Article

Open-loop Control Strategies and Feasible System Response for a Rear-Axle Bicycle Robot

Masiala Mavungu

Faculty of Engineering and Built Environment, University of Johannesburg, South Africa; masialam@uj.ac.za

"This work was supported by the University of Johannesburg"

Abstract: This paper aims at computing feasible control strategies and the corresponding feasible state trajectories to drive an autonomous rear-axle bicycle robot from a given initial state to a final state such that the total running cost is minimized. Pontryagin's Minimum Principle is applied and derives the optimality conditions from which the feasible control functions, expressed as functions of state and costate variables, are substituted into the combined state-costate system to obtain a new free-control state-costate nonlinear system of ordinary differential equations. A computer program was written in Scilab to solve the combined state-costate system and obtain the feasible state functions, the feasible costate functions and the feasible control functions. Associated Computational Simulations were provided to show the effectiveness and the reliability of the approach.

Keywords: Autonomous Vehicle, Bicycle Robot, Open-loop Control, Path Planning, Differential Equation, Initial Value Problem, Runge-Kutta Method, Scientific Computing.

1. Introduction

Most of vehicle crash investigations performed by some country transport departments certified that vehicle accidents are mostly caused by drivers who sometimes, when driving long distances, get tired and lose control, and then cannot manage the business of obstacle avoidance properly. The use of autonomous vehicles in general allows to reduce transport costs and can, in fast ways, avoid obstacles and then prevent accidents and thus save lives. Path planning of autonomous bicycles and study of their performances are subject to equilibrium constraints and create a lot of research questions, business and research opportunities. The study and the control of the dynamics of autonomous bicycles have become current and major concerns for the industrial operators and managers, and then attract the attention of research engineers, research mathematicians, research physicists, research computer scientists, etc. who develop suitable strategies, tools and methods to tackle and address all the problems associated to autonomous bicycles. There exists a considerable number of works on autonomous bicycles carried out since some years. Paper [1] developed a bicycle robot with balancing control. Paper [2] controlled the balance of a two-wheeled vehicle. Paper [3] controlled a bicycle to illustrate the control of nonholonomic and under-actuated systems with symmetry. Paper [4] designed and controlled a ball riding bicycle robot. Paper [5] proposed a dynamical model for a bicycle. Book [6] presents a different and unified perspective of the mechanics, planning and control of robots. The authors present the state of the art, the serew theoretic techniques which capture the most salient features of a robot in an intuitive way. Book [7] provides fundamental notions on mobile robots, navigation, localization, manipulator-arm kinematics, dynamics, and point level control as well as camera modelling, image processing, feature extraction and multi-view geometry. Paper

[8] developed and analysed the kinematic and the dynamic models for a three wheeled bicycle called pedicab. Paper [9] presented modelling and control strategies for an autonomous three wheeled mobile robot with front wheel steer and front wheel drive. Book [10] provides a summary of common vehicle control systems and the dynamic models which are used to develop those control systems. All the above-mentioned papers are closely connected with the main subject of this paper because they deal in one way or another with control. To the best of my knowledge, none of those articles considered the control of an autonomous bicycle robot where the desired point is the rear Axle and uses optimal control theory methods. None of them computed the optimal state of the system, the corresponding costate functions and optimal speed . Generally it is noticeable that for other articles, before applying optimal control theory, the system is first reduced into a simpler one so that its characteristics can be determined easily. Most of time, the state system is linearized with respect to an equilibrium point, and inthose papers certain parameters are set in simpler values. The simplifications are performed to easily apply certain operations such as designing a quadratic regulator. However, when a nonlinear control system is reduced into a linear one, if no strategy is adopted, there will be loss of information, and then the final results will be considerably and significantly different from the exact ones. In this paper, the control system is highly nonlinear and has been studied as it is. No other paper has addressed the minimization of the running energy. No paper has computed the optimality conditions. No article has performed the derivation of a costate system of ordinary differential equations adjoining the state system. In the literature no paper found as considering the impact of the size of the initial costate vector on the length of the trajectory. So, most of the above-cited papers and others not cited addressed the bicycle stability problem, the tracking problem, etc., without considering the minimization of the running costs. This article is based on the control of the kinematics of a bicycle robot where the reference point is the center of the rear wheel. This paper uses optimal control theory to compute feasible control strategies of an autonomous bicycle and feasible state trajectories such that the bicycle running cost is minimize. The main contributions of this paper are the design and computation of two feasible control strategies, the computation of the solution of the combined state-costate system of ordinary differential equations which is defined by six feasible state functions representing the feasible system responses to reference commands, six costate (adjoint) functions. To perform computation, scilab computer programs were developed and applied.

This paper is organized as follows: Section 2 develops different relevant mathematical models and formulates the problem as an optimal control problem. Section3 computes the system Hamiltonian, derives and solves the normal equations of optimality. Section 4 applies Pontryagin's Minimum Principle to derive all the relevant nonlinear ordinary differential equations. Section 5 develops relevant computer programs in Scilab to compute the feasible control trajectories, the corresponding state and costate (adjoint) trajectories, the bicycle speed function and all the other outputs. Section 6 designs the associated computational simulations.

2-. Mathematical Models

2.1-. Control System, Kinematic Model

The reference point for the motion is at the centre of the rear-axle of the bicycle. The motion of an autonomous bicycle on the horizontal plane is modelled as follows:

$$\frac{dx}{dt} = c_1 \omega cos(\theta) \qquad (1)$$

$$\frac{dy}{dt} = c_1 \omega sin(\theta) \quad (2)$$
$$\frac{d\theta}{dt} = c_2 \omega tan(\delta) \quad (3)$$
$$\frac{d\delta}{dt} = c_3 \varphi \quad (4)$$

where $c_1 = R$ and $c_2 = R/L$ are constant of proportionality, R and L are respectively the radius of the wheels and the distance between the centre of the rear and the front wheels. (x, y) are the coordinates of the projection of the rear-axle's center (which is the reference point for the motion) on the horizontal plane, θ is the heading angle, δ is the steering angle, φ is the steering angle's velocity.

The reference commands which regulate the bicycle angular velocity and the steering angle velocity are modelled as solution to a closed-loop system defined by:

$$\frac{d\omega}{dt} = -a_1\omega + a_1\tau_1 \quad (5)$$
$$\frac{d\varphi}{dt} = -a_2\varphi + a_2\tau_2 \quad (6)$$

τ_1 and τ_2 are the unknown input control variables to be developed. a_1 and a_2 are constants of proportionality.

The whole robot kinematic control system is

$$\frac{dx}{dt} = c_1 \omega cos(\theta) \quad (7)$$
$$\frac{dy}{dt} = c_1 \omega sin(\theta) \quad (8)$$
$$\frac{d\theta}{dt} = c_2 \omega tan(\delta) \quad (9)$$
$$\frac{d\delta}{dt} = c_3 \varphi \quad (10)$$
$$\frac{d\omega}{dt} = -a_1\omega + a_1\tau_1 \quad (11)$$
$$\frac{d\varphi}{dt} = -a_2\varphi + a_2\tau_2 \quad (12)$$

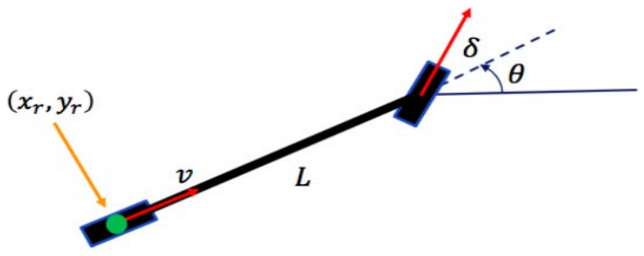

Figure 1. Rear-Axle Bicycle Robot Geometric Model.

2.2-. *Objective functional*

In this paper, the total running cost to be minimized is as follows:

$$J(\boldsymbol{u}) = J(\tau_1, \tau_2) = \int_{t_0}^{t_f} ({\tau_1}^2 + {\tau_2}^2) dt \quad (13)$$

where t_0 and t_f are respectively the bicycle motion's starting and final times, τ_1 and τ_2 are the reference commands which control respectively the bicycle angular velocity and steering angle velocity. ${\tau_1}^2 + {\tau_2}^2$ is called cost rate.

2.3. *Problem Formulation*

This paper addresses the following problem: Compute the feasible control strategies and the associated feasible state functions, also called feasible robot system responses, to bring the autonomous bicycle robot from a given initial state to a final state such that the total running cost is minimized.

3-. Hamiltonian and Feasible Controls

The Hamiltonian of the system is given by

$H(t,\boldsymbol{Y}(t),\boldsymbol{\alpha}(t),\omega(t),\varphi(t)) = h(t)+\sum_{k=1}^{6}\alpha_k\,(t)f_k(\boldsymbol{Y}(t),\omega(t),\varphi(t))$ (14)

Where we have

$h(t)=\tau_1{}^2+\tau_2{}^2$ is the energy cost rate,

$f_1(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = c_1\omega cos(\theta)$ is the x component of the linear velocity of the bicycle,

$f_2(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = c_1\omega sin(\theta)$ is the y component of the linear velocity of the bicycle,

$f_3(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = c_2\omega tan(\delta)$ is the rate of change of the heading angle of the bicycle,

$f_4(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = c_3\varphi$ is the rate of change of the steering angle of the bicycle,

$f_5(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = -a_1\omega+a_1\tau_1$ is the rate of change of the bicycle angular velocity,

$f_6(\boldsymbol{Y}(t),\omega(t),\varphi(t)) = -a_2\varphi+a_2\tau_2$ is the rate of change of the velocity of steering angle.

$\boldsymbol{Y}(t)=(x(t),y(t),\theta(t),\delta(t),\omega(t),\varphi(t))$ is the unkown state vector function. It is built to store all the state functions.

$\boldsymbol{\alpha}(t)=(\alpha_1(t),\alpha_2(t),\alpha_3(t),\alpha_4(t),\alpha_5(t),\alpha_6(t))$ is the unkown costate (adjoint) vector function.

It is built to store all the costate functions. Concerning the control functions, the normal equations for optimality are as follows:

$\frac{\partial H}{\partial \tau_1} = 2\tau_1{}^* + a_1\alpha_5 = 0$ (15)

$\frac{\partial H}{\partial \tau_2} = 2\tau_2{}^* + a_2\alpha_6 = 0$ (16)

The feasible control functions are given by

$\tau_1{}^* = -0.5a_1\alpha_5$ (17)

$\tau_2{}^* = -0.5a_2\alpha_6$ (18)

4-. Normal Condition for Optimality

If $\boldsymbol{u}^*=(\tau_1{}^*,\tau_2{}^*)$ is the optimal control of the above problem and $\boldsymbol{Y}^*=(x^*,y^*,\theta^*,\delta^*,\omega^*,\varphi^*)$ the corresponding optimal system response to the input control $\boldsymbol{u}^*=(\tau_1{}^*,\tau_2{}^*)$, then there exists a costate vector function $\boldsymbol{\alpha}^*=(\alpha_1{}^*,\alpha_2{}^*,\alpha_3{}^*,\alpha_4{}^*,\alpha_5{}^*,\alpha_6{}^*)$, also called adjoint vector such that the following are satisfied:

$J(\boldsymbol{u}^*)\le J(\boldsymbol{u})$ (19)

$\frac{dx^*}{dt} = c_1\omega^* cos(\theta^*)$ (20)

$\frac{dy^*}{dt} = c_1\omega^* sin(\theta^*)$ (21)

$\frac{d\theta^*}{dt} = c_2\omega^* tan(\delta^*)$ (22)

$\frac{d\delta^*}{dt} = c_3\varphi^*$ (23)

$\frac{d\omega^*}{dt} = -a_1\omega^*+a_1\tau_1{}^*$ (24)

$\frac{d\varphi^*}{dt} = -a_2\varphi^*+a_2\tau_2{}^*$ (25)

$\frac{d\alpha_1{}^*}{dt} = 0$ (26)

$\frac{d\alpha_2{}^*}{dt} = 0$ (27)

$\frac{d\alpha_3{}^*}{dt} = c_1\omega^*(\alpha_1{}^*\sin(\theta^*)-\alpha_2{}^*\cos(\theta^*))$ (28)

$\frac{d\alpha_4{}^*}{dt} = -c_2\alpha_3{}^*\omega^* sec^2(\delta^*)$ (29)

$\frac{d\alpha_5{}^*}{dt} = -c_1\big(\alpha_1{}^* cos(\theta^*)-\alpha_2{}^* sin(\theta^*)\big)+a_1\alpha_5{}^*$ (30)

$\frac{d\alpha_6{}^*}{dt} = -c_3\alpha_4{}^*+a_2\alpha_6{}^*$ (31)

By letting $\boldsymbol{u}^* = (\tau_1{}^*, \tau_2{}^*)$, with $\tau_1{}^* = -0.5a_1\alpha_5{}^*$ and $\tau_2{}^* = -0.5a_2\alpha_6{}^*$ for the control functions,

$z_1{}^* = x^*$, $z_2{}^* = y^*$, $z_3{}^* = \theta^*$, $z_4{}^* = \delta^*$, $z_5{}^* = \omega^*$ and $z_6{}^* = \varphi^*$ for the state functions,

$z_7{}^* = \alpha_1{}^*$, $z_8{}^* = \alpha_2{}^*$, $z_9{}^* = \alpha_3{}^*$, $z_{10}{}^* = \alpha_4{}^*$, $z_{11}{}^* = \alpha_5{}^*$ and $z_{12}{}^* = \alpha_6{}^*$ for the costate functions

and by combining all the state and costate functions into a vector $\boldsymbol{z} = [\boldsymbol{Y}, \boldsymbol{\alpha}]$, then the combined state-costate system can be rewritten as

$$\frac{dz_1{}^*}{dt} = c_1 z_5{}^* cos(z_3{}^*) \qquad (32)$$

$$\frac{dz_2{}^*}{dt} = c_1 z_5{}^* sin(z_3{}^*) \qquad (33)$$

$$\frac{dz_3{}^*}{dt} = c_2 z_5{}^* tan(z_4{}^*) \qquad (34)$$

$$\frac{dz_4{}^*}{dt} = c_3 z_6{}^* \qquad (35)$$

$$\frac{dz_5{}^*}{dt} = -a_1 z_5{}^* + a_1\tau_1 \qquad (36)$$

$$\frac{dz_6{}^*}{dt} = -a_2 z_6{}^* + a_2\tau_2 \qquad (37)$$

$$\frac{dz_7{}^*}{dt} = 0 \qquad (38)$$

$$\frac{dz_8{}^*}{dt} = 0 \qquad (39)$$

$$\frac{dz_9{}^*}{dt} = c_1 z_5{}^*(z_7{}^*\sin(z_3{}^*) - z_8{}^*\cos(z_3{}^*)) \quad (40)$$

$$\frac{dz_{10}{}^*}{dt} = -c_2 z_9{}^* z_5{}^* sec^2(z_4{}^*) \qquad (41)$$

$$\frac{dz_{11}{}^*}{dt} = -c_1\left(z_7{}^* cos(z_3{}^*) - z_8{}^* sin(z_3{}^*)\right) + a_1 z_{11}{}^* \quad (42)$$

$$\frac{dz_{12}{}^*}{dt} = -c_3 z_{10}{}^* + a_2 z_{12}{}^* \quad (43)$$

Vector $\boldsymbol{z}$ is built to vectorize the combined state and costate system of equations so that the system can be analyzed and processed in a easy and rigorous manner.

Notice that in the system (32)-(43) the right hand side and its gradient are continuous. Therefore, the solution exists and is unique.

5-. Computer Program

Pontryagin's Minimum Principle applied in this article involves a costate system of ordinary differential equations which combined to the state system and the feasible controls gives a combined free-control state-costate system of ordinary differential equations. I order to solve any system of ordinary differential equations which is an initial value problem, I developed (in Matlab) an algorithm coding a fourth-order Runge-Kutta method. Such an algorithm can be translated into many other programming languages. It is as follows:

```
function [t,y] = runge_v2(fs,t0,tf,N,y0)
h=(tf-t0)./(N-1); // h is the step size for the discretization.
t=t0:h:tf; // t is the time vector.
t=t';
y = zeros(N,length(y0)); // y0 is the initial vector solution. y is initialized to zero.
y(1,:) = y0.'; // The solution at the starting time..
for n = 2:N
k1 = feval(fs,t(n-1),y(n-1,:));
k2 = feval(fs,t(n-1)+(h/2),y(n-1,:)+(h/2)*k1');
k3 = feval(fs,t(n-1)+(h/2),y(n-1,:)+(h/2)*k2');
k4 = feval(fs,t(n-1)+h,y(n-1,:)+h*k3');
y(n,:) = y(n-1,:)+(h/6)*(k1'+2*k2'+2*k3'+k4');
end
```

The above algorithm can be used to solve any initial value problem. Let's use it to solve the system (32)-(43) of equations. By translating and developing the algorithm into Scilab codes we have the following set of codes:

```
clear all
clc; // To clear the screen
c1=2; c2=1; c3=0.75; a1=0.25; a2=0.25; // These are Constants of proportionality c1=R; c2=R/L;
// R is the radius of each wheel; L is the distance between the front and the rear wheels.
t0=0; // t0 is the initial time of the motion;
tf=6; //    tf is the final time of the motion;
N=601; // N is the number of discrete point;
h=(tf-t0)/(N-1);// Step size.
t=t0:h:tf; // vector of discrete times
z=zeros(N,12); // Initialization of z
z(1,:)= [zeros(1,6),ones(1,6)]; // Good
k=zeros(1,12);
for n = 2:N
// For the 1st ordinary differential equation, equation (32), we have the following codes
k(1,1)=c1*z(n-1,5)*cos(z(n-1,3));
k(2,1)= c1*(z(n-1,5) +(h/2)* k(1,1))*cos(z(n-1,3) +(h/2)*k(1,1)); // The program continues
k(3,1)= c1*(z(n-1,5) +(h/2)* k(2,1))*cos(z(n-1,3) +(h/2)*k(2,1));
k(4,1)= c1*(z(n-1,5) +(h/2)* k(3,1))*cos(z(n-1,3) +(h/2)*k(3,1));
z(n,1)=z(n-1,1)+(h/6)*(k(1,1)+ 2*(k(2,1)+k(3,1))+k(4,1));
// For the 2nd ordinary differential equation, equation (33), we have the following codes
k(1,2)=c1*z(n-1,5)*sin(z(n-1,3));
k(2,2)= c1*(z(n-1,5)+(h/2)*k(1,1))*sin(z(n-1,3)+(h/2)*k(1,2));
k(3,2)= c1*(z(n-1,5)+(h/2)*k(2,2))*sin(z(n-1,3)+(h/2)*k(2,2));
k(4,2)= c1*(z(n-1,5)+(h/2)*k(3,2))*sin(z(n-1,3)+(h/2)*k(3,2));
z(n,2)=z(n-1,1)+(h/6)*(k(1,2)+2*(k(2,2)+k(3,2))+k(4,2));
```

```
// For the 3rd ordinary differential equation, equation (34), , we have the following codes
k(1,3)=c2*z(n-1,5)*tan(z(n-1,4));
k(2,3)= c2*(z(n-1,5)+(h/2)*k(1,3))*tan(z(n-1,4)+(h/2)*k(1,3));
k(3,3)= c2*(z(n-1,5)+(h/2)*k(2,3))*tan(z(n-1,4)+(h/2)*k(2,3));
k(4,3)= c2*(z(n-1,5)+(h/2)*k(3,3))*tan(z(n-1,4)+(h/2)*k(3,3));
z(n,3)=z(n-1,3)+(h/6)*(k(1,3)+2*(k(2,3)+k(3,3))+k(4,3));
// For the 4th ordinary differential equation, equation (35), we have the following codes
k(1,4) = c3*z(n-1,6);
k(2,4)= c3*(z(n-1,6)+(h/2)*k(1,4));
k(3,4)= c3*(z(n-1,6)+(h/2)*k(2,4));
k(4,4)= c3*(z(n-1,6)+(h/2)*k(3,4));
z(n,4)=z(n-1,4)+(h/6)*(k(1,4)+ 2*(k(2,4)+k(3,4))+k(4,4));
// For the 5th ordinary differential equation, equation (36), we have the following codes
k(1,5)= a1*(-0.5*a1*z(n-1,11)-z(n-1,5));
k(2,5)= a1*(-0.5*a1*(z(n-1,11)+(h/2)*k(1,5))-(z(n-1,5)+(h/2)*k(1,5)));
k(3,5)= a1*(-0.5*a1*(z(n-1,11)+(h/2)*k(2,5))-(z(n-1,5)+(h/2)*k(2,5)));
k(4,5)= a1*(-0.5*a1*(z(n-1,11)+(h/2)*k(3,5))-(z(n-1,5)+(h/2)*k(3,5)));
z(n,5)=z(n-1,5)+(h/6)*(k(1,5)+2*(k(2,5)+k(3,5))+k(4,5));
// For the 6th ordinary differential equation, equation (37), we have the following codes
k(1,6)=a2*(-0.5*a2*z(n-1,12)-z(n-1,6));
k(2,6)=a2*(-0.5*a2*(z(n-1,12)+(h/2)*k(1,6))-(z(n-1,6)+(h/2)*k(1,6)));
k(3,6)=a2*(-0.5*a2*(z(n-1,12)+(h/2)*k(2,6))-(z(n-1,6)+(h/2)*k(2,6)));
k(4,6)=a2*(-0.5*a2*(z(n-1,12)+(h/2)*k(3,6))-(z(n-1,6)+(h/2)*k(3,6)));
z(n,6)=z(n-1,6)+(h/6)*(k(1,6)+ 2*(k(2,6)+k(3,6))+k(4,6));
// For the 7th ordinary differential equation, equation (38), we have the following codes
k(1,7)=0;
k(2,7)=(h/2)*k(1,7);
k(3,7)=(h/2)*k(2,7);
k(4,7)=(h/2)*k(3,7);
z(n,7)=z(n-1,7)+(h/6)*(k(1,7)+ 2*(k(2,7)+k(3,7))+k(4,7));
// For the 8th ordinary differential equation, equation (39), we have the following codes
k(1,8)=0;
k(2,8)=(h/2)*k(1,8);
k(3,8)=(h/2)*k(2,8);
k(4,8)=(h/2)*k(3,8);
z(n,8)=z(n-1,8)+(h/6)*(k(1,8)+ 2*(k(2,8)+k(3,8))+k(4,8));
// For the 9th ordinary differential equation, equation (40), we have the following codes
k(1,9)=c1*z(n-1,5)*(z(n-1,7)*sin(z(n-1,3))- z(n-1,8)*cos(z(n-1,3)));
k(2,9)=c1*(z(n-1,5)+(h/2)*k(1,9))*((z(n-1,7)+(h/2)*k(1,9))*sin(z(n-1,3)+(h/2)*k(1,9))-(z(n-1,8)+(h/2)*k(1,9))*cos(z(n-
1,3)+(h/2)*k(1,9)));
k(3,9)=c1*(z(n-1,5)+(h/2)*k(2,9))*((z(n-1,7)+(h/2)*k(2,9))*sin(z(n-1,3)+(h/2)*k(2,9))-(z(n-1,8)+(h/2)*k(2,9))*cos(z(n-
1,3)+(h/2)*k(2,9)));
```

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 January 2023 doi:10.20944/preprints202301.0527.v1

```
k(4,9)=c1*(z(n-1,5)+(h/2)*k(3,9))*((z(n-1,7)+(h/2)*k(3,9))*sin(z(n-1,3)+(h/2)*k(3,9))-(z(n-1,8)+(h/2)*k(1,9))*cos(z(n-
1,3)+(h/2)*k(3,9)));
z(n,9)=z(n-1,9)+(h/6)*(k(1,9)+2*(k(2,9)+k(3,9))+k(4,9));
// For the 10th ordinary differential equation, equation (41), we have the following codes
k(1,10)= -c2*z(n-1,5)*z(n-1,9)*((sec(z(n-1,4)))^2);
k(2,10)= -c2*(z(n-1,5)+(h/2)*k(1,10))*(z(n-1,9)+(h/2)*k(1,10))*((sec(z(n-1,4) +(h/2)*k(1,10)))^2);
k(3,10)= -c2*(z(n-1,5)+(h/2)*k(2,10))*(z(n-1,9)+(h/2)*k(2,10))*((sec(z(n-1,4) +(h/2)*k(2,10)))^2);
k(4,10)= -c2*(z(n-1,5)+(h/2)*k(3,10))*(z(n-1,9)+(h/2)*k(3,10))*((sec(z(n-1,4) +(h/2)*k(3,10)))^2);
z(n,10)=z(n-1,10)+(h/6)*(k(1,10)+2*(k(2,10)+k(3,10))+k(4,10));
// For the 11th ordinary differential equation, equation (42) , we have the following codes
k(1,11)=-c1*(z(n-1,7)*cos(z(n-1,3))- z(n-1,8)*sin(z(n-1,3)))+a1*z(n-1,11);
k(2,11)=-c1*((z(n-1,7)+(h/2)*k(1,11))*cos(z(n-1,3)+(h/2)*k(1,11))-(z(n-1,8)+(h/2)*k(1,11))*sin(z(n-1,3)+(h/2)*k(1,11))
)+a1*(z(n-1,11)+(h/2)*k(1,11));
k(3,11)=-c1*((z(n-1,7)+(h/2)*k(2,11))*cos(z(n-1,3)+(h/2)*k(2,11))-(z(n-1,8)+(h/2)*k(2,11))*sin(z(n-1,3)+(h/2)*k(2,11))
)+a1*(z(n-1,11)+(h/2)*k(2,11));
k(4,11)=-c1*((z(n-1,7)+(h/2)*k(3,11))*cos(z(n-1,3)+(h/2)*k(3,11))-(z(n-1,8)+(h/2)*k(3,11))*sin(z(n-1,3)+(h/2)*k(3,11))
)+a1*(z(n-1,11)+(h/2)*k(3,11));
z(n,11)=z(n-1,11)+(h/6)*(k(1,11)+2*(k(2,11)+k(3,11))+k(4,11));
// For the 12th ordinary differential equation, equation (43), we have the following codes
k(1,12)=-c3*z(n-1,10)+a2*z(n-1,12);
k(2,12)=-c3*(z(n-1,10)+(h/2)*k(1,12))+a2*(z(n-1,12)+(h/2)*k(1,12));
k(3,12)=-c3*(z(n-1,10)+(h/2)*k(2,12))+a2*(z(n-1,12)+(h/2)*k(2,12));
k(4,12)=-c3*(z(n-1,10)+(h/2)*k(3,12))+a2*(z(n-1,12)+(h/2)*k(3,12));
z(n,12)=z(n-1,12)+(h/6)*(k(1,12)+2*(k(2,12)+k(3,12))+k(4,12));
end
t=t';
// The program continues
control1=-0.5*a1*z(:,11); control2=-0.5*a2*z(:,12); control=[ control1, control2];
dx= c1*z(:,5).*cos(z(:,3)); // x component of the velocity
dy= c1*z(:,5).*sin(z(:,3)); // y component of the velocity
dTheta=c2*z(:,5).*tan(z(:,4)); // Heading angular velocity
dDelta= c3*z(:,6); // Steering angular velocity
dOmega = a1*(-0.5*a1*z(:,11)-z(:,5)); // Rate of change of the heading angular velocity
dPhi= a2*(-0.5*a2*z(:,12)-z(:,6)); // Rate of change of the steering angular velocity
Ham=z(:,7).*( c1*z(:,5).*cos(z(:,3))) + z(:,8).*(c1*z(:,5).*sin(z(3))) + z(:,9).*(c2*z(:,5).*tan(z(:,4))) + z(:,10).*(c3*z(:,6))
+ z(:,11).*(a1*(-0.5*a1*z(11)-z(5)))+z(:,12).*(a2*(-0.5*a2*z(12)-z(6)));
// Optimal trajectory
scf(0);
plot(z(:,1),z(:,2), 'r'); xlabel('x (in meters) ');ylabel('y=f(x) (in meters)');
z(1,:)=[zeros(1,6),rand(1,6)]
distance=sqrt(max(z(:,1))^2 + max(z(:,2))^2)
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicycletrajectory.png')
```

```
// First 3 state functions
scf(0);
subplot(3,1,1);plot(t,z(:,1), 'r');xlabel('Time t in seconds');ylabel('State 1 ');
subplot(3,1,2);plot(t,z(:,2) , 'r');xlabel('Time t in seconds ');ylabel('State 2 ');
subplot(3,1,3);plot(t,z(:,3) , 'r');xlabel('Time t in seconds ');ylabel('State 3 ');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicyclefirst3states.png ')
// Last 3 state functions
scf(0);
subplot(3,1,1);plot(t,z(:,4) , 'r');xlabel('Time t in seconds ');ylabel('State 4');
subplot(3,1,2);plot(t,z(:,5) , 'r');xlabel('Time t in seconds ');ylabel('State 5');
subplot(3,1,3);plot(t,z(:,6) , 'r');xlabel('Time t in seconds ');ylabel('State 6');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicyclelast3States.png ')
// First 3 costate functions
scf(0);
subplot(3,1,1);plot(t,z(:,7) , 'r');xlabel('Time t in seconds ');ylabel('Costate 1');
subplot(3,1,2);plot(t,z(:,8) , 'r');xlabel('Time t in seconds ');ylabel('Costate 2');
subplot(3,1,3);plot(t,z(:,9) , 'r');xlabel('Time t in seconds ');ylabel('Costate 3');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicyclefirst3Costates.png ')
// Last 3 costate functions;      // The program continues
scf(0);
subplot(3,1,1);plot(t,z(:,10) , 'r'); xlabel('Time t in seconds ');ylabel('Costate 4');
subplot(3,1,2);plot(t,z(:,11) , 'r'); xlabel('Time t in seconds ');ylabel('Costate 5');
subplot(3,1,3);plot(t,z(:,12) , 'r'); xlabel('Time t in seconds ');ylabel('Costate 6');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicyclelast3Costates.png ')
// Control strategies
scf(0);
subplot(2,1,1);plot(t,control1, 'r');xlabel('Time t in seconds ');ylabel('Control1');
subplot(2,1,2);plot(t,control2, 'r');xlabel('Time t in seconds ');ylabel('Control2');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicycleControls.png ')
// Velocities
scf(0);
subplot(3,1,1);plot(t,dx, 'r'); xlabel('Time t in seconds ');ylabel('Linear velocity along x axis');
subplot(3,1,2);plot(t,dy, 'r'); xlabel('Time t in seconds ');ylabel('Linear velocity along y axis');
subplot(3,1,3);plot(t, c1*z(:,5) , 'r'); xlabel('Time t in seconds ');ylabel('Robot speed');
xs2png(0,'C:\Users\Guest\Documents\17january2023\bicyclex_velocity.png ');
 // The program ends
```

6-. Computational simulations

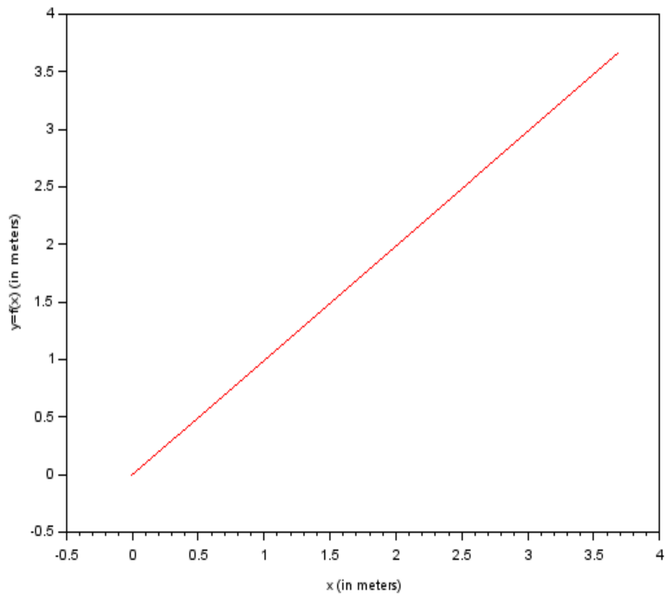

Figure 2. Feasible Bicycle Robot Trajectory.

Figure 2 gives the feasible trajectory of the bicycle robot. The starting point of the path is (0,0).

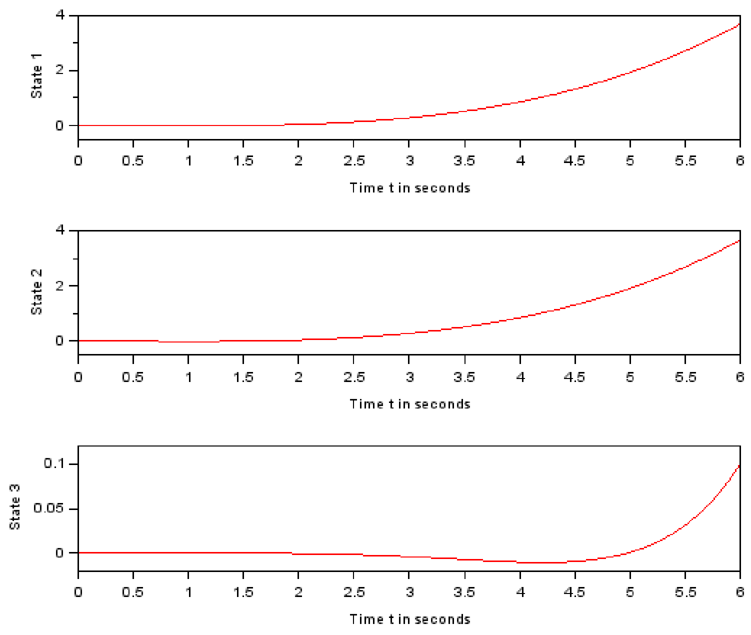

FIGURE 3. First three feasible state functions.

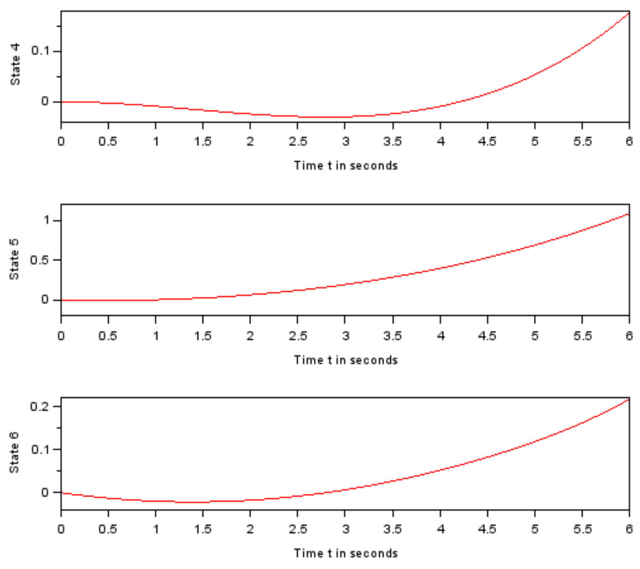

FIGURE 4. Last three feasible state functions.

The above plotted function state1 describes the instantaneous robot's position in the x direction. In other words we have state1 $= x(t),\ t\ \in [0,6]$. The above plotted function state2 describes the instantaneous robot's position in the y direction. In other words we have state2 $= y(t),\ t\ \in [0,6]$. The above plotted function state3 describes the instantaneous robot's heading angle. In other words we have state3 $= \theta(t),\ t\ \in [0,6]$. The above plotted function state4 describes the instantaneous steering angle. state4 $= \delta(t),\ t\ \in [0,6]$. The above plotted function state5 describes the instantaneous angular velocity. state5 $= \omega(t),\ t\ \in [0,6]$. The above plotted function state6 describes the instantaneous steering angle's velocity. state6 $= \varphi(t),\ t\ \in [0,6]$.

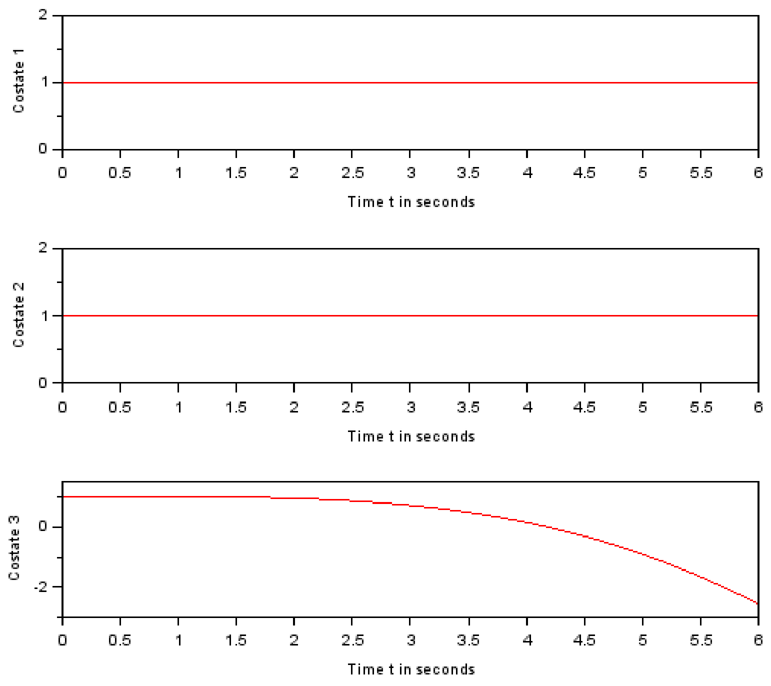

Figure 5. First three feasible costate functions.

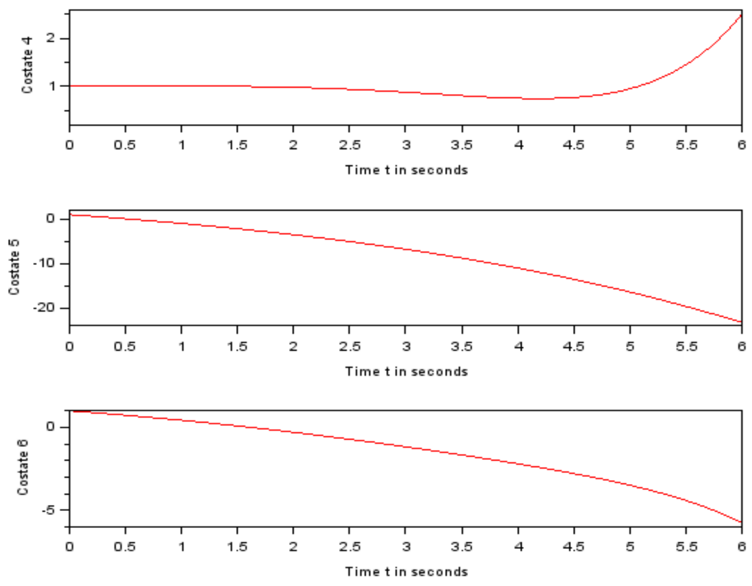

Figure 6. Last three feasible costate functions.

The function $costate1$ is the adjoint function to the function state1. $costate1 = \alpha_1(t),\ t \in [0,6]$. The function $costate2$ is the adjoint function to the function state2. $costate2 = \alpha_2(t),\ t \in [0,6]$. The function $costate3$ is the adjoint function to the function state3. $costate3 = \alpha_3(t),\ t \in [0,6]$. The function $costate4$ is the adjoint function to the function state4. $costate4 = \alpha_4(t),\ t \in [0,6]$.

The function $costate5$ is the adjoint function to the function state5. $costate5 = \alpha_5(t),\ t \in [0,6]$. The function $costate6$ is the adjoint function to the function state4. $costate6 = \alpha_6(t),\ t \in [0,6]$.

One can notice that $\alpha_1(t)$ and $\alpha_2(t)$ are constant because their time derivatives are zeros.

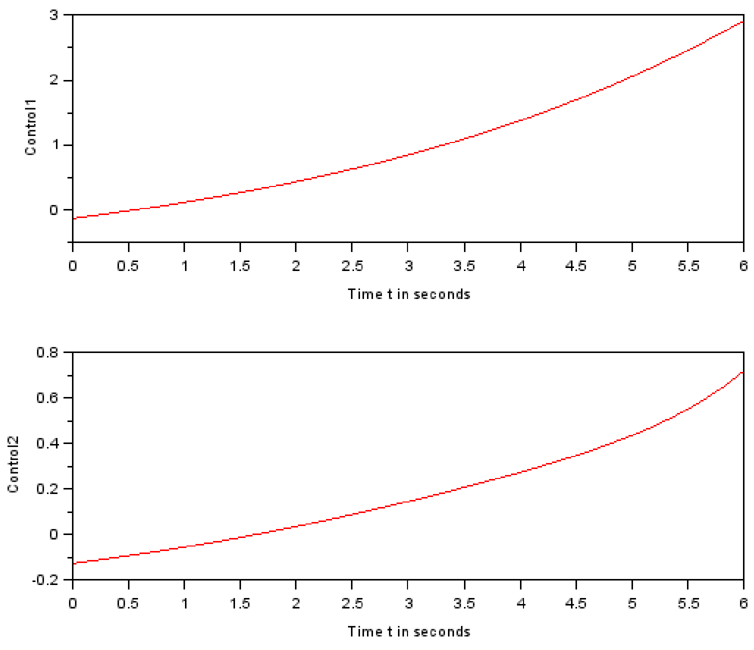

FIGURE 7. Feasible control strategies.

The function $control1$ describes the instantaneous reference command to the robot angular velocity. In other words we have $1 = \tau_1(t),\ t \in [0,6]$. The function $control1$ describes the instantaneous reference command to the robot angular velocity. In other words we have $2 = \tau_2(t),\ t \in [0,6]$.

7-. Conclusion

The aim of this paper was to compute the feasible control strategies and the associated feasible state functions, also called feasible robot system response to control, for an autonomous rear-axle bicycle robot to bring it from a given initial state to a final state such that the total running cost is minimized. A fourth-order Runge-Kutta was used to solve the combined free-control state-costate system of ordinary differential equations obtained from Pontryagin's Minimum Principle. The obtained results enable to predict the performance of the autonomous bicycle robot so that it can be controlled accurately and efficiently. The computer programs are useful to any reader or any researcher who is familiar with programming and who would want to learn more. Computational Simulations are provided to show the effectiveness and the reliability of the approach. In future, control policies will be developed for the rear-axle bicycle robot to asymptotically track a prescribed trajectory. Some methodology will be performed to develop optimal control strategies using Lagrange Interpolating polynomial.

Funding: This research was funded by the University of Johannesburg, Faculty of Engineering through UJ-GES.

Data Availability Statement: This study did not report any data.

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

References

1- Anan Suebsomran, Balancing Control of Bicycle Robot (2012), 2012 IEEE International Conference on cyber technology in Automation, control and Intelligent Systems(Cyber), pp. 69-73.

2- Ngoc Kien Vu., Hong Quang Nguyen, Balancing control of a Two-wheeled bicycle (2020), Mathematical Problems in Engineering. Published by Hindawi.

3- Neil H. Getz, Jerrold E. Marsden, Control for an Autonomous Vehicle, Department of Electrical Engineering, University of California. Berkeley, CA 94720, getz@eecs.berkeley.edu.

4- Pongsakorn Seekkao and Manukid Parnichkun, Development and control of a bicycle robot based on steering and pendulum balancing, Mechatronics, Volume 69, August 2020.

5- Y. Tanaka and T. Murakami, Self Sustaining Bicycle Robot with Steering Controller, Advanced Control, 2004 IEEE International Workshop on Advanced Control, 27-31 may 2012.

6- Lynch, Kevin M., and franck C. Park, Modern Robotics: Mechanics, Planning, and Control 1st edition, Cambridge Press, 2017.

7- Corke, Peter I. Robotics, Vision and Control: Fundamental Algorithms in Matlab, Springer, 2011.

8- Noverina, Nur Hamid, Grafika Jati, M. Anwar Ma'sum, Wisnu Jatmiko, Kinematics and Dynamics Analysis of an Autonomous Three-Wheeled Bicycle Modeling, 2019 4th Asia-Pacific Conference on Intelligent Robot Systems (Conference Paper), July 2019.

9- A. Pandey, S. Jha, and D. Chakravarty, Modeling and Control of an autonomous three-wheeled mobile robot with front steer, Proceeding 2017 1st Conference on Robot and Computing. IRC 2017, pp. 136-142, 2017.

10- R. Rajamani, Vehicle Dynamics and Control, vol. Spec No. 2001.