IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 58, No. 2, February, 2025, pp. 353-364
Journal homepage: http://iieta.org/journals/jesa

Computation of Feasible Controls and Feasible States for a Four-Wheeled Autonomous Vehicle Robot

Masiala Mavungu*, Daniel Mashao

Faculty of Engineering and Built Environment, University of Johannesburg, Johannesburg 2006, South Africa

Corresponding Author Email: masmavp@yahoo.fr

https://doi.org/10.18280/jesa.580216

Received: 16 August 2024
Revised: 29 October 2024
Accepted: 20 November 2024
Available online: 28 February 2025

Keywords:
autonomous vehicle robot, differential equation, four-wheeled vehicle, non-holonomic constraints, optimal control, path planning, open-loop control

ABSTRACT

This research paper deals with computation of Feasible command strategies to guide a four-wheeled autonomous vehicle to move from a specified initial state to an eventual final state while minimizing a running cost. The problem is solved as follows: The vehicle is mathematically modeled as a non-linear system of seven ODEs with seven state and four command variables. The commands represent the controls. A costate system of seven ODEs is generated. The control's Feasibility conditions generate four constrained Feasible command strategies and each of them is defined as a function of states and costates. The state system and the costate system are rewritten accordingly. Such systems are then joined to give a nonlinear system of fourteen ODEs and an initial value problem. The results are the Feasible system response involving the robot path in the horizontal XY plane and the robot velocity, the costate functions, the command functions. Computational Simulations are developed and presented to summarize the results and to convince the readers on the accuracy.

1. INTRODUCTION

Nowadays, there exist several activities which are very difficult and dangerous for human to execute. This is the case for work in hazardous nuclear fields, mine fields, very narrow or confined fields, or fields requiring a lot of effort and attention for humans, etc. This inspired some researchers to focus their studies on robots and specifically autonomous vehicles. This increased interest from researchers, which can be noticed by the amount of papers on autonomous vehicles carried out over the last decades.

An autonomous vehicle is a self-propelled vehicle which moves without a driver and without human intervention. For example, it may be automated that a vehicle moves from a point to another. The self-propelled vehicle may also be planned to move from any point and follow asymptotically a prescribed trajectory. It may also be set to move from a random point to another or from a fixed point to a possible final point. This paper deals specifically with a four-wheeled vehicle robot in which the front wheels are connected to an axle as well as the rear wheels. The wheels are assumed to roll without slipping. This implies the existence of nonholonomic constraints from which a system of three ordinary differential equation (ODE)s defining a part of the vehicle kinematics is constructed. The second assumption is that each wheel is regulated independently by a reference command which controls its angular velocity. The existence of four wheels leads to the existence of four desired and reference commands. The third assumption is that all the four-wheel instantaneous angular velocities are different. The combination of all the assumptions enables the construction of a state-space non-linear system of seven ODEs involving seven state variables and four command variables which are the control variables. In this case the above-mentioned desired commands and the state variables (SV)s constitute the input control variables and vehicle system response respectively.

A numerical scheme is developed to compute Feasible Command strategies to instruct the autonomous vehicle to drive from a specified initial state to a desired final state in a given time slot such that a performance index is minimized. Such performance index is the running energy cost. Existing literature has a considerable and significant number of papers carried out over the past years investigating the mathematical modelling and computational simulations of autonomous vehicles and on the robot in general. Costa et al. [1] examined the robot competition which happens in an emulated factory plant, Zyada et al. [2] applied Fuzzy Logic technique to deal with manipulation control, Wang et al. [3] suggests a control policy for a robot fish. Grandi et al. [4] implemented the UniBot remote Lab, Schneider et al. [5] investigated on a surgical robot motion, Huang [6] dealt with one-wheeled vehicles for which he develops a self-balancing controller (SBC), Sayyaadi et al. [7] addressed a problem on car-like multi robots, Ata and Myo [8] controlled a riderless autonomous bicycle by using inverse dynamic control. Becerra [9] described an obstacle avoidance method for a nonholonomic vehicle, Chachuat [10] introduced the nonlinear programming problem. Magrab and Azarm [11] provided notes to develop to readers a grasp of MATLAB programming techniques. Gu and Hu [12] developed a receding horizon (RH) controller for tracking control of a nonholonomic mobile robot. Hussein and Bloch [13] used an affine connection formulation to analyze and solve an optimal control problem for a class of nonholonomic, underactuated mechanical systems. Kirk [14] introduces the three facets of optimal control which are dynamic programming, Pontryagin's Minimum Principle and numerical techniques for trajectory optimization. Masoud and Masoud [15] discussed the generation of a control signal that would instruct the actuators of a robotics manipulator to drive motion along a safe and well-behaved path to a desired target. Roberts and Shipman [16] developed some shooting techniques to solve two-point boundary value problems. Yang et al. [17] addressed the problem of formation control and obstacle avoidance for a group of nonholonomic mobile robots. Tsuji et al. [18] considered a new trajectory generation method of the artificial potential field approach

to a real-time motion planning problem. Klancarc and Skrjanc [19] presented a model-predictive trajectory-tracking control for the path planning of a mobile robot. Krotov [20] used a fireman method to solve optimal control problems. Kumar and Sukavanam [21] developed a trajectory tracking control based on the kinematics for a four wheeled mobile robot. Lamber [22] discussed numerical methods for solving initial value problems of ordinary differential equations. Moore [23] provided some notions to get readers master MATLAB programming. This paper uses Octave / MATLAB to solve the problem. Zhang et al. [24] investigated the optimization problem by using optimal control method. Tyatyushkin et al. [25] developed an algorithmic support for some classes of nonlinear optimal control problems. Buldaev [26] addressed a new procedure to optimization of nonlinear control systems. All the above-cited papers concern autonomous robots. To the best of my knowledge, none of those papers considered the control of a four-wheeled autonomous vehicle and applied optimal control theory methods. None of them performs the computation of feasible state functions of the system, the computation of the corresponding feasible costate functions and the computation of feasible speed. Generally, one can clearly see in most of papers dealing with a four-wheeled autonomous vehicle that, before applying optimal control theory, the system is primarily simplified to enable easily the computation of its variables and parameters. Most often, the state system is transformed, using a critical point, into a linear one. In those papers certain parameters and / or are set to zeros. The simplifications are executed to enable the application of certain operations such as designing a quadratic regulator. However, when a nonlinear control system is simplied into a linear one, if no further operation is performed to prevent loss of information and alteration, then the final solution will not be significantly reliable. The approximate solution will be significantly different to the exact solution. As one can see, in this paper the control system is highly nonlinear and no simplification has been performed on it. No other paper has considered the problem in which each wheel has a reference command. No paper has considered the problem of running cost minimization. None of them has computed the optimality conditions. No paper has performed the derivation of a costate system of ODEs adjoining the state system. Also, in the literature, no paper reduces the impact of the size of the initial costate vector on the length of the robot trajectory. This paper uses optimal control theory to compute feasible control strategies of an autonomous vehicle and feasible state trajectories such that the vehicle running cost is minimized.

This article relates to operations research, optimal control theory, robotics, signal and image processing and computer vision. It makes use of optimal control theory to solve a robotic problem. This paper develops mathematical models and computational simulations and most of the solved problems are formulated in terms of optimal control problems.

This paper develops open-loop control strategies for a four-wheeled autonomous vehicle. In most papers, the control vector is defined by the translational velocity and the angular velocity whereas in this paper each wheel's angular velocity is independently regulated by a reference command variable which is a control variable. Conditions of optimality are derived and have generated an initial value problem. Different cases for the initial state vector are considered to predict and assess the trajectory of the vehicle. Since the system of ODEs joining the state and the costate system is highly nonlinear, the authors use a numerical method to solve it. This study developed Computer programs to produce computational simulations, which summarize the results and provide evidence about their reliability and effectiveness.

The substantial contributions of this paper are the construction of mathematical models, the computation of four feasible control strategies, the computation of seven feasible state functions representing the feasible system responses to reference commands, and the computation of seven costate (adjoint) functions. All the computations are made possible by the underlying computer programs developed in Octave / Matlab. The way the four-wheeled autonomous vehicle is modelled can inspire the modelling of any other autonomous vehicle and in some ways and any autonomous system in general. In order to doublecheck and confirm the reliability and the effectiveness of my results obtained with Octave / MATLAB, some computer programs were developped with Scilab and results were computed and compared. For the gaps, the controls found in literature have some limitations in handling the state system. Most of time the system is simplified before computing those controls. But in my paper the controls can handle any size and any kind of state system nonlinearity. They can handle any state system as it is. There will be no need to simplify the system. The developed robust Octave / MATLAB and Scilab Computer Program functions can handle any system nonlinearity or singularity.

The advancement of science and technology led the universities to define new and multidisciplinary research areas. It also enabled automation industries to find new and optimal strategies to design and manage projects for their businesses such that profits are maximized and costs of operation are minimized. Depending on the economic and political world, each industrial research project can be associated to an optimization aspect. Further, it inspired the governments to update their regulations accordingly. This article seeks to contribute to the advancement of science and technology as it can be used by any academic and / or any industrial researcher working on robotics and / or optimal control theory. In the academic environment, depending on the specific research focus, this study provides several significant and considerable insights to define new research directions in robotics, in optimal control, in numerical and computational simulations. Also, it inspires studies of kinematics and dynamics of autonomous vehicles which are very critical in the manufacturing industries. Further, this article can also enable researchers and academics to predict the behaviour of self-propelled vehicles and to perform an optimal path planning based on the vehicle's settings. The Mathematical Models and Computational Simulations developed in this study provide an unprecedented and unique approach for predicting the behaviour of autonomous vehicles. From the developed mathematical models, one can study the vehicle performance against uncertainties. In the industrial sector, this paper is useful to all staffs and practitioners working in the manufacturing of autonomous vehicles. It enables them to perform simulations in the path planning projects of autonomous vehicles. It enables the simulations of any other robots. The paper is structured as follows: After this introductory part, Section 2 outlines the mathematical models defined by the running cost to minimize, the state-space system from which is based this paper, Section 3 defines the methodology and designs a schematic to summarize the procedure for solving the problem. Section 4 focuses on computing the system Hamiltonian, which prepares the derivation of normal equations of optimality addressed in Section 4. Section 5 provides solutions to the normal equations of optimality as well as numerical and computational simulations for the results of the joined state-costate system of ODEs. To confirm the accuracies of the results, two versions of Scilab computer programs were developed for the same problems and one can see that the results are the same as the ones obtained by using Octave / MATLAB computer programs. All the computer programs are in the Appendix.

2. MATHEMATICAL MODEL

2.1 Kinematic model

The nonholonomic constraints of the vehicle are defined as:

$$\frac{dx}{dt} = k_1 \sum_{i=1}^{4} \omega_i \cos(\theta) \quad (1)$$

$$\frac{dy}{dt} = k_1 \sum_{i=1}^{4} \omega_i \sin(\theta) \quad (2)$$

$$\frac{d\theta}{dt} = k_2 \sum_{i=1}^{4} \omega_i \quad (3)$$

The wheels' angular velocities are governed by a closed-

loop system of separable ODEs given by:

$$\frac{d\omega_1}{dt} = -l_1\omega_1 + l_1u_1, -\frac{2\pi}{3}rad/s \le u_1 \le \frac{2\pi}{3}rad/s \quad (4)$$

$$\frac{d\omega_2}{dt} = -l_2\omega_2 + l_2u_2, -\frac{2\pi}{3}rad/s \le u_2 \le \frac{2\pi}{3}rad/s \quad (5)$$

$$\frac{d\omega_3}{dt} = -l_3\omega_3 + l_3u_3, -\frac{2\pi}{3}rad/s \le u_3 \le \frac{2\pi}{3}rad/s \quad (6)$$

$$\frac{d\omega_4}{dt} = -l_4\omega_4 + l_4u_4, -\frac{2\pi}{3}rad/s \le u_4 \le \frac{2\pi}{3}rad/s \quad (7)$$

with, $-\frac{2\pi}{3}rad/s \le u_i \le \frac{2\pi}{3}rad/s$, i=1, 2, 3, 4.

The whole nonlinear state-space system determining the kinematics of the vehicle is given by:

$$\frac{dx}{dt} = k_1\sum_{i=1}^{4}\omega_i\cos(\theta) \quad (8)$$

$$\frac{dy}{dt} = k_1\sum_{i=1}^{4}\omega_i\sin(\theta) \quad (9)$$

$$\frac{d\theta}{dt} = k_2\sum_{i=1}^{4}\omega_i \quad (10)$$

$$\frac{d\omega_1}{dt} = -l_1\omega_1 + l_1u_1 \quad (11)$$

$$\frac{d\omega_2}{dt} = -l_2\omega_2 + l_2u_2 \quad (12)$$

$$\frac{d\omega_3}{dt} = -l_3\omega_3 + l_3u_3 \quad (13)$$

$$\frac{d\omega_4}{dt} = -l_4\omega_4 + l_4u_4 \quad (14)$$

where, (x, y) is the vehicle's position on the horizontal plane, θ is the heading angle that the robot makes with the x axis, ω_i is the angular velocity of wheel i, u_i is the commanded angular velocity constituting the control variable, k_i, i=1, 2 and l_i, i=1, 2, 3, 4 are constant of proportionality. From what that precedes, notice that in general the four wheel angular velocities satisfy the following rules: $\frac{d\omega_i}{dt} = -l_i\omega_i + l_iu_i$ for i=1, 2, 3, 4, where u_i is the reference command to wheel i which is its control variable.

Figure 1. Four-wheeled autonomous vehicle (DARPA)
Note: DARPA: Defence Advanced Research Project Agency. It is a United States Agency

The system defined by the Eqs. (8)-(14) constitutes the kinematic model of the vehicle. Such a vehicle is represented by Figure 1. In vector form, such a model can be rewritten as:

$$\frac{dY}{dt} = f(t, Y, u) \quad (15)$$

where, t is the independent time variable, $Y = [x, y, \theta, \omega_1, \omega_2, \omega_3, \omega_4] \in \mathbb{R}^7$ is the state vector, $u = [u_1, u_2, u_3, u_4] \in \mathbb{R}^4$ is the reference command (control) vector.

2.2 Objective functional

The total cost to minimize is given by:

$$J = \int_0^T(\sum_{i=1}^4 c_iu_i^2)dt \quad (16)$$

where, for each i=1, 2, 3, 4, c_i is a constant of proportionality, u_i is a reference command for the vehicle's wheel i; $\sum_{i=1}^4 c_iu_i^2$ is the cost rate. $J = \int_0^T(\sum_{i=1}^4 c_iu_i^2)dt$ is the total running cost. Each unknown reference command u_i is a control function.

2.3 Problem formulation

This paper's problem is the following: Compute the feasible command strategies u_i i=1, 2, 3, 4 and the corresponding feasible state functions x, y, θ, ω_1, ω_2, ω_3, ω_4, also called feasible robot system responses, to guide the four-wheeled vehicle from a specified initial state to a final state while minimizing the total running cost.

3. METHODOLOGY

Mathematical Modeling and Computational Simulations constitute the methodology used. As one can see above, the car is modeled as a system of ODEs. For computational simulations, Octave / MATLAB and Scilab Computer Programs are developed. The procedure for solving the problem is summarized in Figure 2.

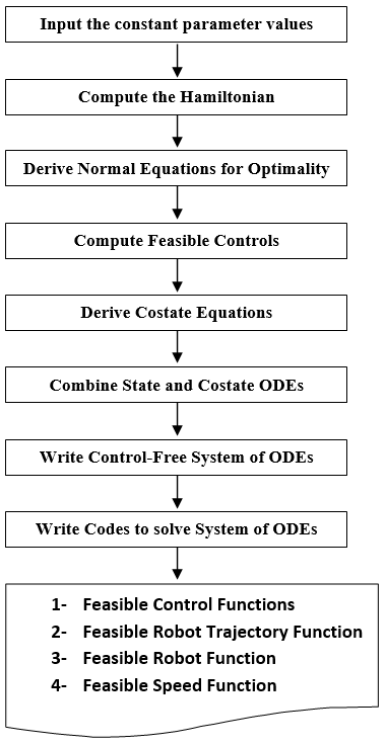


Figure 2. Problem solving procedure

4. COMPUTATION OF THE HAMILTONIAN

The Hamiltonian is defined as follows:

$$H(t, Y, u, \alpha)=\sum_{i=1}^{4} c_i {u_i}^2 + \sum_{i=1}^{7} \alpha_i f_i(t, Y, u) \quad (17)$$

where, $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7]^T \in \mathbb{R}^7$ is the vector storing the costate variables. $\frac{1}{2}\sum_{i=1}^{4} c_i {u_i}^2$ is the rate at which the energy is spent by the vehicle. Each of the following function represents the right hand side of the state system, $f_1(t, Y, u) = k_1 \sum_{i=1}^{4} \omega_i \cos(\theta)$ is the rate of change of the robot position along x direction, $f_2(t, Y, u) = k_1 \sum_{i=1}^{4} \omega_i \sin(\theta)$ is the rate of change of the robot position along y direction, $f_3(t, Y, u) = k_2(\omega_1 - \omega_2)$ is the rate of change of the heading angle,

$f_4(t, Y, u) = -l_1\omega_1 + l_1 u_1$,
$f_5(t, Y, u) = -l_2\omega_2 + l_2 u_2$,
$f_6(t, Y, u) = -l_3\omega_3 + l_3 u_3$,
$f_7(t, Y, u) = -l_4\omega_4 + l_4 u_4$.

In general we have, $f_i(t, Y, u) = -l_{i-3}\omega_{i-3} + l_{i-3}u_{i-3}$ is the rate of change of the wheel i angular velocity.

5. CONDITIONS FOR OPTIMALITY

5.1 Normal conditions of optimality

Principle of optimality is going to be considered in this paper. Such a principle will generate a set of equations leading to the computation of feasible commands from which can be obtained the feasible system response and additional derivations.

The normal equations (for the controls) for optimality are given by:

$$\frac{\partial H}{\partial u_1} = 2c_1u_1 + \alpha_4 l_1 = 0 \quad (18)$$

$$\frac{\partial H}{\partial u_2} = 2c_2u_2 + \alpha_5 l_2 = 0 \quad (19)$$

$$\frac{\partial H}{\partial u_3} = 2c_3u_3 + \alpha_6 l_3 = 0 \quad (20)$$

$$\frac{\partial H}{\partial u_4} = 2c_4u_4 + \alpha_7 l_4 = 0 \quad (21)$$

which can be written in general as:

$$\frac{\partial H}{\partial u_i} = 2c_i u_i + l_i \alpha_{i+3} = 0 \text{ for } i = 1,2,3,4 \quad (22)$$

which solved for the control functions, give:

$$u_1 = -0.5(l_1/c_1)\alpha_4 \quad (23)$$

$$u_2 = -0.5(l_2/c_2)\alpha_5 \quad (24)$$

$$u_3 = -0.5(l_3/c_3)\alpha_6 \quad (25)$$

$$u_4 = -0.5(l_4/c_4)\alpha_7 \quad (26)$$

and can be expressed in general as $u_i = -(l_i/c_i)\alpha_{i+3}$ where $u_{min} \leq -(l_i/c_i)\alpha_{i+3} \leq u_{max}$ for $i = 1,2,3,4$.

5.2 Costate equations

The costate system is given by:

$$\frac{d\alpha_1}{dt} = -\frac{\partial H}{\partial y_1} = -\frac{\partial H}{\partial x} = 0 \quad (27)$$

$$\frac{d\alpha_2}{dt} = -\frac{\partial H}{\partial y_1} = -\frac{\partial H}{\partial y} = 0 \quad (28)$$

$$\frac{d\alpha_3}{dt} = -\frac{\partial H}{\partial \theta} = k_1 \sum_{i=1}^{4} \omega_i \left(\alpha_1 \sin(\theta) - \alpha_2 \cos(\theta)\right) \quad (29)$$

$$\frac{d\alpha_4}{dt} = -\frac{\partial H}{\partial \omega_1} = -k_1(\alpha_1 \cos(\theta) + \alpha_2 \sin(\theta)) - (k_2\alpha_3 - l_1\alpha_4) \quad (30)$$

$$\frac{d\alpha_5}{dt} = -\frac{\partial H}{\partial \omega_2} = -k_1(\alpha_1 \cos(\theta) + \alpha_2 \sin(\theta)) - (k_2\alpha_3 - l_2\alpha_5) \quad (31)$$

$$\frac{d\alpha_6}{dt} = -\frac{\partial H}{\partial \omega_3} = -k_1(\alpha_1 \cos(\theta) + \alpha_2 \sin(\theta)) - (k_2\alpha_3 - l_3\alpha_6) \quad (32)$$

$$\frac{d\alpha_7}{dt} = -\frac{\partial H}{\partial \omega_4} = -k_1(\alpha_1 \cos(\theta) + \alpha_2 \sin(\theta)) - (k_2\alpha_3 - l_4\alpha_7) \quad (33)$$

6. STATE-COSTATE SYSTEM

The first-order necessary conditions of optimality are derived and then yield a system of ODEs containing the feasible state and the feasible costate equations. Initial conditions are associated to the system from which are obtained the feasible solution.

Define $z=[Y, \alpha]$. The combined state system and costate system can be defined in general as:

$$z=[z_1, \ldots, z_{14}] \quad (34)$$

where, $z \in \mathbb{R}^{14}$ is the vector storing the state variables and the costate variablhes in which we have $z_1=x$; $z_2=y$; $z_3=\theta$; $z_4=\omega_1$; $z_5=\omega_2$; $z_6=\omega_3$; $z_7=\omega_4$; $z_8=\alpha_1$; $z_9=\alpha_2$; $z_{10}=\alpha_3$; $z_{11}=\alpha_4$; $z_{12}=\alpha_5$; $z_{13}=\alpha_6$; $z_{14}=\alpha_7$, $u \in \mathbb{R}^4$ is the vector storing the control variables. Then, the combination of the state and costate systems gives the following:

$$\frac{dz_1}{dt} = k_1 \sum_{i=4}^{7} z_i \cos(z_3) \quad (35)$$

$$\frac{dz_2}{dt} = k_1 \sum_{i=4}^{7} z_i \sin(z_3) \quad (36)$$

$$\frac{dz_3}{dt} = k_2 \sum_{i=4}^{7} z_i \quad (37)$$

$$\frac{dz_4}{dt} = -l_1 z_4 + l_1 u_1 \quad (38)$$

$$\frac{dz_5}{dt} = -l_2 z_5 + l_2 u_2 \quad (39)$$

$$\frac{dz_6}{dt} = -l_3 z_6 + l_3 u_3 \quad (40)$$

$$\frac{dz_7}{dt} = -l_4 z_7 + l_4 u_4 \quad (41)$$

$$\frac{dz_8}{dt} = 0 \tag{42}$$

$$\frac{dz_9}{dt} = 0 \tag{43}$$

$$\frac{dz_{10}}{dt} = k_1 \sum_{i=4}^{7} z_i \ (z_8 \sin(z_3) - z_9 \cos(z_3)) \tag{44}$$

$$\frac{dz_{11}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_1 z_{11}) \tag{45}$$

$$\frac{dz_{12}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_2 z_{12}) \tag{46}$$

$$\frac{dz_{13}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_3 z_{13}) \tag{47}$$

$$\frac{dz_{14}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_4 z_{14}) \tag{48}$$

In order to get the control functions vanished in the control system and remain with only the state variables and the costate variables, substitute the expressions of the control functions defined by Eqs. (23)-(26) into the combined state and costate system. We then obtain the following system:

$$\frac{dz_1}{dt} = k_1 \sum_{i=4}^{7} z_i \cos(z_3) \tag{49}$$

$$\frac{dz_2}{dt} = k_1 \sum_{i=4}^{7} z_i \sin(z_3) \tag{50}$$

$$\frac{dz_3}{dt} = k_2 \sum_{i=4}^{7} z_i \tag{51}$$

$$\frac{dz_4}{dt} = -l_1 z_4 - \frac{0.5 {l_1}^2 z_{11}}{c_1} \tag{52}$$

$$\frac{dz_5}{dt} = -l_2 z_5 - \frac{0.5 {l_2}^2 z_{12}}{c_2} \tag{53}$$

$$\frac{dz_6}{dt} = -l_3 z_6 - \frac{0.5 {l_3}^2 z_{13}}{c_3} \tag{54}$$

$$\frac{dz_7}{dt} = -l_4 z_7 - \frac{0.5 {l_4}^2 z_{14}}{c_4} \tag{55}$$

$$\frac{dz_8}{dt} = 0 \tag{56}$$

$$\frac{dz_9}{dt} = 0 \tag{57}$$

$$\frac{dz_{10}}{dt} = k_1 \sum_{i=4}^{7} z_i \ (z_8 \sin(z_3) - z_9 \cos(z_3)) \tag{58}$$

$$\frac{dz_{11}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_1 z_{11}) \tag{59}$$

$$\frac{dz_{12}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_2 z_{12}) \tag{60}$$

$$\frac{dz_{13}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_3 z_{13}) \tag{61}$$

$$\frac{dz_{14}}{dt} = -k_1(z_8 \cos(z_3) + z_9 \sin(z_3)) - (k_2 z_{10} - l_4 z_{14}) \tag{62}$$

subject to the following initial conditions:

$$z_k(t_0) = z_{0k} \text{ for } 1 \le k \le 14 \tag{63}$$

where, t_0 is the initial time and t_f is the final time for the robot motion.

7. NUMERICAL AND COMPUTATIONAL SIMULATIONS

The following numerical constants are used: a = 0.4; Radius of each wheel. L=1; Length of each axle. k1=2; Constant of proportionality. k2=0.5*k1; Constant of proportionality. k1=a/2; k2=a/L; l1=0.6; l2=0.6; l3=0.48; l4=0.48; c1=1; c2=1; c3=1; c4=1; t0=0; tf=15; n=501; umin=-2*pi/3; umax=2*pi/3; The aim is to first apply a 4th order R-K method to solve the associated initial value problem:

$$\frac{dZ}{dt} = g(t, Z, u), Z(t_0) = Z_0 \tag{64}$$

The discrete points of each component Z_i, i=1, 2, …, 14 of Z are obtained as:

$$Z_{i,j} = Z_i(t_j) \tag{65}$$

where, we have:

$$Z_{i,0} = Z_i(t_0) \tag{66}$$

In other words, for each feasible state function or costate function Z_i, $Z_{i,0}=Z_i(t_0)$ is the initial condition. The set of ODEs (49)-(63) is coded into an Octave / MATLAB function and called "four_wheel_robot(t,z)". Such a system is solved using the function called "runge(fs, t0, tf, N, z0)".

Such a function is called by an Octave / MATLAB function which is the main function called function main_four_wheel.

For the set ODEs' initial conditions, 4 cases of z0 are considered, as shown below:

Case1:
[zeros(7,1);0.96;0.83;0.17;0.56;0.24;0.70;0.29]
Case2:
[0;0;pi/2;zeros(4,1);0.38;0.91;0.35;0.21;0.34;0.32;0.46]
Case3:
[zeros(7,1);0.01;0.84;0.01;0.40;0.16;0.89;0.33]
Case4:
[0;0;pi/2;zeros(4,1);0.68;0.08;0.54;0.45;0.02;0.17;0.83]

For each given initial condition, the first seven components represent the initial state vector and the last seven components represent the initial costate vector. The order in which they are processed to give the below computational simulations depends on the order in which the above matrix elements z0 are stored.

Figure 3 gives the controls1 defined by $u_1(t)$. $u_1(t)$ regulates the angular velocity $\omega_1(t)$ of wheel1. Each $u_1(t)$ is associated to a given initial condition. Since we have 4 initial conditions then we have 4 computational simulations for $u_1(t)$.

Figure 4 gives the controls2 defined by $u_2(t)$. $u_2(t)$ regulates the angular velocity ω_2(t) of wheel2. Each $u_2(t)$ is associated

to a given initial condition. Since we have 4 initial conditions then we have 4 computational simulations for $u_2(t)$.

Figure 5 gives the controls3 defined by $u_3(t)$. $u_3(t)$ regulates the angular velocity $\omega_3(t)$ of wheel1. Each $u_3(t)$ is associated to a given initial condition. Since we have 4 initial conditions then we have 4 computational simulations for $u_3(t)$.

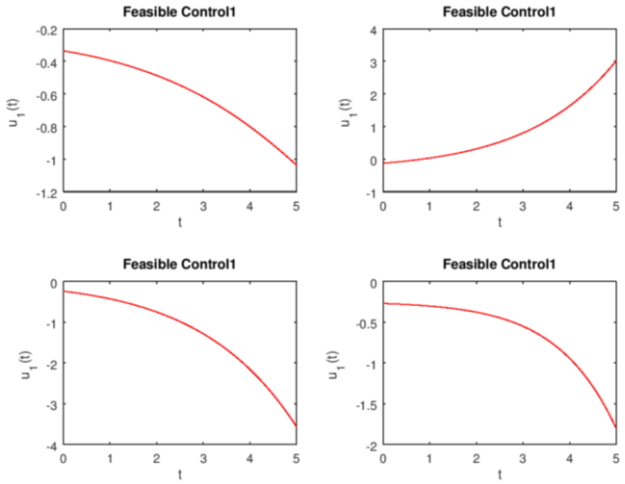


Figure 3. Feasible control strategy 1

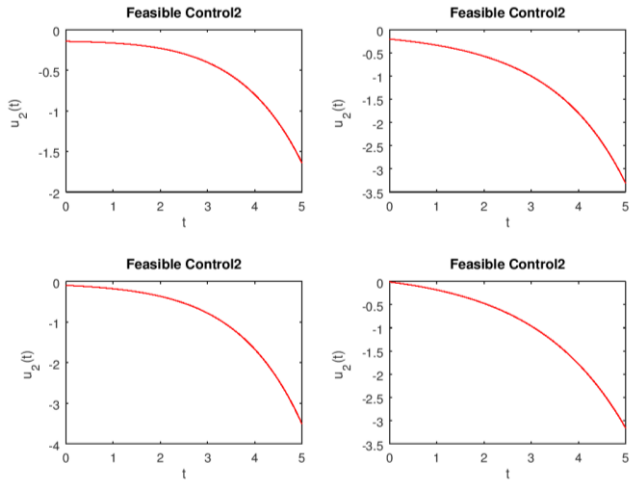


Figure 4. Feasible control strategy 2

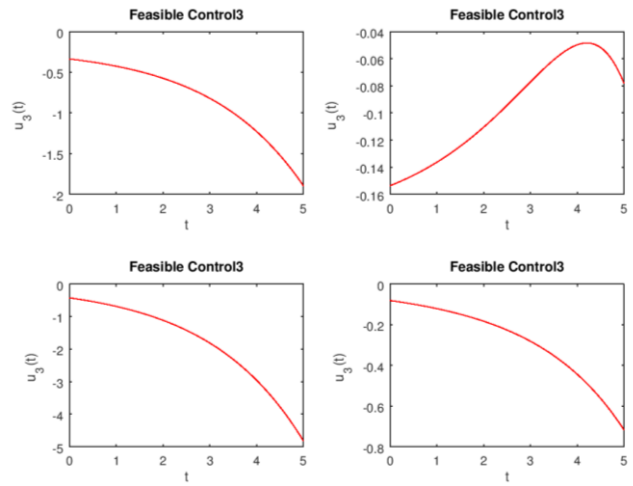


Figure 5. Feasible control strategy 3

Figure 6 gives the controls4 defined by $u_4(t)$. $u_4(t)$ regulates the angular velocity $\omega_4(t)$ of wheel4. Each $u_4(t)$ is associated to a given initial condition. Since we have 4 initial conditions then we have 4 computational simulations for $u_4(t)$. $u_1(t)$, $u_2(t)$, $u_3(t)$ and $u_4(t)$ are the feasible controls that regulate respectively the angular velocities $\omega_1(t)$, $\omega_2(t)$, $\omega_3(t)$ and $\omega_4(t)$ of the four wheels.

Figure 7 gives the trajectories travelled by the autonomous vehicle robot. (0.0) is the starting point for each trajectory. Each trajectory is associated to a given initial condition. Since we have 4 initial conditions then we have 4 computational simulations. The robot trajectory depends on the initial condition. The starting point is the origine. For each point on the robot trajectory, the direction of the corresponding tangent vector to the curve at the point depends on the initial condition of the state-costate system of ODEs. Figure 8 represents the robot speed function. At each point of the curve defining the robot speed function, the direction of the corresponding tangent vector depends on the initial condition. The above plotted computational simulations are the speeds of the robot. Each speed function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 speed computational simulations.

Figure 9 gives the first state, the x position of the robot. Each graph of state1 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state1 computational simulations.

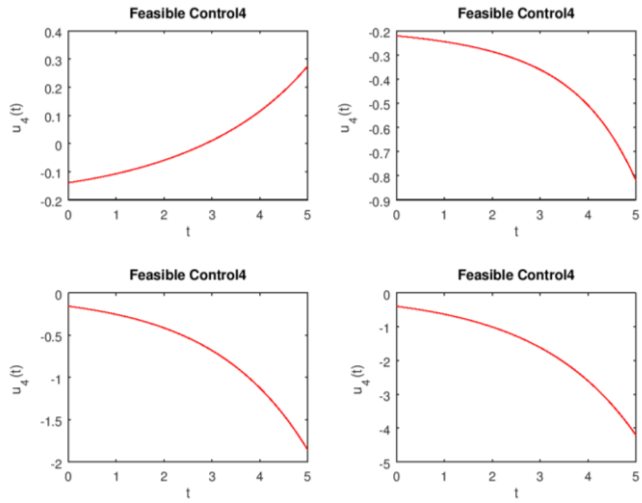


Figure 6. Feasible control strategy 4

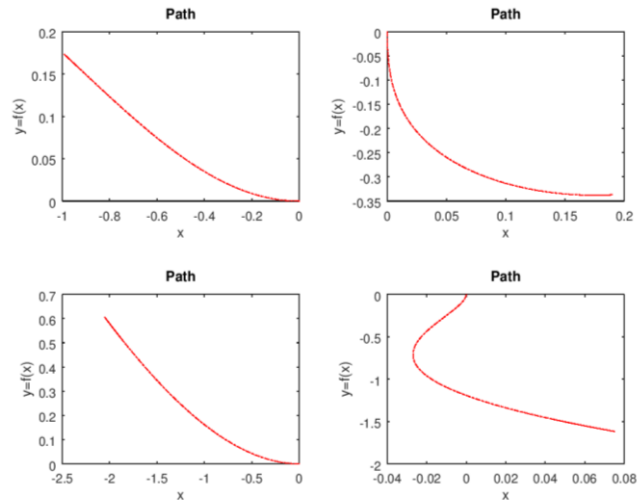


Figure 7. Feasible trajectories of the vehicle robot

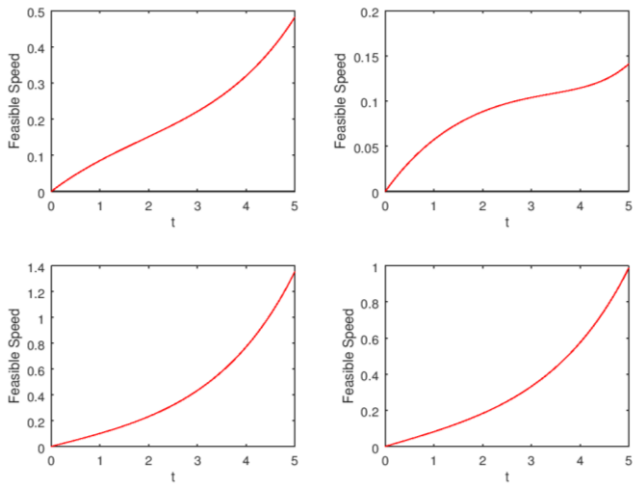


Figure 8. Feasible robot speeds

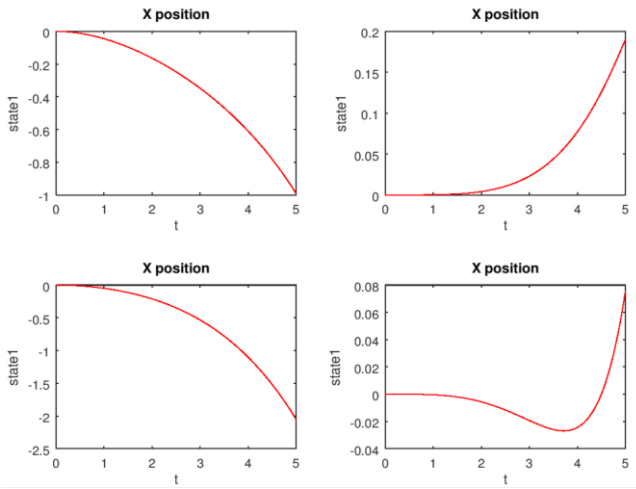


Figure 9. Feasible state1 (x position) of the vehicle robot

Figure 10 gives the second state, the y position of the robot. Each graph of state2 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state2 computational simulations.

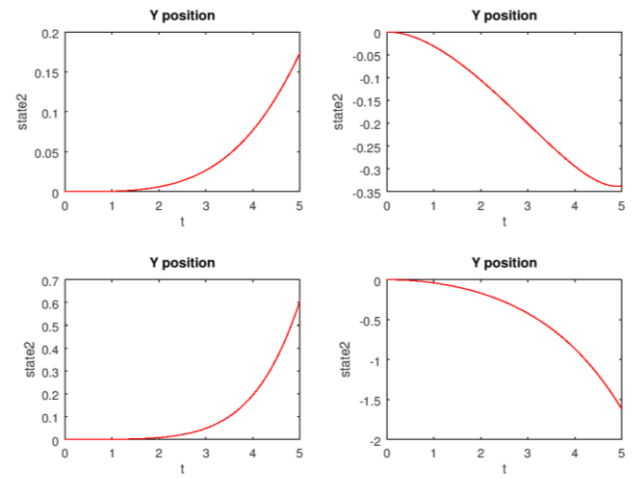


Figure 10. Feasible state2 (y position) of the vehicle robot

Figure 11 gives the third state, the heading angle of the robot. Each graph of state3 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state3 computational simulations.

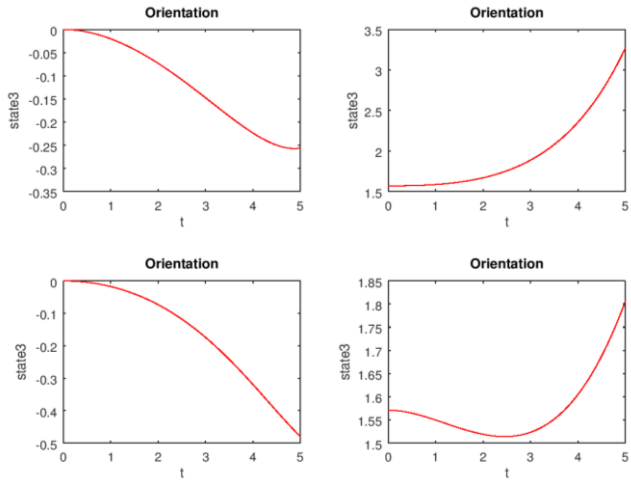


Figure 11. Feasible state3 (heading angle) of the vehicle robot

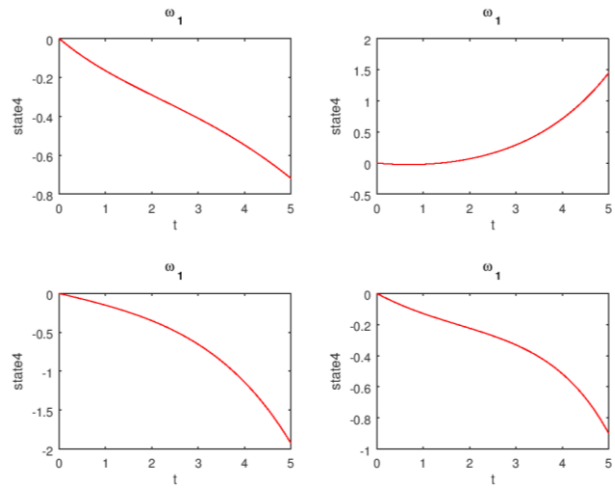


Figure 12. Feasible state4 of the vehicle robot

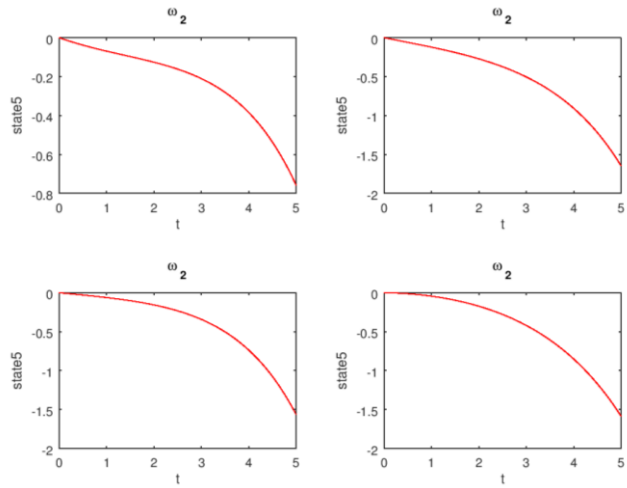


Figure 13. Feasible state5 of the vehicle robot

Figure 12 gives the fourth state of the robot. Such a state function represents the angular velocity of Wheel1. Each graph of state4 function is associated to a given initial

condition. Since we have 4 initial conditions then we have 4 state4 computational simulations.

Figure 13 gives the fifth state of the robot. Such a state function represents the angular velocity of Wheel2. Each graph of state5 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state5 computational simulations.

Figure 14 gives the sixth state of the robot. Such a state function represents the angular velocity of Wheel3. Each graph of state6 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state6 computational simulations.

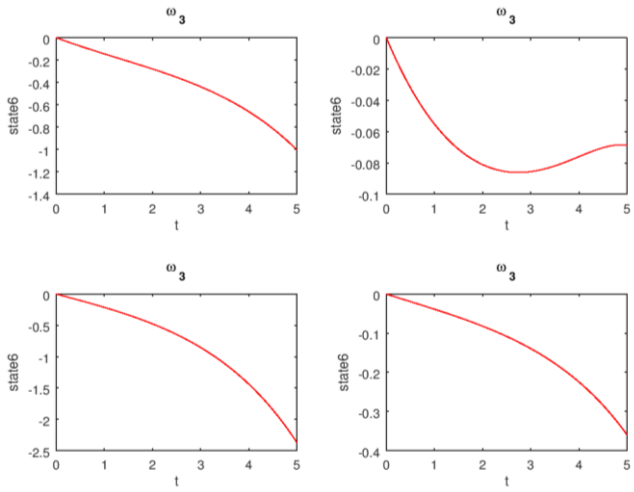


Figure 14. Feasible state6 of the vehicle robot

Figure 15 gives the seventh state of the robot. Such a state function represents the angular velocity of Wheel4. Each graph of state7 function is associated to a given initial condition. Since we have 4 initial conditions then we have 4 state7 computational simulations. The plots all the seven feasible states in the same coordinate systems, where each coordinate system is associated to an initial condition is the following.

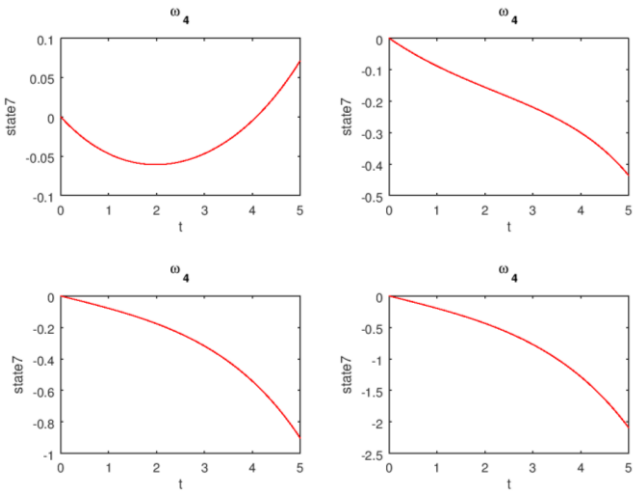


Figure 15. Feasible state7 of the vehicle robot

Figure 16 gives the computational simulations of the combined 7 states of the robot. Each coordinate system graphs are associated to a given initial condition. Since we have 4 initial conditions then we have 4 states computational simulations. In order to confirm the accuracy of the results obtained by using Octave / MATLAB program, additional computer programs are developed in Scilab. From those computer programs, the following are obtained: Figures 17-22 correspond respectively to Figures 3-8. They are plotted to verify and confirm the reliabilities and accuracies of Figures 3-8.

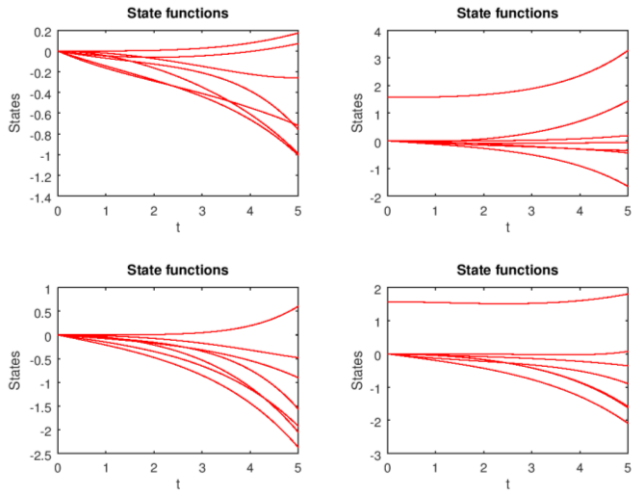


Figure 16. Feasible states of the vehicle robot

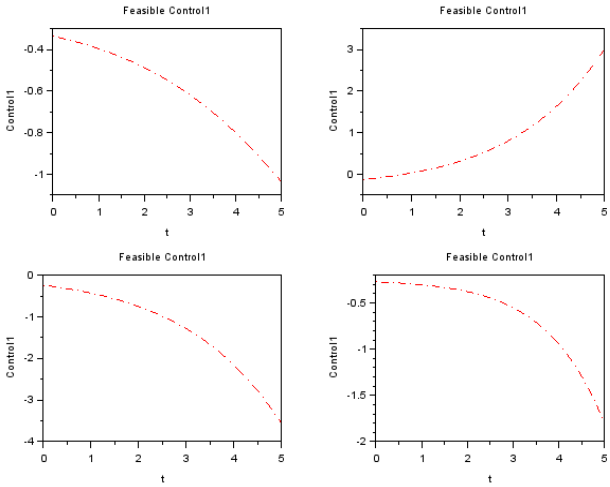


Figure 17. Feasible control strategy 1

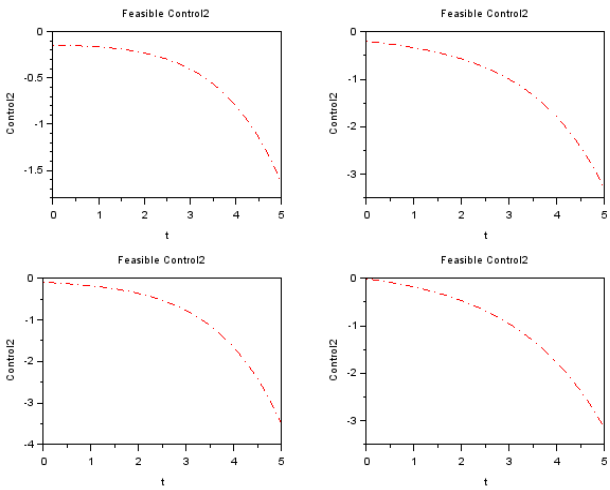


Figure 18. Feasible control strategy 2

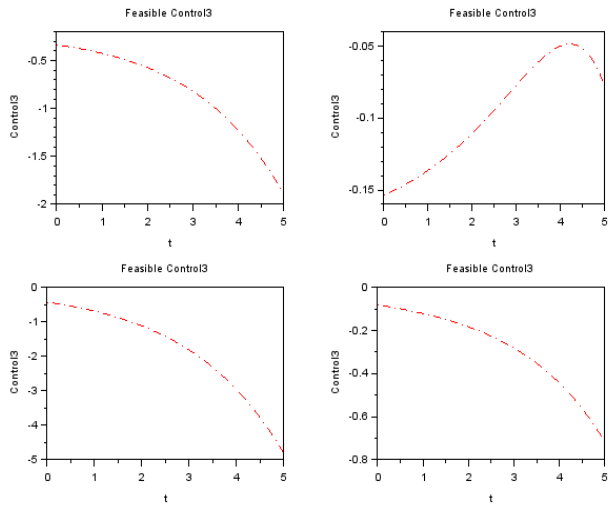


Figure 19. Feasible control strategy 3

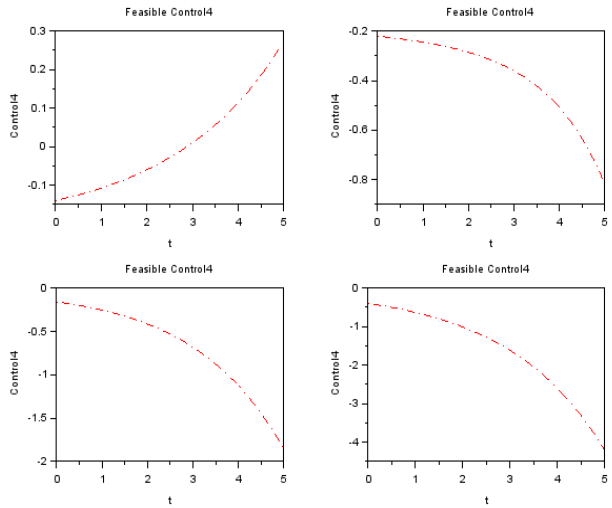


Figure 20. Feasible control strategy 4

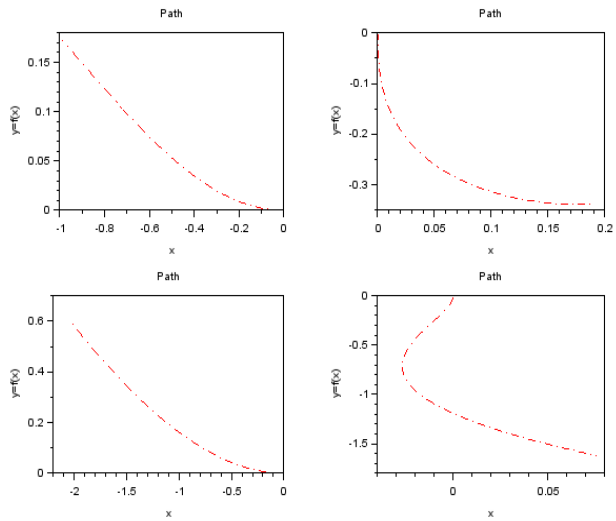


Figure 21. Feasible trajectories of the vehicle robot

To confirm the computational simulations for the states and the costate functions, the reader can follow the same procedure as previously.

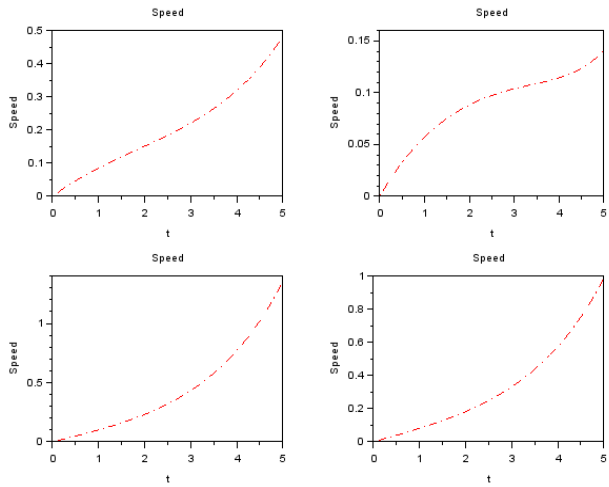


Figure 22. Feasible vehicle robot speeds

8. CONCLUSIONS

The application of Principle of Optimality to solve the Optimal Control Problem of a Four-Wheeled Autonomous Vehicle led to a Nonlinear System of fourteen ODEs involving seven state variables, seven costate variables and four control variables. The substitution of the Feasible Control expressions into the system gave a combined control-free system of state and costate differential equations. An Octave / MATLAB computer program based on the above described fourth-order Runge-Kutta method is developed to solve numerically the system. To confirm the results accuracies, additional computer programs written in scilab are used. The written computer programs are useful to any reader or any researcher who is familiar with programming so that he can replicate or learn more. The results are the Feasible Control Strategies, the Feasible State and Costate functions from which are extracted the Feasible Trajectories and the Feasible Speeds of the autonomous four-wheeled vehicle robot.

ACKNOWLEDGMENT

The authors are grateful to the University of Johannesburg, Faculty of Engineering who funded this research. thank the university for the funding.

REFERENCES

[1] Costa, P., Moreira, A., Gonçalves, J., Lima, J. (2011). Proposal of a new real-time cooperative challenge in mobile robotics. IFAC Proceedings Volumes, 44(1): 9836-9841. https://doi.org/10.3182/20110828-6-IT-1002.02539

[2] Zyada, Z., Hayakawa, Y., Hosoe, S. (2011). Fuzzy nonprehensile manipulation control of a two-rigid-link object by two cooperative arms. IFAC Proceedings Volumes, 44(1): 14614-14621. https://doi.org/10.3182/20110828-6-IT-1002.01337

[3] Wang, C., Cao, M., Xie, G. (2011). Antiphase formation swimming for autonomous robotic fish. IFAC Proceedings Volumes, 44(1): 7830-7835. https://doi.org/10.3182/20110828-6-IT-1002.02524

[4] Grandi, R., Falconi, R., Melchiorri, C. (2011). Unibot remote laboratory: A scalable web-based set-up for education and experimental activities in robotics. IFAC Proceedings Volumes, 44(1): 8521-8526. https://doi.org/10.3182/20110828-6-IT-1002.03103
[5] Schneider, H.C., Wahrburg, J., Roth, H. (2011). Model-aided controller evaluation for a surgical assistance robot to track patient movements in real time. IFAC Proceedings Volumes, 44(1): 10313-10318. https://doi.org/10.3182/20110828-6-IT-1002.00483
[6] Huang, C.N. (2010). The development of self-balancing controller for one-wheeled vehicles. Engineering, 2(4): 212.
[7] Sayyaadi, H., Kouhi, H., Salarieh, H. (2011). Control of car-like (wheeled) multi robots for following and hunting a moving target. Scientia Iranica, 18(4): 950-965. https://doi.org/10.1016/j.scient.2011.07.006
[8] Ata, A., Myo, T. (2008). Optimal trajectory planning and obstacle avoidance for flexible manipulators using generalized pattern research. World Journal of Modelling and Simulation, 4(3): 163-171.
[9] Becerra, V.M. (2008). Optimal control. UK: Scholarpedia.
[10] Chachuat, B. (2007). Nonlinear and dynamic optimization: From theory to practice. Course Notes. Switzerland: EPFL.
[11] Magrab, E.B., Azarm, S. (2000). An Engineer's Guide to MATLAB. Prentice Hall PTR.
[12] Gu, D., Hu, H. (2006). Receding horizon tracking control of wheeled mobile robots. IEEE Transactions on Control Systems Technology, 14(4): 743-749. https://doi.org/10.1109/TCST.2006.872512
[13] Hussein, I.I., Bloch, A.M. (2008). Optimal control of underactuated nonholonomic mechanical systems. IEEE Transactions on Automatic Control, 53(3): 668-682. https://doi.org/10.1109/TAC.2008.919853
[14] Kirk, D.E. (1970). Optimal Control Theory, An Introduction. New-York: Prentice-Hall.
[15] Masoud, S.A., Masoud, A.A. (2000). Constrained motion control using vector potential fields. IEEE Transactions on Systems, Man, And Cybernetics-Part A: Systems and Humans, 30(3): 251-272. https://doi.org/10.1109/3468.844352
[16] Roberts, S.M., Shipman, J.S. (1972). Two-Point Boundary Value Problems, Shooting Methods. Michigan: American Elsevier Pub. Co.
[17] Yang, T.T., Liu, Z.Y., Chen, H., Pei, R. (2008). Formation control and obstacle avoidance for multiple mobile robots. Acta Automatica Sinica, 34(5): 588-592. https://doi.org/10.3724/SP.J.1004.2008.00588
[18] Tsuji, T., Morasso, P.G., Kaneko, M. (1996). Trajectory generation for manipulators based on artificial potential field approach with adjustable temporal behavior. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 438-443. https://doi.org/10.1109/IROS.1996.570811
[19] Klancarc, G. Skrjanc, I. (2007). Tracking-error model-based predictive control for mobile robots in real time. Robotics and Autonomous Systems, 55: 460-469. https://doi.org/10.1016/j.robot.2007.01.002
[20] Krotov, V.F. (1993). Global methods in optimal control theory. In Advances in Nonlinear Dynamics and Control: A Report from Russia. Progress in Systems and Control Theory. Birkhäuser, Boston, MA. https://doi.org/10.1007/978-1-4612-0349-0_3
[21] Kumar, U., Sukavanam, N. (2008). Backstepping based trajectory tracking control of a four wheeled mobile robot. International Journal of Advanced Robotic Systems, 5(4). https://doi.org/10.5772/6224
[22] Lambert, J.D. (1991). Numerical Methods for Ordinary Differential Systems: The Initial Value Problems. John Wiley & Sons, Inc.
[23] Moore, H. (2014). MATLAB for Engineers. Prentice Hall Press.
[24] Zhang, H.S., Wang, H.X., Xu, Y.M., Guo, Z.Y. (2024). Optimization methods rooted in optimal control. Science China Information Sciences, 67: 222208. https://doi.org/10.1007/s11432-024-4207-5
[25] Tyatyushkin, A.I., Zarodnyuk, T.S., Gornov, A.Y. (2019). Algorithms for nonlinear optimal control problems based on the first and second order necessary conditions. Journal of Mathematical Sciences, 239(2): 185-196. https://doi.org/10.1007/s10958-019-04293-y
[26] Buldaev, A.S. (2024). Fixed-point methods in optimization problems for control systems. Journal of Mathematical Sciences, 279(5): 594-606. https://doi.org/10.1007/s10958-024-07041-z

APPENDIX

The combined state and costate nonlinear system of ODEs (49)-(63) is coded by the following Octave / MATLAB function:

```
function dzdt= four_wheel_robot(t,u)
dzdt = zeros(14,1); % Vector Initialization. c1=R;c2=R/L;
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.6; l2=0.6; l3=0.48;
l4=0.48; c1=1; c2=1; c3=1; c4=1;
dudt(1)=k1*(u(4)+u(5)+u(6)+u(7))*cos(u(3)); % Eq. (49)
dudt(2)=k1*(u(4)+u(5)+u(6)+u(7))*sin(u(3)); % Eq. (50)
dudt(3)=k2*(u(4)-u(5)); % Eq. (51)
dudt(4)=-l1*u(4)+l1*(-(l1/c1)*u(11)); % Eq. (52)
dudt(5)=-l2*u(5)+l2*(-(l2/c2)*u(12)); % Eq. (53)
dudt(6)=-l3*u(6)+l3*(-(l3/c3)*u(13)); % Eq. (54)
dudt(7)=-l4*u(7)+l4*(-(l4/c4)*u(14)); % Eq. (55)
dudt(8)=0; % Eq. (56)
dudt(9)=0; % Eq. (57)
dudt(10)=k1*(u(4)+u(5)+u(6)+u(7))*(u(8)*sin(u(3))-
u(9)*cos(u(3))); % Eq. (58)
dudt(11)=-k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))-
k2*u(10)+l1*u(11); % Eq. (59)
dudt(12)=-
k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)+l2*u(12);%
Eq. (60)
dudt(13)=-
k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l3*u(13); % Eq. (61)
dudt(14)=-
k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l4*u(14); % Eq. (62)
```

The above function can be written in a compact form

```
function dudt= four_wheel_robot(t,u)
dudt = zeros(14,1); % Vector Initialization.
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.07; l2=0.65; l3=0.58;
l4=0.87; c1=1; c2=1; c3=1; c4=1;
dudt=[k1*(u(4)+u(5)+u(6)+u(7))*cos(u(3));
      k1*(u(4)+u(5)+u(6)+u(7))*sin(u(3));
```

```
        k2*(u(4)+u(5)+u(6)+u(7));
        -l1*u(4)+l1*(-0.5*(l1/c1)*u(11));
        -l2*u(5)+l2*(-0.5*(l2/c2)*u(12));
        -l3*u(6)+l3*(-0.5*(l3/c3)*u(13));
        -l4*u(7)+l4*(-0.5*(l4/c4)*u(14));
        0;
        0;
        k1*(u(4)+u(5)+u(6)+u(7))*(u(8)*sin(u(3))-
u(9)*cos(u(3)));
        -k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))-
k2*u(10)+l1*u(11);
  -
k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)+l2*u(12);
        -k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l3*u(13);
        -k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l4*u(14))];
```

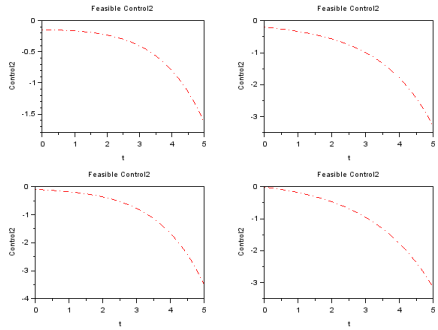
Such a system is solved by using a fourth-order Runge-Kutta numerical method coded by the following Octave / MATLAB function:

```
function [t,u] = runge(fs,t0,tf,N,u0)
%Implementation of a fourth-order Runge-Kutta
numerical method;
h=(tf-t0)./(N-1);%The step size
s=t0:h:tf;%subdivision of the time interval into discrete
subintervals
t=s';% column vector t; u0 is the initial vector solution
u = zeros(N,length(u0));%Each column is a discrete
function;
u(1,:) = u0.'; % The first row of the solution matrix is
given;
for n = 2:N
k1 = feval(fs,t(n-1),u(n-1,:));
k2 = feval(fs,t(n-1)+(h/2),u(n-1,:)+(h/2)*k1');
k3 = feval(fs,t(n-1)+(h/2),u(n-1,:)+(h/2)*k2');
k4 = feval(fs,t(n-1)+h,u(n-1,:)+h*k3');
u(n,:) = u(n-1,:)+(h/6)*(k1'+2*k2'+2*k3'+k4');
end
```

To solve the system of ODEs (defined by the function dudt=four_wheel_robot(t,u)) and then obtain all the state and costate functions, the function [t,u] = runge(fs,t0,tf,N,u0) is called by the following Octave / MATLAB function.

```
function main_four_wheel

clear all
clc
format short
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.6; l2=0.6; l3=0.48;
l4=0.48; c1=1; c2=1; c3=1; c4=1;
%These are Constants of proportionality c1=R;c2=R/L;
t0=0; tf=5; N=501; h=(tf-t0)/(N-1); t=t0:h:tf;
u=zeros(N,14);
u01 =[zeros(7,1);0.96;0.83;0.17;0.56;0.24;0.70;0.29];
u02 =[zeros(2,1);pi/2;zeros(4,1);0.38;0.91;0.35;0.21;0.34;0.32;0.46];
u03 =[zeros(7,1);0.01;0.84;0.01;0.40;0.16;0.89;0.33];
u04 =[zeros(2,1);pi/2;zeros(4,1);0.68;0.08;0.54;0.45;0.02;0.17;0.83];
M=[u01,u02,u03,u04];
t=t';
for k=1:4
  u0=M(:,k);
[t,u]=runge_v2('four_wheel_robot',t0,tf,N,u0);
control1=-(l1/c1)*u(:,11); control2=-(l2/c2)*u(:,12);
control3=-(l3/c3)*u(:,13); control4=-(l4/c4)*u(:,14);
control=[control1,control2,control3,control4];
end
for k=1:4
subplot(2,2,k); plot(t,control(:,1),'r', t,control(:,2),'r',
t,control(:,3),'r', t,control(:,4),'r');
xlabel('t');ylabel('Controls');
end
print
C:\Users\Guest\Documents\8october2024\fourWheel_Rob
ot_Controlss1.png
for k=1:4
subplot(2,2,k);plot(u(:,1),u(:,2),'r-.');
xlabel('x');ylabel('y=f(x)'); title('Feasible Path')% Robot
trajectory
end
print
C:\Users\Guest\Documents\8october2024\fourWheel_Traj
ectory.png
dx= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*cos(u(:,3)); % x
velocity;
dy= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*sin(u(:,3)); % y
velocity;
speed=(dx.^2 + dy.^2).^(0.5);
for k=1:4
subplot(2,2,k);plot(t,speed,'r'); xlabel('t');ylabel('speed');
end
print
C:\Users\Guest\Documents\8october2024\fourWheel_Rob
ot_speed.png
for k=1:4
subplot(2,2,k);plot(t,u(:,1),'r-.',t,u(:,2),'r-.',t,u(:,3),'r-.',t,u(:,
4),'r-.',t,u(:,5),'r-.',t,u(:,6),'r-.', t,u(:,7),'r-.');
xlabel('t');ylabel('States');
end
print
C:\Users\Guest\Documents\8october2024\fourWheel_Rob
ot_States.png
for k=1:4
subplot(2,2,k);plot(t,u(:,8),'r-.',t,u(:,9),'r-.',t,u(:,10),'r-.',t,u(:
,11),'r-.',t,u(:,12),'r-.',t,u(:,13),'r-.',
t,u(:,14),'r-.');xlabel('t');ylabel('Costates');
end
print
C:\Users\Guest\Documents\8october2024\fourWheel_Rob
ot_Costates.png
```



In order to confirm the results (obtained by using Octave / MATLAB program) and convince the reader on their reliability, additional computer programs are developped in Scilab.

The following set of codes can also yield the same above results. Such a set of codes is called four_wheel_Robot1.

```
function dudt= four_wheel_robot1(t,u)
dudt = zeros(14,1); // Vector Initialization.
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.6; l2=0.6; l3=0.48;
l4=0.48; c1=1; c2=1; c3=1; c4=1;
dudt(1)=k1*(u(4)+u(5)+u(6)+u(7))*cos(u(3));
dudt(2)=k1*(u(4)+u(5)+u(6)+u(7))*sin(u(3));
dudt(3)=k2*(u(4)-u(5));
dudt(4)=-l1*u(4)+l1*(-(l1/c1)*u(11));
dudt(5)=-l2*u(5)+l2*(-(l2/c2)*u(12));
dudt(6)=-l3*u(6)+l3*(-(l3/c3)*u(13));
dudt(7)=-l4*u(7)+l4*(-(l4/c4)*u(14));
dudt(8)=0;
dudt(9)=0;
dudt(10)=k1*(u(4)+u(5)+u(6)+u(7))*(u(8)*sin(u(3))-
u(9)*cos(u(3)));
dudt(11)=-k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))-
k2*u(10)+l1*u(11);
dudt(12)=-
k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)+l2*u(12);
dudt(13)=-k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l3*u(13);
dudt(14)=-k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l4*u(14);

endfunction
// u1, u2, u3, u4, u5, u6 and u7 are the state variables,
// u8, u9, u10, u11, u12, u13 and u14 are the costate
variables,
t0=0; tN=5; N=501; h=(tN-t0)/(N-1); t=t0:h:tN;
u=zeros(N,14);
u01 =[zeros(7,1);0.96;0.83;0.17;0.56;0.24;0.70;0.29];
u02 =[zeros(2,1);%pi/2;zeros(4,1);0.38;0.91;0.35;0.21;0.34;0.32;0.46];
u03 =[zeros(7,1);0.01;0.84;0.01;0.40;0.16;0.89;0.33];
u04 =[zeros(2,1);%pi/2;zeros(4,1);0.68;0.08;0.54;0.45;0.02;0.17;0.83];
M=[u01,u02,u03,u04];
t=t';
for k=1:4
u0=M(:,k);
u = ode(u0,t0,t,four_wheel_robot1); t=t '; u=u ';
//subplot(2,2,k);plot(u(:,1),u(:,2), 'r-.');
//xlabel('x');ylabel('y=f(x)'); title('Path')
//end
//xs2png(0,'C:\Users\Guest\Documents\8october2024\
//fourWheel_Trajectory2.png')
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.6; l2=0.6; l3=0.48;
l4=0.48; c1=1; c2=1; c3=1; c4=1;
dx= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*cos(u(:,3));
dy= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*sin(u(:,3));
speed=(dx.^2 + dy.^2).^(0.5);
control1=-(l1/c1)*u(:,11); control2=-(l2/c2)*u(:,12);
control3=-(l3/c3)*u(:,13); control4=-(l4/c4)*u(:,14);
control=[control1,control2,control3,control4];
subplot(2,2,k);plot(t,speed, 'r-.'); xlabel('t');ylabel('Speed');
title('Speed')
end
//xs2png(0,'C:\Users\Guest\Documents\8october2024\
//fourWheel_Trajectory2.png')

subplot(2,2,k); plot(t,control(:,1),'r-.');
xlabel('t');ylabel('Control1'); xtitle('Feasible Control1')
end
xs2png(0,'C:\Users\Guest\Documents\8october2024\
fourWheel_Controlss4.png')

//for k=1:4
subplot(2,2,k); plot(t,control(:,1),'r', t,control(:,2),'r',
t,control(:,3),'r', t,control(:,4),'r');
xlabel('t');ylabel('Controls');
end
xs2png(0,'C:\Users\Guest\Documents\8october2024\
fourWheel_Robot_Controlss2.png')
// All the other functions can be plotted similarly as in
Octave / MATLAB. The reader can execise on that.
```

```
four_wheel_Robot1
clear all
clc
a=0.4; L=1; k1=a/2; k2=a/L; l1=0.07; l2=0.65; l3=0.58;
l4=0.87; c1=1; c2=1; c3=1; c4=1; //These are Constants of
proportionality c1=R;c2=R/L;
deff('[w]=four_wheel_robot2(x,u)',
['u1 = k1*(u(4)+u(5)+u(6)+u(7))*cos(u(3))',
'u2 = k1*(u(4)+u(5)+u(6)+u(7))*sin(u(3))',
'u3 = k2*(u(4)+u(5)+u(6)+u(7))',
'u4 =-l1*u(4)+l1*(-0.5*(l1/c1)*u(11))',
'u5 = -l2*u(5)+l2*(-0.5*(l2/c2)*u(12))',
'u6 = -l3*u(6)+l3*(-0.5*(l3/c3)*u(13))',
'u7 = -l4*u(7)+l4*(-0.5*(l4/c4)*u(14))',
'u8 = 0',
'u9 = 0',
'u10 = -k1*(u(4)+u(5)+u(6)+u(7))*(u(8)*sin(u(3))-
u(9)*cos(u(3)))',
'u11 =-(k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)-
l1*u(11))',
'u12 =-(k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)-
l2*u(12))',
'u13= -(k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+k2*u(10)-
l3*u(13))',
'u14=-(k1*(u(8)*cos(u(3))+u(9)*sin(u(3)))+l4*u(14))',
'w=[u1;u2;u3;u4;u5;u6;u7;u8;u9;u10;u11;u12;u13;u14]']);
 t0=0; tN=5; N=501;h=(tN-t0)/(N-1);t=t0:h:tN;
u=zeros(N,14);
u01 =[zeros(7,1);0.96;0.83;0.17;0.56;0.24;0.70;0.29];
u02 =[zeros(2,1);%pi/2;zeros(4,1);0.38;0.91;0.35;0.21;0.34;0.32;0.46];
u03 =[zeros(7,1);0.01;0.84;0.01;0.40;0.16;0.89;0.33];
u04 =[zeros(2,1);%pi/2;zeros(4,1);0.68;0.08;0.54;0.45;0.02;0.17;0.83];
M=[u01,u02,u03,u04];
t=t';
for k=1:4
   u0=M(:,k);
u = ode(u0,t0,t,four_wheel_robot2); t=t '; u=u ';
control1=-0.5*(l1/c1)*u(:,11); control2=-
0.5*(l2/c2)*u(:,12); control3=-0.5*(l3/c3)*u(:,13);
control4=-0.5*(l4/c4)*u(:,14);
control=[control1,control2,control3,control4];
dx= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*cos(u(:,3));  x
velocity;
dy= k1*(u(:,4)+u(:,5)+u(:,6)+u(:,7)).*sin(u(:,3));  y
velocity;
dtheta=k2*(u(:,4)+u(:,5)+u(:,6)+u(:,7));   Heading angular
velocity;
dDelta=-l1*u(:,4)+l1*(-0.5*(l1/c1)*u(:,11));  Steering
angular %velocity;
dOmega =-l2*u(:,5)+l2*(-0.5*(l2/c2)*u(:,12));  Rate of
change of the heading angular velocity;
dPhi=-l3*u(:,6)+l3*(-0.5*(l3/c3)*u(:,13));  Rate of change
of the steering angular velocity
dudt7=-l4*u(:,7)+l4*(-0.5*(l4/c4)*u(:,14));
speed=(dx.^2 + dy.^2).^(0.5);
end
```