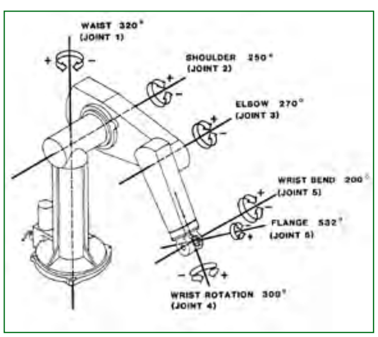
Robot Arm Transformations, Path Planning, and Trajectories

Robert Stengel
Robotics and Intelligent Systems
MAE 345, Princeton University, 2017

- **Forward and inverse kinematics**
- **Path planning**
  - Voronoi diagrams and Delaunay triangulation
  - Probabilistic Road Map
  - Rapidly Exploring Random Tree
- **Closed-form trajectories; connecting the dots**
  - Polynomials and splines
  - Acceleration profiles

Copyright 2017 by Robert Stengel. All rights reserved. For educational use only.
http://www.princeton.edu/~stengel/MAE345.html



Manipulator Maneuvering Spaces

- **Joint space**: Vector of joint variables, e.g.,

$$\mathbf{r}_J = \begin{bmatrix} \theta_{waist} & \theta_{shoulder} & \theta_{elbow} & \theta_{wrist-bend} & \theta_{flange} & \theta_{wrist-twist} \end{bmatrix}^T$$

- **End-effecter space**: Vector of end-effecter positions, e.g.,

$$\mathbf{r}_E = \begin{bmatrix} x_{tool} & y_{tool} & z_{tool} & \psi_{tool} & \theta_{tool} & \phi_{tool} \end{bmatrix}^T$$

- **Task space**: Vector of task-dependent positions, e.g., locating a symmetric grinding tool above a horizontal surface:

$$\mathbf{r}_T = \begin{bmatrix} x_{tool} & y_{tool} & z_{tool} & \psi_{tool} & \theta_{tool} \end{bmatrix}^T$$

Forward and Inverse Transformations of a Robotic Assembly

Forward Transformation

Transforms homogeneous coordinates from tool frame to reference frame coordinates

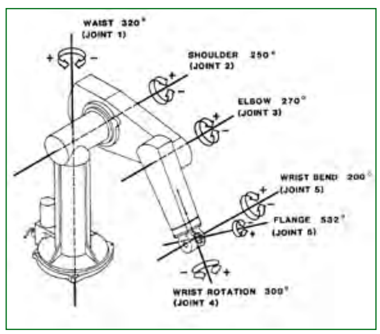
$$s_{base} = \mathbf{A}_{tool}^{base}\mathbf{s}_{tool}$$
$$= \mathbf{A}_{waist}\mathbf{A}_{shoulder}\mathbf{A}_{elbow}\mathbf{A}_{wrist-bend}\mathbf{A}_{flange}\mathbf{A}_{wrist-twist}\mathbf{s}_{tool}$$

Inverse Transformation

Transform homogeneous coordinate from reference frame to tool frame coordinates

$$s_{tool} = \mathbf{A}_{base}^{tool}\mathbf{s}_{base}$$
$$= \mathbf{A}^{-1}{}_{wrist-twist}\mathbf{A}^{-1}{}_{flange}\mathbf{A}^{-1}{}_{wrist-bend}\mathbf{A}^{-1}{}_{elbow}\mathbf{A}^{-1}{}_{shoulder}\mathbf{A}^{-1}{}_{waist}\mathbf{s}_{base}$$

Forward and Inverse Kinematics Between Joints, Tool Position, and Tool Orientation



Forward Kinematic Problem: Compute the position of the tool in the reference frame that corresponds to a given joint vector (i.e., vector of link variables)

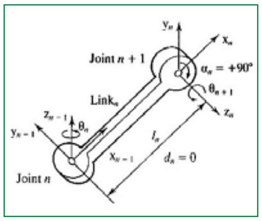
$$s_{base} = \mathbf{A}_{waist}\mathbf{A}_{shoulder}\mathbf{A}_{elbow}\mathbf{A}_{wrist-bend}\mathbf{A}_{flange}\mathbf{A}_{wrist-twist}\mathbf{s}_{tool} = \mathbf{A}_{tool}^{base}\mathbf{s}_{tool}$$

To Be Determined ⇐ Given

Inverse Kinematic Problem: Find the vector of link variables that corresponds to a desired task-dependent position

$$\mathbf{A}_{waist}\mathbf{A}_{shoulder}\mathbf{A}_{elbow}\mathbf{A}_{wrist-bend}\mathbf{A}_{flange}\mathbf{A}_{wrist-twist}\mathbf{s}_{tool} = \mathbf{A}_{tool}^{base}\mathbf{s}_0 = s_{base}$$

To Be Determined ⇐ Given



Forward and Inverse Kinematics
Single-Link Example

<u>Forward Kinematic Problem</u>: Specify task-dependent position that corresponds to a given joint variable (= θ_n)

$$\mathbf{s}_{n-1} = \mathbf{A}(z_{n-1}, \theta_n)\mathbf{A}(z_{n-1}, d_n)\mathbf{A}(x_{n-1}, l_n)\mathbf{A}(x_{n-1}, \alpha_n)\,\mathbf{s}_n$$

$$= \begin{bmatrix} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & l_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & l_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}\mathbf{s}_n$$

***Red*: Known**
Blue: Unknown

Forward and Inverse Kinematics
Single-Link Example

<u>Inverse Problem</u>: Find the joint variable, θ, that corresponds to a desired task-dependent position

$$\mathbf{s}_{n-1} = \mathbf{A}(z_{n-1}, \theta_n)\mathbf{A}(z_{n-1}, 0)\mathbf{A}(x_{n-1}, l_n)\mathbf{A}(x_{n-1}, 90°)\,\mathbf{s}_n$$

$$= \begin{bmatrix} \cos\theta_n & 0 & \sin\theta_n & l_n\cos\theta_n \\ \sin\theta_n & 0 & -\cos\theta_n & l_n\sin\theta_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\mathbf{s}_n$$

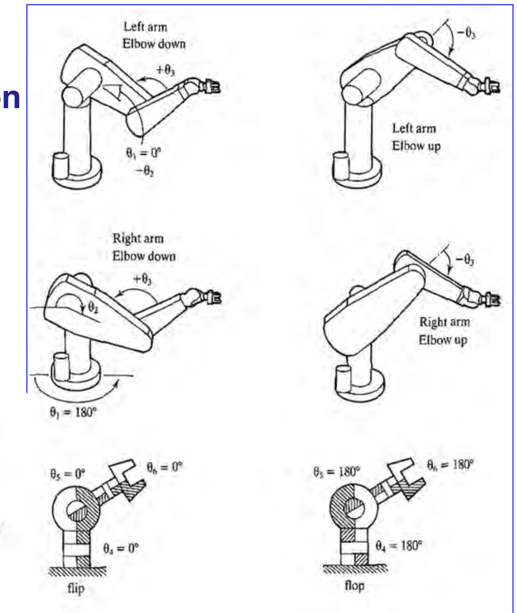
Red: Known
Blue: Unknown

$$x_{n-1} = x_n\cos\theta_n + z_n\sin\theta_n + l_n\cos\theta_n$$
$$y_{n-1} = x_n\sin\theta_n - z_n\cos\theta_n + l_n\sin\theta_n$$

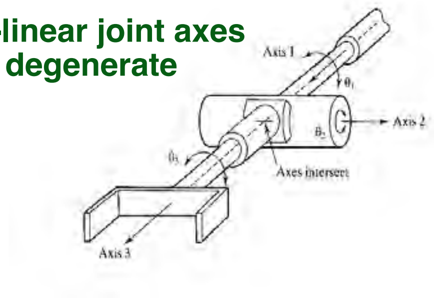
In this simple case,
check by elimination and inverse trig functions

Manipulator Redundancy and Degeneracy

- More than one link configuration may provide a given end point
- **Redundancy**: Finite number of joint vectors provide the same task-dependent vector
- **Degeneracy**: Infinite number of joint vectors provide the same task-dependent vector

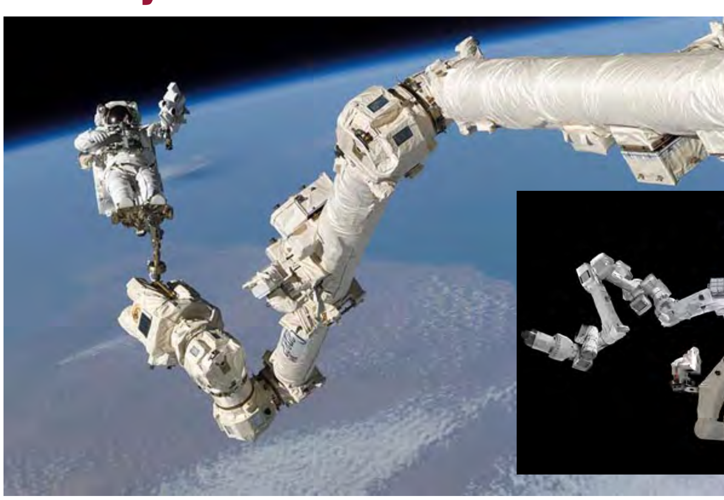


- Co-linear joint axes are degenerate



Space Robot Arms are Highly Redundant

- Why?



Link	variable	θ	α	l	d
1	θ_1	θ_1	0	l_1	0
2	θ_2	θ_2	0	l_2	0

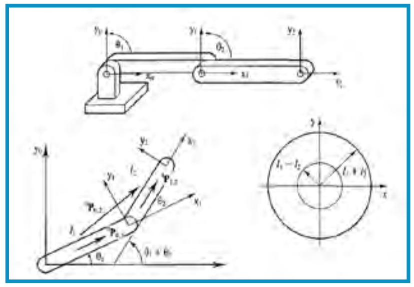
Transformations for a Two-Link Manipulator

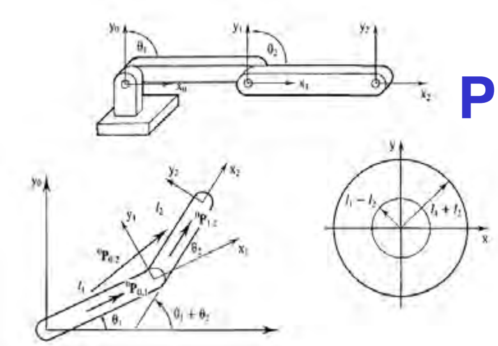
- **Example: Type 1 Two-Link Manipulator, neglecting offset (e.g., Puma geometry without waist and wrist)**

$$\mathbf{H}_0^1 = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 \\ -\sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad ; \quad \mathbf{r}_0 = \begin{bmatrix} -l_1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}_0^1 = \begin{bmatrix} \mathbf{H}_0^1 & \mathbf{r}_0 \\ \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 & -l_1 \\ -\sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{A}_1^2 = \begin{bmatrix} \mathbf{H}_1^2 & \mathbf{r}_1 \\ \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & 0 & -l_2 \\ -\sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$





Position of Distal Joint Relative to the Base

(2-link manipulator)

$$\theta_B = \theta_1 + \theta_2$$

$$\mathbf{s}_{base} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{base} = \mathbf{A}_1^0 \mathbf{A}_2^1 \mathbf{s}_{distal} = \begin{bmatrix} \cos\theta_B & -\sin\theta_B & 0 & l_1\cos\theta_1 + l_2\cos\theta_B \\ \sin\theta_B & \cos\theta_B & 0 & l_1\sin\theta_1 + l_2\sin\theta_B \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}_{distal}$$

$$= \begin{bmatrix} l_1\cos\theta_1 + l_2\cos\theta_B \\ l_1\sin\theta_1 + l_2\sin\theta_B \\ 0 \\ 1 \end{bmatrix}$$

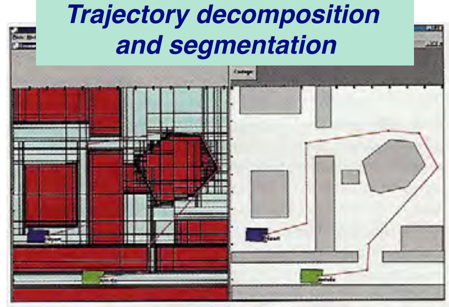
Path Planning

Baxter Path Planning (UNC, 2014)
https://www.youtube.com/watch?v=oY1FfytaD-c

Path Planning

Environment idealization and nominal path

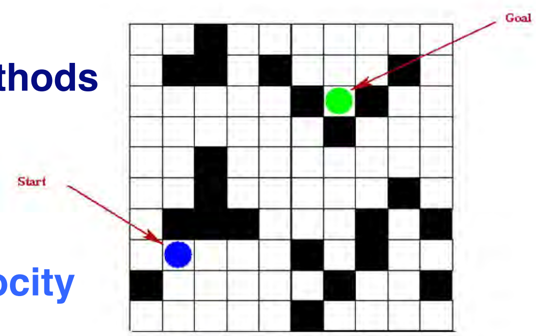
Trajectory decomposition and segmentation



- **Well-defined Start and Goal**
- **Waypoints**
- **Path primitives (line, circle, etc.)**
- **Timing and coordination**
- **Obstacle detection and avoidance**
- **Feasibility and regulation**
- **Optimization and constraint**

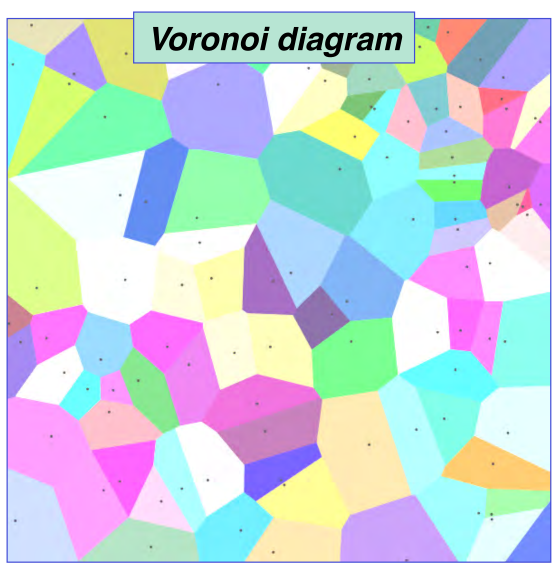
Path Planning with Waypoints

- Define Start, Goal, and Waypoints by position and time
- Connect the dots
- Various interpolation methods
  - Straight lines
  - Polynomials
  - Splines
- Generate associated velocity and acceleration
- Satisfy trajectory constraints

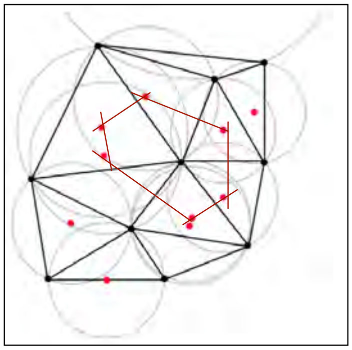


Path Planning with Obstacles and Destinations

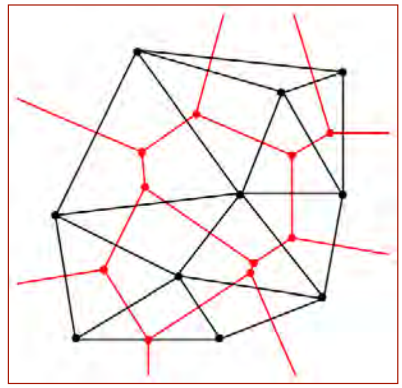
- Given set of points, e.g., obstacles, destinations, or centroids of multiple points
- Chart best path from start to goal
- Tessellation (tiling) of decision space
- 2-D Voronoi diagram
  – Polygons with sides equidistant to two nearest points (black dots)



Delaunay Triangulation
Constructs the Voronoi Diagram



- Threats/obstacles are **black** points
- Edges (**black**) connect all triplets of **black** points lying on circumferences of empty circles, *i.e., circles containing no other* ***black*** *points*
- "Circumcircle" centers are **red** points



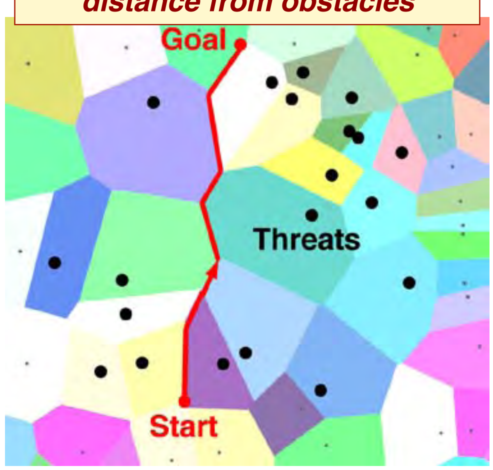
- Voronoi segment boundaries (**red**) connect centers and are perpendicular to each edge

https://en.wikipedia.org/wiki/Delaunay_triangulation

Voronoi Diagrams in Path Planning

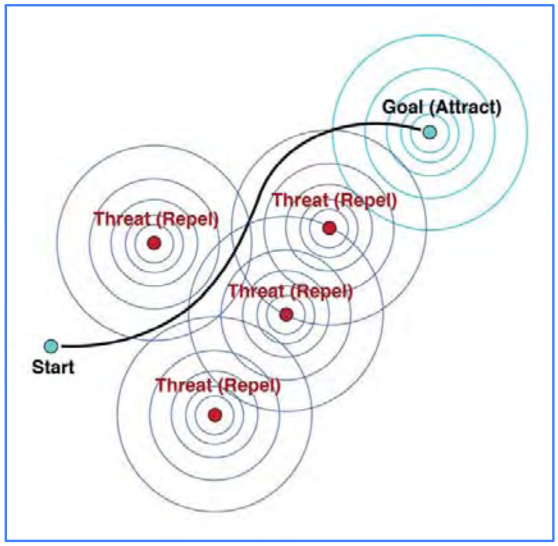
Threat/obstacle avoidance

Choose path with farthest distance from obstacles



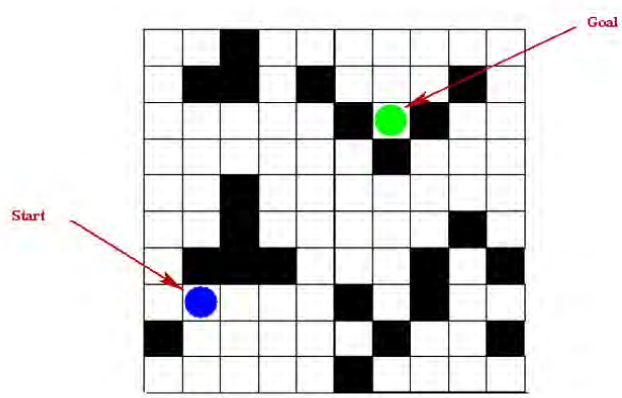
Path Planning with Potential Fields

Map features attract or repel path from Start to Goal, *e.g., +/– gravity fields*



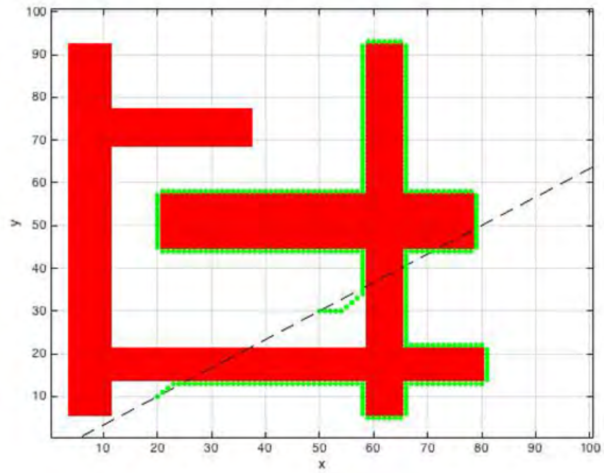
Path Planning on Occupancy Grid

Admissible and Inadmissible Blocks



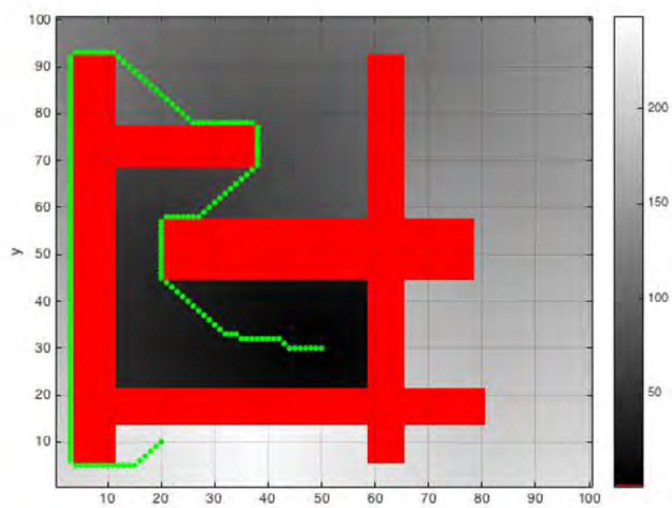
- Identify feasible paths from Start to Goal
- Chose path that best satisfies criteria, e.g.,
  - Simplicity of calculation
  - Lowest cost
  - Highest performance

Bug Path Planning



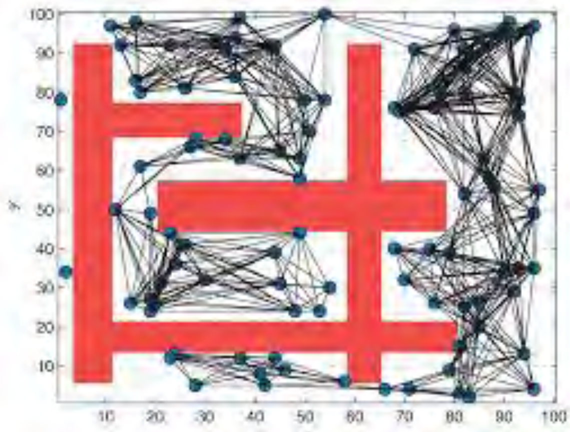
1) **Identify shortest unconstrained path from Start to Goal,** ***i.e., green path***
2) **Chose path that navigates the boundary**
    1) **Stays as close as to possible to unconstrained path (dashed line)**
    2) **Satisfies constraint**
    3) **Follows simple rule, e.g., "stay to the left"**

D* or A* Path Planning *(TBD)*



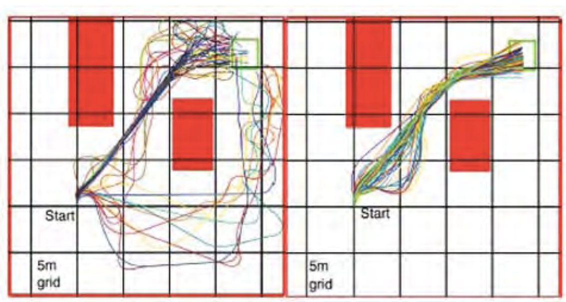
- **Determine *occupancy cost* of each block**
- **Chose path from Start to Goal that**
  - **Reduce occupancy cost with each step**

Probabilistic Road Map (PRM)



- Construct random configuration of admissible points
- Connect admissible points to nearest neighbors
- ***Assess incremental cost*** of traveling along each "edge" between points
- Query to find all feasible paths from Start to Goal
- ***Select lowest cost path***

Rapidly Exploring Random Tree (RRT*)



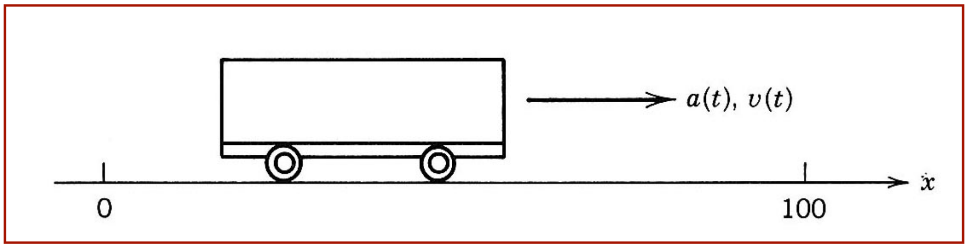
(a) RRT (b) RRT*

Space-filling tree evolves from Start
Open-loop trajectories with state constraints
Initially feasible solution converges to optimal solution through searching
Committed trajectories
Branch-and-bound tree adaptation

Trajectories

One-Dimensional Trajectory

Constant Velocity, *v*



Velocity, *v(t)* vs. *t*, is constant

$$v(t) = \dot{x}(t) = v(0)$$

Position, *x(t)* vs. *t*, is a straight line

$$x(t) = x(0) + v(0)t$$

One-Dimensional Trajectory

Constant Velocity, *v*

Position specified at 0 and *t*

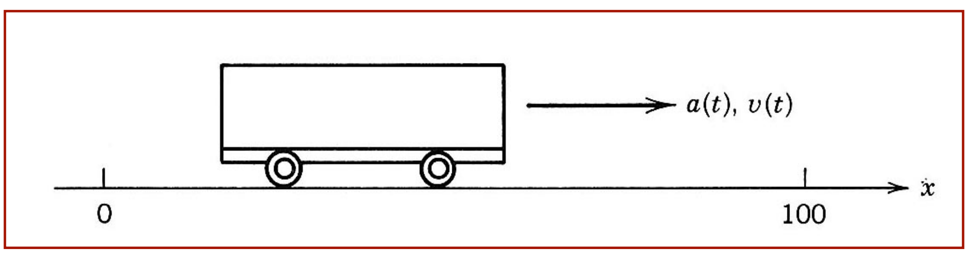
$$\begin{bmatrix} x(0) \\ x(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & t \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \end{bmatrix}$$

Velocity at 0 to be determined

$$\begin{bmatrix} x(0) \\ v(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & t \end{bmatrix}^{-1} \begin{bmatrix} x(0) \\ x(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1/t & 1/t \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \end{bmatrix}$$

One-Dimensional Trajectory

Constant Acceleration, *a*



Velocity, *v(t)* vs. *t*, is a straight line

$$v(t) = \dot{x}(t) = v(0) + at$$

Position, *x(t)* vs. *t*, is a parabola

$$x(t) = x(0) + v(t) + at^2/2$$

One-Dimensional Trajectory

Constant Acceleration, *a*

Position specified at 0 and *t*; velocity specified at 0

$$\begin{bmatrix} x(0) \\ x(t) \\ v(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & t & t^2/2 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \end{bmatrix}$$

Acceleration at 0 to be determined

$$\begin{bmatrix} x(0) \\ v(0) \\ a(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & t & t^2/2 \\ 0 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ -2/t^2 & 2/t^2 & -2/t \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \end{bmatrix}$$

One-Dimensional Trajectory

Constant Jerk, *j*, = Derivative of Acceleration, *a*

Acceleration, *a(t)* vs. *t*, is a straight line

$$a(t) = \dot{v}(t) = \ddot{x}(t) = a(0) + jt$$

Velocity, *v(t)* vs. *t*, is a parabola

$$v(t) = \dot{x}(t) = v(0) + a(0)t + jt^2/2$$

Position, *x(t)* vs. *t*, is cubic

$$x(t) = x(0) + v(0)t + a(0)t^2/2 + jt^3/6$$

One-Dimensional Trajectory

Constant Jerk, *j*

Position and **velocity specified at 0 and *t***;
acceleration and jerk at **0** to be determined

$$\begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & t & t^2/2 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j \end{bmatrix}$$

One-Dimensional Trajectory

Constant Jerk, *j*

Find *a*(0) and *j* to produce desired position and velocity

Start / Finish / Start / Finish

$$\begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & t & t^2/2 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j \end{bmatrix}$$

Start / Start / *TBD* / *TBD*

Inverse of (4 x 4) relationship defines required *a*(0) and *j*

$$\begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & t & t^2/2 \end{bmatrix}^{-1} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -6/t^2 & 6/t^2 & -4/t & -2/t \\ 12/t^3 & -12/t^3 & 6/t^2 & 6/t^2 \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \end{bmatrix}$$

Further Derivatives

- **Snap, *s*, = Derivative of Jerk, *j***
- **Crackle, *c*, = Derivative of Snap, *s***
- **What is the derivative of Crackle?**

Pop!

31

One-Dimensional Trajectory

Constant Crackle, *c*

Snap, *s(t)* vs. *t*, is linear in time

$$s(t) = d\left[j(t)\right]/dt = +s(0) + ct$$

Jerk, *j(t)* vs. *t*, is quadratic

$$j(t) = \dot{a}(t) = j(0) + s(0)t + ct^2/2$$

Acceleration, *a(t)* vs. *t*, is cubic

$$a(t) = \dot{v}(t) = \ddot{x}(t) = a(0) + j(0)t + s(0)t^2/2 + ct^3/6$$

One-Dimensional Trajectory
with Constant Crackle, *c*

Velocity, *v(t)* vs. *t*, is quartic

$$v(t) = \dot{x}(t) = v(0) + a(0)t + jt^2/2 + s(0)t^3/6 + ct^4/24$$

Position, *x(t)* vs. *t*, is quintic

$$x(t) = x(0) + v(0)t + a(0)t^2/2 + j(0)t^3/6 + s(0)t^4/24 + ct^5/120$$

One-Dimensional Trajectory
with Constant Crackle, *c*

Position, velocity, and acceleration specified at 0 and *t*

$$\begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & t & t^2/2 & t^3/6 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c \end{bmatrix}$$

One-Dimensional Trajectory

Inverse of (6 x 6) relationship defines controls

$$\begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & t & t^2/2 & t^3/6 \end{bmatrix}^{-1} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \end{bmatrix}$$

$$\begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ -60/t^3 & 60/t^3 & -36/t^2 & -24/t^2 & -9/t & 3/t \\ 360/t^4 & -360/t^4 & 192/t^3 & 168/t^3 & 36/t^2 & -24/t^2 \\ -720/t^5 & 720/t^5 & -360/t^4 & -360/t^4 & -60/t^3 & 60/t^3 \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \end{bmatrix}$$

One-Dimensional Trajectory

Eliminate unnecessary equations and define acceleration constants

$$\begin{bmatrix} j(0) \\ s(0) \\ c \end{bmatrix} = \begin{bmatrix} -60/t^3 & 60/t^3 & -36/t^2 & -24/t^2 & -9/t & 3/t \\ 360/t^4 & -360/t^4 & 192/t^3 & 168/t^3 & 36/t^2 & -24/t^2 \\ -720/t^5 & 720/t^5 & -360/t^4 & -360/t^4 & -60/t^3 & 60/t^3 \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \end{bmatrix}$$

Corresponding acceleration and force are specified by

$$a(t) = a(0) + j(0)t + s(0)t^2/2 + ct^3/6$$

$$= a_{control}(t) + a_{gravity}(t) + a_{disturbance}(t)$$

$$= \left[f_{control}(t) + f_{gravity}(t) + f_{disturbance}(t) \right] / m(t)$$

One-Dimensional Trajectory

Calculate trajectory components, given acceleration constants

$$\begin{bmatrix} x(t) \\ v(t) \\ a(t) \end{bmatrix} = \begin{bmatrix} 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 \\ 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 \\ 0 & 0 & 1 & t & t^2/2 & t^3/6 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c \end{bmatrix}$$

Example

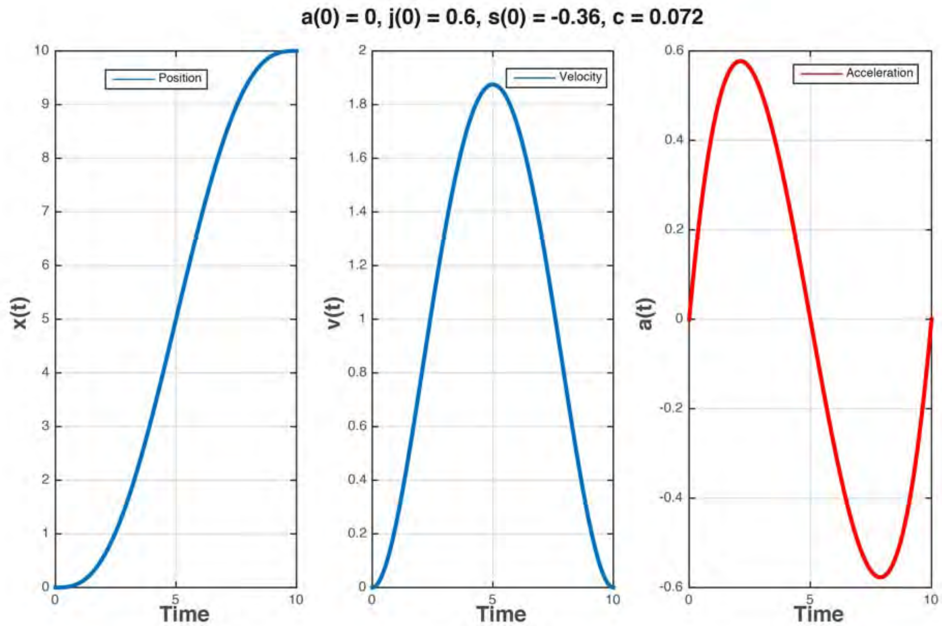
Calculate constants for x(0) = 0, x(10) = 10

$$\begin{bmatrix} 0.6 \\ -0.36 \\ 0.072 \end{bmatrix} = \begin{bmatrix} -60/10^3 & 60/10^3 & -36/10^2 & -24/10^2 & -9/10 & 3/10 \\ 360/10^4 & -360/10^4 & 192/10^3 & 168/10^3 & 36/10^2 & -24/10^2 \\ -720/10^5 & 720/10^5 & -360/10^4 & -360/10^4 & -60/10^3 & 60/10^3 \end{bmatrix} \begin{bmatrix} 0 \\ 10 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Calculate trajectory, given constants for $t_f = 10$

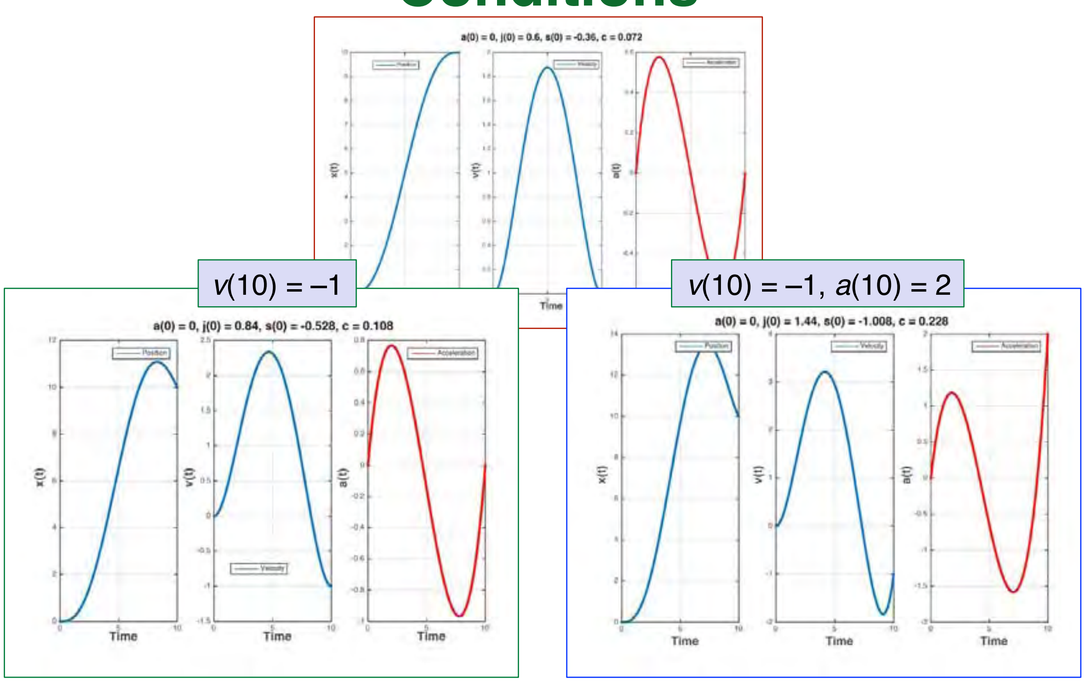
$$\begin{bmatrix} x(t) \\ v(t) \\ a(t) \end{bmatrix} = \begin{bmatrix} 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 \\ 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 \\ 0 & 0 & 1 & t & t^2/2 & t^3/6 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.6 \\ -0.36 \\ 0.072 \end{bmatrix}$$

1-D Example



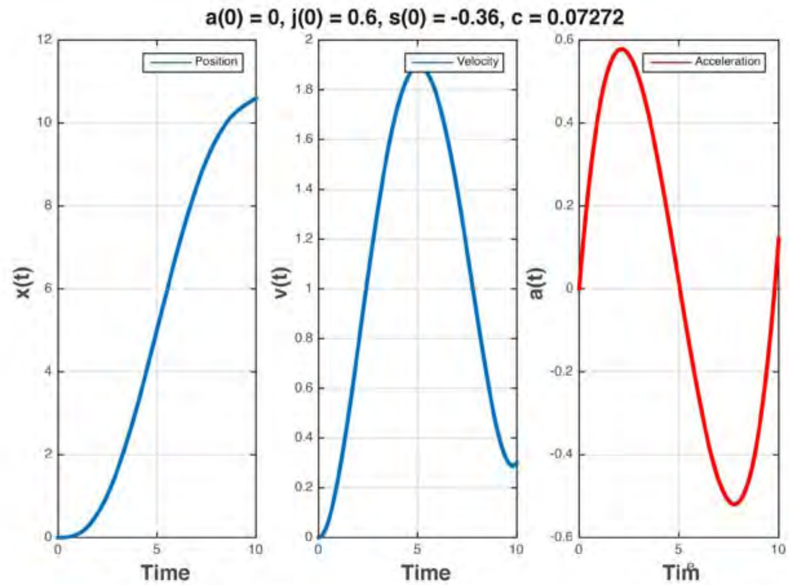
$$a_{net}(t) = (0) + 0.6t - 0.36t^2/2 + 0.072t^3/6$$

Examples with Different End Conditions

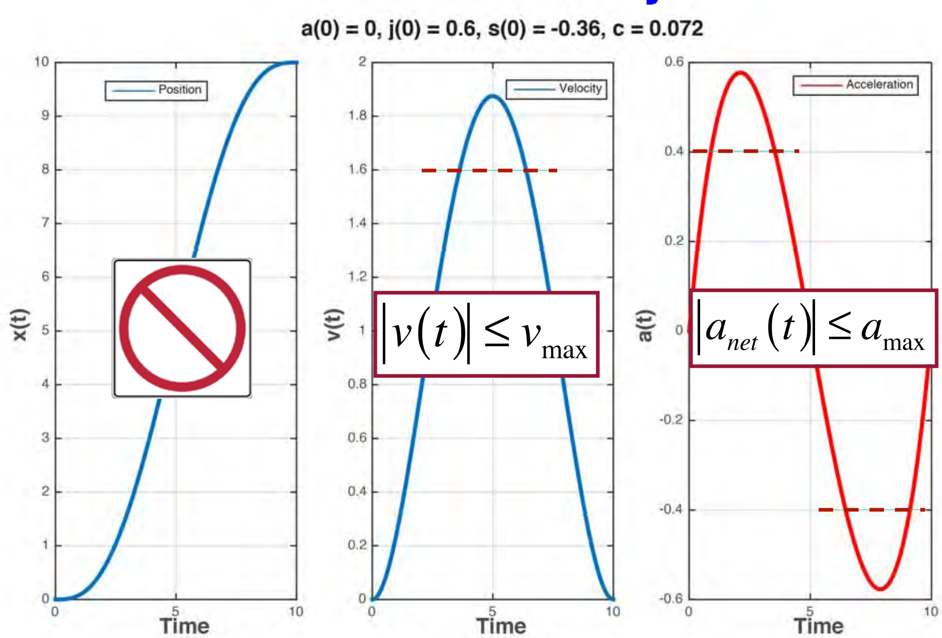


Sensitivity to Errors

1% error in Crackle



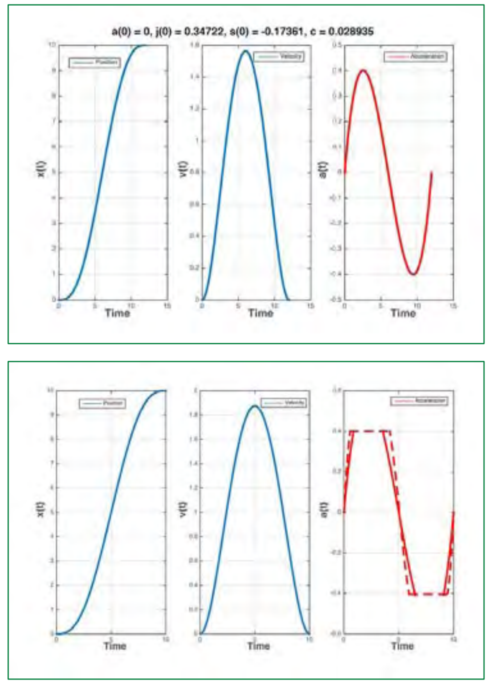
Constrained 1-D Trajectories



What are the alternatives for achieving desired end conditions?

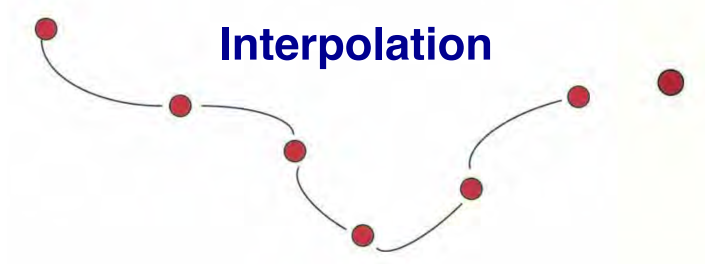
Alternatives for Reaching End Position

- Increase end time
  - Lower max/min values of velocity and acceleration
- "Fatten" velocity and acceleration profiles
  - Multi-segment trajectory
  - Unconstrained arcs
  - Constrained arcs (velocity and/or acceleration held constant



Connect the Dots

Interpolation

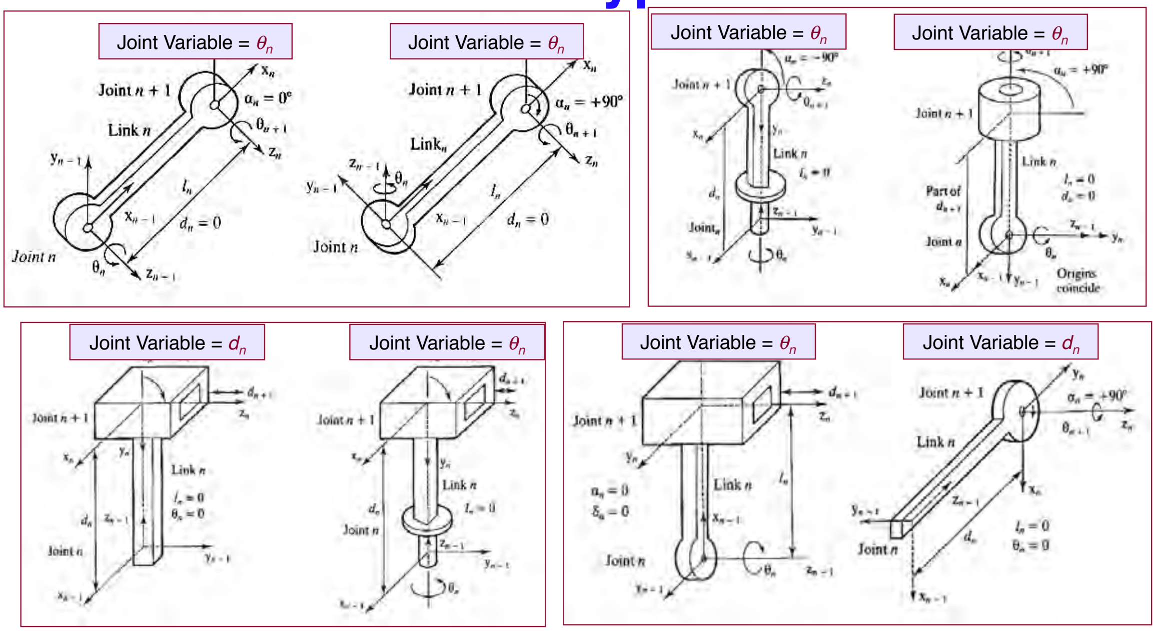


- Piecewise polynomials (linear -> quintic)
  - End-point discontinuities
  - End-point constraints
  - Parabolic blend
- Single polynomial through all points
  - Polynomial degree = # of points
  - Sensitivity to high-degree terms (e.g., ct^6)
  - Possibility of large excursions between points
- Polynomials through adjacent points
  - e.g., cubic B splines
  - Kriging

Next Time:
Time Response of Dynamic Systems

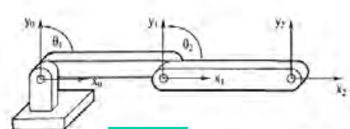
Supplemental Material

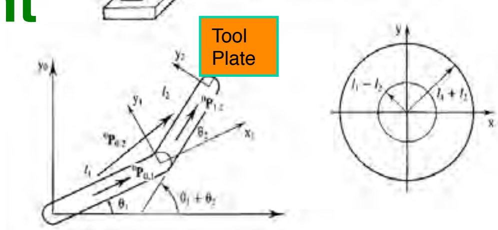
Joint Variables for Different Link Types



Position of Distal Joint Relative to the Base

(2-link manipulator)





- Suppose a **tool plate** is fixed to the distal joint at $(x\ y\ z)_{distal}^T$; then

$$\mathbf{s}_{base} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{base} = \mathbf{A}_1^0 \mathbf{A}_2^1 \mathbf{s}_{distal} = \begin{bmatrix} \cos\theta_B & -\sin\theta_B & 0 & l_1\cos\theta_1 + l_2\cos\theta_B \\ \sin\theta_B & \cos\theta_B & 0 & l_1\sin\theta_1 + l_2\sin\theta_B \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{distal}$$

$$= \begin{bmatrix} x\cos\theta_B - y\sin\theta_B + l_1\cos\theta_1 + l_2\cos\theta_B \\ x\sin\theta_B + y\cos\theta_B + l_1\sin\theta_1 + l_2\sin\theta_B \\ z \\ 1 \end{bmatrix}$$

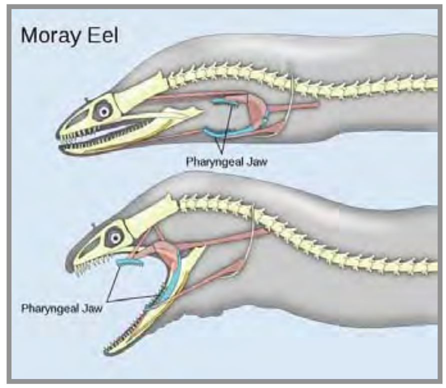
Alternatively, straightforward trigonometry could be used in this example

Tool Plates and Jaws

Tool Changer
http://www.youtube.com/watch?v=G8ZqoOIEDHY&feature=related

Another Tool Changer
http://www.youtube.com/watch?v=LkPnt_nudLc&feature=related

Four-Bar Linkage and 2nd Set of Jaws



Robot Arms for Space

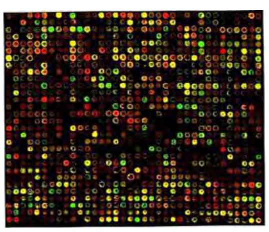
Multi-Jointed Arms

Snake-Like Manipulator

Octopus Arms

OctArm
http://www.youtube.com/watch?v=Qzvqni7O_XQs

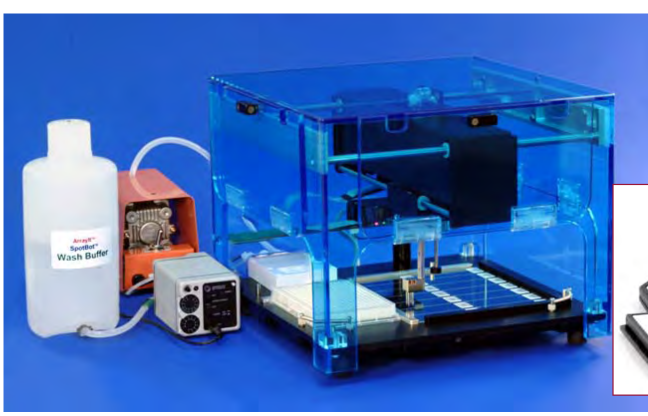
Tentacle Arm
http://www.youtube.com/watch?v=Yk7Muaigd4k



DNA Microarray-Spotting Robot

- **DNA strands representing different genes are spotted on a microscope slide**
- **Finished slide is used to analyze DNA from tissue samples**

http://www.youtube.com/watch?v=Z_KNhD1jz-k



American Android Multi-Arm UGV

(David Handelman, *89)

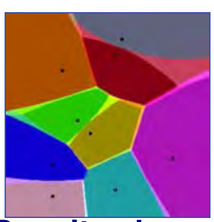
http://www.youtube.com/watch?v=pOi6OdcPKfk

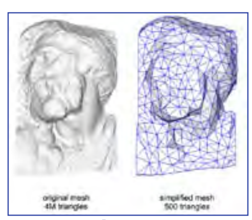
http://www.youtube.com/watch?v=tVZFJ7yivxI

http://www.youtube.com/watch?v=qdM48cAg0U4

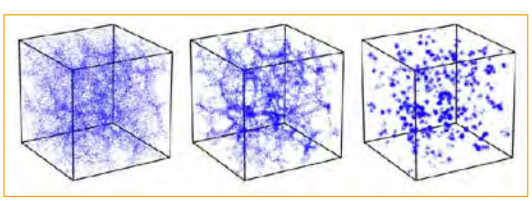
Voronoi Diagrams in Data Processing

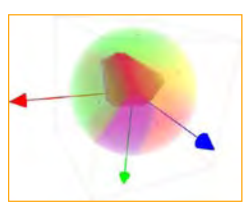
Computer graphics textures (2-D and 3-D meshes)





Density characterization (3-D mesh)





Vector quantization in data compression

http://www.data-compression.com/vqanim.shtml

One-Dimensional Trajectory
with Constant Kix, *k*

Position, velocity, acceleration, and jerk specified at 0 and *t*

$$\begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c(0) \\ p(0) \\ k(0) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1800/(19t^4) & 1800/(19t^4) & -1680/(19t^3) & -120/(19t^3) & -36/(t^2) & -96/(19t^2) & -156/(19t) & 12/(19t) \\ 14400/(19t^5) & -14400/(19t^5) & 11160/(19t^4) & 3240/(19t^4) & 192/(t^3) & 312/(19t^3) & 564/(19t^2) & -96/(19t^2) \\ -50400/(19t^6) & 50400/(19t^6) & -28800/(19t^5) & -21600/(19t^5) & -360/(t^4) & 3240/(19t^4) & -720/(19t^3) & -120/(19t^3) \\ 72000/(19t^7) & -72000/(19t^7) & 21600/(19t^6) & 50400/(19t^6) & 0 & -14400/(19t^5) & -600/(19t^4) & 1800/(19t^4) \end{bmatrix} \begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \\ j(0) \\ j(t) \end{bmatrix}$$

Snap, crackle, pop, and kix computed

One-Dimensional Trajectory

Inverse of (8 x 8) relationship defines controls

$$\begin{bmatrix} x(0) \\ x(t) \\ v(0) \\ v(t) \\ a(0) \\ a(t) \\ j(0) \\ j(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 & t^6/600 & t^7/3600 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 & t^5/120 & t^6/600 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & t & t^2/2 & t^3/6 & t^6/600 & t^5/120 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & t & t^2/2 & t^3/6 & t^4/24 \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \\ a(0) \\ j(0) \\ s(0) \\ c(0) \\ p(0) \\ k(0) \end{bmatrix}$$