driven by the same inputs. When a fault occurs the outputs of the process are dissimilar from those of the mathematical model. The difference between the measured process variables and their estimates through the model is called residual.

Among the various methods in the design of a residual generator only few deal with nonlinear systems. Structural analysis is a technique that provides feasible solutions to residual generation for nonlinear systems. The goal is to detect the faulty components as early as possible and provide a timely warning in order for a fault tolerant control scheme to be utilized so that the system can remain functional.

Fault accommodation of autonomous vehicles is a well-known problem in the robotics community and plenty of studies have dealt with it. However, prior to fault accommodation, a fault diagnosis scheme has to be applied. Fault diagnosis for wheeled mobile robots is a complex problem due to the large number of faults that can be present such as faults of sensors and actuators.

In this work, we propose a model-based actuator fault diagnosis for an omni-directional mobile robot with 4 mecanum wheels. We consider actuation faults in which the wheels are not able to receive commands, but still can rotate freely due to the friction with the ground. We assume that only the measurements of the position and the heading angle of the robot are available.

The residuals are calculated based on the position differences between the observer and the plant, available by an external position sensor. We are able to detect actuator faults in case one or two simultaneous faults appear (since fault control schemes can compensate for up to two faulty wheels) and identify the wheels that the fault has occurred on.

3.2 Structural analysis for fault diagnosis

Structural analysis is a qualitative modeling method that takes into consideration the properties of a dynamical system by analyzing its structural model and captures whether relations between equations exist. The structural model of a system is a set of relations between a set of variables. These relations are also referred to as constrains. The structural model may offer useful information as concerning the components of the system that are not measurable as well as to provide a parity equation for residual generation. The system's structural model can be represented by a bipartite graph that connects constraints and variables. An incidence matrix can be used as a representation of the structure graph.

In order to utilize the structural analysis technique, we must first consider the kinematic model of the mobile platform that serves to the creation of the structural model of the system. To simplify our calculations, here we will use the conventional kinematics of the mobile platform, which is given by:

$$\begin{bmatrix} u_x \\ u_y \\ \omega \end{bmatrix} = J_V \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix} \tag{3.1}$$

where

$$J_V = \frac{R_w}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\frac{1}{L+l} & \frac{1}{L+l} & -\frac{1}{L+l} & \frac{1}{L+l} \end{bmatrix} \tag{3.2}$$

while the robot's body velocities in the inertial frame are given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = J_I \begin{bmatrix} u_x \\ u_y \\ \omega \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ \omega \end{bmatrix} \tag{3.3}$$

In this work we are only interested in abrupt faults which occur in the actuators of the mobile robot and as consequence, we make the following assumptions:

- Assumption 1: When the mobile robot starts functioning, all of its components are in normal mode.
- Assumption 2: The magnitude of the noise is assumed to be significantly smaller than the magnitude of the faults.
- Assumption 3: When a fault occurs, the system will remain in that faulty state.

We construct the set of model constraints C, with elements c_i, corresponding to each of the equations comprising the mathematical model of our system. The omni-directional mobile robot's structural model is represented by the following set of constrains:

$$C = \{c_1, c_2, \dots, c_7\} \tag{3.4}$$

and a set of variables:

$$Z = X \cup K \tag{3.5}$$

where X is a subset of unknown variables and K is a subset of known variables which are measurements and inputs.

The set of unknown variables is:

$$X = \{\dot{x}, \dot{y}, \dot{\theta}, u_x, u_y\} \tag{3.6}$$

The set of known variables is:

$$K = \{\theta, x, y, \omega, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4\} \tag{3.7}$$

The set of constrains for the four-wheel omni-directional mobile robot is:

$$c_1: \quad \dot{x} = cos\theta u_x - sin\theta u_y \tag{3.8}$$

$$c_2: \quad \dot{y} = sin\theta u_x + cos\theta u_y \tag{3.9}$$

$$c_3: \quad \dot{\theta} = \omega \tag{3.10}$$

$$c_4: \quad x = \int_0^t \dot{x} d\tau \tag{3.11}$$

$$c_5: \quad y = \int_0^t \dot{y}\, d\tau \tag{3.12}$$

$$c_6: \quad \theta = \int_0^t \omega d\tau \tag{3.13}$$

$$c_7: \quad u_x = \frac{R_w}{4}(\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4) \tag{3.14}$$

The system structure can be described by its incidence matrix, the structure of which is shown in Table 3-1.

	KNOWN								UNKNOWN				
	θ	x	y	ω	$\dot{\theta}_1$	$\dot{\theta}_2$	$\dot{\theta}_3$	$\dot{\theta}_4$	$\dot{x}$	$\dot{y}$	$\dot{\theta}$	u_x	u_y
c_1	1								[1]			1	1
c_2	1									[1]		1	1
c_3				1							[1]		
c_4		1							1				
c_5			1							1			
c_6	1										1		
c_7					1	1	1	1				[1]	

Table 3-1 Incidence matrix

3.3 Residual generation

As we have previously stated, model-based fault detection and isolation is a method to perform fault diagnosis using mathematical models. The mathematical model of the process runs in parallel to the real system and is driven by the same inputs. When a fault occurs the outputs of the process are dissimilar from those of the mathematical model. The difference between the measured process variables and their estimates is called residual. The procedure of creating the estimates of the process outputs and constructing the difference between the process outputs and their estimates is called residual generation and is the core of the method.

Using the above incidence matrix (Table 3-1) and applying a matching algorithm, we may extract the matched and unmatched constrains. As a result, the matched constrains are:

$$M = \{c_1, c_2, c_3, c_7\} \tag{3.15}$$

The unmatched constrains are:

$$U = \{c_4, c_5, c_6\} \tag{3.16}$$

The parity equations that that can be used as residual generators are:

$$c_4(x, \dot{x}) = 0 \tag{3.17}$$

$$c_5(y, \dot{y}) = 0 \tag{3.18}$$

$$c_6(\theta, \dot{\theta}) = 0 \tag{3.19}$$

Using the above equations by backtracking from the unknown variables to known ones, we get the following three residuals:

$$\boldsymbol{r_1} = x - \int_0^t \dot{x} d\tau = x - \int_0^t (cos\theta u_x - sin\theta u_y) d\tau \tag{3.20}$$

$$\Rightarrow \quad \boldsymbol{r_1} = x - \int_0^t \left[cos\theta \frac{R_w}{4} \left(\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4\right) - sin\theta \frac{R_w}{4} \left(-\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 - \dot{\theta}_4\right) \right] d\tau \tag{3.21}$$

$$\boldsymbol{r_2} = y - \int_0^t \dot{y} d\tau = y - \int_0^t (sin\theta u_x + cos\theta u_y) d\tau \tag{3.22}$$

$$\Rightarrow \quad \boldsymbol{r_2} = y - \int_0^t \left[sin\theta \frac{R_w}{4} \left(\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4\right) + cos\theta \frac{R_w}{4} \left(-\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 - \dot{\theta}_4\right) \right] d\tau \tag{3.23}$$

$$\boldsymbol{r_3} = \theta - \int_0^t \omega d\tau \tag{3.24}$$

The residuals are usually checked against a threshold value r_{th}. This limit checking, yields binary outputs:

$$r_i = \begin{cases} 0 & if \quad |r_i| < r_{th} \\ 1 & if \quad |r_i| > r_{th} \end{cases} \tag{3.25}$$

The selection of thresholds can greatly affect the performance of the detection system, since the residuals are used as an alarm while monitoring the robot's behavior and provide a timely warning so that the robotic system can remain functional and prevent unwanted behaviors.

3.4 Fault diagnosis

Consider an omni-directional mobile platform with four mecanum wheels moving on a flat surface. We assume that measurements of its position and orientation are available and that any slipping between the wheels and the flat surface is negligible.

As a faulty wheel, we consider the case where it cannot be actuated and hence it rotates freely around its driveshaft owing to the friction with the flat surface.

When the mobile robot starts functioning, all of its components are operating normally. Suddenly, a fault appears on one or more actuators and the robot must detect not only that there is a fault present but also identify the wheel or wheels that the fault has occurred on. This procedure is called fault diagnosis and will be achieved through the use of the preceding residuals.

We will consider two cases depending on the multitude of the faulty wheels:

- **Case 1** One Faulty Wheel
- **Case 2** One or Two Faulty Wheels

Case 1: One faulty wheel

In this first part, we will examine the case where only one wheel can become faulty. We program the mobile robot to follow a straight trajectory and we apply four different faults:

- front left wheel faulty (wheel 1)
- front right wheel faulty (wheel 2)
- rear left wheel faulty (wheel 3)
- rear right wheel faulty (wheel 4)

The robot starts from the origin of the inertial frame with heading $\vartheta(0) = 0°$ and moves along its x-axis in a straight trajectory. Figure 3-1 depicts the results of the simulation and in particular the path of the vehicle.

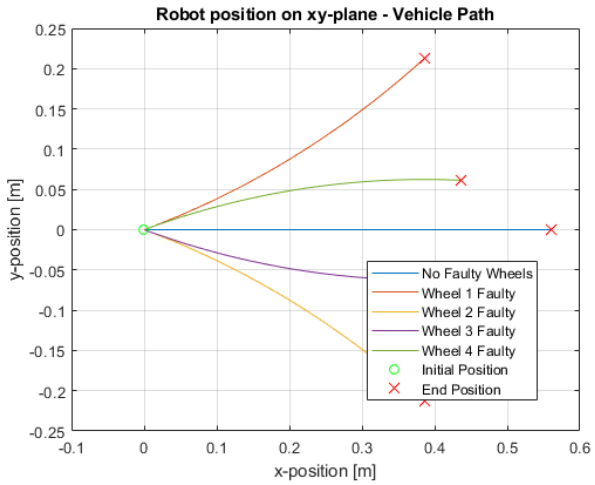

Fig. 3-1 Vehicle path

Figure 3-2 presents the yaw angle of the platform and Figure 3-3 the trajectory of the robot in the y-axis for all possible faults.

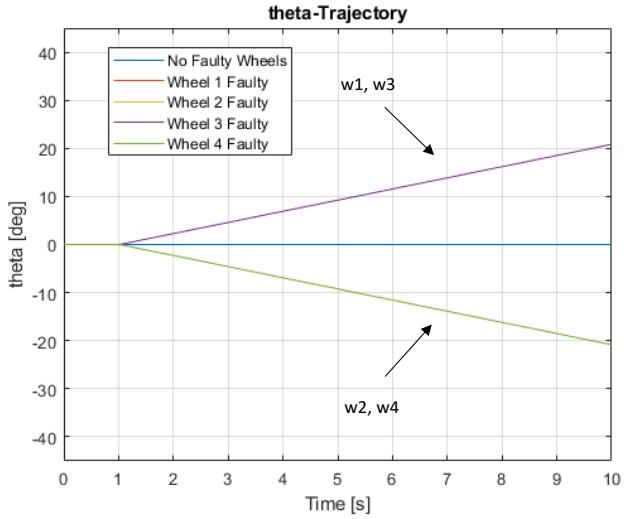

Fig. 3-2 Robot heading angle

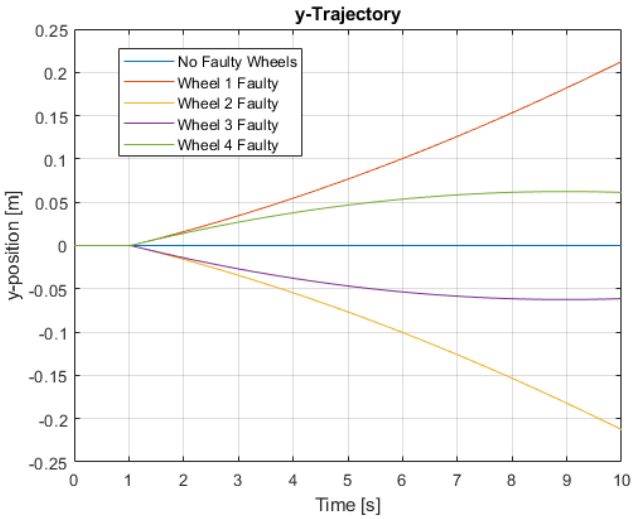

Fig. 3-3 y-axis Trajectory

We can clearly see that when there is a fault present, the vehicle deviates from its nominal path. After t=1s, a deviation in the heading angle begins to appear. If the heading angle becomes positive, then either wheel 1 or 3 is faulty. If the heading angle becomes negative, then either wheel 2 or 4 is faulty. In order to distinguish between faults in wheels 1 and 3 or in wheels 2 and 4, we also need to take into consideration the trajectory of the robot in the opposite axis that the robot is moving along. But in order for the robot to be able to identify the fault, it should consider the residuals.

If a fault occurs in one wheel, the heading angle will deviate from the reference angle and the residual signal r3 will become active. Then, in order to distinguish between the two possible faults, we also examine the trajectory residual signal r2.

Let us now present a simple example illustrating the aforementioned idea. Figure 3-4 shows the deviation between the heading angle and the reference angle when wheel 1 is faulty. As long as the robot follows its nominal trajectory, the residual is inactive and when the deviation from its nominal path starts, the residual signal r3 gets activated.

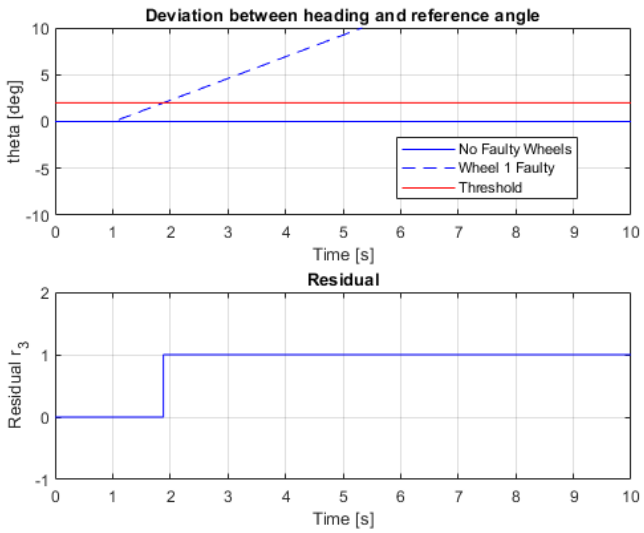

Fig. 3-4 Heading angle residual in case of wheel 1 faulty

At around t=2s, the heading angle exceeds the threshold of 2° that we have set and consequently the residual r3 becomes active to alert us that a deviation from the nominal path has occurred. We also examine the trajectory residual signal r2 which is illustrated in Figure 3-5.

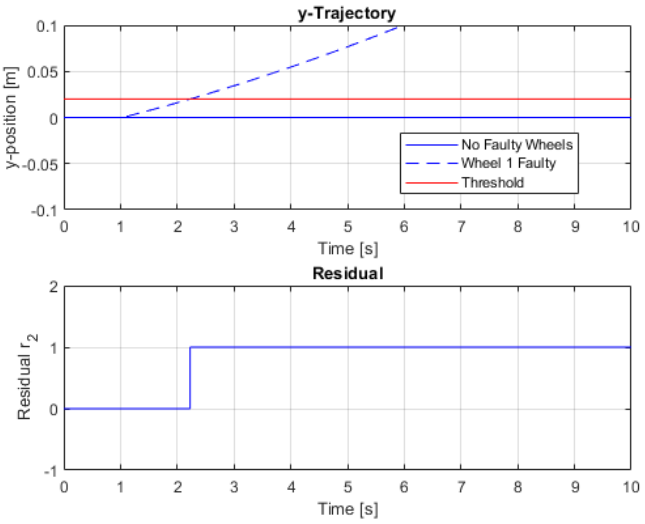

Fig. 3-5 Trajectory residual in case of wheel 1 faulty

Both of the residual signals r2 and r3 are activated. The heading angle is positive and the robot's y-position is increasing. Therefore, we can safely conclude that the actuator fault has occurred in wheel 1.

The approach behind this relationship is the following:

In case the heading angle is positive, either wheel 1 or 3 is faulty. If the robot's y-position is increasing (meaning the robot moves to the left), then we conclude that the fault has occurred in wheel 1. If the robot's y-position is decreasing (meaning the robot moves to the right), then we conclude that the fault has occurred in wheel 3.

In case the heading angle is negative, either wheel 2 or 4 is faulty. If the robot's y-position is increasing (meaning the robot moves to the left), then we conclude that the fault has occurred in wheel 4. If the robot's y-position is decreasing (meaning the robot moves to the right), then we conclude that the fault has occurred in wheel 2.

The possible outcomes from the fault detection algorithm, in the case where only one wheel can become faulty, are represented in the following table:

	Direction 1a		Direction 1b	
	$y > y_{thr}$	$y < -y_{thr}$	$y > y_{thr}$	$y < -y_{thr}$
$\theta > \theta_{thr}$	w1	w3	w2	w4
$\theta < -\theta_{thr}$	w4	w2	w3	w1
	Direction 2a		Direction 2b	
	$x > x_{thr}$	$x < -x_{thr}$	$x > x_{thr}$	$x < -x_{thr}$
$\theta > \theta_{thr}$	w4	w3	w2	w1
$\theta < -\theta_{thr}$	w1	w2	w3	w4

Table 3-2 Fault detection results for 1 faulty wheel

where direction:

- 1a: Forward - Movement towards the robot's positive x-axis
- 1b: Backwards - Movement towards the robot's negative x-axis
- 2a: To the left - Movement towards the robot's positive y-axis
- 2b: To the right - Movement towards the robot's negative y-axis

Case 2: One or two faulty wheels

This is the general case where we do not have prior knowledge on the multitude of the faulty wheels. It could be one or two wheels faulty. We consider ten different faults:

- wheel 1 faulty
- wheel 2 faulty
- wheel 3 faulty
- wheel 4 faulty
- wheels 1&2 faulty
- wheels 1&3 faulty
- wheels 1&4 faulty
- wheels 2&3 faulty
- wheels 2&4 faulty
- wheels 3&4 faulty

The robot has to follow one or two separate straight trajectories along it's x or y-axis. At first, it starts from the origin of the inertial frame with heading $\vartheta(0) = 0°$ and moves along its x-axis in a straight trajectory. Figure 3-6 depicts the results of the simulation and in particular the yaw angle of the platform. The trajectories of the robot in the opposite axis that the robot is moving along are shown in Figure 3-7 for all possible faults.

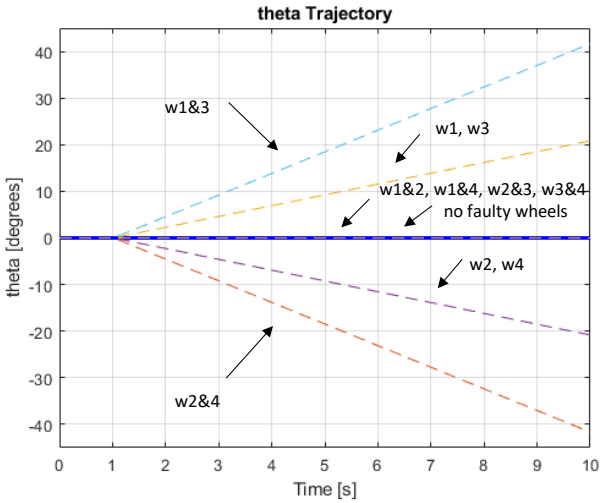

Fig. 3-6 Robot heading angle

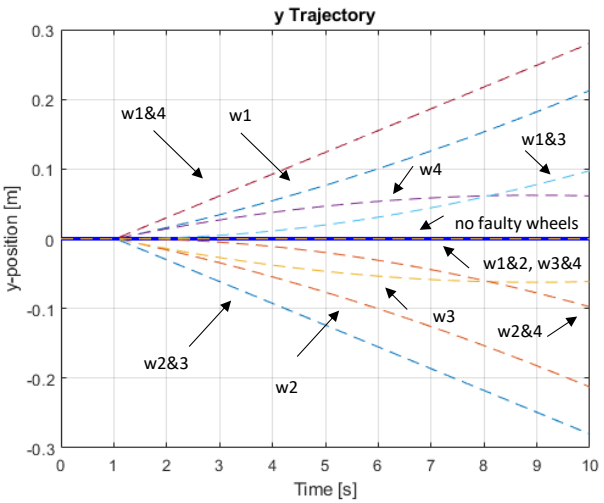

Fig. 3-7 y-axis Trajectory

If the heading angle deviates from the reference angle and is positive, then wheels 1, 3 or 1&3 are faulty. If the heading angle is negative, then wheels 2, 4 or 2&4 are faulty. If the heading angle does not deviate, maybe wheels 1&2, 1&4, 2&3 or 3&4 are faulty.

We will provide some examples to illustrate the detection algorithm. Figure 3-8 shows the fault detection residuals along with the heading angle and trajectory when wheel 3 is faulty. As long as the robot follows its nominal trajectory, the residuals are inactive and when the deviation from its nominal path starts, the residual signals get activated. The heading angle is positive, which means that either wheels 1, 3 or 1&3 are faulty. Furthermore, the robot's y-position is decreasing (the robot moves to the right) which means that either wheels 2, 3, 2&3 or 2&4 are faulty. Therefore, we conclude that the fault has occurred in wheel 3.

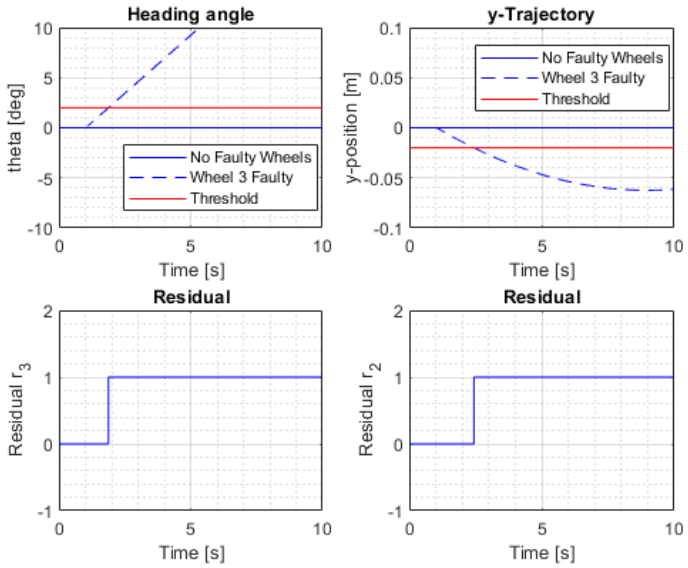

Fig. 3-8 Fault detection in case of wheel 3 faulty

There are some faults though, that have the same effect on the residual signals and as a result the robot needs to follow one more straight trajectory in order to be able to identify the faulty wheels. Figure 3-9 illustrates the fault detection residuals along with the heading angle and trajectory when wheels 1&3 are faulty.

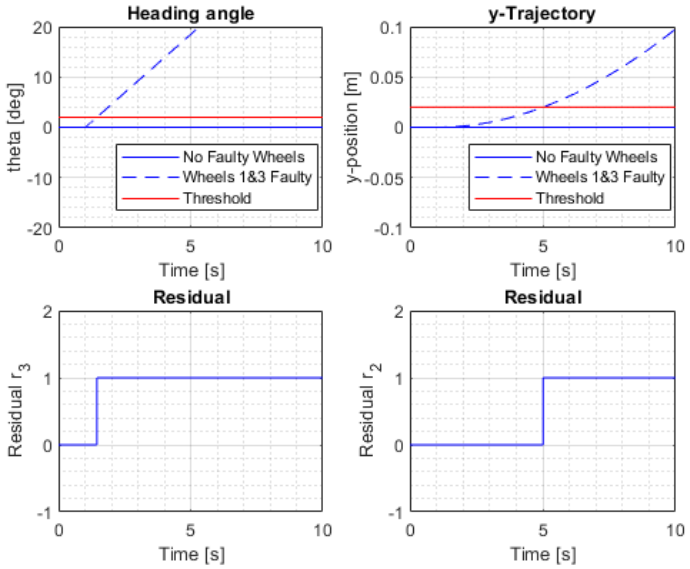

Fig. 3-9 Fault detection in case of wheels 1&3 faulty

The heading angle is positive, which means that either wheels 1, 3 or 1&3 are faulty. The robot's y-position is increasing (robot moves to the left) which means that either wheels 1, 4, 1&4 or 1&3 are faulty. By combining the two outcomes, we get that the possible faulty wheels are wheel 1 or wheels 1&3. Since the results are inconclusive, the robot needs to follow one more trajectory. For the 2nd movement, the robot moves along its y-axis in a straight trajectory. The results of the experiments for a movement along the robot's y-axis are shown in Figure 3-10.

In our case though, there are only 2 possible faults (wheel 1 or wheels 1&3). Consequently, we can conclude that the fault has occurred in wheels 1&3 if the heading angle does not deviate in the 2nd movement or that the fault has occurred in wheel 1 if the heading angle is negative. Figure 3-11 illustrates the heading angle and the reference angle for the 2nd movement when wheels 1&3 are faulty. Since the residual signal r3 is inactive, we conclude that the fault has occurred in wheels 1&3.

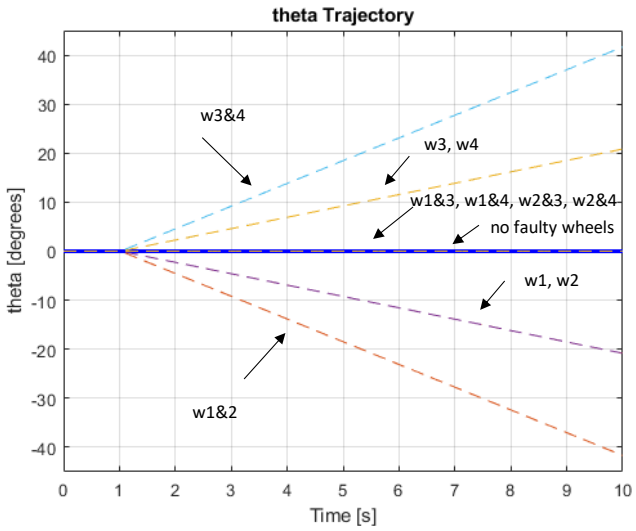

Fig. 3-10 Robot heading angle

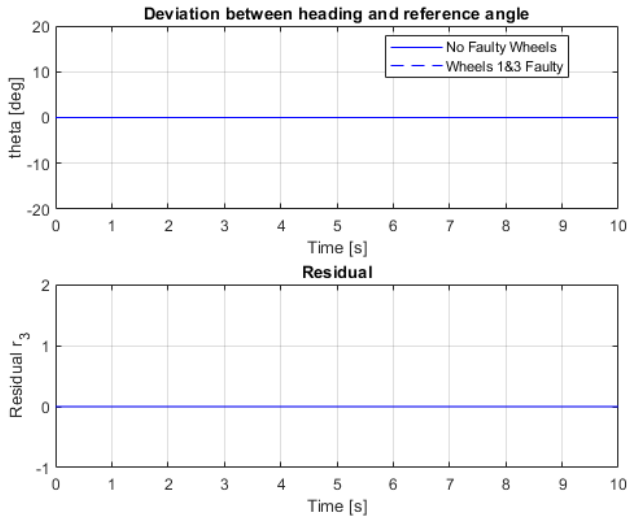

Fig. 3-11 Fault detection in case of wheels 1&3 faulty for 2nd movement

The possible outcomes from the fault detection algorithm, in the case where one or two wheels can become faulty, are represented in the following table:

Direction 1a			
	$y > y_{thr}$	$y < -y_{thr}$	$-y_{thr} < y < y_{thr}$
$\theta > \theta_{thr}$	w1 or 1&3	w3	-
$\theta < -\theta_{thr}$	w4	w2 or 2&4	-
$-\theta_{thr} < \theta < \theta_{thr}$	w1&4	w2&3	w1&2 or 3&4
Direction 1b			
$\theta > \theta_{thr}$	w2	w4 or 2&4	-
$\theta < -\theta_{thr}$	w3 or 1&3	w1	-
$-\theta_{thr} < \theta < \theta_{thr}$	w2&3	w1&4	w1&2 or 3&4
Direction 2a			
	$x > x_{thr}$	$x < -x_{thr}$	$-x_{thr} < x < x_{thr}$
$\theta > \theta_{thr}$	w4	w3 or 3&4	-
$\theta < -\theta_{thr}$	w1 or 1&2	w2	-
$-\theta_{thr} < \theta < \theta_{thr}$	w1&4	w2&3	w1&3 or 2&4
Direction 2b			
$\theta > \theta_{thr}$	w2 or 1&2	w1	-
$\theta < -\theta_{thr}$	w3	w4 or 3&4	-
$-\theta_{thr} < \theta < \theta_{thr}$	w2&3	w1&4	w1&3 or 2&4

Table 3-3 Fault detection results for 1 or 2 faulty wheels

where direction:

- 1a: Forward - Movement towards the robot's positive x-axis
- 1b: Backwards - Movement towards the robot's negative x-axis
- 2a: To the left - Movement towards the robot's positive y-axis
- 2b: To the right - Movement towards the robot's negative y-axis

For the occurrences where the results are inconclusive from the initial movement, meaning there are two possible faults, the robot needs to follow one more trajectory and by combining the results, we are able to identify all possible faults.

Chapter 4

Fault Tolerant Control

4.1 Introduction

In this chapter, we'll describe an approach for guiding robots into obstacle filled environments based on artificial potential fields.

The objective of a fault-tolerant control system is to navigate the mobile platform towards the goal configuration, maintain the current performance close to the desirable one and preserve stability conditions in the presence of faults. Depending on the multitude of the faults, a robust motion control scheme is developed that achieves any desired configuration within the operational workspace, avoids collisions with the obstacles and does not violate the workspace boundaries.

4.2 Artificial Potential Fields

The basic idea behind artificial potential fields is to try to construct a smooth function over the extent of the configuration space, which has high values when the robot is near to an obstacle and lower values when it's further away by incorporating the concept of the potential energy. We also want this function to have its lowest value at the desired goal location while its values increase as we move to configurations that are further away. If we can construct such a function, we can use it's gradient to guide the robot to the desired configuration.

We consider:

- A virtual positive charge on the robot
- A virtual negative charge on the goal destination q_g
- A virtual positive charge on every obstacle in the workspace

The negative charge on the goal destination results to an attractive potential $\varphi_{attr}(q)$ (where q is the robot pose) while the positive charges on the obstacles introduce a repulsive potential $\varphi_{rep}(q)$. The total potential field is described by:

$$U(q) = U_{attr}(q) + U_{rep}(q) \tag{4.1}$$

This way, the robot, which has a positive charge, moves to a lower energy configuration towards the goal based on the control law:

$$\dot{q} = u = -\nabla_q \varphi(q) \tag{4.2}$$

4.3 Navigation Functions

Navigation Functions is a methodology that was introduced in the seminal work of Rimon and Koditschek and is based on artificial potential fields. Its main advantage is that convergence is guaranteed since the potential field has only one local minimum.

Consider a Navigation Functions' based potential $\varphi_i : F_i \rightarrow [0,1]$

$$\varphi \triangleq \frac{\gamma_d}{(\gamma_d{}^k + \beta)^{1/k}} \tag{4.3}$$

where $k \in \mathbb{R} > 0$ is a gain parameter, while γ_d and β are functions that encode the desired behavior and the safety path of the robot respectively.

The attractive potential is given by:

$$\gamma_d = \|q - q_g\|^2 \tag{4.4}$$

The repulsive potential is given by:

$$\beta = \prod \beta_i(q) = \beta_W(q) \cdot \prod \beta_{Oj}(q) \tag{4.5}$$

The first term $\beta_W(q)$ is responsible for retaining the robot inside the workspace, whereas the terms $\beta_{Oj}(q)$ are responsible for collision avoidance with the static obstacles.

$$\beta_W(q) = -\|q - q_o\|^2 + (\rho_o - \rho_{robot})^2 \tag{4.6}$$

$$\prod \beta_{Oj}(q) = \prod \|q - q_j\|^2 - (\rho_j + \rho_{robot})^2 \tag{4.7}$$

where q_j is the position of the center of the obstacles, $q_o = [x_o \quad y_o]^T$ the position of the center of the circular workspace, ρ_o is its radius and ρ_{robot} is the robot equivalent radius.

We observe that the Navigation Function has the following two properties:

1) $\varphi(q) = 0 \Leftrightarrow q = q_g$
2) $\varphi\big(q \in D_j\big) = 1 \quad for\ j = 0,1, \dots, M$

where D_j denotes the obstacles within the workspace and j represents the multitude of them.

4.4 Controller

The control objective is to navigate the mobile platform towards the goal configuration while avoiding collisions with the static obstacles that lie within the constrained workspace and despite the presence of up to two faulty wheels.

In the section, we propose a control scheme that guarantees the asymptotic stabilization of the platform to the goal configuration q_g while achieving the preceding objectives. Its design relies on Navigation Functions.

Firstly, we consider the kinematic model that we developed earlier. In section 2.4 we derived the equations (2.57), (2.58) which represent the frictional forces F_{fric} as a function of the torques on the wheels and the wheel velocities. Substituting those frictional forces on the first three kinematic equations that portray the mobile robot's body velocities (eq. 2.31-2.32) and solving with respect to the torques, we get the following relationship:

$$\begin{pmatrix} M_T \dot{u}_x \\ M_T \dot{u}_y \\ I_T \dot{\omega} \end{pmatrix} = A_m \cdot \tau - B_m \tag{4.8}$$

where A_m is a matrix of constants and B_m depends on the wheel velocities.

Rewriting this in a more compact form, we get:

$$M_m \cdot \dot{V} = A_m \cdot \tau - B_m \tag{4.9}$$

where

$$M_m = \begin{bmatrix} M_T & 0 & 0 \\ 0 & M_T & 0 \\ 0 & 0 & I_T \end{bmatrix} \tag{4.10}$$

$$\dot{V} = \begin{bmatrix} \dot{u}_x \\ \dot{u}_y \\ \dot{\omega} \end{bmatrix} \tag{4.11}$$

Additionally, solving equation (4.8) with respect to τ, gives us:

$$\tau = A_m^T (A_m \cdot A_m^T)^{-1} \cdot M_m (\dot{V} + M_m^{-1} \cdot B_m) \tag{4.12}$$

Controller Design: By utilizing Lyapunov's stability theory, the proposed control scheme for an omni-directional mobile platform with 4 mecanum wheels, which is based on Navigation Functions, is designed as follows:

$$\tau = A_m^T (A_m \cdot A_m^T)^{-1} \cdot M_m (M_m^{-1} \cdot B_m + \dot{u}_d + u_d + K \cdot e_v) \tag{4.13}$$

where

$$u_d = -k_{NF} J_I^T \nabla_q \varphi(q) \tag{4.14}$$

$$\dot{u}_d = -k_{NF} (d J_I^T \cdot \nabla_q \varphi(q) + J_I^T \cdot \nabla^2{}_q \varphi(q)) \tag{4.15}$$

$$e_v = V - u_d \tag{4.16}$$

With this design we are able to navigate the mobile platform to the goal and avoid collisions with the obstacles despite the presence of up to two faulty wheels.

Chapter 5

Implementation

5.1 Parameters of the KUKA youBot

The KUKA youBot omni-directional mobile platform is intended primarily for research and education. KUKA is well known as one of the world's leading industrial robot manufacturer and has designed a rich and open platform with maximal freedom of movement. It comes with a Linux-based, open-source control API compatible with ROS, an optional small arm with 5 degrees of freedom and a payload of 15kg (20kg if we remove the arm). With open interfaces, simply extended with sensors or actors, and equipped with an integrated controller PC, it is possible to work on a broad spectrum of topics, which makes the KUKA youBot an ideal platform for mobile manipulation experiments and application development.

The mobile platform is equipped with 4 mecanum wheels mounted by pairs at each side of the vehicle, evenly with respect to its center of mass, that allow for movement in all directions without mechanical steering via independently controllable servo motors.

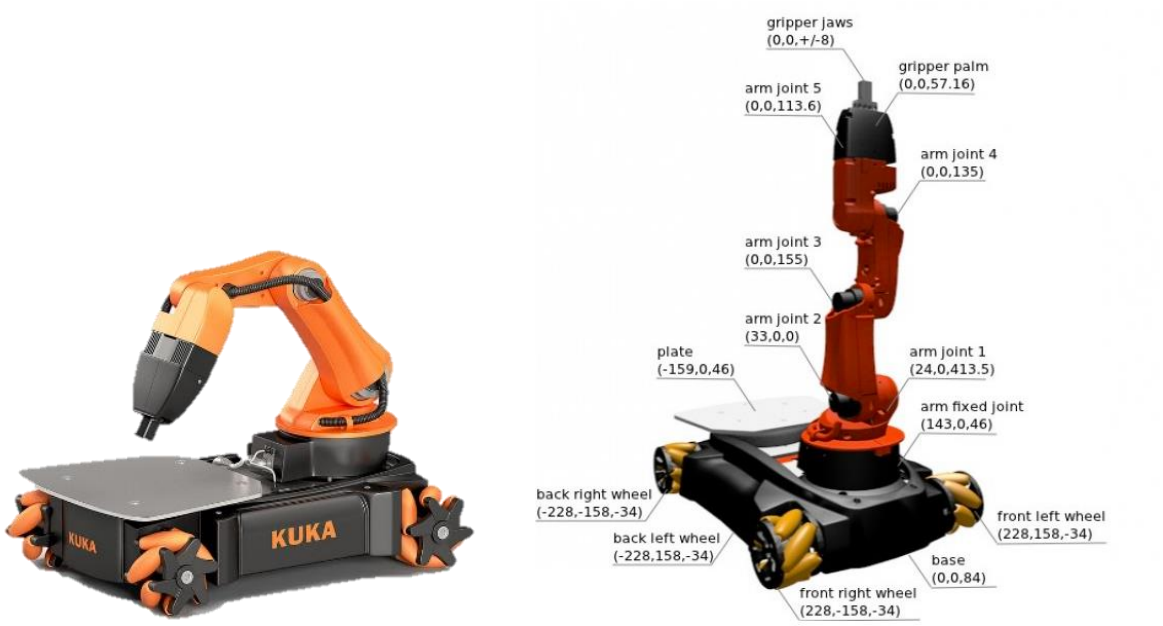

Fig. 5-1 The Kuka youBot omni-directional mobile robot

The technical specifications and needed parameters for our modeling of the KUKA youBot mobile robot are shown below:

- $l = 0.158\ m$ Distance between MC and wheel in MCx-axis
- $L = 0.228\ m$ Distance between MC and wheel axes in MCy-axis
- $l_i = \sqrt{l_{12}^2 + l_{MC}^2} = 0.2774\ m$ Distance between MC and wheel i
- $\alpha_1 = atan2(l_{MC}, l_{12}) = 55.2787°$ Angle between l_1 and MCx-axis
- $\alpha_2 = atan2(l_{MC}, -l_{12}) = 124.7213°$ Angle between l_2 and MCx-axis
- $\alpha_3 = atan2(-l_{MC}, l_{12}) = -55.2787°$ Angle between l_3 and MCx-axis
- $\alpha_4 = atan2(-l_{MC}, -l_{12}) = -124.7213°$ Angle between l_4 and MCx-axis
- $\gamma_1 = +45°$ Roller 1 angle
- $\gamma_2 = -45°$ Roller 2 angle
- $\gamma_3 = -45°$ Roller 3 angle
- $\gamma_4 = +45°$ Roller 4 angle
- $m_b = 19.803\ kg$ Base mass
- $m_w = 1.4\ kg$ Wheel mass
- $m_r \approx 0.1\ kg$ Roller mass
- $Width = 0.380\ m$ WMR width
- $Length = 0.580\ m$ WMR length
- $M_T = m_b + 4m_w = 25.403\ kg$ WMR mass
- $r = 0.050\ m$ Wheel Radius
- $r_r = 0.014\ m$ Roller Radius
- $I_T = \frac{1}{2} M_T (Width^2 + Length^2) = 6.107\ kg \cdot m^2$ Moment of inertia of WMR
- $I_r = \frac{1}{2} m_w r^2 = 175 \cdot 10^5\ kg \cdot m^2$ Moment of inertia of wheels
- $I_{rr} = \frac{1}{2} m_r r_r^2 = 98 \cdot 10^7 kg \cdot m^2$ Moment of inertia of rollers
- ${}^R d_{E1} = [l \quad L \quad 0]^T = [158 \quad 228 \quad 0]^T$ Distance from origin of frame E_1 to R
- ${}^R d_{E2} = [-l \quad L \quad 0]^T = [-158 \quad 228 \quad 0]^T$ Distance from origin of frame E_2 to R
- ${}^R d_{E3} = [l \quad -L \quad 0]^T = [158 \quad -228 \quad 0]^T$ Distance from origin of frame E_3 to R
- ${}^R d_{E4} = [-l \quad -L \quad 0]^T = [-158 \quad -228 \quad 0]^T$ Distance from origin of frame E_4 to R
- ${}^R\theta_{E1} = {}^R\theta_{E4} = 45°$ Angle between frames E and R
- ${}^R\theta_{E2} = {}^R\theta_{E3} = -45°$ Angle between frames E and R

5.2 Kinematic modeling with slip for the KUKA youBot

The Kinematic Modeling with slip, given the previous parameters, takes the following form:

$$\begin{aligned} M_T\ddot{x} = {} & \cos(45)\left(F_{fric\,x1} + F_{fric\,x2} + F_{fric\,x3} + F_{fric\,x4}\right) \\ & - \sin(45)\left(F_{fric\,y1} - F_{fric\,y2} - F_{fric\,y3} + F_{fric\,y4}\right) \end{aligned} \tag{5.1}$$

$$\begin{aligned} M_T\ddot{y} = {} & \cos(45)\left(F_{fric\,y1} + F_{fric\,y2} + F_{fric\,y3} + F_{fric\,y4}\right) \\ & + \sin(45)\left(F_{fric\,x1} - F_{fric\,x2} - F_{fric\,x3} + F_{fric\,x4}\right) \end{aligned} \tag{5.2}$$

$$\begin{aligned} I_T\ddot{\theta} = {} & L\cos(45)\left(-F_{fric\,x1} - F_{fric\,x2} + F_{fric\,x3} + F_{fric\,x4}\right) \\ & + l\cos(45)\left(F_{fric\,y1} - F_{fric\,y2} + F_{fric\,y3} - F_{fric\,y4}\right) \\ & + l\sin(45)\left(F_{fric\,x1} + F_{fric\,x2} - F_{fric\,x3} - F_{fric\,x4}\right) \\ & + L\sin(45)\left(F_{fric\,y1} - F_{fric\,y2} + F_{fric\,y3} - F_{fric\,y4}\right) \end{aligned} \tag{5.3}$$

$$I_{r1}\ddot{\varphi}_1 = \tau_1 + r_1(\sin(45)\,F_{fric\,x1} + cos(45)F_{fric\,y1}) - D_r \cdot \dot{\varphi}_1 \tag{5.4}$$

$$I_{r2}\ddot{\varphi}_2 = \tau_2 + r_2(\sin(-45)\,F_{fric\,x2} + cos(-45)F_{fric\,y2}) - D_r \cdot \dot{\varphi}_2 \tag{5.5}$$

$$I_{r3}\ddot{\varphi}_3 = \tau_3 + r_3(\sin(-45)\,F_{fric\,x3} + cos(-45)F_{fric\,y3}) - D_r \cdot \dot{\varphi}_3 \tag{5.6}$$

$$I_{r4}\ddot{\varphi}_4 = \tau_4 + r_4(\sin(45)\,F_{fric\,x4} + cos(45)F_{fric\,y4}) - D_r \cdot \dot{\varphi}_4 \tag{5.7}$$

$$I_{rr1}\ddot{\varphi}_{r1} = r_{r1}F_{fric\,y1} \tag{5.8}$$

$$I_{rr2}\ddot{\varphi}_{r2} = r_{r2}F_{fric\,y2} \tag{5.9}$$

$$I_{rr3}\ddot{\varphi}_{r3} = r_{r3}F_{fric\,y3} \tag{5.10}$$

$$I_{rr4}\ddot{\varphi}_{r4} = r_{r4}F_{fric\,y4} \tag{5.11}$$

where the frictional forces are a function of the torques on the wheels and the wheel velocities $F_{fric} = f_1(\tau_i, \dot{\varphi}_i)$ as described in equation (2.57).

5.3 Simulation results

In this section, we will showcase some simulation results that validate our approach. Three different scenarios are presented. The first presents the kinematic modeling with slip while there are no faulty wheels, the second scenario involves one fault happening after the robot has started moving towards the goal configuration and the third shows the results for two faults happening at the same time the moment the robot begins functioning. All of the simulations were implemented in MATLAB while some 3D graphical representations of the simulation environment including the omni-directional robot, the workspace and the obstacles located within, were developed in the V-REP framework.

The operating workspace along with the static obstacles are modeled as circles. The platform is aware of both the workspace boundaries and the obstacles' positions in it and is assigned a desired goal configuration q_g.

The parameters for the simulations are as follows:

Workspace

- Center $(x_o, y_o) = (0, 0)\ m$
- Radius $\rho_o = 2.5\ m$

Obstacle 1

- Center $(x_1, y_1) = (-0.6, 1.2)\ m$
- Radius $\rho_1 = 0.1\ m$

Obstacle 2

- Center $(x_2, y_2) = (0.4, 0.1)\ m$
- Radius $\rho_2 = 0.2\ m$

Goal configuration

- $q_g = [x_g \quad y_g \quad \theta_g]^T = [-0.5 \quad -1.5 \quad -\pi/6]^T$

System properties

- Limiting friction coefficient $\mu_s = 0.6$

5.3.1 Scenario 1

To verify the behavior of the proposed kinematic model with slip for an omni-directional mobile platform with four mecanum wheels, we will present the following scenario where the robot initializes close to two obstacles within the constrained operational workspace and has to bypass these obstacles towards the desired configuration. This is the nominal case where there are not any faults present.

The main goal of this simulation is to show that the platform can indeed reach the desired final state with the proposed kinematic model and controller design, even though some wheels might experience slippage, while also avoiding collisions with the static obstacles.

The numerical values of the controller parameters are shown below:

Initial robot configuration

- $q_{init} = [x_{init} \quad y_{init} \quad \theta_{init}]^T = [0.5 \quad 1 \quad 0]^T$

Navigation Functions-based Controller gains

- $k = 2$
- $k_{NF} = 2$
- $K = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 50 \end{bmatrix}$

The operational workspace with the obstacles located within, along with the initial and desired position of the robot are depicted in Figure 5-2. The potential field produced by this arrangement that will guide our robot to the goal is shown in Figure 5-3. The yellow color indicates the obstacles and workspace boundaries while the deep blue indicates the desired final position.

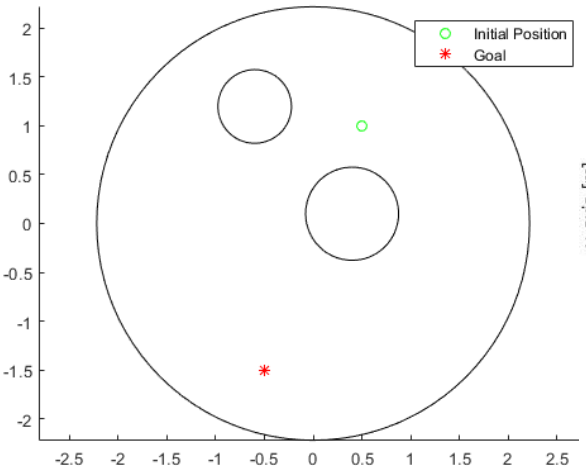

Fig. 5-2 Operational workspace, initial and goal position

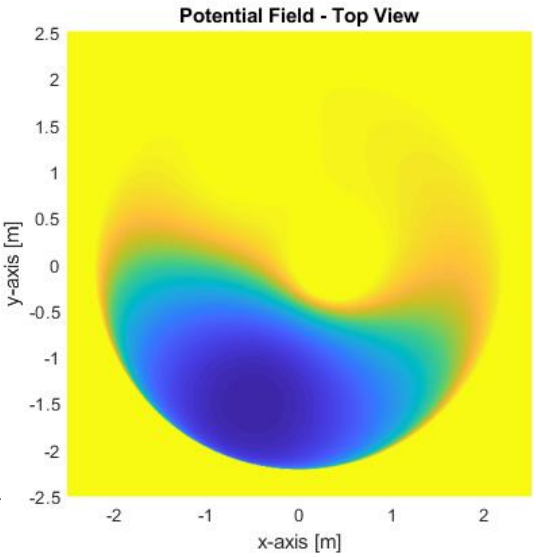

Fig. 5-3 Potential field

In order to make our simulation more realistic and match the experimental setup which will be introduced later on, we decided to place a limit in the torque values that our controller produced for this scenario since in the actual youBot we would be obligated to do so due to physical constraints on the motors. This limit is 5Nm and the actual torque input commands that drive the robot to the goal can be observed in Figure 5-4.

The associated wheel velocities of the mobile platform are depicted in Figure 5-5.

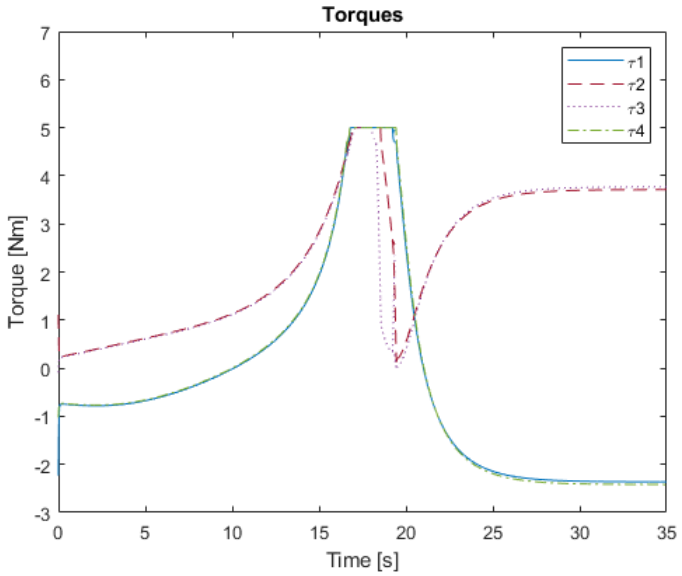

Fig. 5-4 Torques produced by the controller

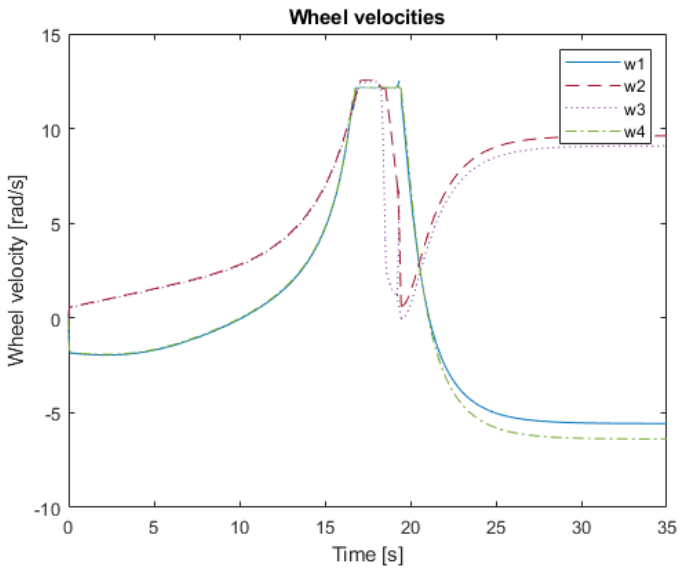

Fig. 5-5 Wheel velocities

Using the aforementioned relationship (equation (2.57)) from the *kinematic modeling with slip* section 2.4, one can observe how the frictional forces evolve with time (Figure 5-6).

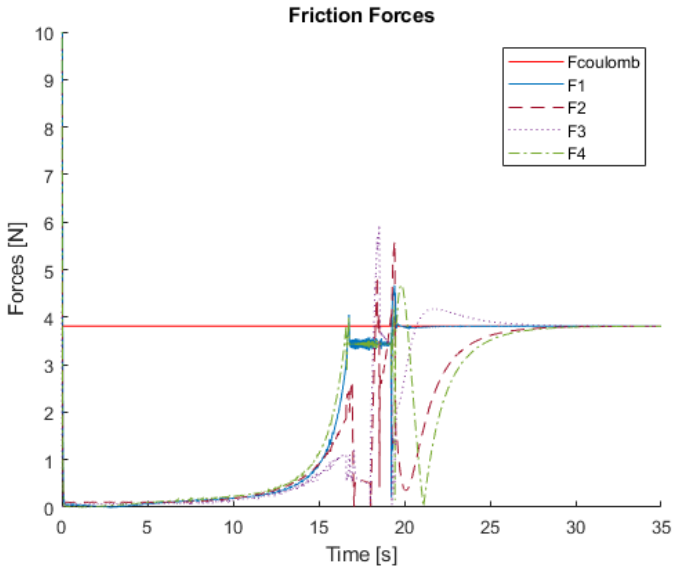

Fig. 5-6 Friction forces

At around t=16s, we observe that the frictional forces on the wheels 1 and 4 have reached the Coulomb friction $F_c = \mu_s F_N$, where μ_s is the limiting friction coefficient. Thus, these wheels have just started slipping. At this point, we change our kinematic model and we replace the no-slip equation with the condition that the frictional force on the slipping wheels is equal to the Coulomb friction force. We are now on the roll with slip mode. After a few milliseconds these frictional forces drop below the F_c threshold, so we go back to the roll mode since none of the wheels are slipping anymore.

Even though we notice that the frictional forces on the wheels might have a value greater than the Coulomb friction F_c, this is just due to the mathematical equations that produce this output. In reality, these forces are bounded by F_c and any value above it should be considered to be equal to F_c.

In the rest of the movement, we observe the frictional forces reaching the Coulomb friction F_c a few times, meaning that the wheels that these forces act upon have started slipping a few times during the vehicle's mission.

The final result of this simulation, the robot path in the x−y plane, is depicted in Figure 5-7, where it can be observed that the vehicle successfully reaches the desired goal configuration while simultaneously avoids the workspace boundaries and the obstacles within. The Navigation Functions-based controller successfully navigates the robotic platform using the proposed kinematic modeling, despite the presence of wheel slippage between the rollers of the wheels and the ground.

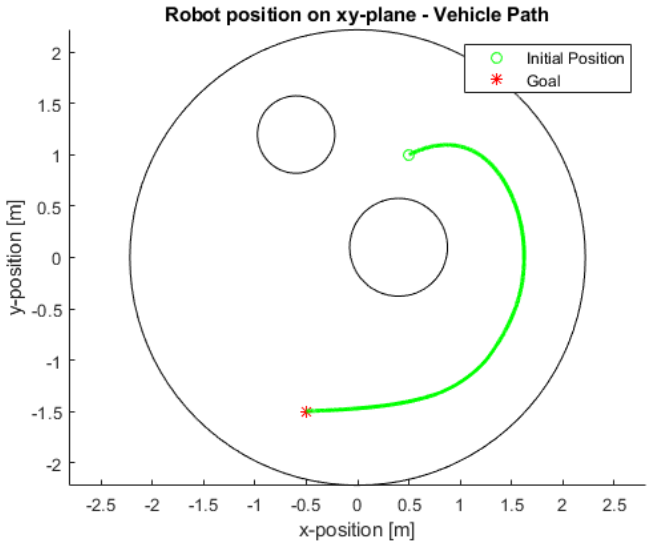

Fig. 5-7 Robot path in x-y plane

A more detailed view of the robot pose state responses versus time (x, y, θ trajectories), where we can clearly observe that the robot has reached the desired configuration $q_g = [x_g \quad y_g \quad \theta_g]^T = [-0.5 \quad -1.5 \quad -\pi/6]^T$, is presented in Figure 5-8.

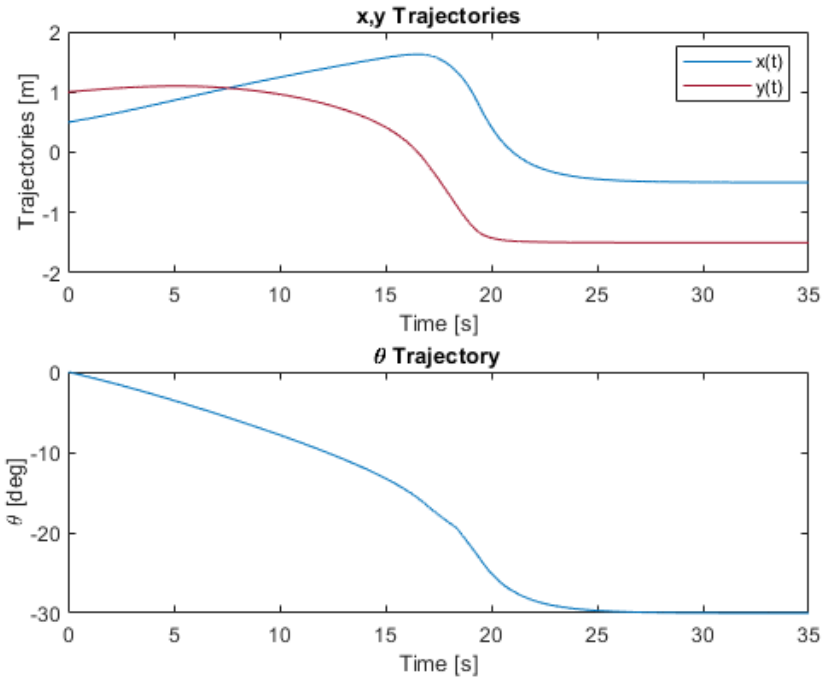

Fig. 5-8 Robot trajectories

The robot's body velocities are depicted in Figure 5-9.

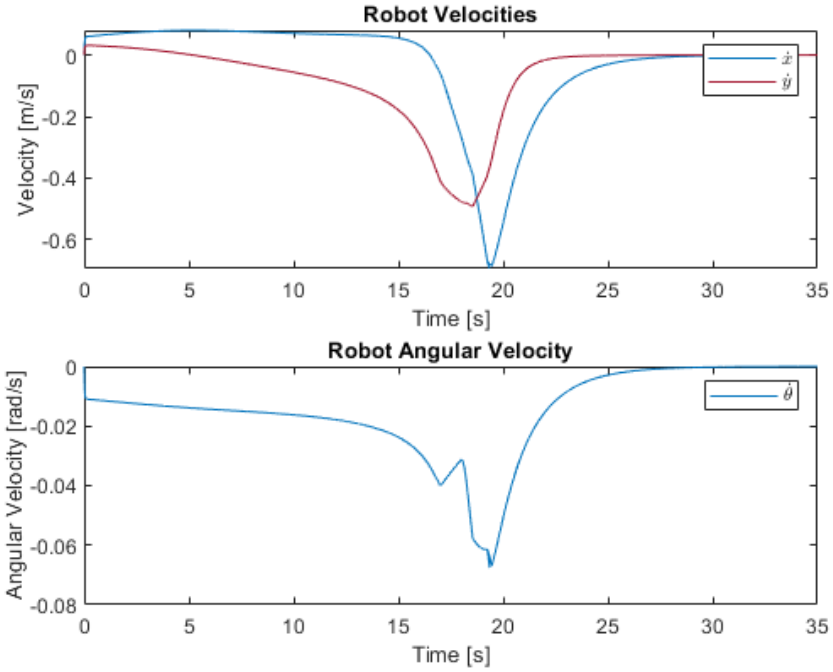

Fig. 5-9 Robot velocities

The 3D graphical representation of the simulation environment in the V-REP framework, along with the robot trajectory, are illustrated in the following Figures. Figure 5-10 shows the initial configuration of the robot, the workspace, obstacles and the goal configuration.

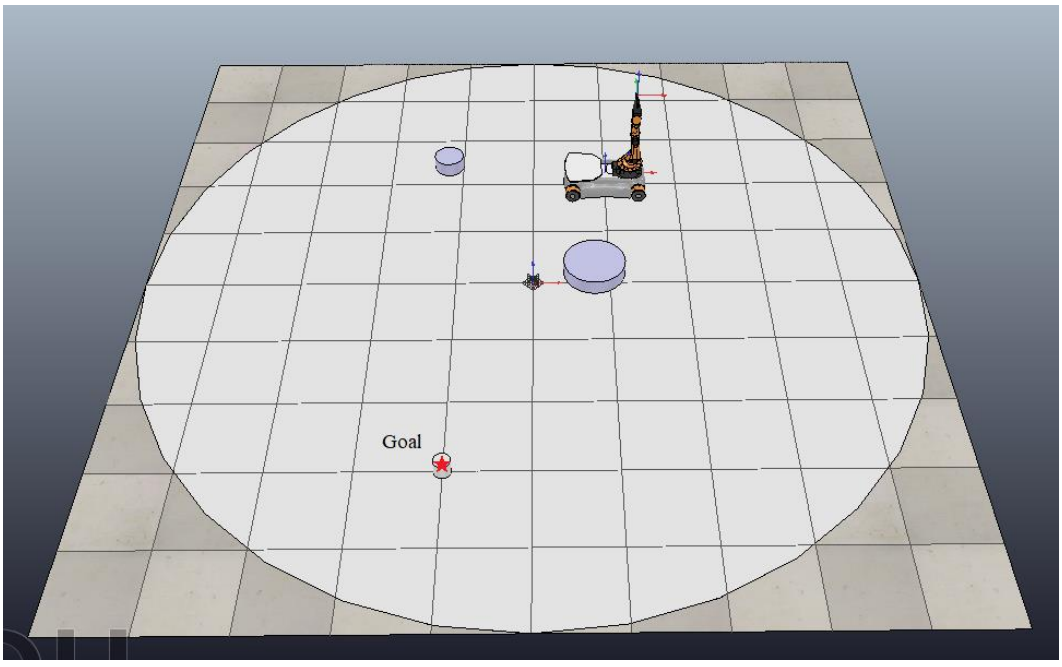

Fig. 5-10 Graphical representation of the simulation environment

Figure 5-11 shows a configuration of the robot in a halfway point during its mission along with the path it has followed up to that point. Figure 5-12 depicts the final configuration and path of the robot, where we can observe that the youBot has indeed reached the goal position and orientation.

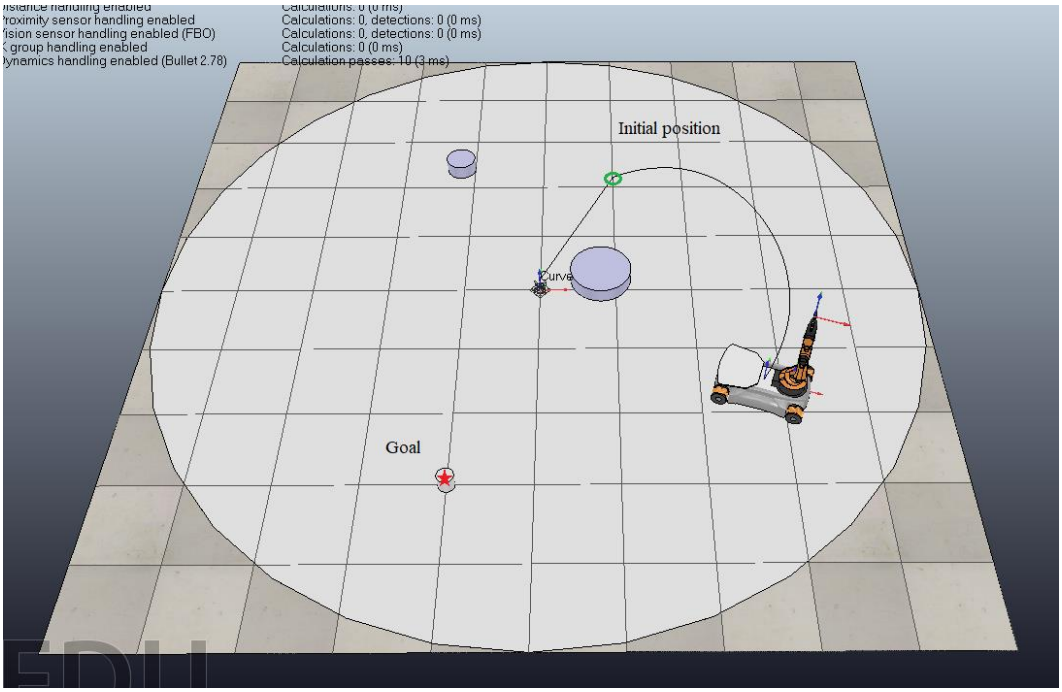

Fig. 5-11 Path of robot halfway during its mission

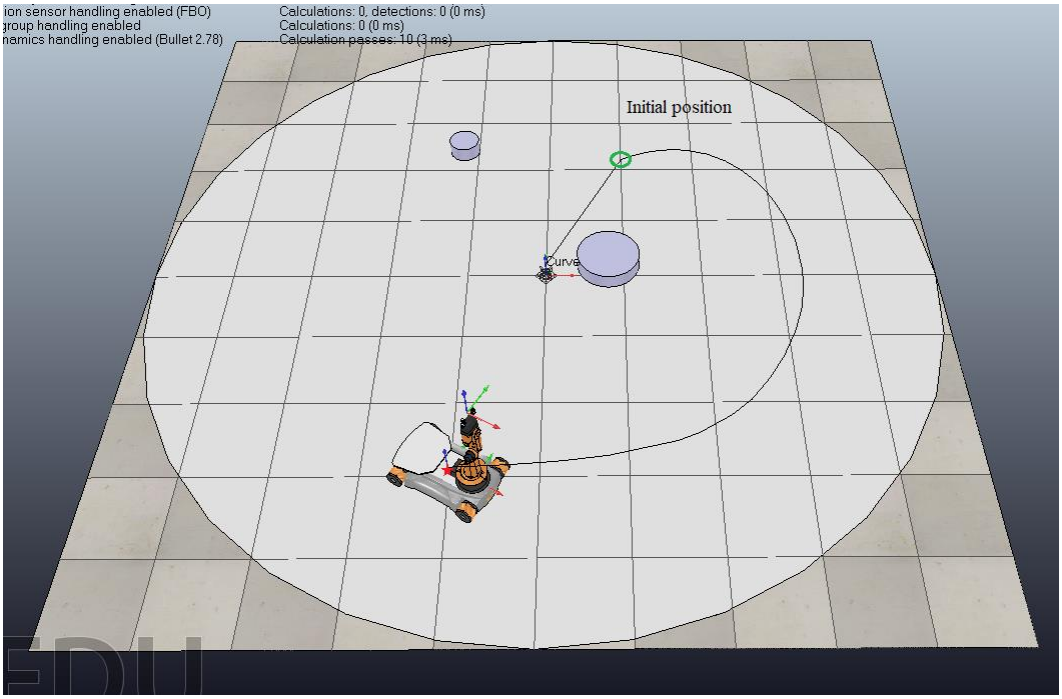

Fig. 5-12 Final configuration and path of the robot

5.3.2 Scenario 2

In the second scenario, we will present the case where when the robot starts its mission all of its components are working properly and suddenly one abrupt fault happens in one of its wheel's motors.

The main goal of this simulation is to show that the platform can indeed reach the desired final state with the proposed kinematic model and controller design even though some wheels might experience slippage, while also avoiding collisions with the static obstacles even in the case where there is one faulty wheel.

When the robot starts moving towards the goal with all of its wheels operating normally, the controller parameters are the same as in the first scenario and are shown below:

Initial robot configuration

- $q_{init} = [x_{init} \quad y_{init} \quad \theta_{init}]^T = [0.5 \quad 1 \quad 0]^T$

Navigation Functions based Controller gains

- $k = 2$
- $k_{NF} = 2$
- $K = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 50 \end{bmatrix}$

Suddenly, at t=5s, a fault appears in wheel 1 of the vehicle. At this point, it should be pointed out that the faulty wheel state was virtually induced by the control software via sending constantly zero torque commands to the corresponding "faulty" wheel.

The controller immediately changes the control gains and the ones selected for that faulty state are presented below:

Navigation Functions based Controller gains for wheel 1 faulty

- $k = 1.2$
- $k_{NF} = 9$
- $K = \begin{bmatrix} 190 & 0 & 0 \\ 0 & 190 & 0 \\ 0 & 0 & 190 \end{bmatrix}$

The torque input commands that drive the robot to the goal are depicted in Figure 5-10. We observe that at the moment the fault happens, increased torque values appear on some wheels because the controller is trying to compensate for that fault but they gradually return to normal. We can also see that the torques for the faulty wheel (wheel 1) are equal to zero after t=5s.

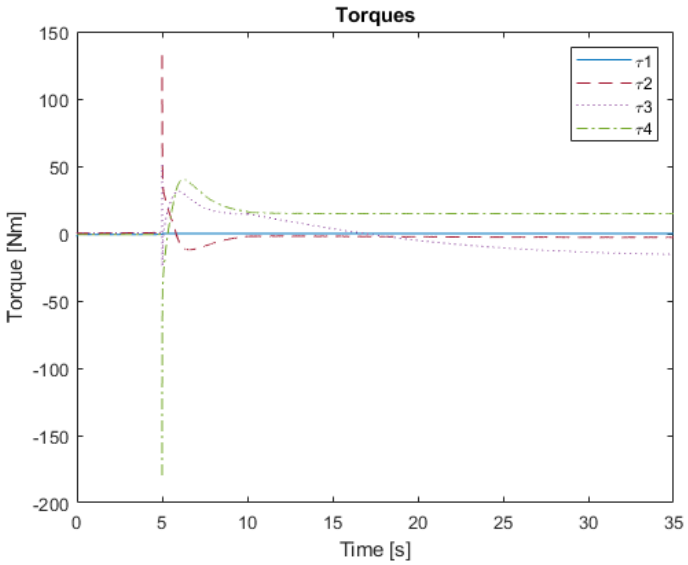

Fig. 5-13 Torques produced by the controller

The associated wheel velocities of the mobile platform are depicted in Figure 5-11.

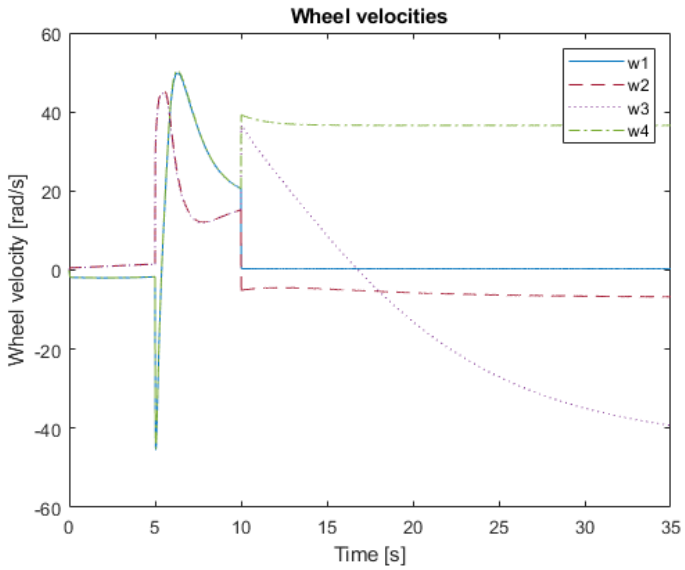

Fig. 5-14 Wheel velocities

The frictional forces on the wheels are shown in Figure 5-12. Until the fault appears, the frictional forces on all wheels are less than the Coulomb friction F_c. Accordingly, there is not wheel slip present. At the time the fault happens though, momentarily all of the wheels start slipping. This is actually the exact behavior that we would expect, since the vehicle is much more prone to slippage when a fault appears in one or more of its wheels. This behavior also verifies the correctness of the proposed modeling.

At t=10s, we again observe a peak in the diagram as all of the friction forces reach the Coulomb friction F_c, which means that all of the wheels experience slippage. If we take a look at the previous plot, Figure 5-11, we can see that at that moment (t=10s), the wheel velocities suddenly increase in absolute value. This phenomenon justifies why the wheels start slipping at this point.

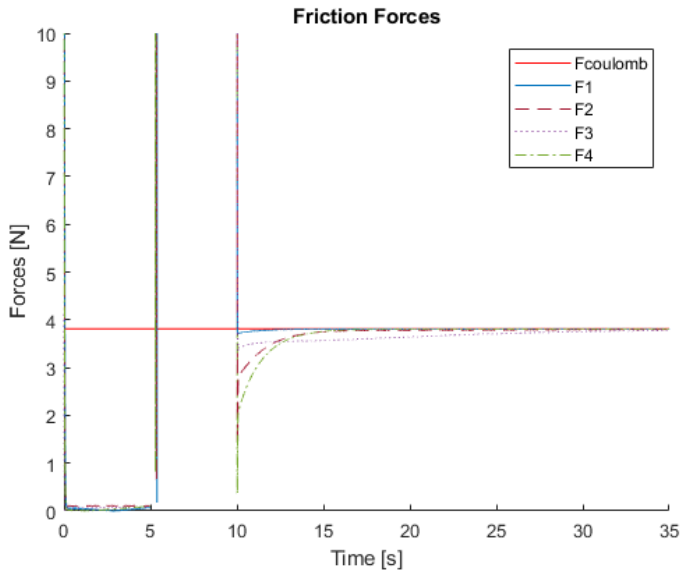

Fig. 5-15 Friction forces

The final result of this simulation, the robot path in the x−y plane, is depicted in Figure 5-13, where it can be observed that the vehicle successfully reaches the desired goal configuration while simultaneously avoids the workspace boundaries and the obstacles within. The Navigation Functions-based controller successfully navigates the robotic platform using the proposed kinematic modeling, despite the presence of wheel slippage between the rollers of the wheels and the ground and despite the presence of one faulty wheel.

We can see that in the beginning stages of the robot's movement, after it has covered a small distance towards the desired final configuration, the fault appears. The position of the robot in the x-y plane that indicates when the fault happened is illustrated with a green star.

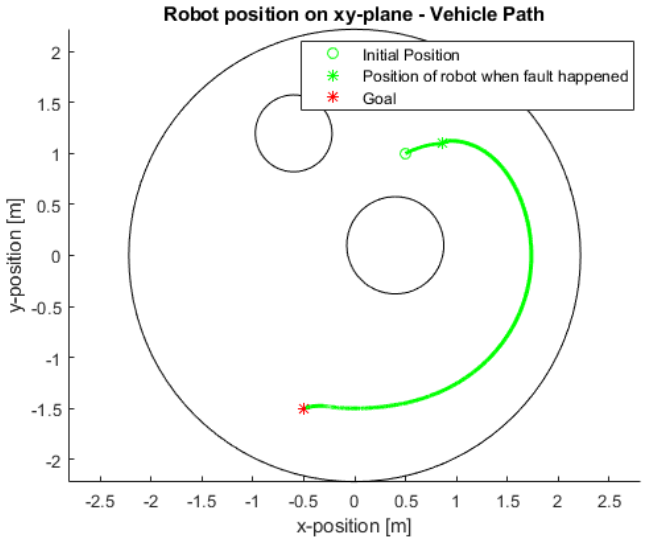

Fig. 5-16 Robot path in x-y plane

A more detailed view of the robot pose state responses versus time (x, y, θ trajectories), where we can clearly observe that the robot has reached the desired configuration $q_g = [x_g \quad y_g \quad \theta_g]^T = [-0.5 \quad -1.5 \quad -\pi/6]^T$, is presented in Figure 5-14.

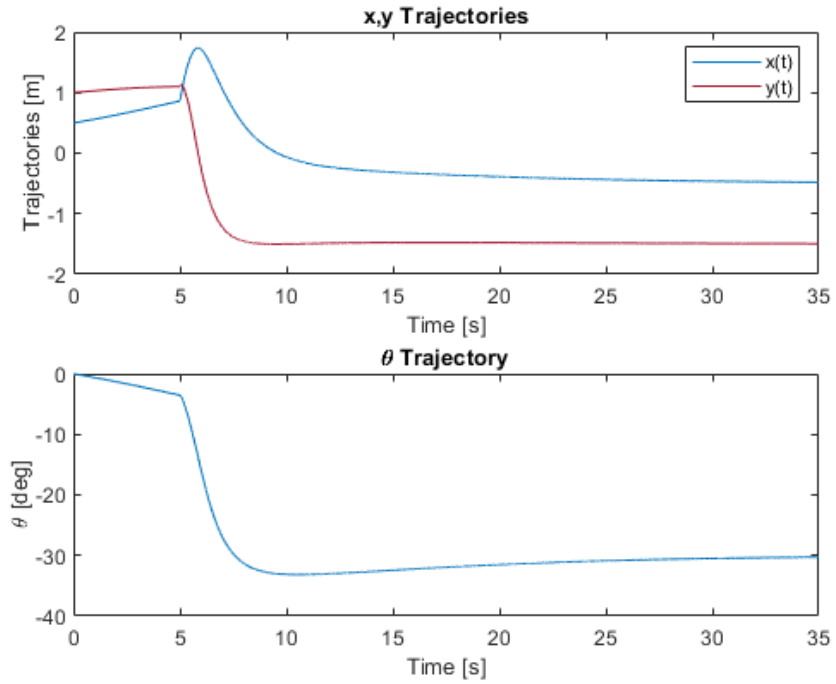

Fig. 5-17 Robot trajectories

The robot's body velocities are depicted in Figure 5-15.

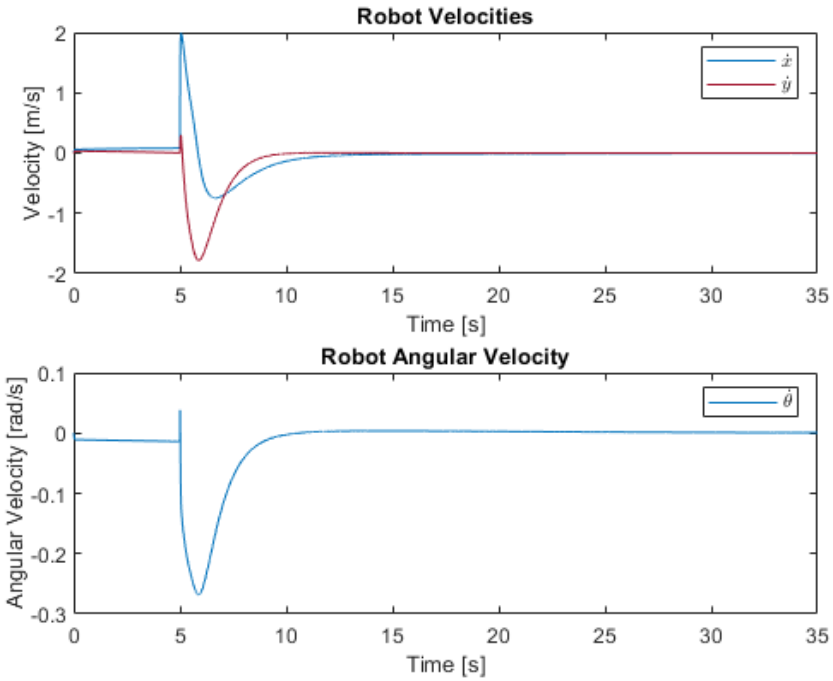

Fig. 5-18 Robot velocities

The 3D graphical representation of this scenario, showing the final configuration and path of the robot where it has reached the goal position and orientation is depicted in Figure 5-19.

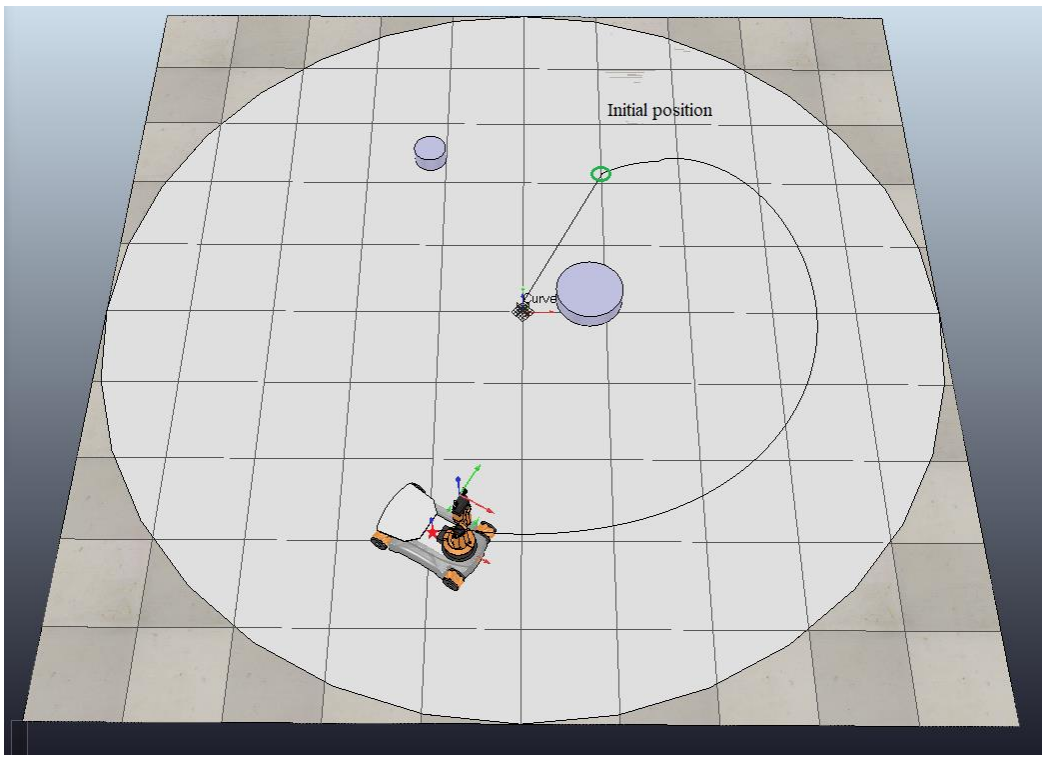

Fig. 5-19 Final configuration and path of the robot

5.3.3 Scenario 3

In the third and final scenario, we will present the case where two faults happen at the same time, the moment that the robot begins functioning. The main goal of this simulation is to show that the platform can indeed reach the desired final position but not necessarily the desired orientation while avoiding collisions with the static obstacles in the case where there are two faulty wheels.

In the occurrences, where there are two faulty wheels, we are only interested to reach the desired final position in the operational workspace irrespectively of the orientation of the platform.

The parameters of this simulation are as follows:

Initial robot configuration

- $q_{init} = [x_{init} \quad y_{init} \quad \theta_{init}]^T = [1.5 \quad 0.5 \quad 0]^T$

Goal configuration

- $q_g = [x_g \quad y_g \quad \theta_g]^T = [-1 \quad -1.5 \quad 0]^T$

Navigation Functions based Controller gains

- $k = 1.3$
- $k_{NF} = 120$
- $K = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix}$

Faulty wheels

- Wheels 2&3

The robot path in the x−y plane, is depicted in Figure 5-16, where it can be observed that the vehicle successfully reaches the desired goal position while simultaneously avoids the workspace boundaries and the obstacles within. The Navigation Functions-based controller successfully navigates the robotic platform despite the presence of two faulty wheels.

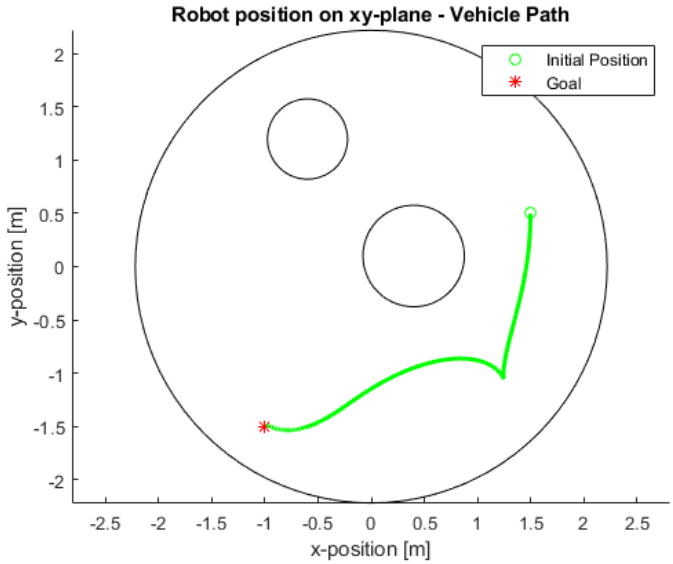

Fig. 5-20 Robot path in x-y plane

The trajectories of the robot are illustrated in the Figures 5-17 and 5-18. We can see that the robot manages to reach the goal position, but as expected, was not able to reach the desired heading angle. The final heading angle of the robot was 65°.

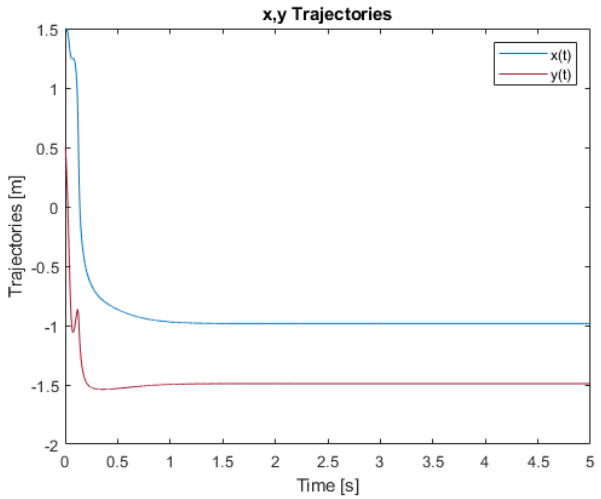

Fig. 5-21 Robot x, y trajectories

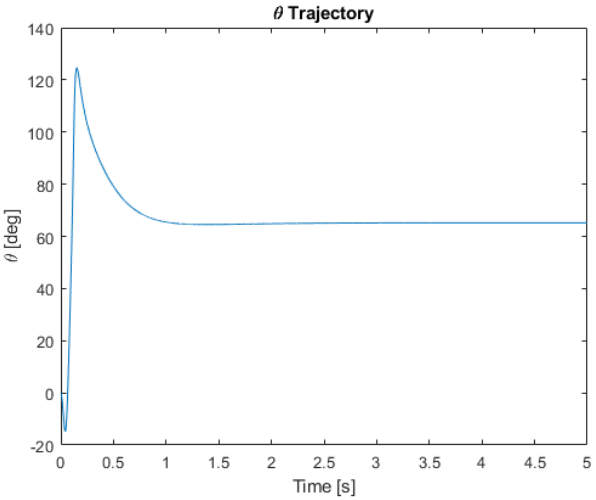

Fig. 5-22 Robot heading angle trajectory

The 3D graphical representation of the 3rd scenario in the V-REP framework, along with the robot trajectory, are illustrated in the following Figures. Figure 5-23 shows the initial configuration of the robot, the workspace, obstacles and the goal configuration.

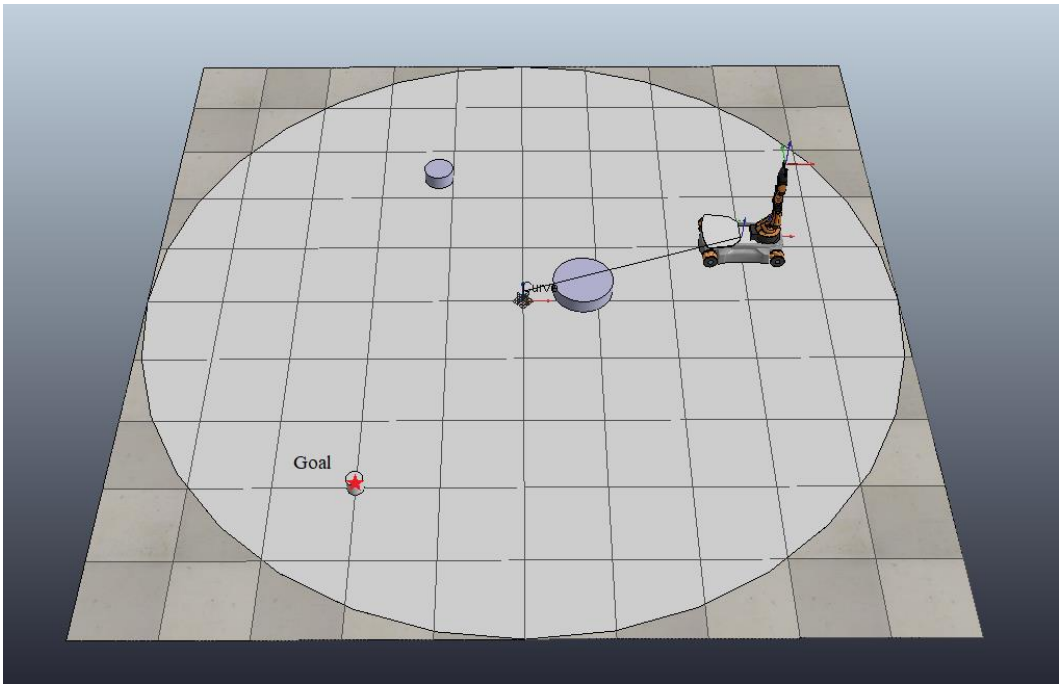

Fig. 5-23 Simulation environment and initial robot configuration

Figures 5-24 and 5-25 show a configuration of the robot in a halfway point during its mission along with the path it has followed up to that point.

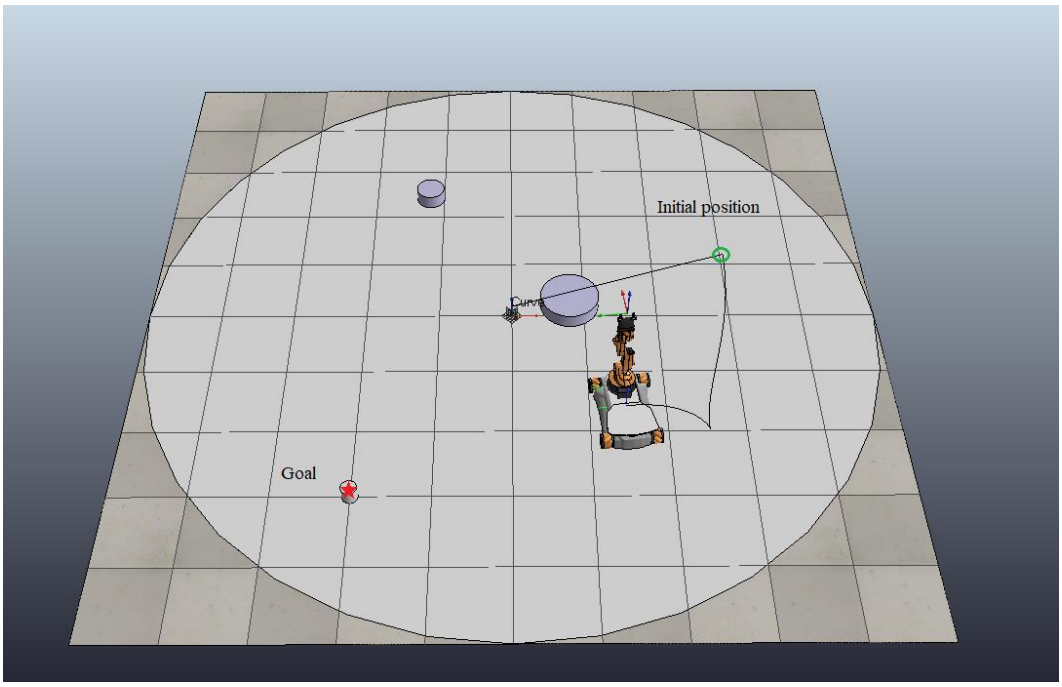

Fig. 5-24 Path of robot halfway during its mission

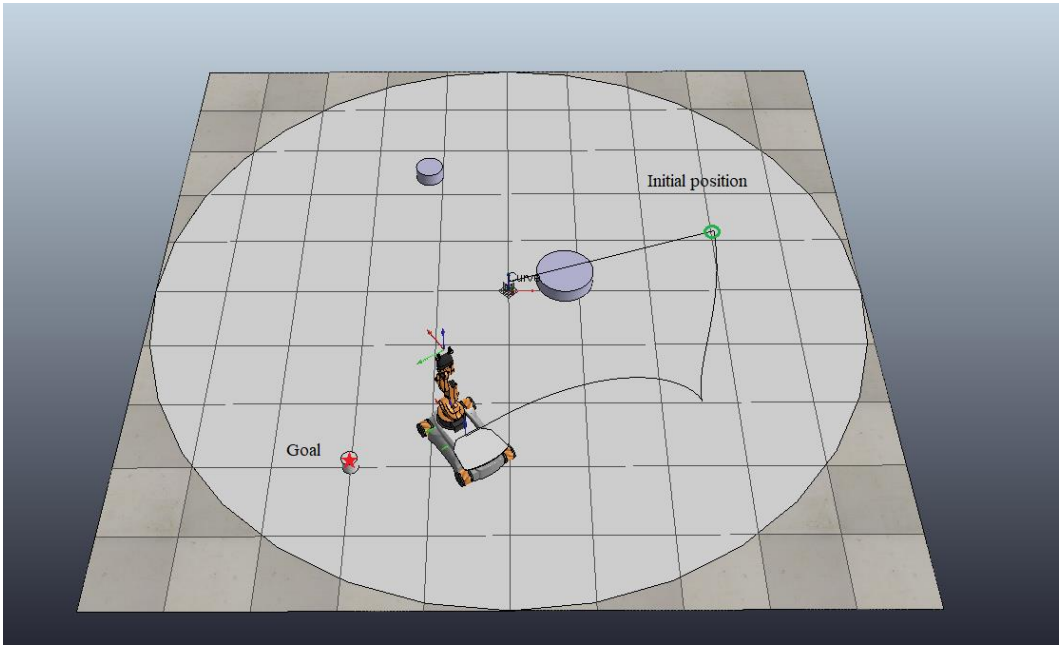

Fig. 5-25 Path of robot halfway during its mission

Figure 5-26 depicts the final configuration and path of the robot, where we can observe that the youBot has indeed reached the goal position but not the goal orientation.

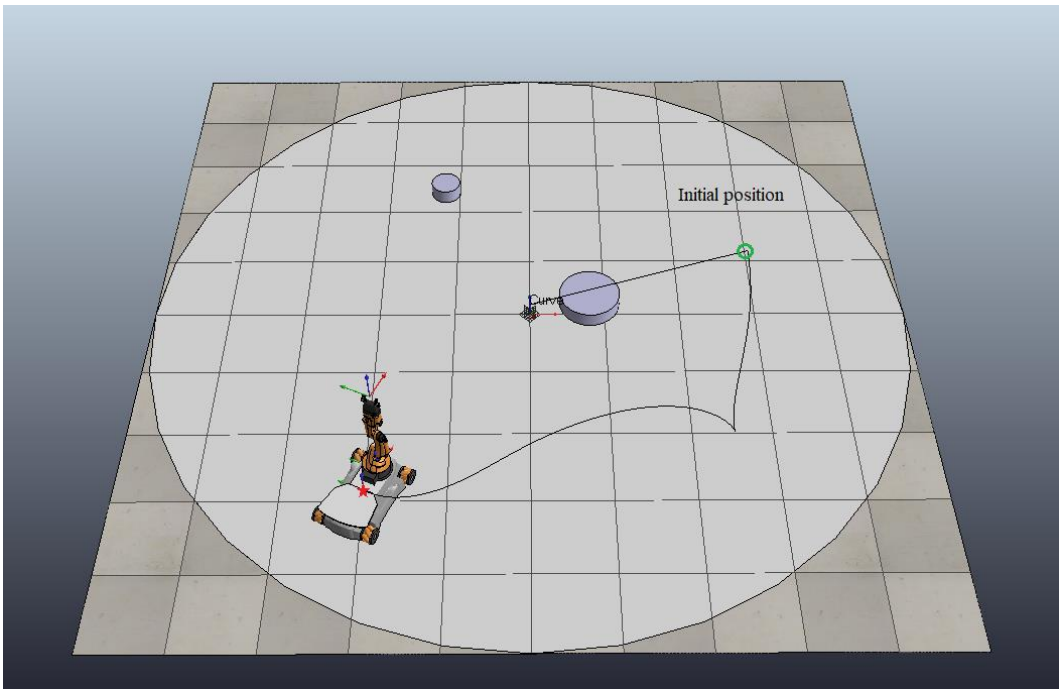

Fig. 5-26 Final configuration and path of the robot

5.4 Robot Operating System (ROS)

The Robot Operating System (ROS) is an open-source, meta-operating system for robots. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

The primary goal of ROS is to support code reuse in robotics research and development. ROS is a distributed framework of processes (aka Nodes) that enables executables to be individually designed and loosely coupled at runtime. These processes can be grouped into Packages and Stacks, which can be easily shared and distributed. ROS currently only runs on Unix-based platforms such as Ubuntu systems and uses the C++, Python and Lisp programming languages.

Computation graph model

ROS processes are represented as nodes in a graph structure, connected by edges called topics. ROS nodes can pass messages to one another through topics, make service calls to other nodes, provide a service for other nodes, or set or retrieve shared data from a communal database called the parameter server. A process called the ROS Master makes all of this possible by registering nodes to itself, setting up node-to-node communication for topics, and controlling parameter server updates. Messages and service calls do not pass through the master, rather the master sets up peer-to-peer communication between all node processes after they register themselves with the master. This decentralized architecture lends itself well to robots, which often consist of a subset of networked computer hardware, and may communicate with off-board computers for heavy computation or commands.

Nodes

A node represents a single process running the ROS graph. Every node has a name, which it registers with the ROS master before it can take any other actions. Multiple nodes with different names can exist under different namespaces, or a node can be defined as anonymous, in which case it will randomly generate an additional identifier to add to its given name. Nodes are at the center of ROS programming, as most ROS client code is in the form of a ROS node which takes actions based on information received from other nodes, sends information to other nodes, or sends and receives requests for actions to and from other nodes.

Topics

Topics are named buses over which nodes send and receive messages. Topic names must be unique within their namespace as well. To send messages to a topic, a node must publish to said topic, while to receive messages it must subscribe. The publish/subscribe model is anonymous: no node knows which nodes are sending or receiving on a topic, only that it is sending/receiving on that topic. The types of messages passed on a topic vary widely and can be user-defined. The content of these messages can be sensor data, motor control commands, state information, actuator commands, or anything else.

Services

A node may also advertise services. A service represents an action that a node can take which will have a single result. As such, services are often used for actions which have a defined beginning and end, such as capturing a single-frame image, rather than processing velocity commands to a wheel motor or odometer data from a wheel encoder. Nodes advertise services and call services from one another.

Parameter server

The parameter server is a database shared between nodes which allows for communal access to static or semi-static information. Data which does not change frequently and as such will be infrequently accessed, such as the distance between two fixed points in the environment, or the weight of the robot, are good candidates for storage in the parameter server.

Basic Nodes in our Application

The Robot Operating System (ROS) will be utilized in our experiments with the real omni-directional mobile platform. The basic nodes that comprise the experimental setup and their primary functions are demonstrated below.

- *cv_camera*: This package uses OpenCV to capture camera images from video devices. Several parameters, such as the frame rate and image width and height can be set in the launch file. It will be used in order to acquire the robot's position and orientation.

- *aruco_detect*: Fiducial detection based on the aruco library. It detects the pose of aruco markers in the camera frame.

- *controller*: The fault tolerant controller which receives robot pose state data and computes the required torques to drive the vehicle to the desired configuration by incorporating Navigation Functions.

- *fault_detection*: The fault diagnosis scheme which detects faults on the actuators of an omni-directional mobile robot with four mecanum wheels.

- *control_for_fault_detection*: Complimentary controller which is used with the fault diagnosis scheme.

- *base_controller*: This node gets the torque commands from the controller and sends them to the motors of the actual youBot platform.

5.5 Experimental results

To demonstrate the efficacy of the proposed kinematic modeling with slip, fault diagnosis scheme and fault tolerant control system, an experimental procedure was carried out using the KUKA youBot omni-directional mobile platform. The KUKA youBot platform consists of the robot chassis, four mecanum wheels, motors, power and an onboard PC board running Ubuntu Linux.

Additionally, a vision system, consisting of a Microsoft camera and a distinctive marker, was used to acquire external position and orientation feedback. Specifically, the camera was mounted on the ceiling of the Lab, observing the workspace of the robotic vehicle and a marker was placed on the top of the vehicle (Figures 5-28, 5-29). The pose of the vehicle with respect to the camera was then calculated using the ArUco library for Augmented Reality applications based on OpenCV.

The software architecture of the integrated system was developed in the Robot Operating System (ROS) and all of the components were programmed using the Python programming language.

The overall fault tolerance architecture is illustrated in Figure 5-27.

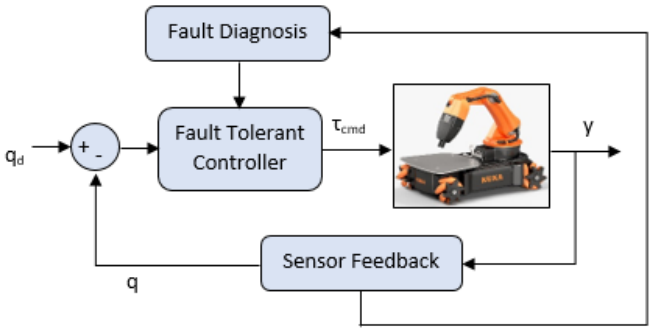

Fig. 5-27 Overall fault tolerance architecture

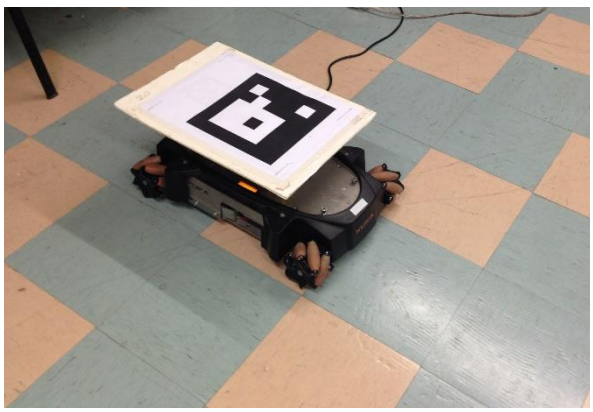

Fig. 5-28 Actual youBot platform with ArUco marker

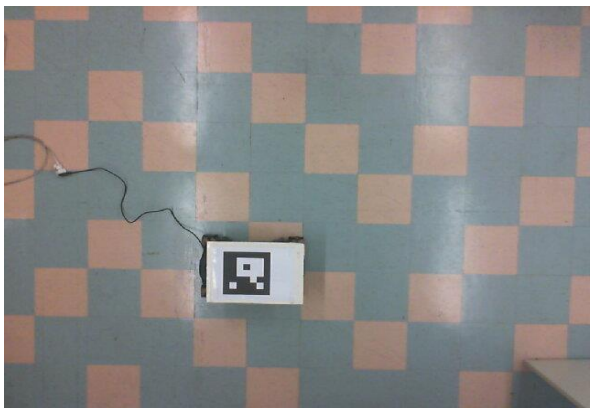

Fig. 5-29 View from camera

Fault Diagnosis Experiments

At first, we tested the fault diagnosis scheme through a series of experiments involving every combination of fault, where we acquired only measurements of the position and orientation of the robotic platform through the camera on the ceiling.

The faulty wheel state was virtually induced by the controller via sending constantly zero torque commands to the corresponding “faulty” wheel.

The robot was programmed to start from an arbitrary pose and follow a straight trajectory. Through the examination of the residuals, the robot was capable of detecting whether there is a fault present and also identifying the wheel or wheels that the fault had occurred on.

In the subsequent Figures, we can observe the results of the fault detection and isolation scheme for some of the faults. Figure 5-30 depicts the results in the case where the front left wheel - wheel 1 - was simulated to be faulty and the detection system expects only one fault. The upper left window depicts the controller which shows that the platform was programmed to move forwards while the fault was induced on wheel 1. The upper right window depicts the fault diagnosis. The platform starts from an arbitrary initial position with a heading angle $\theta \approx 61°$. We can see that the robot detects a deviation from the reference angle and therefore assumes that either wheel 1 or 3 is faulty. By also examining the residuals of its position, it correctly concludes that the fault has occurred in wheel 1. In only a few seconds where the robot has moved only a couple of centimeters, the fault diagnosis scheme was able to correctly identify that a fault has occurred in the front left wheel.

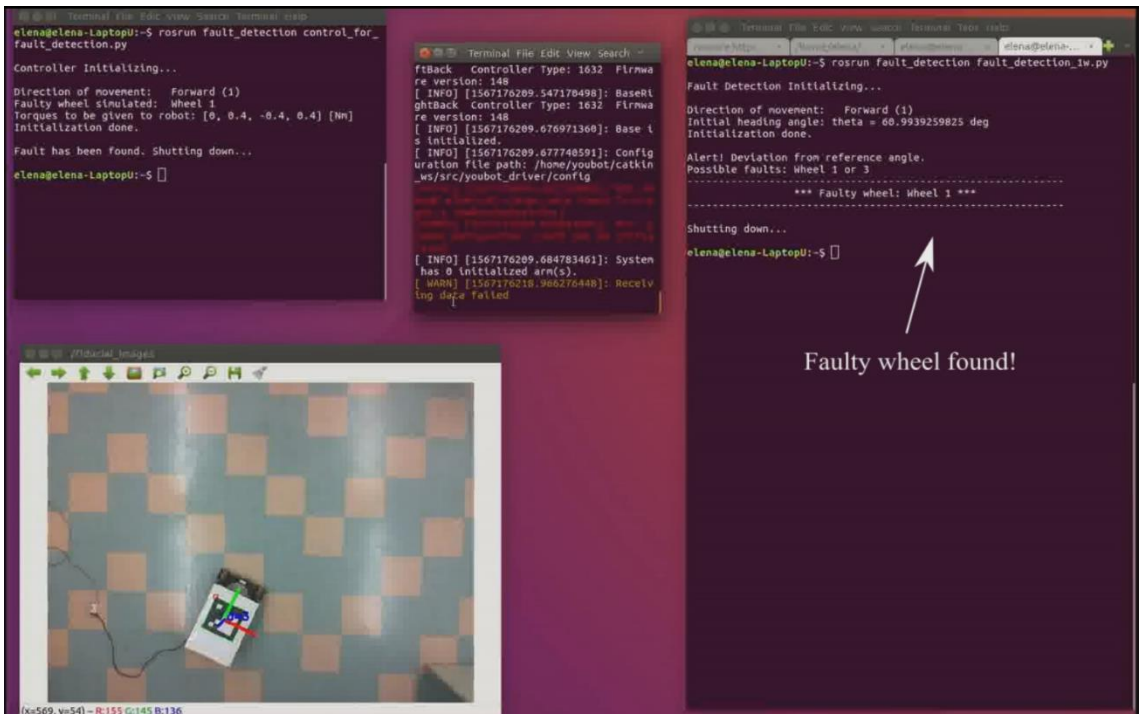

Fig. 5-30 Fault diagnosis for wheel 1 faulty

In the following part, we examine the case where exactly two wheels can become faulty and we apply a fault on the front right and rear right wheels – wheels 2&4. Figure 5-31 depicts the results from the experiment. The controller window shows that the platform was programmed to move forwards while the fault was induced on wheels 2&4 (the torque commands on these wheels was zero). The upper right window depicts the fault diagnosis. The platform starts from an arbitrary initial position with a heading angle $\theta \approx 45°$. We can see that the robot detects a deviation from the reference angle and correctly concludes that the fault has occurred in wheels 2&4 in only a couple of seconds.

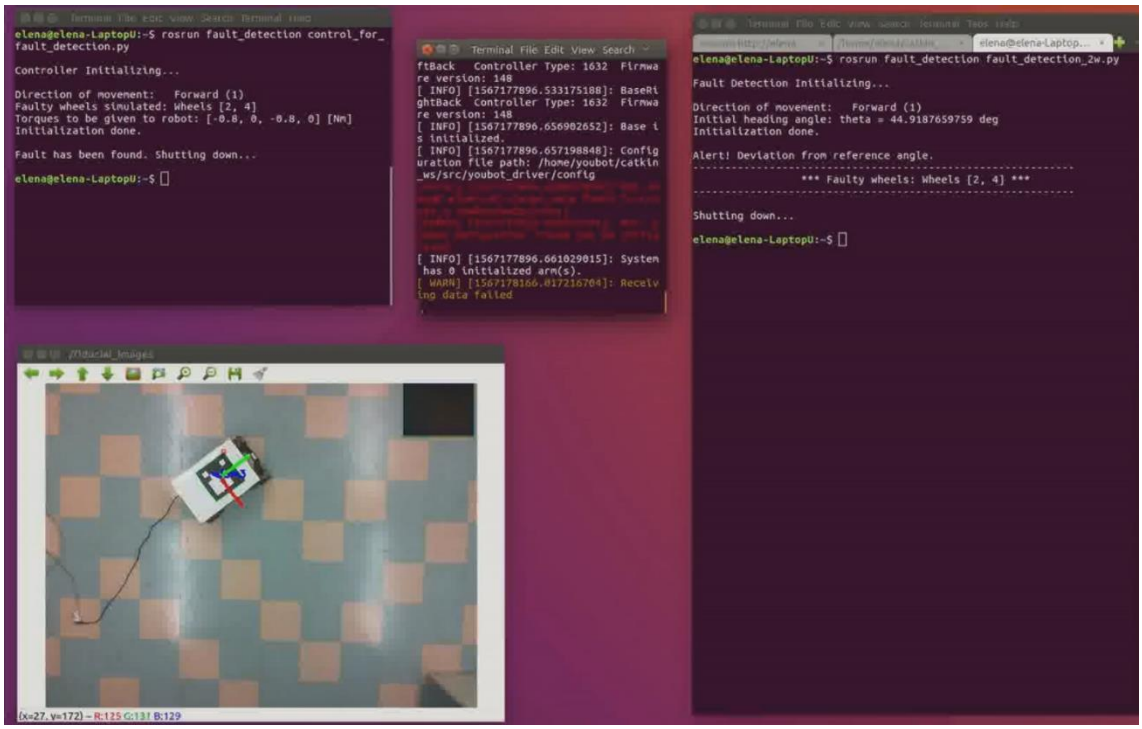

Fig. 5-31 Fault diagnosis for wheels 2&4 faulty

Figures 5-32 and 5-33 illustrate the results in the case where the front left and front right wheels - wheels 1&2 - were simulated to be faulty in the general case where one or two faults are able to occur. The platform starts from an arbitrary initial position with a heading angle $\theta \approx 58°$. By examining the residuals of its position and orientation, the robot does not detect a deviation from the reference angle or the reference trajectory and therefore assumes that either wheels 1&2 or 3&4 might be faulty. In order to identify the possible fault, it then starts moving to the left, where a deviation in the reference angle happens and thus correctly concludes that the fault has occurred in wheels 1&2.

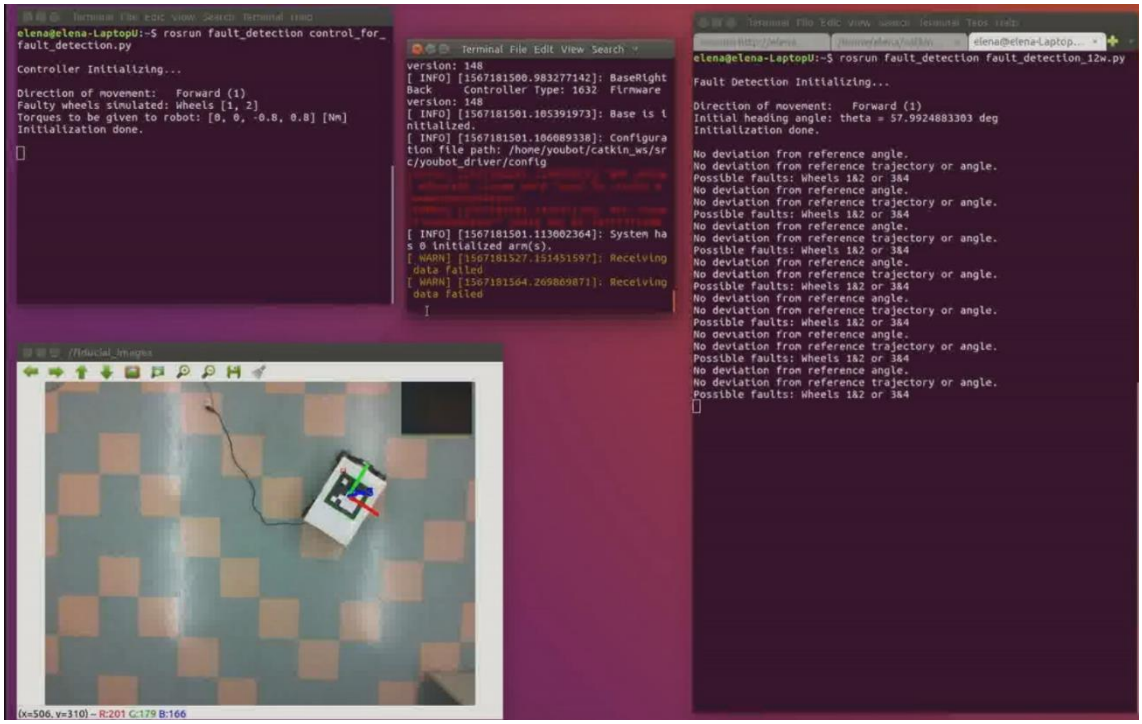

Fig. 5-32 Fault diagnosis for wheels 1&2 faulty (a)

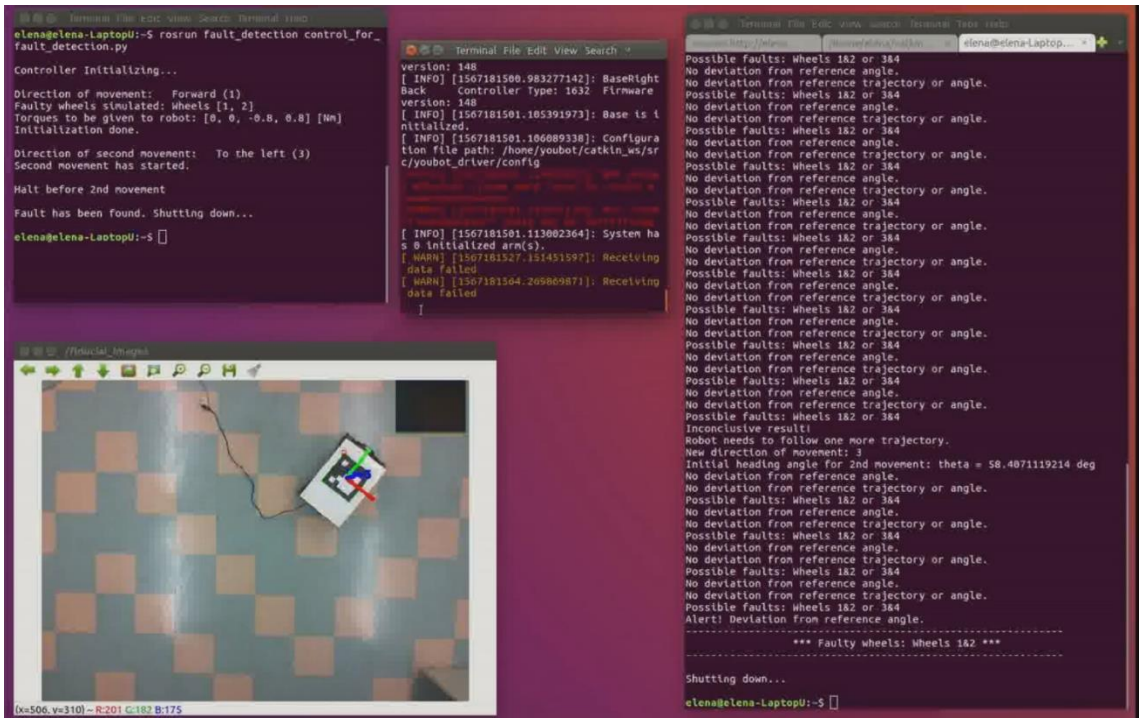

Fig. 5-33 Fault diagnosis for wheels 1&2 faulty (b)

Controller with Slip Modeling Experiments

In this experimental study, we considered the nominal case where there are not any faults present. The operating workspace was modeled as a circle of radius 1.5 m to depict the available space in the Lab where the experiments were performed and a circular obstacle located at (x, y) = (-0.8, 04) m depicting the presence of a chair was also included.

The control gains were selected as follows:

- $k = 2$
- $k_{NF} = 2$
- $K = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 50 \end{bmatrix}$

The robot starts from an initial configuration:

$$q_{init} = [x_{init} \quad y_{init} \quad \theta_{init}]^T \approx [0.513 \quad 0.526 \quad 0.2]^T$$

The goal configuration is set as follows:

$$q_g = [x_g \quad y_g \quad \theta_g]^T = [-0.5 \quad -0.8 \quad 0]^T$$

The experimental results are given below. Figure 5-34 shows the initial pose of the robot, while Figure 5-35 illustrates the final position where the robot has reached the desired configuration q_g.

Notice that with the slip kinematic modeling and the proposed fault tolerance control scheme the robotic platform achieves the desired configuration without colliding with the obstacle or violating the workspace/visual boundaries.

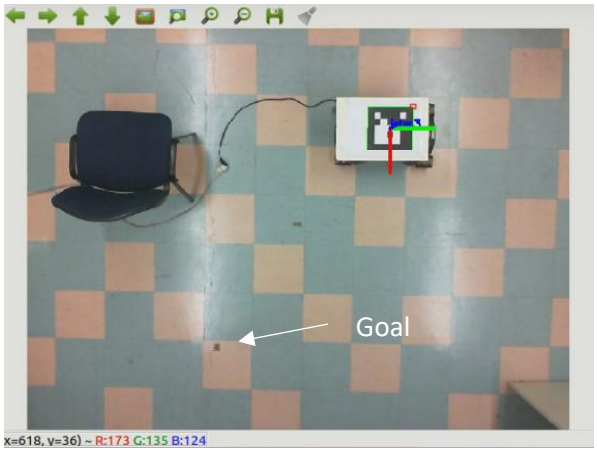

Fig. 5-34 Initial position

Fig. 5-35 Final position

Chapter 6

Conclusion & Future Work

6.1 Conclusion

In this thesis, we investigated the problem of wheel slippage on omni-directional mobile robots with 4 mecanum wheels. We implemented a kinematic modeling with slip, which takes into account the slippage of the wheels after considering the kinematic and dynamic equations that govern our model.

We also introduced a model-based actuator fault diagnosis system for omni-directional mobile robots with 4 mecanum wheels by using structural analysis-based techniques in order to generate residuals. While only utilizing measurements of the robot's position and orientation, we were able to detect actuator faults in one or two wheels and identify the wheels that the fault had occurred on.

We also implemented a fault tolerant control scheme that incorporated Navigation Functions and we were able to make the robotic platform achieve any desired configuration while avoiding collisions with the static obstacles in its workspace, even in the presence of one or two faulty wheels.

Finally, we performed a series of extensive simulations in the MATLAB software and experiments using the omni-directional mobile platform youBot by KUKA that were programmed using the Python programming language and the Robotic Operating System (ROS).

6.2 Future directions

In this section, some possible future extensions to the work conducted in the present thesis and directions for further research are suggested.

For the fault diagnosis framework, a development of a systematic method for threshold selection could lead to faster results, since the selection of thresholds can greatly affect the performance of the detection system.

Further research in the friction coefficients and friction forces that act upon the wheels of an omni-directional vehicle or the introduction of possible measurement devices to compute them numerically, could lead to tremendous improvements on the efficiency of the proposed kinematic modeling with slip.

Improvements in the control scheme, especially in the case where there are two faulty wheels, could have a significant impact in the accuracy of the system as well as a decrease in the overall computational cost. A more efficient gain selection methodology, as another direction for future modifications, would also constitute a great enhancement.

Bibliography

[1] L.Gracia, J. Tornero, “Kinematic modeling of wheeled mobile robots with slip”, Advanced Robotics, Volume 21, No. 7, pp. 1253-1279, 2007

[2] L.Gracia, J. Tornero, “Kinematic modeling and singularity of wheeled mobile robots”, Advanced Robotics, Volume 21, No. 11, pp. 793-816, 2007

[3] M. Blanke, H. Niemann, and T. Lorentzen, “Structural analysis – a case study of the Rømer satellite” in Proc. of IFAC Safeprocess 2003, Washington, DC, USA, 2003.

[4] G.K. Fourlas, S. Karkanis, G.C. Karras, K.J. Kyriakopoulos, “Model based actuator fault diagnosis for a mobile robot” IEEE International Conference on Industrial Technology (ICIT 2014), Busan, Korea, Feb. 26 - Mar. 1, 2014.

[5] A. Monteriù, P. Asthan, K. Valavanis and S. Longhi, “Model-Based Sensor Fault Detection and Isolation System for Unmanned Ground Vehicles: Theoretical Aspects (part I)”, 2007 IEEE International Conference on Robotics and Automation Roma, Italy, 10-14 April, 2007.

[6] A. Monteriù, P. Asthan, K. Valavanis and S. Longhi, “Model-Based Sensor Fault Detection and Isolation System for Unmanned Ground Vehicles: Experimental Validation (part II), 2007 IEEE International Conference on Robotics and Automation Roma, Italy, 10-14 April, 2007.

[7] G.C. Karras, G.K. Fourlas, “Model Predictive Fault Tolerant Control for Omni-directional Mobile Robots”, Journal of Intelligent & Robotic Systems, 2019

[8] Steinbauer G. “A Survey about Faults of Robots Used in RoboCup”. In: Chen X., Stone P., Sucar L.E., van der Zant T. (eds) RoboCup 2012: Robot Soccer World Cup XVI. RoboCup 2012. Lecture Notes in Computer Science, vol 7500. Springer, Berlin, Heidelberg, 2013

[9] P. Vlantis, C.P. Bechlioulis, G.C. Karras, G.K. Fourlas, K.J. Kyriakopoulos, “Fault tolerant control for omni-directional mobile platforms with 4 mecanum wheels” IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, May 16 – 21, 2016.

[10] M. Jung, J. Kim, “Development of a Fault-Tolerant Omnidirectional Wheeled Mobile Robot Using Nonholonomic Constraints”, The International Journal of Robotics Research, Volume 21, No. 5-6, May-June 2012

[11] M. Hashimoto, H. Kawashima, F. Oba, “A Multi-Model Based Fault Detection and Diagnosis of Internal Sensor for Mobile Robot”, Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems Las Vegas, Nevada, October 2003.

[12] G. Ippoliti, S. Longhi, A. Monteriμu, “Model-based sensor fault detection system for a smart wheelchair”, IFAC 2005.

[13] H. Taheri, B. Qiao, N. Ghaeminezhad, “Kinematic Model of a Four Mecanum Wheeled Mobile Robot”, International Journal of Computer Applications, Volume 113, No. 3, March 2015

[14] J.P. Mendoza, M.M. Veloso, R.G. Simmons, “Mobile Robot Fault Detection based on Redundant Information Statistics”, 2012

[15] A. Kuestenmacher, P.G. Ploger, “Model-Based Fault Diagnosis Techniques for Mobile Robots”, International Federation of Automatic Control (IFAC), 2016

[16] D. Crestani, K. Godary-Dejean, L. Lapierre, “Enhancing Fault Tolerance of Autonomous MobileRobots”, Robotics and Autonomous Systems, Volume 68, pages 140-155, June, 2015

[17] D. Rotondo, V.Puig, F.nejjari, J. Romera, “A Fault-Hiding Approach for the Switching Quasi-LPV Fault-Tolerant Control of a Four-Wheeled Omnidirectional Mobile Robot”, IEEE Transactions on Industrial Electronics, Volume 62, June 2015

[18] F. G. Pin and S. M. Killough, “New family of omnidirectional and holonomic wheeled platforms or mobile robots,” IEEE Transactions on Robotics and Automation, vol. 10, no. 4, pp. 480–489, 1994.

[19] G. Campion, G. Bastin, and B. D’Andra-Novel, “Structural properties and classification of kinematic and dynamic models of wheeled mobile robots,” IEEE Transactions on Robotics and Automation, vol. 12, no. 1, pp. 47–62, 1996.

[20] J. Tang, K. Watanabe, and Y. Shiraishi, “Design and traveling experiment of an omnidirectional holonomic mobile robot,” in IEEE International Conference on Intelligent Robots and Systems, vol. 1,

1996, pp. 66–73.

[21] H. Oliveira, A. Sousa, A. Moreira, and P. Costa, “Modeling and assessing of omni-directional robots with three and four wheels,” in Contemporary Robotics - Challenges and Solutions, A. Rodi, Ed., 2009.

[22] H. Kim and B. K. Kim, “Online minimum-energy trajectory planning and control on a straight-line path for three-wheeled omnidirectional mobile robots,” IEEE Transactions on Industrial Electronics, vol. 61, no. 9, pp. 4771–4779, 2014.

[23] R. L. Williams II, B. E. Carter, P. Gallina, and G. Rosati, “Dynamic model with slip for wheeled omnidirectional robots,” IEEE Transactions on Robotics and Automation, vol. 18, no. 3, pp. 285–293, 2002.

[24] M. M. Olsen and H. G. Petersen, “A new method for estimating parameters of a dynamic robot model,” IEEE Transactions on Robotics and Automation, vol. 17, no. 1, pp. 95–100, 2001.

[25] M. . Jung and J. . Kim, “Development of a fault-tolerant omnidirectional wheeled mobile robot using nonholonomic constraints,” International Journal of Robotics Research, vol. 21, no. 5-6, pp. 527–539, 2002.

[26] S. Yin, H. Luo, and S. X. Ding, “Real-time implementation of fault-tolerant control systems with performance optimization,” IEEE Transactions on Industrial Electronics, vol. 61, no. 5, pp. 2402–2411, 2014.

[27] Y. Zhang and J. Jiang, “Bibliographical review on reconfigurable fault tolerant control systems,” Annual Reviews in Control, vol. 32, no. 2, pp. 229–252, 2008.

[28] J. Jiang and X. Yu, “Fault-tolerant control systems: A comparative study between active and passive approaches,” Annual Reviews in Control, vol. 36, no. 1, pp. 60–72, 2012.

[29] D. E. Koditschek and E. Rimon, “Robot navigation functions on manifolds with boundary,” Advances in Applied Mathematics, vol. 11, no. 4, pp. 412–442, 1990.

[30] H. G. Tanner and K. J. Kyriakopoulos, “Nonholonomic motion planning for mobile manipulators,” IEEE International Conference on Robotics and Automation, vol. 2, 2000, pp. 1233–1238.

[31] J. Carlson and R. R. Murphy, “How ugvs physically fail in the field,” IEEE Transactions on Robotics, vol. 21, no. 3, pp. 423–437, 2005.

[32] E. Rimon and D. E. Koditschek, “Exact robot navigation using artificial potential functions,” IEEE Transactions on Robotics and Automation, vol. 8, no. 5, pp. 501–518, 1992.

[33] J. Milnor, “Morse Theory, Annals of Mathematics Studies”, Princeton University Press: NJ, Princeton, 1963.

[34] S. G. Loizou and K. J. Kyriakopoulos, “Closed loop navigation for multiple non-holonomic vehicles”, IEEE International Conference on Robotics and Automation, vol. 3, 2003, pp. 4240–4245.

[35] I. Doroftei, V. Grosu, V. Spinu, “ Omnidirectional Mobile Robot - Design and Implementation”, Bioinspiration and Robotics Walking and Climbing Robots, 10.5772/5518, 2007.

[36] P. Viboonchaichep, A. Shimada, Y. Kosaka, “Position rectification control for mecanum wheeled omni-directional vehicles”, 2003.

[37] E. Velenis, P. tsiotras, C. Canudas-de-wit, M. Sorine, “Dynamic tyre friction models for combined longitudinal and lateral vehicle motion”, Vehicle System Dynamics, Volume 43, No. 1, January 2005.

[38] N. Seegmiller, A. Kelly, “Enhanced 3D kinematic modeling of wheeled mobile robots”, Robotics: Science and Systems Conference, 2014.

[39] K. Han, H. Kim, J.S. Lee, “The sources of position errors of omni-directional mobile robot with mecanum wheel”, 2010.

[40] Y.Maddahi, A. Maddahi, N. Sepehri, “Calibration of omnidirectional wheeled mobile robots: method and experiments”, Robotica, Volume 31, pp. 969-980, 2013.

[41] R.C.Luo, Y. Tsai, “On-line adaptive control for minimizing slippage error while mobile platform and manipulator operate simultaneously for robotics mobile manipulation”, IECON, 2015.

[42] B. Croft, C. Jones, D. Thompson, “Velocity-dependent friction in a model of wheel-rail rolling contact and wear”, Vehicle System Dynamics, Volume 49, No. 11, November 2011.

[43] X. Claeys, J. Yi, L. Alvarez, R. Horowitz, C. Canudas de Wit, “A dynamic tire/road friction model for 3D vehicle control and simulation”, IEEE Intelligent Transportation Systems Conference, 2001.

[44] R. Balakrishma, A. Ghosal, “Modeling of slip for wheeled mobile robots”, IEEE Transactions on Robotics and Automation, vol. 11, no. 1, February 1995.

[45] W. Kim, B.J. Yi, D. J. Lim, “Kinematic modeling of mobile robots by transfer method of augmented generalized coordinates”, Journal of Robotic Systems, Volume 21, Issue 6, 2004.