Project Report

ED5314: Design, Analysis and Control of Robot Manipulators

R Gautham, ED14B015
Akhil Sathuluri, ED14B037

November 29, 2017

1 Inverse Kinematics

The robot model given below is taken and assumed that the robot is in its quadruple stance phase. It is nothing but a phase in which all the legs are on the ground and the robot is not performing any walking or running gaits. In such a case the body movement is achieved by legs actuations in parallel. Hence can be assumed as a parallel manipulator for analysis.

The aim of this part was to make the robot orient itself in such a way that it keeps looking at a given point in space. For this, any given point in space is taken and two rotations are attributed to it. One leg is taken into consideration and a code is written to find out the end point given the DH parameters of it.

One global co-ordinate system was chosen at the point where the center of the body plate of the robot is located and one local axes at the same location, which moves along with the particle where it is fixed in space. All the ends of the legs are fixed at some known positions on the ground. As the point moves the rotation matrix is computed and all the hip joints are now computed in the global frame.

Now the code is used to compute inverse kinematic solutions of the leg for the corresponding hip joint location. A suitable branch is chosen and the data is stored.

This data is later plotted for visualization.

1.1 Kinematic Model

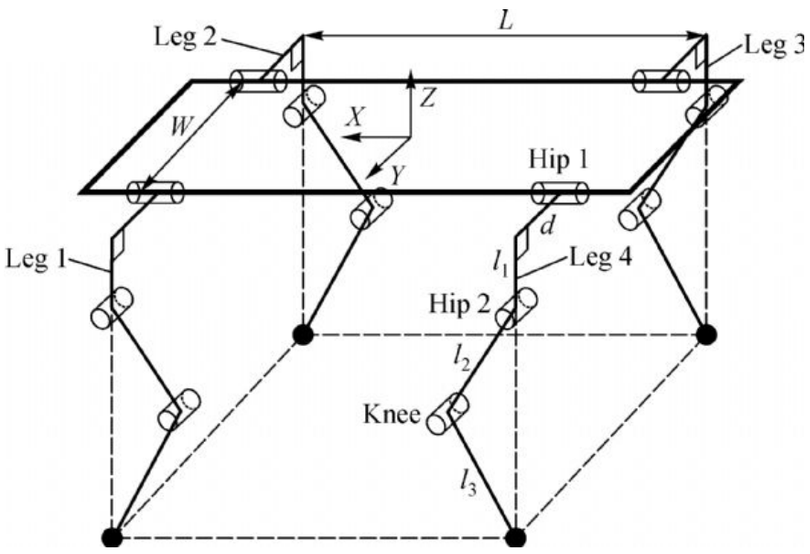


Figure 1: Kinematic Model of the robot[1]

The above model of the leg was used for formulating the DH parameters. Its architecture is, ground constraint is spherical followed by a 2-R dyad and a rotary joint connecting the leg and the body. Nomenclature of the joints used in the code, the second rotary joint from the ground is knee joint, followed by two joints each perpendicular to the other. This is the hip joint and allows the in plane motion and the out of plane motion of the leg. The total number of degrees of freedom are 6 for this quadruped in a quadruple stance phase.

2 Control of Four Bar

The idea was to explore various optimal control strategies for the robot to change its position from sitting to standing configuration. But the dynamics of a six bar has turned out to be very complicated and so we started with a four bar to see if normal control strategies would give any better solutions.
The aim was to balance a ball on the body of the quadruped.
The following control strategies were explored.

2.1 PID Control

A non-linear dynamic model was formulated based on Lagrangian method. A separate dynamic equation was also formulated for the ball. Two situations with two different PID values are illustrated below. The feedback was taken for the ball and the coupler angle.

Case-1: PID was used for the ball position and PD was used for coupler angle.
Kp = -7.1; Kd = -120; Ki = -9.9; kc1 = 100; kcd = -4.9

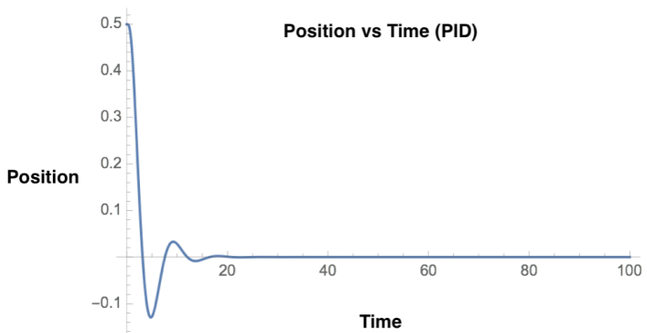


Figure 2: Position (y-axis) vs loop variable(x-axis)

Case-2: Only PID was used for the ball position
Kp = -0.1 Kd = -120 Ki = -2.9

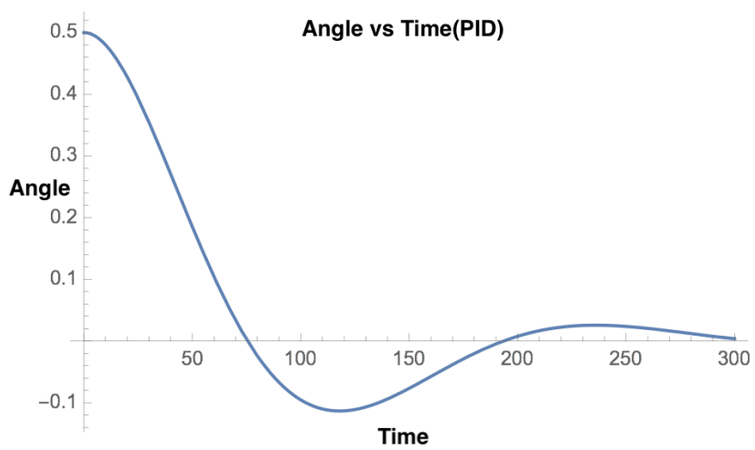


Figure 3: Position (y-axis) vs loop variable(x-axis)

2.2 Model Predictive Control

Since the model is simple, model predictive control is tested for the same problem.
Model predictive control is one of the latest control algorithm developed and widely used in process industries. This control algorithm heavily relies on the model and by incorporating a proper state estimator in the loop, the controller can give very good performance in comparison with other traditional controllers. Unlike PID control which is widely used for linear systems, MPC performs better despite model being non-linear. In recent advances

with computing power MPC can easily be implemented in robotics. With lesser number of tuning parameters any ad-hoc tuning which is the case in PID is completely eliminated. In this case an MPC is implemented to balance a ball in the center of a four bar mechanism. The model was linearized about the stable operating point and was then used to drive the ball to the center. A great advantage of MPC is it uses the dynamics to predict the future values and appropriately changes its current control values so as to reduce any deviations from the actual trajectory.

One major advantage in MPC is its ability to handle constraints. Non-linearity of the actuators often causes problems when a PID is used due to saturation. In MPC actuator limits can be preset and saturation is prevented. The aggressiveness of the controller can easily be tweaked by modifying the control parameters.

Parametric MPCs remove the need for a high performance computer. This is done by computing the optimization off line and achieving a state based control strategy. Hence the corresponding gain values are retrieved from memory whenever required.

The prediction horizon and control horizon values are 15 and 8 respectively.

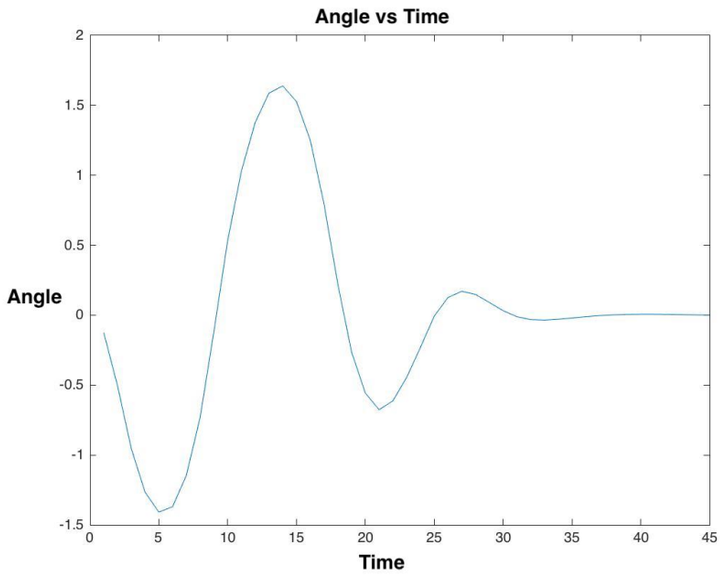


Figure 4: Coupler angle(y-axis) vs loop variable(x-axis)

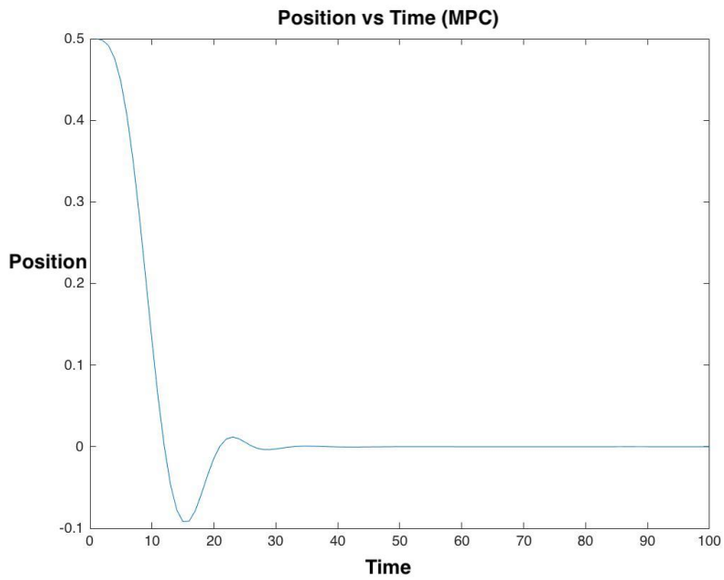


Figure 5: Ball position(y-axis) vs (x-axis)

3 Optimal Control

Given the end conditions of the manipulator, the controller should come up with control algorithm that minimizes the time taken by the manipulator to reach the endpoint incorporating the inherent constraints in the actuators. The control algorithm heavily depends on the model and may become very complicated if the dynamics become non-linear. Due to this fact solving Optimal control problems can be broadly divided into two sub methods, Direct and Indirect.

Direct Methods:

Direct methods use PMP (Pontryagins Maximum Principle). The optimization problem is reduced to solve a two point boundary value problem. This may become very complicated if the dynamics are non-linear. Several methods like Leap-frog, Simple Shooting, Multiple - Shooting can be used to solve such problems. Though the computation involved is intensive the results obtained are very accurate and results in a global optimal value rather than local optimum.

Indirect Methods:

Indirect Methods convert the entire problem into one huge Non-Linear Program and runs the optimizer. The trajectory can be parametrized using several spline algorithms . Generally a first order polynomial is used to interpolate control histories and a third order polynomial to interpolate the states.

3.1 Time Optimal Control

Since the dynamics of the robot in actuator space is non-linear, a feedback linearized model was used. The model obtained by feedback linearization is a simple double integrator. Time optimal control of double integrators are well researched topics. The following SIMULINK model was created to get the trajectories of the angles.

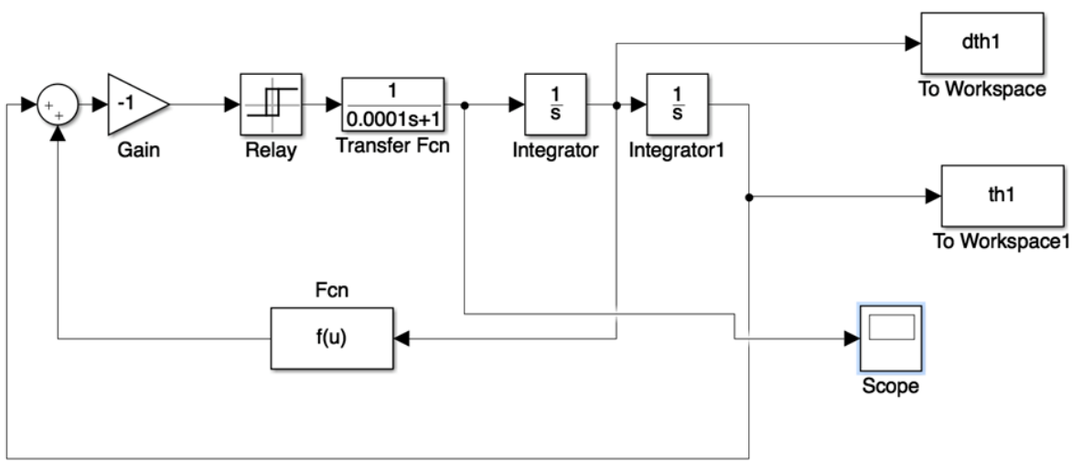
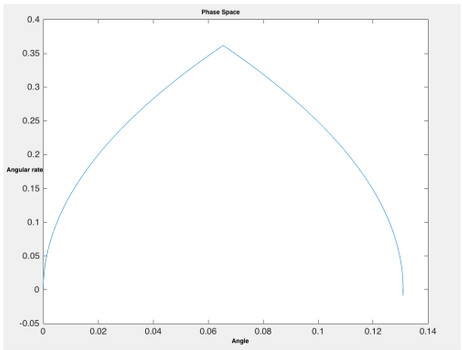
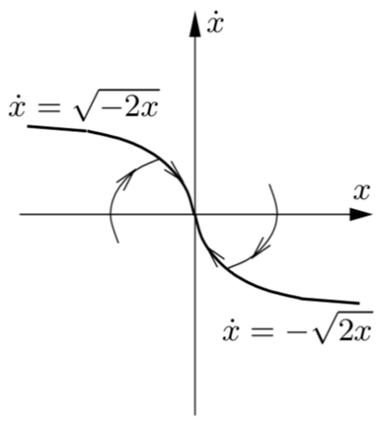


Figure 6: Simulink model for Time Optimal Control

Since the control is bang-bang in nature it results in a sharp change in the velocity, this resulted in abrupt change in the angle values which led to a discontinuity in passive joint angles. In order to overcome this problem a transfer function was created to smoothen the transition. The corresponding values of active and passive variable are plotted below.



(a) Phase portrait of the states



(b) switching function of a double integrator

Figure 7: Comparing the generated trajectory vs an example

The obtained solution is tested for gain type singularity by plotting the ϕ values.

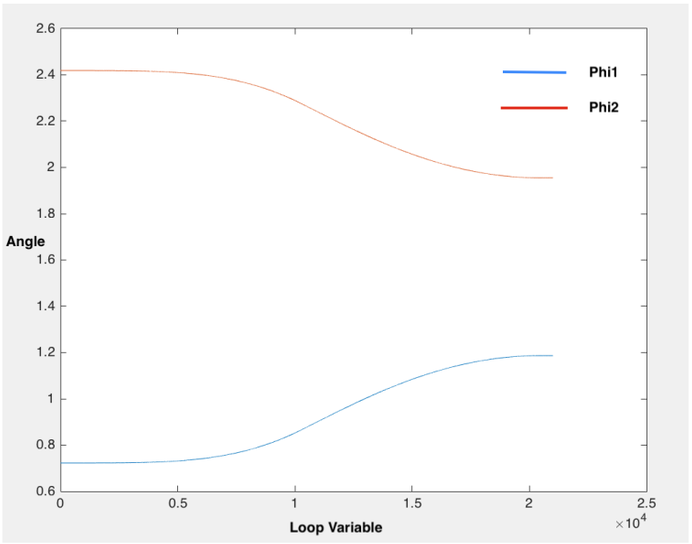


Figure 8: Angles vs loop variables, phi1 - red and ph2 - blue

The corresponding actuator torques are also plotted as shown below.

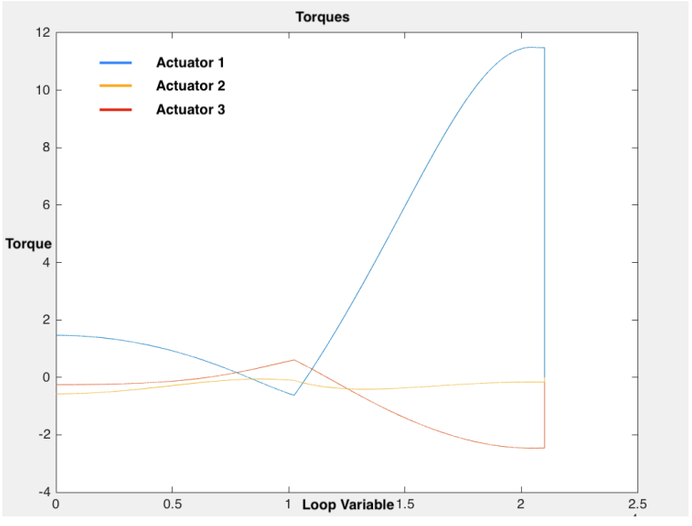


Figure 9: Actuator torques(y-axis), loop variables(x-axis)

3.2 Energy Optimal Control

A similar approach is followed for solving the energy optimal control of the six-bar. The model is feedback linearized and then this decoupled model was used to solve the problem. Two of the following approaches show the variation in the results in two different methods.

3.2.1 Dynamics Reformulation

For solving the optimal control problem, the dynamics has to be reformulated in terms of both states and co-states. For this, the Pontryagins Maximum Principle is used. A Hamiltonian is first derived and then its derivatives along optimal trajectories are computed to find relation between the control and the co-state variables. The used equations are the following.
The cost function of an optimal control problem is as described below.

$$J = \Phi\left[\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f\right] + \int_{t_0}^{t_f} \mathcal{L}\left[\mathbf{x}(t), \mathbf{u}(t), t\right] \mathrm{d}\,t$$

Figure 10: The cost function for an optimal control problem[4]

The Hamiltonian and its canonical equation are as below,

$$H(x^*(t), u^*(t), \lambda^*(t), t) \leq H(x^*(t), u, \lambda^*(t), t)$$

Figure 11: Property of Hamiltonian along optimal trajectory

$$-\dot{\lambda}^{\mathrm{T}}(t) = H_x(x^*(t), u^*(t), \lambda(t), t) = \lambda^{\mathrm{T}}(t) f_x(x^*(t), u^*(t)) + L_x(x^*(t), u^*(t))$$

Figure 12: Canonical equation

3.2.2 Simple Shooting

This is an algorithm to solve a two point boundary value problem. The method is that the BVP is posed as an IVP by guessing some co-state values. Then the difference in the propagated end value and the actual boundary value is calculated. Now the initially guessed co-states are perturbed and the final values are noted. This gives a sense of sensitivity of each parameter. Then a Jacobian is formulated for each step and this is used to compute the updated values of the initial co-state guesses, this is done iteratively until the difference between the final value of the IVP and the given boundary condition.
The obtained control algorithm is demonstrated in six bar sim.m file.

3.2.3 Polynomial Interpolation

Typically in industries optimizing large non-linear functions of time for control algorithms are time consuming and cumbersome. So they resort to interpolation techniques[2]. The most commonly used interpolation is the cubic spline interpolation using b-splines. The usage of b-splines vs Bzier curves is a matter of choice and convenience. But majority stick with b-spline interpolation of state variables when the state trajectories are unknown.
But using such interpolation is not straight forward. Since the b-spline basis functions span in their own span intervals, numerically integrating them requires specifying the ranges and formulation of the basis for each interval.

Since the simple shooting method has only solved a linearized model, there are no coupling terms and hence would only give a sub-optimal solution. To explore the global optima, we will have to consider the full dynamics. So the following method has been used.

First the dynamics has been formulated in the full-configuration space and is converted into actuator space. Then, the expression for torque is written in terms of mass and angle

variables. Now the norm of this expression gives us the norm of the torque vector. Thus minimizing this expression over time, means that the actuator torques are minimized over the given time for the initial and final conditions posed on it.

To further simplify the problem, the actuator angles are interpolated as polynomials functions of time. This particular choice is because of our unfamiliarity with the integration of b-splines and also that the feedback linearized solutions seem to fit a polynomial interpolation. So since we already have four constraints on the angles, we interpolated with a fourth order interpolation, leaving only one design variable for optimization. Using these the norm of torque expression is rewritten.

Now, this turns out to be a huge expression with non-linear trigonometric terms and variables of optimization. To obtain the energy optimal torques this huge expression is to be integrated from the initial to final time and then minimized. NIntegrate of Mathematica would take a lot of time to complete such an integration. Hence we have resorted to an approximate integration using Gauss Quadrature rule. The integration was evaluated for different cases like 3 point, 5 point and 10 point sampling. Then this expression was used to minimize using NMinimize function in Mathematica. This gives us the design parameter values for actuator angles.

The preocess is iterated twice increasing the polynomial order and the Gauss Quadrature sampling. The results are illustrated in the six bar plotter.m function along with the sample data set for each iteration.

Later the problem was solved including gravity and is observed to correlate with the solution of the rising camel problem!

4 Updated Information

This section expands all the discussions that we had in the class after the presentation.

Q. In the case where the mass was unrealistically increased, what does the path being in horizontal mean ? Does it have any physical meaning ?
A. The manipulator was enforced to reach a final end state, so the final total energy is the potential energy of all the links only. Since the net energy is the only energy being pumped by the motors, the interest of the motors is to take the body link to the top and hence the energy pumped into the system shouldnt be wasted as potential energy of the system. So the manipulator tries to maintain the body in the horizontal position i.e. no energy is used as potential energy and all the energy pumped in is the kinetic energy of the high massed link. Thus once the transmission angle is optimal the other actuator simply goes to full actuation pushing the link to the final state. This explains that horizontal path.

Q. Are we lucky that we found a solution which doesnt get lost in a singularity resulting in an infeasible solution ? A. Ofcourse the quality of the optimizer matters. But the following claim clears the reason on why it would converge irrespective of the quality of the minimizer.

Loss type singularity:
Since the optimizer is just the norm of the torque values, there is a high chance that the ϕ's or the passive angles become imaginary and hence the optimizer will still give a solution and that would mean the mechanism teleporting(shifting branches) from one state to the other without any continuation between them. Infact we faced a similar issue when the optimizer gave a solution. Later the problem was totally eliminated by adding a constraint that the imaginary part of the torques should be zero at all states and the real part should be minimized.

Gain type singularity:
The discussion also was on gain type singularity this might occur even if we were to constraint the imaginary part. But even that doesnt mean that the solution was lucky. The argument is that, we already know that the solution exists as the feedback linearized system has given a solution. Hence in the worst case, the optimizer still should be able to give me this solution.

Put in other words, it is just a feasibility problem of any optimization setting. In this case we know that it is feasible and hence also know that I **should** get a solution out of my optimizer.
This answers the two questions asked in the class.

5 References

1. Jingjun YU, et.al., "Motion capability analysis of a quadruped robot as a parallel manipulator", Front. Mech. Eng., 2014.

2. J.E.Bobrow, et.al., "Optimal Robot Motions for Physical Criteria", Journal of Robotic Systems, 2001.

3. J.E.Bobrow, et.al., "Time-Optimal Control of Two Robots Holding the Same Workpiece ", Journal of Dynamic Systems, Measurement, and Control

4. D.Librezon, "Calculus of Variations and Optimal Control Theory: A Concise Introduction", Princeton University Press Princeton, 2011.