Question Sheet for

Computational Robot Dynamics 2022

Instructor: Roy Featherstone, Italian Institute of Technology
Website: http://royfeatherstone.org/teaching

Note: All questions that involve the use of Matlab, but do not require the use of either Simulink or 'showmotion' from `spatial_v2`, can be performed equally well on Matlab or GNU Octave.

Questions for Part 1 — Modelling

Question Mod1

Write Matlab functions to perform each of the following calculations. In each case, `lam` is a parent array. Avoid using recursion, and avoid algorithms with a worse-than-linear complexity.

(a) `Ka=kappa(lam)` where `Ka` is a cell array such that `Ka{i}` is $\kappa(i)$.

(b) `Mu=mu(lam)` where `Mu` is a cell array such that `Mu{i+1}` is $\mu(i)$, so `Mu{1}` is $\mu(0)$.

(c) `Nu=nu(lam)` where `Nu` is a cell array such that `Nu{i}` is $\nu(i)$.

(d) `d=depth(lam)` where `d` is the depth of the tree, defined as $d = \max_i |\kappa(i)|$ (where $|S|$ is the number of elements in set S).

Question Mod2

Following the examples on Slide 16, work out the joint models for each of the following joint types:

(a) a cylindrical joint on the x axis;

(b) a helical joint of pitch h (in metres per radian) on the y axis, using the rotation angle as its joint variable;

(c) a rack-and-pinion joint (i.e., a cylinder rolling without slipping on a plane) in which the parent body is the rack (plane), the child is the pinion (cylinder), the pinion has a radius of r, the pinion's rotation axis is parallel to the x axis, and positive rotation of the pinion causes it to translate in the $-y$ direction, so the rack is parallel to the x–y plane and below the pinion. Put the origin of the successor frame at the centre of the pinion.

Questions for Part 2 — Inverse Dynamics

Question ID1

Write a function that uses the function `ID` from Spatial_v2 to calculate the coefficients $\boldsymbol{H}$ and $\boldsymbol{C}$ of the general joint-space equation of motion

$$\boldsymbol{\tau} = \boldsymbol{H}\ddot{\boldsymbol{q}} + \boldsymbol{C}\,.$$

Ignore the possibility that there may be external forces. Compare your answers with the values computed by `HandC`.

Questions for Part 3 — Efficiency

Question Effi1

Create two robots using `planar(2)`, and modify one of them by adding a point mass to link 2 and a corresponding negative poit mass to link 1, as described on slides 13–15. Do they have the same inverse dynamics? (Tip: read the documentation for `mcI`, and remember that `planar` produces models that use planar vectors rather than spatial vectors.)

Questions for Part 4 — Forward Dynamics (CRBA)

Question FD1

Enter the following lower-triangular matrix into Matlab:

```
L = [1 0 0 0 0; 2 3 0 0 0; 4 0 5 0 0; 6 0 7 8 0; 9 0 10 0 11];
```

(a) What value of lambda goes with this matrix?

(b) Calculate the matrix `A=L'*L`. Does it have the same sparsity pattern?

(c) Calculate `L2=LTL(A,lambda)` using your answers to parts (a) and (b). Is `L2` the same as `L`?

(d) Now try `L3=LTL(A,[0,1,2,3,4])`. What do you notice?

Questions for Part 6 — Simulation

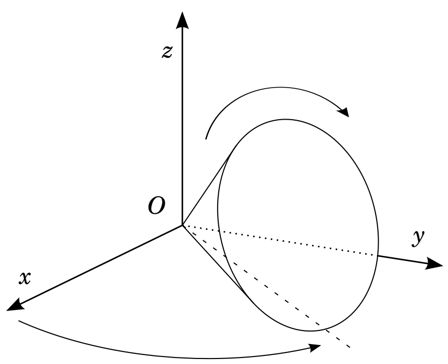

Figure 1

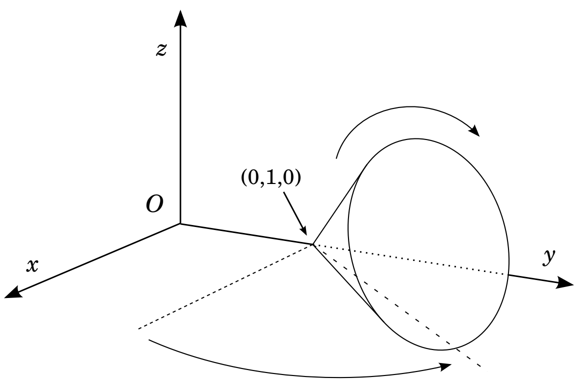

Figure 2

Question Sim1

Figure 1 shows a cone that is rolling without slipping on the x–y plane. Its apex is at the origin, and the cone is rolling at a constant speed such that the line of contact makes one rotation around the z axis per second. At time $t = 0$ the cone is touching the x axis; at time $t = 0.25$ it is touching the y axis; and at time $t = 1$ it is again touching the x axis. The half angle at the cone's apex is 30°, which means that the cone will rotate twice around its own axis for each one rotation of the line of contact about the z axis.

(a) Write a Matlab function `w=conew(t)` that calculates the cone's angular velocity as a function of time.

(b) Write a Matlab function `qd=coneqd(t,q)` to calculate the time derivative of a quaternion, $q(t)$, representing the orientation of the cone at time t, given both t and $q(t)$. This function should use the functions `conew` from your answer to part (a) and `rqd` from *spatial_v2*.

(c) Issue the Matlab command `[T,Q]=ode45(@coneqd,[0,1],q0);` where `q0=[1;0;0;0]` is the initial value of q (i.e., $q(0)$). This function will integrate the motion of the cone for one second. Plot the result using `plot(T,Q)`. Do you understand the resulting graph? Is it correct? **Tip:** you can force the integrator to take smaller steps, to get a smoother-looking graph, with an optional extra argument to `ode45` of the form `odeset('MaxStep',0.01)`.

Note: Matlab normally updates its internal copy of a user-defined function each time it is called from the command line; but it does not do so if the function is called via its handle, as in `@coneqd`. Therefore, each time you modify `coneqd` while working on part (c), you will have to call it explicitly at least once to make Matlab update its internal copy.

Question Sim2

Figure 2 shows a situation identical to the one in Figure 1 except that the cone is translated one metre in the y direction so that its apex is at the point $(0, 1, 0)$. All other aspects of the motion are the same as in question 1.

(a) Write a Matlab function `v=conev(t)` that calculates the cone's spatial velocity as a function of time.

(b) Write a Matlab function `pd=conepd(t,p)` to calculate the time derivative of a 7-element position coordinate vector $\boldsymbol{p}(t) = [\boldsymbol{q}(t)^{\mathrm{T}}\ \boldsymbol{r}(t)^{\mathrm{T}}]^{\mathrm{T}}$, in which $\boldsymbol{q}(t)$ is a quaternion representing the orientation of the cone at time t, as in question 1, and $\boldsymbol{r}(t)$ is a vector giving the position of the body-fixed point on the cone that coincides with the origin (not the apex) at time $t = 0$. This function should use the functions `conev` from your answer to part (a) and `rqd` from *spatial_v2*. When calculating $\dot{\boldsymbol{r}}$, bear in mind that $\dot{\boldsymbol{r}}$ is the velocity of the body-fixed point at $\boldsymbol{r}$, whereas the linear part of the spatial velocity is the velocity of the body-fixed point that coincides with the origin at the current instant. These two points are the same only when $\boldsymbol{r} = \mathbf{0}$, which happens only at $t = 0$ and $t = 1$.

(c) Issue the Matlab command `[T,P]=ode45(@conepd,[0,1],p0);` where `p0=[1;0;0;0;0;0;0]` is the initial value of $\boldsymbol{p}$. This function will integrate the motion of the cone for one second. Plot the quaternion part of the result using `plot(T,P(:,1:4))` and the positional part of the result using `plot(T,P(:,5:7))`. Is the quaternion plot the same as in question 1? Do you understand the plot for $\boldsymbol{r}$?

(d) Repeat part (c) using an initial value of `p0=[1;0;0;0;0;1;0]`. Can you explain the result?

Question Sim3

Continuing with the cone in Figure 2, write a function `a=conea(t)` that calculates the spatial acceleration of the cone, and use the command `[T,V]=ode45(@(t,v)conea(t),[0,1],v0);`, where `v0=conev(0)`, to integrate it to obtain the spatial velocity. (The expression `@(t,v)conea(t)` creates an anonymous function that takes two arguments, `t` and `v`, and calls `conea(t)`.) Compare the result with the velocity calculated directly by `conev`. Are they the same?

Question Sim4

Using the functions in *spatial_v2*, create a model of a spinning top and create a Simulink model demonstrating the precession of the spinning top under gravity. **Tips:** (1) the top will have to spin quickly in order to precess. If your top doesn't precess, then maybe you didn't give it a big enough initial velocity. (2) for easy compatibility with `showmotion`, save the joint position vector in 3D array format using a 'to workspace' Simulink block. (3) By default, Matlab saves only the last 1000 time values in `tout`. If that is not enough then you can change it in the data import/export pane of the model configuration parameters window, which is accessed via the simulation menu.