Trajectory Planning for Pioneer Robot

Jessé Alves, Camila Larocca and Márcio Silva

***Resumo*— This report presents the results of the development and simulation of the Pionner robot's trajectory planning and control project, using the Python programming language and *software* V-REP as the robot's simulation environment. The trajectory generation method considered the robot non-holonomic and proportional control was implemented in this experiment.**

***Palavras-Chave*— Trajectory planning, Pioneer robot, non-holonomic, proportional control.**

I. Introduction

A. Trajectory Planning

In mobile robotics, generating a trajectory for the robot's path is an essential part when it comes to movement planning. This is because designing an autonomous robotic system that performs satisfactory tasks is not a trivial task. This block requires a reconciliation between an admissible path - taking into account not only the environment, but also the construction of the robot - and the time to execute the task. Furthermore, the motion planner block of a mobile robot is usually divided into smaller parts, as shown in Fig **??**. In block (1), a set of configurations and positions is determined and subsequently transmitted to block (2). In this, a path achievable by the robot in function form is generated based on the received path configurations. Finally, block (3) of the controller receives the trajectory as a reference signal, sending control signals to the actuators to perform the task.

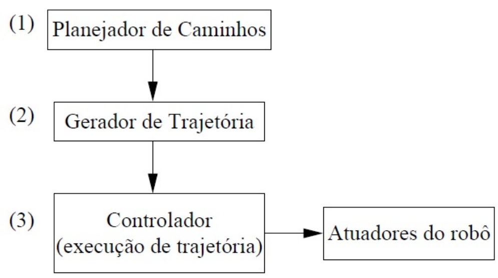

Fig. 1. Block diagram of a motion planner.
Source: Adapted from (Pedrosa, 2003).

II. Development and Simulation

Initially, a kinematic model for the Pioneer robot was developed using two classes in Python, whose drive is of the differential type, as shown in Fig 2. In the modeling, some pre-established data were also inserted, such as the dimensions of the Pioneer, and some relations for assistance, such as the relationship between the linear speeds of the right and left wheel and angular and the rotation matrix R(θ), which relates the coordinate system fixed to the robot with the global coordinate system.

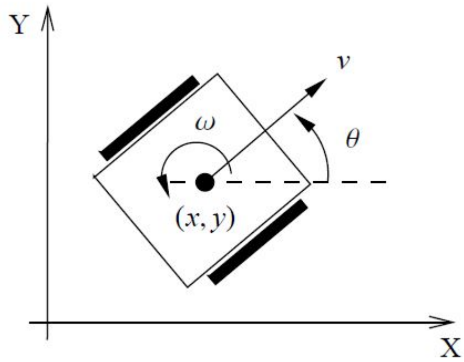

Fig. 2. Robot with differential drive.
Source: Adapted from (Pedrosa, 2003).

In V-REP, the Pioneer robot and a Cuboid were added to the simulation environment, as shown in Fig. 3. The cuboid was named *Target* and *Non-threaded child Script* was disabled as the initial configuration. of the Pioneer robot. In the first part of the code, a connection with the V-REP simulator was opened . The position $\{X, Y, Z\}$ of the Pioneer and *Target* are acquired, considered the initial and final position of the robot.

With the initial and final positions defined, modeling of a non-Holonomic path generator is used to find the global velocity of the robot v and the angular velocity of the robot w, step by step. Then, the synchronous communication channel between the Python development environment and the Simulator was opened again, sending information about the speeds of the right and left wheels with the command *simxSetJointTargetVelocity*.

Control of the robot's trajectory was based on a proportional type controller - P, where the trajectory is calculated based on the error, being considered a closed loop with unity proportional gain. The P controller is a simple controller and presents a steady-state error, which we can verify by comparing the trajectory calculated from the developed kinematic model and the trajectory simulated in V-REP, as shown in figure 4 below.

Jessé Alves, Postgraduate Program in Electrical Engineering (PPGEE), Federal University of Bahia (UFBA), Salvador-BA, e-mail: jessalves2@gmail.com; Camila Larocca, Postgraduate Program in Electrical Engineering (PPGEE), Federal University of Bahia (UFBA), Salvador-BA, e-mail: camilla.larocca@gmail.com; Márcio Silva, Postgraduate Program in Electrical Engineering (PPGEE), Federal University of Bahia (UFBA), Salvador-BA, email: marcio.cs@ufba.br.

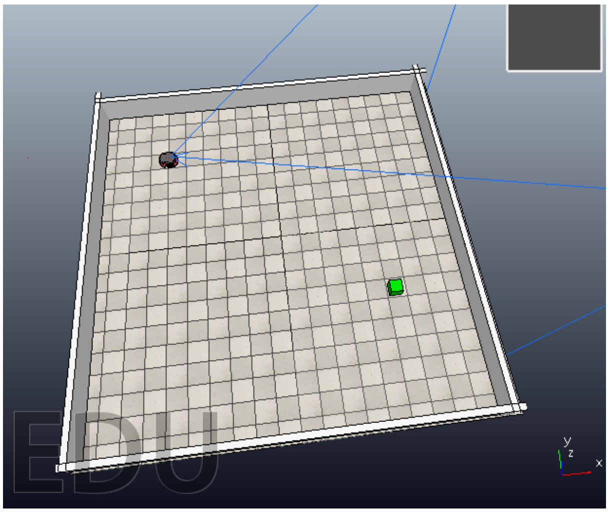

Fig. 3. V-REP simulation environment

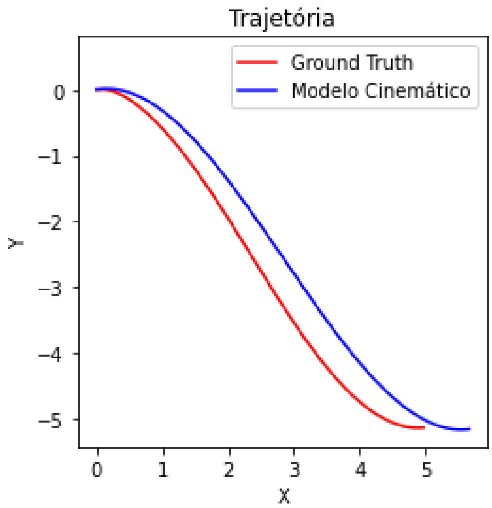

Fig. 4. Comparison between trajectories calculated from the kinematic model and the trajectory simulated in V-REP

Complementarily, Figure 5 demonstrates the angular speed of the left and right wheels during the robot's trajectory. For the robot to make the initial turn to the right, the angular velocity of the left wheel is greater than the angular velocity of the right wheel. When making the turn to the left, at the end of the route, this scenario is reversed.

III. Conclusions

The results shown in the II section demonstrate a significant difference between the trajectory obtained by *Ground Truth* and the trajectory obtained by the kinematic model. This is because the proportional control alone was not enough to follow the reference accurately, requiring better tuning. Furthermore, it is noteworthy that carrying out this project demonstrated the challenges and non-triviality of planning a trajectory for an autonomous system.

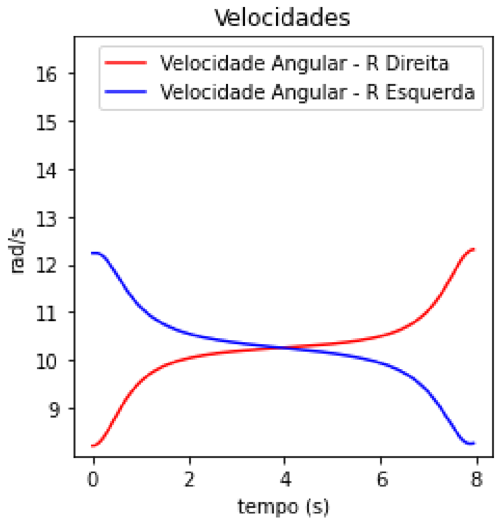

Fig. 5. V-REP simulation environment

Referências

[1] Pedrosa DPF, Medeiros AAD and Alsina PJ, *A trajectory generation method for robots non-holonomic with differential drive*. VI Brazilian Symposium on Intelligent Automation. Bauru, 2003.

[2] R. Siegwart, I.R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, The MIT Press, 2004.

[3] Park, JJ, Kuipers, B. *A Smooth Control Law for Graceful Motion of Differential Wheeled Mobile Robots in 2D Environment*, The MIT Press, 2004.