Workspace Multiplicity and Fault Tolerance of Cooperating Robots

Daniel A. Brake Daniel J. Bates Vakhtang Putkaradze
Anthony A. Maciejewski

Notre Dame, Colorado State Univ., Univ. Alberta

D. Brake partially supported by grants NSF-DMS-09087551 and NSF-IIS-0812437.
D. Bates partially supported by grants NSF DMS–0914674.
V. Putkaradze partially supported by grant NSF-DMS-09087551.
A. Maciejewski partially supported by grant NSF-IIS-0812437.

November 12, 2015

Scenario

Many robots are designed to work in remote or dangerous locations. Suppose:

- System has multiple robotic arms,
- Workspaces of arms overlap,
- Arms capable of grasping each other,
- One arm undergoes free-swinging joint failure. [1]

Then, to maximize a function of the pre- and post-failure workspaces:

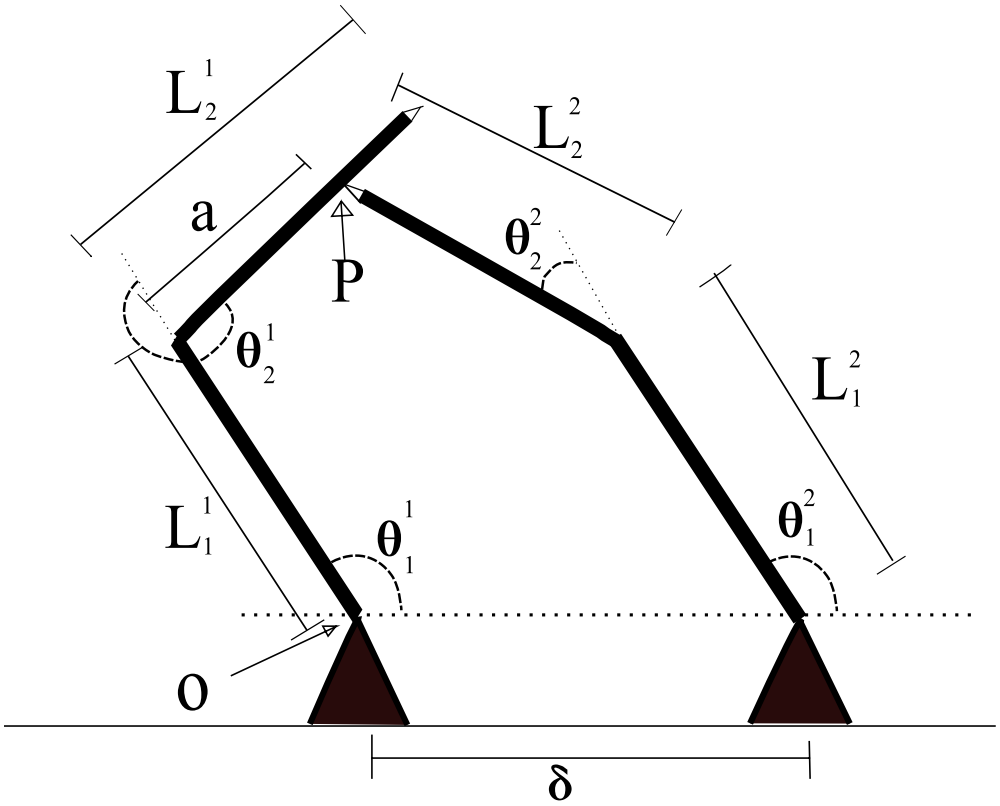
1. How close should the arms be?
2. Where should they connect post-failure?

Our method to answer the questions is:

- Use *homotopy continuation* to solve inverse kinematics at a random sample of space.

[1] English, Maciejewski, IEEE Trans. Robot. Automat., 1998.

The Problem



2D Example of problem.

Optimize to determine a and δ:

- Robots separated by distance δ,
- Right robot (#2) grasps left (#1) at point P, at distance a from the second joint of robot 1,

so that some function of the post-failure workspace is maximized.

The Problem

There are four workspaces to consider:

- W_1, W_2, the *pre-failure* workspaces for robots 1 & 2.
- $W_\cap = W_1 \cap W_2$, the *intersection* of the two pre-failure workspaces. Depends on δ.
- W_f, the *post-failure* workspace. Depends on δ and a.

Our objective is to maximize some function Ω of these four workspaces. We used,

$$\Omega = (1 - \lambda) \cdot |W_f| + \lambda(|W_1| - |W_\cap|),$$

where λ is a problem-specific weighting factor.
Note: $|W_1| - |W_\cap|$ is a measure of the benefit of having the second robot.

Workspace measure

We define a multiplicity-weighted workspace measure,

$$|W| = \int_W m(x)\mathrm{d}x$$

- $m(x)$ is the number of configurations placing the robot in a position x in the workspace.
- If $x \notin W$, then $m(x) = 0$.
- Since the set of singular points is a set of measure zero, they don't affect this number.

Jacobian Method

Traditionally, one would use *Jacobian Method* to solve problem:

- Start robot from initial state, use Jacobian Control[2] to move robot to desired point.:

$$\dot{x} = J(\theta) \cdot \dot{\theta}$$

- If robot can come within ϵ of point, it is in workspace. Else, out.
- Alternately, one could move robot to boundary of workspace, and move in *nullspace* of Jacobian to trace boundary.

Method has shortcomings, however:

- Can only find *one* solution per point.
- Can miss voids in workspace.
- Hard to measure volume of calculated workspace.

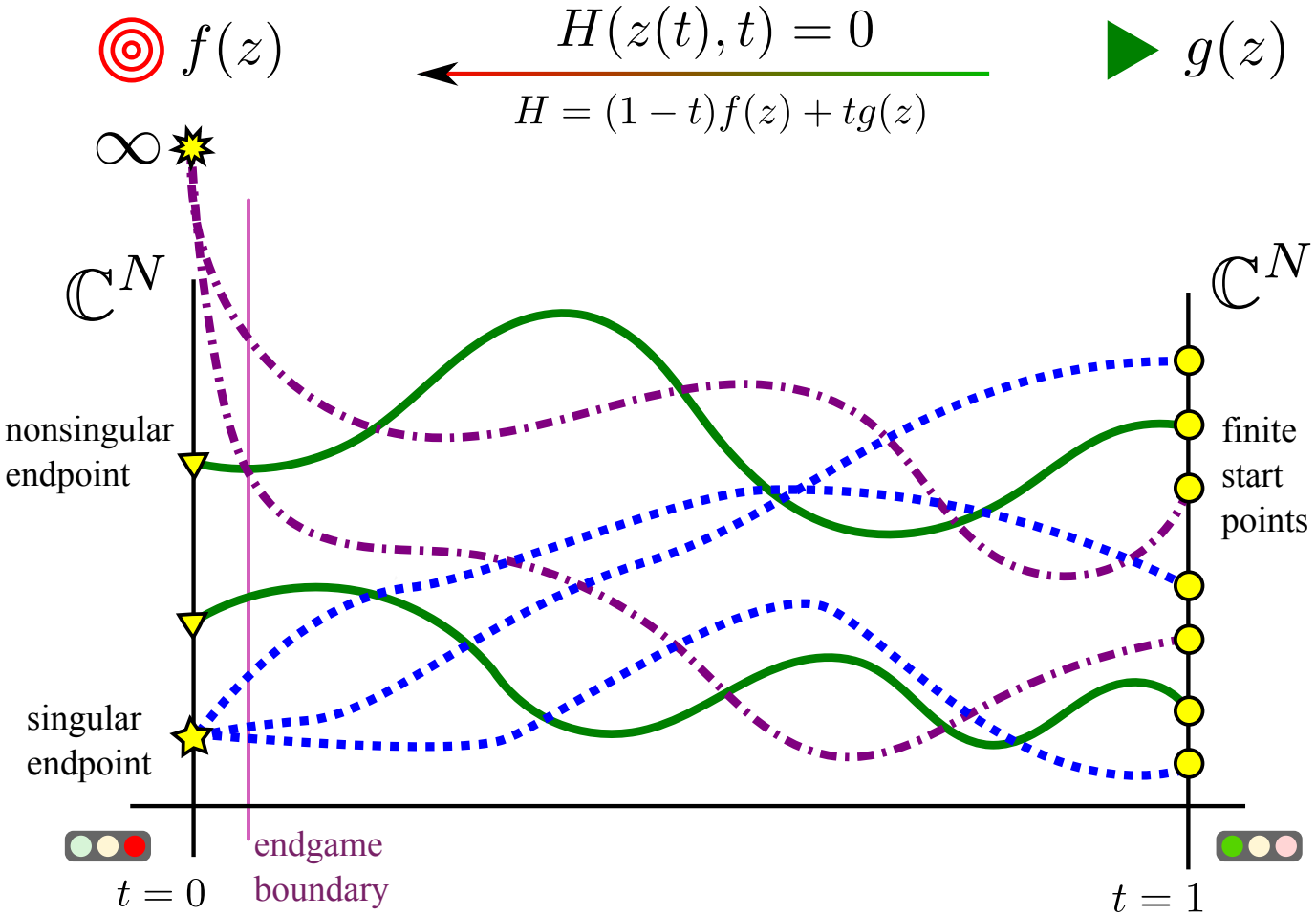
[2] Fu, Gonzalez, Lee. *Robotics : Control, Sensing, Vision, and Intelligence.* New York: McGraw-Hill, 1987

How to compute $m(x)$

We solve the inverse kinematics equations using *homotopy continuation*.

- Inverse kinematics equations are polynomial systems.
- Algebraic Geometry (AG) is the study of solving polynomial systems.
- Numerical AG mature enough to solve large systems numerically with high accuracy, quickly.
- We use the freely available package *Bertini*.
- Full 6D inverse equations have no closed form solution - solve numerically.
- Finds *all* isolated solutions.

Homotopy Continuation



Bertini and Paramotopy

The *Bertini* [3] software package is a sophisticated tool for solving polynomial systems with homotopy continuation. Some benefits and features include:

- Arbitrary and adaptive precision,
- Parameter and custom homotopy definitions,
- Positive-dimensional component detection and (complex) sampling,
- Parallel version with MPI,
- Free to use,
- New version Bertini2 under development, with many additions and improvements.

To compute the workspaces, we use *Paramotopy* [4], which lives on top of Bertini.

[3] Bates, Hauenstein, Sommese, Wampler. 2015. Available: http://bertini.nd.edu

[4] Brake, Bates, Niemerg. paramotopy.com

Optimization method

Input : DH parameters for each robot, λ,
samplings of δ, a, S
Output: Optimal $\delta_{\max}, a_{\max}$, $\Omega_\lambda(\delta_{\max}, a_{\max})$

1 Compute $W_1 = \{x \in S \mid m(x) > 0\}$;
2 Compute $W_\cap(\delta)$ using W_1;
3 Compute $W_f(\delta, a)$ using W_1;
4 Evaluate $\Omega_\lambda(\delta, a)$;
5 Find $\delta_{\max}, a_{\max}$ maximizing Ω_λ;
6 **return** Maximizers $\delta_{\max}, a_{\max}$ and value $\Omega_\lambda(\delta_{\max}, a_{\max})$;

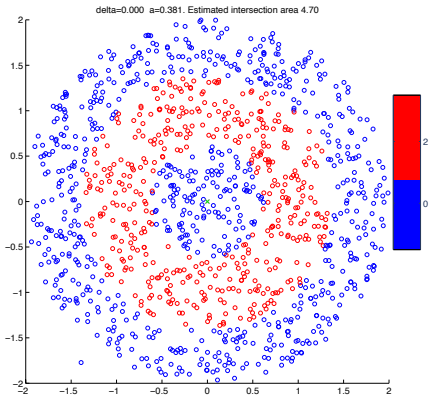
Monte Carlo Estimation

- For each of the four workspaces,
  - W_1, W_2, the prefailure independent workspaces,
  - $W_\cap$, the intersection of the prefailure workspaces, and
  - W_f, the cooperative postfailure workspace.

  we randomly sample space *guaranteed to contain* the workspace.

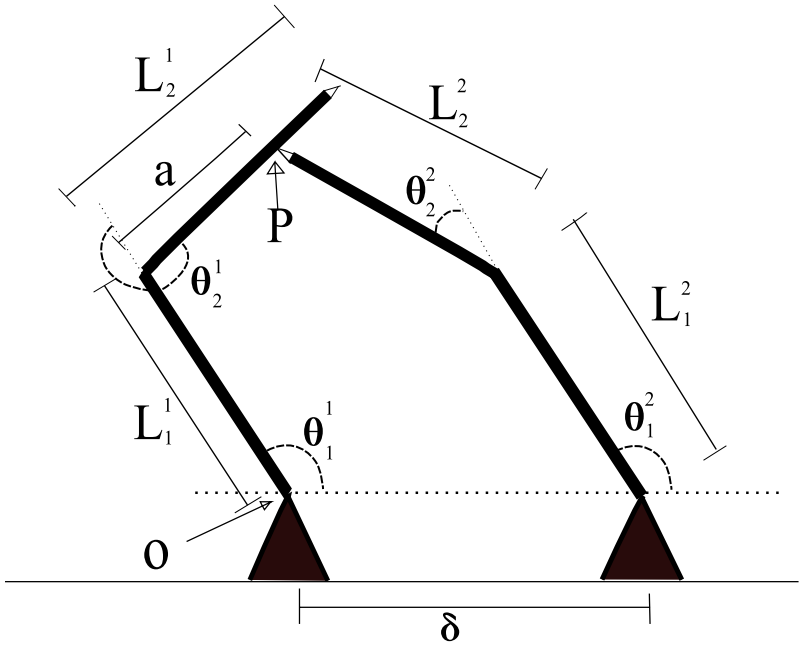
$$|W| = \lim_{r\to\infty} |S| \frac{1}{r} \sum_{j=1}^{r} m(x_j), \qquad (1)$$

where $|S|$ is the typical Euclidean measure of the sampling space.



2-Joint Planar Robot, Step 1

- Both robots identical, with all links of length 1. Then $W_1 = W_2$, translated by δ.
- Initial sampling of $[-2\,,2] \times [-2\,,2]$ for computing W_i.
- Inverse kinematics equations is a system of two equations in two variables (2 by 2), which we transform into a 4 by 4 system:



$$0 = \begin{cases} c_1 c_2 - s_1 s_2 + c_1 - x \\ s_1 c_2 + c_1 s_2 + s_1 - y \end{cases} . \tag{2}$$

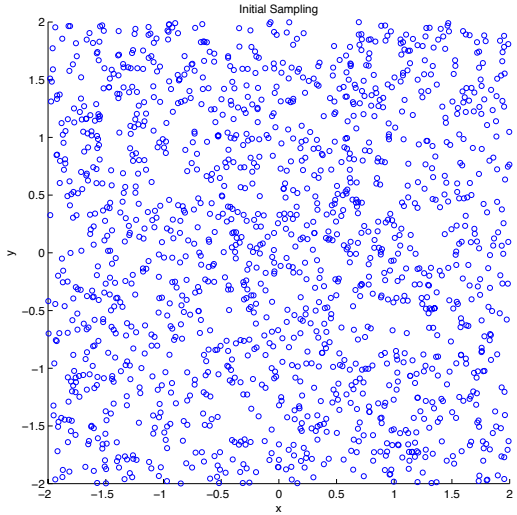
These equations are augmented by trigonometric identities,

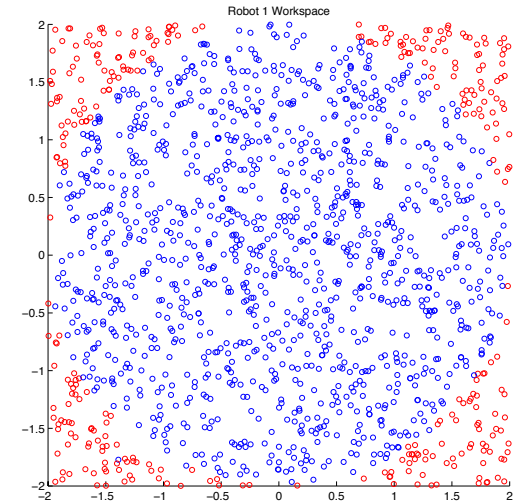
$$c_i^2 + s_i^2 - 1 = 0, \quad i = 1, 2. \tag{3}$$

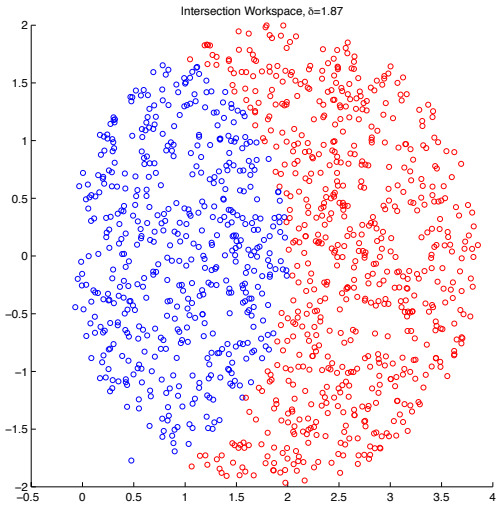
2-Joint Planar Robot, Step 2

Having W_1, next we compute $W_\cap$.

- Since $W_\cap \subset W_1$, throw out points from original sample that had no solutions.
- $W_\cap = W_\cap(\delta)$, so translate sample for each δ.
- Use same equations, solve with respect to robot 2.

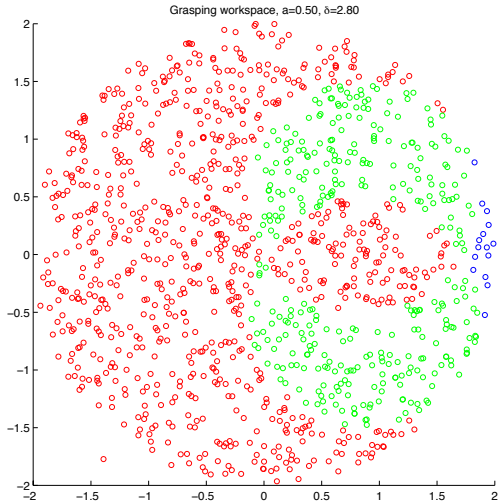






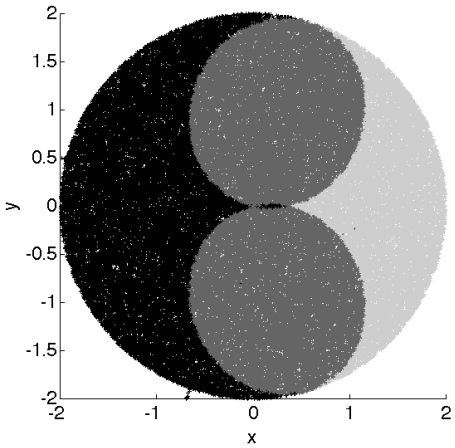
2-Joint Planar Robot, Step 3

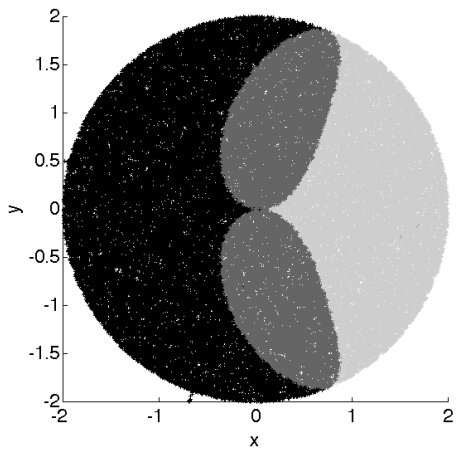
- Using $W_f \subset W_1$, solve new equations for grasping configuration.
- Solve for each pair of (δ, a) values.
- W_f system is 8 by 8:

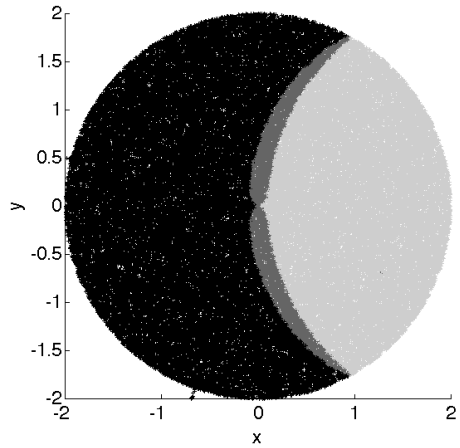


$$0 = \begin{cases} c_1 c_2 - s_1 s_2 + c_1 - x \\ s_1 c_2 + c_1 s_2 + s_1 - y \\ a c_1 c_2 - a s_1 s_2 + c_1 - (c_3 c_4 - s_3 s_4 + c_3 + \delta) \\ a s_1 c_2 + a c_1 s_2 + s_1 - (s_3 c_4 + c_3 s_4 + s_3) \\ s_i^2 + c_i^2 - 1 \end{cases}$$

W_F Examples





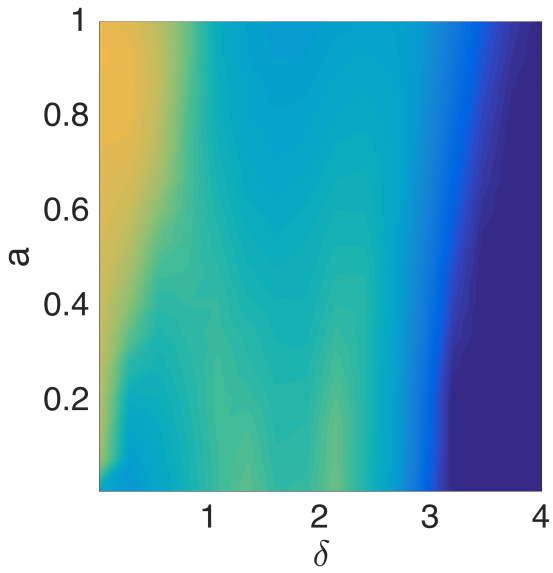


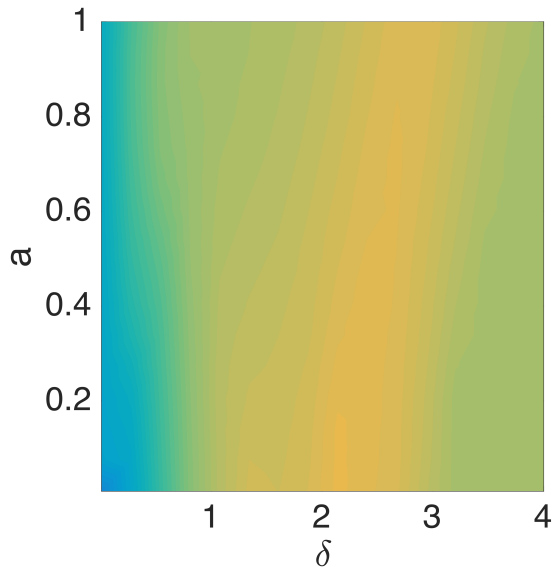
Cooperative workspaces for $\delta = 2$, $a = 0.1, 0.5, 0.9$.

Black: unreachable by robot 1 in cooperative mode, dark gray : two configurations reachable, light gray: four configurations reachable. White area is outside the workspace of robot 1.

$\Omega = (1-\lambda)\cdot|W_f| + \lambda(|W_1| - |W_\cap|)$

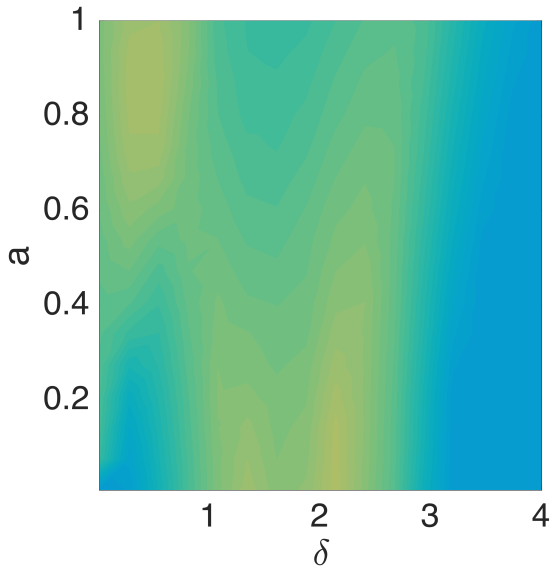
$\lambda = 0.$

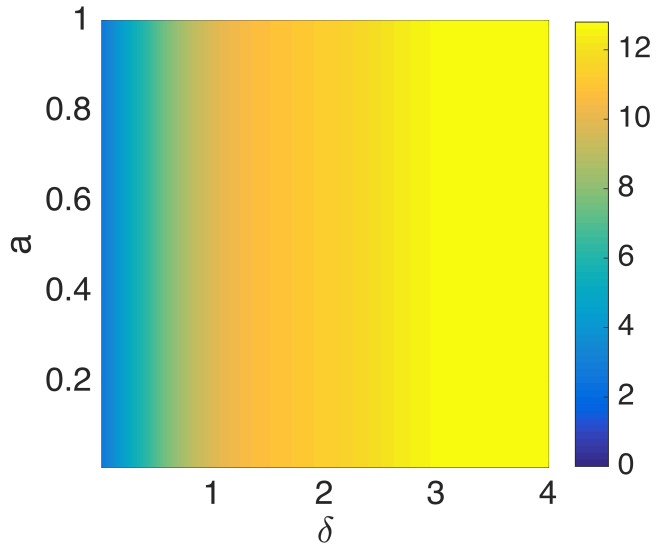




$\lambda = 2/3.$

$\lambda = 1/3.$



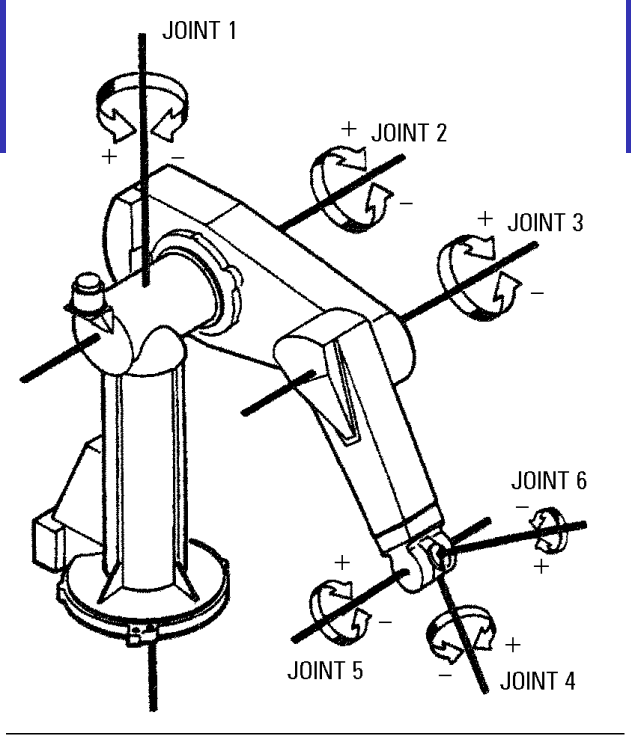


$\lambda = 1.$

Puma 5xx Robot

A fully 6D robot.

- First three joints control position,
- Last three, a wrist, controls orientation.



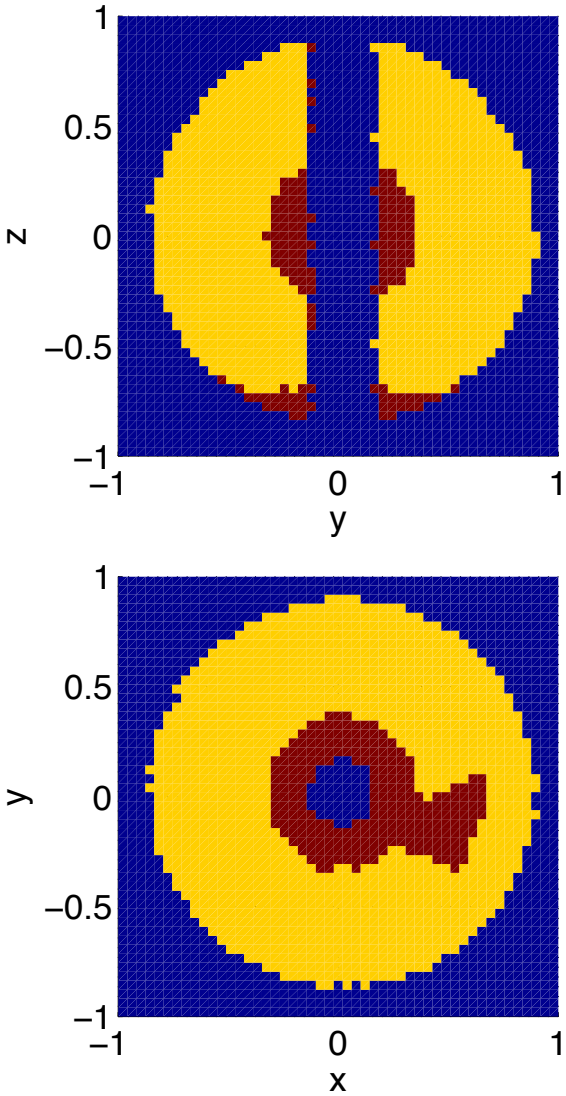
PUMA Denavit Hartenberg Parameters [5] , first three joints					
Joint i	θ_i	α_i	a_{i-1}	d_i	Type
1	θ_1	$-\pi/2$	0	243.5mm	R
2	θ_2	0	431.8 mm	-93.4 mm	R
3	θ_3	$\pi/2$	-20.32 mm	433.1mm	R

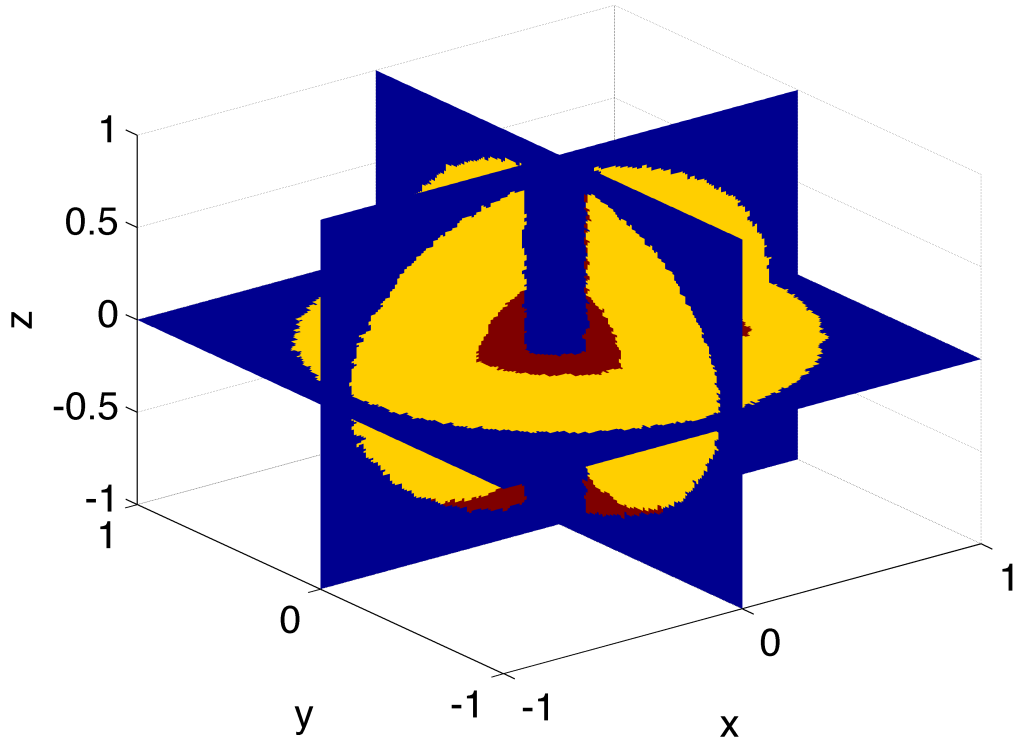
[5] Corke, Armstrong-Helouvry. Robotics and Automation. 1994

3D W_i Equations for Puma 500

$$0 = \begin{cases} 0.15005s_1 + 0.4318c_1c_2 + 0.4318c_1c_2c_3 - 0.4318c_1s_2s_3 - x \\ 0.4318c_2s_1 - 0.15005c_1 + 0.4318c_2c_3s_1 - 0.4318s_1s_2s_3 - y \\ 0.4318s_2 + 0.4318c_2s_3 + 0.4318c_3s_2 + 0.67 - z \\ s_i^2 + c_i^2 - 1 \end{cases}$$

One equation per dimension, plus the Pythagorean identity relating the sine and cosine pairs, for each $i = 1..3$.

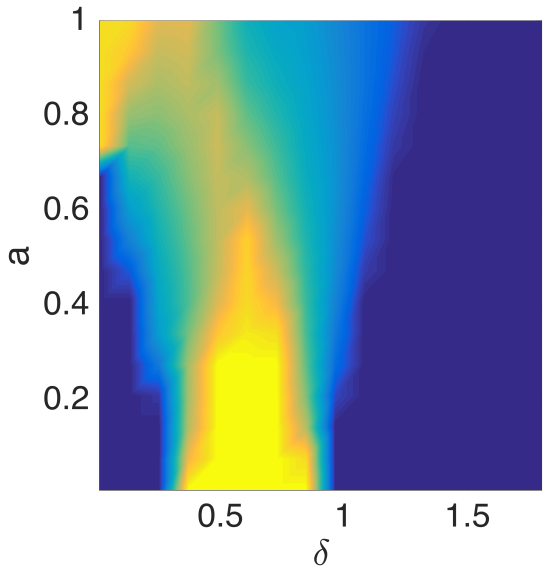


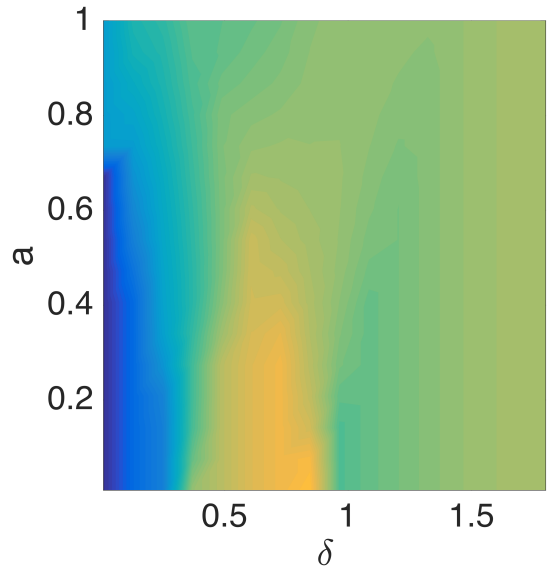


Left: Slices of the pre-failure PUMA workspace W_1 by the planes $x = 0$ (top) and $z = 0$ (bottom). Right: Combined picture of workspace with three slices $x = 0$, $y = 0$, $z = 0$. Yellow color: areas accessible the angles satisfying joint limits. Red color: real solutions violating joint limits. Dark blue region: inaccessible.

Objective Ω

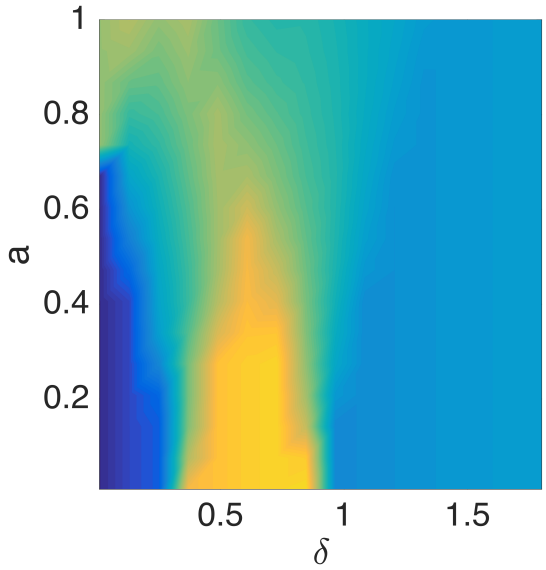
$\lambda = 0.$

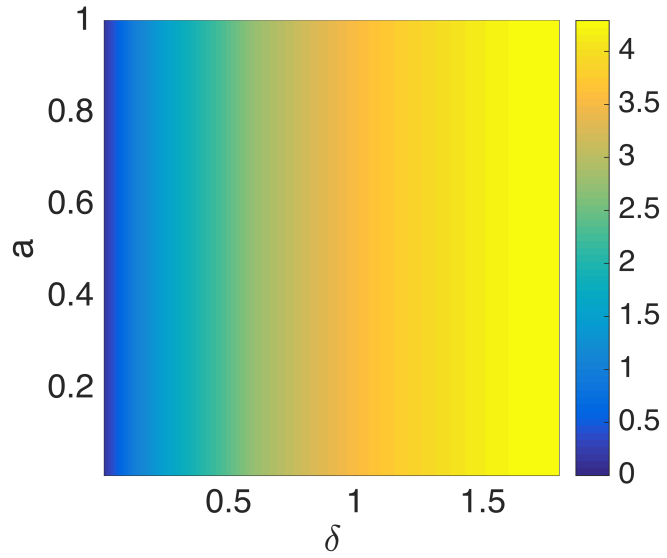




$\lambda = 2/3.$

$\lambda = 1/3.$





$\lambda = 1.$

Conclusion and outlook

- Inverse Kinematics equations may be turned into polynomial systems, and solved via homotopy continuation.
- Robot workspaces may be estimated via Monte Carlo methods - random sampling, and repeated solving of the inverse kinematics.
- Optimal placement of robot bases, and sockets on links, depends on Ω, which in turn depends on user's choice of weighting factor λ.
- Generalization to 6D robots will require uniform sampling of 3D rotations.