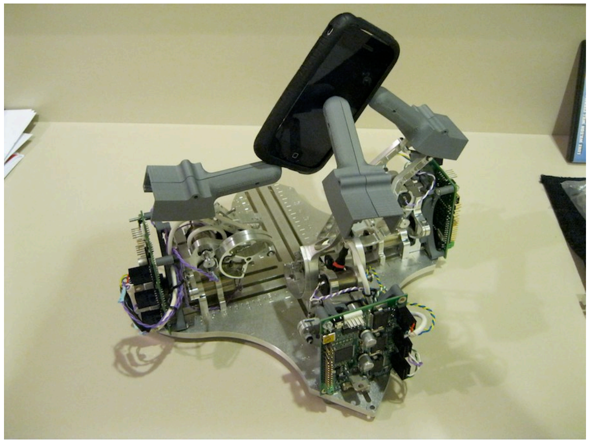
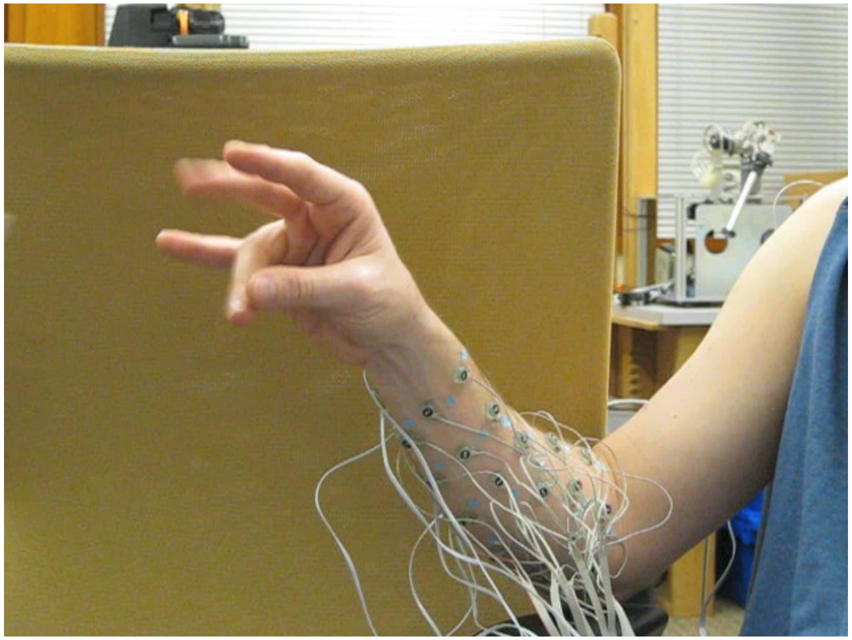
Complex object manipulation with hierarchical optimal control

Dr. Alex Simpkins and
Prof. Emanuel Todorov
University of Washington, Seattle

Object Manipulation



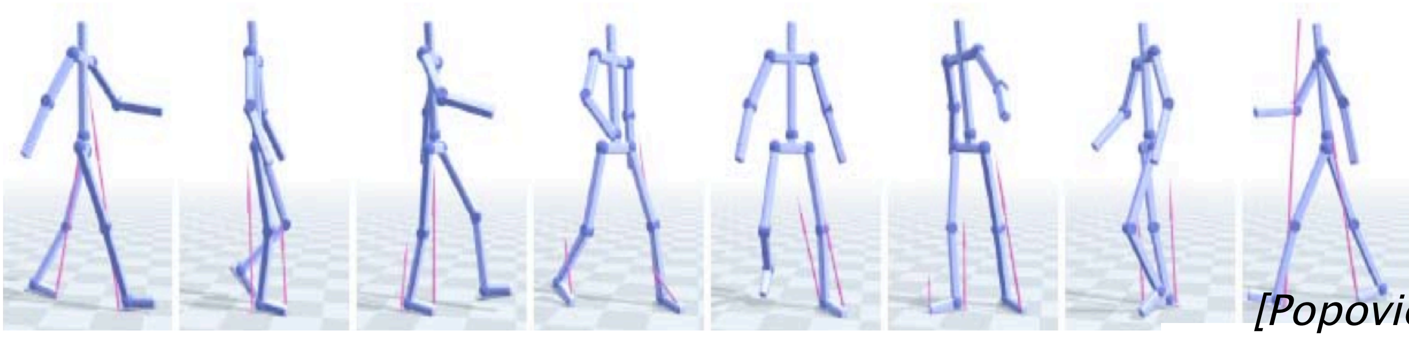
- *Controlling the trajectory or state of an object with external forces applied by manipulators*
- Complex problem
  - Nonlinearities
  - Contacts/collisions
    - Discontinuities
    - Changing problem
  - Friction
  - Redundancy
  - High dimensionality
  - Under-actuation



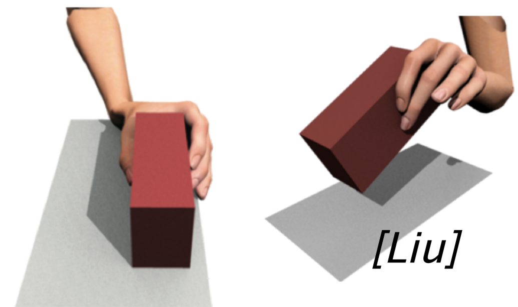
Object Manipulation

- Significance/Relevance
  - Interaction with the world is part of learning and normal functioning
    - We explore and process Haptic information, expand knowledge
    - Better method, improved learning
  - Humans are excellent at interacting dynamically with our dynamic world!
    - This is poorly understood from a control/ machine learning perspective
      - Crumple a piece of paper 60 times, you will do it 60 different ways
  - Significant portion of neurons in brain involved in sensorimotor processes
  - Artificial systems need to interact with the world intelligently

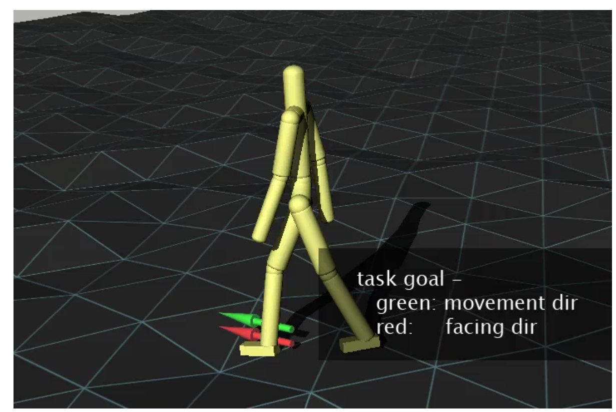
Existing approaches



[Popovic]



[Liu]



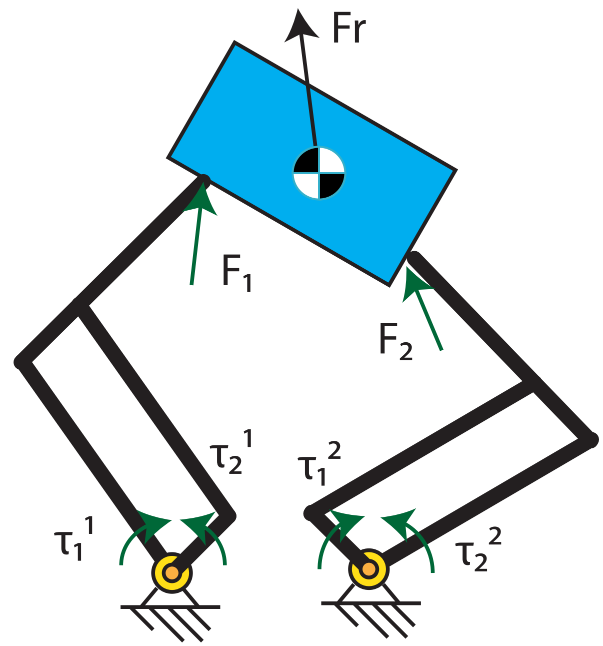
[Popovic]

- Manipulation
  - Assume a grip predefined, contacts static/quasi-static
  - Determine forces to move object in prescribed manner
  - Optimal control
- Grasp studies – how do humans grasp objects?
  - Quasi-static notion yet again
- Locomotion approaches
  - Manipulation and locomotion are highly related
    - Many of the challenges are similar
  - Relevant locomotion approaches which are promising based upon optimal control and RL
    - Typically hierarchical, often offline components

Model

- 2D – first step
- Objects

  Mass, center of mass, inertia, boundary
- Given # of manipulators
  - Modeled as point masses, since actuators are very capable, and dynamics are approximately linear (dominated by rotor J)
- Simple/efficient/fast contact/collision algorithm here
  - No-slip condition (equivalent to very sticky objects/manipulator end points)
  - Facilitates realtime implementation
    - Parallel GPU processing possible
- 2-part computation - total forces on
  - Object
  - Manipulators (contact or not)



$$\sum F_{x,o} = \sum_i f_{x_i} - m_o a_{x,o} \tag{1}$$

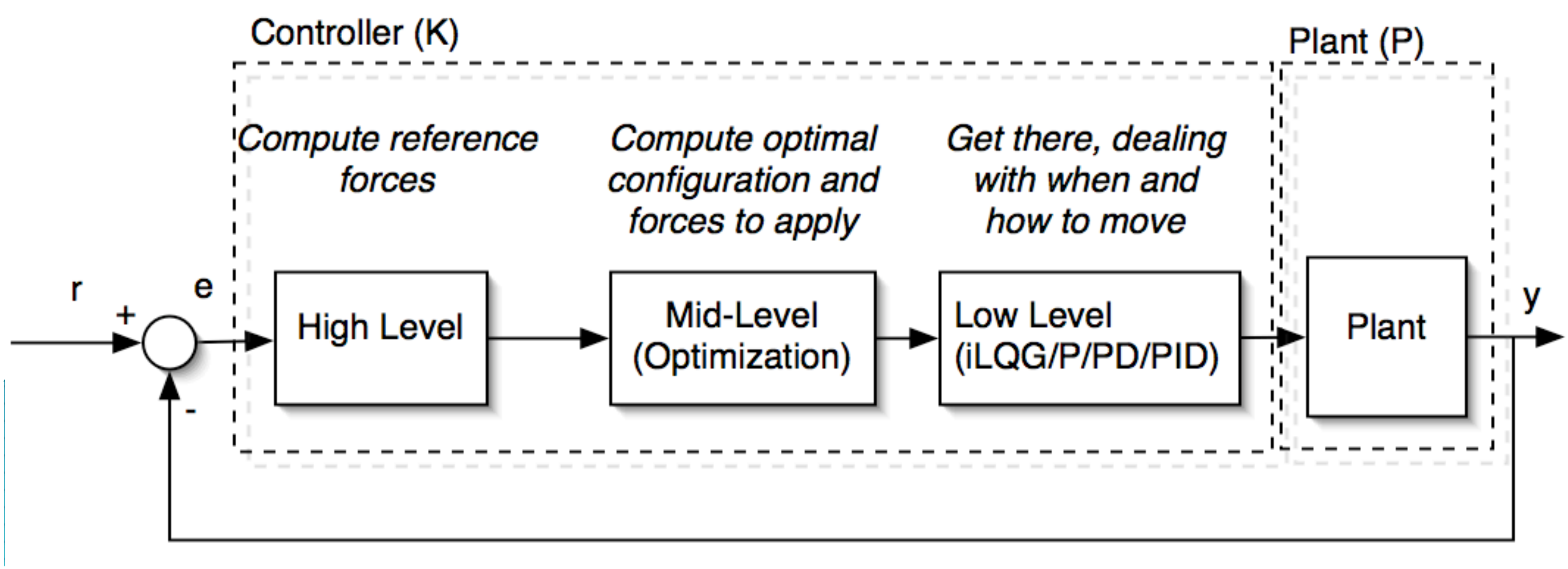
$$\sum F_{y,o} = \sum_i f_{y_i} - m_o g - m_o a_{y,o}$$

$$\sum M_o = \sum_i \big(- f_x d_y(\theta) + f_y d_x(\theta) \big)_i - J\ddot{\theta}_o$$

$$a_{x,i} = a_{x,o} + \ddot{\theta}_i d_x(\theta_i)$$
$$a_{y,i} = a_{y,o} + \ddot{\theta}_i d_y(\theta_i)$$

Hierarchical control methodology

- Break up single complex problem into multiple more tractable problems
- Hierarchical – allows different strategies at each stage, more freedom
- Behaviors can be encoded at various levels

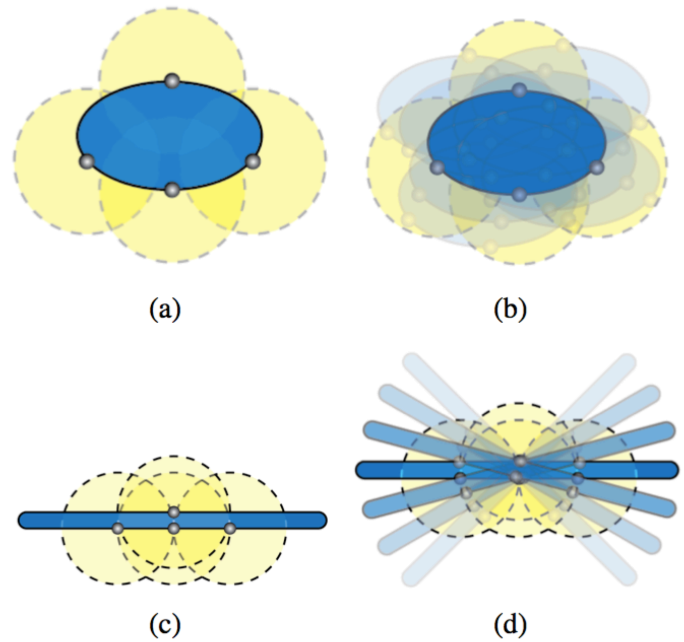


Force-Cost function

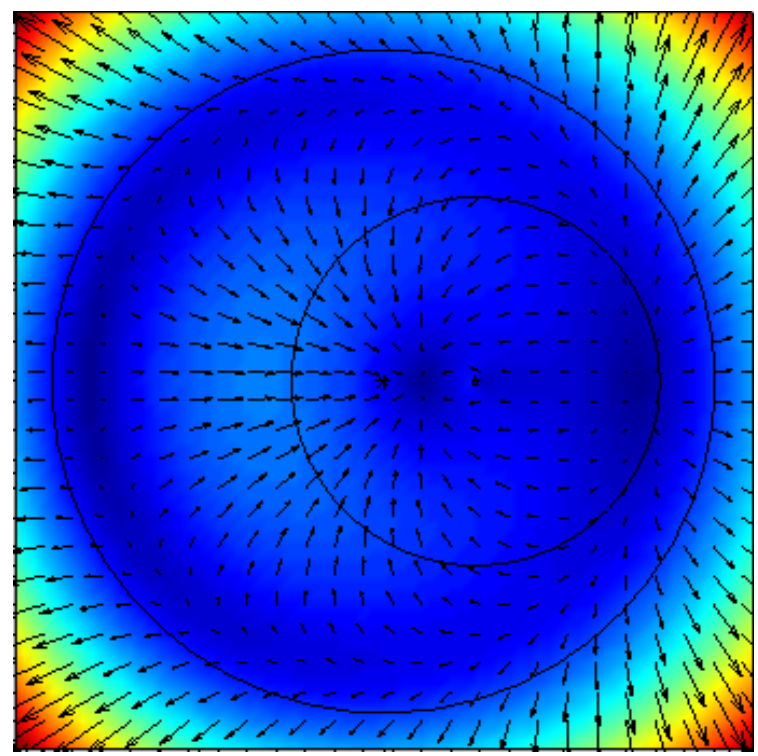
$$F_e = K_e \frac{(x - x_c)}{1 + ||x - x_w||^2}$$

$$F_s = K_s n ||x - x_w|| e^{-a(x - x_c)^2}$$

$$F_t = F_e + F_s$$



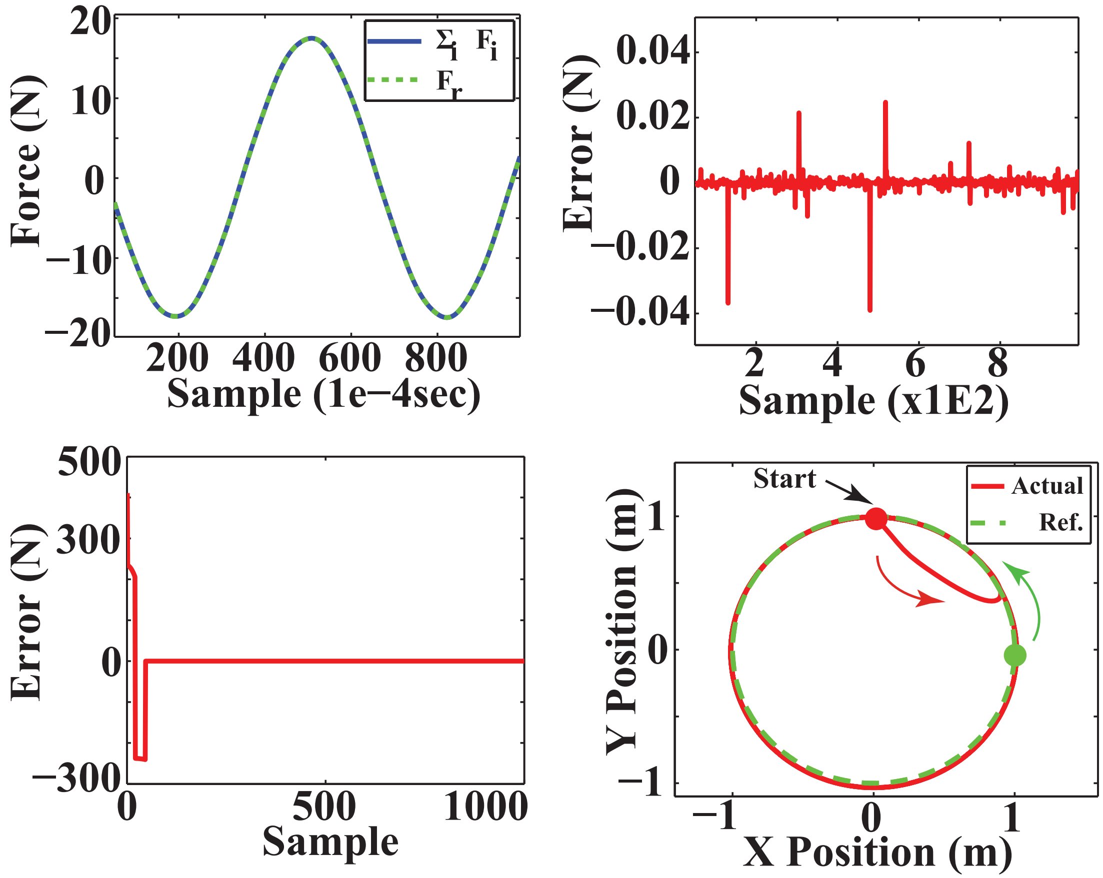
- Mid-Level
  - Given required force, and the state of the system
    - What should the individual forces be?
    - Where should they be applied?
  - Optimization and FC function
    - Part I : Arranged as quadratic optimization, terms account for
      - Force tracking error
      - Minimizing individual forces
    - Part II : Virtual force field, similar to a cost function
      - State of system, workspace of each manipulator, location of end effector, location and parameters (or estimated) of object

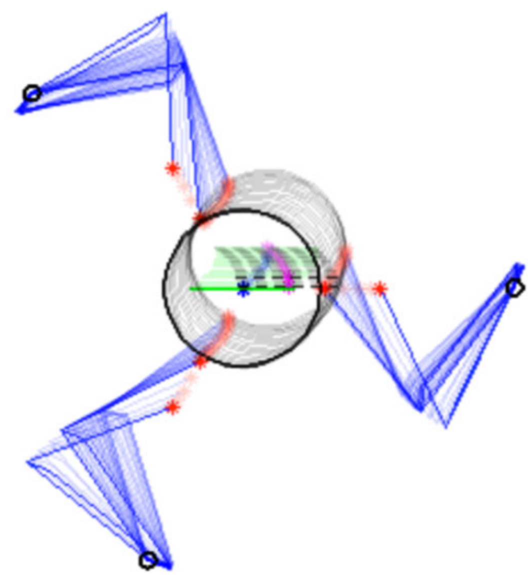


Experimental tasks

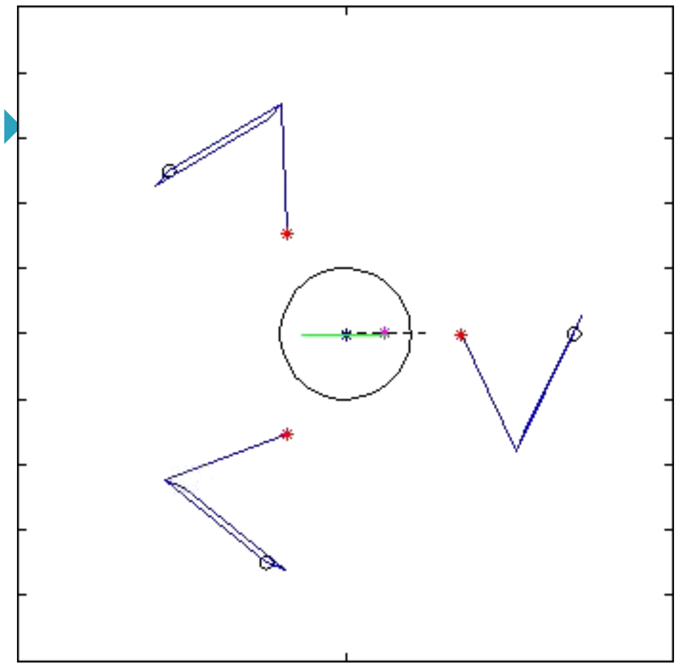
- Given a # of end effectors, manipulate an object of known parameters
  - Grab object, then track some reference for the object
  - Perform continuous rotation of the object in place
  - In both cases manipulator bases are fixed
    - Everything happens due to manipulators, not an arm

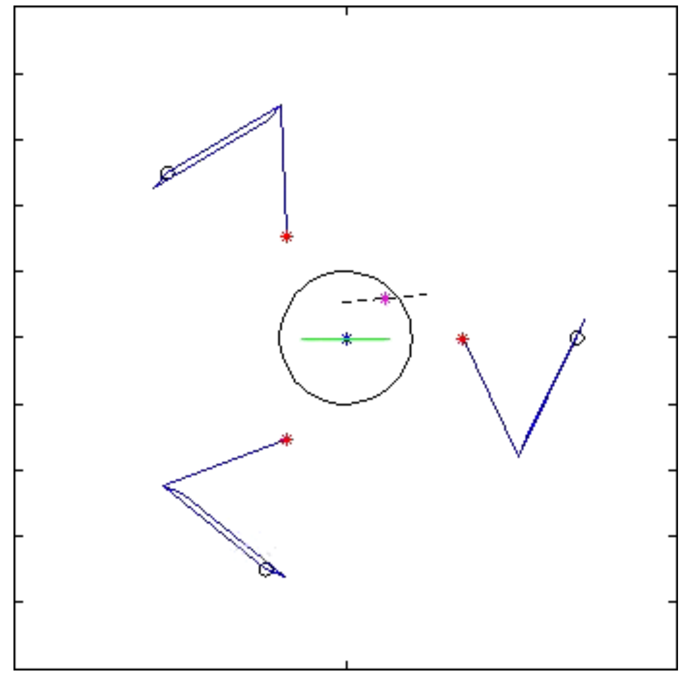
Results – force tracking

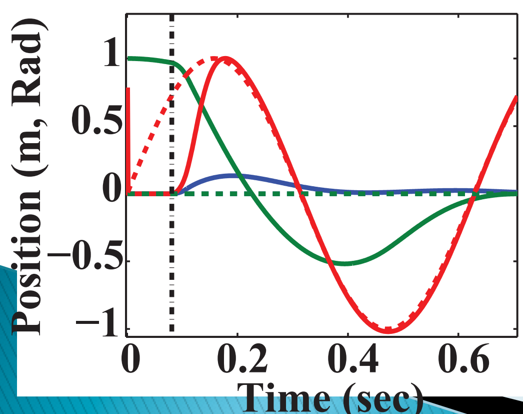


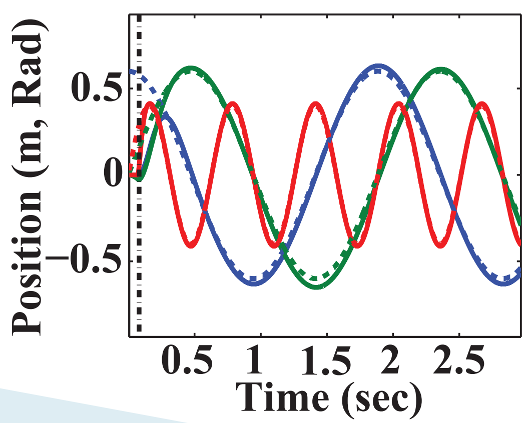


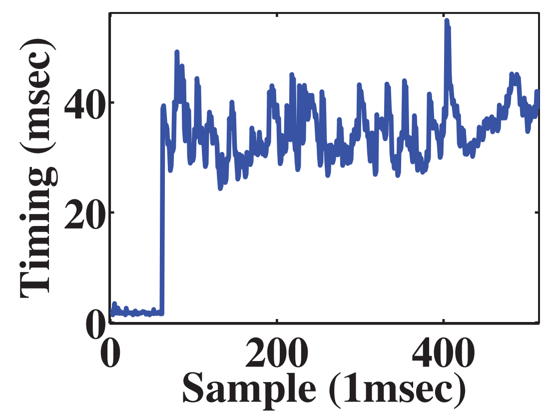
Results – grasp and track



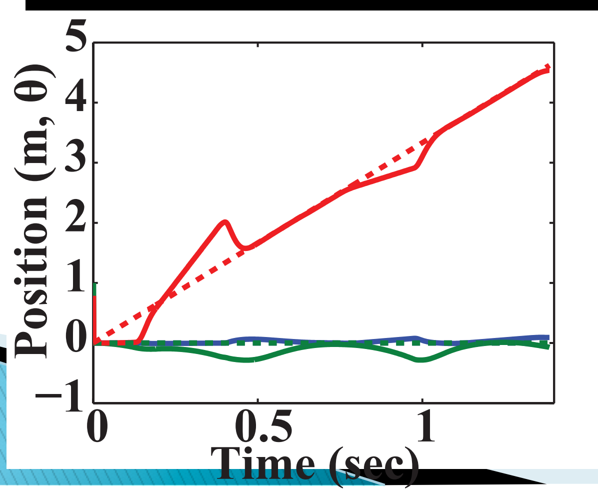


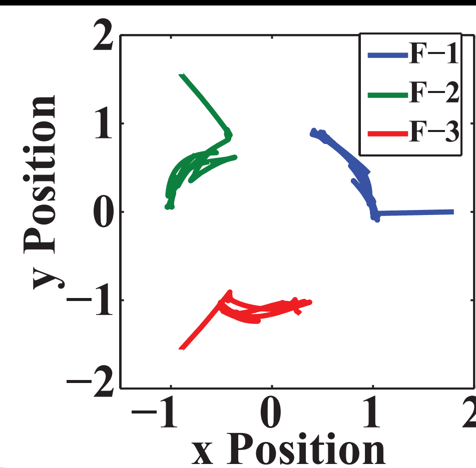






Results – contact breaking





Conclusions

- Contributions
  - New dynamic hierarchical control approach to manipulation, also applicable to locomotion
  - Minimal manual tweaking (as opposed to some ‘painful’ methods), tuned with intuitive parameters (only a few), easy to expand behavioral complexity or blend/change behaviors online
  - Realtime implementation – no offline computation, fast enough to run on real robots
  - Manipulation and locomotion treated as the same problem
  - Joint limit handling with soft boundary acting as a spring when moving outside workspace, contributes to numerical stability, smoothness, parallels biological systems
- In progress/next steps
  - 3D
  - Implementation on real robots (object spinning task)
  - MPC at lower level vs. force fields (comparison)
  - Stability/robustness analysis more quantitative
  - Active sensing – learn object while manipulating, or manipulate to learn object
  - Compare with trajectory methods, also being developed in our lab

Thank you

More information at casimpkinsjr.radiantdolphinpress.com

Appendix – 1

- Ease of tuning – high level doesn't affect low level stability
  - Build stable system in pieces if necessary with difficult systems