applied sciences

MDPI

Article

Design, Modeling, and Control of an Autonomous Legged-Wheeled Hybrid Robotic Vehicle with Non-Rigid Joints

Vítor H. Pinto [1,2,*], Inês N. Soares [1], Marco Rocha [1], José Lima [2,3], José Gonçalves [2,3] and Paulo Costa [1,2]

[1] FEUP—Faculty of Engineering, University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal; up201604981@edu.fe.up.pt (I.N.S.); up201604163@edu.fe.up.pt (M.R.); paco@fe.up.pt (P.C.)

[2] CRIIS—Centre for Robotics in Industry and Intelligent Systems, INESC TEC—Institute for Systems and Computer Engineering, Technology and Science, Campus of FEUP, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal; jllima@ipb.pt (J.L.); goncalves@ipb.pt (J.G.)

[3] CeDRI—Research Centre in Digitalization and Intelligent Robotics, School of Technology and Management, Polytechnic Institute of Bragança, Santa Apolónia Campus, 5300-253 Bragança, Portugal

* Correspondence: vitorpinto@fe.up.pt

Abstract: This paper presents a legged-wheeled hybrid robotic vehicle that uses a combination of rigid and non-rigid joints, allowing it to be more impact-tolerant. The robot has four legs, each one with three degrees of freedom. Each leg has two non-rigid rotational joints with completely passive components for damping and accumulation of kinetic energy, one rigid rotational joint, and a driving wheel. Each leg uses three independent DC motors—one for each joint, as well as a fourth one for driving the wheel. The four legs have the same position configuration, except for the upper hip joint. The vehicle was designed to be modular, low-cost, and its parts to be interchangeable. Beyond this, the vehicle has multiple operation modes, including a low-power mode. Across this article, the design, modeling, and control stages are presented, as well as the communication strategy. A prototype platform was built to serve as a test bed, which is described throughout the article. The mechanical design and applied hardware for each leg have been improved, and these changes are described. The mechanical and hardware structure of the complete robot is also presented, as well as the software and communication approaches. Moreover, a realistic simulation is introduced, along with the obtained results.

Keywords: control design; state space; legged-wheeled; hybrid; robot; mechatronics

check for updates

Citation: Pinto, V.H.; Soares, I.N.; Rocha, M.; Lima, J.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of an Autonomous Legged-Wheeled Hybrid Robotic Vehicle with Non-Rigid Joints. *Appl. Sci.* **2021**, *11*, 6116. https://doi.org/10.3390/app11136116

Academic Editors: Augusto Ferrante, Mihaiela Iliescu and Mingcong Deng

Received: 14 June 2021
Accepted: 28 June 2021
Published: 30 June 2021

1. Introduction

Although the research in the field of robotics in multilegged vehicles is becoming increasingly common, particularly in the field of centaur-like quadrupeds, there is no clear solution that is unanimously better than others, largely depending on the vehicle's purpose. However, it is consensual that these legged vehicles consume more power than the traditional wheeled ones. Another difference between these vehicles is the fact that the legged robot is less stable compared to the traditional mobile vehicle, given the fact that it has several movable points along its members, which are floating. Despite having the mentioned disadvantages, which are difficult to deal with, walking vehicles have the ability to move on uneven and unstructured terrain that no traditional mobile vehicle can match, being one of the main reasons that has contributed to the growth of these vehicles' research over the years.

There are several projects concerning legged vehicles, such as BigDog [1], developed by Dr. Marc Raibert and his team; the Cheetah "family", namely, the Cheetah 3 [2], developed at MIT by Bledt et al.; the metamorphic robot [3], developed by Tang et al.; ANYmal [4], developed at ETH Zurich by Hutter et al.; SCalf-III [5], developed by Yang et al. at Shandong University; or ALPHRED [6], designed by Hooks et al. at California University. In addition to these, there is a research stream of hybrid vehicles with legs and wheels, called legged-wheeled (also known as wheel-legged), which combine both methods and try to take advantage of a vehicle with legs, but also improve and reduce its disadvantages.

Vehicles like the evolved ANYmal on Wheels [7], developed by Bjelonic et al.; Complios [8], developed by Bouton et al. at Université Pierre et Marie Curie; MOMARO [9], developed by Schwarz et al. at the University of Bonn; or CENTAURO [10], developed by Kashiri et al. at Istituto Italiano di Tecnologia (IIT), use the concept of hybrid legged-wheeled robots to navigate and perform tasks, for example, in disaster scenarios (like the DARPA challenge, where some of the mentioned robots were tested).

However, a common issue in all of these robotic platforms is the increased mechanical complexity, which raises the vehicle's price and replication cost. Improving the construction techniques, as well as using higher quality materials, leads to an increase in the cost of the vehicle. Typically, the usage of more robust materials has the disadvantage of increasing the robot's weight which, in turn, requires the use of higher quality hardware. Additionally, it is not uncommon that these platforms end up being a closed solution dedicated to a specific task.

In academia, it is normal to reduce costs in projects, considering that this community is, naturally, not profitable. Therefore, both from an educational and research point of view, the use of cost-oriented robots is somewhat common. Spiderino, developed by Jdeed et al. [11] at Alpen-Adria-Universität, is a low-cost, spiderlike robot, adapted to introduce an Arduino microcontroller, sensors, and motor drives, making it capable of performing swarm robotic experiments. For robotic manipulation, Gealy et al. [12] introduced a low-cost, quasi-direct drive approach to a robotic manipulator, achieving a platform capable of competing with off-the-shelf manipulators at a lower cost. Moreover, Armesto et al. [13] presented a benchmark with several 3D-printable cost-oriented robots for education, demonstrating that this trend is growing.

Besides the cost, concerns related to the objectives and utilization of the platforms built are considered when undertaking a project in this context. Normally, in a research group, building a vehicle of this kind takes into account that it will be used for more than one project, so its design must be as modular as possible. As mentioned by Brunete et al. [14], modular robots are widely studied for research purposes, being a current trend. An example of this trend is the Solo robot, developed by Grimminger et al. [15], which is a modular, cost-oriented quadruped built for research purposes. In modular robots, there may be a set of constraints that reduce the robot construction budget. Furthermore, if the robot is supposed to fulfill more than one objective, the best strategy is to build the platform to be as modular as possible, being adaptable to (almost) any situation.

Considering what was mentioned above, two main objectives for the construction of the robot presented in this article were the cost reduction—thus, designing a low-cost robot—as well as the maximization of its modularity. Another major objective is focused on energy saving, in order to increase the robot autonomy, with the inclusion of a low-power mode.

Development of realistic simulation models of robotic systems represents an extremely important stage for refinement of the robot, as precise operation can only be assured by finely tuning the variables related to control strategies, dynamics, and the robot mechanics themselves. Simulation tools are valuable for testing purposes due to allowing modeling of a variety of situations for assessment of the performance of the robot, while avoiding damaging the real robot. Whereas the presented platform is a quadruped with several coupled, non-rigid joints, the realistic simulation model significantly reduces the probability of the robot suffering unforeseen damage, as well as decreases, for example, the control algorithms' testing.

Available simulators with incorporated dynamics for robotics include Gazebo, Webots, and SimTwo [16]. Several publications report the use of simulations to test legged-wheeled robots. Kamedula et al. [17] employed a Gazebo-ROS (Robotic Operating System) framework to validate the wheeled-legged motion control scheme for CENTAURO. A 3D dynamic simulation was conducted by Suzumura et al. [18] to validate the developed methodologies, namely, for posture control of the robot. Chen et al. [19] evaluated their proposed control strategy for stable walking with cosimulation.

The present work is the continuation of the work published in prior articles [20,21]. Each leg uses almost the same physical configuration. However, there was an evolution to improve the robot's mechanical robustness, sensor data acquisition, data communication, leg synchronism, and robot kinematic control. Despite these, the robot maintains the same low-cost, low-power, and modular concerns and methods that were previously introduced. The complete robot is described, as well as the improved mechanical design and hardware. The software used to control the robot, as well as the communication approach, is also depicted.

The built prototype is controlled across the x, y, and z axes, and is capable of moving omnidirectionally. The robot can also be controlled across the roll and pitch axes, improving its locomotion on unsmooth terrain. Each leg is controlled individually; the non-rigid joints are controlled using state-space controllers, while the hip joint is controlled using a PID position controller with a trajectory smoothness function, and the wheel is controlled through a PID speed controller.

A realistic simulation model of the presented vehicle has been developed and tested in the SimTwo simulator software. This free robotics simulator provides an environment with incorporated dynamics and features the possibility to model different types of robots—from manipulators to mobile robots.

Throughout the article, the mechanical design and construction of the robot will be presented, and both the individual and complete control strategy for each leg and the robot are described. The applied communication technique will be detailed, along with the message flow between the robot and the legs. The process followed for obtaining the models for the motor and joint used, and the models themselves, will be described. The kinematic model will also be presented.

The paper is structured as follows: After the introduction in Section 1, which presents the literature review, Section 2 presents the design and mechanical model of the vehicle, the mathematical model, and its kinematics. Throughout Section 3, the obtained simulation results are presented, and the tests carried out are explained. Finally, Section 4 presents the conclusions and future work.

2. Design, Modeling, and Control

As mentioned previously, the robot design process considered the modularity of the system, as well as the lower cost, as two of the main focuses for this project. For validation purposes, the authors have decided to create a four legged vehicle as a test bed platform. The created platform is presented in Figure 1.

Throughout the article, the subscript letters r and j will be used to identify the variables referring to the robot and joint. The superscript indexes l and w represent the leg and wheel, respectively. l varies between 0 and 3 to identify the legs, and j between 0 and 2 to identify each joint. Specifically, regarding the positioning and orientation variables, the robot uses x_r, y_r, and z_r as linear spatial variables, and α_r, β_r, and γ_r (roll, pitch, and yaw, respectively) as the angular ones. Each leg has the linear spatial variables x^l and z^l, as well as the angular variable θ_0^l (since the leg orientation concerns joint 0). Regarding the remaining joints, each joint uses variables θ_j^l. For linear and angular speeds, the robot uses v_r, v_{n_r}, and ω_r, while each wheel speed is represented by v_3^l (since the wheel speed concerns joint 3). Figure 2 represents the robot's position and orientation variables.

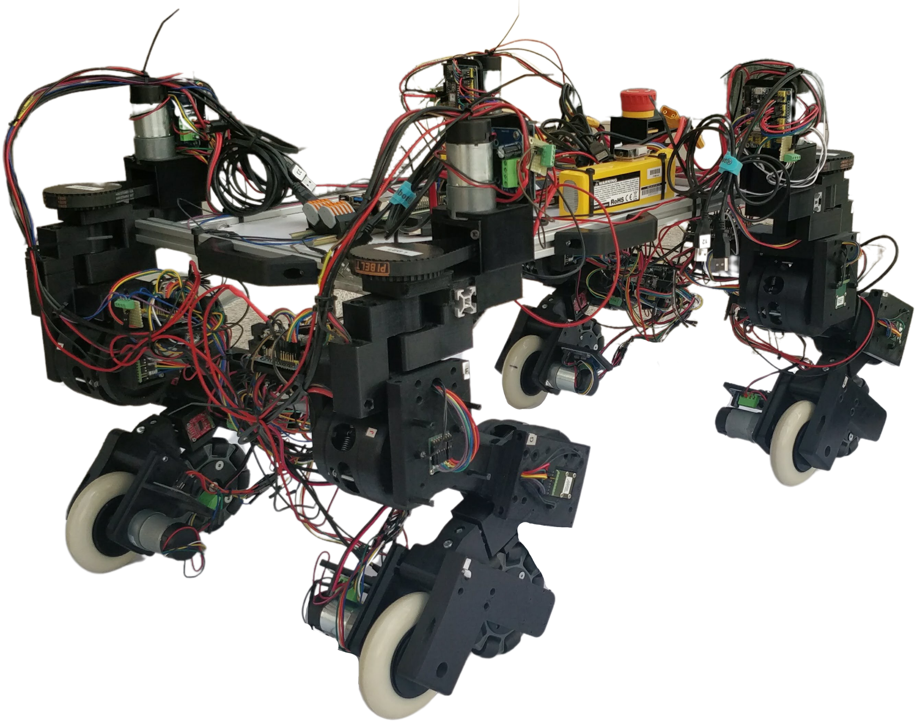


Figure 1. Hybrid legged-wheeled robotic vehicle.

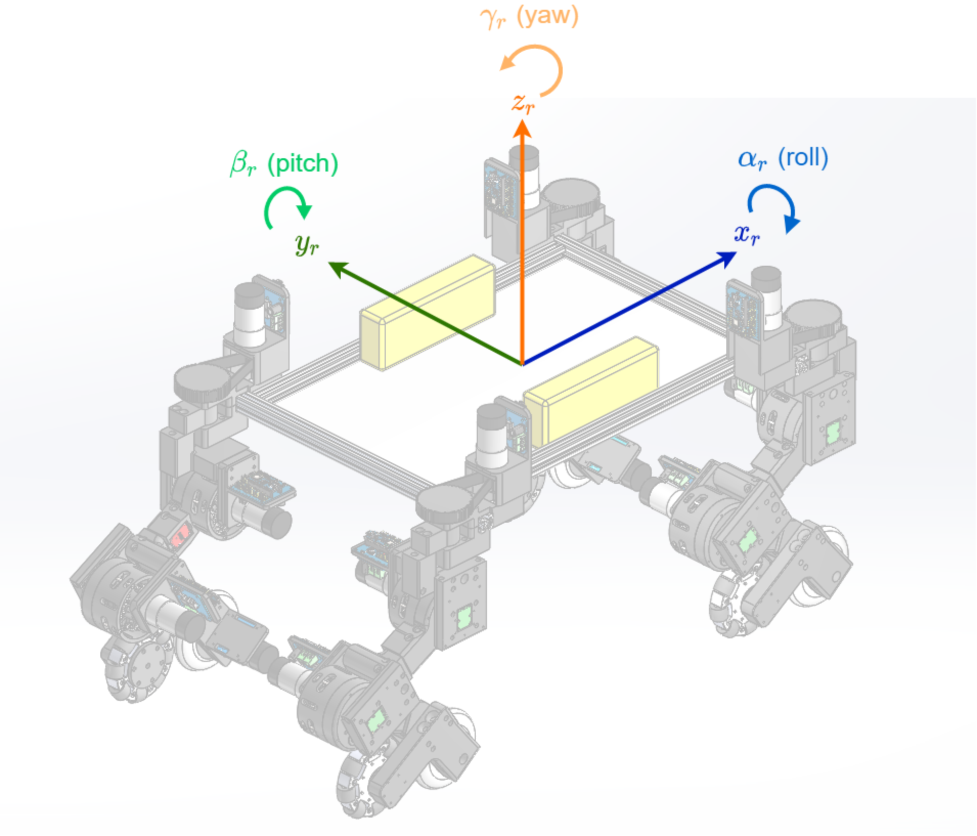


Figure 2. Robot position and orientation variables.

2.1. Mechanical Design

The vehicle has four legs, each one fully designed in Computer-Aided Design (CAD) and printed using 3D materials, for low-cost and fast development purposes. Throughout a previous research [21], a single leg for this vehicle was designed and modeled, and its control was tested in a practical context. Although the obtained results were quite satisfactory, the torsional moment required from the motors, especially on the rigid joint, was noteworthy.

This can be explained since the rotational movements are sometimes sudden, and the combination of rigid and non-rigid joints makes it difficult, for example, to brake the joints, which is mechanically more critical. A way to solve this problem would be to use a motor with gearboxes built with more resistant materials, which can transmit and support larger torques. However, this solution increases the cost of the vehicle, which is not compatible with the low-cost paradigm. The solution that avoids these increased mechanical efforts and complies with the construction paradigm is the replacement of the rigid joint with a non-rigid one. However, the previously presented non-rigid joint revealed that the required torque could not be achieved. This torque is used for positioning over the z_r-axis. So, a planetary gearbox was added at that particular position. This reduction gearset increases the rotational torque to the desired values. Moreover, being planetary, it also helps in reducing the joint backlash. This additional gear has a 3.2:1 ratio, which maintains the system modularity, solving the reported problem. Figure 3 presents the improved non-rigid joint configuration, while Figure 4 presents the improved leg configuration.

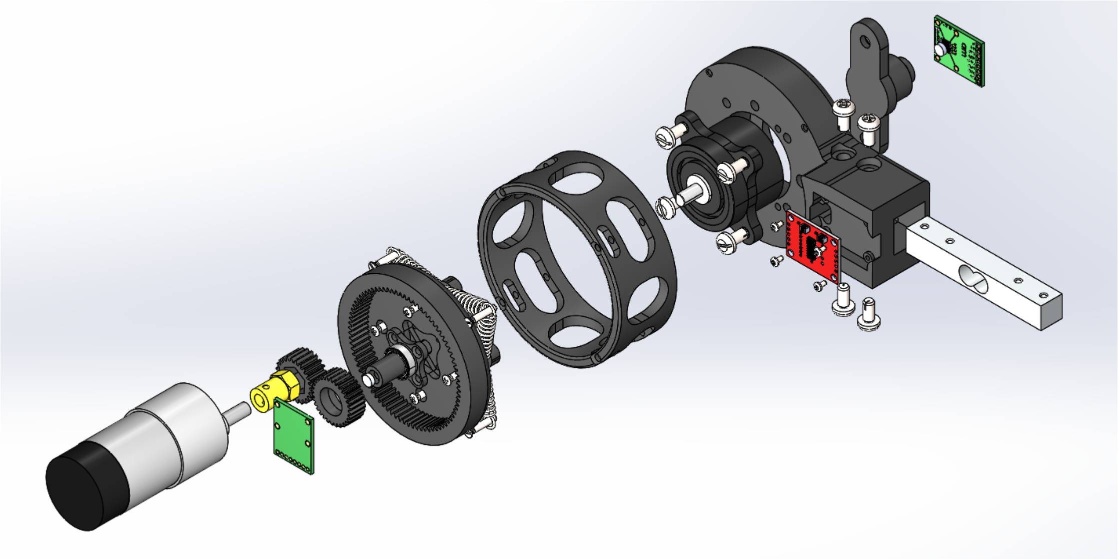


Figure 3. Improved non-rigid joint—explosion render.

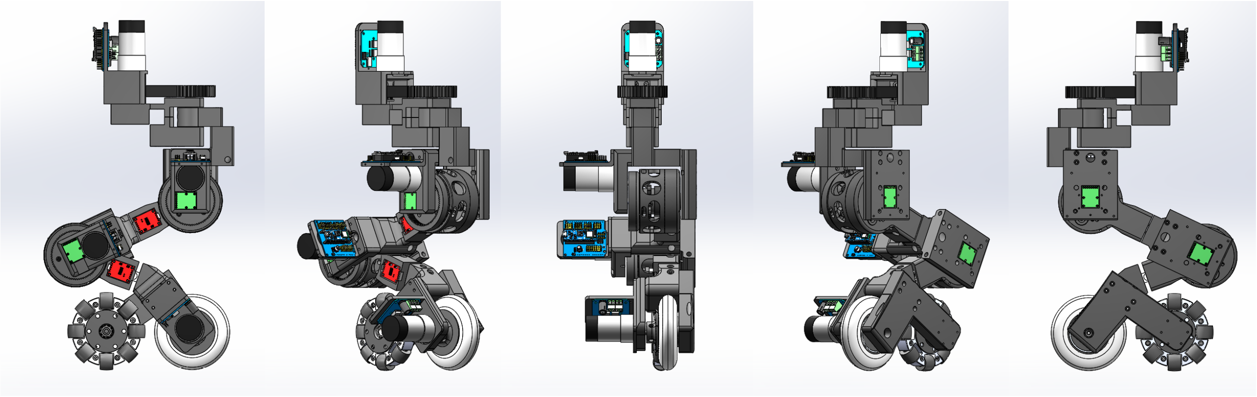


Figure 4. Improved leg configuration (front left leg).

Each leg has a defined joint configuration, as shown in Figure 5. Their numbering goes from joint 0 to joint 2, from top to bottom, respectively. The linear and rotational axes are also represented.

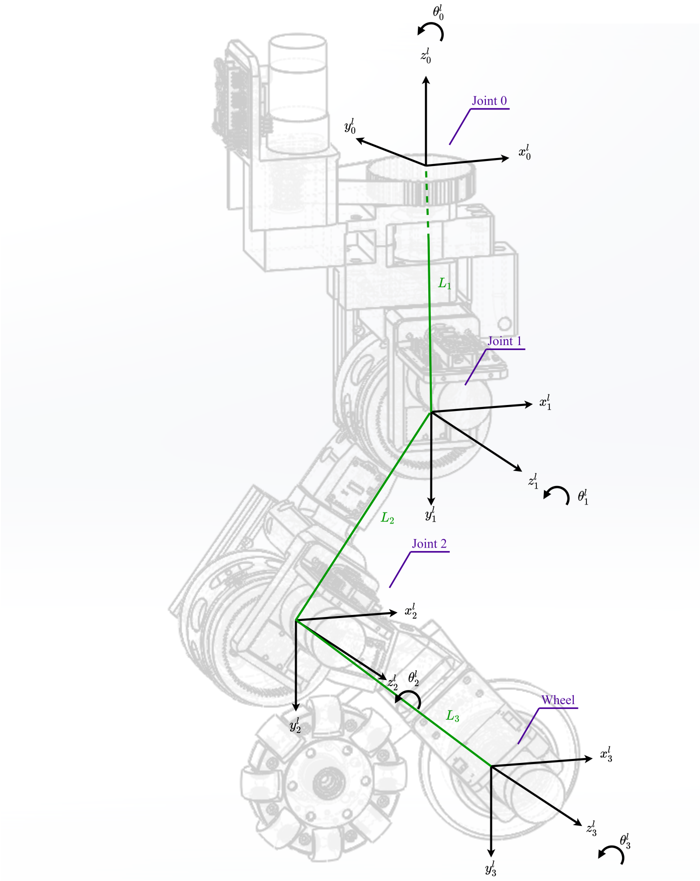


Figure 5. Leg with axes, links, and joints numbering.

The hip joint also suffered from torsional moment, and could be damaged for this reason, since it is the only exactly vertical axis, and therefore more prone to impacts. Furthermore, this is also the joint responsible for steering the vehicle when it is moving, so it is extremely important that its positioning is as accurate as possible. Considering these, to improve the mechanical robustness of this joint, as well as to decrease the gearbox backlash, a reduction gear was added, with a 4:1 ratio, connected through a geared belt, that improves both specifications. The obtained rigid hip joint is the one present in Figure 6.

The platform that was built to serve as a test bed is presented in Figure 7, which show the robot's rendering in CAD as well as its real version, respectively.

As can be seen in the diagram of Figure 8a, the robot's maximum dimensions are 850 × 465 mm (L × W). Figure 8b shows that its height can vary between 320 and 505 mm. In its current configuration, the robot weighs approximately 18 kg. In Table 1, the data from the vehicle's dimensions are aggregated.

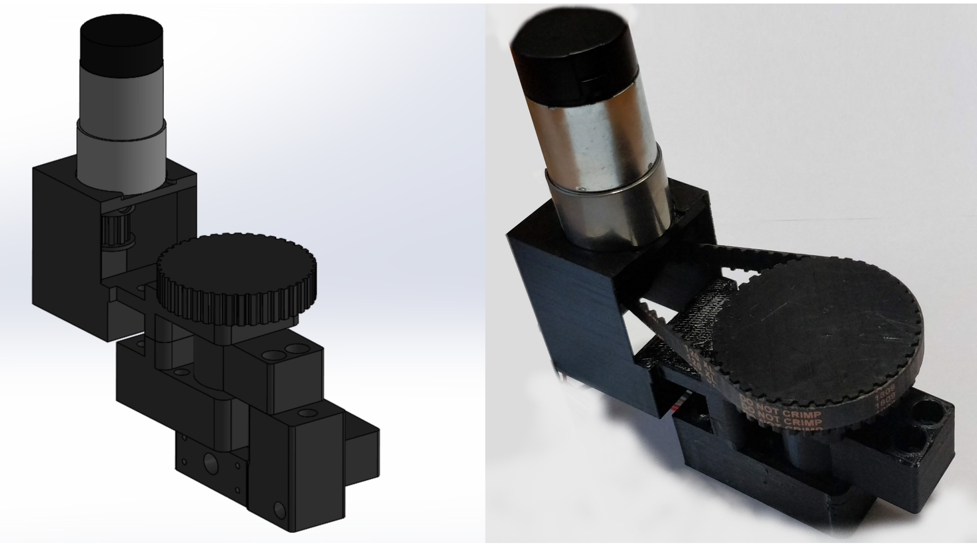
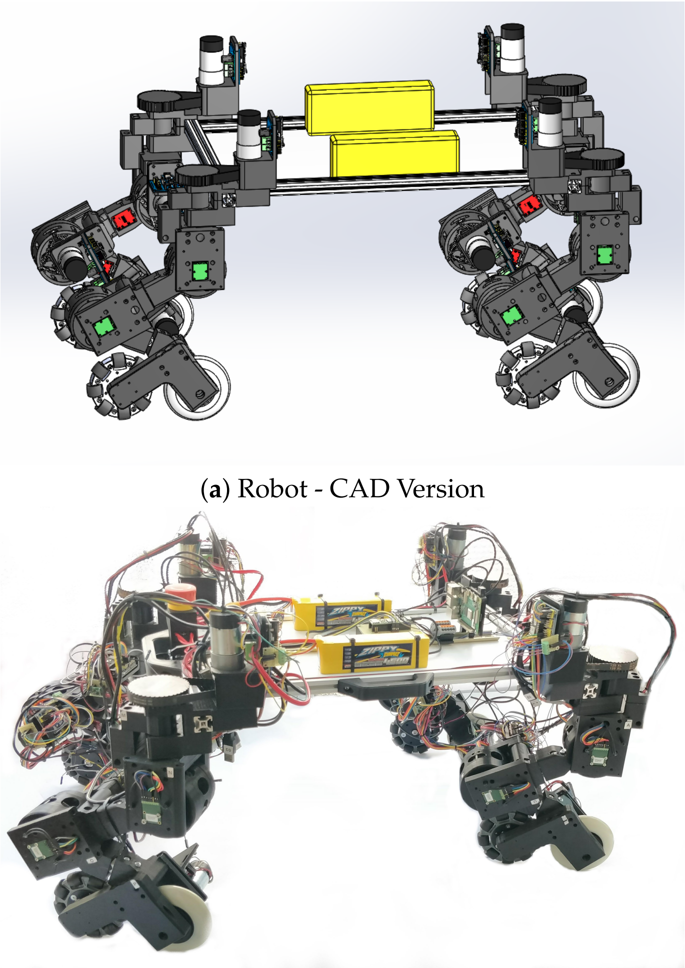


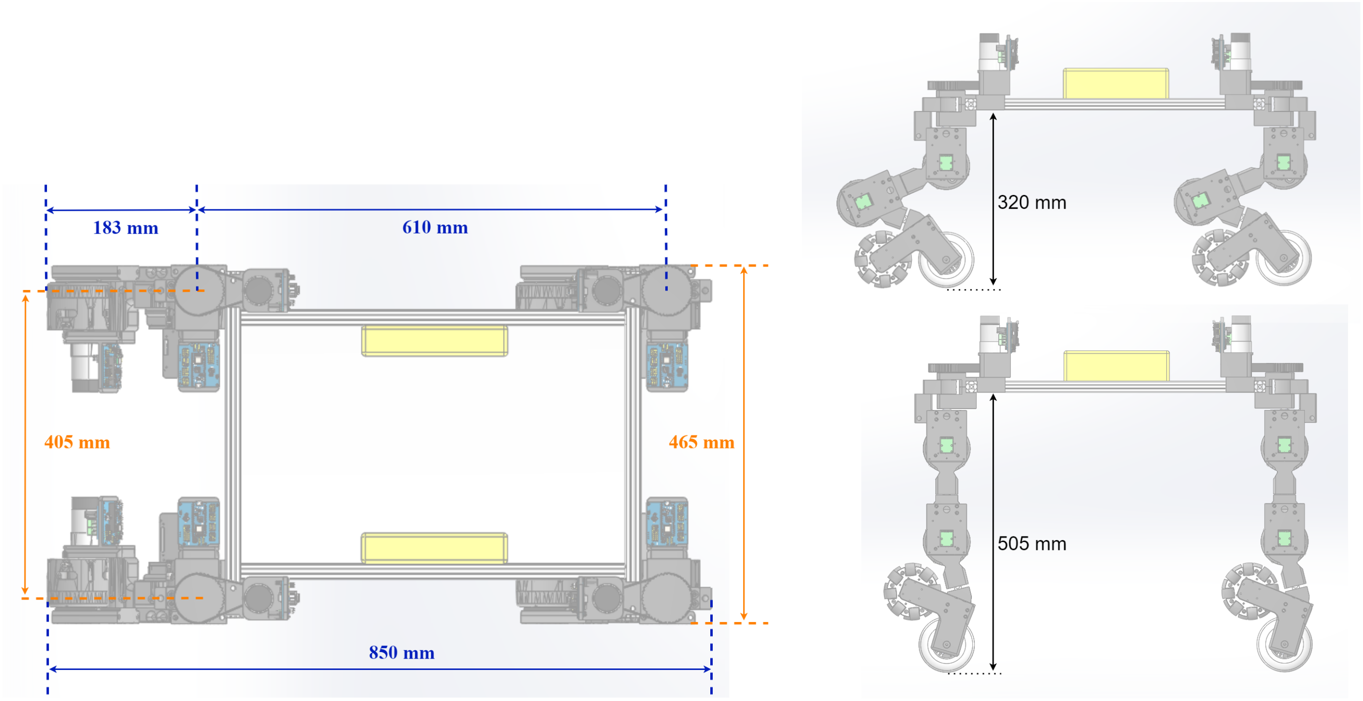
Figure 6. Improved hip rigid joint (**left**—render, **right**—real).



(**a**) Robot - CAD Version

(**b**) Robot - Real Version

Figure 7. Rendered and real robot.



(**a**) Length × Width (**b**) Height

Figure 8. Robot dimensions.

Table 1. Robot dimensions.

Dimensions	
Length	0.850 m
Width	0.465 m
Minimum Height	0.320 m
Maximum Height	0.505 m
Weight	18 kg

2.2. Electronics Design

Regarding the communication system between all subsystems (robot and joint microcontrollers), it must be robust, fast, and consistent. Previously [21], it was necessary to implement two serial COM (communication port) on the Arduino to communicate between the main microcontroller and the ones on the joints. However, the complexity of the message flow between the joints and the microcontroller increased drastically. Therefore, the system's expandability was very limited. For example, it would be difficult to increase the number of sensors/actuators for each joint, which would impact the vehicle modularity. Another added difficulty was that the joints had no knowledge of their position in the leg.

Considering the described issues, several changes were performed. The first one was the introduction of a third microcontroller associated with the hip joint, since it was necessary to divide all the peripherals being used.

Another one was to replace the Arduino main microcontroller with a Teensy 4.1 board. This board has increased processing power, a greater number of available serial ports, as well as I/O pins.

Another adopted strategy was to use the flash memory of the NINA W102 ESP32 module that the Arduino Nano 33 IOT has built-in to save each joint position on the robot. For example, joint 1 of the leg 0 was numbered *joint10*, making it possible to identify the low-, middle-, or top-level joints. This flash memory usage serves also to implement another change, which is saving the parameters associated with each joint, namely, absolute encoder offsets and controller tuning parameters, among others.

Another important improvement was the addition of an absolute encoder on the angle before the joint, i.e., directly to the motor shaft (defined as *Absolute Encoder In*, from here onwards). Using this information, it is possible to calculate the springs extension, and avoid an extension greater than the allowed one, which would cause damage to the springs. To obtain this measurement, AS5048B encoders were used, which communicate through the I2C (Inter-Integrated Circuit) protocol.

As another safety measure, Reed effect end switches were included in each non-rigid joint, preventing them from hitting the vehicle itself. The sensors used for this function were the LittelFuse 59025, and the respective 57025 magnets.

Another consideration regarding the electronic design was that the previously used motor driver was capable of sending high currents to the motors, which caused some accidental motor damaging, when some problems on the joint's references occurred. Considering this, the motor drives of the non-rigid joints were also changed, now using the Adafruit 3190 (DRV8871).

This set of changes made it possible to transfer all the required calculations to the main processor, such as calculating direct and inverse kinematics, angular and speed references for each joint, as well as their smoother trajectories of each of the joints. Additionally, it is also in this microcontroller that the robot's kinematics is calculated for its trajectory tracking mode.

With this configuration, the joint microcontrollers are only responsible for receiving the position and speed references, calculating them through the respective controllers, and defining the PWM references of the respective motors.

Figure 9 illustrates the mentioned hardware, as well as their positioning in the leg and, specifically, in the joint.

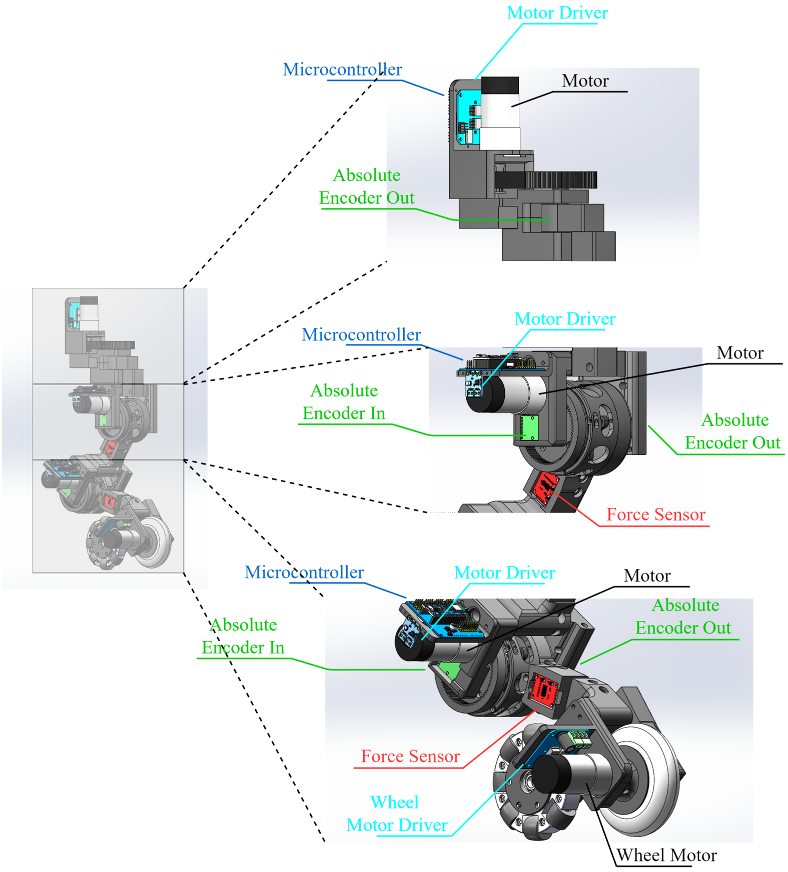


Figure 9. Single leg hardware diagram.

Considering that the low-cost is a concern, and that all the hardware used for its construction has already been presented, Table 2 contains a detailed version of all the materials, as well as the individual and total price range.

Table 2. Components and price range for the prototype construction.

Quantity	Reference	Description	Price Range (Unit.)
1	TEENSY41	Teensy 4.1	35–40 €
12	ABX00032	Arduino Nano 33 IOT	18–22 €
12	ARD03162	Arduino Nano I/O Expansion Board	5–8 €
8	CQRGL20180525-52GB	12 V DC Motor with Encoder and 210:1 Gearbox	28–32 €
4	CQR37D64CPR70GR	12 V DC Motor with Encoder and 70:1 Gearbox	28–32 €
4	CQR37D64CPR43.8GR	12 V DC Motor with Encoder and 43.8:1 Gearbox	28–32 €
8	BTS7960	High Power Motor Driver Module	16–20 €
8	DRV8871	Adafruit DC Motor Driver Breakout Board	6–10 €
8	TAL 220	10 kg Load Cell	8–12 €
8	HX711	Load Cell Amplifier	10–14 €
12	AS5048A	High-Resolution Rotary Position Sensor	6–10 €
8	AS5048B	High-Resolution Rotary Position Sensor	6–10 €
16	59025 + 57025	Reed Switch + Magnet	4–6 €
64	S-1148	Traction Spring	1–3 €
8	FRT-K2-103	Continuous Rotation Damper	53–57 €
4	RB-Nex-49	100 mm Plastic Omni Wheel	10–14 €
4	905225	100 mm/83 A Powerslide Swell Rubber Wheel	5–9 €
n/a	n/a	PLA Printing Material	32–40 €
n/a	n/a	Jumper Wire Cables	16–20 €
n/a	n/a	Screws	40–60 €
		TOTAL	1899–2516 €

2.3. Communication

Regarding the communication, it was decided to use serial protocol. The adopted communication strategy for the vehicle was chosen based on its reliability, simplicity, and robustness. Considering this, the serial communication channels were used.

The Teensy 4.1 board uses the USB port to communicate with the main board, bidirectionally, through the defined message structure. To communicate with the microcontrollers placed on the joints, the board uses 4 dedicated serial ports, each one predefined to a leg, which allows the software to always know to/from which leg it is transmitting/receiving. In each leg, a serial bus is implemented, through a board that allows for hardware connections to be easily performed, as well as to protect the general purpose input output (GPIO). Figure 10 presents the diagram that demonstrates the implemented connections, representing the example of only one leg.

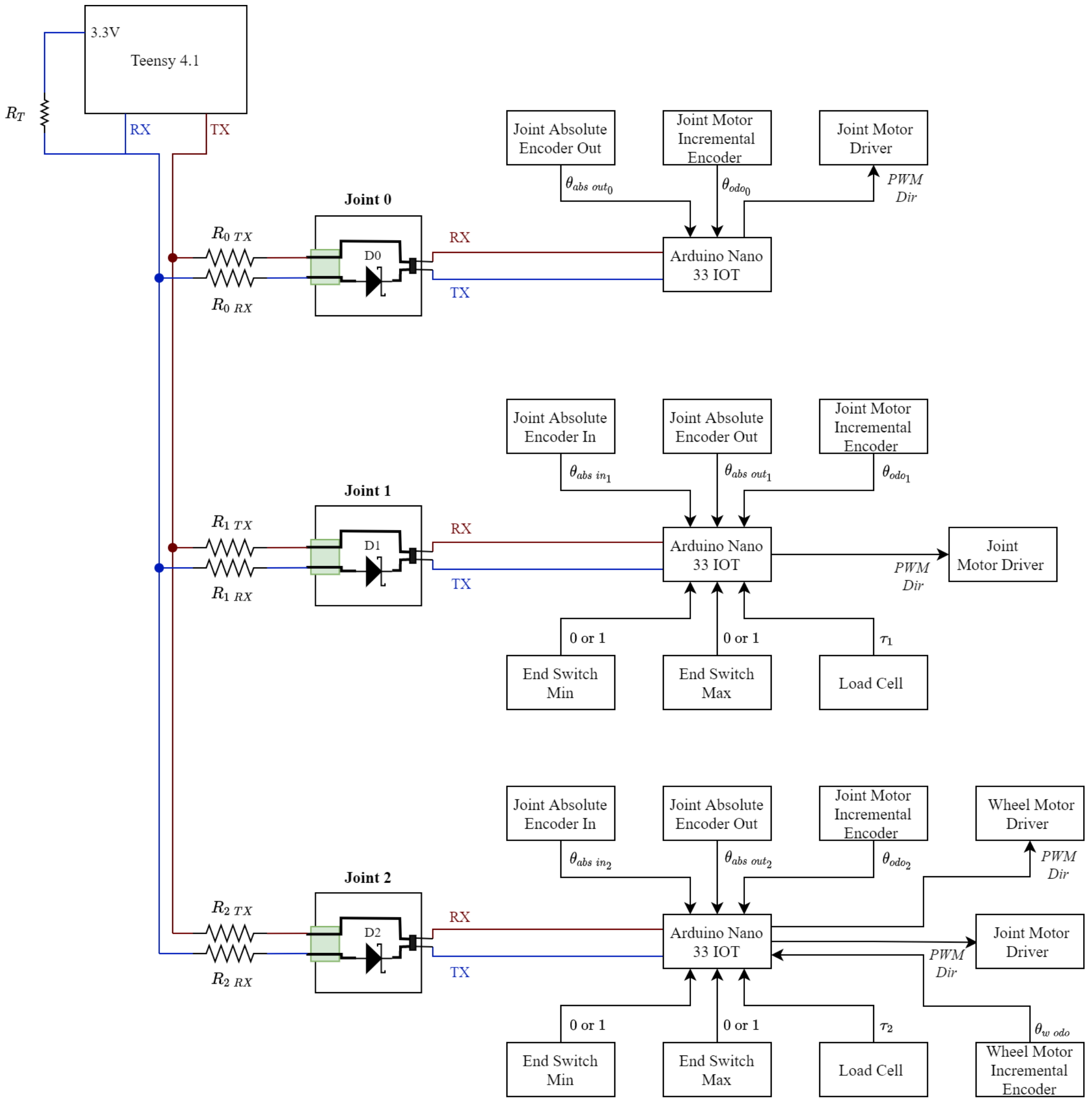
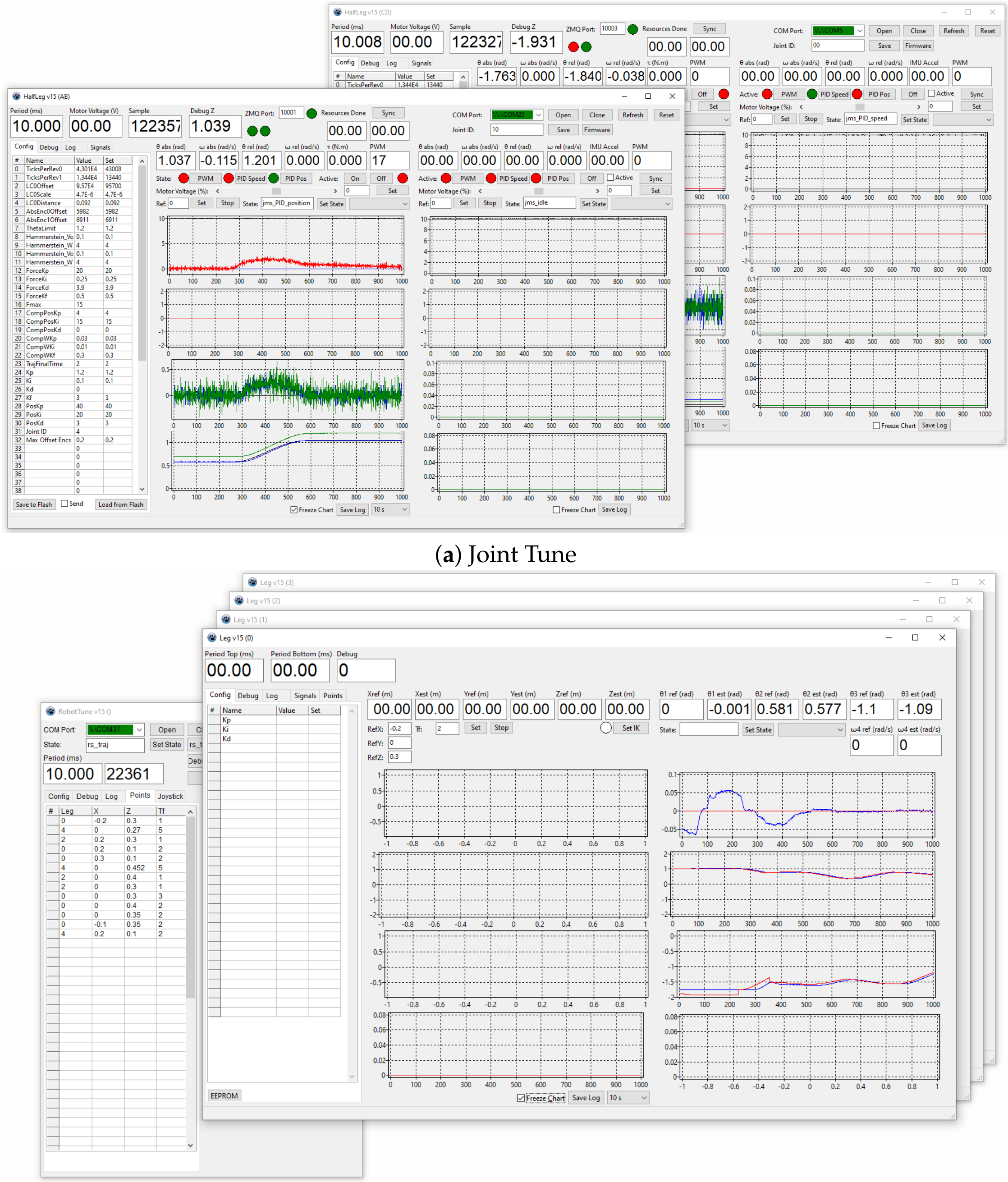


Figure 10. Single leg communication diagram.

The USB port can also be used simultaneously with the serial communication through the GPIO, by connecting it directly to the main microcontroller, through the software Joint Tune (presented in Figure 11a), created specifically for this purpose. One of the reasons for this software creation was to test each of the joints individually to obtain, for example, local parameters or graphics from each of the joints. Another reason was to easily and quickly debug one of the hardware components, the DC motor for example, enabling local tests without the need to actuate the entire system. This software was very important to collect the required data to obtain the joint and motor models that are presented. Another important task enabled by this software is the controller implementation and testing.



HalfLeg v15 (CD)

Period (ms) 10.008 | Motor Voltage (V) 00.00 | Sample 122327 | Debug Z -1.931 | ZMQ Port: 10003 | COM Port: \\.\COM5 | Joint ID: 00

θ abs (rad) -1.763 | ω abs (rad/s) 0.000 | θ rel (rad) -1.840 | ω rel (rad/s) -0.038 | τ (N.m) 0.000 | PWM 0

HalfLeg v15 (AB)

Period (ms) 10.000 | Motor Voltage (V) 00.00 | Sample 122357 | Debug Z 1.039 | ZMQ Port: 10001 | COM Port: \\.\COM20 | Joint ID: 10

θ abs (rad) 1.037 | ω abs (rad/s) -0.115 | θ rel (rad) 1.201 | ω rel (rad/s) 0.000 | τ (N.m) 0.000 | PWM 17

State: jms_PID_position | Ref: 0 | State: jms_idle

#	Name	Value	Set
0	TicksPerRev0	4.301E4	43008
1	TicksPerRev1	1.344E4	13440
2	LC0Offset	9.57E4	95700
3	LC0Scale	4.7E-6	4.7E-6
4	LC0Distance	0.092	0.092
5	AbsEnc0Offset	5982	5982
6	AbsEnc1Offset	6911	6911
7	ThetaLimit	1.2	1.2
8	Hammerstein_Vo	0.1	0.1
9	Hammerstein_W	4	4
10	Hammerstein_Vo	0.1	0.1
11	Hammerstein_W	4	4
12	ForceKp	20	20
13	ForceKi	0.25	0.25
14	ForceKd	3.9	3.9
15	ForceKf	0.5	0.5
16	Fmax	15	
17	CompPosKp	4	4
18	CompPosKi	15	15
19	CompPosKd	0	0
20	CompWKp	0.03	0.03
21	CompWKi	0.01	0.01
22	CompWKf	0.3	0.3
23	TrajFinalTime	2	2
24	Kp	1.2	1.2
25	Ki	0.1	0.1
26	Kd	0	
27	Kf	3	3
28	PosKp	40	40
29	PosKi	20	20
30	PosKd	3	3
31	Joint ID	4	
32	Max Offset Encs	0.2	0.2
33		0	
34		0	
35		0	
36		0	
37		0	
38		0	

(**a**) Joint Tune

RobotTune v15 ()

COM Port: \\.\COM37 | State: rs_traj | Period (ms) 10.000 22361

#	Leg	X	Z	Tf
	0	-0.2	0.3	1
	4	0	0.27	5
	2	0.2	0.3	1
	0	0.2	0.1	2
	0	0.3	0.1	2
	4	0	0.452	5
	2	0	0.4	1
	2	0	0.3	1
	0	0	0.3	3
	0	0	0.4	2
	0	0	0.35	2
	0	-0.1	0.35	2
	4	0.2	0.1	2

Leg v15 (0)

Period Top (ms) 00.00 | Period Bottom (ms) 00.00 | Debug 0

Xref (m) 00.00 | Xest (m) 00.00 | Yref (m) 00.00 | Yest (m) 00.00 | Zref (m) 00.00 | Zest (m) 00.00 | θ1 ref (rad) 0 | θ1 est (rad) -0.001 | θ2 ref (rad) 0.581 | θ2 est (rad) 0.577 | θ3 ref (rad) -1.1 | θ3 est (rad) -1.09

RefX: -0.2 | Tf: 2 | RefY: 0 | RefZ: 0.3 | ω4 ref (rad/s) 0 | ω4 est (rad/s) 0

#	Name	Value	Set
	Kp		
	Ki		
	Kd		

(**b**) Robot Tune

Figure 11. Joint Tune and Robot Tune.

The Arduino joint firmware can communicate in two independent ways—USB and SERIAL—and, as far as possible, maintain the working compatibility of both modes. This enables the use of the Robot Tune application (presented in Figure 11b) to control the robot, which communicates through USB with the main computer and, at the same time, controls each joint individually through the serial port. This duality allows, for example, the tuning of parameters in real time, as well as their recording to flash memory, obtaining data directly from the Arduino, and debugging serial communications, among others.

The communication dynamics between the multiple applications and processing boards is shown in Figure 12.

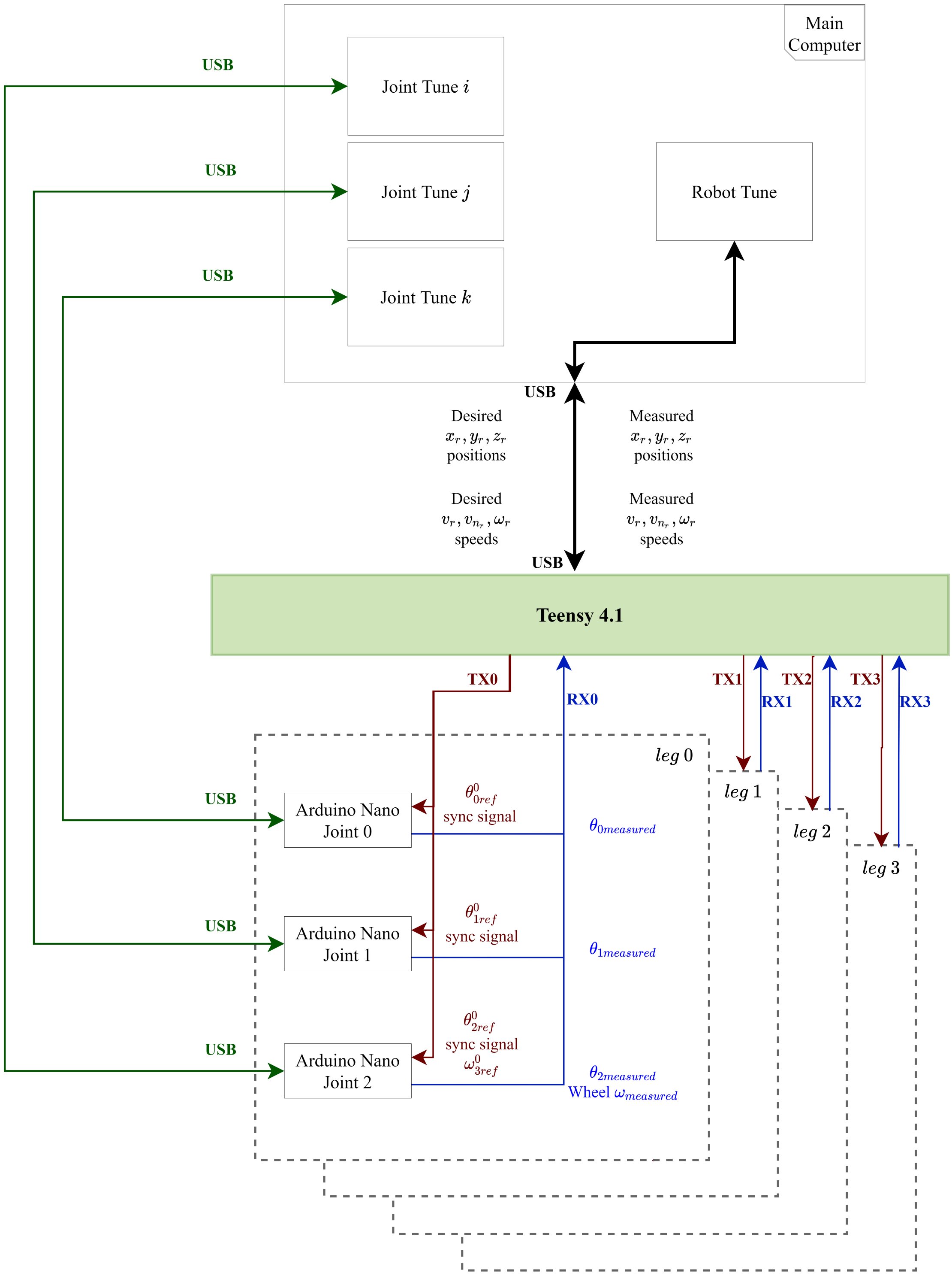

Main Computer
USB
Joint Tune i
USB
Joint Tune j
Robot Tune
USB
Joint Tune k
USB
Desired x_r, y_r, z_r positions
Measured x_r, y_r, z_r positions
Desired v_r, v_{n_r}, ω_r speeds
Measured v_r, v_{n_r}, ω_r speeds
USB
Teensy 4.1
TX0
RX0
TX1
RX1
TX2
RX2
TX3
RX3
leg 0
leg 1
leg 2
leg 3
USB
Arduino Nano Joint 0
θ^0_{0ref} sync signal
$\theta_{0measured}$
USB
Arduino Nano Joint 1
θ^0_{1ref} sync signal
$\theta_{1measured}$
USB
Arduino Nano Joint 2
θ^0_{2ref} sync signal ω^0_{3ref}
$\theta_{2measured}$ Wheel $\omega_{measured}$

Figure 12. Communication software and hardware macro diagram.

An important feature is the possibility of remote OTA (Over-the-air) firmware upload to the Arduino boards. Since these are typically not connected through USB, there was the need to connect each and every board to the PC to upload the joint firmware to each Arduino. To avoid this issue, a remote OTA upload feature is embedded in the joint firmware, which enables the remote request to the server, triggered by a message sent by the main computer.

2.4. Proposed Models

Throughout this section, the models referring to the motor used will be presented, as well as the non-rigid joint. These models are extremely important, as they are later used in the development of the realistic simulation model of the robot.

2.4.1. Motor Model

As in the past [22], the generic model for a DC motor is used, since the motor-gearbox system is equivalent. Equation (1), rearranged to obtain Equation (2), are the main equations used to model the motor. In both equations, u expresses the voltage (V) applied to the motor, R is the motor resistance (Ω), i stands for the current flow (A), k for the motor constant (N·m/$\sqrt{\text{W}}$), and ω the angular speed (rad/s). The static friction torque T_q (N·m) of the motor is obtained through the multiplication $k \cdot i$. As already mentioned in [21], it is noteworthy that this motor model assumes that the static torque friction T_q = T_m while $T_m < T_{q\_max}$.

$$u = R \cdot i + k\omega \tag{1}$$

$$i = \frac{u - k\omega}{R}. \tag{2}$$

The performed hardware tests to estimate the mentioned parameters were detailed in [22]. Figures 13, 14, and 15 present the collected data, which allow the estimation of the motor model parameters. The performed tests were carried out with the planetary reduction gear coupled to the motor shaft. This way, the gear effects are incorporated in the motor model, instead of the joint model.

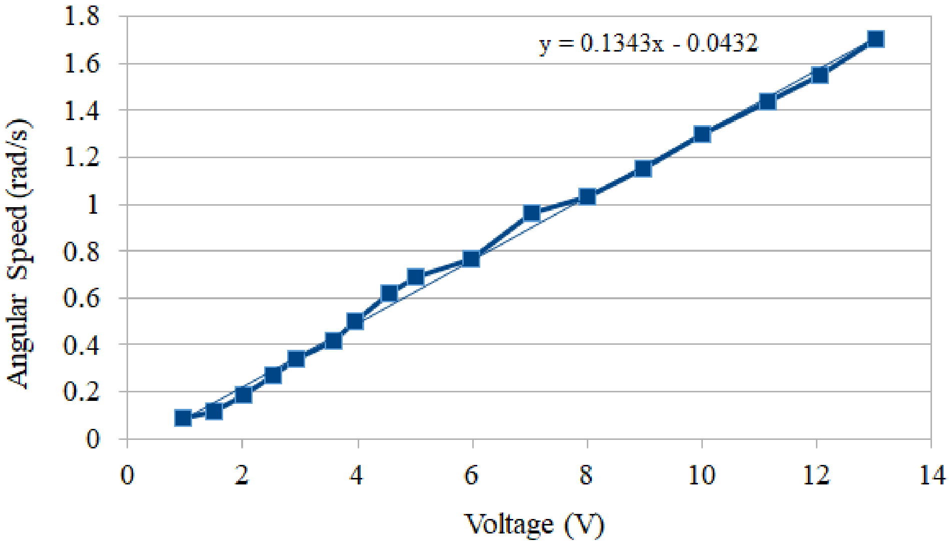


Figure 13. Motor Angular Speed versus Voltage.

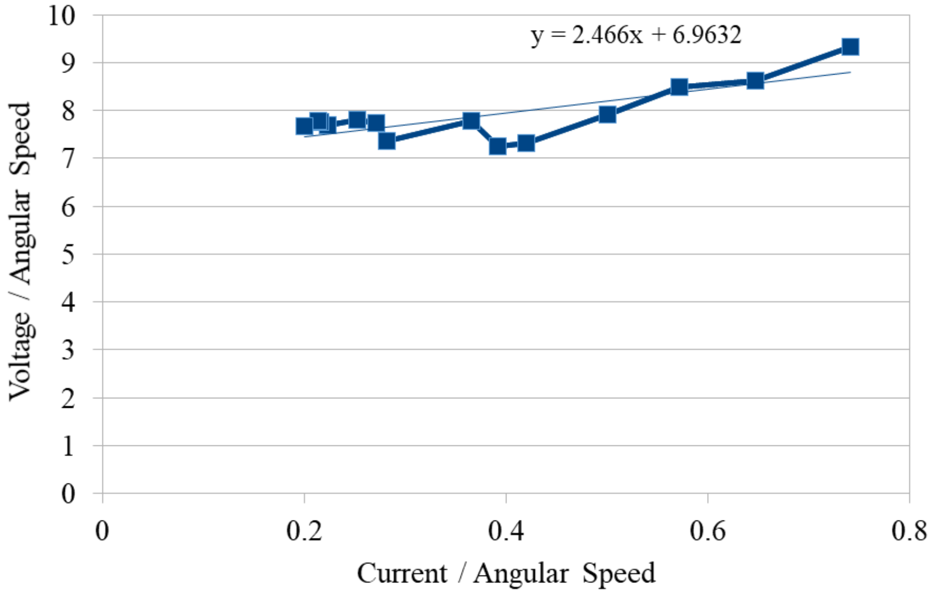


Figure 14. Motor $\frac{Voltage}{AngularSpeed}$ versus $\frac{Current}{AngularSpeed}$.

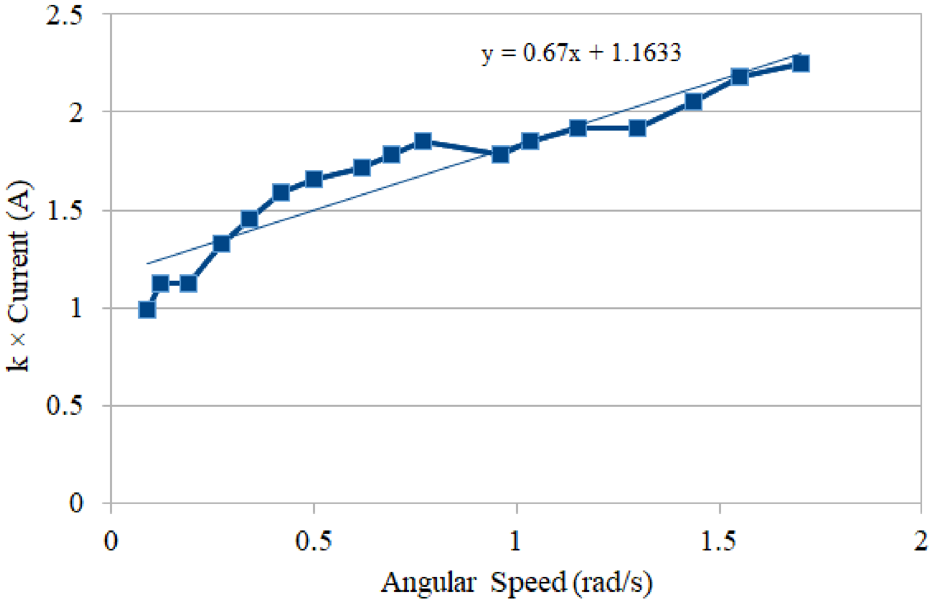


Figure 15. k × Current versus Motor Angular Speed.

The obtained estimated parameters for the motor are presented in Table 3.

Table 3. Estimated motor parameters.

Parameters	
R	2.466 Ω
k	6.9632 N·m/$\sqrt{\text{W}}$
T_q	1.1633 N·m
B_v	0.67 N·m·s

2.4.2. Joint Model

For modeling the non-rigid joint, the conceptual representation used in [21], as well as the applied model were used for the present joint. It is noteworthy that the joint was modeled based on the angular velocity instead of the angular position, unlike the mentioned publication. The obtained estimated parameters for the joint are presented in Table 4.

Table 4. Estimated non-rigid joint parameters.

Parameters	
K	10 N/m
B	0.027 N·s/m
J	0.003 kg·m^2

Through the estimated model, the results presented in Figures 16 and 17 were obtained. It is also important to mention that the initial values of the obtained data were not considered to obtain the model, as can be seen in the graphs. Considering the modeling process, in addition to the commonly presented comparison of the angular positions, the comparison between angular speeds is also presented. The model presents the errors compiled in Table 5.

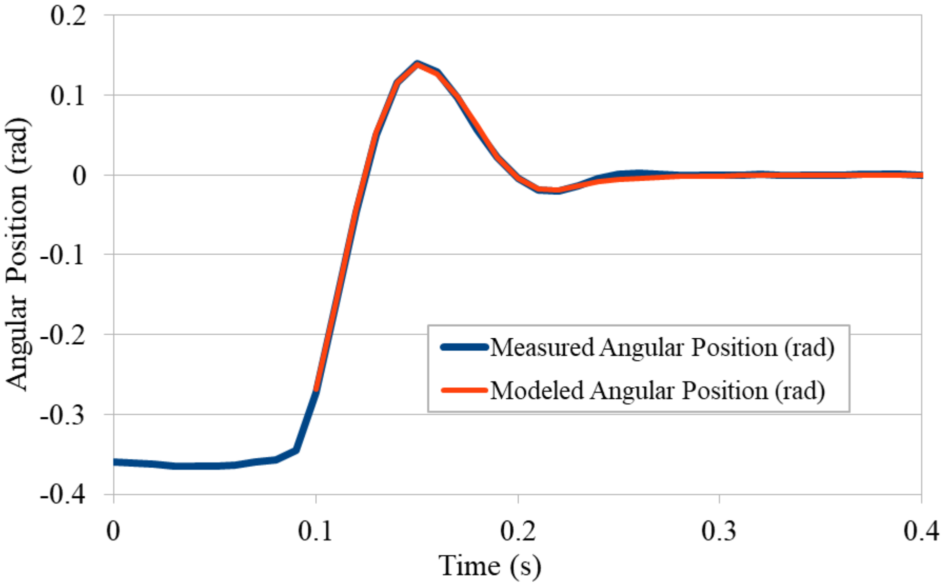


Figure 16. Angular position (rad) over time (s).

Table 5. Errors in angular position and speed.

	Average	Absolute Maximum
Angular Position (rad)	−0.0004	0.0046
Angular Speed (rad/s)	0.1401	0.5491

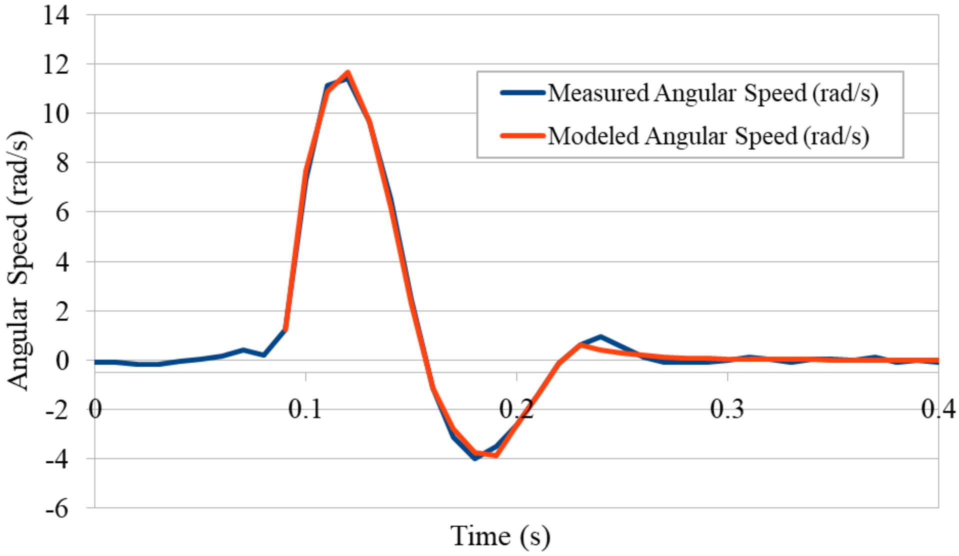


Figure 17. Angular speed (rad/s) over time (s).

2.4.3. Control

Regarding the control technique, a state-space approach is used, similar to the one presented in [21]. Given the use of an absolute encoder on both sides of the joint, this technique improves its behavior. Since now both joints are non-rigid, the controller is equal, except for the different parameter matrices. As described in the previous article, the control model is continuous time using a Zero-Order Hold (ZOH) equivalent to discretize the system, as proposed by Vaccaro [23]. Once again, the considered reference signal is the absolute position, and the block diagram for the implemented controller is presented in Figure 18.

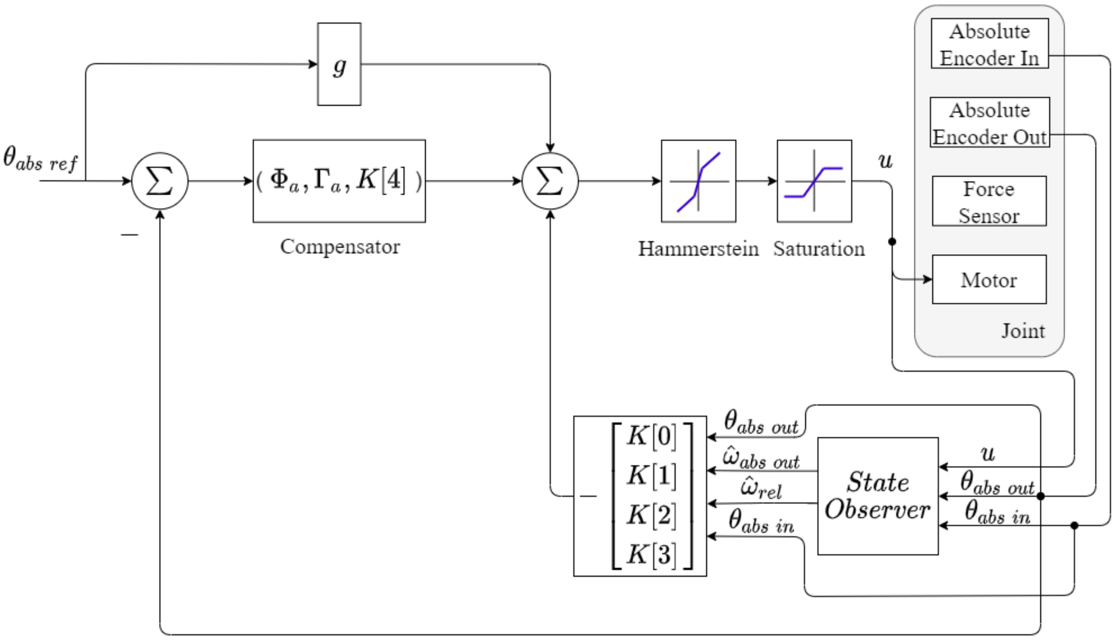


Figure 18. State-space controller block diagram non-rigid joint.

Similar to [21], the controllers receive a reference for a desired wheel position. The complete controller block diagram for all legs is presented in Figure 19.

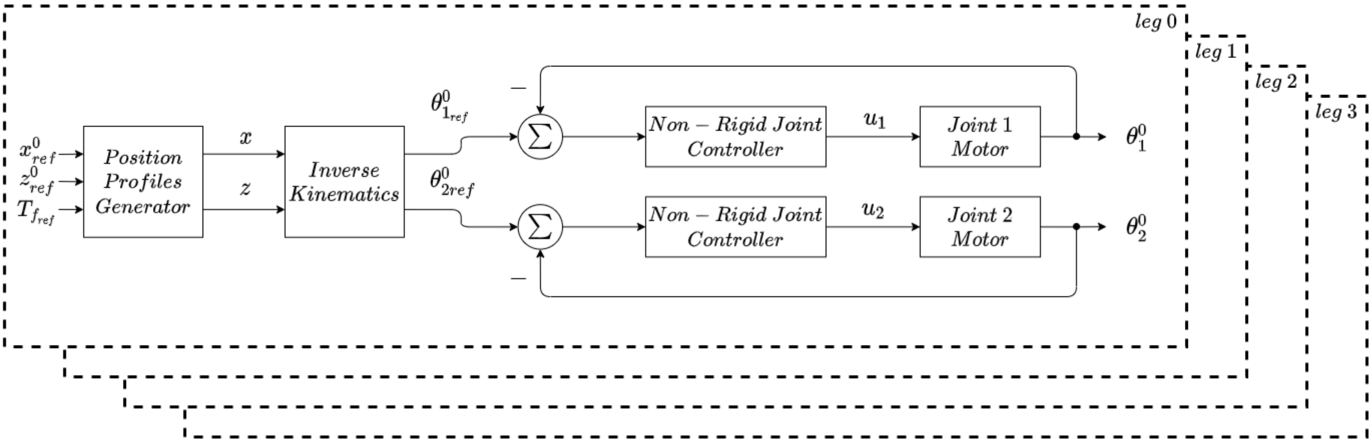


Figure 19. Complete joint control block diagram.

2.5. Robot Kinematics

Regarding the movement of the vehicle, there are three possible locomotion modes—purely legged, purely wheeled, and hybrid. In the purely wheeled mode, whether in low-power or when it is at a defined height, the robot always tries to maintain the wheels' x_l axis aligned with the rotation axis of joint 0, for all legs. This strategy prevents the wheel from dragging laterally when moving the leg, guaranteeing that the contact point of the wheel with the floor does not move in y, rather rotating only on its axis.

The variables used to calculate the robot kinematics are pictured in Figure 20. The variables a, b, and d are the linear distances to the center of the robot. v_r and v_{n_r} represent the robot's linear speeds along the x and y axes, respectively, while ω_r represents its angular speed. For each wheel, a v_{i_w} speed is defined, as well as its orthogonal projections v_x and v_y. The angle θ_{i_w} is the wheel angular position relative to the x axis.

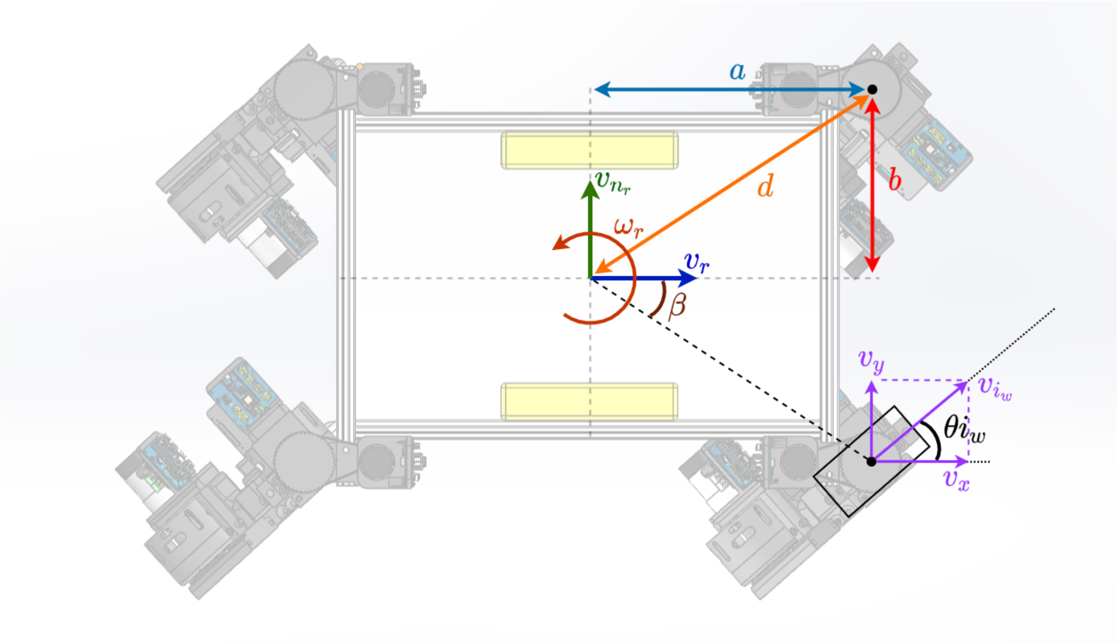


Figure 20. Robot kinematic variables.

Following are the equations relative to the robot kinematics, considering, as already mentioned, that the rotation axis of joint 0 is aligned with the rotation axis of the wheel, regardless of the robot's height. The variables to be controlled are each leg joint 0 angle, which defines the wheel's angle (θ^l_0), as well as the speed of the wheel (v^l_3).

Equations (3) to (12) present the generic deduction of the contribution that each leg will have in the calculations of the robot's v_r, v_{n_r}, and ω_r speeds. Although Equations (3) to (5) are quite intuitive, three computationally demanding functions were used (sin, cos, and atan2). So, Equations (6) to (8) present a simplification set on trigonometric rules, which makes it possible to decrease the number of computationally demanding functions being used. The only one that was not possible to stop using was atan2. The simplified equations are presented in Equations (9) to (12).

$$\beta = \text{atan2}(y, x) \tag{3}$$

$$v_x = v - \omega \cdot d \cdot \sin\beta \tag{4}$$

$$v_y = v_n + \omega \cdot d \cdot \cos\beta \tag{5}$$

$$d = \sqrt{x^2 + y^2} \tag{6}$$

$$\sin\beta = \frac{y}{\sqrt{x^2 + y^2}} = \frac{y}{d} \tag{7}$$

$$\cos\beta = \frac{x}{\sqrt{x^2 + y^2}} = \frac{x}{d} \tag{8}$$

$$v_x = v - \omega \cdot y \tag{9}$$

$$v_y = v_n + \omega \cdot x \tag{10}$$

$$v_{wheel} = \sqrt{v_x^2 + v_y^2} \tag{11}$$

$$\theta_{wheel} = \text{atan2}(v_y, v_x). \tag{12}$$

Next, Table 6 presents the equations that were implemented in software, which allow the calculation of the individual angle of each leg, as well as the respective speed of each wheel.

Table 6. Individual leg kinematic calculation.

Leg 0	Leg 1	Leg 2	Leg 3
$x = a$	$x = a$	$x = -a$	$x = -a$
$y = -b$	$y = b$	$y = b$	$y = -b$
$v_x = v + \omega \cdot b$	$v_x = v - \omega \cdot b$	$v_x = v - \omega \cdot b$	$v_x = v + \omega \cdot b$
$v_y = v_n + \omega \cdot a$	$v_y = v_n + \omega \cdot a$	$v_y = v_n + \omega \cdot a$	$v_y = v_n - \omega \cdot a$

After obtaining $\theta_{0_j\ ref}$ and $\omega_{w_l\ ref}$ for the desired v_r, v_{n_r}, and ω_r, joints 0 are controlled with PID position controllers, while the wheels' speeds are controlled through PID speed controllers. The diagram in Figure 21 presents the mentioned controllers.

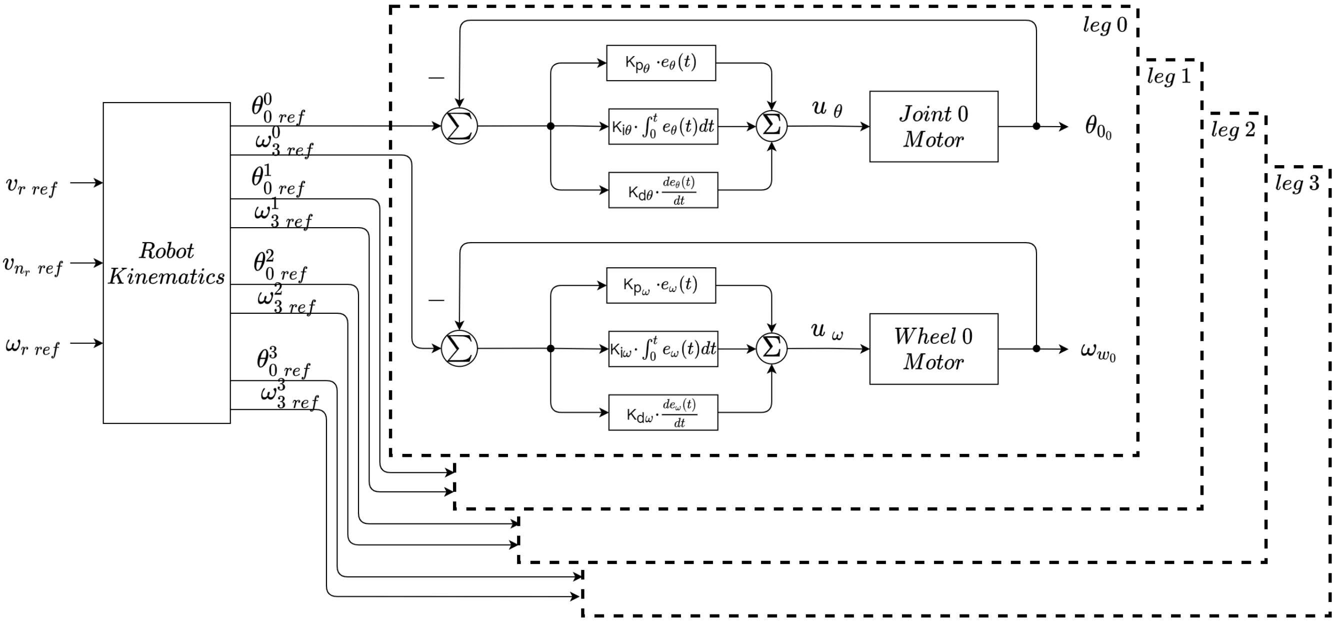


Figure 21. Complete robot control block diagram.

2.6. Simulation Model

The simulation model of this legged-wheeled robot, built in the SimTwo simulator, was created by decomposing it into a set of rigid bodies having mass and size equal to the corresponding parts of the real robot.

Leg links and the main body were represented as cuboids, while joints and wheels as cylinders (joints allow connection between different rigid bodies). Except for the joint for rotation of the omnidirectional wheel, all the others are actuated. Thus, and as shown in Figure 22, the model comprises four legs that connect to the main body on their uppermost joint, and eight wheels.

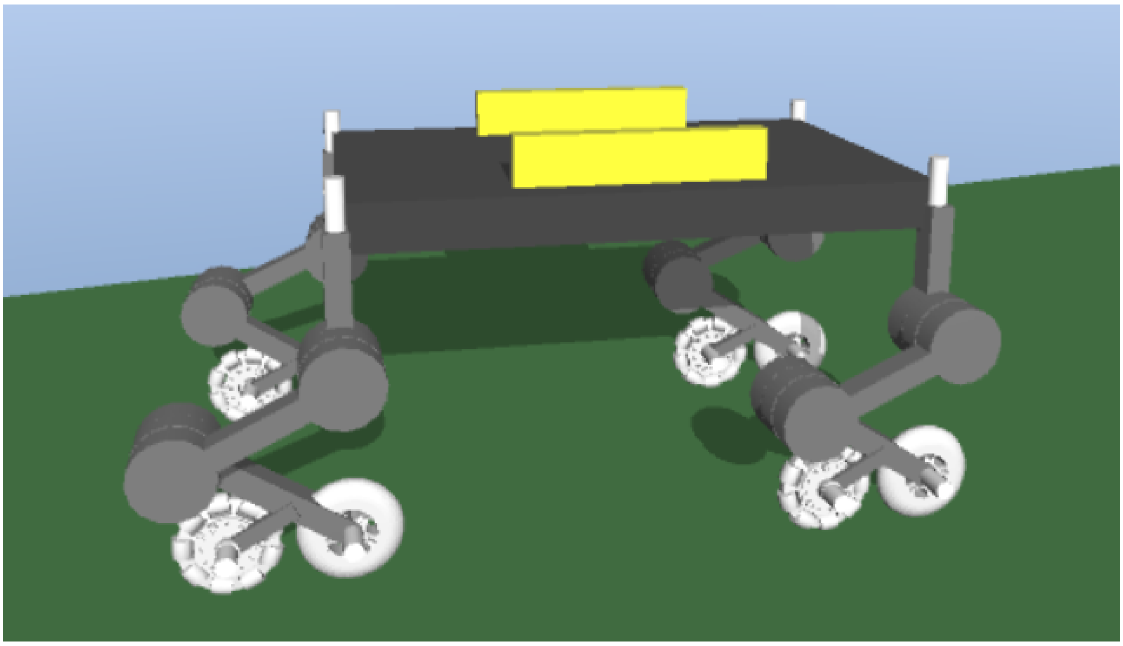


Figure 22. Robot simulation model.

The locomotion system of the robot was modeled by combining, per leg, three links with four joints (includes the wheels' joint). The z-axis is the actuation axis of the hip joint, while the y-axis is that of the remaining ones. Non-rigidity of the two joints that follow the hip joint was modeled by assembling a compound joint: two separate joints connect the leg links to an intermediate cylindrical link; one of them is actuated and the other has a spring (simulating the elastic component of the joint). Omnidirectionality of each non-actuated wheel was established by defining two coefficients of friction onto their surface: one equal to less than one to allow lateral motion, and the other equal to one to hinder forward/backward motion.

In the real robot, while in purely wheeled mode and performing in low power, its lowest height is kept due to mechanical constraints. Modeling this behavior required placing cuboid shells that restrict the robot position to remain at the lowest height as, upon collision, they stay fixed, avoiding rigid bodies from plunging into one another.

As previously mentioned, the robot implements PID controllers for position control of each hip joint and speed control of the wheels, while state-space controllers were developed for control of the non-rigid joints. SimTwo has built-in position and speed PID controllers, which were thus used to control the hip joints and wheels of the model. The state-space controllers were fully implemented within SimTwo using its IDE (integrated development environment).

Control of the vehicle in the x and y axes has been included in simulations by applying the equations of the robot kinematics described in Section 2.5. Consequently, a first approach to trajectory control has been developed.

The implemented tracking method requires the robot to follow a line or curve from an initial point to a target point. A line is defined by a set of two points, while a curve by three. The omnidirectional motion ability of this robot enabled the use of controllers to correct its position with both lateral and angular motion, relative to the error in the distance to the reference line and in the angular position, respectively. When following a line, the reference is the line itself, while, when following a curve, it corresponds to the tangent to the curve in the nearest point between the curve and the robot's position. The linear speed of the robot is constant, up to a defined distance to the target, where it initiates a smooth, linear deceleration motion to reach and stop at the final position. Line tracking requires P controllers in v_{n_r} and w_r, while curve tracking uses PD controllers.

3. Results

To validate the approach used to model, in the simulation, the non-rigid joints of the robot, the elastic behavior was evaluated.

A simple setup to replicate the test described in [20], using the model of the improved non-rigid joint, was built in the simulator with a single (non-actuated) joint connecting two links (one of them fixed); this joint was set as a spring.

The parameters of this spring-mass system include mass (0.400 kg), spring constant k (11.5 N/m), the viscous friction coefficient B_v (0.210 N·m·s, which corresponds to B_v of the rotary damper), and the spring's torque f_c (0.024 N·m).

The oscillatory response of the system was observed by pulling the free link up to a defined position and releasing it (the pull motion was performed by another link driven by a generic motor). The obtained results are shown in Figure 23.

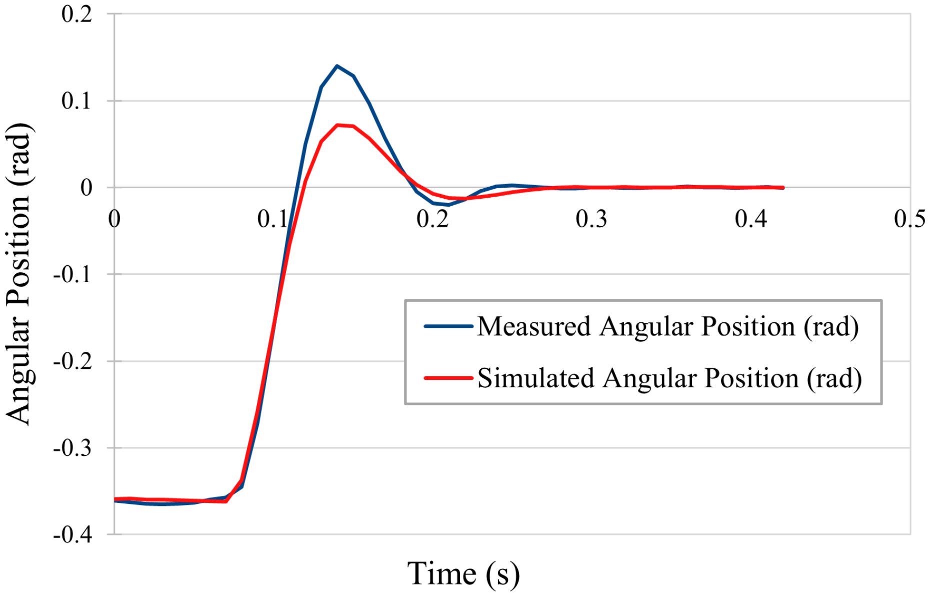


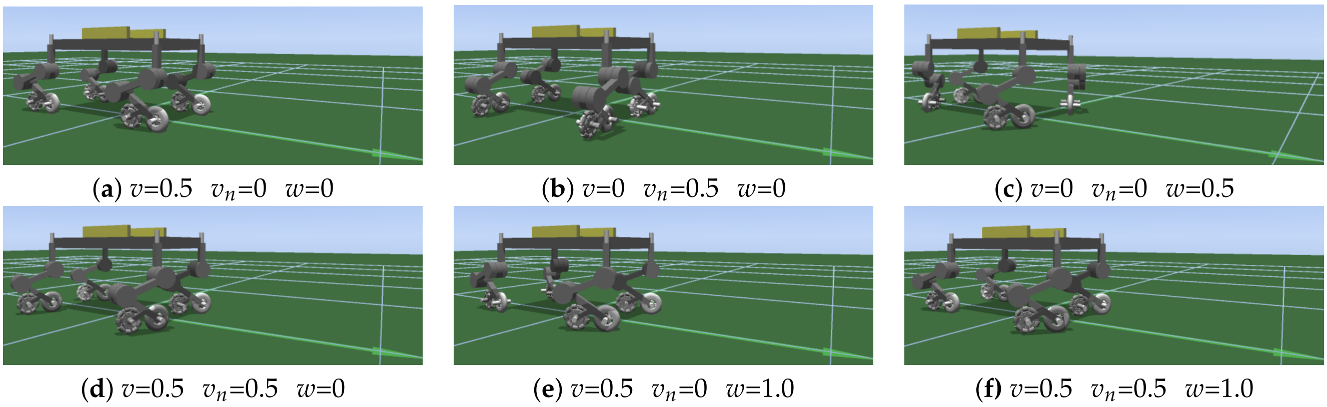
Figure 23. Angular position (rad) over time (s) (simulated non-rigid joint).

The settling time is approximately 188 ms (considering 2% range) and the 98% rise time of around 37 ms. The data collected in the simulation (shown in blue in Figure 23) was compared against the results of the test performed on the real non-rigid joint.

The mean and maximum absolute errors of the position in simulation compared to the real position are, respectively, 0.017 and 0.113 radians. From the analysis of the plotted results, it is noticeable that, even though with similar waveforms, the response in simulation does not reach the peak value observed in the response from the real joint (consequently generating high absolute errors). This can be explained considering the dead zone of the joint (which is yet to be modeled in simulation). Furthermore, uncertainties that derive from measurements in real settings (e.g., load cell noise) are difficult to model in a simulated environment.

Measuring the time interval between the first two zero-crossings of both responses showed a difference of 1 ms; the responses stabilize at around the same time. Therefore, it is confirmed that the simulation models the real behavior of the joint.

The implementation of $x-y$ control of the robot was first assessed by inputting a set of values for v_r, v_{n_r}, and w_r (this test required fixing the main body in the simulation environment). Figure 24 shows the obtained positions of the robot legs. It can be visually determined that, according to the speed references given to the robot, the legs are positioned as expected.



(**a**) v=0.5 v_n=0 w=0

(**b**) v=0 v_n=0.5 w=0

(**c**) v=0 v_n=0 w=0.5

(**d**) v=0.5 v_n=0.5 w=0

(**e**) v=0.5 v_n=0 w=1.0

(**f**) v=0.5 v_n=0.5 w=1.0

Figure 24. Control in the x and y axes: setting the speeds v (m/s), v_n (m/s), and w (rad/s) of the robot (all images were taken from the same perspective; the axis in the foreground corresponds to the x-axis).

The performance of the developed trajectory tracking technique can be analyzed in the results presented in Figures 25, 26, 27, and 28. The characteristics of each trajectory are presented in Table 7.

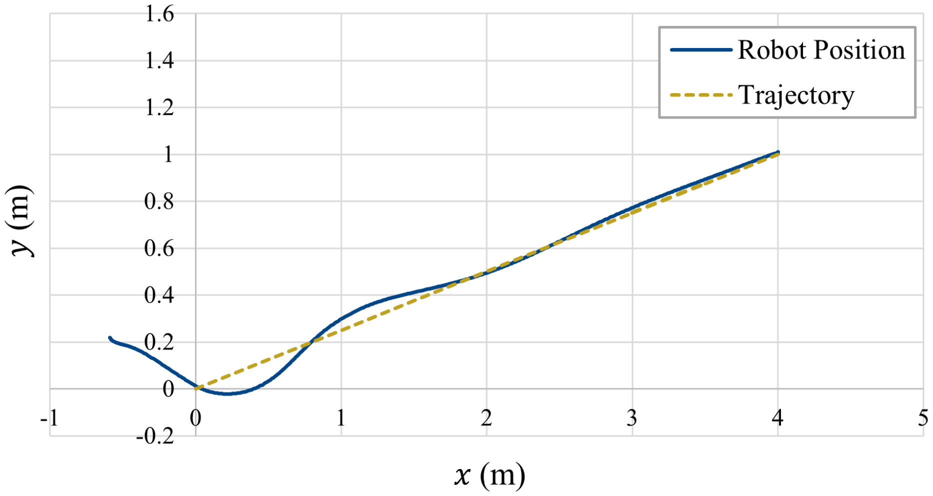


Figure 25. Line tracking: 2D position (m).

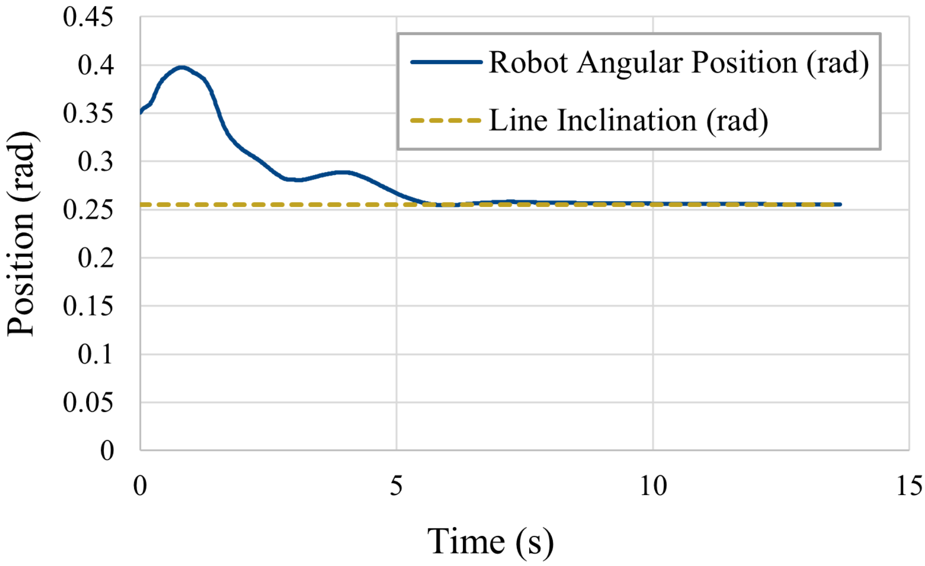


Figure 26. Line tracking: angular position (rad).

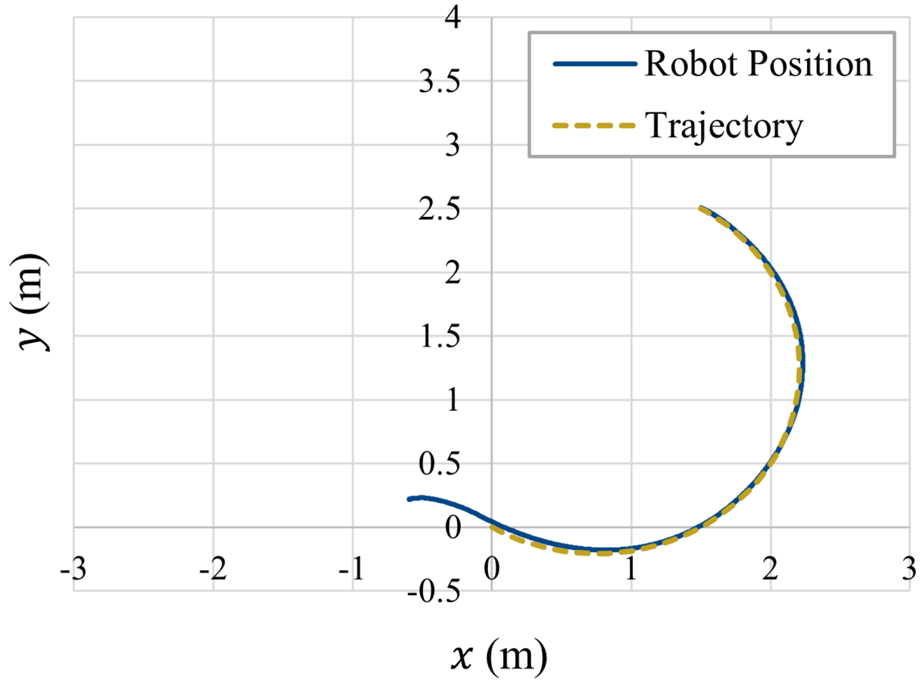


Figure 27. Curve tracking: 2D position (m).

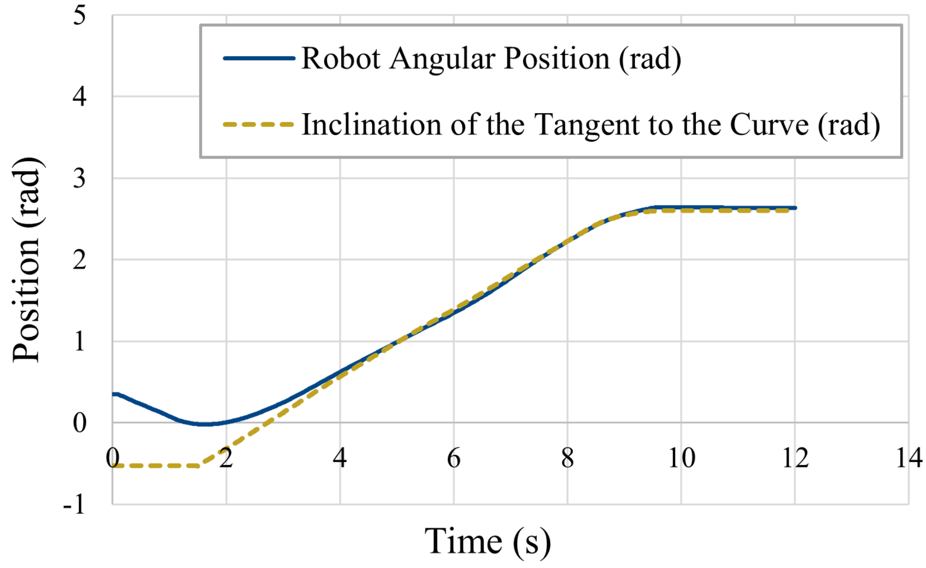


Figure 28. Curve tracking: angular position (rad).

Table 7. Characteristics of the trajectories presented in Figures 25 and 27.

	Line	Curve
Initial position (m)	(0, 0)	(0, 0)
Target position (m)	(4.0, 1.0)	(4.0, 1.0)
Inclination (rad)	0.255	—
Center (m) ∣ Radius (m)	—	(0.75, 1.25) ∣ 1.458

The robot is able to converge to the trajectory it is given and stop at the target position. The absolute errors of the robot's position and angle in relation to the trajectory are summarized in Table 8.

Table 8. Absolute errors in distance and angular position in relation to the trajectory.

	Average	**Maximum**
	Line	
Distance (m)	0.018	0.097
Angular position (rad)	0.014	0.129
	Curve	
Distance (m)	0.027	0.138
Angular position (rad)	0.060	0.514

In addition to the simulation tests, tests on the real robot have already been performed. The following link can be used to observe the robot performing some movements: https://youtu.be/0xVQSwRMrDs, accessed on 30 June 2021.

4. Conclusions and Future Work

As described throughout this article, the designed platform has been successfully implemented. The robot was assembled and has already been tested in indoor environments. The current implemented control techniques allow the robot to move with controlled v_r, v_{n_r}, and ω_r speeds, as well as its height (z_r).

The models for the motors used and for the improved joints were obtained and included in the robot's simulation model developed with the SimTwo simulator. The simulation results were compared with the experimental data, revealing a high similarity.

The low errors obtained allowed us to proceed with the implementation of $x - y$ control of the robot and a trajectory tracking algorithm in the simulated environment. Regarding the latter, its performance was assessed and, from the small errors observed, it can be concluded to be a viable strategy for the motion of the vehicle in its workspace.

In the future, as the simulation has been validated and proved to be realistic, further work will continue in order to include z_r-axis control of the robot, testing algorithms for trajectory control and planning (with obstacle detection and avoidance), and developing the gait planning framework of the robot.

Concerning the real vehicle, to perceive the surrounding space, it is necessary to add LiDAR sensors to this platform, acquire their data, and use them to recognize obstacles and contours such as steps and walls. It will then become possible to continue exploring the robot, namely, for stair climbing, reactive or predictive obstacle avoidance, as well as gait planning—particularly trotting. Furthermore, interesting data to collect includes the robot's maximum achievable speeds and energy consumption in each mode.

Supplementary Materials: A video presenting the experiments is available online at https://youtu.be/0xVQSwRMrDs.

Author Contributions: Conceptualization, V.H.P. and P.C.; methodology, V.H.P. and P.C.; software, P.C., V.H.P. and I.N.S.; validation, V.H.P., I.N.S., M.R., J.G., J.L. and P.C.; investigation, V.H.P., I.N.S. and M.R.; writing—original draft preparation, V.H.P. and I.N.S.; writing—review and editing, V.H.P., I.N.S., J.G., J.L. and P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project UIDB/50014/2020.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Raibert, M.; Blankespoor, K.; Nelson, G.; Playter, R. BigDog, the Rough-Terrain Quadruped Robot. *IFAC Proc. Vol.* **2008**, *41*, 10822–10825. [CrossRef]
2. Bledt, G.; Powell, M.J.; Katz, B.; Di Carlo, J.; Wensing, P.M.; Kim, S. MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2245–2252. [CrossRef]
3. Tang, Z.; Qi, P.; Dai, J. Mechanism design of a biomimetic quadruped robot. *Ind. Robot. Int. J.* **2017**, *44*, 512–520. [CrossRef]
4. Hutter, M.; Gehring, C.; Jud, D.; Lauber, A.; Bellicoso, C.D.; Tsounis, V.; Hwangbo, J.; Bodie, K.; Fankhauser, P.; Bloesch, M.; et al. ANYmal—A highly mobile and dynamic quadrupedal robot. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 38–44.
5. Yang, K.; Zhou, L.; Rong, X.; Li, Y. Onboard hydraulic system controller design for quadruped robot driven by gasoline engine. *Mechatronics* **2018**, *52*, 36–48. [CrossRef]
6. Hooks, J.; Ahn, M.S.; Yu, J.; Zhang, X.; Zhu, T.; Chae, H.; Hong, D. ALPHRED: A Multi-Modal Operations Quadruped Robot for Package Delivery Applications. *IEEE Robot. Autom. Lett.* **2020**, *5*, 5409–5416. [CrossRef]
7. Bjelonic, M.; Sankar, P.K.; Bellicoso, C.D.; Vallery, H.; Hutter, M. Rolling in the Deep–Hybrid Locomotion for Wheeled-Legged Robots using Online Trajectory Optimization. *IEEE Robot. Autom. Lett.* **2020**, *5*, 3626–3633. [CrossRef]
8. Bouton, A.; Grand, C.; Benamar, F. Design and Control of a Compliant Wheel-on-Leg Rover Which Conforms to Uneven Terrain. *IEEE/ASME Trans. Mechatronics* **2020**, *25*, 2354–2363. [CrossRef]
9. Schwarz, M.; Rodehutskors, T.; Droeschel, D.; Beul, M.; Schreiber, M.; Araslanov, N.; Ivanov, I.; Lenz, C.; Razlaw, J.; Schüller, S.; et al. NimbRo Rescue: Solving disaster-response tasks with the mobile manipulation robot Momaro. *J. Field Robot.* **2017**, *34*, 400–425. [CrossRef]
10. Kashiri, N.; Baccelliere, L.; Muratore, L.; Laurenzi, A.; Ren, Z.; Hoffman, E.M.; Kamedula, M.; Rigano, G.F.; Malzahn, J.; Cordasco, S.; et al. CENTAURO: A hybrid locomotion and high power resilient manipulation platform. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1595–1602. [CrossRef]
11. Jdeed, M.; Zhevzhyk, S.; Steinkellner, F.; Elmenreich, W. Spiderino—A low-cost robot for swarm research and educational purposes. In Proceedings of the 2017 13th Workshop on Intelligent Solutions in Embedded Systems (WISES), Hamburg, Germany, 12–13 June 2017; pp. 35–39. [CrossRef]
12. Gealy, D.V.; McKinley, S.; Yi, B.; Wu, P.; Downey, P.R.; Balke, G.; Zhao, A.; Guo, M.; Thomasson, R.; Sinclair, A.; et al. Quasi-Direct Drive for Low-Cost Compliant Robotic Manipulation. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 437–443. [CrossRef]
13. Armesto, L.; Fuentes-Durá, P.; Perry, D. Low-cost printable robots in education. *J. Intell. Robot. Syst.* **2016**, *81*, 5–24. [CrossRef]
14. Brunete, A.; Ranganath, A.; Segovia, S.; de Frutos, J.P.; Hernando, M.; Gambao, E. Current trends in reconfigurable modular robots design. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1729881417710457. [CrossRef]
15. Grimminger, F.; Meduri, A.; Khadiv, M.; Viereck, J.; Wüthrich, M.; Naveau, M.; Berenz, V.; Heim, S.; Widmaier, F.; Flayols, T.; et al. An Open Torque-Controlled Modular Robot Architecture for Legged Locomotion Research. *IEEE Robot. Autom. Lett.* **2020**, *5*, 3650–3657. [CrossRef]
16. Camargo, C.; Gonçalves, J.; Conde, M.Á.; Rodríguez-Sedano, F.J.; Costa, P.; García-Peñalvo, F.J. Systematic Literature Review of Realistic Simulators Applied in Educational Robotics Context. *Sensors* **2021**, *21*, 4031. [CrossRef]
17. Kamedula, M.; Kashiri, N.; Tsagarakis, N.G. On the Kinematics of Wheeled Motion Control of a Hybrid Wheeled-Legged CENTAURO robot. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2426–2433.
18. Suzumura, A.; Fujimoto, Y. Real-Time Motion Generation and Control Systems for High Wheel-Legged Robot Mobility. *IEEE Trans. Ind. Electron.* **2013**, *61*, 3648–3659. [CrossRef]
19. Chen, Z.; Wang, S.; Wang, J.; Xu, K.; Lei, T.; Zhang, H.; Wang, X.; Liu, D.; Si, J. Control strategy of stable walking for a hexapod wheel-legged robot. *ISA Trans.* **2020**. [CrossRef] [PubMed]
20. Pinto, V.H.; Gonçalves, J.; Costa, P. Towards a More Robust Non-Rigid Robotic Joint. *Appl. Syst. Innov.* **2020**, *3*, 45. [CrossRef]
21. Pinto, V.H.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of a Single Leg for a Legged-Wheeled Locomotion System with Non-Rigid Joint. *Actuators* **2021**, *10*, 29. [CrossRef]
22. Pinto, V.H.; Gonçalves, J.; Costa, P. Modeling and Control of a DC Motor Coupled to a Non-Rigid Joint. *Appl. Syst. Innov.* **2020**, *3*, 24. [CrossRef]
23. Vaccaro, R.J. *Digital Control: A State-Space Approach*; McGraw-Hill Higher Education: New York, NY, USA, 1995.