Motion Planning

METR 4202: Advanced Control & **Robotics**

Dr Surya Singh
Lecture # 11

October 11, 2013

metr4202@itee.uq.edu.au

http://itee.uq.edu.au/~metr4202/

Schedule

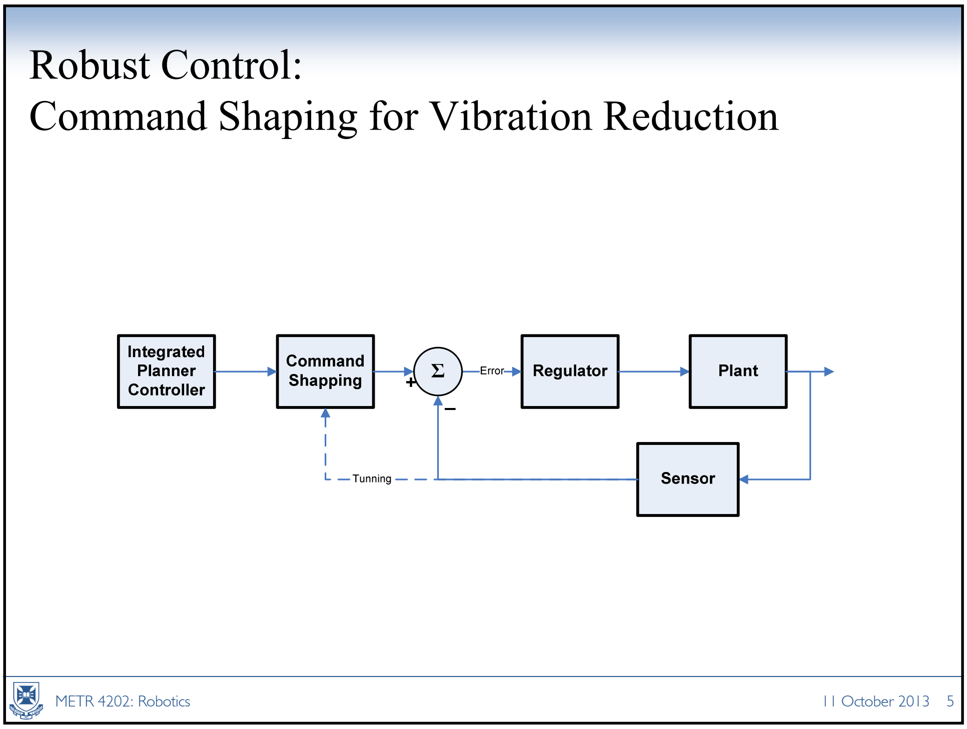
Week	Date	Lecture (F: 9-10:30, 42-212)
1	26-Jul	Introduction
2	2-Aug	Representing Position & Orientation & State (Frames, Transformation Matrices & Affine Transformations)
3	9-Aug	Robot Kinematics
4	16-Aug	Robot Dynamics & Control
5	23-Aug	Robot Trajectories & Motion
6	30-Aug	Sensors & Measurement
7	6-Sep	Perception / Computer Vision
8	13-Sep	Localization and Navigation
9	20-Sep	State-Space Modelling
10	27-Sep	State-Space Control
	4-Oct	*Study break*
11	**11-Oct**	**Motion Planning**
12	18-Oct	Vision-based control (+ Prof. P. Corke or Prof. M. Srinivasan)
13	25-Oct	Applications in Industry (+ Prof. S. LaValle) & Course Review

Announcements: We're Working On It!

- **Lab 3:**
  - Working on it!
  - Due: October 24 (!!), "Free Extension" to October 31.
- **Take Home "Quiz":**
  - Working on it!
  - Designing it so that you can do it in less than 24 hours if you know the question
  - Via Platypus: Scanning of Handwritten solutions okay
- Cool Robotics Share Site
  ➔ http://metr4202.tumblr.com/

Cool Robotics Video Share

From Last Week: Command Shaping



Robust Control:
Command Shaping for Vibration Reduction

Integrated Planner Controller

Command Shapping

Σ

+

−

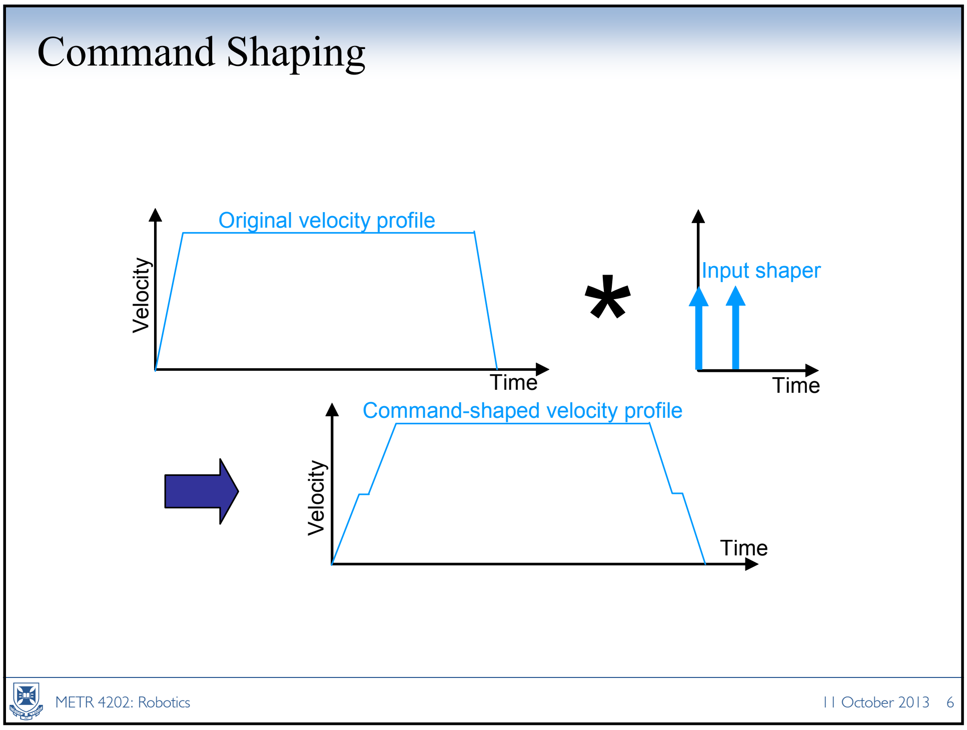
Error

Regulator

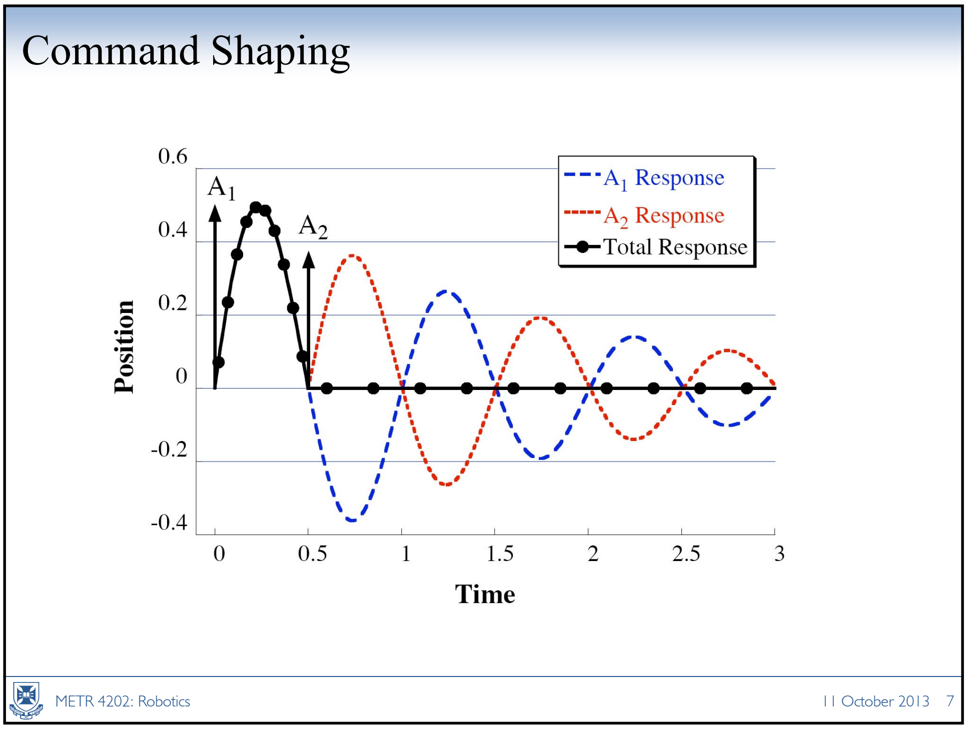
Plant

Sensor

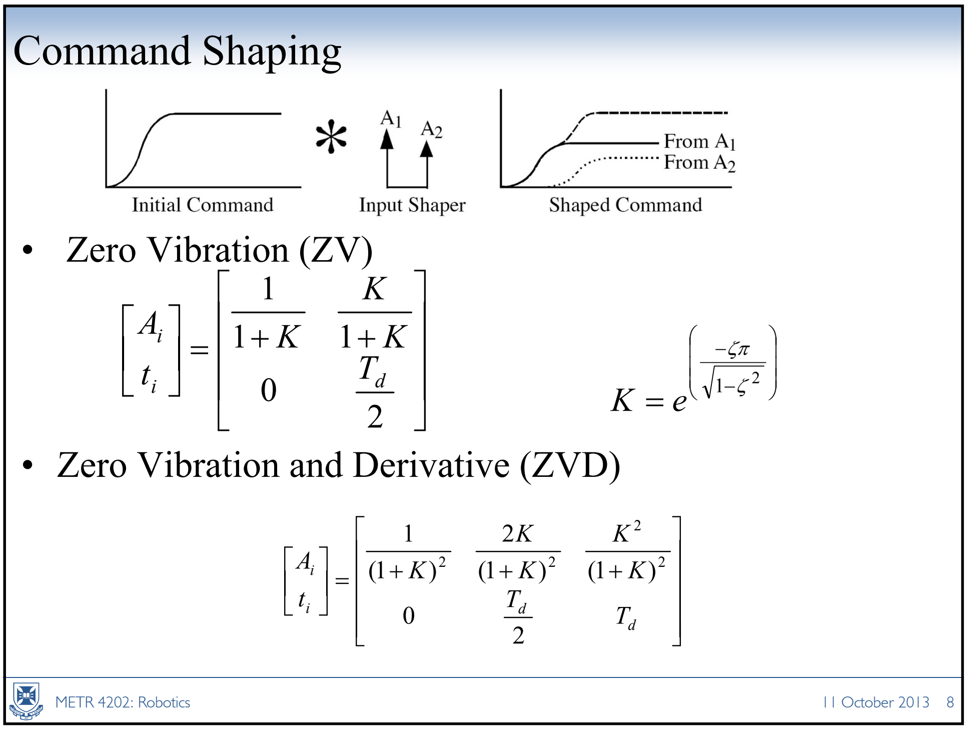
Tunning



Command Shaping



Command Shaping



Command Shaping

Initial Command

Input Shaper

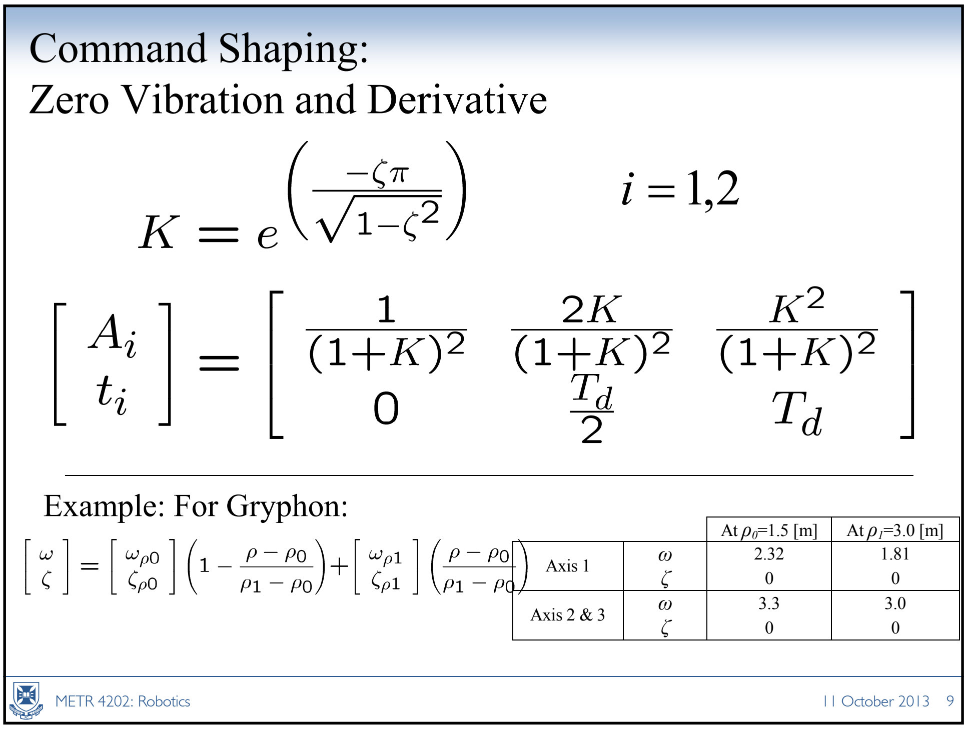
Shaped Command

- Zero Vibration (ZV)

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \dfrac{1}{1+K} & \dfrac{K}{1+K} \\ 0 & \dfrac{T_d}{2} \end{bmatrix} \qquad K = e^{\left(\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}\right)}$$

- Zero Vibration and Derivative (ZVD)

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \dfrac{1}{(1+K)^2} & \dfrac{2K}{(1+K)^2} & \dfrac{K^2}{(1+K)^2} \\ 0 & \dfrac{T_d}{2} & T_d \end{bmatrix}$$



Command Shaping: Zero Vibration and Derivative

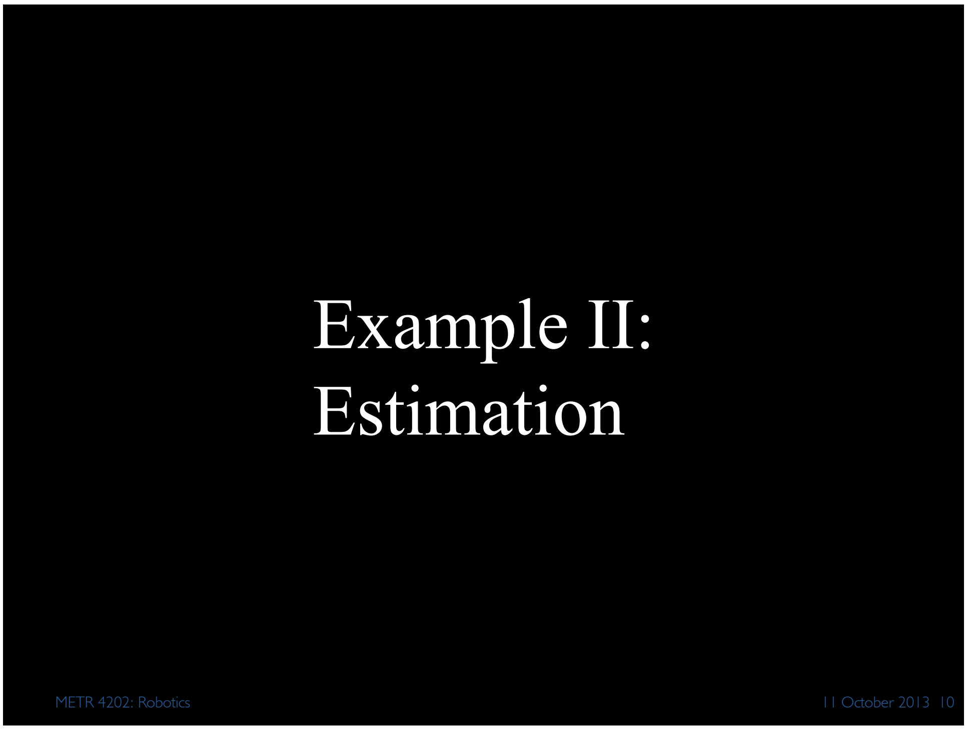
$$K = e^{\left(\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}\right)} \qquad i = 1,2$$

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \frac{1}{(1+K)^2} & \frac{2K}{(1+K)^2} & \frac{K^2}{(1+K)^2} \\ 0 & \frac{T_d}{2} & T_d \end{bmatrix}$$

Example: For Gryphon:

$$\begin{bmatrix} \omega \\ \zeta \end{bmatrix} = \begin{bmatrix} \omega_{\rho 0} \\ \zeta_{\rho 0} \end{bmatrix} \left(1 - \frac{\rho - \rho_0}{\rho_1 - \rho_0}\right) + \begin{bmatrix} \omega_{\rho 1} \\ \zeta_{\rho 1} \end{bmatrix} \left(\frac{\rho - \rho_0}{\rho_1 - \rho_0}\right)$$

		At ρ_0=1.5 [m]	At ρ_1=3.0 [m]
Axis 1	ω	2.32	1.81
	ζ	0	0
Axis 2 & 3	ω	3.3	3.0
	ζ	0	0

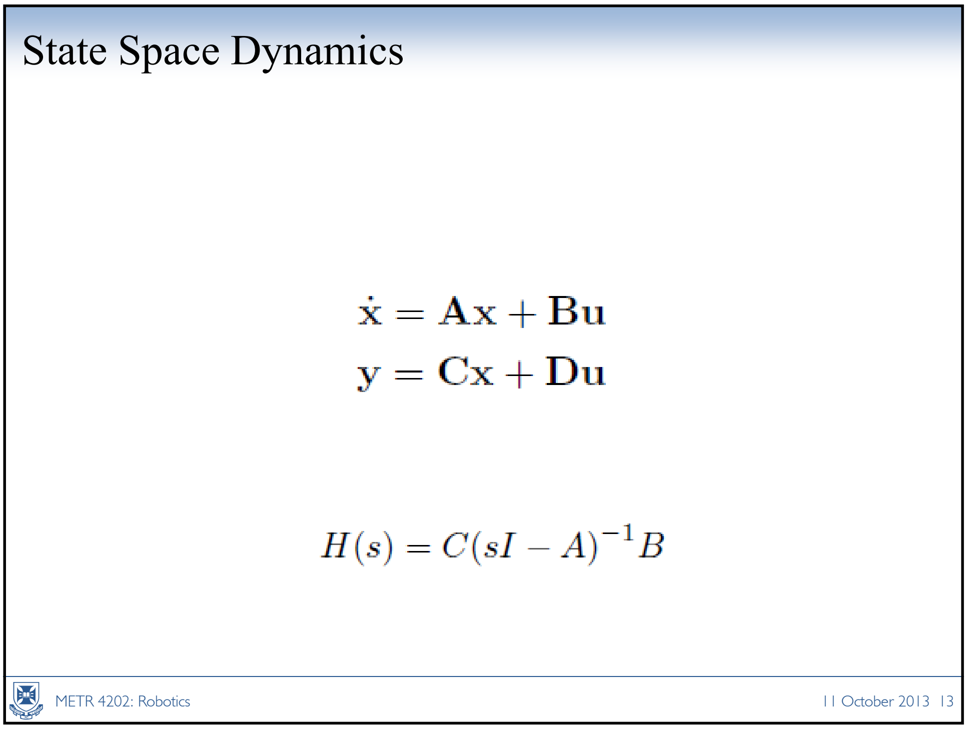


Example II: Estimation

Along multiple dimensions

State Space

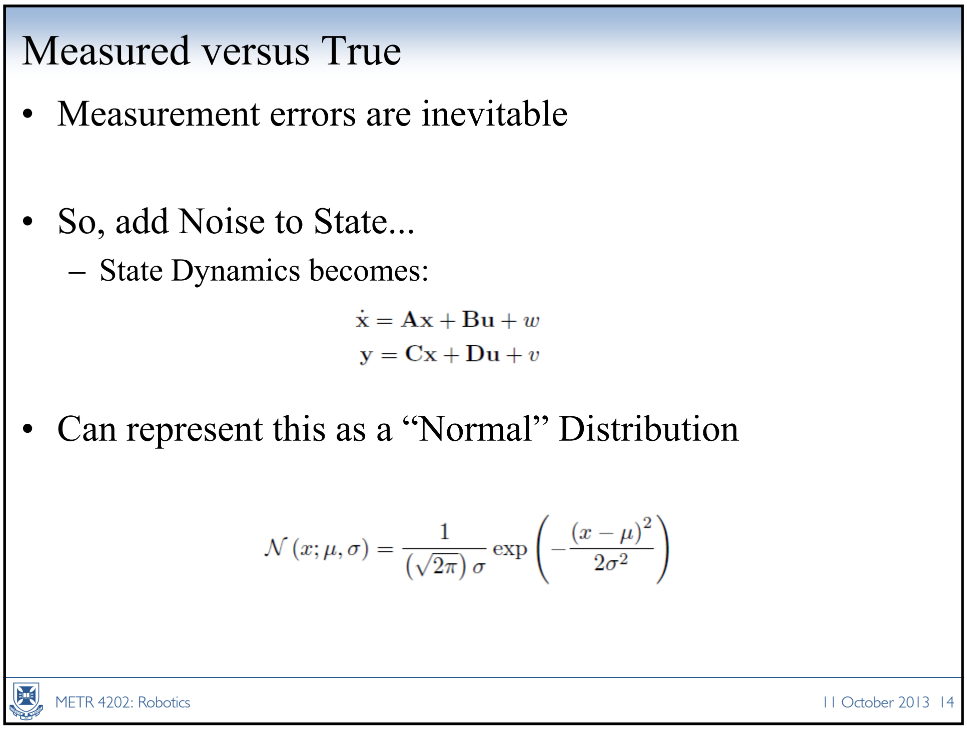
- We collect our set of uncertain variables into a vector …
  $\mathbf{x} = [x_1, x_2,\ldots, x_N]^T$
- The set of values that **x** might take on is termed the *state space*
- There is a *single* true value for **x**,
  but it is unknown



State Space Dynamics

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

$$H(s) = C(sI - A)^{-1}B$$



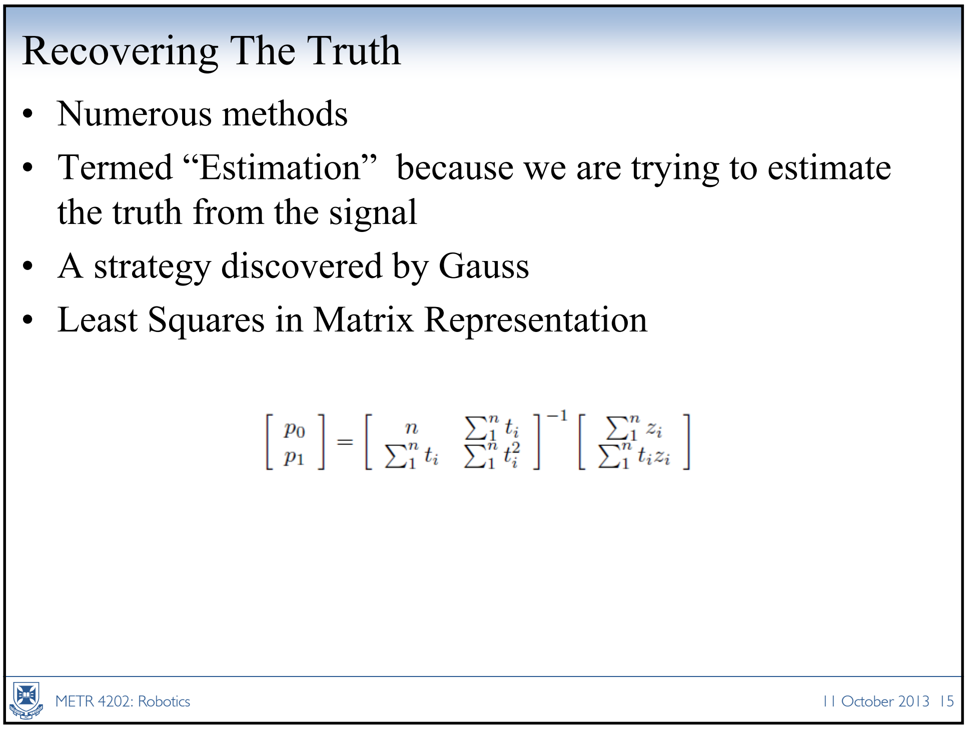
Measured versus True

- Measurement errors are inevitable
- So, add Noise to State...
  - State Dynamics becomes:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + w$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} + v$$

- Can represent this as a “Normal” Distribution

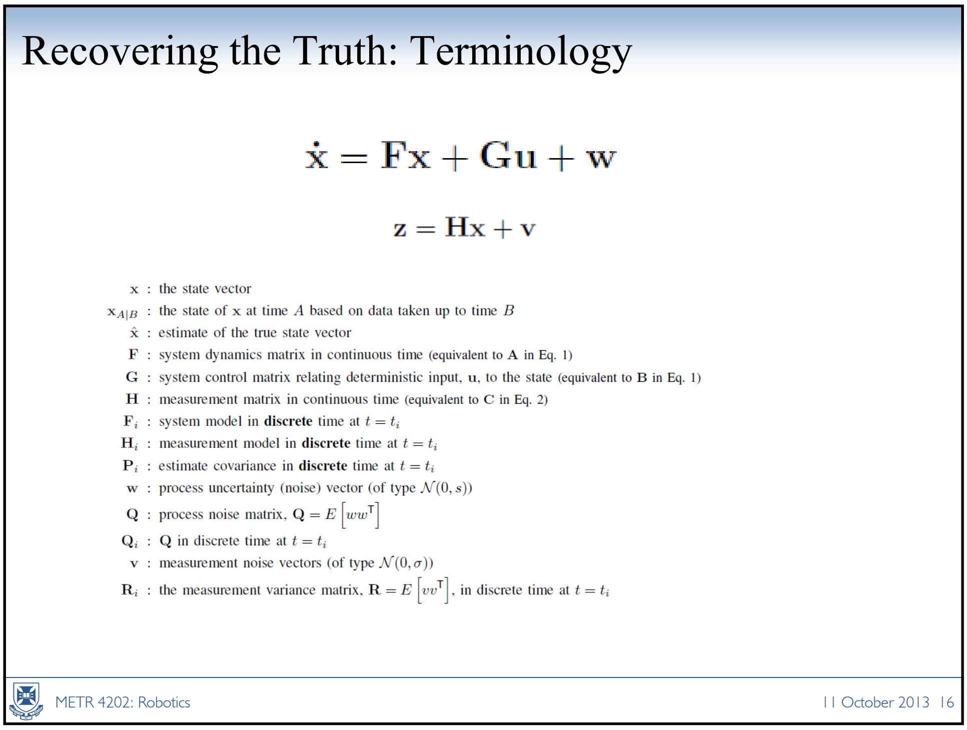
$$\mathcal{N}(x; \mu, \sigma) = \frac{1}{\left(\sqrt{2\pi}\right)\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$



Recovering The Truth

- Numerous methods
- Termed “Estimation” because we are trying to estimate the truth from the signal
- A strategy discovered by Gauss
- Least Squares in Matrix Representation

$$\begin{bmatrix} p_0 \\ p_1 \end{bmatrix} = \begin{bmatrix} n & \sum_1^n t_i \\ \sum_1^n t_i & \sum_1^n t_i^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_1^n z_i \\ \sum_1^n t_i z_i \end{bmatrix}$$



Recovering the Truth: Terminology

$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{u} + \mathbf{w}$$

$$\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{v}$$

$\mathbf{x}$: the state vector

$\mathbf{x}_{A|B}$: the state of $\mathbf{x}$ at time A based on data taken up to time B

$\hat{\mathbf{x}}$: estimate of the true state vector

$\mathbf{F}$: system dynamics matrix in continuous time (equivalent to $\mathbf{A}$ in Eq. 1)

$\mathbf{G}$: system control matrix relating deterministic input, $\mathbf{u}$, to the state (equivalent to $\mathbf{B}$ in Eq. 1)

$\mathbf{H}$: measurement matrix in continuous time (equivalent to $\mathbf{C}$ in Eq. 2)

$\mathbf{F}_i$: system model in **discrete** time at $t = t_i$

$\mathbf{H}_i$: measurement model in **discrete** time at $t = t_i$

$\mathbf{P}_i$: estimate covariance in **discrete** time at $t = t_i$

$\mathbf{w}$: process uncertainty (noise) vector (of type $\mathcal{N}(0, s)$)

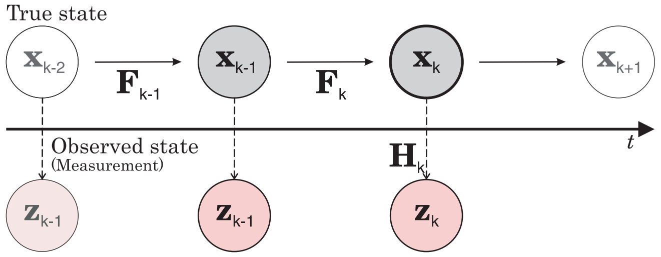
$\mathbf{Q}$: process noise matrix, $\mathbf{Q} = E\left[ww^{\mathsf{T}}\right]$

$\mathbf{Q}_i$: $\mathbf{Q}$ in discrete time at $t = t_i$

$\mathbf{v}$: measurement noise vectors (of type $\mathcal{N}(0, \sigma)$)

$\mathbf{R}_i$: the measurement variance matrix, $\mathbf{R} = E\left[vv^{\mathsf{T}}\right]$, in discrete time at $t = t_i$

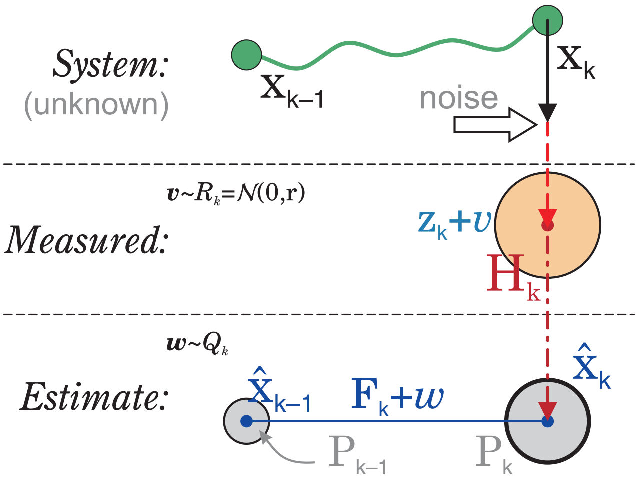
General Problem…



Duals and Dual Terminology

	Estimation		Control
Model:	$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x}$ (discrete: $\mathbf{x} = \mathbf{F}_k\mathbf{x}$)	↔	$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$, $\mathbf{A} = \mathbf{F}^{\top}$
Regulates:	P (covariance)	↔	M (performance matrix)
Minimized function:	Q (or $GQG^{\top}$)	↔	V
Optimal Gain:	K	↔	G
Completeness law:	Observability	↔	Controllability

Estimation Process in Pictures

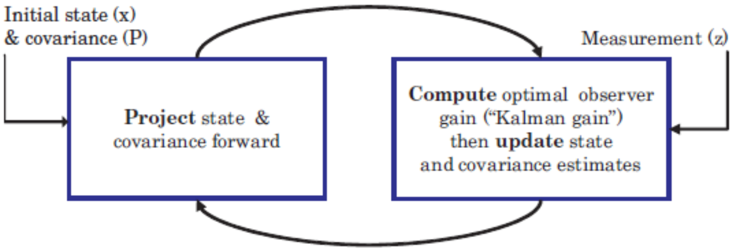

System:

(unknown)

Measured:

Estimate:

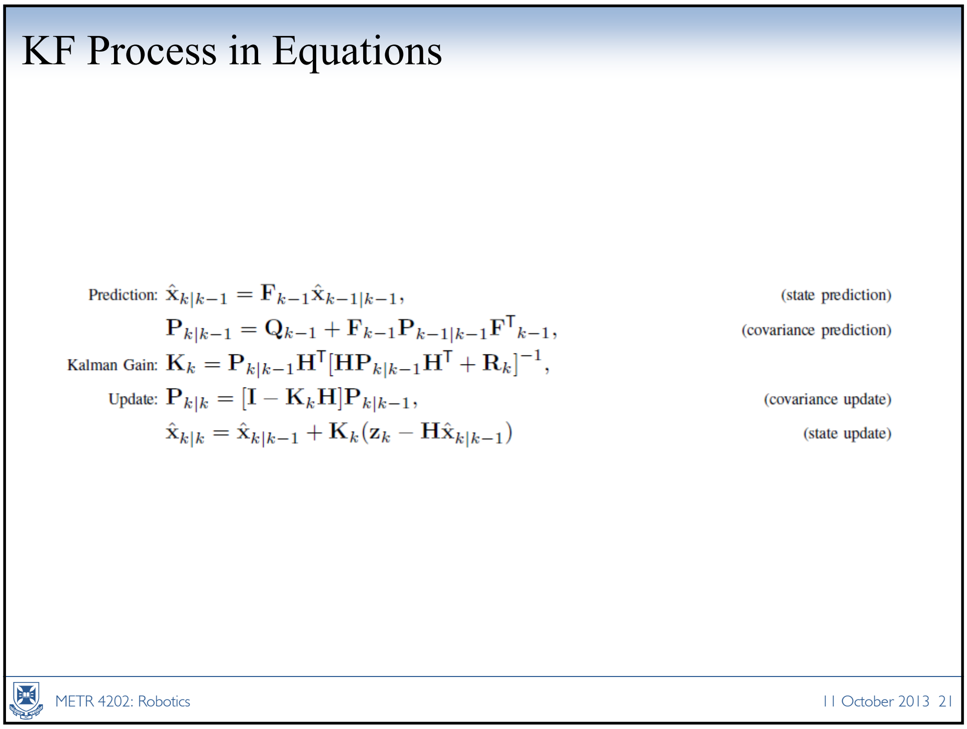
Kalman Filter Process


Initial state (x)
& covariance (P)

Project state &
covariance forward

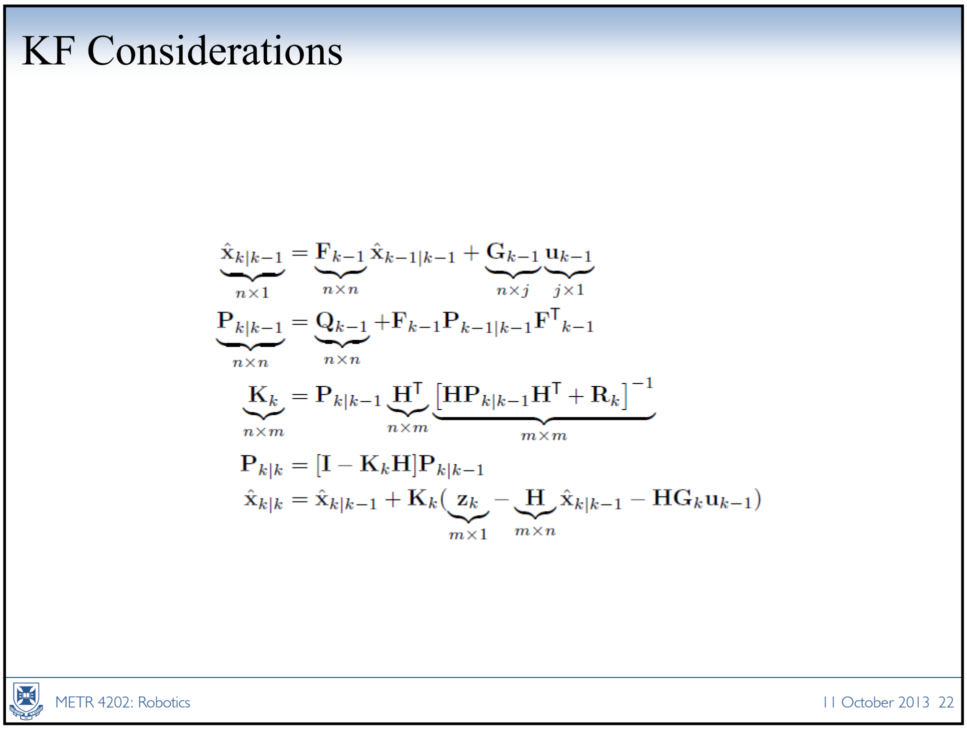
Compute optimal observer
gain ("Kalman gain")
then **update** state
and covariance estimates

Measurement (z)



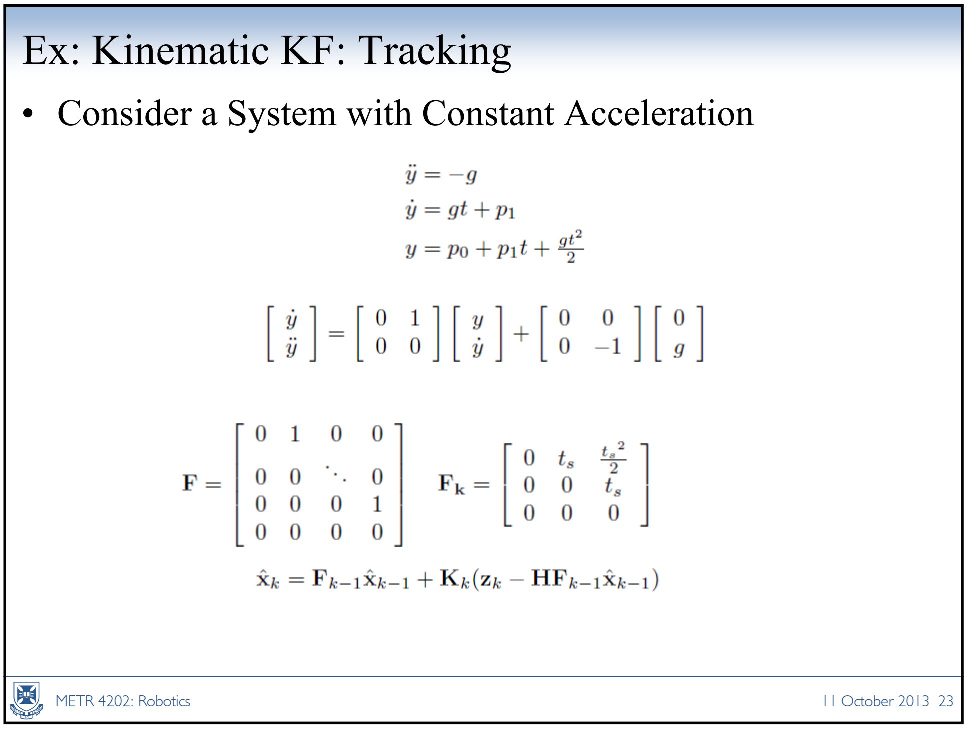
KF Process in Equations

$$\begin{aligned}
\text{Prediction: } \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}, && \text{(state prediction)}\\
\mathbf{P}_{k|k-1} &= \mathbf{Q}_{k-1} + \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}^{\mathsf{T}}{}_{k-1}, && \text{(covariance prediction)}\\
\text{Kalman Gain: } \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}^{\mathsf{T}}[\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^{\mathsf{T}} + \mathbf{R}_k]^{-1}, &&\\
\text{Update: } \mathbf{P}_{k|k} &= [\mathbf{I} - \mathbf{K}_k\mathbf{H}]\mathbf{P}_{k|k-1}, && \text{(covariance update)}\\
\hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}) && \text{(state update)}
\end{aligned}$$



KF Considerations

$$\begin{aligned}
\underbrace{\hat{\mathbf{x}}_{k|k-1}}_{n\times 1} &= \underbrace{\mathbf{F}_{k-1}}_{n\times n}\hat{\mathbf{x}}_{k-1|k-1} + \underbrace{\mathbf{G}_{k-1}}_{n\times j}\underbrace{\mathbf{u}_{k-1}}_{j\times 1}\\
\underbrace{\mathbf{P}_{k|k-1}}_{n\times n} &= \underbrace{\mathbf{Q}_{k-1}}_{n\times n} + \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}^{\mathsf{T}}{}_{k-1}\\
\underbrace{\mathbf{K}_k}_{n\times m} &= \mathbf{P}_{k|k-1}\underbrace{\mathbf{H}^{\mathsf{T}}}_{n\times m}\underbrace{[\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^{\mathsf{T}} + \mathbf{R}_k]^{-1}}_{m\times m}\\
\mathbf{P}_{k|k} &= [\mathbf{I} - \mathbf{K}_k\mathbf{H}]\mathbf{P}_{k|k-1}\\
\hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\underbrace{\mathbf{z}_k}_{m\times 1} - \underbrace{\mathbf{H}}_{m\times n}\hat{\mathbf{x}}_{k|k-1} - \mathbf{H}\mathbf{G}_k\mathbf{u}_{k-1})
\end{aligned}$$



Ex: Kinematic KF: Tracking

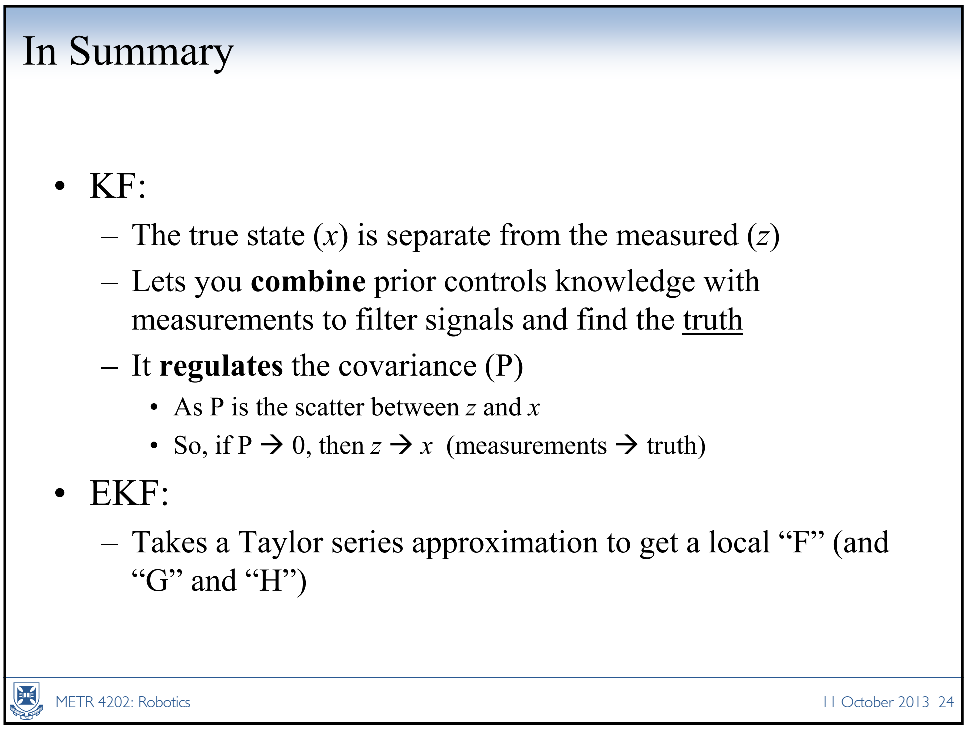
- Consider a System with Constant Acceleration

$$\ddot{y} = -g$$
$$\dot{y} = gt + p_1$$
$$y = p_0 + p_1 t + \frac{gt^2}{2}$$

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ g \end{bmatrix}$$

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{F_k} = \begin{bmatrix} 0 & t_s & \frac{{t_s}^2}{2} \\ 0 & 0 & t_s \\ 0 & 0 & 0 \end{bmatrix}$$

$$\hat{\mathbf{x}}_k = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1})$$

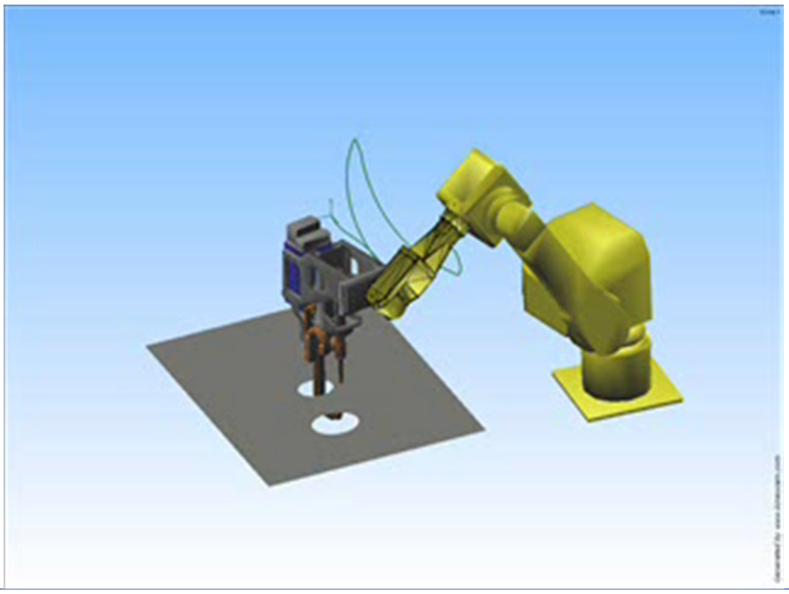


In Summary

- KF:
  - The true state (*x*) is separate from the measured (*z*)
  - Lets you **combine** prior controls knowledge with measurements to filter signals and find the truth
  - It **regulates** the covariance (P)
    - As P is the scatter between *z* and *x*
    - So, if P → 0, then *z* → *x* (measurements → truth)
- EKF:
  - Takes a Taylor series approximation to get a local “F” (and “G” and “H”)

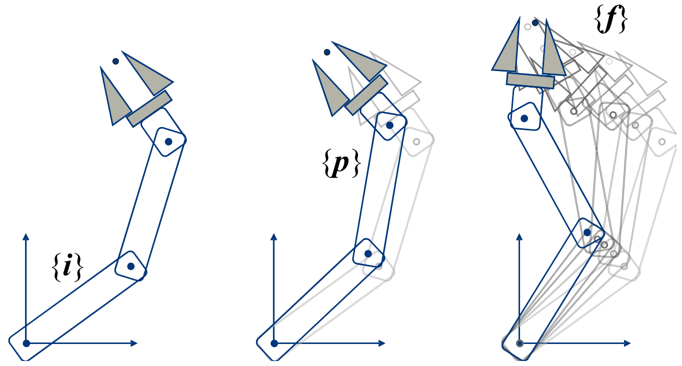
Motion
PLANNING

Trajectory Generation & Planning



Trajectory Generation

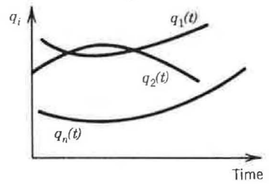
- The goal is to get from an initial position $\{\boldsymbol{i}\}$ to a final position $\{\boldsymbol{f}\}$ via a path points $\{\boldsymbol{p}\}$



Joint Space

Consider only the **joint positions** as a function of time

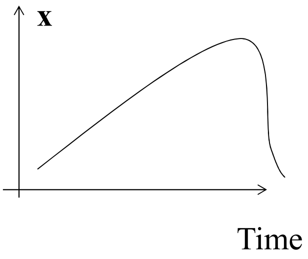
- + Since we control the joints, this is more direct
- -- If we want to follow a particular trajectory, <u>not easy</u>
  - – at best lots of intermediate points
  - – No guarantee that you can solve the Inverse Kinematics for all path points



Cartesian Workspace

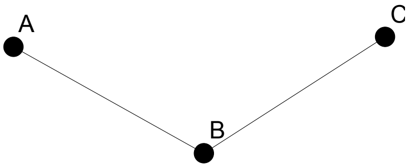
Consider the **Cartesian positions** as a function of time

- + Can track shapes exactly
- -- We need to solve the inverse kinematics and dynamics



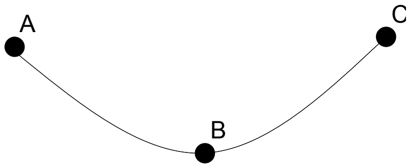
Polynomial Trajectories

- Straight line Trajectories



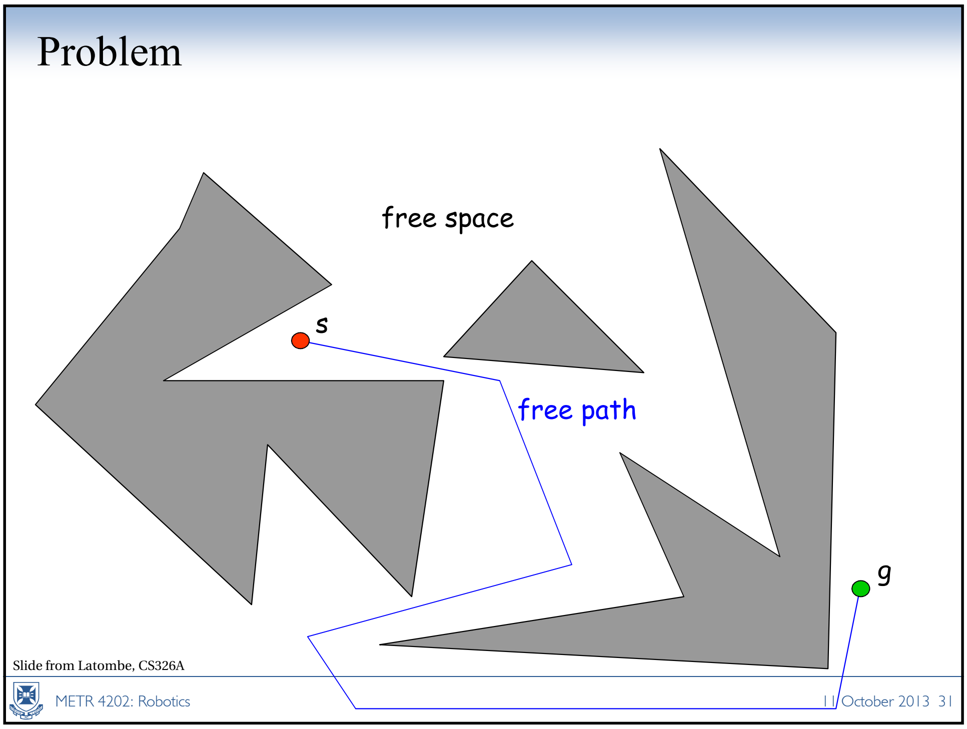
- Simpler

- Polynomial Trajectories



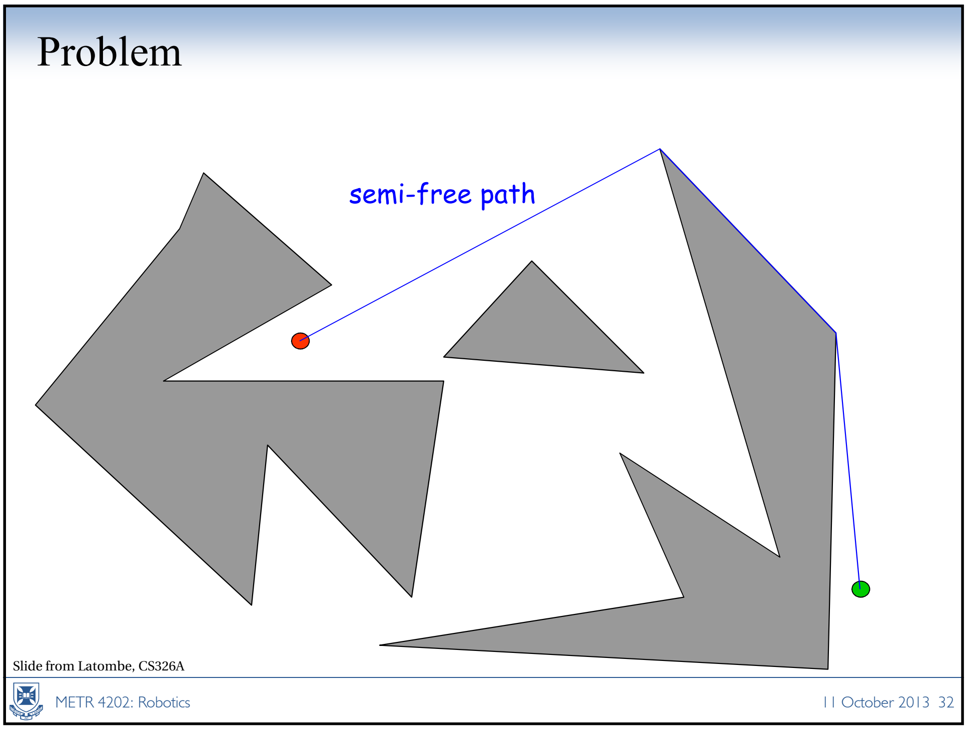
$u(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$

- Parabolic blends are smoother
- Use “pseudo via points”



Problem

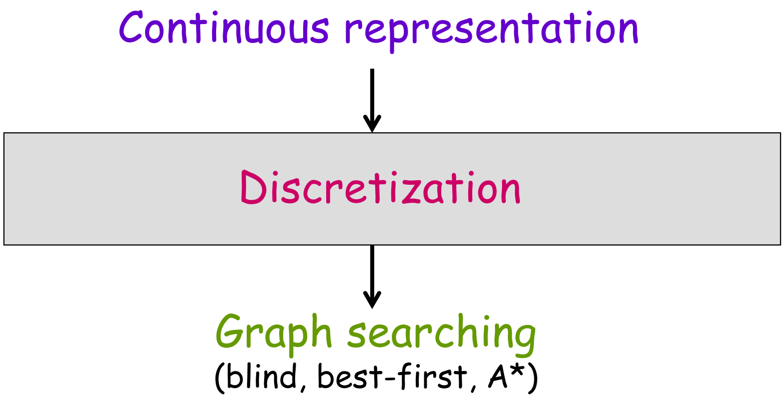
Slide from Latombe, CS326A



Problem

Slide from Latombe, CS326A

Motion-Planning Framework



Continuous representation

Discretization

Graph searching

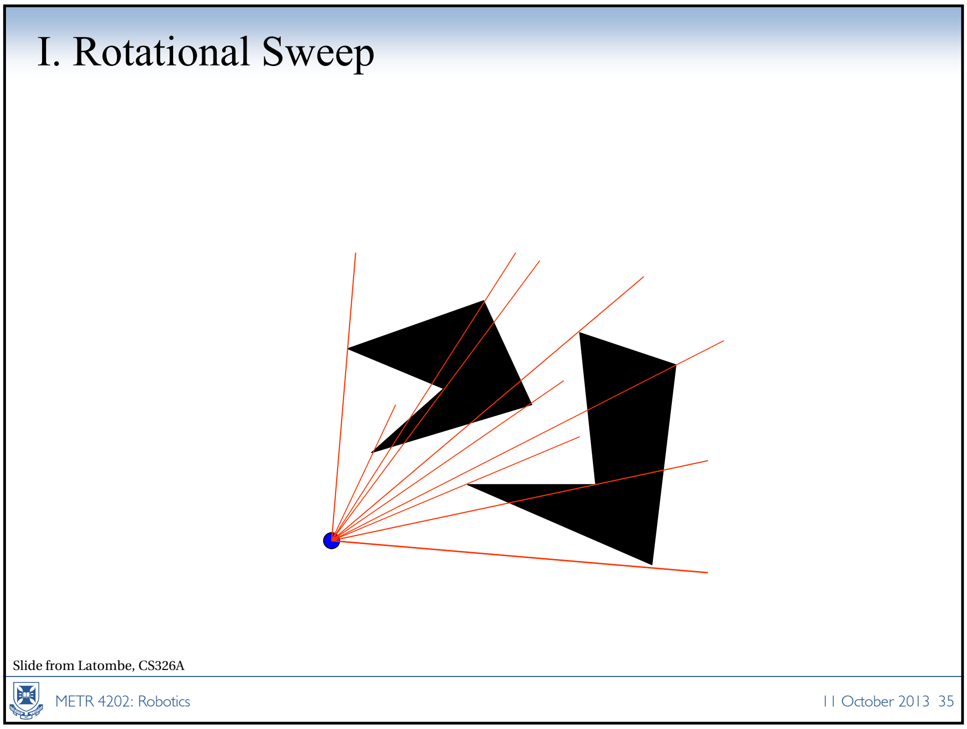
(blind, best-first, A*)

Slide from Latombe, CS326A

Path-Planning Approaches

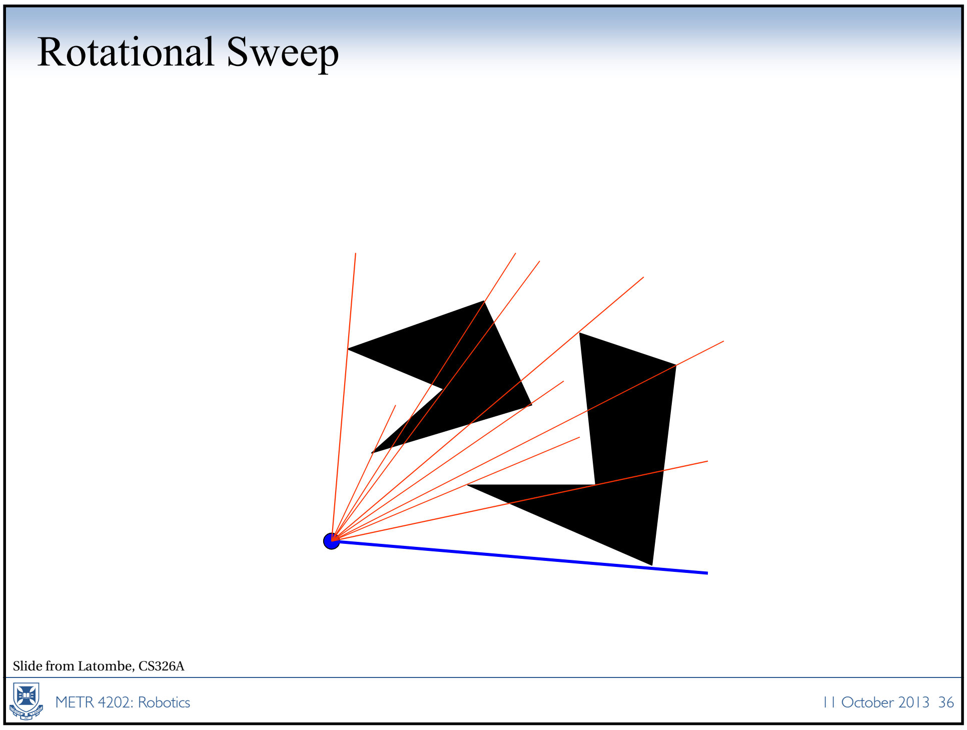
- Roadmap
  Represent the connectivity of the free space by a network of 1-D curves
- Cell decomposition
  Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells
- Potential field
  Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

Slide from Latombe, CS326A



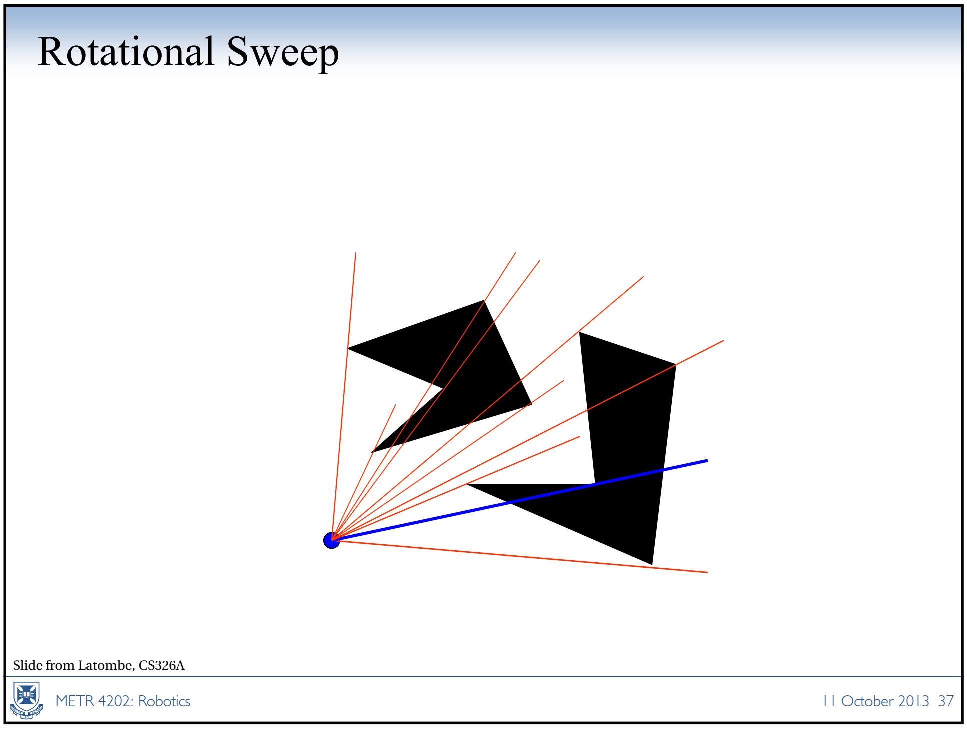
I. Rotational Sweep

Slide from Latombe, CS326A



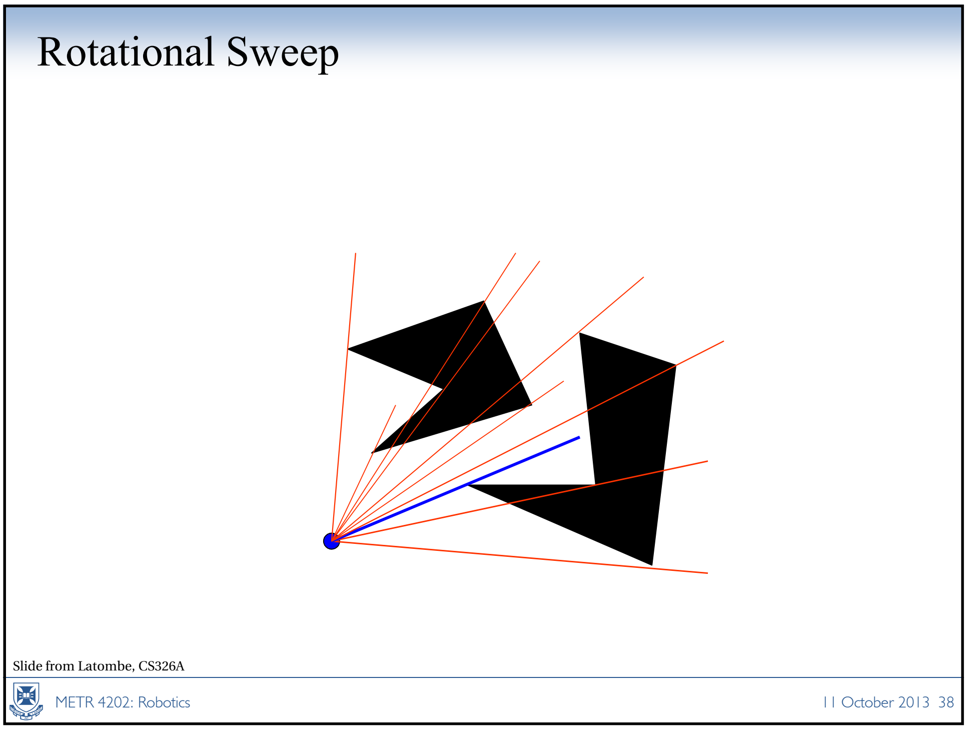
Rotational Sweep

Slide from Latombe, CS326A



Rotational Sweep

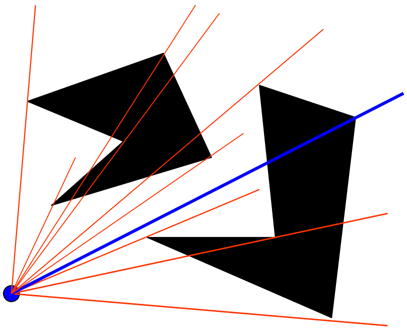
Slide from Latombe, CS326A



Rotational Sweep

Slide from Latombe, CS326A

Rotational Sweep

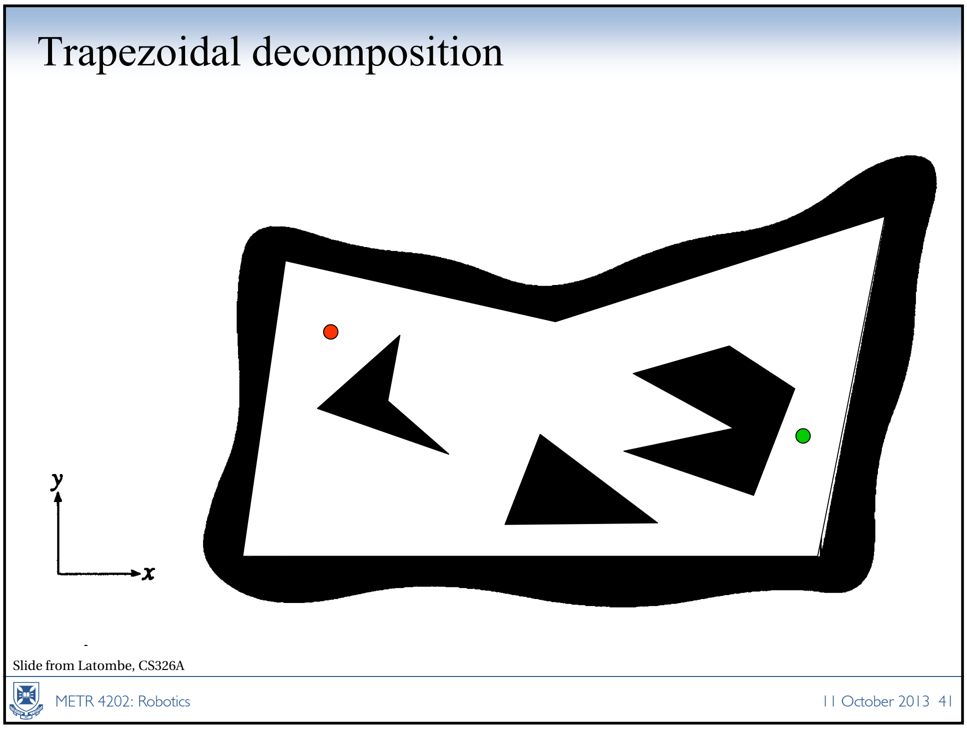


Slide from Latombe, CS326A

II. Cell-Decomposition Methods

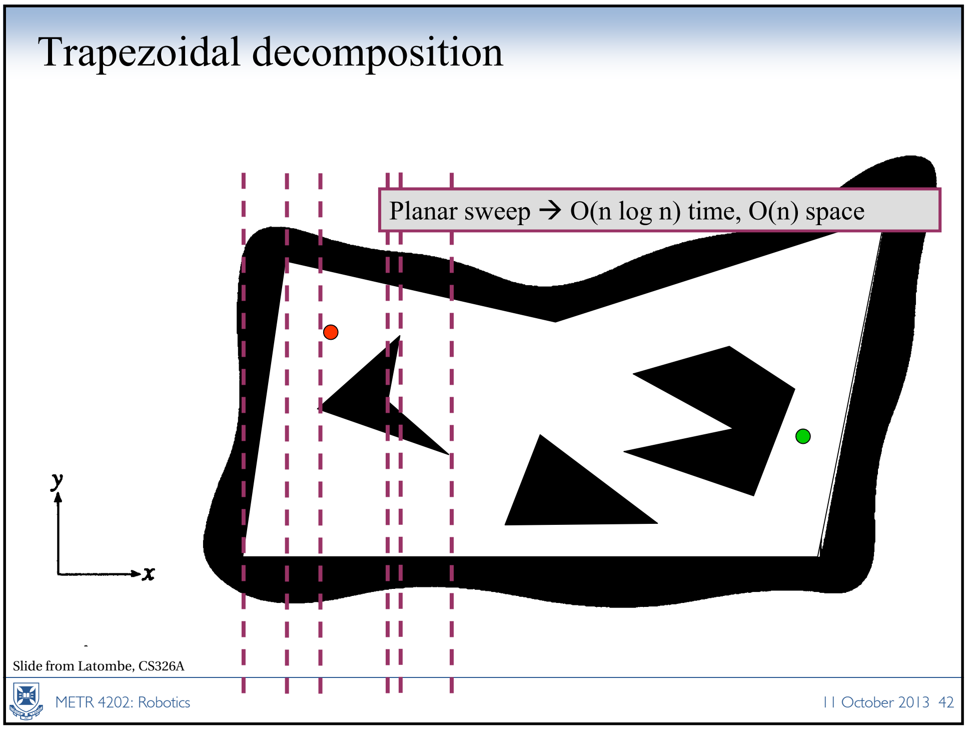
Two classes of methods:

- Exact cell decomposition
  - The free space **F** is represented by a collection of non-overlapping cells whose union is exactly **F**
  - Example: trapezoidal decomposition
- Approximate cell decomposition
  - **F** is represented by a collection of non-overlapping cells whose union is contained in **F** Examples: quadtree, octree, 2n-tree



Trapezoidal decomposition

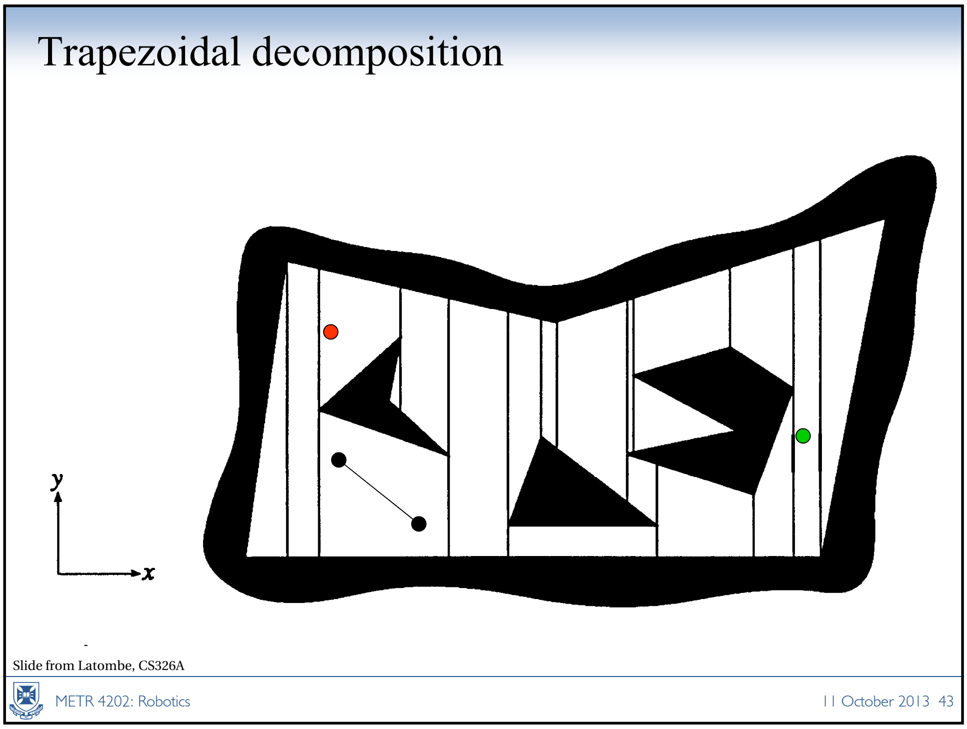
Slide from Latombe, CS326A



Trapezoidal decomposition

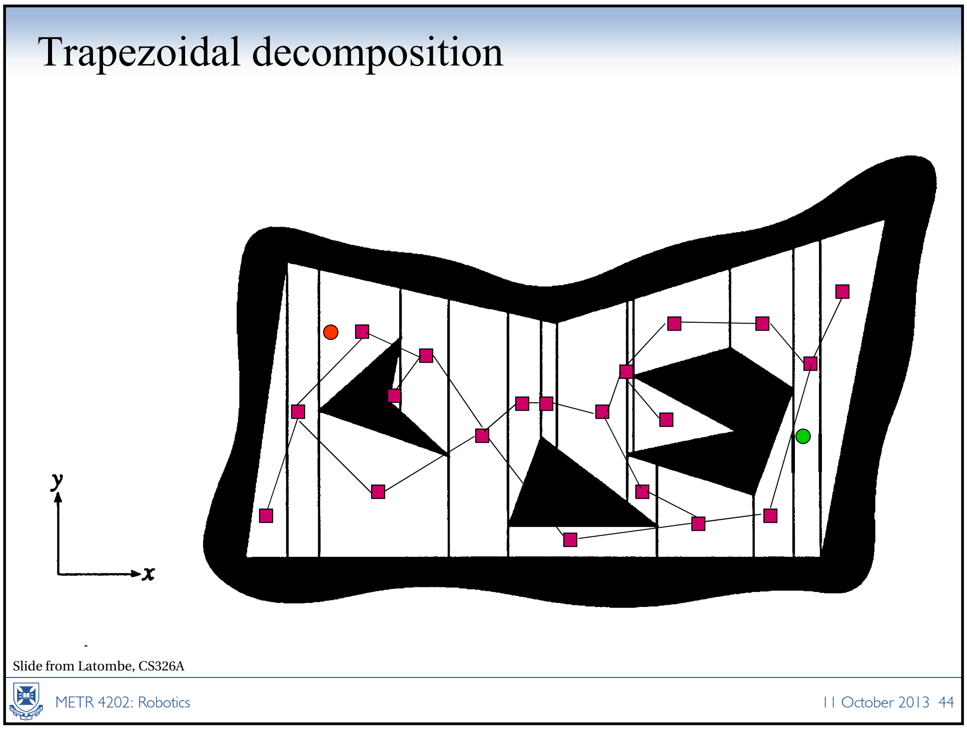
Planar sweep → O(n log n) time, O(n) space

Slide from Latombe, CS326A



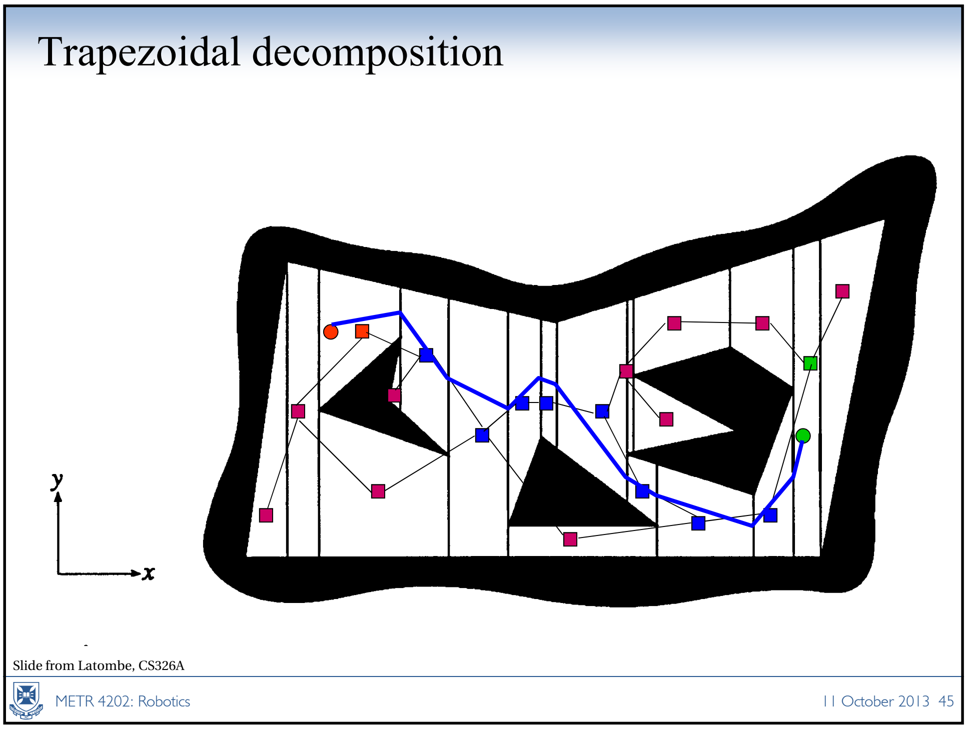
Trapezoidal decomposition

Slide from Latombe, CS326A



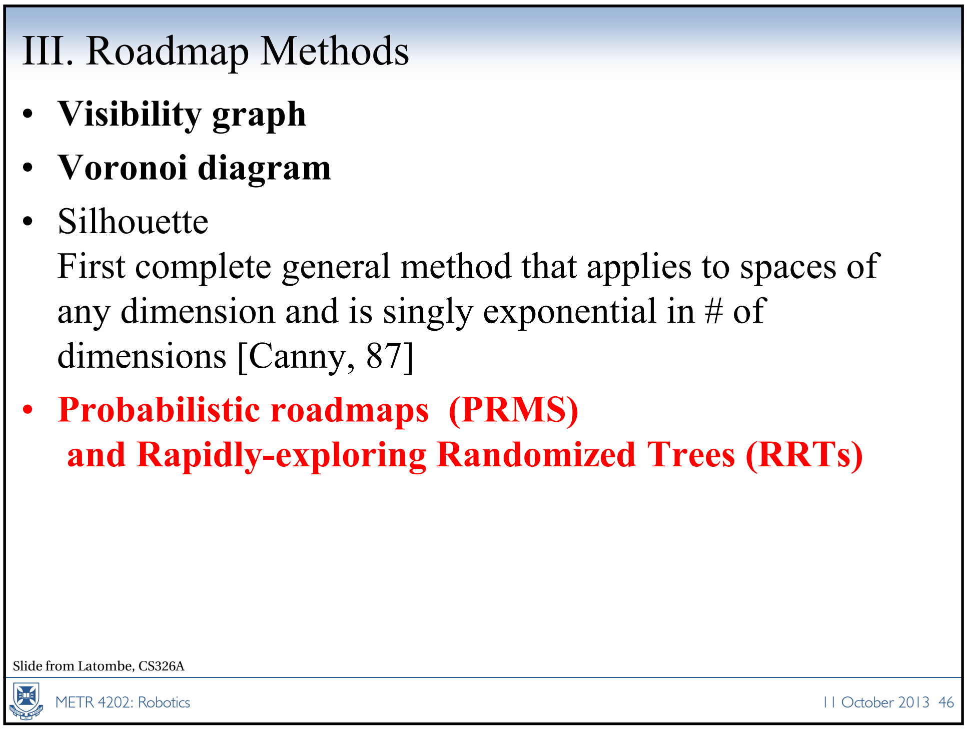
Trapezoidal decomposition

Slide from Latombe, CS326A



Trapezoidal decomposition

Slide from Latombe, CS326A



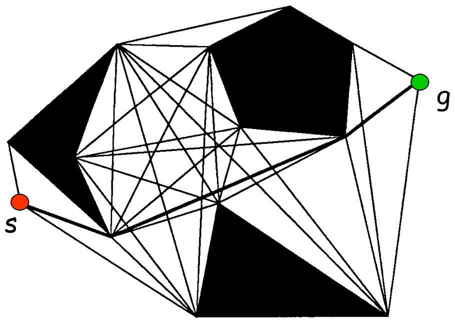
III. Roadmap Methods

- **Visibility graph**
- **Voronoi diagram**
- Silhouette
  First complete general method that applies to spaces of any dimension and is singly exponential in # of dimensions [Canny, 87]
- **Probabilistic roadmaps (PRMS)
  and Rapidly-exploring Randomized Trees (RRTs)**

Slide from Latombe, CS326A

Roadmap Methods

- **Visibility graph**
  Introduced in the Shakey project at SRI in the late 60s.
  Can produce shortest paths in 2-D configuration spaces

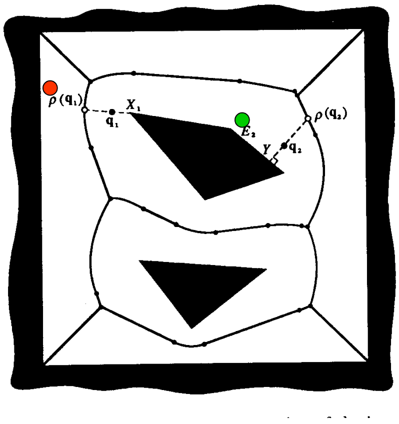


Slide from Latombe, CS326A

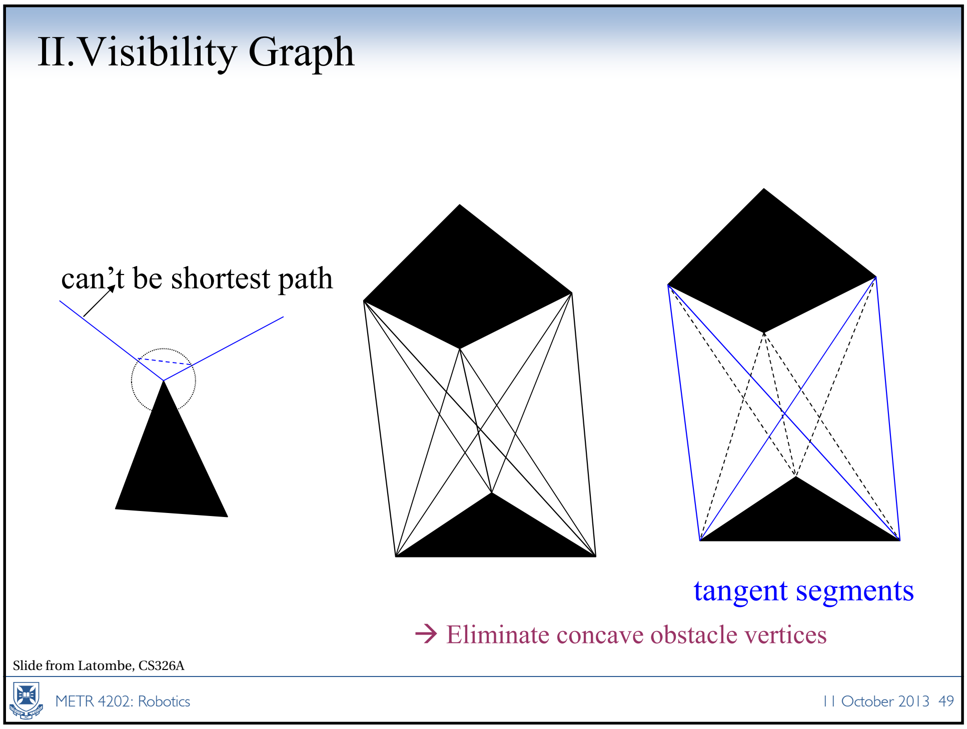
Roadmap Methods

- Voronoi diagram
  Introduced by Computational Geometry researchers. Generate paths that maximizes clearance.

  O(n log n) time
  O(n) space



Slide from Latombe, CS326A



II.Visibility Graph

can't be shortest path

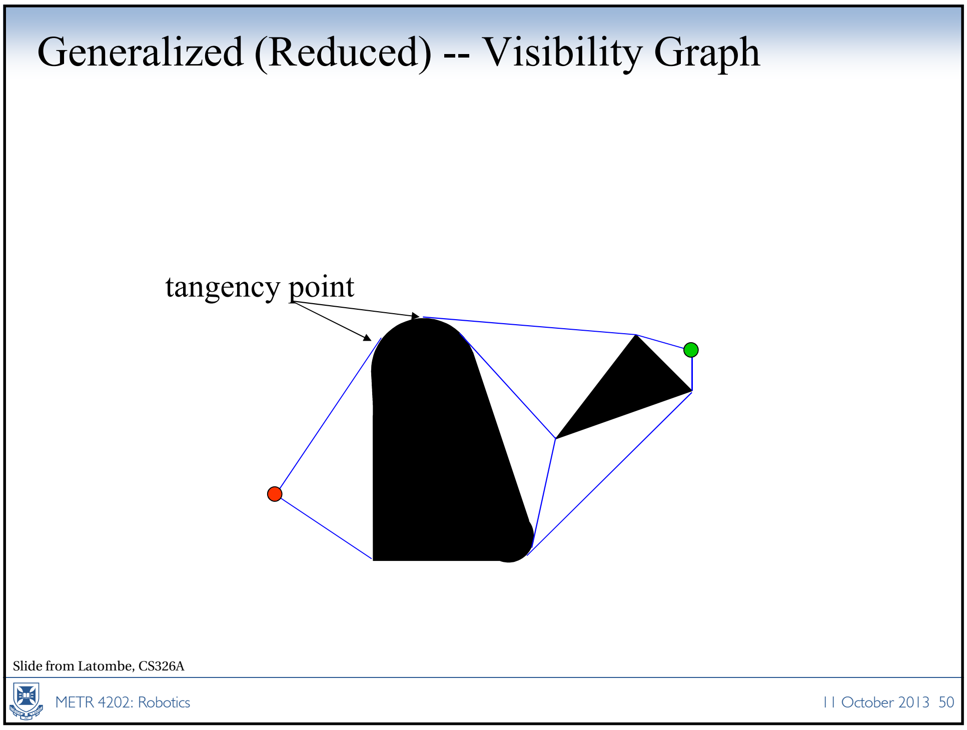
tangent segments

→ Eliminate concave obstacle vertices

Slide from Latombe, CS326A

METR 4202: Robotics

11 October 2013 49



Generalized (Reduced) -- Visibility Graph

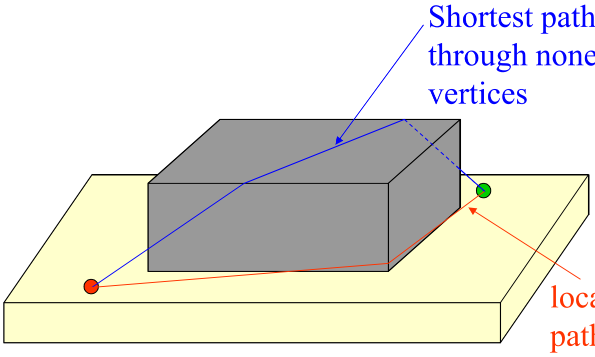
tangency point

Slide from Latombe, CS326A

METR 4202: Robotics

11 October 2013 50

Three-Dimensional Space



Shortest path passes through none of the vertices

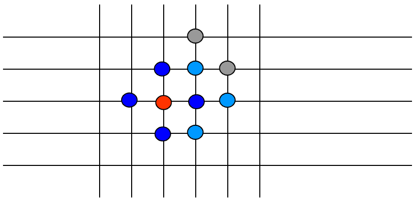
locally shortest path homotopic to globally shortest path

Computing the shortest collision-free path in a polyhedral space is NP-hard

Slide from Latombe, CS326A

Sketch of Grid Algorithm (with best-first search)

- Place regular grid G over space
- Search G using best-first search algorithm with potential as heuristic function



Slide from Latombe, CS326A

Simple Algorithm (for Visibility Graphs)

- Install all obstacles vertices in VG, plus the start and goal positions
- For every pair of nodes u, v in VG

```
    If segment(u,v) is an obstacle edge then
            insert (u,v) into VG
    else
    for every obstacle edge e
            if segment(u,v) intersects e
                    then go up to segment
  insert (u,v) into VG
```

- Search VG using A*

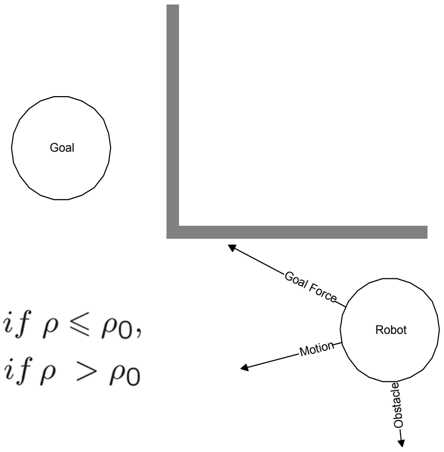
Slide based on Latombe, CS326A

IV. Potential Field Methods

- Approach initially proposed for real-time collision avoidance [Khatib, 86]

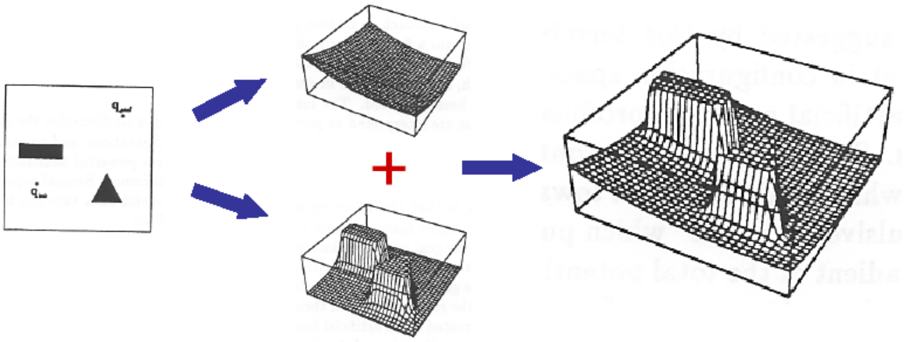
$$F_{Goal} = -k_p(x - x_{Goal})$$

$$F_{Obstacle} = \begin{cases} \eta \left(\frac{1}{\rho} - \frac{1}{\rho_0}\right) \frac{1}{\rho^2} \frac{\partial \rho}{\partial x} & if\ \rho \leqslant \rho_0, \\ 0 & if\ \rho\ > \rho_0 \end{cases}$$



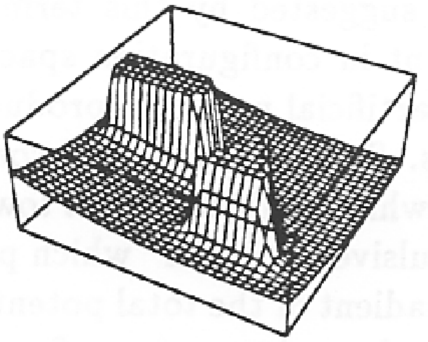
Slide based on Latombe, CS326A

Attractive and Repulsive fields



Slide from Latombe, CS326A

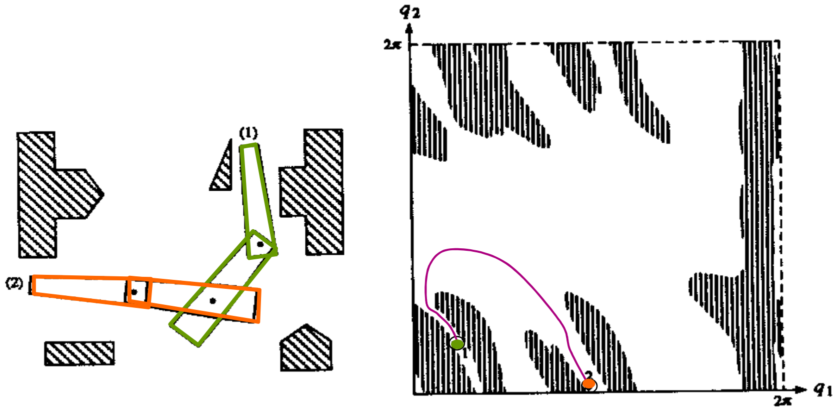
Local-Minimum Issue



- Perform best-first search (possibility of combining with approximate cell decomposition)
- Alternate descents and random walks
- Use local-minimum-free potential (navigation function)

Slide from Latombe, CS326A

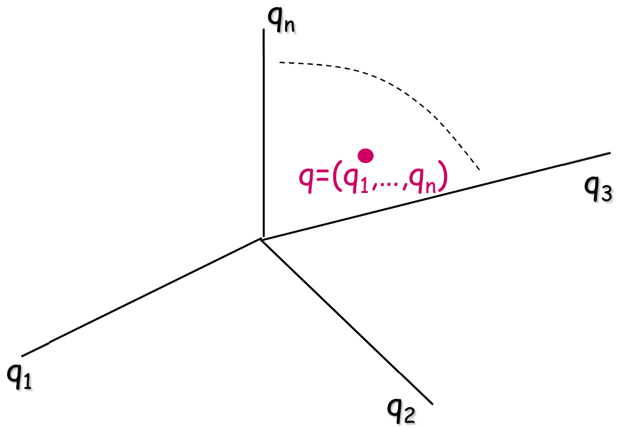
Configuration Space



- A robot configuration is a specification of the positions of all robot points relative to a fixed coordinate system
- Usually a configuration is expressed as a “vector” of position/orientation parameters

Slide from Latombe, CS326A

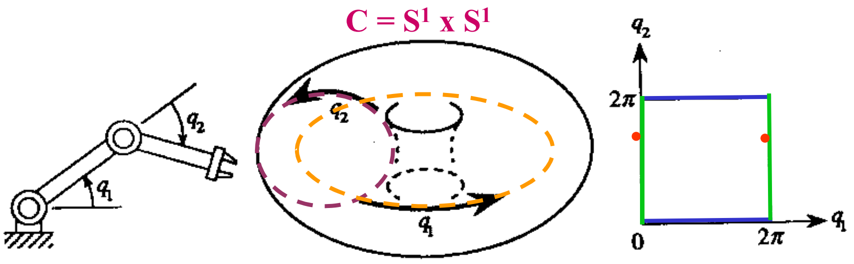
Motion Planning in C-Space



Slide from Latombe, CS326A

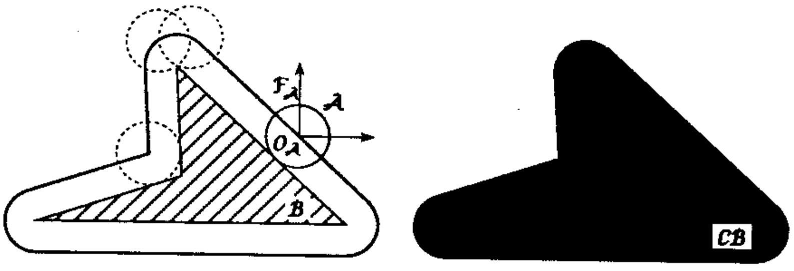
Configuration Space of a Robot

- Space of all its possible configurations
- But the topology of this space is usually not that of a Cartesian space



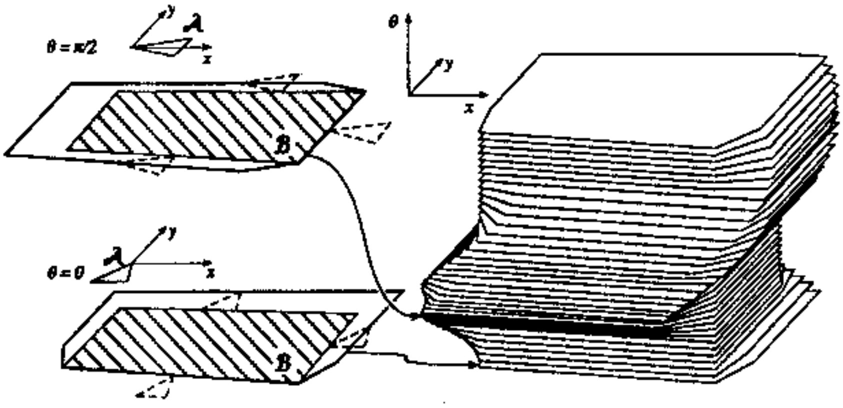
Slide from Latombe, CS326A

Disc Robot in 2-D Workspace



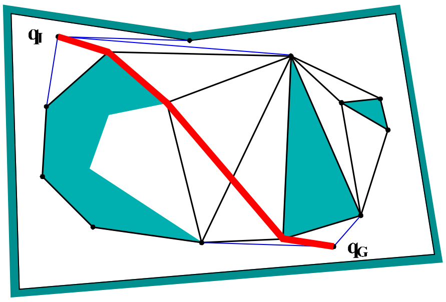
Slide from Latombe, CS326A

Rigid Robot Translating and Rotating in 2-D



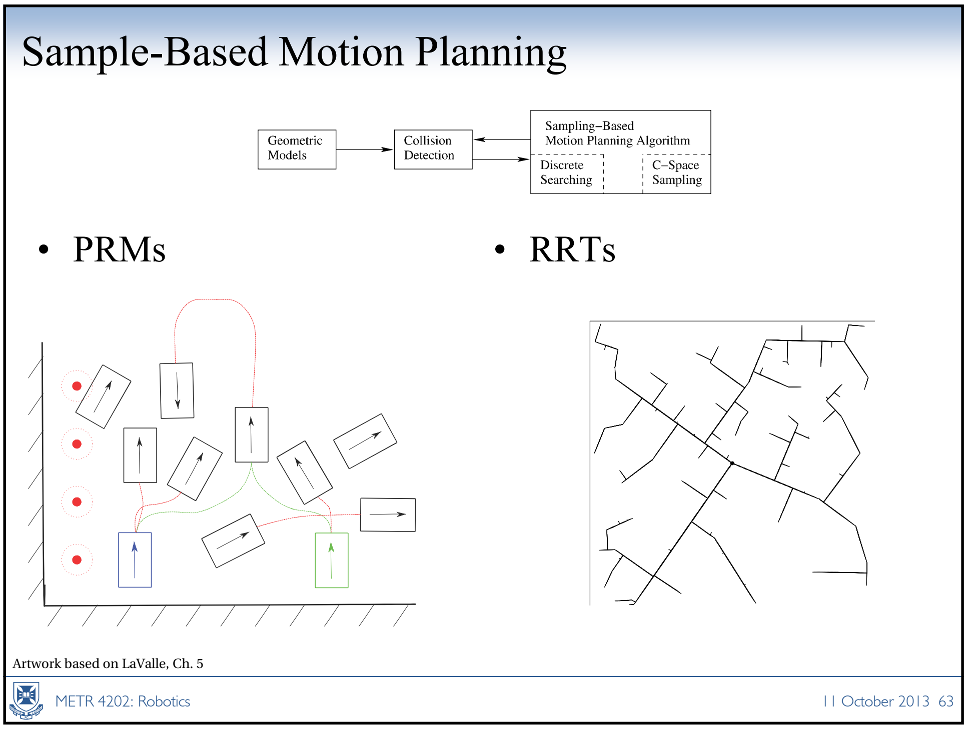
Slide from Latombe, CS326A

Geometric Planning Methods



- Several Geometric Methods:
  - Vertical (Trapezoidal) Cell Decomposition
  - **Roadmap Methods**
    - **Cell (Triangular) Decomposition**
    - Visibility Graphs
    - Veroni Graphs

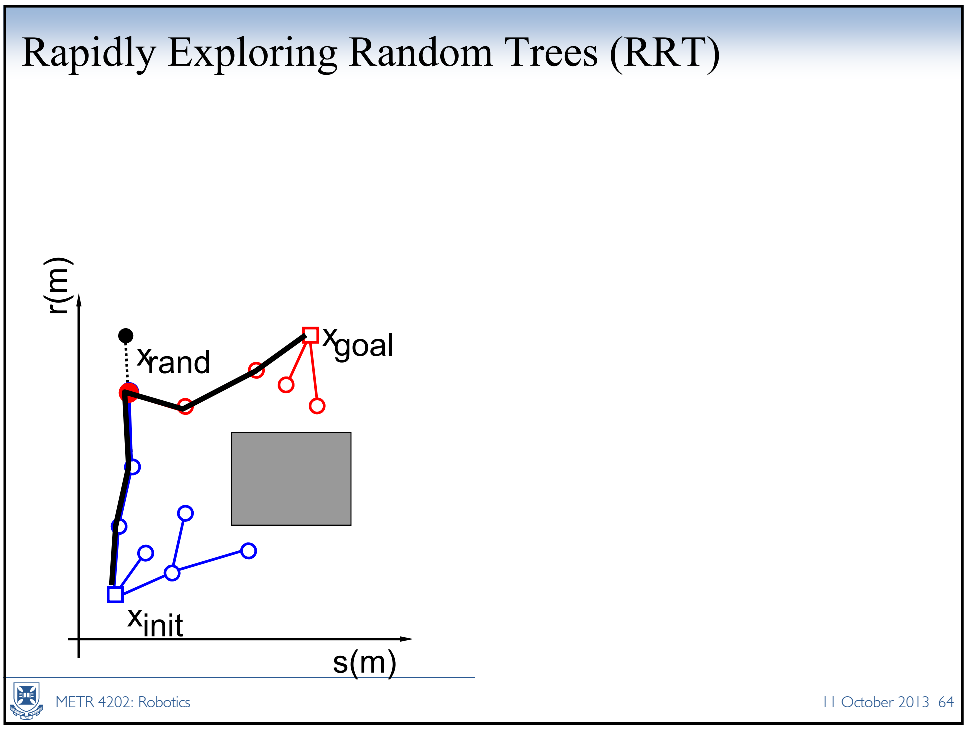
Artwork from LaValle, Ch. 6



Sample-Based Motion Planning

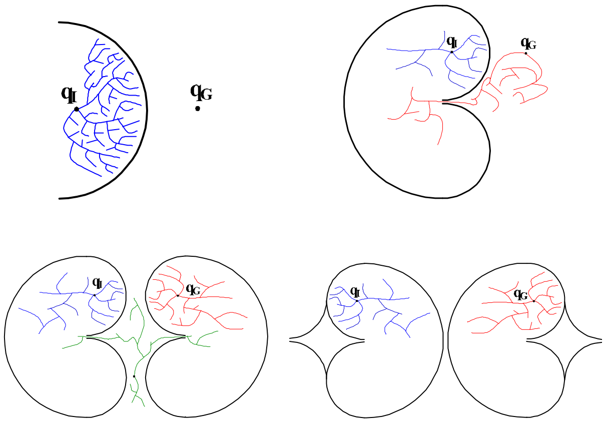
- PRMs
- RRTs

Artwork based on LaValle, Ch. 5



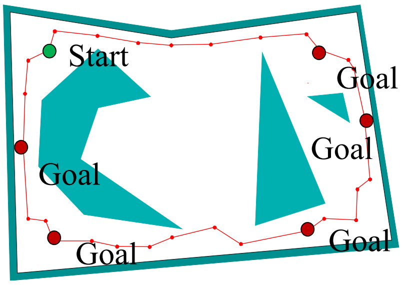
Rapidly Exploring Random Trees (RRT)

Sampling and the "Bug Trap" Problem



Artwork based on LaValle, Ch. 5

Multiple Points & Sequencing



- Sequencing
  - Determining the "best" order to go in

➔ Travelling Salesman Problem

A salesman has to visit each city on a given list exactly once. In doing this, he **starts** from his home city and in the **end he has to return to his home** city. It is plausible for him to select the order in which he visits the cities so that the **total of the distances travelled** in his tour is as small as possible.

Artwork based on LaValle, Ch. 6

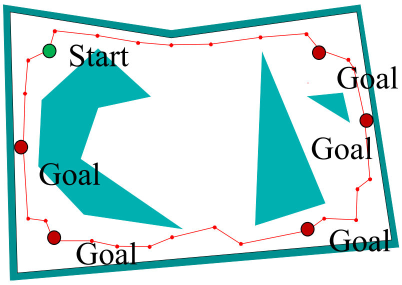
Travelling Salesman Problem

- Given a $n \times n$ distance matrix **C**=*(c_{ij})*

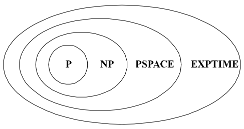
- Minimize:

$$c(\pi) = \sum_{i=1}^{n} c_{i\pi(i)}$$

- Note that this problem is NP-Hard



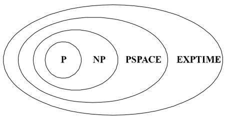
Artwork based on LaVale, Ch. 6



➔ BUT, Special Cases are Well-Solvable!

Travelling Salesman Problem [2]

- This problem is NP-Hard



➔ BUT,
Special Cases are
Well-Solvable!

For the Euclidean case
(where the points are on the 2D Euclidean plane) :

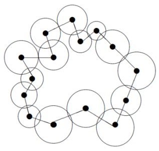
- The shortest TSP tour does not intersect itself, and thus geometry makes the problem somewhat easier.
- If all cities lie on the boundary of a convex polygon, the optimal tour is a cyclic walk along the boundary of the polygon (in clockwise or counterclockwise direction).

***The k*-line TSP**

- The a special case where the cities lie on *k* parallel (or almost parallel) lines in the Euclidean plane.
- EG: Fabrication of printed circuit boards
- Solvable in **O**(n^3) time by Dynamic Programming (Rote's algorithm)

The necklace TSP

- The special Euclidean TSP case where there exist *n* circles around the *n* cities such that every cycle intersects exactly two adjacent circles

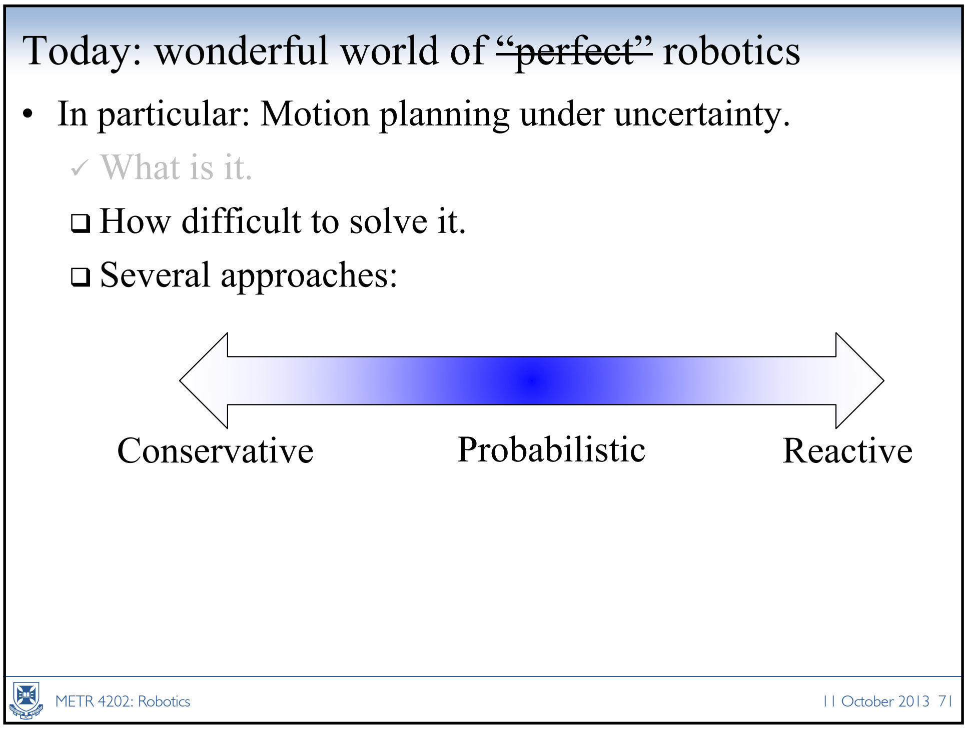


Motion PLANNING & Uncertainty

(Slides C/O Hanna Kurniawati)

Deterministic motion planning vs motion planning under uncertainty

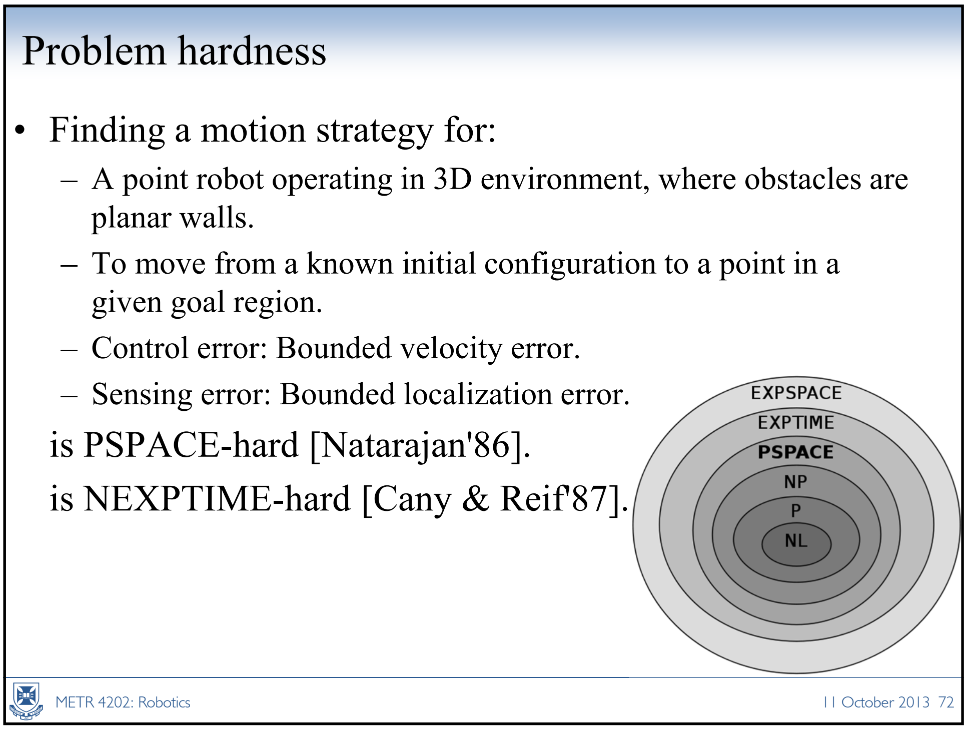
- Deterministic motion planning
  - Find a valid path between two configurations in order to accomplish a task, given:
  - No control error.
  - No sensing.
  - Know the operating environment perfectly.
- Motion planning under uncertainty (today)
  - Find a motion strategy to accomplish a task, where there's a combination of:
  - Control error.
  - Sensing error.
  - Partially / unknown operating environment.



Today: wonderful world of ~~“perfect”~~ robotics

- In particular: Motion planning under uncertainty.
  - ✓ What is it.
  - ❑ How difficult to solve it.
  - ❑ Several approaches:

Conservative Probabilistic Reactive



Problem hardness

- Finding a motion strategy for:
  - A point robot operating in 3D environment, where obstacles are planar walls.
  - To move from a known initial configuration to a point in a given goal region.
  - Control error: Bounded velocity error.
  - Sensing error: Bounded localization error.

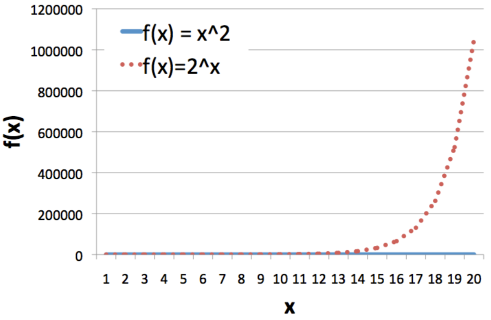
  is PSPACE-hard [Natarajan'86].

  is NEXPTIME-hard [Cany & Reif'87].

EXPSPACE
EXPTIME
PSPACE
NP
P
NL

A little bit on computational complexity 1/3

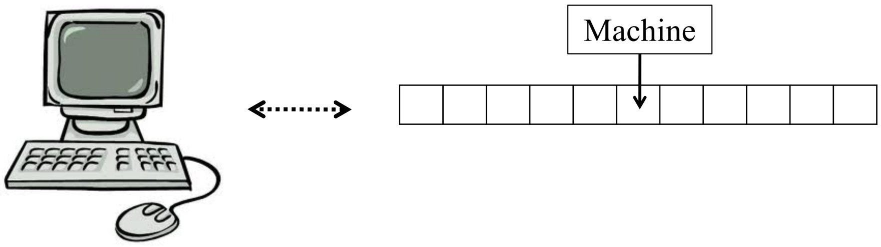
- Algorithms are **not** made to be used only once & are **not** made to be used for only one particular problem.
- How long does it take for the algorithm to find the solution when the input size increases ?
  - In particular, is it polynomial or exponential ?



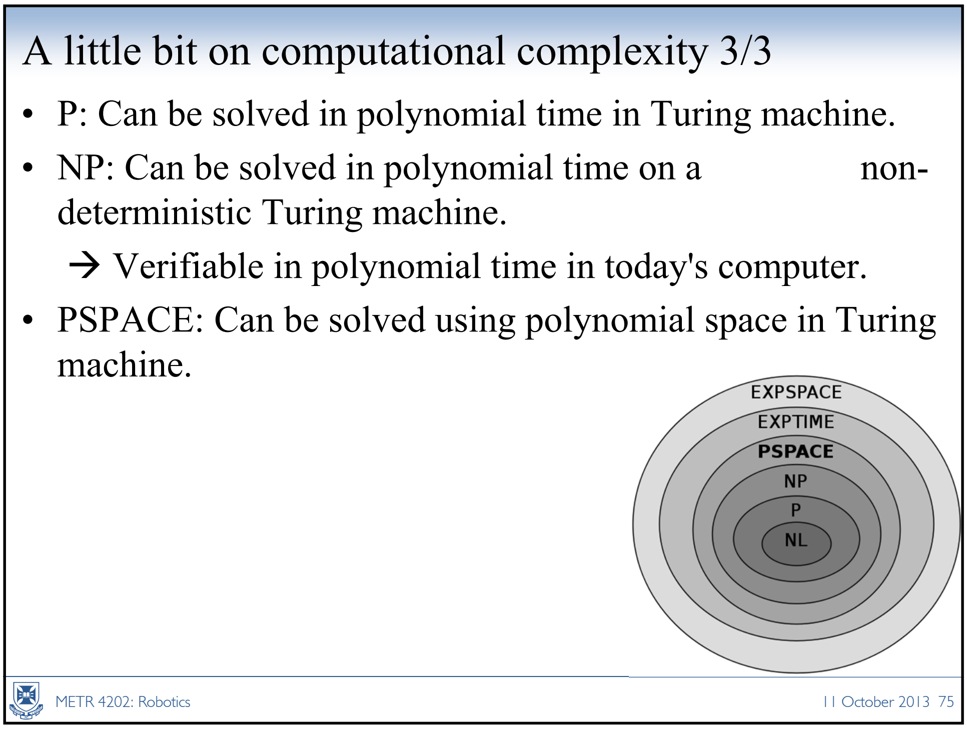
A little bit on computational complexity 2/3

Today’s computer

(Deterministic) Turing machine



Non-Deterministic Turing machine
It can generate multiple possible program executions at once.
Same capability as Turing machine, but can get things done faster.

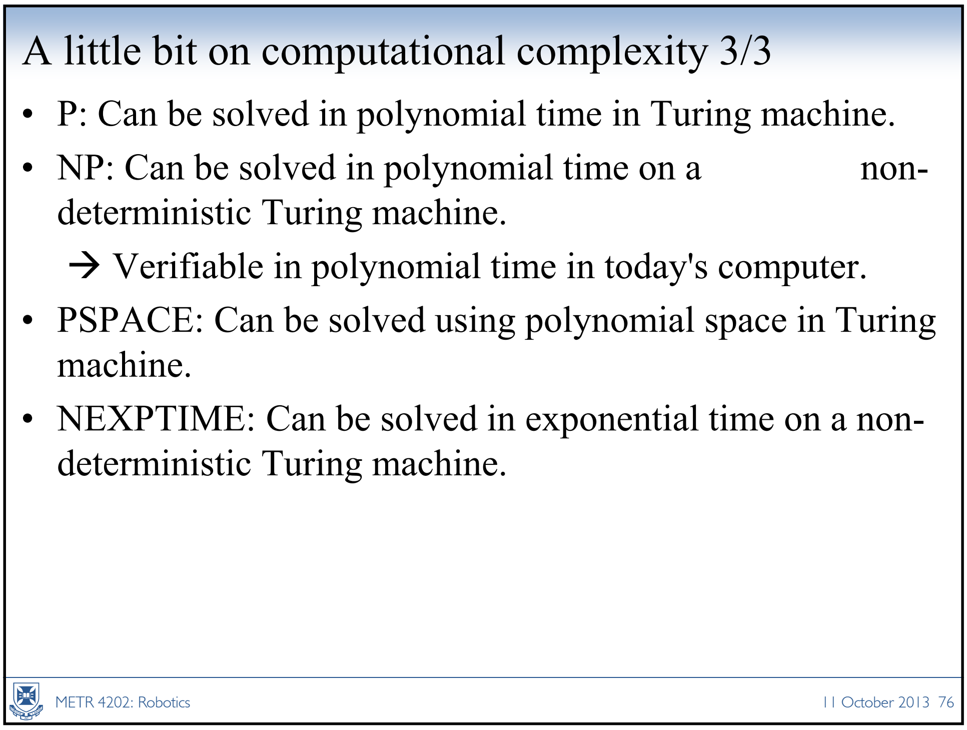


A little bit on computational complexity 3/3

- P: Can be solved in polynomial time in Turing machine.
- NP: Can be solved in polynomial time on a non-deterministic Turing machine.

  → Verifiable in polynomial time in today's computer.
- PSPACE: Can be solved using polynomial space in Turing machine.

EXPSPACE
EXPTIME
PSPACE
NP
P
NL



A little bit on computational complexity 3/3

- P: Can be solved in polynomial time in Turing machine.
- NP: Can be solved in polynomial time on a non-deterministic Turing machine.

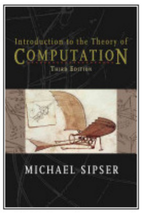
  → Verifiable in polynomial time in today's computer.
- PSPACE: Can be solved using polynomial space in Turing machine.
- NEXPTIME: Can be solved in exponential time on a non-deterministic Turing machine.

A little bit on computational complexity 3/3

- P: Can be solved in polynomial time in Turing machine.
- NP: Can be solved in polynomial time on a non-deterministic Turing machine.

  → Verifiable in polynomial time in today's computer.
- PSPACE: Can be solved using polynomial space in Turing machine.
- NEXPTIME: Can be solved in exponential time on a non-deterministic Turing machine.

Want to know more ?
Introduction to the Theory of Computation
by Michael Sipser.



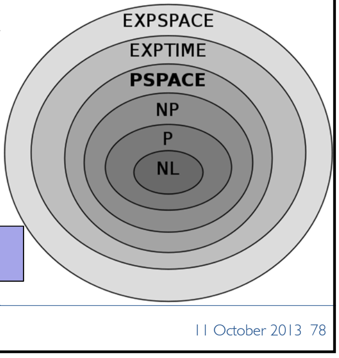
Problem hardness

- Finding a motion strategy for:
  - A point robot operating in 3D environment, where obstacles are planar walls.
  - To move from a known initial configuration to a point in a given goal region.
  - Control error: Bounded velocity error.
  - Sensing error: Bounded localization error.

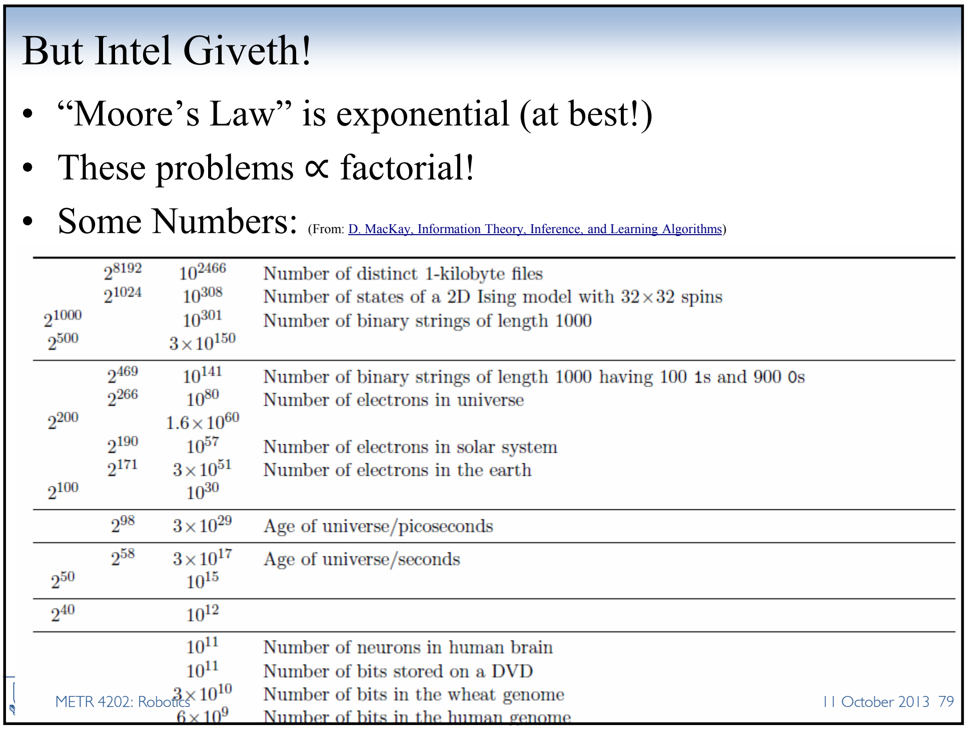
  is PSPACE-hard [Natarajan'86].

  is NEXPTIME-hard [Cany & Reif'87].

  input size: number of planar walls.



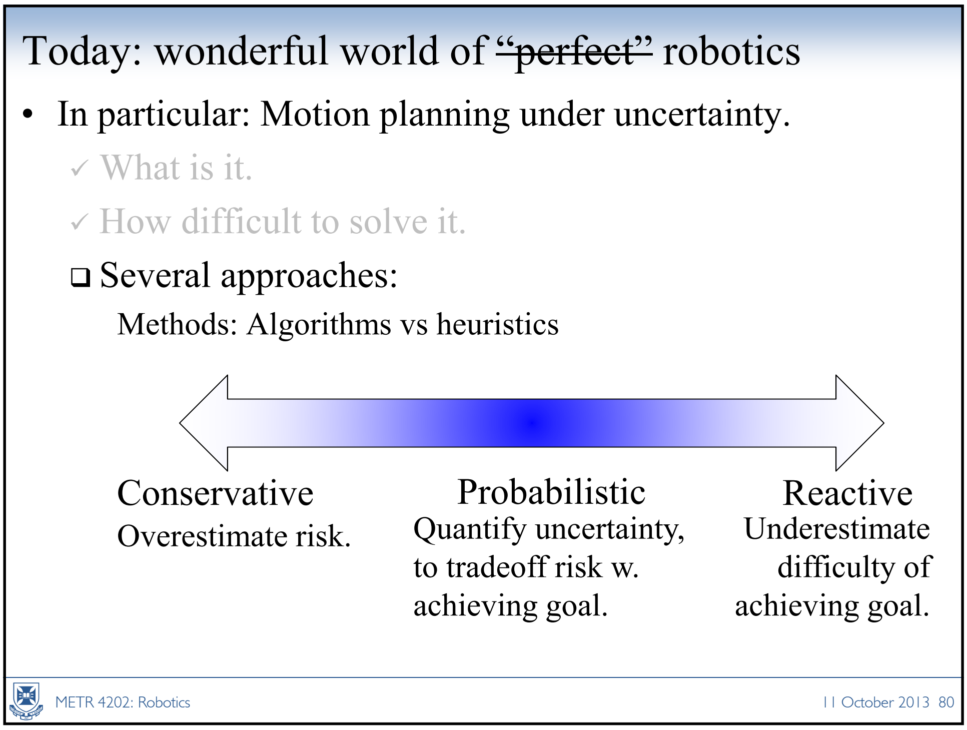
Ok, it's hard… So, what should we do ?



But Intel Giveth!

- "Moore's Law" is exponential (at best!)
- These problems ∝ factorial!
- Some Numbers: (From: D. MacKay, Information Theory, Inference, and Learning Algorithms)

	2^{8192}	10^{2466}	Number of distinct 1-kilobyte files
	2^{1024}	10^{308}	Number of states of a 2D Ising model with 32×32 spins
2^{1000}		10^{301}	Number of binary strings of length 1000
2^{500}		3×10^{150}	
	2^{469}	10^{141}	Number of binary strings of length 1000 having 100 1s and 900 0s
	2^{266}	10^{80}	Number of electrons in universe
2^{200}		1.6×10^{60}	
	2^{190}	10^{57}	Number of electrons in solar system
	2^{171}	3×10^{51}	Number of electrons in the earth
2^{100}		10^{30}	
	2^{98}	3×10^{29}	Age of universe/picoseconds
	2^{58}	3×10^{17}	Age of universe/seconds
2^{50}		10^{15}	
2^{40}		10^{12}	
		10^{11}	Number of neurons in human brain
		10^{11}	Number of bits stored on a DVD
		3×10^{10}	Number of bits in the wheat genome
		6×10^{9}	Number of bits in the human genome



Today: wonderful world of ~~"perfect"~~ robotics

- In particular: Motion planning under uncertainty.
  - ✓ What is it.
  - ✓ How difficult to solve it.
  - ❑ Several approaches:

    Methods: Algorithms vs heuristics

Conservative
Overestimate risk.

Probabilistic
Quantify uncertainty, to tradeoff risk w. achieving goal.

Reactive
Underestimate difficulty of achieving goal.

Probability review 1/4: Probabilistic Modeling

- View:
  - Experiments with random outcome.
  - Quantifiable properties of the outcome.
- Three components:
  - Sample space: Set of all possible outcomes.
  - Events: Subsets of sample space.
  - Probability: Quantify how likely an event occurs.

Probability review 2/4: Probability

- Probability: A function that maps events to real numbers satisfying these axioms:
  1. Non-negativity: $P(E) \quad 0$, where E is an ever
  2. Normalization: $P(S) \quad 1$, where S is the sample space.
  3. Additivity of finite / countably infinite events.

$$P \ \bigcup_{i\ 1}^{/n} E_i \qquad \underset{i\ 1}{\overset{/n}{}} \ P \ E_i \ ,$$

where E $_i$ are disjoint / mutually exclusive, i : natural number.

Probability review 3/4: Random Variables

- Interest is on numerical values associated w. samples, e.g.:
  - Sample 50 students enrolled in METR4202, what's the major of most of the students.
  - Roll a fair dice, get $5 if the outcome is even, & loose $5 if the outcome is odd.
- Random variable X is a function $X : S \rightarrow Num$.
  - Num: countable set (e.g., integer) → discrete random variable.
  - Num: uncountable set (e.g., real) → continuous random variable.

Probability review 4/4: Characterizing Random Variables

- Cumulative distribution function (cdf)

$$F_X(x) = P(X \le x) = P(\{s|X(s) \le x, s \in S\})$$

- Discrete: Probability mass function (pmf)

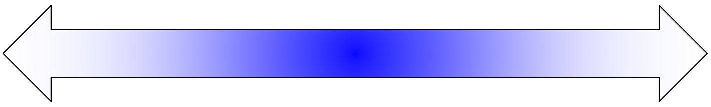
$$f_X(x) = P(X = x)$$

- Continuous: Probability density function/probability distribution function (pdf)

$$f_X(x) = \frac{dF_X(x)}{dx} \quad ; \quad P(a \le X \le b) = \int_a^b f_X(x)\,dx$$

Today: wonderful world of ~~“perfect”~~ robotics

- In particular: Motion planning under uncertainty.
  - ✓ What it is.
  - ✓ How difficult to solve it.
  - ❑ Several approaches:



✓Conservative

Probabilistic

POMDP

✓Probability review.

- POMDP.
- POMDP solvers.

✓Reactive

Cool Robotics Share

Universal Gripper

U. Chicago, Cornell, iRobot
May 2010