ESAM 346
Modeling and Computation in Science and Engineering

Winter 2005

Hermann Riecke

Problem Set 5

March 5, 2005 Due Wednesday, March 16, 2005

The Robot Contest

In this homework you write a code that controls the simple robot discussed in class. One component of the homework will be a speed contest. The goal is therefore to write a fast code that controls the robot efficiently and accurately.

The robot consists of a an extendable arm that rotates about a pivot. The state of the robot is defined by the angle ϕ of the arm with respect to the $x-$axis, the length r of the arm, and the respective time derivatives, $\omega = \frac{d\phi}{dt}$ and $v = \frac{dr}{dt}$. The goal is to control it to move the arm from an initial position $\mathbf{x}_0 = (\phi_0, r_0, \omega_0, v_0)$ to a final position $\mathbf{x}_f = (\phi_f, r_f, \omega_f, v_f)$ while expending the least amount of the cost function

$$P = \frac{1}{2}\int_0^{t_f} \tau(t)^2 + F(t)^2\, dt. \tag{1}$$

Here $\tau(t)$ and $F(t)$ are the applied torque and force, respectively. As discussed in class the equations describing the motion corresponding to the minimum of P are given by

$$\frac{d\phi}{dt} = \omega \tag{2}$$

$$\frac{dr}{dt} = v \tag{3}$$

$$\frac{d\omega}{dt} = \frac{1}{I_{eff}}\left(\tau - 2mr\omega v\right) \tag{4}$$

$$\frac{dv}{dt} = \frac{F}{m} + r\omega^2 \tag{5}$$

Here m is the combined mass of the arm and the transported weight and $I_{eff} = I + mr^2$, where I is the moment of inertia of the body of the robot with the arm retracted. The Lagrange multipliers $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)$, which determine the torque and the force, arise from the minimzation procedure and satisfy

$$\frac{d\lambda_1}{dt} = 0 \tag{6}$$

$$\frac{d\lambda_2}{dt} = \frac{2m}{I_{eff}}\left(\omega v + r\frac{d\omega}{dt}\right)\lambda_3 - \omega^2\lambda_4 \tag{7}$$

$$\frac{d\lambda_3}{dt} = -\lambda_1 + \frac{2m}{I_{eff}} rv\lambda_3 - 2r\omega\lambda_4 \tag{8}$$

$$\frac{d\lambda_4}{dt} = -\lambda_2 + \frac{2m}{I_{eff}} r\omega\lambda_3. \tag{9}$$

The torque and force are related to $\lambda_{3,4}$ via

$$\tau = \frac{\lambda_3}{I_{eff}} \qquad F = \frac{\lambda_4}{m}. \tag{10}$$

The problem to solve is a boundary value problem for the 8 differential equations (2-5) and (6-9) with initial conditions given for $\mathbf{x}$, $\mathbf{x}_0 = (\phi_0, r_0, \omega_0, v_0)$, but not for λ. Instead, there are terminal conditions for $\mathbf{x}$,$\mathbf{x}_f = (\phi_f, r_f, \omega_f, v_f)$. Thus the initial conditions for λ, $\lambda_\mathbf{0} = (\lambda_{10}, \lambda_{20}, \lambda_{30}, \lambda_{40})$, need to be determined using the shooting method using the conditions on $\mathbf{x}_0$ and $\mathbf{x}_f$.

Assignment to hand in:

1. Write a solver for (2-5) and (6-9) for a set of given initial conditions.
   Discuss which method you chose for this solver and why you chose that method.

2. As discussed in the class notes, use the shooting method to solve this boundary-value problem. It requires the Jacobian

$$J_{ij} = \frac{\partial G_i}{\partial \lambda_j}$$

   of the function

$$\mathbf{G}(\lambda_0) \quad \equiv \quad \mathbf{X}_{t_f}(\mathbf{x}_0, \lambda_\mathbf{0}) - \mathbf{x}_f,$$

   where $\mathbf{X}_{t_f}(\mathbf{x}_0, \lambda_0)$ maps the initial condition $\mathbf{x}_0$,λ_0 into the state $\mathbf{x}(t_f)$ at the final time t_f,
   Write a function that approximates the Jacobian by running the solver for slightly varied initial conditions of λ.

3. Combine your solver from part 1 with the function for the Jacobian from part 2 to build a Newton iteration method for solving the boundary-value problem with

$$\mathbf{x}_0 = (0, R, 0, 0) \qquad \mathbf{x}_f = (\frac{\pi}{2}, R, 0, 0) \quad \text{at} \quad t_f = 1.$$

   This corresponds to commanding the arm to rotate a quarter turn counter-clockwise. For $I = 3$ and $R = 5$, plot and compare the resulting optimal paths for three different values of the mass

   (a) $m = 5$
   (b) $m = 0.5$
   (c) $m = 0.02$

   Note: You will find that the convergence of the Newton iteration becomes increasingly tricky when the mass is decreased. In fact, with the standard Newton you may not even get it to converge in the case $m = 5$ and $R = 5$. As an initial test case you may therefore want to consider $R = 1$ instead.
   To improve the convergence of the Newton iteration you may want to implement it in the form

$$u_i^{l+1} = u_i^l - \omega \sum_j \left(J^{-1}\right)_{ij} G_j$$

   where the parameter $0 < \omega \leq 1$ allows you to reduce the distance the Newton step goes along the tangent (see discussion in the class notes). This modification can avoid that the Newton iteration diverges into 'neverland'. Discuss the specific choices you make in your Newton implementation.
   Is rigid rotation (i.e. $\frac{dr}{dt} = 0$) the optimal path? In each of the three cases determine the difference between the value of the cost function for the optimal path obtained

with your Newton method and its value for a path following a circular path. One possible circular path is given by

$$\phi(t) = \frac{\pi}{54}\left(-2t^3 + 9t^2\right) \qquad r(t) = 1. \tag{11}$$

To obtain the cost function for the circular path insert (11) into (2-5) to determine $\tau(t)$ and $F(t)$ and compute the integral (1).

Contest:

Rules for the contest are:

1. you cannot use any canned solvers provided in matlab (e.g. ode45).
2. submit your program as a matlab function of the form
   ```
   function [t,x,u,lambda]=robotname(tf,x0,xf,m)
   ```
   Choose a unique name for your robot
   Inputs:
   (a) `tf` is the terminal time (the initial time is always assumed to 0)
   (b) `x0` is a row vector specifying the initial condition $\mathbf{x}_0 = (\phi_0, r_0, \omega_0, v_0)$
   (c) `xf` is a row vector specifying the terminal condition $\mathbf{x}_f = (\phi_f, r_f, \omega_f, v_f)$
   (d) `m` gives the value of the mass to be used.

   Outputs:
   (a) `t` is vector of arbitrary length N that gives the times at which the solution was computed
   (b) `x` is a $N \times 4$ matrix with `x(i,:)` giving the state of the robot at time `t(i)`
   (c) `u` is a $N \times 2$ matrix with `u(i,1)` giving the torque and `u(i,2)` giving the force applied at time `t(i)`.
   (d) `lambda` gives the initial values $\lambda_0 = (\lambda_{10}, \lambda_{20}, \lambda_{30}, \lambda_{40})$ determined by your code to lead to the optimal path.

   For instance, if the contest problem was the challenge of part 3b then during the contest your program will be called as
   ```
   [t,x,u,lambda]=robotname(1,[0,5,0,0],[pi/2,5,0,0],0.5)
   ```
3. combine all functions into a single file with the main function `robotname` listed first.

In the contest the path produced by your robot will be plotted along with the computed cost function (you do not have to code that part). The winning robot will have the lowest combined total for the time elapsed plus the value of the cost function plus 1000 times the error in the terminal state. The error will be determined by running the equations with `ode45` based on the initial conditions $\lambda_0 = (\lambda_{10}, \lambda_{20}, \lambda_{30}, \lambda_{40})$ provided by your code and comparing the resulting final position $\mathbf{x}(t_f)$ with the specified terminal position $\mathbf{x}_f$.