Control of a Brachiating Robot for Inspection of Aerial Power Lines

Vinicius Menezes de Oliveira, and Walter Fetter Lages, Member, IEEE

Abstract—The present work addresses the problem of real-time predictive control of a brachiating robot. The robot is constrained (underactuated, limited torque) and a multivariable system, which implies a very difficult problem due to the large amount of on-line computation that is required. Previous works demonstrated that it is not possible to consider the nonlinear model-based MPC under real-time constraints. To overcome this problem we present a linearized model-based MPC, which is able to be handled under real-time constraints.

Index Terms—Brachiating Robot, Predictive Control, Real-Time Systems.

I. INTRODUCTION

This paper presents a brachiating robot designed for inspection of aerial power lines. Brachiating is the type of motion used by apes. By moving like an ape the robot is able to overcome obstacles on the power line, such as supporting insulators. The focus of this work is on the development of a control strategy for such a robot.

The kinematic model of the robot is derived by extending the Denavit-Hartenberg convention to handle mechanisms with bifurcation in the kinematic chain, such as shown in figure 1. The dynamic model of the robot was derived by using the Lagrange-Euler formalism. The analysis of the properties of the models showed that this type of mechanism are such that the development of a controller to force it to track a reference path is a challenging task. Such a system is underactuated and is non-holonomic in the second degree.

A system is underactuated if the number of its control inputs (\( n_u \)) is less than the number of its degrees-of-freedom (\( n_{df} \)). A property of such systems is that they can not be linearized by state feedback, hence the well-known computed torque technique can not be used.

Non-holonomic systems are those subject to restrictions that are not integrable. Those restrictions are said to be of the first degree if they depend only on the generalized coordinates of position (\( q \)) and velocity (and eventually on the time), and therefore can be written in the form \( a(q, \dot{q}) = 0 \). The restrictions are said to be of the second degree if they depend on the generalized coordinates of position, velocity and acceleration (and on the time), and can be written as \( a(q, \dot{q}, \ddot{q}) = 0 \). Underactuated articulated robots such as the brachiating robot used in this work have non-holonomic restrictions of the second degree.

Note that not every restriction in the form \( f(q, \dot{q}, \ddot{q}) = 0 \) is a non-holonomic restriction. If it can be completely integrated to obtain a restriction in the form \( g(q, t) = 0 \), then it is a holonomic restriction. Otherwise, it is a non-holonomic restriction.

It is a well-known consequence of a theorem from Brockett, that a non-holonomic system can not be stabilized to a point by a smooth time-invariant feedback. Therefore, control laws for such systems are usually time-varying or non-smooth or the problem is redefined as a trajectory tracking problem, which can be solved by time-invariant smooth feedback. However, for underactuated non-holonomic systems the conditions on which the trajectory tracking problem can be solved by time-invariant smooth feedback are still unknown and is still an open research question.

Thus, the control method for a brachiating robot should be able to consider the characteristics of under-actuation and the non-holonomic restrictions the robot is subject to. Furthermore, there are constraints on the available torque for the motors and physical limits of joints.

Many approaches have been proposed for the control of underactuated robots, specially for underactuated manipulators [1]–[3]. Most control schemes that have been proposed in the literature to control brachiating robots are limited in the way the robot executes the brachiation motion and/or are not able to deal with constraints on the state and/or control variables. In this work a control scheme based on the predictive control strategy with receding horizon is presented. The method can take into account such constraints while computing the control input.

The first paper introducing the problem of control of brachiating robots is [4], which considers a simple brachiating robot with two links. In that work, a heuristic method for generating feasible trajectories is proposed.

Fukuda, Hasegawa, Shimojima and Saito developed a self-tuning reinforced learning algorithm to generate feasible trajectories with properties of robustness to some perturbations, while Saito has include a feedback controller to increase the robustness of the system [5]. Those works do not consider the use a mathematical model for the dynamics of the robot for the learning process. The main drawback of that method is the requirement for a long training period (about 200 experiments) for each robot configuration.

Nakanishi, Fukuda and Koditschek proposed another approach by considering a target dynamics for the control of underactuated systems [6]. The dynamics approach is a variant of the standard plant inversion methods. Authors inspired themselves on biomechanics, by defining the dynamic task...
from a simpler target dynamics and by systematically using a reverse temporal symmetry. However, a precise mathematical model for the robot dynamics is needed.

In the sequence, Hasegawa and Fukuda proposed a brachiating robot with 7 links. That robot is a redundant system and is able to execute complex motions, similar to a real ape, in a bi-dimensional plane [7]. In that work, authors introduced a control architecture to cope with the many input and output variables. The behavior-based approach was used to reduce the number of degrees-of-freedom to be considered, in order to facilitate the design process by decomposing in simple behaviors and coordinating them together. The problem with this strategy is the tuning of the controller whenever a change in the robot or in the task is needed.

In [8] a control strategy for brachiation based on local behavior control is presented while another strategy, based on energy control for the swinging phase is presented in [9], where a simplified model with 4 links is used. The objective is to inject the minimum energy in the robot between the swinging and displacement phases. In [10] another method based on Passive Dynamic Autonomous Control (PDAC) is presented. The central idea is to develop a energy efficient method.

In the current paper a dynamic model for the brachiating robot is proposed and a linearized model is developed, which enables the further development of a feedback control that can be implemented in real-time. The proposed control method is the Model-based Predictive Control.

Model predictive control (MPC) is based on on-line optimization, generating an implicit control law. Its application to brachiating robots is promising since under-actuation and non-holonomity are handled in a straightforward way by the optimization procedure inside the MPC.

The controller is based on linearizing the robot model around the reference at each sampling time, thus enabling the use of quadratic programming for the optimization. The quadratic programming problem is a convex one, and therefore the existence of a single solution is granted and the required computing power is much less then in the nonlinear case. However, due to the complexity of the dynamic model of the brachiating robot, it is not easy to linearize it by analytical methods. Our solution is to use the least-squares method to identify a linearized model on-line, thus also endowing the system with adaptive capabilities as time-varying parameters are tracked by the identification. Even thought this controller is conceptually more involved, it can be more easily implemented in real-time since the optimization step is much less demanding on computing resources.

By using the proposed control methods, the robot is able to execute different types of brachiation (just one cycle or continuously motion, with under-swing and over-hand motion) along the supporting line. The type of motion is handled automatically by the controller. By just defining the admissible range for each joint, the controller is able to interpret the restrictions and generate another type of brachiation motion. That capability is not seen in other types of controllers that has been proposed for brachiating robots. Another desired feature that is implicitly generated by the controller is the recovering whenever the robot is not able to grasp the line in a first attempt. The objective function forces the controller to compute a control sequence such that the robot executes a swing motion to acquire enough energy to move upwards and try to grasp the line again in order to complete the brachiation motion. It is important to note that no additional information is given to the controller. It just detects that the goal was not satisfied and generates a control signal to complete the task that was initially proposed.

II. Dynamic Model

The underactuated brachiating robot considered in this work is shown in Fig. 1. This robot has two arms and a body where the payload can be allocated. Each arm has a gripper to grasp the supporting line enabling the robot to execute its motion. The robot moves by releasing the gripper of one arm and grasping the supporting line in a forward point. It is important to note that although there are three joints, only two of them (joints 2 and 3) are actuated.

The robot coordinate systems (see Fig. 1) are determined accordingly to the Denavit-Hartenberg convention. By considering the brachiating robot an open chain manipulator, its dynamics can be given by [11]:

$$M(\theta)\ddot{\theta}(t) + V(\theta, \dot{\theta}) + G(\theta) = B\tau - F_v$$  \hspace{1cm} (1)

where $\theta$ are the joint variables, $M(\theta)$ is the inertia matrix, $V(\theta, \dot{\theta})$ is the vector of Coriolis and centrifugal terms, $G(\theta)$ is the vector or gravity forces, $\tau$ is the input torque, $F_v$ are the friction forces and $B$ is the control input selection matrix, defined as:

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$  \hspace{1cm} (2)

Hence, the robot dynamics (1) can be rewritten, in a proper state-space form as:

$$\dot{x} = f(x, \tau)$$  \hspace{1cm} (3)

where $x = [\theta \ \dot{\theta}]^T$ is the state vector, $\tau$ is the vector of system inputs and

$$f(x, \tau) = \dot{\theta} - \left[ (M(\theta))^{-1}(B\tau - F_v - V(\theta, \dot{\theta}) - G(\theta)) \right]$$  \hspace{1cm} (4)
III. LINEARIZED MPC

In an earlier paper [12], an MPC strategy which uses the non-linear model of the system dynamics for the prediction horizon was presented. However, that approach is too demanding with respect to computational resources, since a non-convex optimization problem has to be solved on-line. Furthermore, the problem presents a large number of decision variables and therefore, its not easy to find a global minimum solution. Typically, the solution is a sub-optimal one, based on a local minimum.

In this section a version of MPC that uses a linearized model for the system dynamics is introduced. This linearization is performed on an error model between the (pre-computed) robot reference trajectory and the actual robot trajectory, as shown in Fig. 2.

The method is based on successive linearization, which enables a description of the non-linear system through a time-varying linear system. By considering the control inputs as decision variables, it is possible to recast the optimization problem to be solved at each sampling time in a quadratic programming (QP) problem. For QP problems, there are many robust numerical methods which can find a global optimal solution.

Consider, in a first moment, the existence of a virtual reference robot, which dynamics is similar to the real one and which will be used to generate the reference trajectory. Then, the dynamics of the virtual reference system is given by:

\[
\dot{x}_r = f_r(x_r, \tau_r)
\]

where \(x_r\) is the reference state vector and \(\tau_r\) is the reference input torque vector.

The linearized system dynamics can be computed by expanding in a Taylor series around the point \((x_r, \tau_r)\) the right hand side of (3) and discarding the higher order terms to obtain:

\[
\dot{x} = f(x, \tau) + \frac{\partial f(x, \tau)}{\partial x} \bigg|_{x=x_r, \tau=\tau_r} (x-x_r) + \frac{\partial f(x, \tau)}{\partial \tau} \bigg|_{x=x_r, \tau=\tau_r} (\tau-\tau_r)
\]

or

\[
\dot{x} = f(x, \tau_r) + f_x(x-x_r) + f_\tau(\tau-\tau_r)
\]

where \(f_x, f_{\tau}\) are the Jacobians of \(f\) with respect to \(x\) and \(\tau\), respectively, computed at the reference point \((x_r, \tau_r)\).

Now, by subtracting (5) from (7) yields:

\[
\dot{x} = f_x,x\dot{x} + f_\tau,\tau \ddot{\tau}
\]

where \(\ddot{x} \triangleq x-x_r\) is the reference tracking error and \(\ddot{\tau} \triangleq \tau-\tau_r\) is the error associated to the control input.

The discretization of (8) can be obtained by the forward differences method, resulting in:

\[
\ddot{x}(k+1) = A(k)\ddot{x}(k) + B(k)\ddot{\tau}(k)
\]

with

\[
A(k) = I_{n \times n} + T \ast h_x,\tau(k)
\]

\[
B(k) = T \ast h_\tau,\tau(k)
\]

By considering the error linear system (9), it becomes possible to recast the optimization problem in a linear optimization problem, hence solvable by quadratic programming. Therefore define the following vectors:

\[
\pi(k+1) = \begin{bmatrix} \ddot{x}(k+1) \\ \ddot{x}(k+2) \\ \vdots \\ \ddot{x}(k+N) \end{bmatrix}, \quad \pi(k) = \begin{bmatrix} \ddot{\tau}(k) \\ \ddot{\tau}(k+1) \\ \vdots \\ \ddot{\tau}(k+N-1) \end{bmatrix}
\]

The objective function can be written as:

\[
\Phi(k) = \pi^T(k+1)Q \pi(k+1) + \pi^T(k)R \pi(k)
\]

with

\[
Q = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & Q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q \end{bmatrix}, \quad R = \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R \end{bmatrix}
\]

It is possible to rewrite (9) in the form:

\[
\pi(k+1) = A(k)\ddot{x}(k) + B(k)\ddot{\tau}(k)
\]

where \(A\) and \(B\) are given by:

\[
A(k) = \begin{bmatrix} A(k) \\ A(k+1) \ast A(k) \\ \vdots \\ A(k, 2, 0) \\ A(k, 1, 0) \end{bmatrix}
\]

\[
B(k) = \begin{bmatrix} B(k) \\ A(k+1) \ast B(k) \\ \vdots \\ A(k, 2, 0) \ast B(k) \\ A(k, 1, 0) \ast B(k) \end{bmatrix}
\]

with

\[
\alpha(k, j, l) = \prod_{i=N-j}^{i} A(k+1) \]

From (12) and (13), after some algebraic manipulation, it is possible to rewrite (12) as:

\[
\Phi(k) = \frac{1}{2} \pi^T H(k) + \pi(k) + f^T(k)\pi(k) + d(k)
\]
The quadratic term is described by the positive definite Hessian matrix $H(k)$, the linear part is described by the vector $f(k)$ and the term $d(k)$, which is independent of $\tau$, can be neglected for the optimization problem. Then, the objective function can be redefined as:

$$\Psi(k) = \frac{1}{2} \tau^T H(k) + \tau(k) + f^T(k) \tau(k)$$

which is in the standard form used by most QP solvers. Hence, the optimization problem to be solved at each sampling time is given by:

$$\hat{u}^* = \arg \min_{\hat{u}} \Psi(k)$$

subject to:

$$D\hat{u}(k+j|k) \leq d, \ j \in [0, N-1]$$

The amplitude restrictions on the control variables (20) can be rewritten in the form:

$$\tau_{\text{min}} - \tau_r(k+j) \leq \bar{\tau}(k+j) \leq \tau_{\text{max}} - \tau_r(k+j)$$

The complete control diagram is shown in Fig. 3.

The controller parameters are:

$$N = 5$$

$$Q = \begin{bmatrix} 10I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} \end{bmatrix}$$

$$R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

The Fig. 4 shows the Cartesian position of the robot along a brachiation motion and the reference trajectory. The reference torque and the torque applied to the robot, for the joints 1 and 2 can be seen in 5 and in Fig. 6, respectively.

For all simulations, the robot has the following initial position:

$$x_0 = \begin{bmatrix} -3\pi/4 \\ -\pi/4 \\ 0 \\ 0 \end{bmatrix}$$

The angular joint positions and their references, for each joint, are shown in Fig. 7, Fig. 8 and Fig. 9.
Fig. 6. Torque applied to joint 2 (red) and reference torque for joint 2 (green). Both curves are coincident.

Fig. 7. Joint 1 angular position (red) and reference angular position for joint 1 (green).

Fig. 8. Joint 2 angular position (red) and reference angular position for joint 2 (green).

Fig. 9. Joint 3 angular position (red) and reference angular position for joint 3 (green).

Fig. 10. Time used to compute the control signal (linearized MPC).

Fig. 11. Time used to compute the control signal (non-linear MPC [15]).

Forward, in Fig. 10 the time need to compute the control signal is shown. It can be seen that the required time for the method presented here, by using the linearized model, is much smaller than the time required to compute the control with the non-linear model as in [15], shown in Fig. 11. Therefore, since the required computation time is smaller than the sampling period (10 ms), it is possible to use this control law in a real-time controller.
This work presented the characterization of the dynamics of a brachiating robot and the development of a closed-loop control strategy which can directly deal with the underactuation properties of the robot. The proposed control method uses an on-line optimization of an underactuated system, showing that it is indeed possible to solve the optimization problem, which was overcame by considering a linearized error model for the system, computed at each sampling time.

The results show that the control approach proposed in the work is promising. Although an expensive computational cost, the timing results show that it is possible to use this method in real-time. The measured timings for the computation of the control signal were smaller than the sampling time of the system, showing that it is indeed possible to solve the optimization problem, and hence compute the control signal in real-time.

In general, it is possible to state that the control method proposed in this work can make the robot to successfully perform the brachiation motion while avoiding the violation of system constraints on states and control inputs.

ACKNOWLEDGMENT

The first author thanks to Prof. Toshio Fukuda for the comments about the topic of this work while he had and internship at Fukuda’s Lab. at the Department of Micro-Nano Systems of the University of Nagoya.

The authors would like to thank to Companhia Estadual de Energia Elétrica (CEE), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and Conselho Nacional de Pesquisa (CNPq) for the financial support.

REFERENCES


Vinicius Menezes de Oliveira received the B.Sc. in Computer Engineering from the Fundação Universidade Federal do Rio Grande (FURG), Rio Grande, Brazil, in 1998, the M.Sc. degree in Electrical Engineering from the Universidade Federal de Santa Catarina (UFSC), Florianópolis, Brazil, in 2001 and the D.Eng. degree from the Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil in 2008. Currently, Dr. de Oliveira is working at the Center for Computational Sciences at the Federal University of Rio Grande, Rio Grande, Brazil.

Walter Fetter Lages (S’91, M’99) received the B.Sc. in Electrical Engineering from Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Porto Alegre, Brazil in 1989 and the M.Sc. and D.Sc. degrees in Electronic and Computer Engineering from Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, Brazil in 1993 and 1998, respectively. From 1997 to 1999 he was an Adjunct Professor in the Physics Department of the Fundação Universidade Federal do Rio Grande (FURG), Rio Grande Brazil. Currently he is an Associate Professor in the Electrical Engineering Department of the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil. Dr. Lages is a member of IEEE, ACM, Brazilian Automation Society and Brazilian Computer Society.