Robotics: Science & Systems
[Introduction]

Prof. Sethu Vijayakumar
Course webpage: http://wcms.inf.ed.ac.uk/ipab/rss
Lectures

• **Professor Sethu Vijayakumar**
  – Kinematics, Dynamics, Control, Learning

• **Dr. Zhibin (Alex) Li**
  – Planning, Localisation, Decision Making

• **Professor Robert Fisher, Dr. Michael Herrmann**
  – Computer Vision, Object Recognition, Categorisation

• **09:00-10:50 [Lecture attendance is essential]**
  – Mondays and Thursday [7 Bristo Square, LT2]
Practicals

• Two Groups
• Teams of 2-3 people
• Group 1 Lab: Monday 11.00 - 13.00
• Group 2 Lab: Thursdays 11.00 - 13.00
• Venue: (Forrest Hill G.A11)
• Demonstrators: Dr. Vladimir Ivan & Wolfgang Merkt
Key challenges due to
1. Close interaction with multiple objects
2. Multiple contacts
3. Hard to model non-linear dynamics
4. Guarantees for safe operations
5. Highly constrained environment
6. Under significant autonomy
7. Noisy sensing with occlusions

...classical methods do not scale!
DARPA Robotics Challenge
The story of the DARPA Robotics Challenge (DRC) begins on March 12, 2011, the day after the Tohoku, Japan earthquake and tsunami struck the Fukushima-Daiichi nuclear power plant. On that day, a team of plant workers set out to enter the darkened reactor buildings and manually vent accumulated hydrogen to the atmosphere. Unfortunately, the vent team soon encountered the maximum level of radiation allowed for humans and had to turn back. In the days that followed, with the vents still closed, hydrogen built up in each of three reactor buildings, fueling explosions that extensively damaged the facility, contaminated the environment and drastically complicated stabilization and remediation of the site.

At Fukushima, having a robot with the ability to open valves to vent the reactor buildings might have made all the difference. But to open a valve, a robot first has to be able to get to it. The DRC tasks test some of the mobility, dexterity, manipulation and perception skills a robot needs to be effective in disaster response.
Why Robotics?

• Robotics as a **scientific tool** for Fundamental Research

(Machine Intelligence, AI, Computer Science, Computer Vision, Language)

Why do plants have no *brains*? Because they do not *move* ...

– motion needs control and decision making
  ↔ Fast information processing
– motion needs anticipation and planning
– motion needs perception
– motion needs spatial representation
Aim: The Bigger Picture

• Machines that **autonomously** perform **intelligent** tasks in the real world
What does ‘autonomous’ mean?

No human in the control loop (automatic – “self-moving”)
Not attached to anything for power or processing (self-contained in operation)
Capable of maintaining behaviour against disturbance (autopilot – “self-regulating” – cybernetic)
Generates own capabilities (self-organising)
Not dependent on human intervention to survive (self-sufficient)
Generates own goals (self-governing - autonomous)
Generates own existence (autopoietic – “self-producing”)

Prof. Sethu Vijayakumar // R:SS 2016
What does ‘autonomous’ mean?

Crucial aspects of autonomy for this course are:

• The system can achieve a task on its own

• The system is affected by and affects the real world around it *directly*, with no intervention (at least for the duration of its task)

As a consequence we have a closed loop:

• Output affects subsequent input (and task achievement) in ways governed by real world physics (e.g. time, forces, materials...)

Prof. Sethu Vijayakumar // R:SS 2016
What does ‘intelligent’ mean?

- Can carry out a task that requires more than a pre-programmed sequence, e.g., with decision points depending on the real state of the world
- Adapts to dynamic environments
- Can plan (and re-plan) appropriate actions given high-level goals
- Learns to improve performance from experience
What is hard?

• Intrinsic uncertainty is inherent to robotics
• A robot’s knowledge of the problem is limited to what it has been told and what its sensors can tell it
  – Typically high level prior info
  – Typically limited sensor range
• The actual effects of a robot’s actions is usually uncertain
  – And the world might change
# Different Approaches to the Problem

<table>
<thead>
<tr>
<th>Model-based</th>
<th>Assume everything is known, or engineer robot or situation so this is approximately true</th>
<th>sense $\rightarrow$ plan $\rightarrow$ act</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principled but brittle</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reactive</th>
<th>Assume nothing is known, use immediate input for control in multiple tight feedback loops</th>
<th>sense $\rightarrow$ act</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robust and cheap but unprincipled</td>
<td></td>
<td>sense $\rightarrow$ act</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hybrid</th>
<th>Plan for ideal world, react to deal with run-time error</th>
<th>plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best and worst of both ?</td>
<td></td>
<td>sense $\rightarrow$ act</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Probabilistic</th>
<th>Explicitly model what is not known</th>
<th>sense $\rightarrow$ plan $\rightarrow$ act</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principled, robust but computationally expensive</td>
<td></td>
<td>with uncertainty</td>
</tr>
</tbody>
</table>
What is this course intended for?

• Give you sufficient exposure to fundamental topics relevant to robotics
  – Sensing (mainly vision)
  – Planning
  – Dynamics, Kinematics and Control

• Give you hands on (practical) experience in conceptualising a robotic solution to a problem
  – Build a robot (by making design decision)
  – Program it
  – Compete in a real-world environment
Lectures: Four Key Themes

• **Generating Motion**
  
  *goal: orchestrate joint movements for desired endeffector movements*
  
  (kinematics chains, Jacobian, inverse kinematics, multiple tasks, collision, dynamics and control, operational space control, singularities)

• **Planning and Optimization**
  
  *goal: planning around obstacles, optimizing trajectories*
  
  (path finding, sampling based methods, configuration space, RRTs, differential constraints, metrics, trajectory optimization, cost function, Dynamic Prog.)

• **Robot Vision**
  
  *goal: identifying, localizing and tracking objects*

• **Mobile Robots**
  
  *goal: localize and map yourself; walk, navigate*
  
  (state estimation problem, Bayesian filter, Odometry, Particle & Kalman filter, simultaneous localization and mapping (SLAM))
Robotics: Science & Systems
[Topic 1: 3D Geometry]

Prof. Sethu Vijayakumar
Course webpage: http://wcms.inf.ed.ac.uk/ipab/rss
Rigid Body Position & Pose

Position

Pose = Position + Orientation
Rotation Matrices

• Properties

\[ R \in \mathbb{R}^{n \times n} \]

**orthonormal matrix**
(orthogonal vectors stay orthogonal, normal vectors stay normal)

\[ R^{-1} = R^\top \]

columns and rows are orthogonal unit vectors

\[ \det(R) = 1 \]

Let the new basis vectors be (for e.g.)

\[ e'_1 = R_{11}e_1 + R_{21}e_2 + R_{31}e_3 \]

Then, the coordinate transformation from frame \( e'_{1:3} \) to \( e_{1:3} \) is:

\[
R = \begin{pmatrix}
R_{11} & R_{12} & R_{13} \\
R_{21} & R_{22} & R_{23} \\
R_{31} & R_{32} & R_{33}
\end{pmatrix}
\]
Coordinate Transform

\[ x = \text{coordinates of red point in world coordinate frame } (\mathbf{o}, \mathbf{e}_{1:3}) \]
\[ x' = \text{coordinates of red point in coordinate frame } (\mathbf{o}', \mathbf{e}'_{1:3}) \]
\[ p = \text{coordinates of } \mathbf{o}' \text{ in world coordinate frame } (\mathbf{o}, \mathbf{e}_{1:3}) \]

\[ x = p + Rx' \]
Simple Rotation Matrices

• 2D

\[ R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \]

• 3D

\[ R_z(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \]

\[ R_y(\theta) = \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix} \]

\[ R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{pmatrix} \]
Rotation Matrix: Good & Bad

• Pros
  – Rotates vectors directly
  – Easy composition

• Cons
  – 9 numbers
  – Difficult to enforce constraints

Degrees of Freedom (DOF) of a Rotation Matrix
• $R^{3 \times 3}$ has 9 numbers
• 6 constraints (3 orthogonal, 3 normal)
• only 3 DOF

OK...then, can we represent with **minimal** (=3) independent parameters
Rotation: Euler Angles

- Describe rotations by consecutive rotations about different axes:
  “first rotate $\phi$ about $\hat{z}$, then $\theta$ about the new $\hat{x}'$, then $\psi$ about the new-new $\hat{z''}$”

- Z-Y-Z (3-1-3) representation
- yaw-pitch-roll or Z-Y-X (3-2-1) ....used in flight!
Euler Angles and Gimbal Lock

• Euler angles have a severe problem:
  – If two axes align: blocks 1 DOF
  – ‘singularity’ of Euler angles

• Pros
  – minimal representation
  – human readable

• Cons
  – Gimbal lock
  – must convert to matrix to rotate vector
  – no easy composition
Rotation: Rotation Vector

• Using 3 numbers...

  vector \( w \in \mathbb{R}^3 \)

  length \( |w| = \theta \) is rotation angle (in radians)
  direction of \( w = \) rotation axis \( (w = w/\theta) \)

• Pros
  – minimal representation
  – human readable

• Cons
  – singularity for small rotations
  – must convert to matrix to rotate vector
  – no easy composition
Rotation: Quarternion

- Maths tells: all scheme with 3 numbers will have a singularity
  A quaternion is $r \in \mathbb{R}^4$

\[
  r = \begin{pmatrix}
    r_0 \\
    r_1 \\
    r_2 \\
    r_3
  \end{pmatrix} = \begin{pmatrix}
    r_0 \\
    \bar{r}
  \end{pmatrix}
\]

\[
  r_0 = \cos(\theta/2)
\]

\[
  \bar{r} = \sin(\theta/2) \, w
\]

with $w$ = unit length rotation axis

Unit length constraint (to represent rotations)

\[
  r^\top r = r_0^2 + r_1^2 + r_2^2 + r_3^2 = 1
\]
Quaternions: Composition

• Conversion to/from matrix

\[
R(r) = \begin{pmatrix}
1 - r_{22} - r_{33} & r_{12} - r_{03} & r_{13} + r_{02} \\
 r_{12} + r_{03} & 1 - r_{11} - r_{33} & r_{23} - r_{01} \\
 r_{13} - r_{02} & r_{23} + r_{01} & 1 - r_{11} - r_{22}
\end{pmatrix}
\]

\[r_{ij} := 2r_ir_j.
\]

\[
r_0 = \frac{1}{2}\sqrt{1 + \text{tr}R}
\]

\[
r_3 = (R_{21} - R_{12})/(4r_0)
\]

\[
r_2 = (R_{13} - R_{31})/(4r_0)
\]

\[
r_1 = (R_{32} - R_{23})/(4r_0)
\]

• Composition

\[
r \circ r' = \begin{pmatrix}
r_0r'_0 - \vec{r}^T\vec{r}' \\
r_0\vec{r}' + r'_0\vec{r} + \vec{r}' \times \vec{r}
\end{pmatrix}
\]
Quaternions: Pros and Cons

• Pros
  – no singularity
  – almost minimal representation
  – easy to enforce constraints
  – easy composition
  – easy interpolation

• Cons
  – somewhat confusing
  – not quite minimal
  – must convert to matrix to rotate vector

• Summary of Rotation representations
  – need rotation matrix to rotate vectors
  – Quaternions good for free rotations
  – Euler angles OK for small angular deviations
    – but beware singularities!
Homogeneous Transformations

A compact way of representing coordinate transformations between two frames

- $x^A = \text{coordinates of a point in frame } A$
- $x^B = \text{coordinates of a point in frame } B$

- Translation and rotation: $x^A = t + Rx^B$

- Homogeneous transform $T \in \mathbb{R}^{4 \times 4}$:

$$T_{A \rightarrow B} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}$$

$$x^A = T_{A \rightarrow B} x^B = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x^B \\ 1 \end{pmatrix} = \begin{pmatrix} Rx^B + t \\ 1 \end{pmatrix}$$

In homogeneous transformations, we append 1 to all coordinate vectors.
Composition of transforms

\[ T_{W \rightarrow C} = T_{W \rightarrow A} T_{A \rightarrow B} T_{B \rightarrow C} \]

\[ x^W = T_{W \rightarrow A} T_{A \rightarrow B} T_{B \rightarrow C} x^C \]

\( x^W \) and \( x^C \) are the \textit{coordinates} of the red dot in frames \( W \) and \( C \)
Robotics: Science & Systems

[Practicals: Lab Intro]

Prof. Sethu Vijayakumar

Course webpage: http://wcms.inf.ed.ac.uk/ipab/rss
Building your Robot: (1) Strength

• Try to have smooth force lines, e.g. straight compression or tension, and appropriate balance
• Use short path/small number of components to transmit forces
• Top tip for Lego: use bracing
Building your Robot: (2) Stability

• Usually want structure as a whole to be statically stable, i.e. no net torque due to gravity

• Depends on centre of mass: force of gravity through centre of mass to ground must fall within base of support.
  – Minimum three points for base of support
  – Wider base of support and lower centre of mass will reduce potential tipping due to inertia

• Rotating around the centre of mass requires the least work
Building your Robot: (3) Friction

- Robot efficiency will depend on how much energy is dissipated through inefficient mechanisms and friction

- E.g. using differential drive and third sliding contact point
  - Force to reach threshold of motion = $\mu N$ where $\mu$ is co-efficient of friction, $N$ normal force
  - Force transferred to ground via wheel is also proportional to $\mu_{\text{wheel}} N$
  - $N = \text{mass} \times \text{gravity} \rightarrow$ so should reduce mass resting on the sliding contact and increase mass resting on drive wheels
  - $\mu$ depends on surfaces $\rightarrow$ should make sliding contact smooth, tire rough
  - N.B. does not depend on area of contact
Building your Robot: (3) Friction

• Want gear train to interlock precisely → needs to be in stiff structure (i.e. doesn’t deform under load

• Want to minimise friction of rotation
  – Avoid any direct contact of gear or wheel to frame
  – Minimise the bend in the axle beams
    • Bend is proportional to $dP/I$, where $I$ is beam inertia, depending on shape and material of beam
    • Reduce mass
    • Reduce distance
    • Add supports (opposing forces)
Building your Robot: (4) Power

• Have fixed amount of power, i.e. rate of work or force x distance/second

• Hence fundamental tradeoff between speed and force (torque) of your robot

• Primarily determined by the gear ratio $r_p/r_f$ where $r_p$ is radius of the powered gear, and $r_f$ the radius of the follower gear

• Gears act like levers:
  – distance/speed changes by $r_p/r_f$
  – Force/torque changes by $r_f/r_p$

• Same ratio can be calculated by counting teeth on each gear
Both ends of axle supported

Gear/wheel not touching surface, well aligned

Motor $\rightarrow$ 8:40 $\rightarrow$ 8:24 $\rightarrow$ 8:40 $\rightarrow$ Wheel

$1:5 \times 1:3 \times 1:5 = 1:75$

Lower ratio (e.g. 1:25) increases velocity but decreases acceleration.
Programming your robot: Overview

System components
- Navigation
  - Obstacle avoidance
  - Motion to target
- Behaviours
  - Homing
  - Resource collection
  - Visual servoing
  - Localisation
- Vision
  - Resource recognition
  - Marker detection

Interface libraries
- Robot Interface

3rd party libraries
- Phidgets library
  - OpenCV

Interface boards
- Interface board
- Motor board
- Servo board

Hardware
- Sensors
- Motors
- Servos
- Camera
Programming your robot: Components

• Navigation
  – Use robot interface to drive motors
  – Build logic for obstacle avoidance

• Vision
  – Use OpenCV to capture and process images
  – Build detection modules (landmarks, targets, ...)

• Behaviours
  – Use vision and navigation to build behaviour logic
Programming your robot:

Tips

• Build modular software
  – Each system component should run independently
  – Pass information between processes

• Use 3\textsuperscript{rd} party libraries
  – OpenCV, boost, ...

• Use multiple processing threads

• Ask for help!
Summary

• Robotics involves physical and computational problems
• Better physical solutions usually reduce the computational problems
  – Don’t try to fight physics!
• Lab: Getting going soon...