A RECONFIGURABLE ROBOT TO OVERCOME AN OBSTACLE SCENARIO USING PREDICTIVE FUNCTIONAL CONTROL

Edison Efrain Alfaro Paucarchuco

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Fernando Cesar Lizarralde

Rio de Janeiro
Julho de 2022

A RECONFIGURABLE ROBOT TO OVERCOME AN OBSTACLE SCENARIO USING PREDICTIVE FUNCTIONAL CONTROL

Edison Efrain Alfaro Paucarchuco

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Fernando Cesar Lizarralde, D.Sc.

Prof. Ramon Romankevicius Costa, D.Sc.

Prof. Gustavo Medeiros Freitas, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
JULHO DE 2022

Alfaro Paucarchuco, Edison Efrain

A Reconfigurable Robot to Overcome an Obstacle Scenario using Predictive Functional Control/Edison Efrain Alfaro Paucarchuco. – Rio de Janeiro: UFRJ/COPPE, 2022.

XI, 73 p.: il.; 29, 7cm.

Orientador: Fernando Cesar Lizarralde

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2022.

Referências Bibliográficas: p. 62 – 67.

1. Reconfigurable robot. 2. Vertical obstacles. 3. Predictive functional control. 4. LiDAR. I. Lizarralde, Fernando Cesar. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM ROBÔ RECONFIGURÁVEL PARA SUPERAR UM CENÁRIO DE OBSTÁCULO USANDO O CONTROLE FUNCIONAL PREDITIVO

Edison Efrain Alfaro Paucarchuco

Julho/2022

Orientador: Fernando Cesar Lizarralde

Programa: Engenharia Elétrica

Neste trabalho, apresentamos um robô reconfigurável para lidar com uma missão de inspeção em ambientes industriais, que devido ao seu elevado número de peças articuladas, acarreta uma carga de trabalho cognitiva para o operador tornando-se necessário alcançar certa autonomia. Esta tese enfoca os procedimentos que o robô deve realizar para chegar ao ponto de inspeção, nas indústrias de mineração e petróleo, onde foram encontrados obstáculos verticais recorrentes. Para enfrentar o problema, foram consideradas duas subtarefas principais; Primeiramente, uma navegação por waypoints em que algoritmos SLAM foram empregados para fazer o mapeamento e localização no cenário e determinar o caminho de inspeção. E segundo, uma sequência de movimento baseada em regras de controle para as articulações ativas do robô para superar obstáculos verticais definidos em que um critério de mobilidade é combinado com a técnica de controle funcional preditivo. As simulações foram realizadas no ambiente *CoppeliaSim*, e os testes experimentais em um wheel/tracked robô móvel.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

Edison Efrain Alfaro Paucarchuco

July/2022

Advisor: Fernando Cesar Lizarralde

Department: Electrical Engineering

In this work, we present a reconfigurable robot to deal with inspection missions on industrial environments, This thesis focuses on the inspection task for mining and oil industries where recurrent vertical obstacles is found. This thesis focuses on the procedure which the robot has to perform to reach an inspection point, on mining & oil industries, where recurrent vertical obstacles is found. To face the problem, two main sub-tasks are considered; First, a navigation via waypoints in which SLAM algorithms are employed to make the mapping and localization on the scenario and determine the inspection path. Second, a motion sequence based on control rules for the robot active joints to overcome defined vertical obstacles in which a mobility criteria is combined with the predictive functional Control technique. Simulations were performed on the *CoppeliaSim* environment, and the experimental tests on a wheel/tracked mobile robot namely Rosi.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mobile robots have achieved impact on a wide range of industrial applications. Generally, they acquire information from the surroundings via sensors, and plan and execute a certain task without external intervention. Nowadays, because of various industrial equipment require maintenance procedures, the use of robots for inspection task like; read sensors, images, test vibration, etc. is increasing.

However, due to the complexity on the perception of scenarios and on the control of mechanisms, there still exists a gap to complete inspection autonomously. Regarding the navigation task, a wide field in robotics, some relevant concepts are necessary to performed the navigation. First, its required to have a map environment where the robot performs the navigation. Second, the robot needs to know in which position, and orientation of the map is located at each moment - Localization. Third, known the map and the localization, a waypoint planning procedure which guide the robot where to go, and how to go there its needed. Regards the articulated mechanisms which embrace; elevator platforms, legged robots, cars with active suspension, flippers, etc. all of them as devices to deal with scenarios that have ladders, steps, barriers, etc. Most of this mechanism are linked to the body robot, increasing the number of joints to be remotely controlled. This fact entails a cognitive workload for the operator, making it necessary to achieve a certain autonomy.

This work involves with the operation of a reconfigurable robot wheel/tracked to deal an inspection task on an industrial environment (mining & oil), where recurrent vertical obstacles can be found. On the way to achieve an autonomous inspection of our reconfigurable robot in a scenario that has a vertical obstacle, two main challenges were defined:

- **Navigate a scenario:** To steer the reconfigurable robot (wheel mode) to the desired inspection point.
- **Motion sequence:** For the flippers to overcome a vertical obstacle.

Regarding the navigation task, existing libraries of SLAM algorithms based on point clouds are tested on a real robot, with the objective to make mapping and localization over the scenario and determine the inspection path on a first task. Having the trajectory, a simple proportional controller through a sequence of waypoints can be used on the next inspection task. With respect to a motion sequence for the flippers of our reconfigurable robot to overcome a limited vertical obstacle. A mobility criteria combined with an optimal control approach is proposed. And, due to the tracked flippers are linked to the robot body, its possible that a coordinated movement allow us to regulate the relative height of the robot. Then, we define some states based on control rules for the flippers to accomplish to overcome the obstacles.

Experiments regarding the mapping and localization were performed with our reconfigurable robot-ROSI, we used SLAM algorithms in indoor and outdoor scenarios getting the inspection path. With respect to the obstacle task; simulations were performed on the *CoppeliaSim* environment, we tested with different shapes of obstacles, then we transfer some parameters to the real experiment on ROSI robot.

1.1 Research Objectives

This research aims to design, test, and validate an inspection mission procedure for a reconfigurable robot. The main objectives of this work include:

- Implementation of SLAM algorithms on Rosi robot, test it on various scenarios to obtain the map, trajectory, etc.
- Compare the inspection path obtained from several sensors; Lidar, IMU, Camera.
- Design, test and validate the formulation of predictive functional control approach with a mobility criteria equation.
- Design, test and validate a flippers motion based on control rules for the Rosi robot, to overcome defined vertical obstacles, simulations and real experiments will be performed .

1.2 Dissertation Organization

This thesis has made efforts to develop an standard solution for a reconfigurable robot to deal with an inspection task. For this purpose, these work is compost into the following chapters:

- Chapter 2 - gives a literature review on the procedures that are involved on a robot inspection mission.
- Chapter 3 - presents general concepts of rigid bodies, and the kinematic theory involved in wheel/tracked robots. Furthermore, gives the mobility criteria equations to be considered.
- Chapter 4 - elaborates the perception techniques for laser sensors to perceive the robot surroundings. Moreover, gives the proceeding to overcome obstacles, and our PFC design is elaborated.
- Chapter 5 - gives simulations and experimental results for the navigation and the obstacle task.
- chapter 6 - concludes this thesis and provides some recommendations and possible future work arising from this thesis.

Chapter 2

Literature Review

This chapter contains relevant information concerning to a robot inspection task on industrial sites. The focus of the concentration is on investigating in wheeled navigation and obstacles negotiation. This chapter is organised as follows: some inspection robots are presented in the following section. In Section 2.2, wheeled configuration to make navigation are discussed. An obstacle action in a vehicle is considered in Section 2.3. In Section 2.4, a motion sequence on reconfigurable robots are discussed. MPC general formulation is considered in Section 2.5. In Section 2.6, the PFC approach is described. Finally, Section 2.7, presents the ROS framework for both environments, simulation and real world.

2.1 Robots for Extreme Conditions

Due to extreme conditions of Oil and Gas (O&G) environment for human operators, (CARVALHO *et al.*, 2017) presents an autonomous rail-guided robot to perform routine monitoring and inspection task on offshore platforms. The system moves through a designed rail, carrying several sensors and a robotic manipulator to identify video and audio anomalies that are send as alarms to remote operators.

(GARCIA *et al.*, 2019) describes a procedure to inspect belt conveyor structures with a robotic device, the robot has a manipulator and a set of sensors to make sound, vibration and temperature inspection of the idler rolls.

In the case of conveyors belt used in underground mining, an inspection robot is proposed to support the maintenance staff (SZREK *et al.*, 2020). Moreover, the robot can identify hot spot based on thermal imaging.

Below, known commercial robots for hazardous environments are shown.

2.1.1 ANYmal

The robot ANYmal (Figure 2.1) is a quadruped robot designed to make inspection task on industrial environments, his legs allow the robot to climb stairs and to deal with obstacles. The robot is mounted with visual and thermal sensors and it also uses a laser to make 3D representations. The robot has been tested on Oil & Gas sites, proving its capability to navigate on challenging terrain (ANYBOTICS AG, 2022).

Figure 2.1: ANYmal inspection robot

2.1.2 ROSI

Rosi is a wheel/tracked mobile robot designed to inspect belt conveyor machinery in the mining industry, and it is equipped with a Kinova manipulator and several sensors (Figure 2.2). This mobile platform has a hybrid traction with wheels and flippers, allowing it to move around and overcome obstacles. The vehicle has 4 identical traction modules which differs only in the assembly of adjacent components, lending a significant impact on parts manufacturing and maintenance. Some robot characteristics are; maximum speed on flat ground of $1\,m/s$, maximum angular speed of the flippers $10\,°/s$, robot mass of $60\,Kg$, with dimensions; $length = 0.89\,m$, $height = 0.62\,m$, $width = 0.67\,m$, and flipper $length = 0.45\,m$, (FARIA *et al.*, ROCHA *et al.*, 2020, 2021).

2.2 Navigation for Reconfigurable Robots

To complete an inspection mission, the Rosi robot operates on a wheeled mode while it navigates on a regular terrain. It starts from an initial to the goal position in which the inspection task is performed. In this configuration, the robot used an architecture of autonomous driving made of three fundamental concepts such

Figure 2.2: Rosi robot on an offshore platform

as perception, planning and control. (KHAN, 2022). This work context is based mainly on the control phase as shown in Figure 2.3.

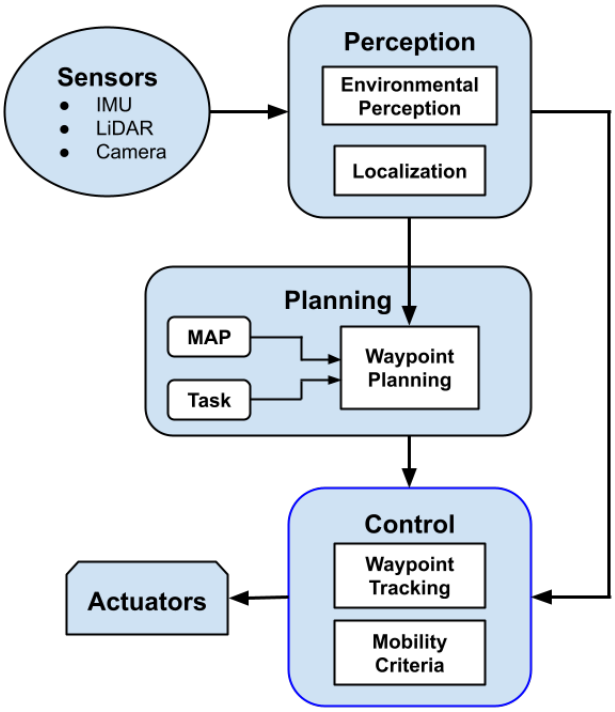


Figure 2.3: Rosi architecture for wheeled configuration

With respect to the perception phase, diverse algorithms can be used. For example, In (SHAN and ENGLOT, 2018) a ROS package (LeGO-LOAM) for simultaneous localization and mapping (SLAM) using a data from a 3D laser sensor is developed. It give us the state and pose estimation for ground vehicles. And, because the registration of point clouds and features sometimes cause large drift, the laser information is not enough, then its typically fuse these data with other sensors like; an inertial measurement unit (IMU) or a GPS. For example, In (SHAN *et al.*, 2020), the LIO-SAM framework is proposed, it uses factor graphs to process the data sen-

sors, the graphs receive measurements of angular velocity and acceleration from the IMU to infer the robot motion.

Regarding the planning phase, In (KIM, 2020) a global planner estimates the path which guides the mobile robot through waypoints to a goal location, the authors use a tracking controller that generates the control inputs for a differential drive robot. Another type of supervisory control is shown in (BAKER *et al.*, 2020). Here, the operator selects a target location on a display interface, then the robot moves autonomously using a waypoints navigation. And, In (BANSAL *et al.*, 2020) presents an algorithm that produces a sequence of intermediate states (waypoints) to guide the robot to the desired target location. These waypoints are used to generate a feasible trajectory that is executed using feedback control. Finally, to deal with the nonholonomic nature of the wheeled configuration. In (MICHALEK and KOZLOWSKI, 2009) a feedback control method for a differential driven vehicle based on polar coordinates is proposed.

2.3 Dealing with Obstacles

Reconfigurable robots are made to deal with complex obstacles. For example, in (LIU and LIU, 2008) analyzes the interaction between a designed tracked mobile robot and the stairs. Moreover, the authors presented a procedure of four steps to climb a stair; First, climbing onto the stairs to interact with the stair lower level. Second, setting back the flippers to put the robot center of gravity forward. Third, going on the nose line. Finally, landing on the upper floor. In (LIN and GOLDENBERG, 2018) presents a tracked vehicle with an enhanced maneuverability to travel over unpredictable surfaces like slopes or stairways. The robot has a pair of rotating flippers to negotiate obstacles. In (WANG *et al.*, 2014), a tracked mobile robot with for rescue missions in coal mines is presented. The robot has front and back arms to adapt to complex terrains. Furthermore, an strategy of vertical obstacle negotiation is proposed.

In (GIANNI *et al.*, 2016) presents an articulated tracked vehicle with flippers, which extend its locomotion capabilities in hazardous environments. Furthermore, a controller which adapts the flippers configuration and generates the track velocities to allow the robot to follow a given path is proposed. Their approach develops the direct and differential kinematic model that correlates the robot body and the flippers motion for a traversal task execution.

In (JUN *et al.*, 2016), a path-planning algorithm for a tracked mobile robot to traverse uneven terrain at low speed is proposed. Moreover, a given environment is defined as the union of traversable and non-traversable regions.

In (COLAS *et al.*, 2013) proposed a system to solve path planning and execution

for ground robots evolving in 3D using a search and rescue robot with flippers. The robot represent their environment based on 3D laser point clouds which provides localization and help it to decide whether a specific pose is feasible.

Mechanisms to deal with vertical obstacles can be found in some vehicles brands. For instance, an active suspension system developed by Mercedes Benz, the ABC concept (active body control), these system seeks to keep constant the car height in relation to the ground, independent of variations in forces acting on the vehicle (GAWAD, 2021). A recent version, the PRE-SCAN system in the F700 research vehicle (RAUH and AMMON, 2011), can register road conditions, react very sensitively to bumps, and compensate for them more effectively using the additional foresight information provided by the two laser sensors in the headlights as 'eyes'. These framework record the road surface ahead of the vehicle to create an accurate ground elevation profile, It determines the driver signals for the individual ABC wheel actuators, then the wheels follow the road profile, while the vehicle body glides over the surface virtually unaffected as shown in Figure 2.4.

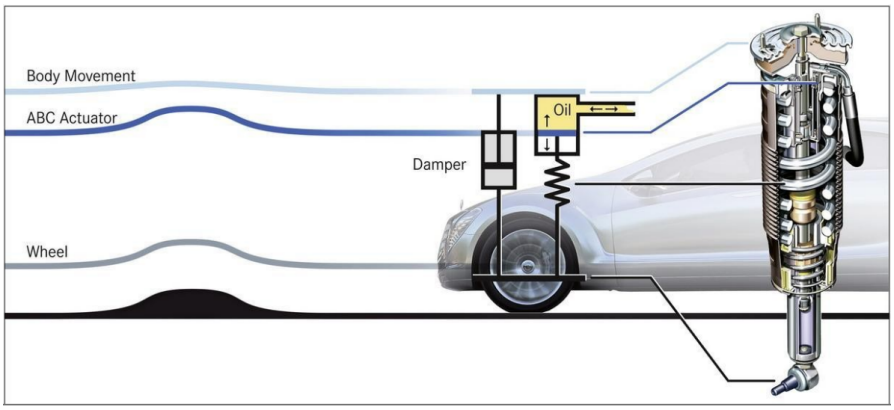


Figure 2.4: Mercedes-Benz F700 Pre-Scan system control

2.4 Motion Sequence

Reconfigurable robots with flippers need to perform a sequence of motion to achieve to deal with steps, stairs, ramps, etc. For example, In (CARVALHO, 2016), presents transition phases for an explosive ordnance disposal (EOD) mobile robot to climb a stair.

These Robot (Figure 2.5) has tracked wheels with flippers that allowed to overcome uneven terrain, their front flippers are command with an actuator and their rear flippers with another, they can rotates 360° with a maximum velocity of $10°/s$, and the robot linear speed has a maximum of 30 cm/s. Moreover, the robot is equipped with an manipulator of three actuated joints that can carry a weight of up to $10kg$.

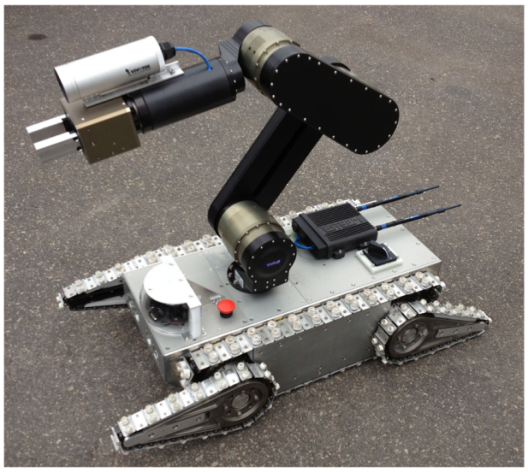


Figure 2.5: EOD Mobile Robot, Diane

In (OHNO *et al.*, 2007), a sequence of control rules for the active flippers is proposed, its based on sensor information, with the objective to overcome unknown steps. These sequence of motions to upward and downward steps is designed to increase their traction, and to reduce the impact of the body with the ground. In (ZIMMERMANN *et al.*, 2014) presents an articulated robot to traverse unknown terrain with obstacles in an optimal way, the authors defined five morphological configurations –different flipper modes, to *train* their algorithm.

2.5 Model Predictive Control

In this section, we present an optimal control theory to be considered.

Model Predictive Control (MPC) reflects the human behaviour whereby we select the control actions which we think will lead to the best predicted outcome. We constantly update our decisions as new observations become available (ROSSITER, 2003).

The basic concept of MPC is the use of a dynamic model to predict the system behavior, optimizing the forecast to produce the best decision. Therefore models are central (RAWLINGS *et al.*, 2017).

The term MPC does not designate a specific control strategy but rather an ample range of control methods which make use of a process model to obtain the control signal by minimizing an objective function (CAMACHO and ALBA, 2013).

The MPC strategy is shown in figure (2.6). Here, the future outputs within a horizon H are predicted at each instant k using the process model $y(k)$. These predicted outputs $\hat{y}(k+i|k)$ for $i = 1 \dots H$ depend of known values up to instant k, and on the future control signals $u(k+i|k)$, $i = 0 \dots H-1$. The set of future control signals is calculated by optimizing a determined quadratic function of the

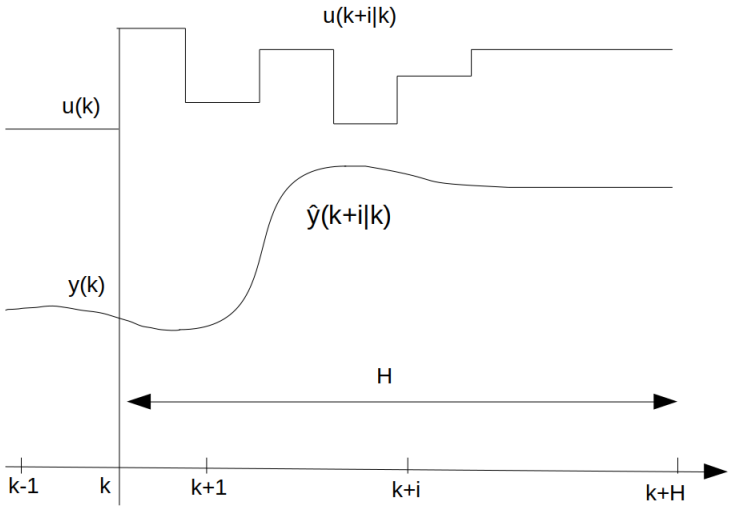


Figure 2.6: MPC Strategy

errors between the predicted output signal and the predicted reference trajectory. In the end, only the control signal $u(k|k)$ is sent to the process whereas the next control signals, and at the next sampling instant $y(k+1)$ the operation is repeated and all the values are brought up to date.

One MPC application related with this work is the Adaptive Cruise Control (ACC) (NAUS *et al.*, 2008). These system is a widespread functionality in modern vehicles, and whose working principle is explained in the Figure 2.7. Here, the host vehicle (on the right side) is equipped with an ACC system, which assures an automatic following of the preceding target vehicle at a set distance, the red lines correspond to radar beams that measure the relative distance and velocity between the two vehicles. This model consider peak acceleration values in combination with the relative distance. Finally, the system is presented as a control of longitudinal comfort related to the handling of a vehicle, focusing on aspects like; pedal response, brake control, etc.

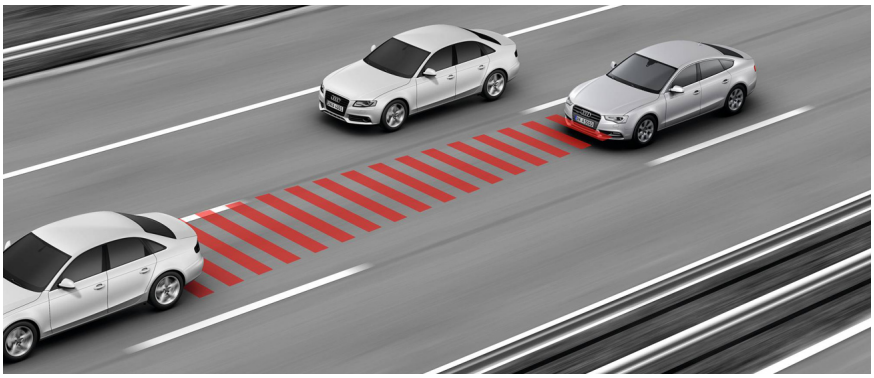


Figure 2.7: ACC Stop & Go system, Audi

Furthermore, due to a discomfort of the ACC system in a traffic jam scenario,

In (TAKAHAMA and AKASAKA, 2018) a MPC with low-order prediction model (computational cost) that can handle with constraints is proposed.

In (FREITAS *et al.*, 2013), the MPC technique is used to anticipate the interactions between the abrupt terrain, through the control of articulated slow actuators for prevent a tipping over.

2.6 Predictive Functional Control

(FREITAS, 2014) presents reconfigurable robots with various active mechanisms that adjust the robot to the terrain, using diverse mobilities criteria such as height, orientation, traction, etc. Furthermore, the authors proposed the Predictive functional Control (PFC) technique to control these mechanisms.

The PFC technique was developed by (RICHALET *et al.*, 1987), and is well-known by its computational load and its flexibility regards to MPC where the resolution of a quadratic equation is involved. A simple representative simulator scheme is shown in Figure 2.8. Here, r denotes a *set point*, u the *control variable*, and y is the *process output* (RICHALET, 1993).

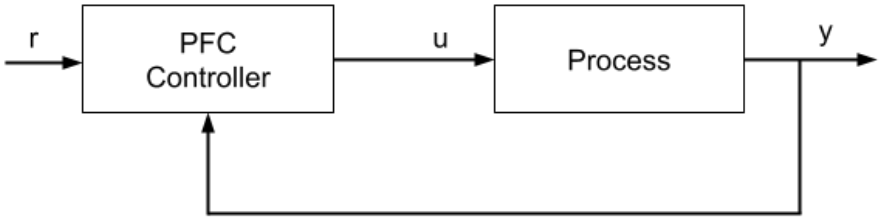


Figure 2.8: PFC scheme

Although PFC structure is well known, their internal variables can grasp different values, depending mainly by the process nature. For instance, on implementation in industrial applications; In (BOUHENCHIR *et al.*, 2006), it was used for the temperature control of a chemical batch reactor, and the evaluation of its robustness in front of the dynamic of the heating/cooling system. The authors also draft a mathematical calculation of the PFC elements. Furthermore, an analogous proceeding for a nonlinear model is proposed in (ZHANG *et al.*, 2011), where Its used to control the liquid level in a coke fractionation tower, describing the combination of a linear model and a nonlinear optimization part to obtain the respective model output.

For robotics applications, In (ZHANG *et al.*, 2005), a scheme and main steps of PFC are presented for the endpoint tracking trajectory control of a two-link robot manipulator. In (SATOH *et al.*, 2019), an intuitive procedure of PFC approach was introduced, for a single-axis positioning system using an estimator-based on internal model; and a comparison with a standard PI controller.

And, to be familiar, a comparison between PFC technique and classic PID controller is shown in (NAGASE *et al.*, 2013), where a system of a tendon-driven balloon actuator for medical care applications is presented, they give us an instructive block diagrams which evaluates both control performances.

Chapter 3

Kinematic Model

In this section, we present the representation for a rigid motion that will be used throughout this thesis.

A rigid body is defined as a collection of a large number of small mass elements which all maintain a fixed spatial relationship with respect to one another (FITZ-PATRIK, 2008).

A rigid motion of an object is one that preserves the distance between points. It is represented by using rigid body transformations that describes the instantaneous position and orientation of an body frame, relative to an inertial coordinate frame (MURRAY *et al.*, 1994).

In this work, we denote a coordinate frame by any chosen origin point O, that belongs to the body, where a set of three orthonormal axes $E = \{x, y, z\}$ is fixed.

For instance, in Figure 3.1. We describe the position of a point P that belongs to the body frame $B = \{O_B, E_B\}$ with respect to an inertial frame $I = \{O_I, E_I\}$.

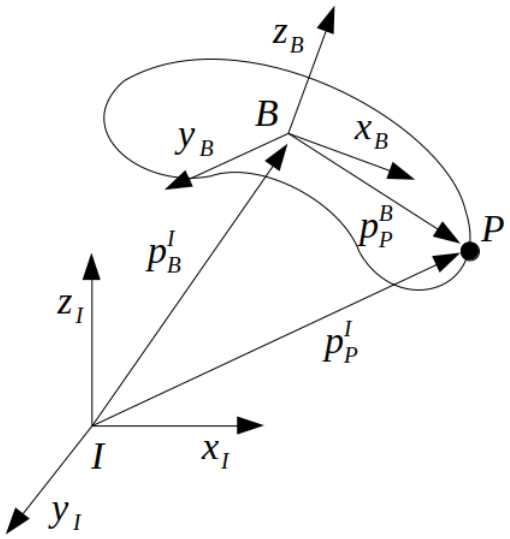


Figure 3.1: Coordinate frames for a rigid body point

3.1 Pose of a Rigid Body

A rigid body is completely described in space by its position and orientation with respect to a reference frame. For example, the pose for a body point frame B with respect to an inertial frame I, depicted in Figure 3.1 consist of the pair:

$$\mathbf{x}_B^I = \left(p_B^I, R_B^I\right)$$

where $p_B^I \in \mathbb{R}^3$ is the position vector of the origin frame B with respect to the inertial frame I, it is given by:

$$p_B^I = \begin{bmatrix} p_{Bx}^I \\ p_{By}^I \\ p_{Bz}^I \end{bmatrix}$$

and, $R_B^I \in SO(3)$ express the orientation matrix of frame B relative to frame I,

$$SO(3) = \left\{R \in \mathbb{R}^{3\times 3} : R^T R = I_3 \text{ and } \det(R) = 1\right\}$$

where $I_3 \in \mathbb{R}^{3\times 3}$ denotes the identity matrix.

The rotation matrix R_B^I is minimally parameterized and defined by the angles *roll* ϕ, *pitch* θ, *yaw* ψ, such that:

$$\varphi_B^I = \begin{bmatrix} \phi_B^I \\ \theta_B^I \\ \psi_B^I \end{bmatrix}$$

Using the parameterized by angles, the matrix R_B^I is compute by elementary rotations on the axes $\{x^I\}$, $\{y^I\}$, $\{z^I\}$.

$$R_B^I = R_{z^I}(\psi) R_{y^I}(\theta) R_{x^I}(\phi)$$

and with the minimum orientation representation φ_B^I, the pose can be defined by:

$$\mathbf{x}_B^I = \left[p_B^I, \varphi_B^I\right]^T$$

Analogous, its possible to combine the pose frames P and B relative to frame I, to represent the vector position p_P^I and the matrix rotation R_P^I, by the equations:

$$p_P^I = p_B^I + R_B^I \, p_P^B \tag{3.1}$$

$$R_P^I = R_B^I \, R_P^B \tag{3.2}$$

3.2 Velocity of Rigid Bodies

The velocity of the rigid body B with respect to an inertial frame I, (Figure 3.1) is defined by the linear v_B^I and the angular velocity ω_B^I.

The linear velocity v_B^I is obtained deriving the position p_B^I on function of time t:

$$v_B^I = \frac{d\, p_B^I}{dt} = \dot{p}_B^I$$

The angular velocity is given by the vector ω_B^I:

$$\omega_B^I = \begin{bmatrix} \omega_{Bx}^I \\ \omega_{By}^I \\ \omega_{Bz}^I \end{bmatrix}$$

And, the relation between rotation matrix and its derivative:

$$\dot{R}_B^I = \omega_B^I \times R_B^I \tag{3.3}$$

where, $\times$ is the vector product.

To represent the linear velocity of a point P attached to the rigid body B with respect to an inertial frame I, we differentiated the position equation (3.1) with respect to time:

$$\dot{p}_P^I = \dot{p}_B^I + \dot{R}_B^I\, p_P^B + R_B^I\, \dot{p}_P^B \tag{3.4}$$

combining the equation (3.3) into equation (3.4) we obtain:

$$\begin{aligned} v_P^I &= \dot{p}_B^I + \omega_B^I \times R_B^I\, p_P^B + R_B^I\, \dot{p}_P^B \\ \omega_P^I &= \omega_B^I + R_B^I\, \omega_P^B \end{aligned}$$

The equations mention above can be represented in a matrix notation, (JAIN and RODRIGUEZ, 1992).

$$\begin{bmatrix} v_P^I \\ \omega_P^I \end{bmatrix} = \begin{bmatrix} I_3 & -R_B^I\, p_P^{B\times} \\ 0 & I_3 \end{bmatrix} \begin{bmatrix} v_B^I \\ \omega_B^I \end{bmatrix} + \begin{bmatrix} R_B^I & 0 \\ 0 & R_B^I \end{bmatrix} \begin{bmatrix} v_P^B \\ \omega_P^B \end{bmatrix} \tag{3.5}$$

where $I_3, 0 \in \mathbb{R}^{3\times 3}$.

3.3 Kinematic Reconfigurability

Reconfigurable robots have the ability to reconfigure its structure to improve stability and ground traction.

These robots usually have articulated mechanisms like; flippers, tracks, legs, etc. which allowing them to reposition their center of mass (CM), and achieve some adaptation to ground conditions.

To obtain a suitable model that represent the interaction between an articulated robot with the terrain. We assume that the total mass is concentrated on the body robot where its CM is fixed regardless the others links configuration, and its correspond with the robot frame $R = \{O_R, R_R\}$.

The other two reference frames to define a system mobility are; the inertial frame $I = \{O_I, E_I\}$, and the terrain frame $\vartheta = \{O_\vartheta, E_\vartheta\}$ (Figure 3.2). Where the plane ϑ is defined by the m flippers contact points p_{fi}, $i = 1, \ldots, m$, whether $m = 2$, the ground is simplified by a straight line, and when $m > 3$ the contact between the robot and the ground is guaranteed.

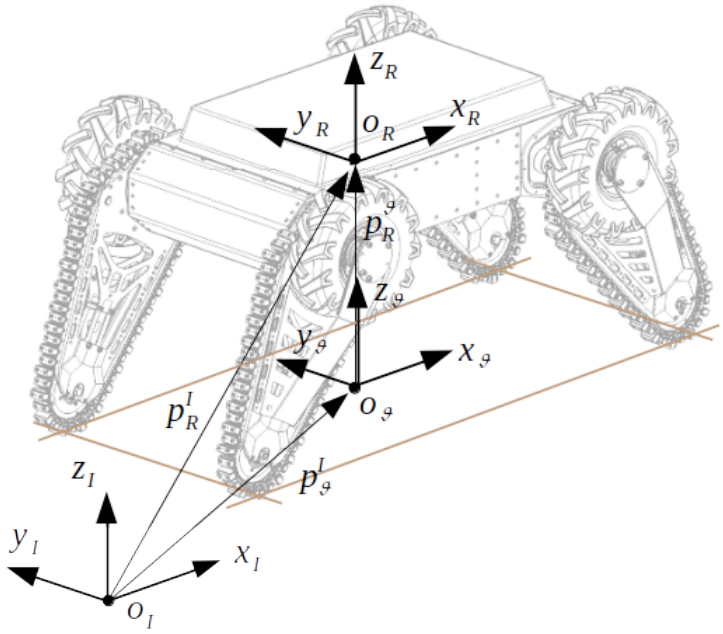


Figure 3.2: Representation of the coordinate transformations for a reconfigurable robot - ROSI

These reconfigurable robot can be described with respect to the inertial by the pose $\mathbf{x}_R^I = \left(p_R^I, R_R^I\right)$, which is computed by combining the terrain pose with respect to an inertial frame $\mathbf{x}_\vartheta^I$, and the robot pose with respect to the terrain frame $\mathbf{x}_R^\vartheta$, as follow:

$$\begin{aligned} p_R^I &= p_\vartheta^I + R_\vartheta^I \, p_R^\vartheta \\ R_R^I &= R_\vartheta^I \, R_R^\vartheta \end{aligned} \tag{3.6}$$

where $p_R^I, p_\vartheta^I, p_R^\vartheta \in \mathbb{R}^3$ are vector positions, and $R_R^I, R_\vartheta^I, R_R^\vartheta \in SO(3)$ are rotations matrix.

The system also can be described with respect to the robot frame, in which the pose $\mathbf{x}_I^R = \left(p_I^R, R_I^R\right)$ is expressed as:

$$\begin{aligned} p_I^R &= p_\vartheta^R + R_\vartheta^R\, p_R^\vartheta = -p_R^I \\ R_I^R &= R_\vartheta^R\; R_I^\vartheta = (R_R^I)^T \end{aligned} \tag{3.7}$$

3.3.1 Flipper differential kinematic

To evaluate the flippers velocity with respect to the robot frame, let us consider one flipper structure sketched in Figure 3.3. Let p_{f1}^R, $p_a^R \in \mathbb{R}^3$ be the positions of the origins of frames $f1 = \{O_{f1}, E_{f1}\}$ and $a = \{O_a, E_a\}$ with respect to frame $R = \{O_R, E_R\}$. Let $p_{f1}^a \in \mathbb{R}^3$ be the position of the origin of frame $f1$ with respect to frame a. Then, we have:

$$p_{f1}^R = p_a^R + R_a^R\, p_{f1}^a \tag{3.8}$$

where, $R_a^R \in SO(3)$ is the rotation matrix of frame a with respect to frame R.

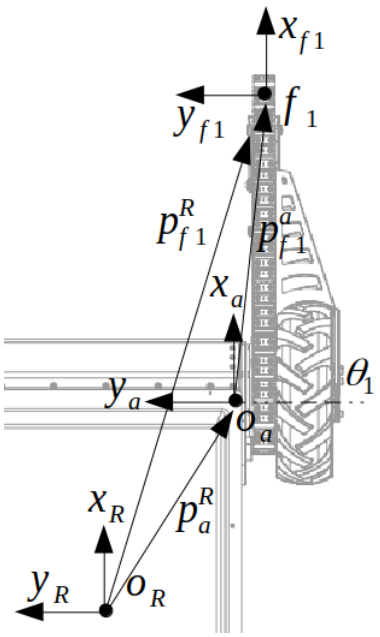


Figure 3.3: Transformations to compute the flipper velocity contribution

By differentiating the equation (3.8) with respect to time, we obtain:

$$\dot{p}_{f1}^R = \dot{p}_a^R + R_a^R\, \dot{p}_{f1}^a + \dot{R}_a^R\, p_{f1}^a. \tag{3.9}$$

where the terms $\dot{p}_a^R$ and $\dot{R}_a^R$ are null, because it describes frames on the same robot link. Thus, $\dot{p}_{f1}^R$ corresponds to the expression of the angular velocity of flipper link

with respect to the robot link, we have:

$$\dot{p}^R_{f1} = R^R_a \, \dot{p}^a_{f1} \tag{3.10}$$

The vector velocity $\dot{p}^a_{f1} \in \mathbb{R}^3$ of frame $f1$ with respect to frame a can be expressed as follow:

$$\dot{p}^a_{f1} = \omega^a_{a,f1} \times p^a_{f1} \tag{3.11}$$

where $\omega^a_{a,f1} \in \mathbb{R}^3$ is the flipper angular velocity of frame $f1$ with respect to a, expressed in frame a. Besides, to express the velocity $\dot{p}^a_{f1}$ in the robot parameters, the equation (3.11) is written as:

$$\dot{p}^a_{f1} = R^a_R \, [\omega^R_{a,f1} \times p^a_{f1}] \tag{3.12}$$

where, $R^a_R \in SO(3)$, and $\omega^R_{a,f1} \in \mathbb{R}^3$ is the flipper angular velocity of frame $f1$ with respect to a, expressed in frame R. Next, substituting the equation (3.12) into (3.10), we have:

$$\dot{p}^R_{f1} = \left[\omega^R_{a,f1} \times p^a_{f1}\right] \tag{3.13}$$

here, its known that $\omega^R_{a,f1} = \dot{\theta}_1 \, y_1$, and $\dot{\theta}_1 \in \mathbb{R}$ is the input flipper angular velocity.

3.3.2 Articulated flippers

In this section, we describe the differential kinematic for all the robot flippers, taking into account the one developed above.

Equation (3.8) shows a flipper position, a desirable point of contact with the terrain, with respect to the robot frame R. Then, the position of all the *contacts points* considering the respective actuated joints θ is written as:

$$p^R_{fi} = f_{pi}(\theta_i) \quad , \quad i = 1, \ldots 4$$

from equation (3.13), and considering the velocity of each actuated joint θ_i, the differential kinematic for the flippers contact point, can be write as:

$$\dot{p}^R_{fi} = J_{p_{fi}}(\theta_i)\dot{\theta}_i$$

Stacking all the flipper contact points $p_f = [p^T_{f1}, ..., p^T_{f4}]^T$ with their respective actuated joints $\theta = [\theta_1, ..., \theta_4]^T$, we have:

$$\dot{p}^R_f = J_{p_f}(\theta)\dot{\theta}$$

where $J_{pf} \in \mathbb{R}^{12\times4}$, and $p^R_f = f_p(\theta)$ defines the relation between the terrain and the robot.

The plane ϑ formed by the contacts points (Figure 3.2) is defined by the normal vector n_ϑ^R and any of p_{fi}^R, such that:

$$(\check{n}_\vartheta^R)^T \, p_{fi}^R - h_R = 0 \tag{3.14}$$

where $h_R \in \mathbb{R}$ is the distance from the terrain to the robot origin frame, and $\check{n} = \frac{n}{||n||}$ is a normalized vector.
The vector n_ϑ^R is computed respect to the robot frame using the three contact points p_{f1}^R, p_{f2}^R and p_{f3}^R:

$$n_\vartheta^R = (p_{f2}^R - p_{f3}^R) \times (p_{f1}^R - p_{f2}^R) \in \mathbb{R}^3 \tag{3.15}$$

These relation is valid where the points p_{f1}^R, p_{f2}^R and p_{f3}^R are non-collinear and non-coincident.

The matrix rotation $R_\vartheta^R \in SO(3)$ of the plane ϑ with respect to the frame R, can be computed with the normal vector $-\check{n}_\vartheta^R = z_\vartheta^R$, through a rotation between vectors.

$$R_\vartheta^R = I_3 + (\widehat{z \times z_\vartheta^R}) + \frac{1}{1 + z^T \, z_\vartheta^R} (\widehat{z \times z_\vartheta^R})^2 \tag{3.16}$$

here, $z = [0, 0, 1]^T$, and $(\widehat{\ })$ represents to the anti-symmetric matrix of a vector.

The position p_ϑ^R of the plane ϑ with respect to the robot frame is computed from equation (3.14).

$$p_\vartheta^R = h_R \, \check{n}_\vartheta^R \tag{3.17}$$

The velocity of actuated joints $\dot{\theta}$ determines the plane ϑ fluctuation with respect to the robot frame, its obtained differentiating the equation (3.17) with respect to time.

$$\dot{p}_\vartheta^R = \dot{h}_R \, \check{n}_\vartheta^R + h_R \, \dot{\check{n}}_\vartheta^R$$

And, the angular velocity ω_v^R is given by:

$$\omega_\vartheta^R = (-\hat{z}_\vartheta^R)^{-1} \, \dot{z}_\vartheta^R = -(\widehat{\check{n}_\vartheta^R})^{-1} \, \dot{\check{n}}_\vartheta^R$$

3.4 Mobility defined by height

The reconfigurable robot ROSI has actuated flippers attached at each corner with a length of d. The body robot is represented by a rectangular polygon of length L and width W, as shown in Figure A.1.

Then, the flippers contact points p_{fi}^R with respect to the robot frame are obtained in function of the actuated joints θ_i using the equation (3.8).

$$p_{f1}^R = \begin{bmatrix} \frac{L}{2} + d\,cos(\theta_1) \\ \frac{W}{2} \\ d\,\sin(\theta_1) \end{bmatrix} \quad ; \quad p_{f2}^R = \begin{bmatrix} \frac{L}{2} + d\,\cos(\theta_2) \\ -\frac{W}{2} \\ d\,\sin(\theta_2) \end{bmatrix}$$

where p_{f1}^R, p_{f4}^R, represents positions of front flippers with the actuated joint θ_f, and p_{f2}^R, p_{f3}^R are the positions of rear flippers with the actuated joint θ_r.

$$p_{f3}^R = \begin{bmatrix} -\frac{L}{2} - d\,\cos(\theta_3) \\ \frac{W}{2} \\ d\,\sin(\theta_3) \end{bmatrix} \quad ; \quad p_{f4}^R = \begin{bmatrix} -\frac{L}{2} - d\,\cos(\theta_4) \\ -\frac{W}{2} \\ d\,\sin(\theta_4) \end{bmatrix}$$

The normal vector n_ϑ^R is obtained with the equation (3.15).

$$n_\vartheta^R = W\,d \begin{bmatrix} -\sin(\theta_f) + \sin(\theta_r) \\ 0 \\ \frac{L}{d} + \cos(\theta_f) + \cos(\theta_r) \end{bmatrix} \tag{3.18}$$

From equation (3.14), we defined the distance from the plane ϑ to the robot frame R, as a height function f_h.

$$h_R = (\check{n}_\vartheta^R)^T\, p_{fi}^R = f_h \tag{3.19}$$

Computing the norm $\check{n}$ of equation (3.18) and considering one contact point of the front joint θ_f, we obtain:

$$h_R = \frac{Wd}{||n_\vartheta^R||} \left[\begin{array}{c} \frac{L}{2}\left(\sin(\theta_f) + \sin(\theta_r)\right) + d\,\sin(\theta_f + \theta_r) \end{array} \right]$$

By differentiating the equation (3.19) with respect to time:

$$\dot{h}_R = (\dot{\check{n}}_\vartheta^R)^T p_{fi}^R + (\check{n}_\vartheta^R)^T \dot{p}_{fi}^R \tag{3.20}$$

where

$$\dot{\check{n}}_\vartheta^R = \frac{W\,d}{||n_\vartheta^R||} \begin{bmatrix} -\cos(\theta_f) & \cos(\theta_r) \\ 0 & 0 \\ -\sin(\theta_f) & -\sin(\theta_r) \end{bmatrix} \tag{3.21}$$

and

$$\dot{p}_{f1}^R = \begin{bmatrix} -d\,\sin(\theta_1) \\ 0 \\ d\,\cos(\theta_1) \end{bmatrix} \tag{3.22}$$

Then, substituting equations (3.21), (3.22) into equation (3.20), we obtain:

$$\dot{h}_R = \frac{Wd}{||n_\vartheta^R||} \left[\begin{array}{cc} \frac{L}{2}\,\cos(\theta_f) + d\,\cos(\theta_f + \theta_r) & \frac{L}{2}\,\cos(\theta_r) + d\,\cos(\theta_f + \theta_r) \end{array} \right] \begin{bmatrix} \dot{\theta}_f \\ \dot{\theta}_r \end{bmatrix}$$

3.5 Mobility defined by orientation

The robot orientation with respect to an inertial frame R_R^I its composed by two rotations; First, the rotation R_ϑ^I of the terrain with respect to the inertial frame. Second, the rotation R_R^ϑ of the robot with respect to plane ϑ:

$$R_R^I = R_\vartheta^I \, R_R^\vartheta = R_z(\psi_\vartheta) \, R_y(\theta_\vartheta) \, R_x(\phi_\vartheta) \, R_R^\vartheta \tag{3.23}$$

The differential kinematic for the angular velocity w_R^I is given by:

$$\omega_R^I = \omega_\vartheta^I + R_\vartheta^I \, \omega_R^\vartheta \tag{3.24}$$

The angular velocity ω_R^ϑ for the body robot with respect to the plane ϑ is given by:

$$\omega_R^\vartheta = \widehat{(\check{n}_\vartheta^R)}^{-1} \, \dot{\check{n}}_\vartheta^R$$

The rotation matrix R_R^ϑ is defined in function of the actuated joints $\theta = [\theta_f, \theta_r]^T$ so:

$$R_R^\vartheta = I + \widehat{(z_R^\vartheta \times z)} + \frac{1}{1 + (z_R^\vartheta)^T \, z} \widehat{(z_R^\vartheta \times z)}^2 \tag{3.25}$$

where $z_R^\vartheta = \check{n}_\vartheta^R$. The angular velocity ω_R^ϑ is obtained with the relation:

$$\dot{z}_R^\vartheta = \omega_R^\vartheta \times z_R^\vartheta$$

To compute $\dot{z}_R^\vartheta$ depending on the derivatives of actuated joints $\dot{\theta} = [\dot{\theta}_f, \dot{\theta}_r]$ according to the expression.

$$\begin{aligned}
\dot{z}_R^\vartheta &= \dot{\check{n}}_\vartheta^R \\
&= \frac{d(\frac{n_\vartheta^R}{||n_\vartheta^R||})}{dt} \\
&= \frac{1}{||n_\vartheta^R||^3} (||n_\vartheta^R||^2 \, I - n_\vartheta^R (n_\vartheta^R)^T) \, J_{n_\vartheta^R} \, \dot{\theta} \\
&= \frac{-1}{||n_\vartheta^R||^3} (\widehat{n}_\vartheta^R)^2 \, J_{n_\vartheta^R} \, \dot{\theta} \\
&= \frac{-1}{||n_\vartheta^R||} (\widehat{z}_R^\vartheta)^2 \, J_{n_\vartheta^R} \, \dot{\theta}
\end{aligned} \tag{3.26}$$

where $\dot{z}_R^\vartheta = \omega_R^\vartheta \times z_R^\vartheta = -z_R^\vartheta \times \omega_R^\vartheta = -\widehat{z}_R^\vartheta \, \omega_R^\vartheta$, then:

$$\omega_R^\vartheta = \frac{1}{||n_\vartheta^R||} \widehat{z}_R^\vartheta \, J_{n_\vartheta^R} \, \dot{\theta}$$

where the Jacobian $J_{n_\vartheta^R} = \dfrac{\partial n_\vartheta^R}{\partial \theta}$ are in function of $\dot{\theta} = [\dot{\theta}_f, \dot{\theta}_r]$.

3.6 Nonholonomic constraints

When ROSI robot moves in his wheeled configuration, it is subject to kinematic constraints that reduces their mobility, these can be classified under various criteria. For instance, a mechanical system with configuration $q \in \mathcal{C}$ where q represent a vector of *generalized coordinates* and $\mathcal{C}$ is a *configuration space* that coincides with $\mathbb{R}^n$, (SICILIANO *et al.*, 2010). An holonomic constraints can be put in the form .

$$h_i(q) = 0 \quad , \quad \forall i = 1, 2, \ldots k < n \tag{3.27}$$

where the function $h_i(q)$: $\mathcal{C} \to \mathbb{R}$, and the effect of these constraint is to reduce the space of accessible configurations to a subset of $\mathcal{C}$ with dimension $n - k$. Then, the system is called *holonomic*.

Kinematic constraints that involve generalized coordinates and velocities are generally expressed in *Pfaffian form* describe as:

$$a_i^T(q)\,\dot{q} = 0 \quad , \quad \forall i = 1, 2, \ldots k < n \tag{3.28}$$

Vectors $a_i : \mathcal{C} \to \mathbb{R}^n$ are assumed to be linearly independent.

If a system with a kinematic constraints of equation (3.28) is not integrable to the equation (3.27) then the system is called *nonholonomic*, reducing the mobility of the mechanical system.

The constrains of equation (3.28) involve that velocities at each configuration q belong to the $(n - k)$-dimensional null space of matrix $a_i^T(q)$, which is denoted with $g_1(q), ..., g_{n-k}(q)$, then the admissible trajectories for the mechanical system is characterized as the solutions of the nonlinear dynamic system:

$$\dot{q} = G(q)\,u \quad , \quad m = n - k \tag{3.29}$$

where $q \in \mathbb{R}^n$ is the state vector and $u = [\ u_1 \quad ... \quad u_m\ ]^T \in \mathbb{R}^m$ is the input vector.

3.6.1 Differential drive robot

The kinematic model for an unicycle configuration can be applied to the differential drive robot system, which we described with $q = [\ x \quad y \quad \phi\ ]^T$, where (x, y) are the geometric center position and ϕ is the orientation with respect to the inertial x_I axis, as shown in Figure 3.4. The nonholonomic constraint for these system can be expressed as:

$$\dot{x}\sin(\phi) - \dot{y}\cos(\phi) = [\sin(\phi) - \cos(\phi) \quad 0]\,\dot{q} = 0 \tag{3.30}$$

Considering the matrix

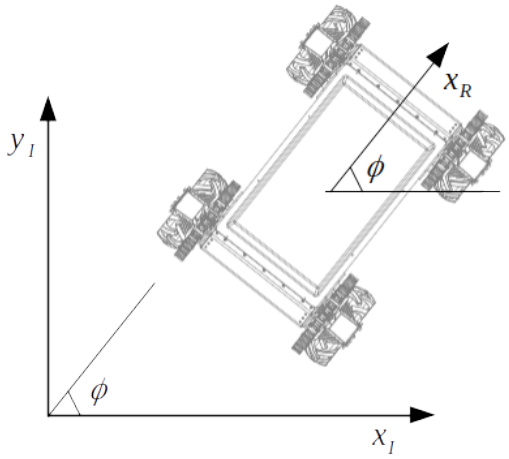


Figure 3.4: ROSI on its wheeled configuration

$$G(q) = [\ g_1(q) \quad g_1(q)\ ] = \begin{bmatrix} \cos(\phi) & 0 \\ \sin(\phi) & 0 \\ 0 & 1 \end{bmatrix} \tag{3.31}$$

where columns $g_1(q)$ and $g_2(q)$ are bases for a null-space for the matrix $a_i^T(q)$. Next, all the admissible generalized velocities at q are obtained as a linear combination of $g_1(q)$ and $g_2(q)$. Thus, we have:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos(\phi) \\ \sin(\phi) \\ 0 \end{bmatrix} \nu + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{3.32}$$

Where ν is the input *driving velocity*, whereas ω is the input *steering velocity* which describes the angular speed around the vertical axis..

Then, it is possible to represent in terms of the angular speeds ω_R and ω_L of the right and left wheel, respectively:

$$\begin{aligned} \nu &= \frac{r_e}{2}(\omega_r + \omega_l) \\ \omega &= \frac{r_e}{d_R}(\omega_r - \omega_l) \end{aligned} \tag{3.33}$$

where r_e is the radius of the wheels and d_R is the distance between their centers.

3.6.2 Posture regulation

To design the feedback controller that is able to regulate the cartesian position and vehicle orientation, it is convenient to formulate the problem in polar coordinates we present a stabilizing feedback control of differential drive robots considering the situation shown in Figure 3.4, where the robot is on an arbitrary position and

orientation, and a predefined goal (position and orientation).

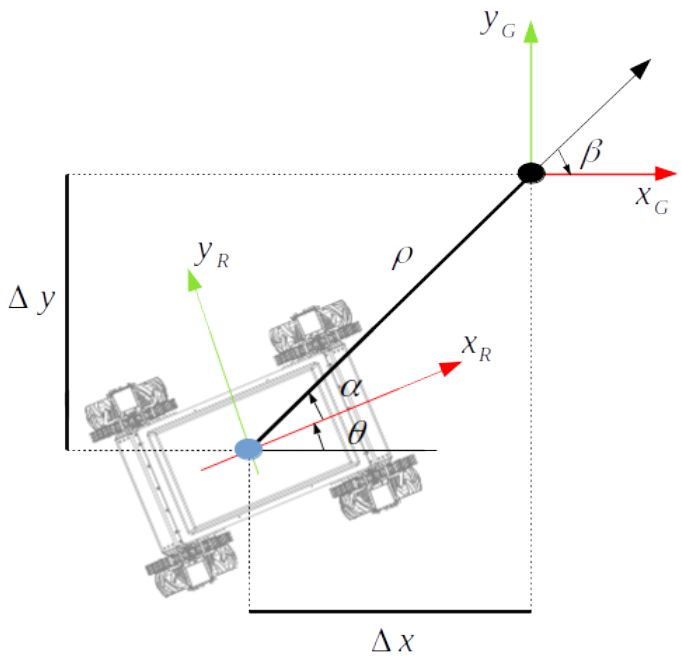


Figure 3.5: ROSI on wheeled configuration

we assume that the goal is at the origin $(0, 0)$ of the reference frame, then position and orientation is represented by the vector $[x, y, \theta]^T$. Then

$$\begin{aligned} \Delta x &= 0 - x \\ \Delta y &= 0 - y \end{aligned} \tag{3.34}$$

Let ρ be the distance between the robot frame R and the goal position o_G, θ denotes the angle between the x_R axis of the robot frame and the x_G axis associated with the goal position. Let α denotes the angle between the x_r axis and the line which connects the robot center of with the goal position. Finally, β is defined as the angle between the robot final orientation and the direction from its current position to the goal position.

And considering a polar coordinate transformation, we have:

$$\begin{aligned} \rho &= \sqrt{\Delta x^2 + \Delta y^2} \\ \alpha &= -\theta + \arctan(\Delta y, \Delta x) \\ \beta &= -\theta - \alpha \end{aligned} \tag{3.35}$$

In these coordinates, the kinematic model for the system is expressed as:

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -\cos(\alpha) & 0 \\ \dfrac{\sin(\alpha)}{\rho} & -1 \\ \dfrac{-\sin(\alpha)}{\rho} & 0 \end{bmatrix} \begin{bmatrix} \nu \\ \omega \end{bmatrix} \tag{3.36}$$

The control signals are designed to drive the robot from its actual configuration to the goal position, considering the control law

$$\begin{aligned} \nu &= k_\rho \, \rho \\ \omega &= k_\alpha \, \alpha + k_\beta \, \beta \end{aligned} \tag{3.37}$$

the description obtained for the closed-loop system is, (SIEGWART *et al.*, 2011):

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -k_\rho \, \rho \, cos(\alpha) \\ k_\rho \, sin(\alpha) - k_\alpha \, \alpha - k_\beta \, \beta \\ -k_\rho \, sin(\alpha) \end{bmatrix} \tag{3.38}$$

It can be shown that the closed-loop control system is locally stable if

$$\begin{aligned} k_\rho \; &> 0 \\ k_\beta \; &< 0 \\ k_\alpha \; - k_\rho &> 0 \end{aligned} \tag{3.39}$$

And the angles α and β are always to be expressed in the range $(-\pi, \pi)$

Chapter 4

Perception and Obstacle negotiation

This chapter expands the use of laser sensors to create a representative environment in which the mobile robot perform the navigation task. This chapter also discusses the obstacle procedure for the reconfigurable robot, and the proposed control technique to be implemented.

4.1 Laser Range Finder

The working principle of a Laser Sensor consists in the emission of light beams, and in the measurement of time in which the beams strikes in the obstacle, then it is possible to obtain the distance covered by the beam, consequently, the distance from sensor to the obstacle.

Laser range sensors are capable of provide structure data; the distance of the measured object and their corresponding direction, which are used to detect the presence of objects nearby, without a physical contact. For instance, In (BARSHAN and KUC, 1992) a sonar system is used to detect obstacles in a two-dimensional (2-D) environment, where the localization is most accurate if the obstacle is located along the line-of-sight.

The operation of a commercial sensor Hokuyo (Hok, 2009), is illustrated in Figure 4.1. This laser makes a scan in a vision plane $\Delta(\psi)$, providing measurements in an array data with r distances, the number of the array elements is called n_r, and its defined with respect to the laser resolution s_l.

$$\begin{aligned} n_r &= \frac{\Delta\left(\psi\right)}{s_l} \\ \Delta\left(\psi\right) &= \psi_{\max} - \psi_{\min} \end{aligned} \tag{4.1}$$

The laser measurements are in polar coordinates, where the angle $(\psi_{\min} + js_l)$ is

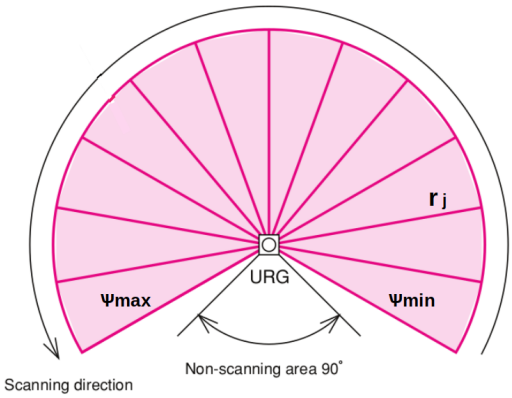


Figure 4.1: Laser range finder operation

associated to each distance r_j according with their position (j) in the array, then the points measured are represented on a cartesian space with respect to the laser:

$$p_{lj} = \begin{bmatrix} r_j \cos(\psi_{min} + j s_l) \\ r_j \sin(\psi_{min} + j s_l) \\ 0 \end{bmatrix} \quad , \quad \text{with} \quad j = 1, 2, \ldots, n_r \tag{4.2}$$

On a set of measures in two dimensions where the laser light beams are emitted from different angles in the same scanning plane, we positioned the laser sensor at the top and in front of the robot to perform vertical scans with the aim of identify the profile of a terrain or an obstacle.

To make the line extraction more robust to the presence of sensor noise, the Random Sample Consensus (Ransac) (FISCHLER and BOLLES, 1981) method is considered, it is widely used to adjust experimental because of it does not require that all the sample points be evaluated. The application of these method extracting the vertical points of a sloping terrain is shown in Figure 4.2.

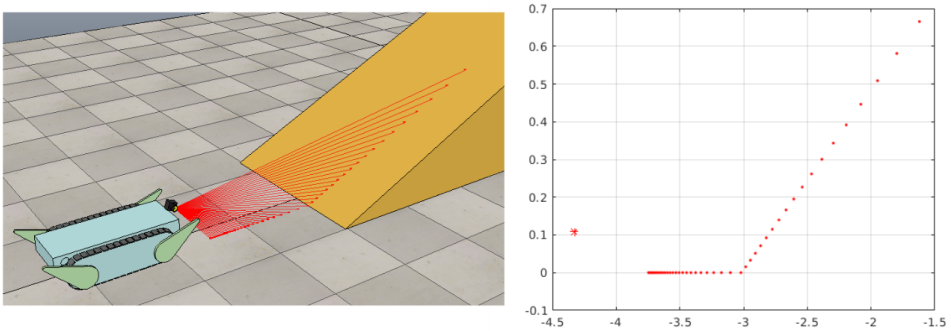


Figure 4.2: Representation of the data laser (Hokuyo)

4.2 3D LiDAR

The collection of highly precise point cloud data is provided by laser scanning systems, and its proven as a solution for mapping applications on moving platforms (PUENTE *et al.*, 2013).

As all the laser scanners, which actively measure the distance between a known reference sensor point and a target that has been illuminated by the laser. LiDAR sensor transmits an electromagnetic pulse of energy to measure a distance to creates 3D point-clouds, describing the environment (KIDD, 2017).

LiDAR sensor reports distances relative to itself in spherical coordinates; $(R, \omega, \alpha) \in \mathbb{R}$, the radius, elevation and azimuth respectively (the angles are shown in Figure 4.3).

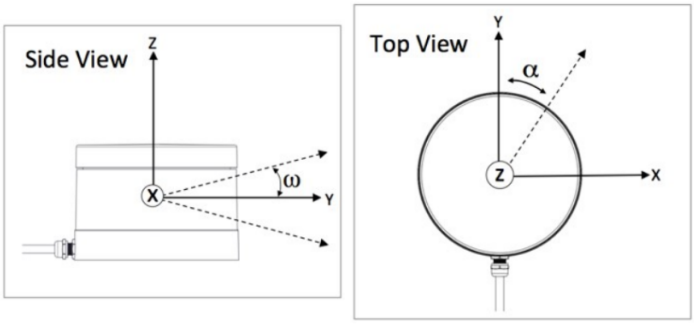


Figure 4.3: elevation and azimuth angles for the velodyne

To convert the spherical data from the sensor to cartesian coordinates (X, Y, Z), A computation depicted in Figure 4.4, and equation (4.3) is necessary.

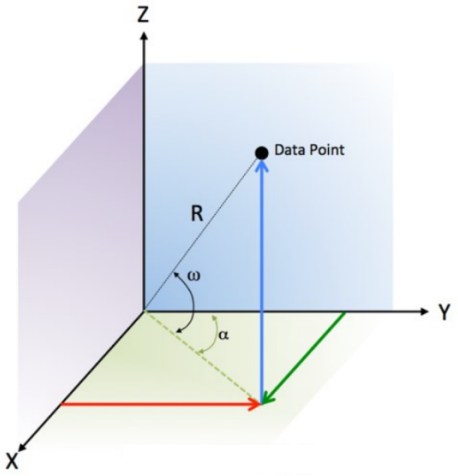


Figure 4.4: velodyne data pointcloud

$$
\begin{aligned}
X &= R\cos(\omega)\sin(\alpha) \\
Y &= R\cos(\omega)\cos(\alpha) \\
Z &= R\sin(\omega)
\end{aligned}
\tag{4.3}
$$

In this work, we use the LiDAR - Velodyne which determines a distance by targeting an object with the laser and measuring the time for the reflected light to return to the receiver. This sensor (Figure 4.5) uses an array of 16 infra-red (IR) lasers paired with IR detectors to measure distances to objects, the array of laser/detector pair spins rapidly within its fixed housing to scan the surrounding environment, firing each laser 18k times per second, providing a set of 3D point cloud data in real time (Vel, 2018).

Figure 4.5: Velodyne VLP-16

4.3 LiDAR Odometry

Having the LiDAR information, diverse SLAM algorithms can be used to compute the pose on ground vehicles. For example, with the LeGO-LOAM framework (SHAN and ENGLOT, 2018), we can make map-building and state estimation. It makes use of the iterative closest point (ICP) technique, which minimize errors from two clouds of points to reconstruct 3D surfaces. The system consists of five modules. First, *Segmentation*, which takes one scan point cloud to project it onto a range image for the *Feature Extraction* module, then the *Lidar odometry* which use features to find transformations relating consecutive scans; these features are processed in *Lidar Mapping*, which registers them to a global point cloud map. Finally, the *transform integration* module combine the pose estimation from lidar odometry and lidar mapping. The application of this framework using our Rosi robot is shown in Figure 4.6.

Nevertheless, the state estimation and mapping is not enough using the LiDAR information, because the registration of point clouds and features, sometimes will

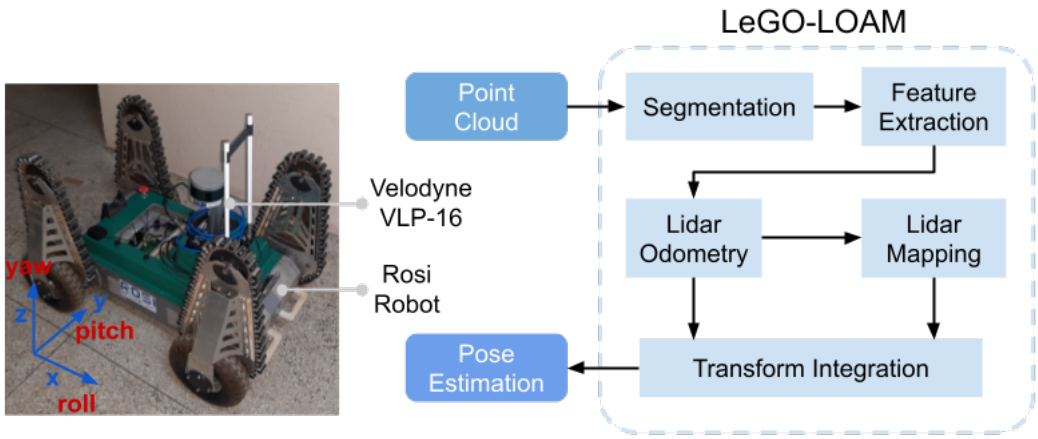


Figure 4.6: LeGO-LOAM on Rosi robot

cause large drift. Thus, LiDAR is typically fused with other sensors like; an inertial measurement unit (IMU) or a GPS. For instance, In (KEARNS, 2020) and (JÚNIOR *et al.*, 2022) the robot wheel odometry and IMU data is fused into the LeGO-LOAM algorithm using an Extended Kalman Filter to improve localization and mapping.

4.3.1 Advanced Navigation - MOTUS

Motus is a miniature Inertial Measurement Unit (IMU) (Adv, 2020). It features the highest accuracy accelerometers and gyroscopes for measuring acceleration and rotation of the body that it is applied to. The accelerometers are mounted in angles perpendicular to each other, so that acceleration can be measured along the X, Y, Z-axis (Figure 4.7). The gyroscopes are mounted in a way that enables measurement of rotation around the Euler angles; X-axis (*roll*), Y-axis (*pitch*), and the Z-axis (*yaw*).

Figure 4.7: IMU Motus

4.3.2 LIO-SAM

The lidar inertial odometry via smoothing and mapping (LIO-SAM) framework (SHAN *et al.*, 2020), fuses data from multiple sensors, becoming more suitable to

capture fine details of an environment in 3D space.

This algorithm use factor graphs to process the data sensors, some of them are; the preintegration factor, which receives measurements of angular velocity and acceleration from the IMU to infer the robot motion. The LiDAR odometry factor, which process features from velodyne scan each time when they arrives. Also exist the GPS factor and Loop Closure Factor, which will not cover in this work. these process is represented in in Figure 4.8.

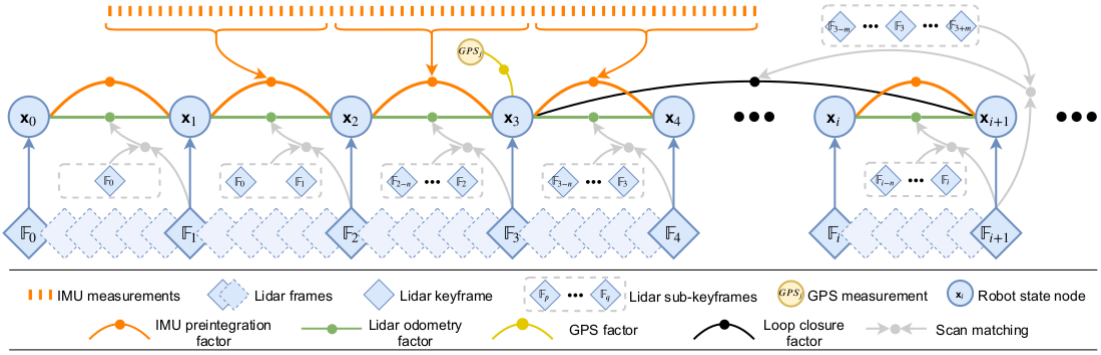


Figure 4.8: Lio-sam graph

4.4 Obstacles negotiation

On the inspection task of the ROSI robot in oil and mining environments. recurrent obstacles were found; steps, barriers and rails, as shown in Figure 4.9. Thus, a prevalent sequence to deal with them can be proposed. Taking into account that to avoid to crash the robot bottom base, a maximum height condition for the obstacle is added.

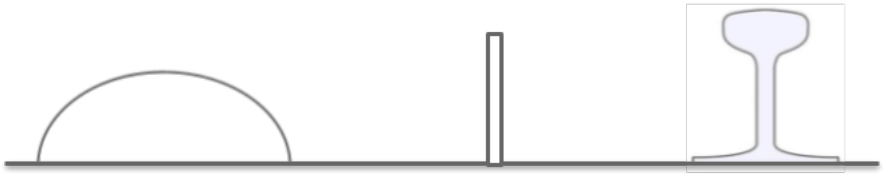


Figure 4.9: Recurrent vertical obstacles found in oil and mining industries

The sequence motion for the flippers was obtained from empirical experimentation and simulation, it is based on states that depend on the variables; a controlled reference height h_R, a flipper angle θ_f, and the robot body angle θ_R *(pitch)* measured by an IMU unit.

The states are is shown in Table 4.1, where a robot moving at a constant linear velocity is considered for each of them. Therefore, on an obstacle dealing process,

it begins with the state A, where θ_R is between a negative θ_{nt} and a positive θ_{pt} threshold, with the condition flipper angle $\theta_f < \pi/2$, then the robot has to reach the reference height $r_{h\,1}$, which depends directly from the obstacle height. On the B state, the body angle θ_R has to exceed the negative threshold angle θ_{nt}, then the robot has to reach a second reference height $r_{h\,2}$. On the C state , the robot already has the flippers downward, verifying that θ_R is between the threshold angles θ_{nt} and θ_{pt}, with the flipper angle condition $\theta_f > \pi/2$, then the robot has to maintain the reference height $r_{h\,2}$. On the state D, the body angle θ_R has to exceed the positive threshold angle θ_{pt}. Then, it has to reach the reference height $r_{h\,1}$. Finally, when the robot has already passed the obstacle, the robot returns to the state A, lifting up their flippers to reach the reference height $r_{h\,1}$.

Table 4.1: flipper control rules for each state to climb an obstacle

State	θ_R		θ_f	h_R
A	$> \theta_{nt}$	$< \theta_{pt}$	$< \pi/2$	$r_{h\,1}$
B	$> \theta_n$	$< \theta_p$	$> \pi/2$	$r_{h\,2}$
B	$< \theta_{nt}$			$r_{h\,2}$
C	$> \theta_{nt}$	$< \theta_{pt}$	$> \pi/2$	$r_{h\,2}$
D		$> \theta_{pt}$		$r_{h\,1}$

4.4.1 System formulation

Described the states for the ROSI robot to deal with an obstacle, we propose the application of a mobility criteria while it moves at a constant linear velocity.

And, for the case in which a coordinated flippers movement impact on the height function f_h of equation (3.19), we present the model for one actuated joint (1-DoF), where the resulting signal control goes to the rest of flippers.

And, considering a robotic system with an articulated joint θ, and a kinematic control model $u = \dot{\theta}$, the system model in a discrete form is presented as follow:

$$\begin{aligned} \theta(k+1) &= \theta(k) + \Delta t\, u(k) \\ y(k) &= f_h(\theta(k)) \end{aligned} \tag{4.4}$$

where $\theta \in \mathbb{R}$ is the variable state, $\Delta t \in \mathbb{R}$ is the sampling time, $u \in \mathbb{R}$ is the input signal, and $y \in \mathbb{R}$ is the model output for the height mobility criteria.

The constant sampling time performs the relation $T = t(k+1) - t(k) > 0$, and the time with respect to these sampling time is: $t = t(k) = kT$.

The constrains considered for this model are represented as:

$$\begin{aligned} \theta(k) &\in \mathcal{X} \\ u(k) &\in \mathcal{U} \end{aligned} \tag{4.5}$$

where, $\mathcal{X} \in [0, \pi]$ is a joint workspace, and $\mathcal{U}$ is a physical restriction.

4.5 PFC Design

PFC is a kind of MPC controller widely described in (RICHALET and O'DONOVAN, 2009), where the main characteristic is the use of some coincident points on the prediction horizon. This technique is flexible, easy to implement, tune, and can be applied to fast process where the objective function compute the predicted errors that corresponds to this selected coincident points.

The main variables involved which define the PFC structure are shown in Figure 4.10.

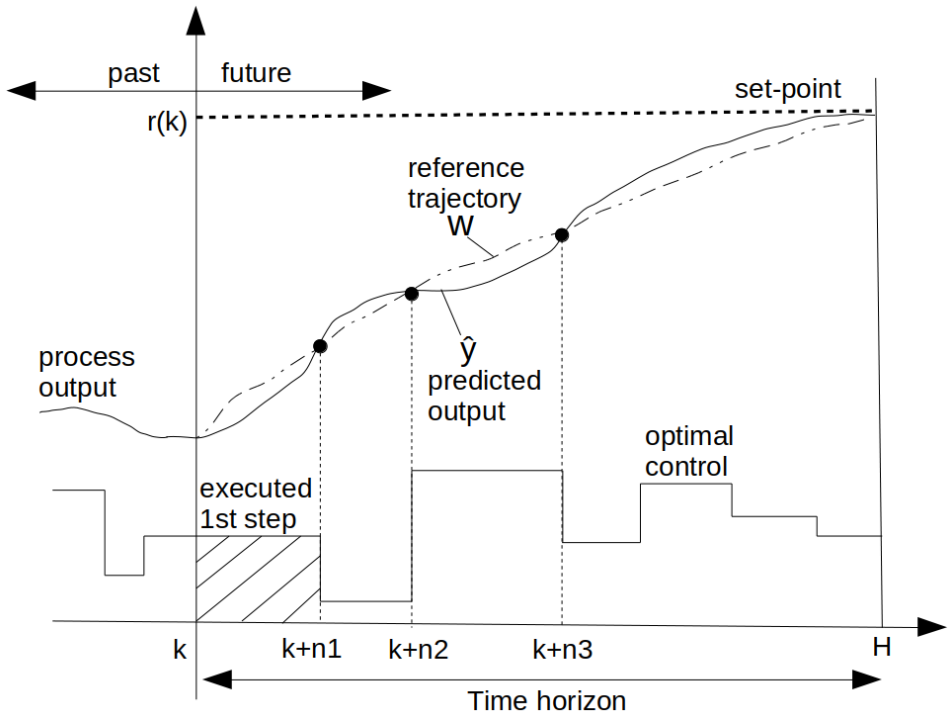


Figure 4.10: Scheme of predictive functional control

Where, $r(k) \in \mathbb{R}$ is the set-point at time k, $\hat{y} \in \mathbb{R}$ is the predictive output which depends of the process model, w is the reference trajectory that has some coincidence points in which the process output is expected to coincide.

Reference trajectory

The reference trajectory w, describes the desired future output towards the set-point $r(k) \in \mathbb{R}$ at time k. It may be interpreted as the temporal path we wish to follow in order to reach the desired set-point value, its given by:

$$w(k+i) = r(k+i) - \alpha^i(r(k) - y_p(k)), \quad i = 1, 2... \tag{4.6}$$

where, $\alpha \in \mathbb{R}$ is chosen between $0 < \alpha < 1$, and $y_p(k)$ is the process output at time k. Therefore, at each sampling time, the reference trajectory is re-initialized using the measured process output.

Objective function

In the PFC approach, the control performance is evaluated through a cost function, (CAMACHO and ALBA, 2013). It take into account the predicted output $\hat{y}(k+n_i)$ and the reference trajectory $w(k+n_i)$ with respect to coincident points. The cost function to be minimized is:

$$V(k) = \sum_{n_i=n_1}^{n_h} \left[\hat{y}\,(k+n_i) - w\,(k+n_i)\right]^2 \; + \; \lambda\left[\Delta\, u\,(k)\right]^2 \tag{4.7}$$

where $n_i \in N_i$ and $N_i = \{n_1, n_2, \ldots, n_h\}$ are the coincident points. The quadratic form with the parameter λ is added in order to penalize the control signal.

The horizon for this coincident points n_h is limited by the sampling time (Figure 4.10), a small number of points can produce an inadequate representation of the system behavior, and a high number of points make the optimizing cost function difficult, demanding more computational processing.

The free and forced solution

The key to any model-based controller lies in its ability to predict the process response. These fact is equivalent to the classical problem of solving differential equations, where their solution, from the instant $k = 0$ to the present time consists on two terms; the free and forced solution.

The free (unforced) solution $\hat{y}_{UF} \in \mathbb{R}$, referred as the homogeneous solution is defined as the output when the input is zero for $k > 0$, but was non-zero in the past. It represents the output when no further external stimulus is applied

The forced solution $\hat{y}_F \in \mathbb{R}$, referred as the inhomogeneous solution, makes the opposite assumption to that of the free. It implies that all past signals, both input and output, are zero. Then, to compute the future process output, the free and

forced responses are added:

$$\hat{y}(k) = \hat{y}_{UF}(k) + \hat{y}_F(k) \tag{4.8}$$

Basis functions

The future control signal is structured as a linear combination of predetermined basis functions:

$$u(k+i) = \sum_{j=1}^{n_b} \mu_j(k) U_j(i), \quad 0 \leq i \leq t_h \tag{4.9}$$

Here, μ_j are the associated gains to be computed during the optimization process, n_b is the number of these predefined functions U_j, which depends of the process nature. They can be in polynomial type; step, ramp, or parabola.

$$\begin{aligned} U_1(i) &= 1 \\ U_2(i) &= i \\ U_3(i) &= i^2 \end{aligned}$$

Predicted output

The predictive output over a defined finite horizon for the discrete model of equation (4.4) it is given by:

$$\begin{aligned} \hat{\theta}(k+i) &= \theta(k) + \Delta t \; u(k+i) \\ \hat{y}(i|k) &= f_h(\hat{\theta}(k+i)) \end{aligned} \tag{4.10}$$

where $\hat{\theta} \in \mathbb{R}$ is the flipper angle response to an input of the base equation (4.9), and $f_h \in \mathbb{R}$ is the correspond mobility height function.

Control law

The control law implies to calculate the gains μ_j of equation (4.9), these coefficients are the optimal at each instant k, thus they are different at each step. The prediction output $\hat{y}$ is obtained adding an autocompensation term calculated as a function of the observed differences between the model and past outputs:

$$\hat{y}(k+i|k) = y(k+i) + \hat{e}(k+i|k) \tag{4.11}$$

Where $y(k+i)$ is the output that is decomposed into a first term; free response or homogeneous solution, and a second term; forced response or homogeneous solution,

see equation (4.8). is given by:

$$y(k+i) = \sum_{j=1}^{n_b} \mu_j(k)\, Y_{Bj}(i|k) \tag{4.12}$$

Here Y_{Bj}, is the system output response to the basis function U_j, and its computed on the equation (4.10). Besides, the predicted output error $\hat{e} \in \mathbb{R}$ is assumed to have the form.

$$\hat{e}(k+i|k) = y_p(k) - \hat{y}(k|k-1) + \sum_{j=1}^{r} e_j i^j \tag{4.13}$$

Where r is the degree of a polynomial approximation for the error. And, the coefficients e_j are obtained on-line knowing the past and present output error.

All of the PFC variables are mounted in the cost function equation (4.7) as follow:

$$V = \sum_{j=1}^{n_h} \left[\hat{y}_h\left(k+n_j\right) - w\left(k+n_j\right)\right]^2 = \sum_{j=1}^{n_h} \left[Y_B\left(n_j\right)\mu(k) - d\left(k+n_j\right)\right]^2 \tag{4.14}$$

where,

$$\begin{aligned} \mu(k) &= \left[\mu_1(k) \ldots \mu_{n_b}(k)\right]^T \\ Y_B\left(n_j\right) &= \left[Y_{B1}\left(n_j\right) \ldots Y_{Bn_b}\left(n_j\right)\right] \\ d\left(k+n_j\right) &= w\left(k+n_j\right) - y(k) - e\left(k+n_j\right) \end{aligned} \tag{4.15}$$

Minimizing the cost function with respect to the coefficients μ :

$$V = (Y_B\mu(k) - d(k))^T (Y_B\mu(k) - d(k)) \tag{4.16}$$

$$\frac{\partial V}{\partial \mu} = Y_B^T Y_B \mu(k) - Y_B^T d(k) = 0 \tag{4.17}$$

here,

$$\begin{aligned} Y_B &= \left[Y_B\left(n_1\right) \ldots Y_B\left(n_{nh}\right)\right]^T \\ d(k) &= \left[d\left(k+n_1\right) \ldots d\left(k+n_{n_h}\right)\right]^T \end{aligned} \tag{4.18}$$

The vector of gains $\mu(k)$ can be compute by

$$Y_B\ \mu(k) = d(k) \tag{4.19}$$

Finally, the first term of the control signal, taking into account a finite horizon is given by:

$$u(k) = \sum_{j=1}^{n_b} \mu_j(k) U_j(0) \tag{4.20}$$

4.5.1 Proposed control scheme

The model presented in equation (4.4) can be represented as the upper flowchart of Figure 4.11, where kinematic control $u = \dot{\theta}$ goes through an mobility function f_h. And, their mathematical equivalence are presented on the lower flowchart, where the Jacobian for this mobility equation appears.

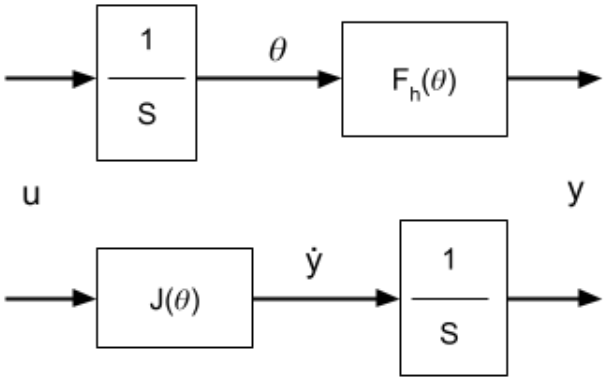


Figure 4.11: Process model equivalence

This equivalence is useful for multi-variables process, where the prediction can be made on the integration to compute the f_h values (upper flowchart). And, by the lower flow chart, an integrator is used to obtain the differential outputs $\dot{y}$, then an inverse Jacobian $J(\theta)^{-1}$ is used to obtain the new control signal gains μ_{nj}.

The PFC flowchart system used in this work is shown in Figure 4.12. Here; r_h is the reference height to be reached, y_p is the process output computed with the flippers measured angles.

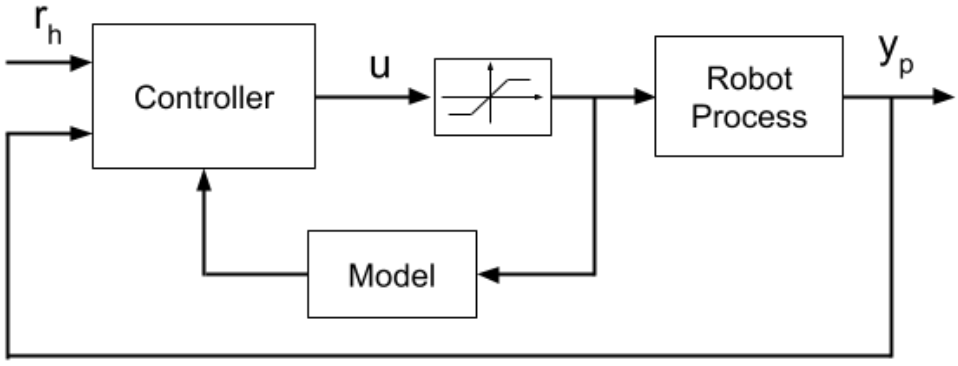


Figure 4.12: Block diagram of PFC controller

The saturator represents a maximum value for the angular velocity on the flippers actuators, where their output is taken into account by the model for a feedback computation.

Chapter 5

Simulations and Experiments

This section presents simulations and experiments we performed with a reconfigurable robot namely ROSI. For a navigation and obstacle negotiation task.

5.1 Robotic Operating System

Robot simulators allow us to build control programs to make an off-line test, where the success programming depends on how similar is the simulated to an real environment. For instance, In (SANTOS *et al.*, 2013) an Oil & Gas virtual simulator based on ROS is presented, It provides realistic scenarios where robot simulations can be validated by robots on a real operation. In (LUCCHI *et al.*, 2020), a framework to reduce the gap between simulation and real world is presented. Some components are depicted in Figure 5.1. It is assumed that the simulated and real robot use the same ROS controller, where a command handler publish the messages to robot components that emulate real actuators or joints. The ROS bridge collects and manages the robot nodes information that can be queried at any time.

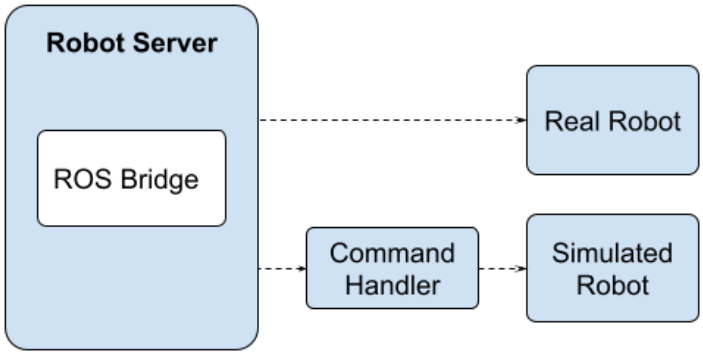


Figure 5.1: Simulation, and reality on ROS framework

In this work, the system was simulated on *CoppeliaSim* software (ROHMER *et al.*, 2013), by its versatility on assemble mobile robots and their supporting with

Figure 5.2: Rosi mobile robot used to test the LIO-SAM framework.

ROS nodes in which ROSI robot is built. Another point in which simulation environments help us is on the getting of simulated control parameters to transfer to the real robot; how it is shown in (MITRIAKOV *et al.*, 2021), where the authors proposed to get *learned* parameters in simulation when the robot traverse a staircase, and transferred to reality, they tested it in two different articulated tracked robots.

5.2 Wheeled Navigation

In this subtask, Rosi is operated on its wheeled configuration through indoor and outdoor scenarios to collect data from the sensors; Lidar, IMU, and camera. Then, to test SLAM algorithms previous configurations are necessary.

5.2.1 LIO-SAM hardware

To test the functionality of the LIO-SAM framework with the Rosi robot (Figure 5.2), two sensor were used; The first, a Velodyne sensor VLP-16 (left of Figure 5.3) which has a measurement range up to $100m$, with an accuracy of $\pm$ $0.03m$, the vertical field of view (FOV) of 30° (±15°) and a horizontal FOV of 360°. These 16-channel sensor provides a vertical resolution of 2°, and the horizontal angular resolution varies from 0.1° to 0.4° based on the rotation rate. The scan rate used was $10Hz$ which provides a horizontal angular resolution of 0.2°.

The second sensor, a miniature IMU Advanced Navigation - Motus (right of Figure 5.3), It has a high accuracy on the accelerometer and gyroscope. The axes: X, Y and Z determines the directions around in which the angles and accelerations are measured.

These package is implemented in `C++` and executed on a laptop equipped with an Intel i5-10210U CPU, using the robot operating system (ROS) in Ubuntu-Linux. The algorithm is freely available on Github.

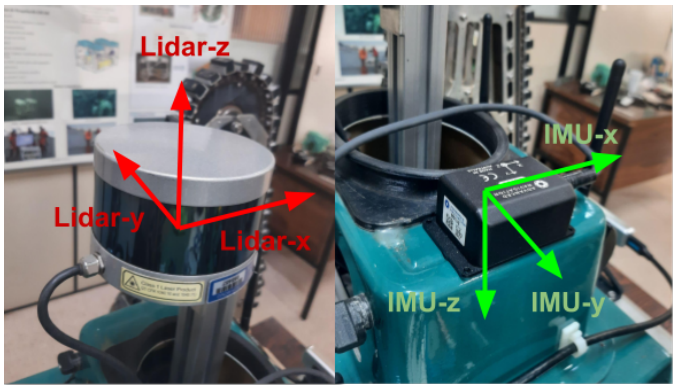


Figure 5.3: Velodyne and IMU equipment.

5.2.2 Composing the data

To achieve the operation of these package, its necessary to prepare the point clouds and the IMU data, both of them need to satisfy a certain alignment between their axes. The chosen configuration are displayed in Figure 5.3. Here, the lidar frame is the same with the robot frame (see Figure 5.2), and the IMU frame is rotated 180 ° around lidar-x axis.

LIO-SAM transforms IMU raw data from the IMU frame to the lidar frame. And, to make the system function properly on the Rosi robot, we modify the extrinsic matrix transformations as follow;

$$extRot = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \quad extRPY = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{5.1}$$

where the matrix rotation $extRot$ denotes the transformation of the IMU data in the lidar frame, and $extRPY$ denotes the rotation attitude measurement by 180 ° around lidar$-x$ to get the corresponding roll, pitch, and yaw readings in the lidar frame.

To check whether the readings correspond to the sensor movement, we did simulations only with the velodyne an the IMU, then we verified the outputs of the transformed IMU data, and checking if the readings correspond to the sensor movement.

We describe a series of experiments developed on the Laboratory environment.

5.2.3 Indoor scenario 1:

In this short scenario (Figure 5.4), we operated the robot with an average linear speed of $0.1m/s$, and an angular body robot speed of $0.1\,°/s$. Moreover, these scenario is affordable to test the performance of the lio-sam algorithm when the

robot make rotations.

Figure 5.4: Indoor scenario 1

(a) First side

(b) Second side

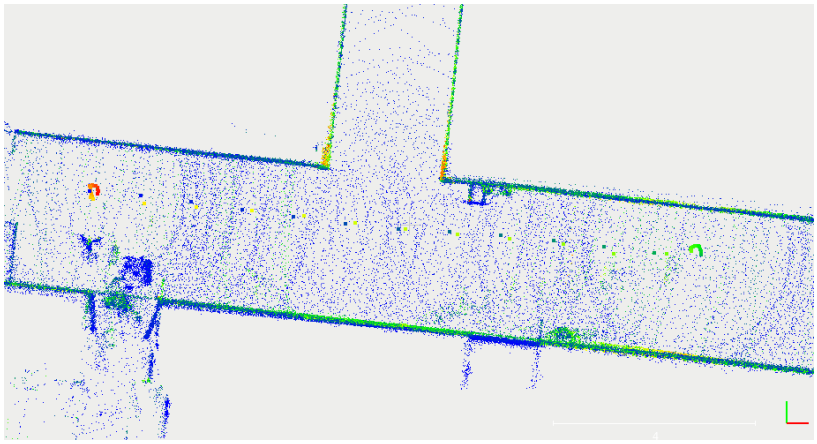


Figure 5.5: 3D map and trajectory on for the indoor scenario 1 on the straight line task

With the information above, two experiments were performed on the scenario 1. On the first, the robot drives in a straight line from an initial point $(0, 0)$ until a final $(12, 0)$ following black dots on the floor that were previously marked; on the final point the robot makes a rotation of $180\,°$, to return to the initial point. The resulting data processing from Lio-sam are shown in Figure 5.5.

To test the lio-sam odometry performance, in Figure 5.6 are displayed the robot path traveled (*base link*) with the marked dots on the floor.

Additionally, The camera Intel T265 is able to make an internal image processing and give us its odometry navigation that we can plot on the Figure 5.7, where the path *camera link* are contrasted with the marked dots on the floor. Here, it is

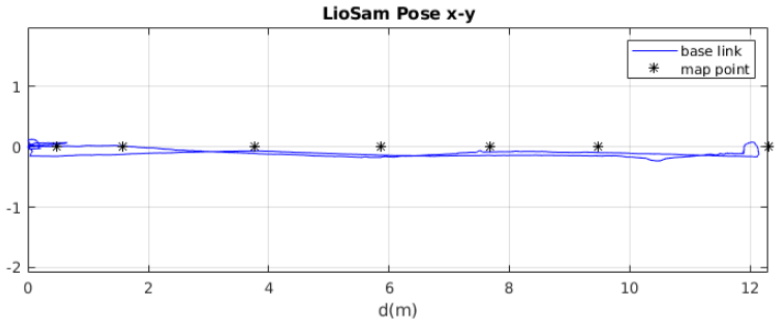


Figure 5.6: LIO-SAM odometry for a linear path.

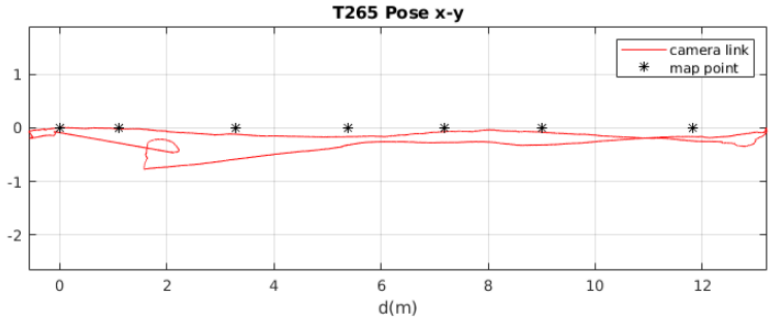
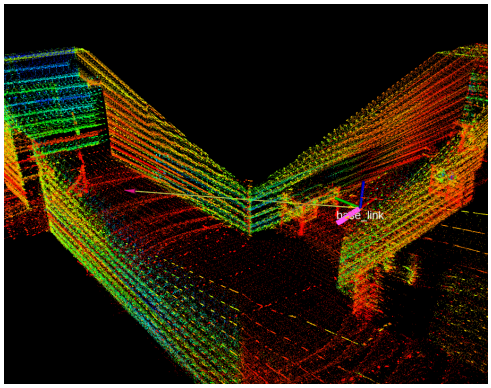


Figure 5.7: Camera odometry for a linear path.

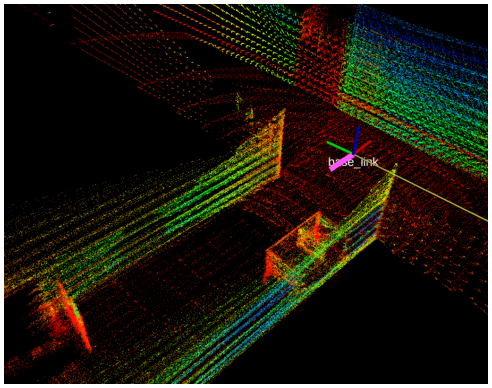
noticeable that the camera path deviates a bit on the way back, nevertheless it manage to recognize the initial scene.

Figure 5.8: Points clouds data processing for a complex scenario

(a) First scene



(b) Second scene



However, due to recurrent deviations from the lio-sam framework when the robot goes through complex scenarios. On the second experiment, the robot makes a rotation task following the black dots marked on the floor. The online LiO-SAM processing is shown in Figure 5.8, the velodyne points clouds, and the robot *base link* are displayed.

The 3D map and the robot trajectory for the second experiment is shown in Figure 5.9.

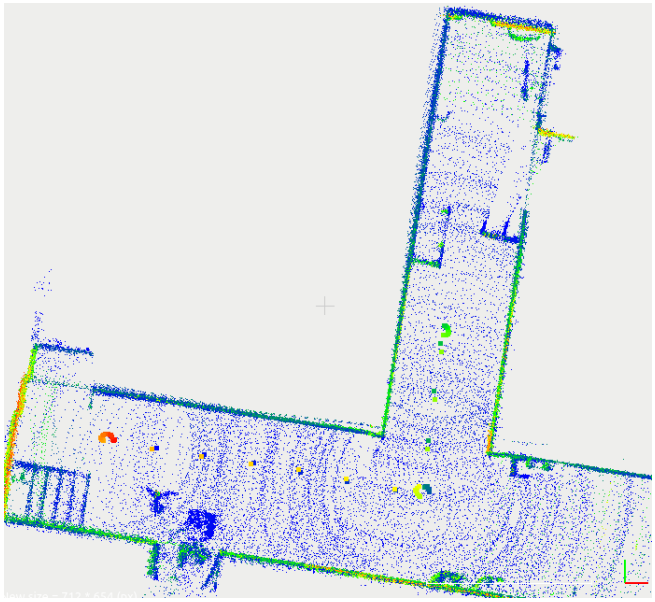


Figure 5.9: 3D map and trajectory for the indoor scenario 1 on a rotation task

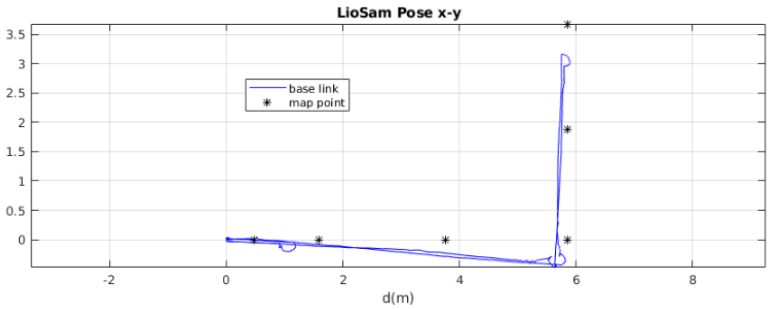


Figure 5.10: LIO-SAM odometry for a rotation task.

The lio-sam odometry for this experiment is shown in Figure 5.10, where the robot starts at point $(0, 0)$ until the point $(6, 3)$ following the black dots marked on the floor. Moreover, the robot performs minimal rotations in order to return to the initial point. Here, the path *base link* represents the CM of the body robot.

The corresponding odometry computed by the camera T265 algorithm for the second experiment is shown in Figure 5.11. it is observed that the path of *camera link* fails when the robot rotates 90 °.

5.2.4 Indoor scenario 2:

For this experiment, we drove the robot on an indoor environment that has a approximate length of $80m$ (Figure 5.12). The robot performs a round trip task.

This experiment was designed to evaluate the LiO-SAM framework when the robot travels long distances. The robot performs rotations up to 180 °, and the resulting map with the traveled path are shown in Figure 5.13.

Next, we computed the odometry robot from the lio-sam and the camera T265

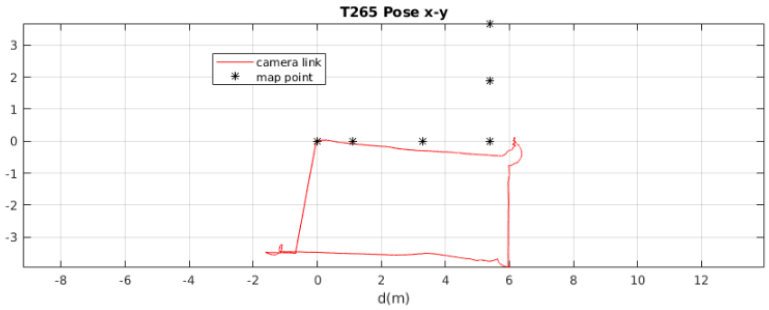


Figure 5.11: Camera odometry for the rotation task.

Figure 5.12: Indoor scenario 2

(a) First side

(b) Second side

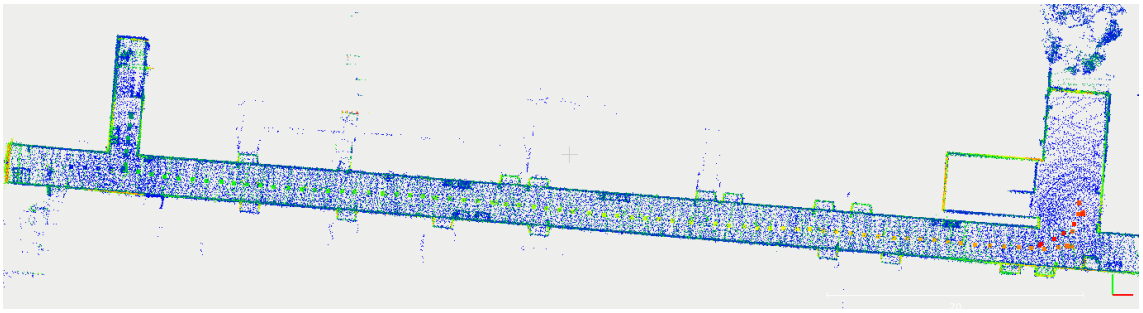


Figure 5.13: 3D map for the indoor scenario 2

to make a comparison how it is shown in Figure 5.14. The blue line represents the *base link*" path made by the lio-sam framework, it is observed a little inclination that is due to an initial deviation on the *yaw* angle measured by the IMU. The red line represents the *camera link* path computed by the camera T265, that shows a good response on the way out, however it presented a deviation on the way back.

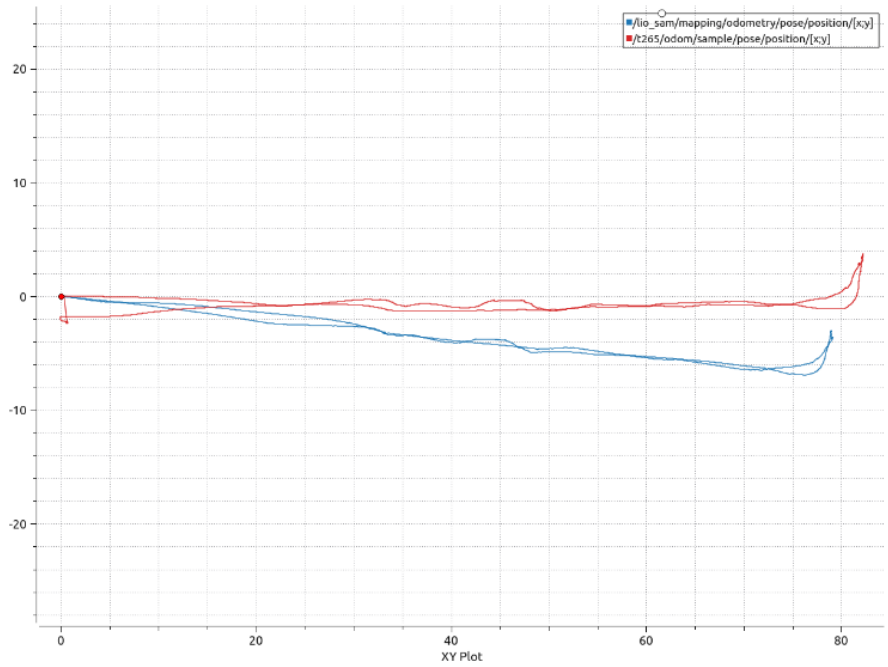


Figure 5.14: LIO-SAM and Camera Odometry.

5.2.5 Outdoor scenario

In this experiment, we operated Rosi robot through an outdoor scenario (Figure 5.15). It has a small slope and is affordable to perform rotations.

Figure 5.15: Outdoor scenario

(a) First side

(b) Second side

The resulting 3D map for this task was obtained analyzing the saved data with LIO-SAM, a top view is displayed on Figure 5.16.

Furthermore, on this point cloud map; the robot path, the rail, and vegetation can be perceived.

Figure 5.17, shows the robot trajectory obtained from LIO-SAM algorithm, It is plotted with respect to the Lidar frame (Velodyne).

And, the robot trajectory obtained from the camera T265 is shown in Figure 5.18, it can be seen that it has a low accuracy with respect to the obtained from

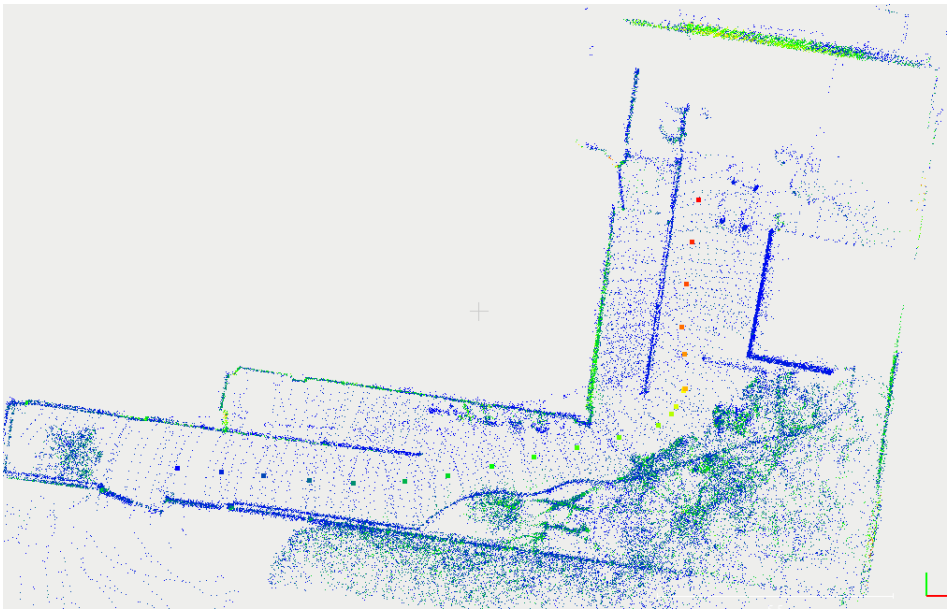


Figure 5.16: 3D map and trajectory for the outdoor scenario

LIO-SAM, specially in the rotation part.

Moreover, its noticed that this trajectory was made with respect to the *base link* (robot CM).

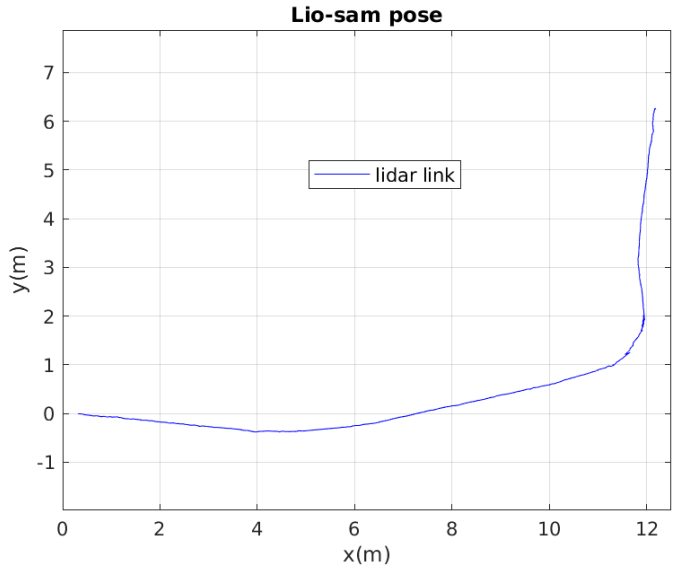


Figure 5.17: Lidar link trajectory for Rosi robot on an outdoor scenario

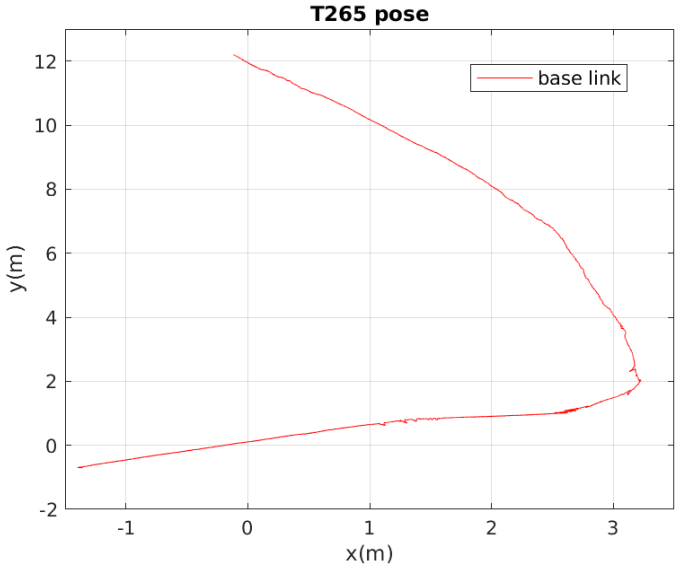


Figure 5.18: Base link trajectory for Rosi robot on an outdoor scenario

5.3 Obstacle Negotiation

In this section, simulations and experiments for a reconfigurable robot regarding to the task of facing obstacles are presented. The simulation environment where the robot was modeled is *CoppeliaSim*, (ROCHA, 2019). And, the experiments were made with the ROSI robot (Figure 2.2).

The simulator enable us to evaluate the robot behavior dealing with different types of obstacles in order to obtain the control parameters to later transfer them to the reality.

In a first experiment, we considered the model of equation (4.4) where a constant d, which is the flipper length , was added to consider the flippers upward as an initial condition, the model resulting is displayed in equation (5.2).

$$\begin{aligned} \theta(k+1) &= \theta(k) + \Delta t\ u(k) \\ y(k) &= d - f_h(\theta(k)) \end{aligned} \tag{5.2}$$

The system behaviour with the PFC approach is shown in the flowchart of Figure 4.12, where r_h is the reference height that represents distance between the robot CM and the plane formed by all the flippers endpoints, u is the signal control (angular velocity) for the flippers, and y_p is the process output that is computed using the flipper angles measured by encoders. Moreover, some remarkable PFC parameters that were used on both simulation and experiment are shown in Table 5.1

Table 5.1: PFC parameters for an initial test

Parameter	Value
α_h	0.4
r_h	0.6m

In this experiment, as we mention above, the flippers are upward as an initial condition $y_p = 0.0$, then they begin to downward until the system reach $r_h = 0.6\,m$, as it is shown in Figure 5.19. The experiment is executed in less than $9s$, and the sampling time was $\Delta\, t = 0.05s$, with a prediction horizon of $3s$.

The internal computation of PFC variables are displayed on Figure 5.20, 10 samples are plotted for the reference trajectory w, and the predicted output $\hat{y}$ at $1.5s$.

On Figure 5.21, the control signal that is applied to all the flippers is shown, thus the plane ϑ moved down horizontally towards the robot CM. The flipper angular velocity has a maximum value of $0.24\,rad/s$ that is defined on the saturator.

Furthermore, it is inferred that a positive response means that flipper is rotating to the floor direction, and the negative response, means that the flipper is moving away the floor.

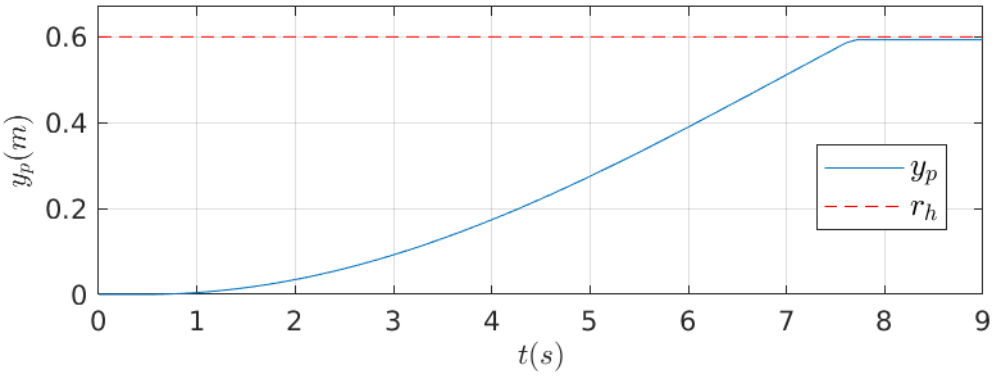


Figure 5.19: Process output y_p for an initial test

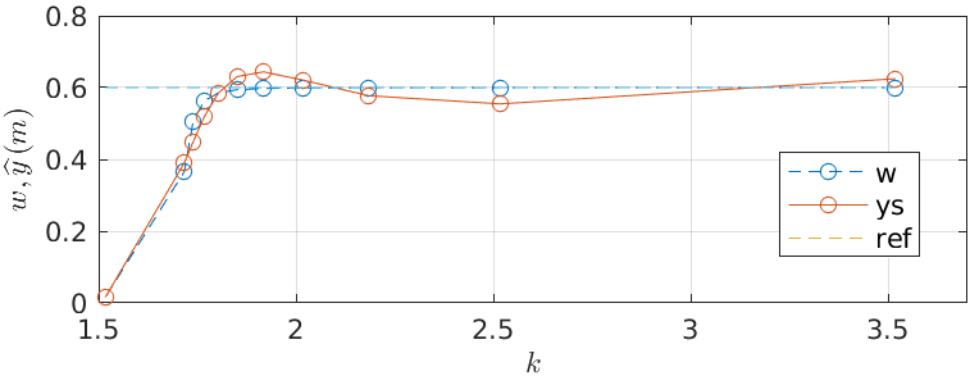


Figure 5.20: Samples (10) for PFC variables on an initial test

The tracking predicted error $\hat{e}$ of equation (4.13) is shown in Figure 5.22

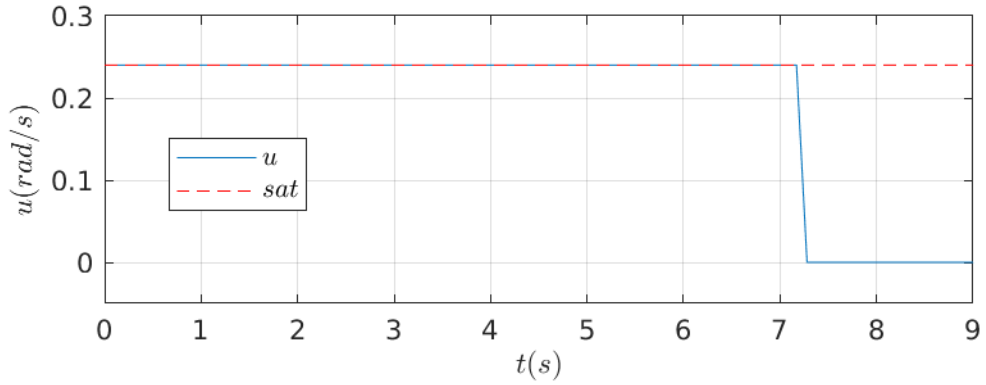


Figure 5.21: Control signal u, real robot on an initial test

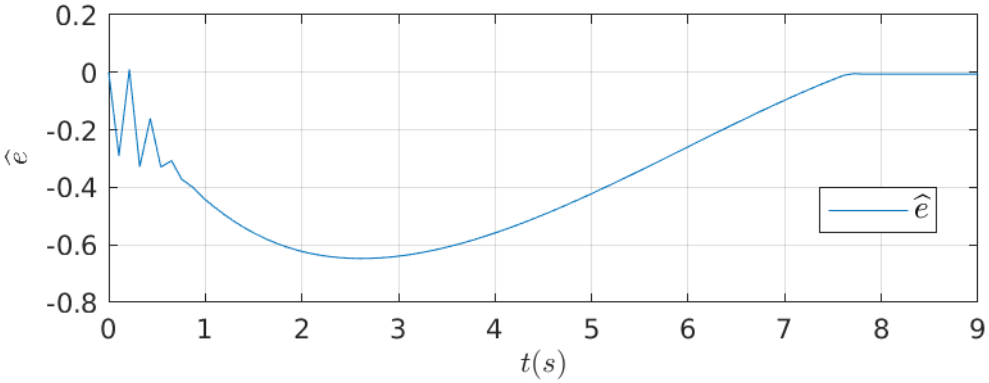


Figure 5.22: Tracking predicted error for an initial test

5.3.1 Step task

To make the ROSI robot to climb a step, extensive simulated and real-world experiments were conducted with varying the step shapes.

The robot go behind the sequence presented on Table 4.1, where the robot goes through five the states to overcome a step of $height = 0.14\,m$.

Then, the PFC parameters used both in simulation and experiment for the step sequence are displayed on Table 5.2.

Table 5.2: PFC parameters for a step task

State	α_h	$r_h(m)$
A	0.5	0.25
B	0.4	0.78
C	0.4	0.78
D	0.3	0.25

The simulation and experiment for these task are shown in Figure 5.23. Here, the robot starts from an initial condition moving with a defined linear velocity, and their flippers upward -first left graphic, going to the state A where the robot flippers must to reach the relative height $r_h = 0.25\,m$, with the parameter $\alpha_h = 0.5$. To go to state B, the body robot must to pass the negative threshold θ_{nt}, and the robot has to reach a $r_h = 0.78\,m$ with $\alpha_h = 0.4$. On the state C, the robot maintains the parameters value of r_h and α_h, at this moment the robot has their flippers downward. To go to state D, the robot has to get over the positive threshold θ_{pt}, where the robot flippers must to reach $r_h = 0.25\,m$, with $\alpha_h = 0.3$. Finally, the robot returns to state A.

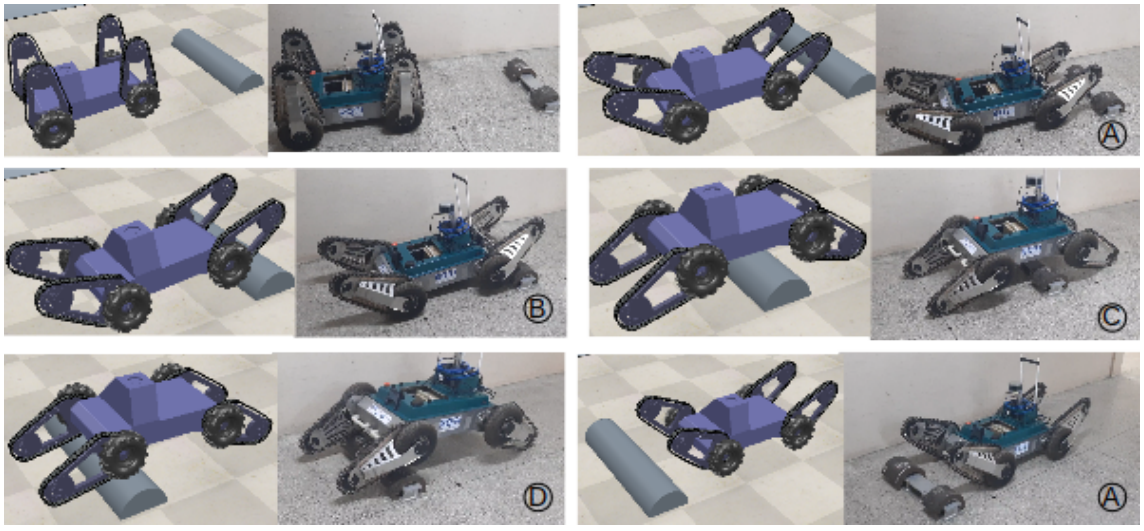


Figure 5.23: Robot sequence to climb a step, on the simulation environment and reality.

The values of positive and negative threshold for the simulated and real step

task are shown in Table 5.3, where the difference between them, was due to IMU deviations on their *pitch* angle.

Table 5.3: Threshold values for simulation and real step task

Threshold	Sim	Real
θ_{pt}	$3.0\,^\circ$	$2.5\,^\circ$
θ_{nt}	$-3.0\,^\circ$	$-4.0\,^\circ$

The process output y_p for the whole step task described above is shown in Figure 5.24, where its noticed the two reference heights; on $r_h = 0.25m$ and $r_h = 0.78m$. And, all the step task was completed in less than $30s$.

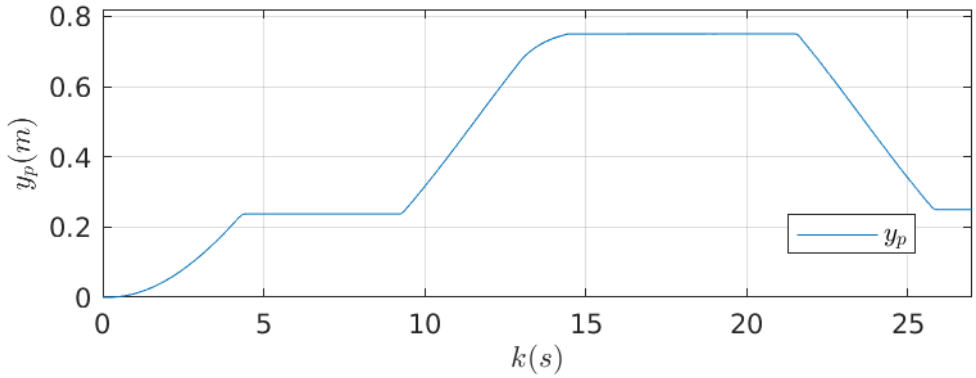


Figure 5.24: Process output of real robot on the step task

The control signal u for the complete step task is shown in Figure 5.25, this angular velocity goes to each flipper independently, being that its maximum value was on $u = 0.22rad/s$ due to the saturator. The linear robot velocity was set on $0.11m/s$. Furthermore, it is noticed that state A is completed before $5s$; the state B starts at $9s$, and its complemented with the state C until the $20s$; the state D starts at $22s$ and its complemented with the condition of state A, finishing the whole process in $27s$.

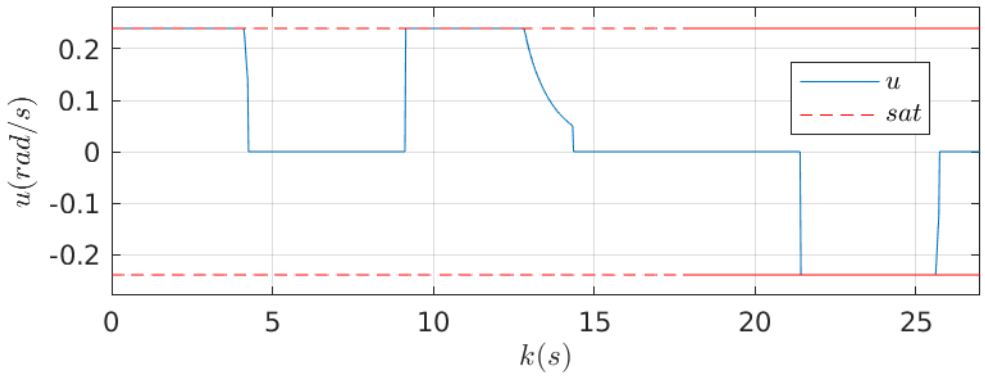


Figure 5.25: Control signal for real robot on step task

In Figure 5.26, the tracking predicted error $\hat{e}$ for the whole step task is indicated, its observed that for each stage of the sequence this error converges to zero.

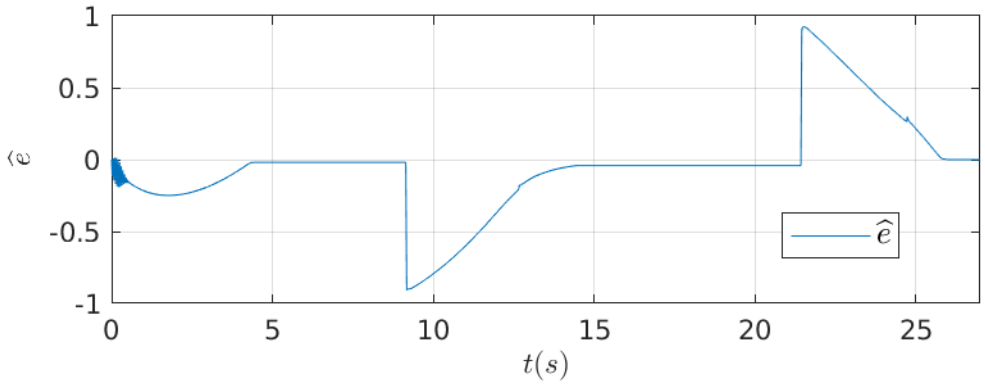


Figure 5.26: Tracking predicted error for real robot on step task

Another aspect for the control program, was its execution from a host computer through ROS nodes; Always taking into account the loss robot connection, which would affect the frequency operation of $40Hz$.

We evaluated the performance of these autonomous control (AC) sequence on the step obstacle, this task is repeated six times (6) to allow statistical evaluation, Figure 5.27 denotes the quartiles and median ($34s$) for measured values in which the step task was finished.

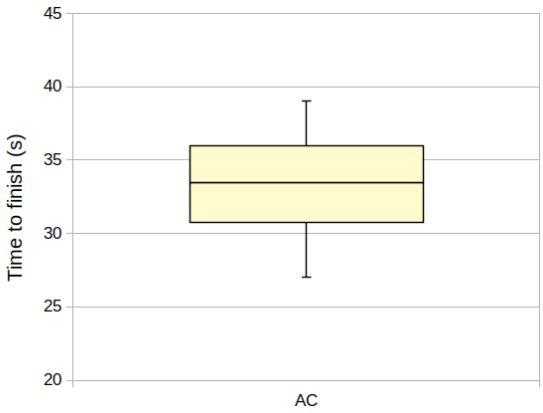


Figure 5.27: Quartiles and median for AC mode on step task

5.3.2 Rail task

For the case of dealing a rail profile; AS68 of $height = 0.18\,m$, simulations and experiments were performed. And, as in the step case, the robot go behind the sequence presented on Table 4.1, where the PFC parameters used both in simulation and experiment for the rail sequence are displayed on Table 5.4.

Table 5.4: PFC parameters for a rail task

State	α_h	$r_h(m)$
A	0.3	0.3
B	0.3	0.86
C	0.3	0.86
D	0.3	0.3

The simulation and experiment for the rail task are shown in Figure 5.28, where the robot starts from an initial condition moving with a defined linear velocity and their flippers upward, going to the state A where the robot flippers must to reach the relative height $r_h = 0.3\,m$, with $\alpha_h = 0.3$, as shown in Table 5.4. To go toward state B, the body robot must to pass a negative threshold θ_{nt}. In this state the robot has to reach $r_h = 0.86\,m$ with the same value of α_h. On the state C, the robot maintains the parameters value of r_h and α_h, at this moment the robot has their flippers downward. To go to state D, the robot has to get over a positive threshold θ_{pt}, where the robot flippers must to reach $r_h = 0.3\,m$, with the same value of α_h. Finally, the robot returns to state A.

Figure 5.28: Robot sequence to climb a rail, on the simulation environment and reality.

As in the step case, the values of positive and negative threshold for the simulated and real rail task are shown in Table 5.5.

The process output y_p for the rail task is displayed in Figure 5.29, where its noticed the two reference heights; on $r_h = 0.3m$ and $r_h = 0.86m$, completing all the rail task in less than $25s$.

Table 5.5: Threshold values for simulation and real rail task

Threshold	Sim	Real
θ_{pt}	$3.0\,^\circ$	$2.5\,^\circ$
θ_{nt}	$-3.0\,^\circ$	$-4.0\,^\circ$

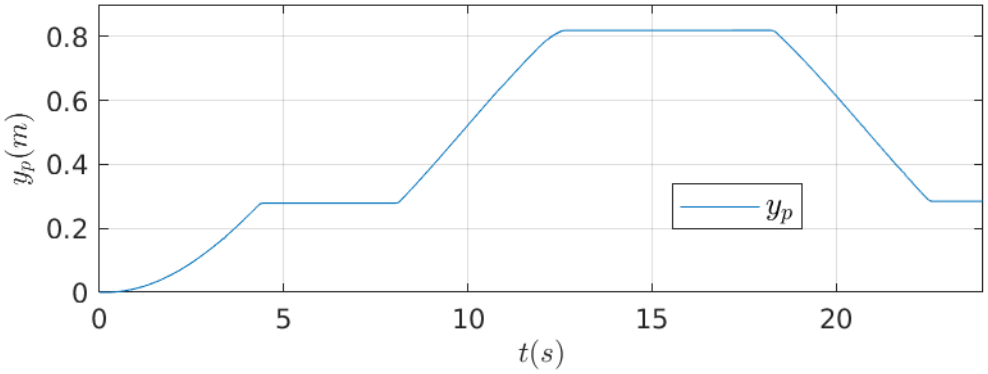


Figure 5.29: Process output for the real robot on rail task

The control signal u for the rail task is shown in Figure 5.30, being that its maximum value was on $u = 0.24rad/s$ due to the saturator. The linear robot velocity was set on $0.13m/s$. Furthermore, it is noticed that state A is completed before $5s$; the state B starts at $8s$, and its complemented with the state C until the $15s$; the state D starts at $18s$ and its complemented with the condition of state A, finishing the task in less than $25s$.

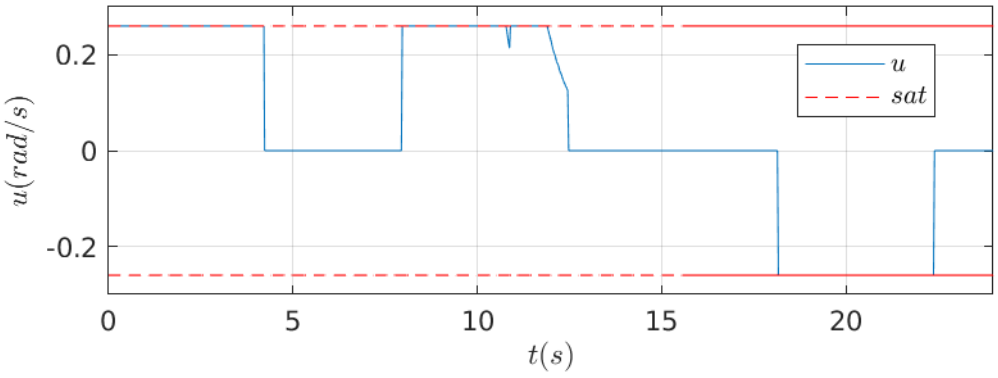


Figure 5.30: Control signal for the real robot on rail task

In Figure 5.31, the tracking predicted error $\hat{e}$ for the whole rail task is displayed, and as in the step case, this error converges to zero on each stage of the sequence.

To evaluate the performance of these AC mode on the rail obstacle, this experiment was performed seven times (7), and Figure 5.32 denotes the quartiles and median ($24s$) for measured values in which the rail task was completed.

Additionally, Figure 5.33 denotes the robot CM trajectory (*base link*) on the $x - z$ axis, measured by the camera T265.

And, Figure 5.34 presents the *lidar link* trajectory on the $x - z$ axis, measured

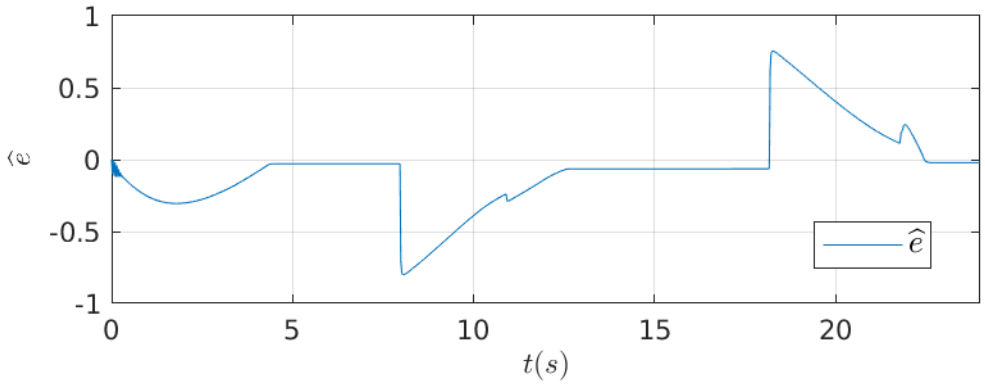


Figure 5.31: Tracking predicted error for the real robot on rail task

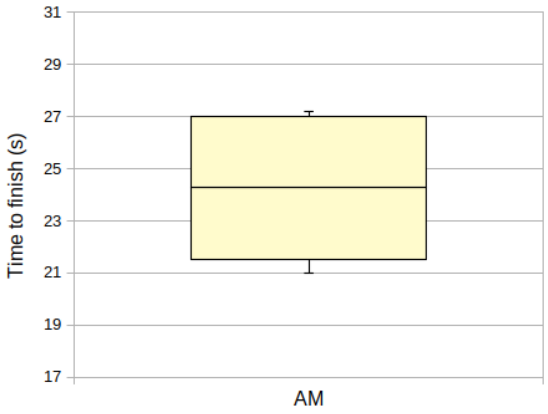


Figure 5.32: Quartiles and median for AC mode on rail task

by the LIO-SAM algorithm.

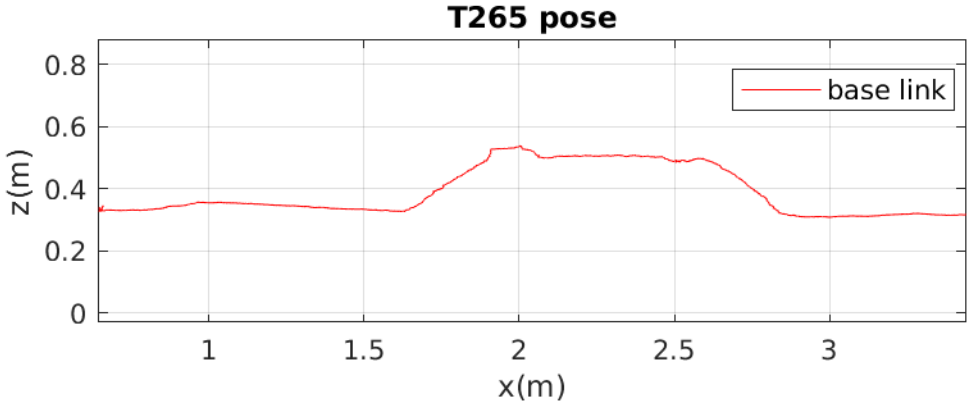


Figure 5.33: CM trajectory for the real robot on the rail task

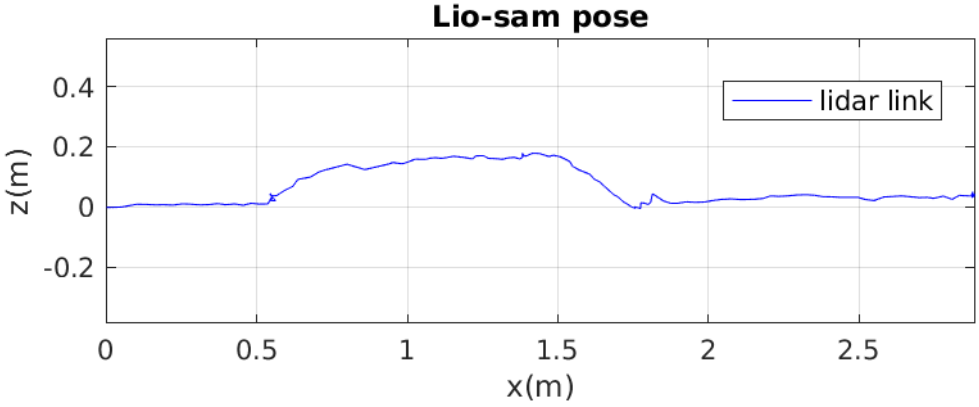


Figure 5.34: Lidar base link trajectory for the real robot on the rail task

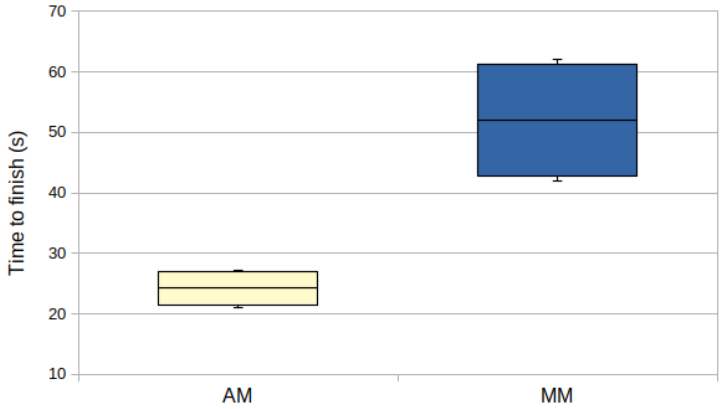


Figure 5.35: Comparison between a manual an automatic mode

5.4 Inspection task

In this experiment, we emulate an inspection task, in which the robot has to reach an inspection point, where there is an obstacle (rail) on the path, made the inspection the robot has to return to the starting point, the trajectory points are displayed in Figure 5.36.

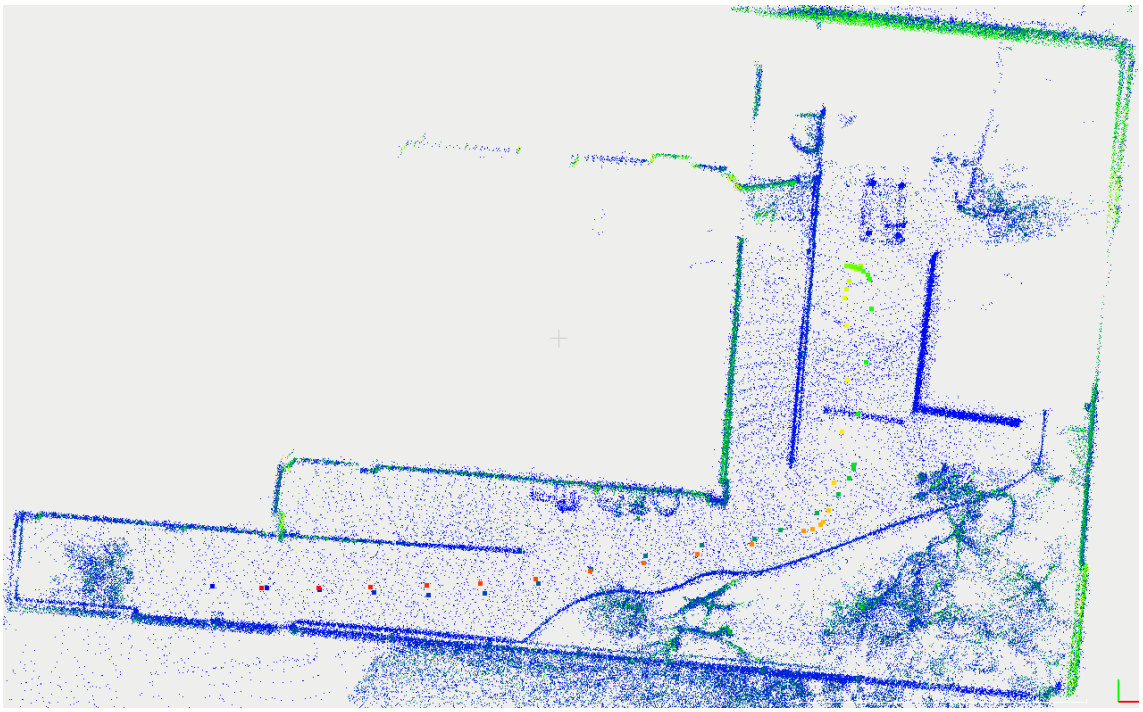


Figure 5.36: 3D map and trajectory for the inspection task

Figure 5.37 shows the odometry obtained from the lio-sam algorithm. Its observed that it has not alteration on the odometry when the robot climb the rail.

Figure 5.38 shows the odometry obtained from the camera T265. contrary to the graphic above, it is observed some differences on the plane x-y on the initial and the final. Moreover some differences que the robot make a rotation an when the robot climb the rail.

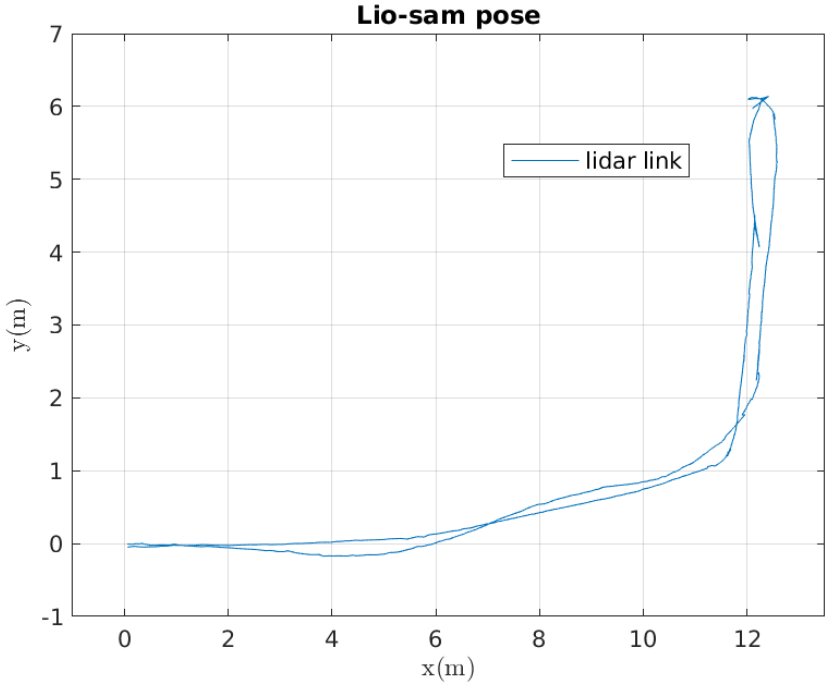


Figure 5.37: Lidar link trajectory for Rosi robot on an inspection task

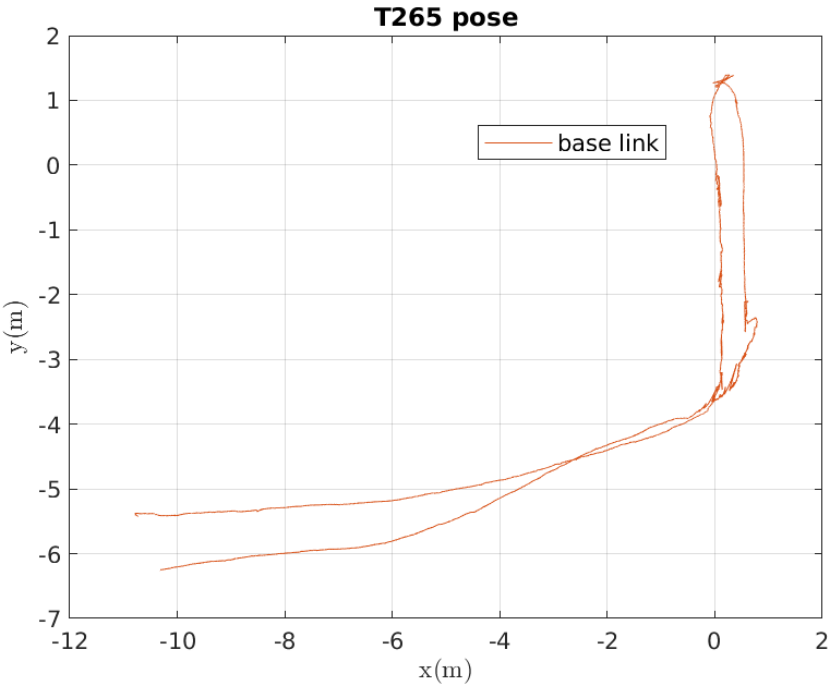


Figure 5.38: Base link trajectory for Rosi robot on an inspection task

Chapter 6

Conclusions and Future Work

In this chapter, we present the conclusions and the future work about the research theme considered in this thesis.

6.1 Conclusions

- We tested and validate the LIO-SAM algorithm in indoor and outdoor scenarios, obtaining the map and the path inspection.
- The odometry given by LIO-SAM framework has a better performance than given by a cameraT265.
- We validate the formulation of PFC to control the relative robot height with a coordinate flippers movement.
- We validate a flippers motion sequence on several experiments on the ROSI robot to overcome a step and a rail task.

6.2 Technical Contributions

- In the LIO-SAM program execution, details about IMU and LiDAR physical orientation had to be solved to run properly, Furthermore, redundancy on the namely of Rosi \tf's, and Kinova \tf's need to be resolved for combined work.
- Sensors; camera, Lidar and IMU, has their own frames (locations on the robot boby), then \tf's with respect to the same *base link* (robot geometric center) are required.
- To guarantee an approximating on the process model, communications rates between the robot and the host computer need to be resolved.

- To perform a waypoint tracking, the on-line LIO-SAM is required, for which a high computing cost is employed, still on a remote PC.

6.3 Future Work

- the adding of a body orientation variable in the cost function, Considering DoF for the fronts and rear flippers, while the robot climb as step, ramp, etc. Moreover, the use of a LiDAR sensor to evaluate the obstacles height.
- The use of a LiDAR sensor to evaluate the obstacles height.
- The use of quaternions approach to face the control of the robot body orientation, due to the independent motion of flippers.

Bibliography

[1] CARVALHO, G. P., XAUD, M. F., MARCOVISTZ, I., NEVES, A. F., COSTA, R. R. "The DORIS offshore robot: recent developments and real-world demonstration results", *IFAC-PapersOnLine*, v. 50, n. 1, pp. 11215–11220, 2017.

[2] GARCIA, G., ROCHA, F., TORRE, M., SERRANTOLA, W., LIZARRALDE, F., FRANCA, A., PESSIN, G., FREITAS, G. "ROSI: A novel robotic method for belt conveyor structures inspection". In: *2019 19th International conference on advanced robotics (ICAR)*, pp. 326–331. IEEE, 2019.

[3] SZREK, J., WODECKI, J., BŁAŻEJ, R., ZIMROZ, R. "An inspection robot for belt conveyor maintenance in underground mine—Infrared thermography for overheated idlers detection", *Applied Sciences*, v. 10, n. 14, pp. 4984, 2020.

[4] ANYBOTICS AG. "ANYmal robot". `https:https://www.anybotics.com/`, 2022.

[5] FARIA, H. D., LIZARRALDE, F., COSTA, R. R., et al. "ROSI: a mobile robot for inspection of belt conveyor", *IFAC-PapersOnLine*, v. 53, n. 2, pp. 10031–10036, 2020.

[6] ROCHA, F., GARCIA, G., PEREIRA, R. F., et al. "ROSI: A Robotic System for Harsh Outdoor Industrial Inspection-System Design and Applications", *Journal of Intelligent & Robotic Systems*, v. 103, n. 2, pp. 1–22, 2021.

[7] KHAN, S. *Design, testing and validation of model predictive control for an unmanned ground vehicle.* Tese de Doutorado, University of New South Wales, 2022.

[8] SHAN, T., ENGLOT, B. "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4758–4765, 2018.

[9] SHAN, T., ENGLOT, B., MEYERS, D., WANG, W., RATTI, C., DANIELA, R. "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5135–5142, 2020.

[10] KIM, P. *EFFECTIVE NAVIGATION AND MAPPING OF A CLUTTERED ENVIRONMENT USING A MOBILE ROBOT*. Tese de Doutorado, Georgia Institute of Technology, 2020.

[11] BAKER, G., BRIDGWATER, T., BREMNER, P., GIULIANI, M. "Towards an immersive user interface for waypoint navigation of a mobile robot", *arXiv preprint arXiv:2003.12772*, 2020.

[12] BANSAL, S., TOLANI, V., GUPTA, S., MALIK, J., TOMLIN, C. "Combining optimal control and learning for visual navigation in novel environments". In: *Conference on Robot Learning*, pp. 420–429. PMLR, 2020.

[13] MICHALEK, M., KOZLOWSKI, K. "Vector-Field-Orientation feedback control method for a differentially driven vehicle", *IEEE Transactions on Control Systems Technology*, v. 18, n. 1, pp. 45–65, 2009.

[14] LIU, Y., LIU, G. "Track-stair and vehicle-manipulator interaction analysis for tracked mobile manipulators climbing stairs". In: *2008 IEEE International Conference on Automation Science and Engineering*, pp. 157–162. IEEE, 2008.

[15] LIN, J., GOLDENBERG, A. A. "Development of a terrain adaptive tracked vehicle and its derivative-dual mode vehicle". In: *2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, pp. 302–307. IEEE, 2018.

[16] WANG, W., DONG, W., SU, Y., WU, D., DU, Z. "Development of search-and-rescue robots for underground coal mine applications", *Journal of Field Robotics*, v. 31, n. 3, pp. 386–407, 2014.

[17] GIANNI, M., FERRI, F., MENNA, M., PIRRI, F. "Adaptive Robust Three-dimensional Trajectory Tracking for Actively Articulated Tracked Vehicles", *Journal of Field Robotics*, v. 33, n. 7, pp. 901–930, 2016.

[18] JUN, J.-Y., SAUT, J.-P., BENAMAR, F. "Pose estimation-based path planning for a tracked mobile robot traversing uneven terrains", *Robotics and Autonomous Systems*, v. 75, pp. 325–339, 2016.

[19] COLAS, F., MAHESH, S., POMERLEAU, F., LIU, M., SIEGWART, R. "3D path planning and execution for search and rescue ground robots". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 722–727. IEEE, 2013.

[20] GAWAD, M. A. A. "Mechatronic Suspension Systems: A Survey and Directions for Future Work", *American Journal of Engineering Research (AJER)*, v. 10, pp. 237–244, 2021.

[21] RAUH, J., AMMON, D. "System dynamics of electrified vehicles: some facts, thoughts, and challenges", *Vehicle System Dynamics*, v. 49, n. 7, pp. 1005–1020, 2011.

[22] CARVALHO, L. *Locomoção Semiautônoma Em Escadas Para Robôs Móveis Com Esteiras*. Tese de Mestrado, Federal University of Rio de Janeiro, 2016.

[23] OHNO, K., MORIMURA, S., TADOKORO, S., KOYANAGI, E., YOSHIDA, T. "Semi-autonomous control system of rescue crawler robot having flippers for getting over unknown-steps". In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3012–3018. IEEE, 2007.

[24] ZIMMERMANN, K., ZUZANEK, P., REINSTEIN, M., HLAVAC, V. "Adaptive traversability of unknown complex terrain with obstacles for mobile robots". In: *2014 IEEE international conference on robotics and automation (ICRA)*, pp. 5177–5182. IEEE, 2014.

[25] ROSSITER, J. A. *Model-based predictive control: a practical approach*. CRC press, 2003.

[26] RAWLINGS, J. B., MAYNE, D. Q., DIEHL, M. *Model predictive control: theory, computation, and design*, v. 2. 1, Nob Hill Publishing Madison, WI, 2017.

[27] CAMACHO, E. F., ALBA, C. B. *Model predictive control*. Springer Science & Business Media, 2013.

[28] NAUS, G., VAN DEN BLEEK, R., PLOEG, J., SCHEEPERS, B., VAN DE MOLENGRAFT, R., STEINBUCH, M. "Explicit MPC design and performance evaluation of an ACC Stop-&-Go". In: *2008 American Control Conference*, pp. 224–229. IEEE, 2008.

[29] TAKAHAMA, T., AKASAKA, D. "Model Predictive Control Approach to Design Practical Adaptive Cruise Control for Traffic Jam", *International Journal of Automotive Engineering*, v. 9, n. 3, pp. 99–104, 2018.

[30] FREITAS, G., LIZARRALDE, F., HSU, L., BERGERMAN, M. "Terrain model-based anticipative control for articulated vehicles with low bandwidth actuators". In: *2013 IEEE International Conference on Robotics and Automation*, pp. 382–389, Karlsruhe, Germany, maio 2013. IEEE. ISBN: 978-1-4673-5643-5 978-1-4673-5641-1. doi: 10.1109/ICRA.2013.6630604.

[31] FREITAS, G. M. *Reconfiguração de Robôs Móveis com Articulação Ativa Navegando em Terrenos Irregulares*. Tese de Doutorado, Universidade Federal do Rio de Janeiro, 2014.

[32] RICHALET, J., EL ATA-DOSS, S. A., ARBER, C., KUNTZE, H., JACUBASCH, A., SCHILL, W. "Predictive functional control-application to fast and accurate robots", *IFAC Proceedings Volumes*, v. 20, n. 5, pp. 251–258, 1987.

[33] RICHALET, J. "Industrial applications of model based predictive control", *Automatica*, v. 29, n. 5, pp. 1251–1274, 1993.

[34] BOUHENCHIR, H., CABASSUD, M., LE LANN, M.-V. "Predictive functional control for the temperature control of a chemical batch reactor", *Computers & Chemical Engineering*, v. 30, n. 6-7, pp. 1141–1154, 2006.

[35] ZHANG, R., XUE, A., WANG, S. "Modeling and nonlinear predictive functional control of liquid level in a coke fractionation tower", *Chemical engineering science*, v. 66, n. 23, pp. 6002–6013, 2011.

[36] ZHANG, Z., WANG, W. Q., SIDDIQUI, S. "Predictive function control of a two-link robot manipulator". In: *IEEE International Conference Mechatronics and Automation, 2005*, v. 4, pp. 2004–2009. IEEE, 2005.

[37] SATOH, T., SAITO, N., NAGASE, J.-Y., SAGA, N. "Predictive functional control of an axis positioning system with an estimator-based internal model", *Control Engineering Practice*, v. 86, pp. 1–10, maio 2019. ISSN: 09670661. doi: 10.1016/j.conengprac.2019.02.006.

[38] NAGASE, J.-Y., HAMADA, K., SATOH, T., SAGA, N., SUZUMORI, K. "Comparison between PFC and PID control system for tendon-driven balloon actuator". In: *IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society*, pp. 3398–3403. IEEE, 2013.

[39] FITZPATRIK, R. *Analytical classical Dynamics An Intermediate Level Course*. Austin, The University of Texas at Austin, 2008.

[40] MURRAY, R. M., SASTRY, S. S., ZEXIANG, L. *A Mathematical Introduction to Robotic Manipulation*. 1st ed. Boca Raton, FL, USA, CRC Press, Inc., 1994. ISBN: 0849379814.

[41] JAIN, A., RODRIGUEZ, G. "Recursive flexible multibody system dynamics using spatial operators", *Journal of Guidance, Control, and Dynamics*, v. 15, n. 6, pp. 1453–1466, 1992.

[42] SICILIANO, B., SCIAVICCO, L., VILLANI, L., ORIOLO, G. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[43] SIEGWART, R., NOURBAKHSH, I. R., SCARAMUZZA, D. *Introduction to autonomous mobile robots*. MIT press, 2011.

[44] BARSHAN, B., KUC, R. "A bat-like sonar system for obstacle localization", *IEEE Transactions on systems, man, and cybernetics*, v. 22, n. 4, pp. 636–646, 1992.

[45] *Scanning Laser Range Finder*. Hokuyo Automatic, 2009. URG-04LX-UG01.

[46] FISCHLER, M. A., BOLLES, R. C. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, v. 24, n. 6, pp. 381–395, 1981.

[47] PUENTE, I., GONZÁLEZ-JORGE, H., MARTÍNEZ-SÁNCHEZ, J., ARIAS, P. "Review of mobile mapping and surveying technologies", *Measurement*, v. 46, n. 7, pp. 2127–2145, 2013.

[48] KIDD, J. R. *Performance evaluation of the Velodyne VLP-16 system for surface feature surveying*. Tese de Doutorado, University of New Hampshire, 2017.

[49] *VLP-16 User Manual*. Velodyne LiDAR, 2018. Rev. D.

[50] KEARNS, K. "The Effects of Sensor Fusion on Localisation in a Sparse, Outdoor Environment". 2020.

[51] JÚNIOR, G. P. C., REZENDE, A. M., MIRANDA, V. R., FERNANDES, R., AZPÚRUA, H., NETO, A. A., PESSIN, G., FREITAS, G. M. "EKF-LOAM: An Adaptive Fusion of LiDAR SLAM With Wheel Odometry and Inertial Data for Confined Spaces With Few Geometric Features", *IEEE Transactions on Automation Science and Engineering*, 2022.

[52] *Motus Reference Manual*. Advanced Navigation, 2020. Version 2.0.

[53] RICHALET, J., O'DONOVAN, D. *Predictive functional control: principles and industrial applications*. Springer Science & Business Media, 2009.

[54] SANTOS, I. H. F., RIBEIRO, G. M., COUTINHO, F., et al. "A robotics framework for planning the offshore robotizing using virtual reality techniques". In: *OTC Brasil*. OnePetro, 2013.

[55] LUCCHI, M., ZINDLER, F., MÜHLBACHER-KARRER, S., PICHLER, H. "robo-gym–an open source toolkit for distributed deep reinforcement learning on real and simulated robots". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5364–5371. IEEE, 2020.

[56] ROHMER, E., SINGH, S. P., FREESE, M. "V-REP: A versatile and scalable robot simulation framework". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326. IEEE, 2013.

[57] MITRIAKOV, A., PAPADAKIS, P., KERDREUX, J., GARLATTI, S. "Reinforcement Learning Based, Staircase Negotiation Learning: Simulation and Transfer to Reality for Articulated Tracked Robots", *IEEE Robotics & Automation Magazine*, v. 28, n. 4, pp. 10–20, 2021.

[58] ROCHA, F. "rosiChallenge-sbai2019". `https://github.com/filRocha/rosiChallenge-sbai2019`, 2019.

Appendix A

Additional information

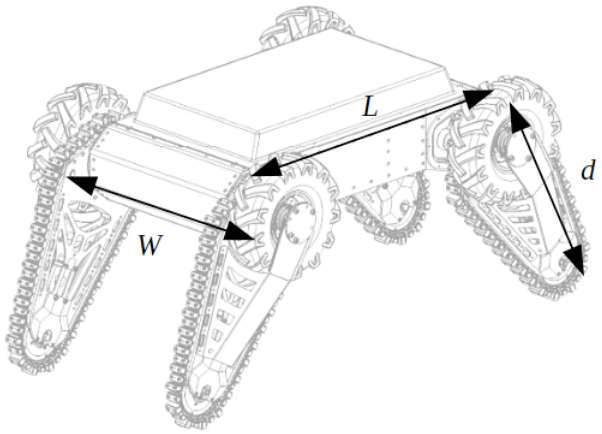


Figure A.1: Representation of dimensions for a reconfigurable robot - ROSI

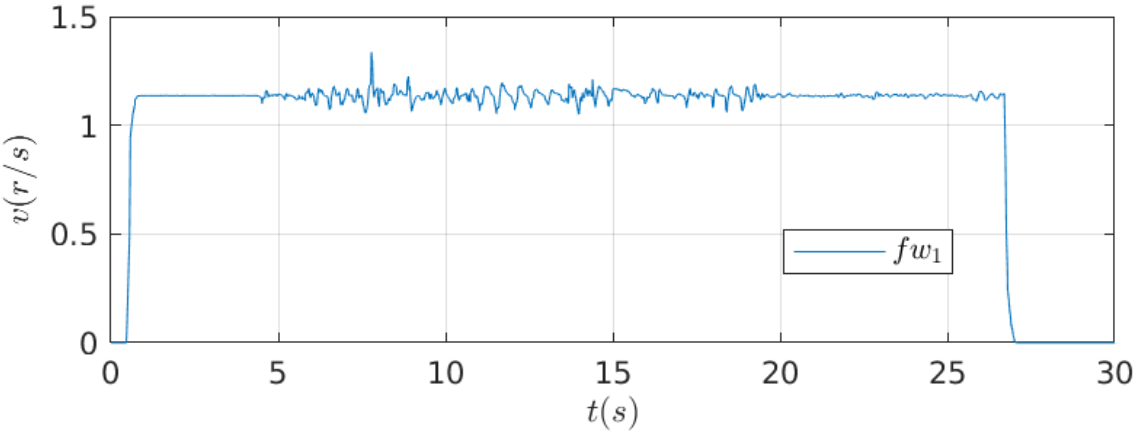


Figure A.2: Front wheel (left) velocity on the rail task

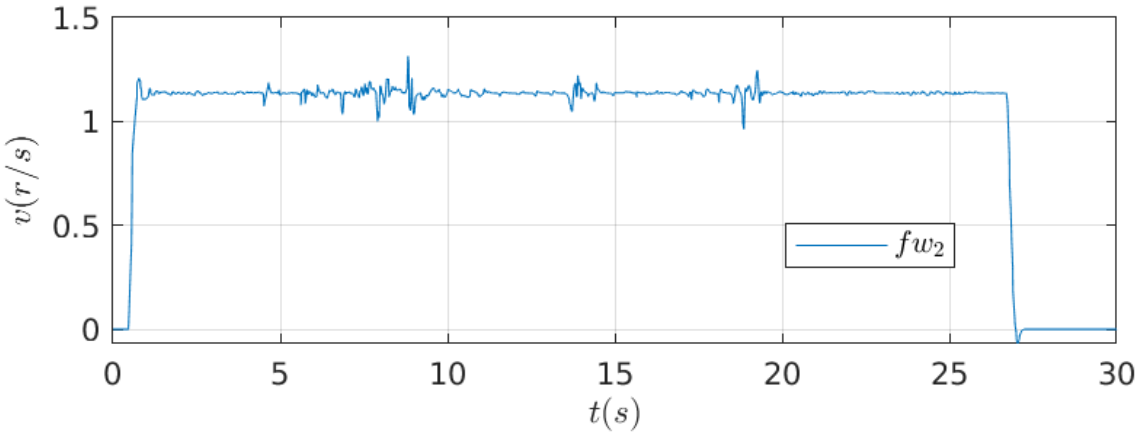


Figure A.3: Front wheel (right) velocity on the rail task

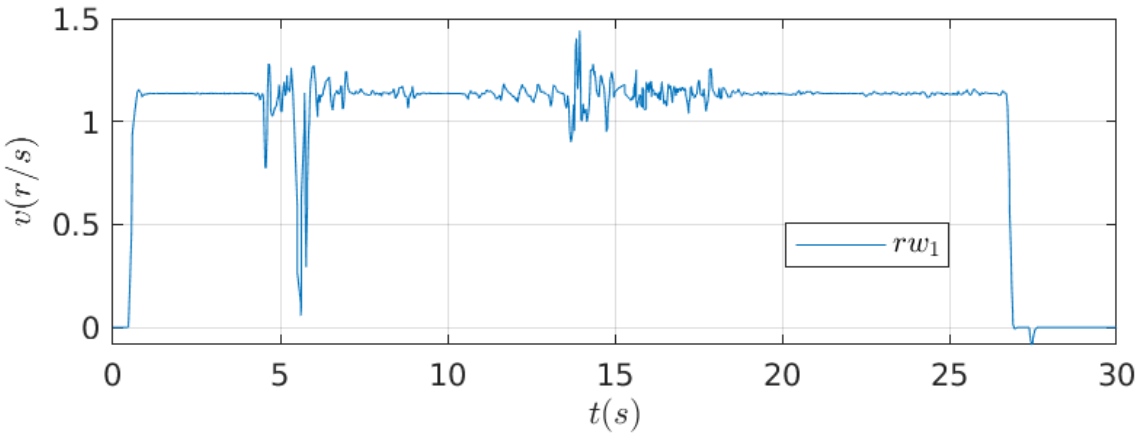


Figure A.4: Rear wheel (left) velocity on the rail task

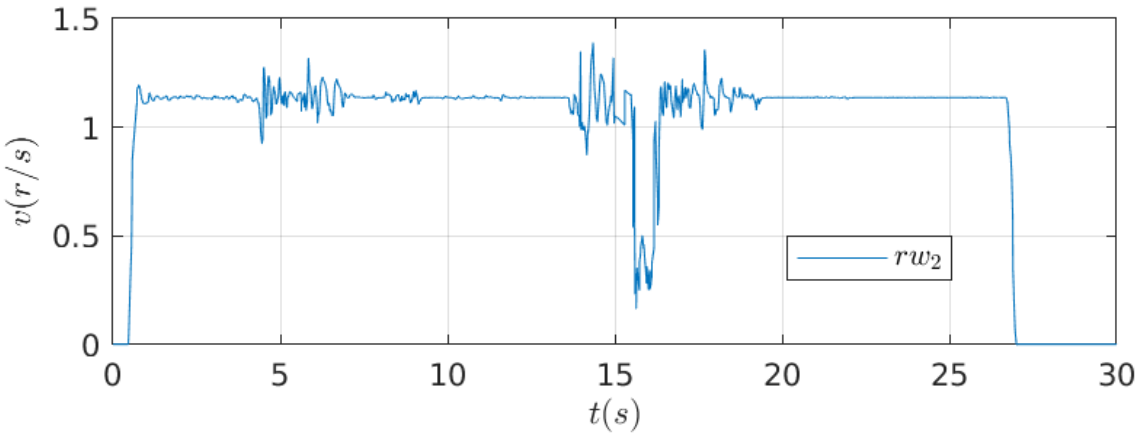


Figure A.5: Rear wheel (right) velocity on the rail task

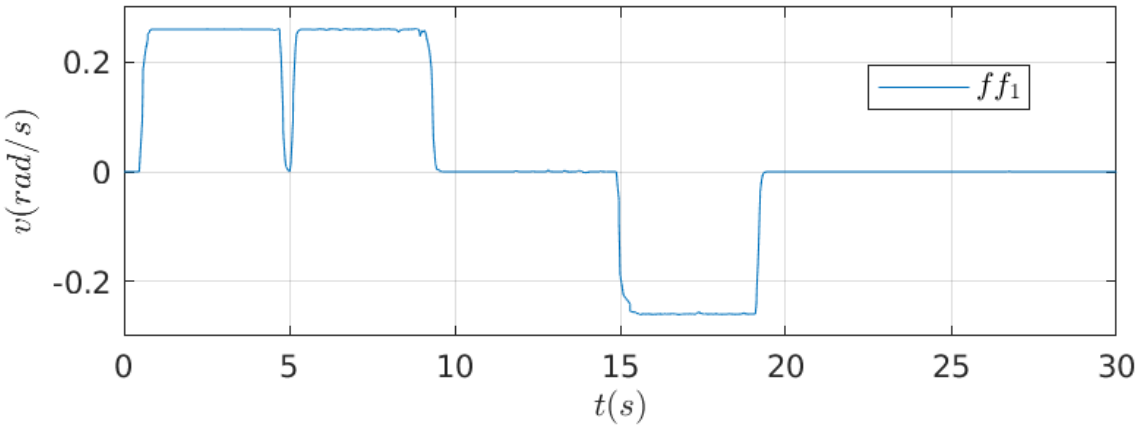


Figure A.6: Front flipper (left) velocity on the rail task

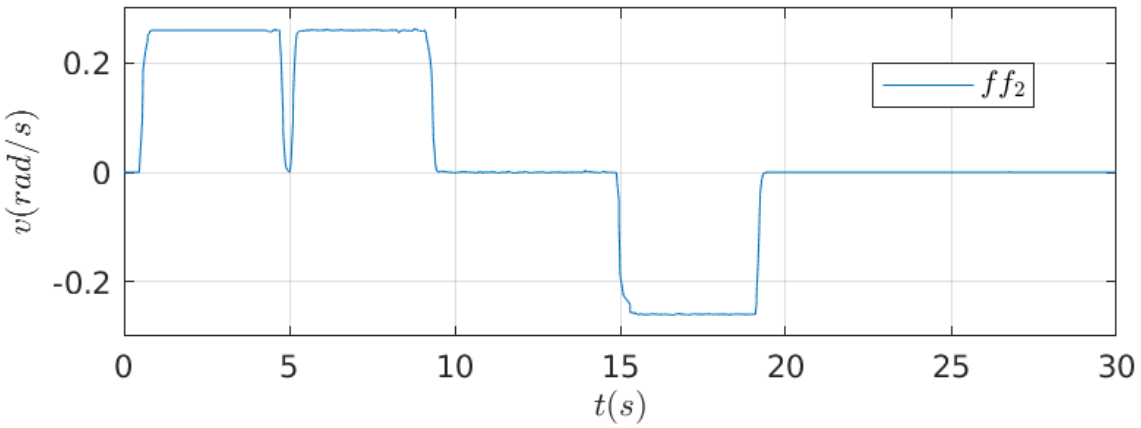


Figure A.7: Front flipper (right) velocity on the rail task

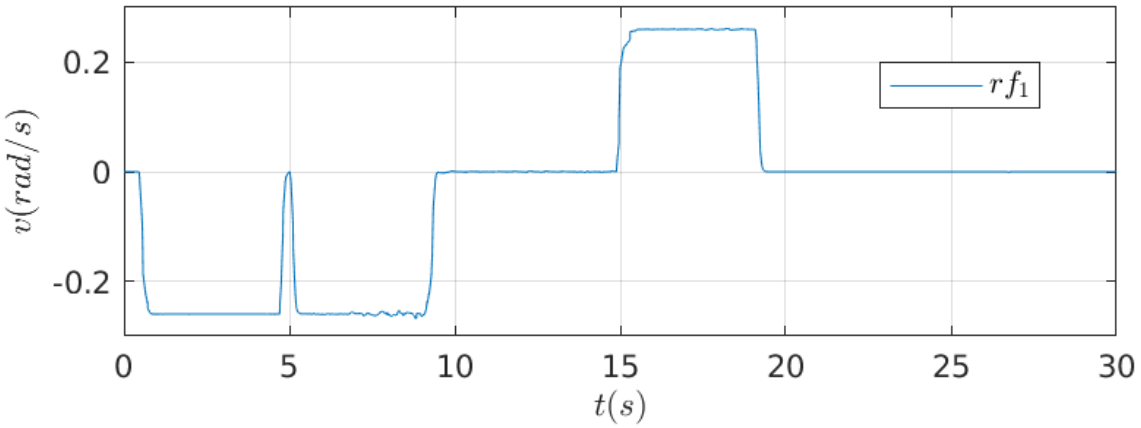


Figure A.8: Rear flipper (left) velocity on the rail task

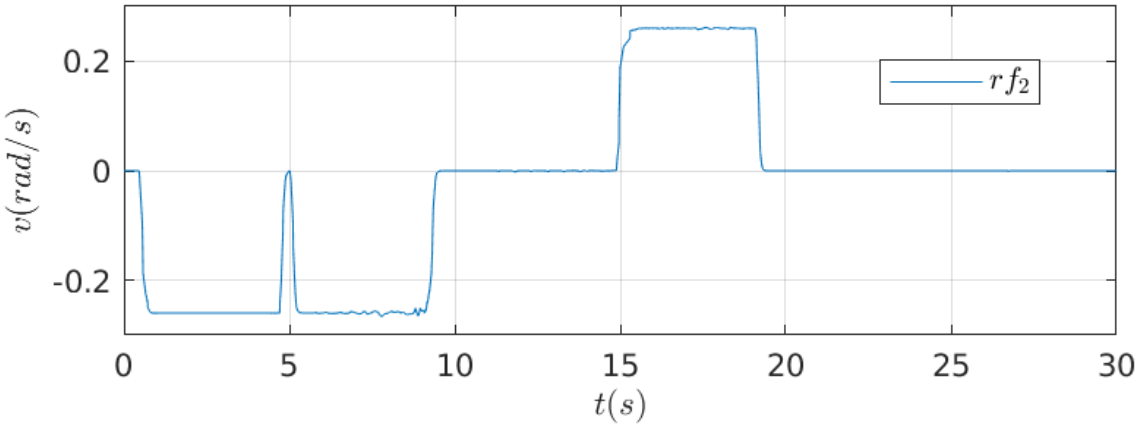


Figure A.9: Rear flipper (right) velocity on the rail task

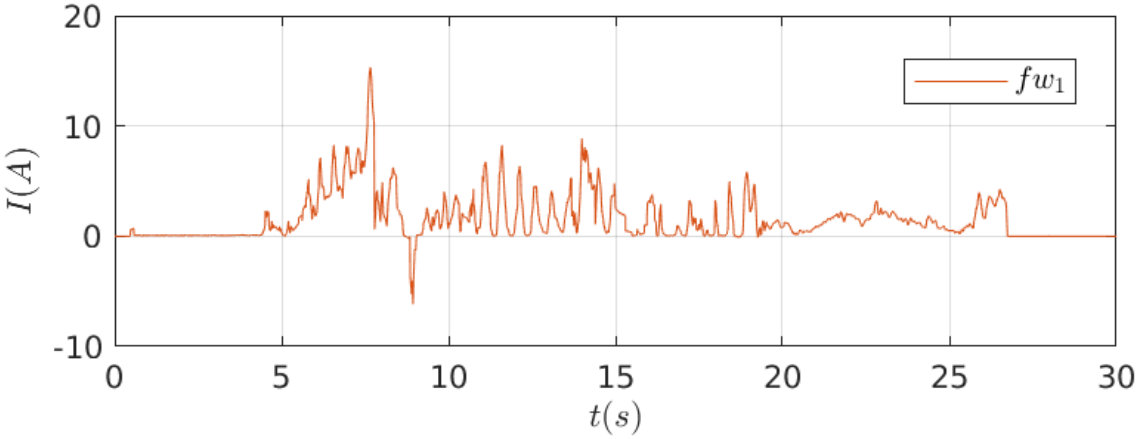


Figure A.10: Front wheel (left) current on the rail task

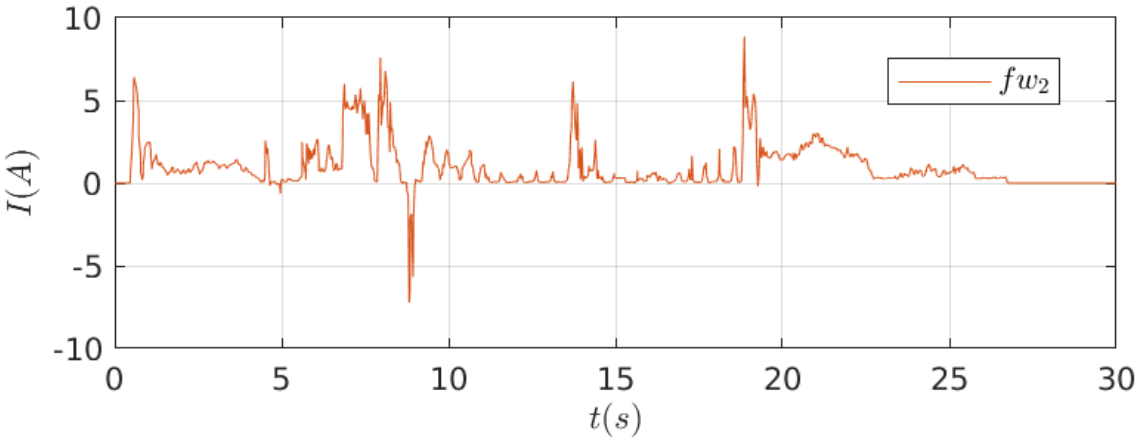


Figure A.11: Front wheel (right) current on the rail task

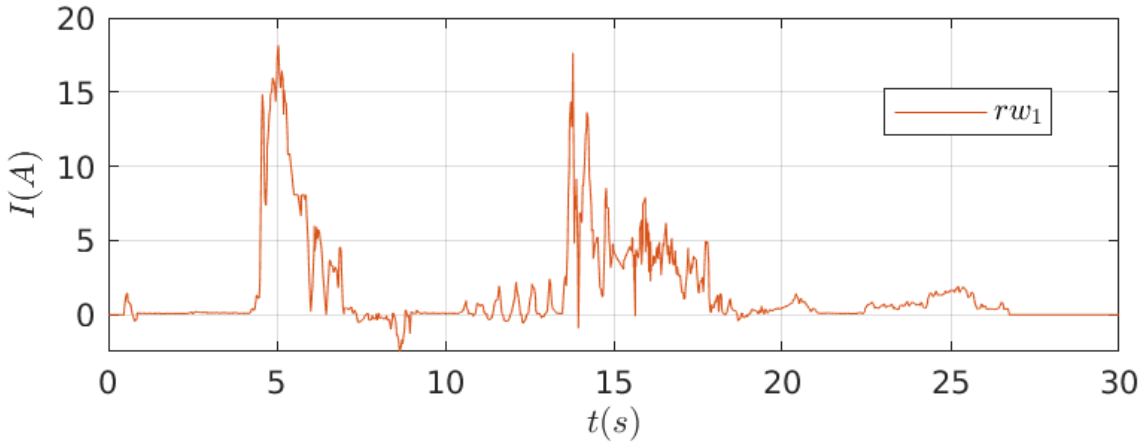


Figure A.12: Rear wheel (left) current on the rail task

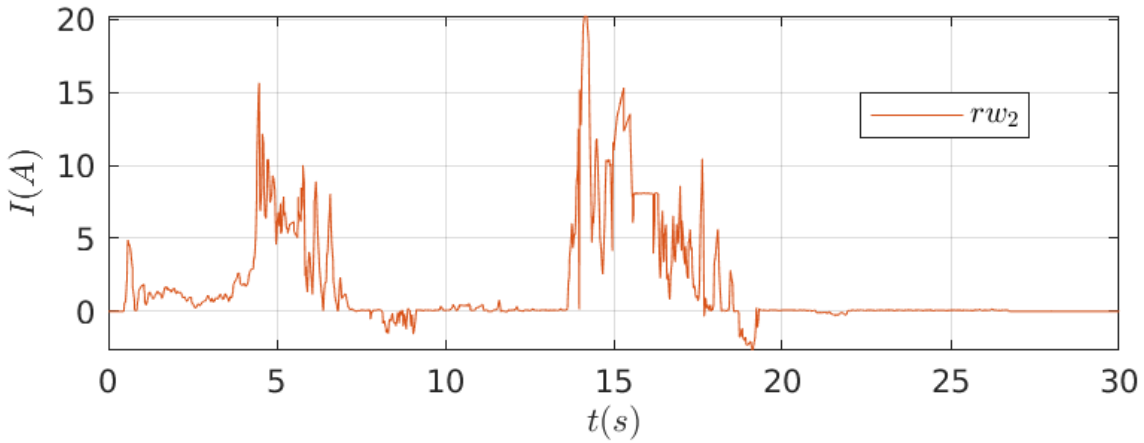


Figure A.13: Rear wheel (right) current on the rail task

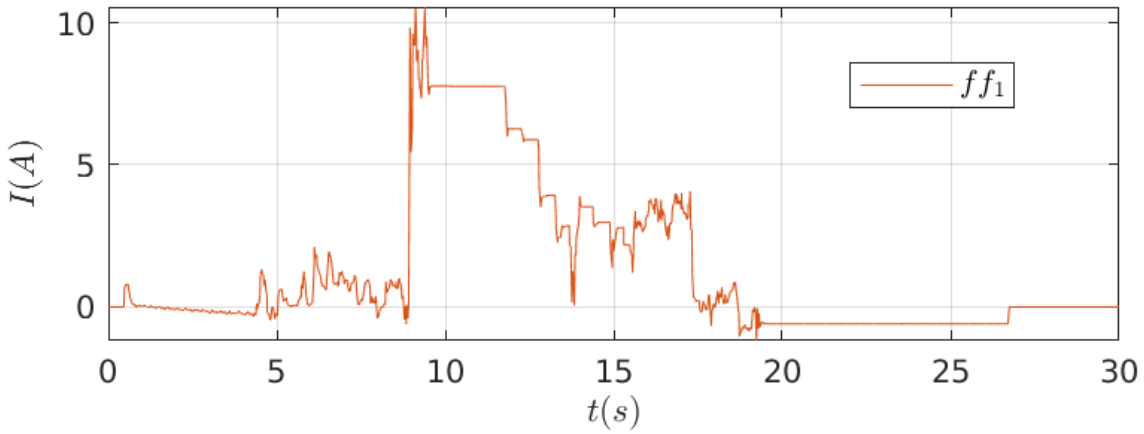


Figure A.14: Front flipper (left) current on the rail task

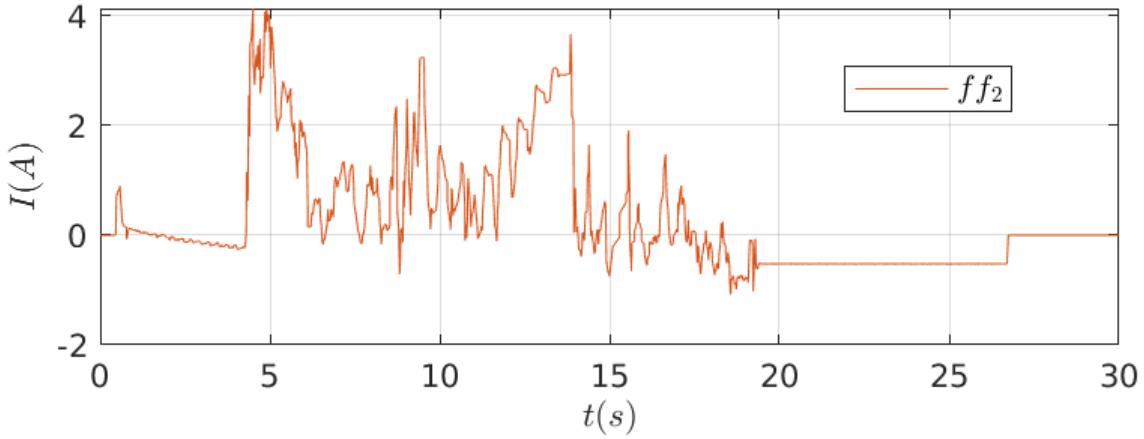


Figure A.15: Front flipper (right) current on the rail task

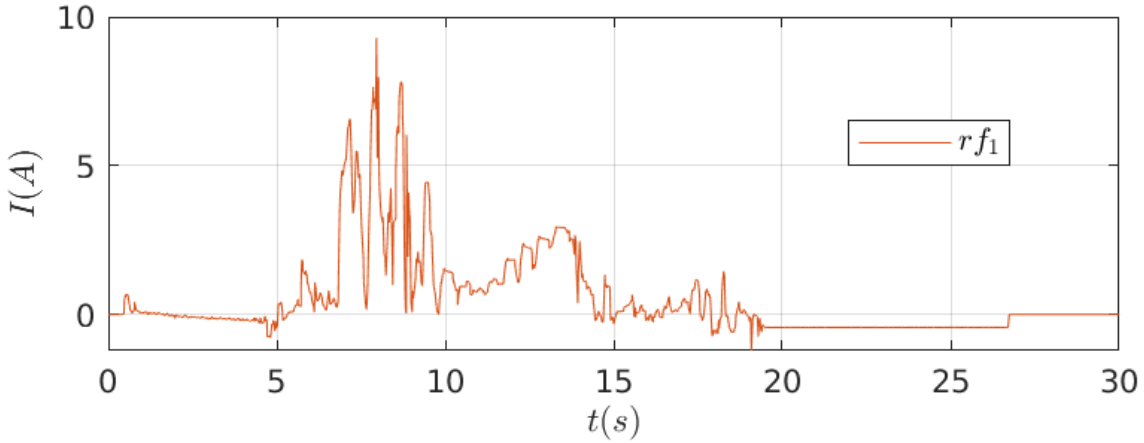


Figure A.16: Rear flipper (left) current on the rail task

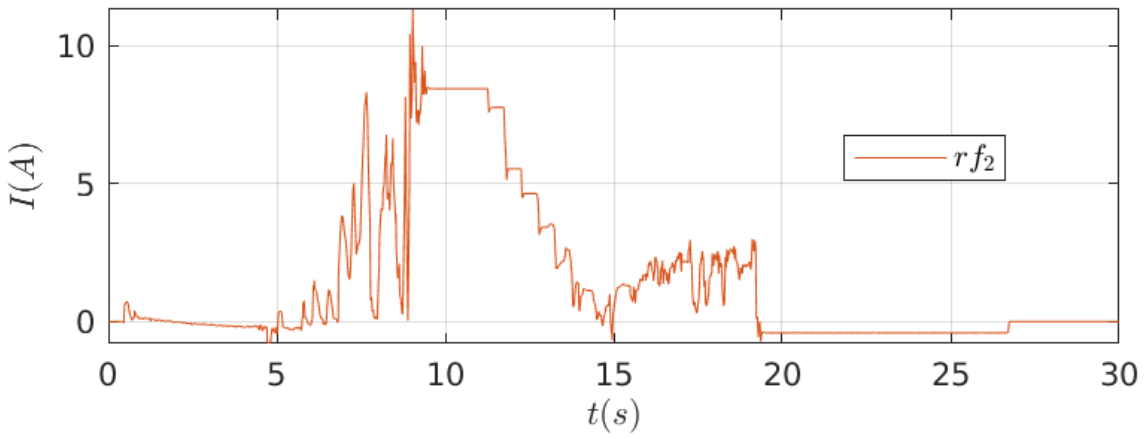


Figure A.17: Rear flipper (right) current on the rail task