## Assignment for Week 4 readings: (due Tues 9 November)

Jasmine Pozzi, Mattia Franchi de' Cavalieri, Niccolo Pagliarani, and Mark Cutkosky ([#week4 on Slack](https://topicsinmulti-b5g1638.slack.com/archives/C02JD0XE0B0))

Goal: Take a closer look at grasping, internal forces and friction. Understand why a linear mapping (grasp matrix) does not always give physically reasonable internal force magnitudes. Understand how the Ferrari/Canny and GraspIt! (Miller/Allen) method works to rapidly evaluate candidate grasps.

* [Week4 Google Presentation](https://docs.google.com/presentation/d/1QwW6uQ9NJ4FOtZs4m8TTbzdPLIy4WplvpCtWnWVlMA0/edit?usp=sharing)

### Q1

Let’s take another look at the wrench matrices from Week 1. Recall that you can build a Grasp matrix that maps from finger contact wrenches to body wrenches. Depending on how many fingers you have, it may not be square, but you can augment it with internal forces and get an invertible matrix. For example, we found in Week 1 that we could augment the grasp wrench matrix with its null space.

For control, we usually want the inverse mapping: we have some forces we want to exert on a grasped object and we want to *find* the corresponding fingertip forces.

But the article by Yoshikawa and Nagai points out that any linear mapping does not always give physically appropriate solutions for the internal grasp force. For example, for a two-finger grasp, they argue that the right answer for the grasp force is: fint = *min*{f\_left, f\_right) where f\_left and f\_right are the inward-pointing components of the left and right finger forces.

Q1.1 Load the script “​​Left-Right-GraspMatrix.py” in the [Week 4 folde](https://drive.google.com/drive/folders/1Lqj8KyyCtpQt4lPkHW888i3eBuNDiQOx?usp=sharing)r and satisfy yourself that you understand how the Part1 script works. Part1 is the linear mapping, very similar to what we did with FlatBot in Week1. The result plots the normal and tangential forces at the left finger and shows that for some directions of the external force, friction will not be satisfied despite having what seems like a reasonable internal force.   
OK... We can increase the magnitude of the requested internal force, *fint*. How big does it need to be to prevent slipping? And how large is the maximum value of the X component of the finger force in this case?

Q1.2 Now run the script for Part2 which implements a nonlinear solution corresponding to the minimum of the normal contact forces: min{|fx1|,|fx2|}. What is the minimum magnitude of *fint* in this case? And how large is the maximum value of the X component of the finger force in this case?

By the way, if you look at [Appendix D.2 p. 190 of Weston Griffin’s thesis](http://bdml.stanford.edu/oldweb/touch/publications/griffin_thesis.pdf) (in the readings list) you’ll see that this is how we controlled the Dexter 2-fingered hand.

Q1.3 Explain why the solution in Q1.2 can be preferable to the linear solution. An optional, more challenging, question: *How could this nonlinear mapping be extended to more than two fingers?* Yoshikawa and Nagai give a solution for 3 fingers but it gets much more complicated. At some point one might just say: “Let’s solve a linear or nonlinear optimization problem with some fast algorithm and get all the finger forces, subject to friction constraints.”

### Q2

Let us experiment with a (slightly simplified) version of the Ferrari/Canny algorithm as used in GraspIt! Figure 2 shows three fingers grasping a trapezoidal block with friction. The fingers are at the midpoints of their respective sides. Load the script **ConvexHullUnion.py,** whichgoes as follows:

* For each finger, assuming a unit normal force and some coefficient of friction, compute the vectors corresponding to the extrema of that finger’s friction cone.
* Compute the corresponding wrenches on the body for those forces. These are collected in a three-dimensional wrench space: [fx,fy,mz].
* Call a program to compute the convex hull of the wrenches.
* Compute distances from the planes associated with each of the “simplices” (facets) of the convex hull to the origin. The method here assumes the hull encloses the origin, which it should for a force-closure grasp.
* The least of these distances corresponds to the “least wrench” (worst-case wrench for this grasp) and determines the largest sphere, centered at the origin, that could be contained in the convex hull.

Q2.1 This is obviously not a very good 3-finger grasp. Even with no external force, what is the minimum value of the coefficient of friction, mu, such that the convex hull contains the origin in [fx,fy,mz] space? Physically, what does it mean for the convex hull to contain the origin?

Q2.2 Now add a fourth finger, identical to the others, with the same unit normal force and friction coefficient = 0.5 for all of them. Assuming you can put all the fingers anywhere you want along the sides of the trapezoid, where would you put them to maximize the least wrench? You don’t need to do a formal optimization, just use your intuition and search for what seems to be best. (It helps that it is symmetric.) Do this using the Convex Hull of the Union of the body wrenches.

Q2.3 (optional) Try the other metric that involves the convex hull of the Minkowski sum using the script **ConvexHullMinkowski.py**. Explain what is different iin this case.