Introduction to Mobile Robotics and Computational Algorithms

Jason J. Gu , PhD, P. Eng.
Professor
Electrical and Computer Engineering
Dalhousie University, Halifax, Nova Scotia, Canada

Outline

- Part I: Introduction to Mobile Robot
- Part II: Kinematics Models of Mobile Robot
- Part III: Sensor Fusion of Mobile Robot
- Part IV: Pose and Map
- Part V: Representing and Reasoning
- Par VI: Control Algorithms of Mobile Robot
- Research on Robotics

Part I: Introduction to Mobile Robot

Introduction

History of Mobile Robots

Practical Mobile Robot Tasks

I.1 Introduction

- Robotics: Combination of four disciplines
  - Electrical Engineering – system integration, sensors, and communication
  - Computer Science – representations, sensing, and planning algorithms.
  - Mechanical Engineering – vehicle design and in particular locomotive mechanisms.
  - Cognitive Science – insights on how biological organisms solve similar problems.

I.1 Introduction

- What is Mobile Robot?
- A combination of various physical and computational components divided into four main subsystems.
  - Locomotion
  - Reasoning
  - Sensing
  - Communication

I.1 Introduction

- A: Locomotion: Power of motion from place to place
  - Kinematics: mathematics of motion without considering force
  - Dynamics: modeling motion using forces

I.1 Introduction

- B: Reasoning
  - Representing space: robot’s environment
  - Representing the robot: feature of the robot
  - Path planning

I.1 Introduction

- C: Sensing: Basic sensors in use: Odometry Sensors
  - Brush Encoders
  - Optical Encoders
  - Magnetic Encoders
  - Inductive Encoders
  - Capacitive Encoders
  - Potentiometers
  - Synchros
  - are rotating electromechenical devices used to transmit angular info with high precision
  - Resolvers
  - special configuration of syschros that develops voltages that proportional to the sine and cosine of the rotor angle

I.1 Introduction

- C: Sensing: Basic sensors in use:
- Non-Visual:
  - Bumpers
  - Accelerometers
  - Gyroscopes
  - Compasses
  - Infrared sensors
  - Sonar,
  - Rader
  - Laser range finder
  - GPS
- visual

I.1 Introduction

- D: Communication
  - Communication strategy: Wireless Ethernet, serial, infrared, audio-based.
  - Tethered: Communication + Power + Less Freedom
  - Un-tethered: Communication+ Less Power + More Freedom

I.1 Introduction-Categories of Mobile Robot

- Aquatic: using surrounding water to support propulsion
  - Torpedo-like structure: propeller (fan)
  - Twin-burger (text): collection of thrusters
- Flying Robot
  - Fixed Wing autonomous vehicle: ground station + GPS
  - Radio-controlled helicopters
  - Buoyant vehicles: energy efficient
- Space Robot
  - Free-flying systems: one or more manipulators are mounted on a thrusters-equipped spacecraft
- Terrestrial Robot: ground-contact-robots
  - Indoor + outdoor
  - Wheeled, tracked, legged

I.2 History of Mobile Robots

- Issac Asimou is the father of robots
  - A robot may not injure a human being, or, through interaction, allow a human being to come to harm.
  - A robot must obey the orders given by human being except when such orders would conflict with the first law.
  - A robot must protect its own existence as long as such protection does not conflict with the first or second laws.

I.2 History of Mobile Robots

- Autonomous robots in fiction
  - Issac Asimou: “Runway”, “Robots of Empire and Robots of dawn”
- “The Day the Earth Stood Still”, Gort is a policeman
- Autonomous robots in movie
  - The Jetsons: Rosie
  - Lost in space: Robot
  - Station’s fall and planet fall: Floyd
  - Star Wars: R2D2 and C3P0
  - Star Trek: Data and Borg

I.2 History of Mobile Robots

- Real robotics started in 1890s.
- 1890s: Nikola Tesla: Radio-controlled vehicle
- 1940s: Norbert Wiener: Automatics antiaircraft gun
- 1950-1951: W. Grey Walter: Electronic turtle: phototube eyes, microphone eyes, contact-switch feelers.
- Late 1950s: Minsky, Richard Greenblood etc: Ping-Pong playing robot: Basket instead of grip.
- 1966-1972: Nils Nilssen, Charles Rosen, etc, at Stanford Research Institute developed Shakey(1969): 5-foot tall, two stepper motors, differential drive, bumpers, range finders, television camera.

I.2 History of Mobile Robots

- 1970s: Stanford Cart: obstacle avoidance using camera
    - JPL Rover: Jet Propulsion Laboratory in Pasadena: TV camera, Laser rangefinder and tactile sensors. Navigation: Dead-reckoning
- HILARE: Toulouse: Computer vision, laser range finder, ultra-sonic
- Above robots are mobile wheeled Robots.
- Legged Robots: Appear in 1960. G.E. Quadruped. Four legs: three simple joints.
- Other Mobile Robot Design:
    - Pet robot. (Sony dog)
    - Snake Robot
    - ROV

I.3 Practical Mobile Robot Tasks

- Practical Robots Properties:
    - The environment is inhospitable, and sending a human is either very costly or very dangerous such as nuclear, chemical, underwater, battle field (4 soldiers died), and outer-space environment.
    - The environment is remote. Mining, outer-space, forestry exhibit.
    - The task has a very demanding duty cycle or very high fatigue factor.
    - The task is highly disagreeable to humans.

I.3 Practical Mobile Robot Tasks

- Delivery:
  - Helpmate robot: Metric map, wireless communication to operate doors elevators
  - FIRST robot: strategically placed laser-scan able targets.
  - AGV: cargo handling
  - MARTHA: Multiple Autonomous Robots for Transport and Handling Applications
- Robotic Assembly and Manufacturing
  - Mobile Robots onto the shop floor
- Intelligent Vehicles
  - Driving Assistant: more sensors to enhance safety and efficiency.
  - Convoy systems: lead vehicle by human, rest is following
    - Autonomous driving system: CMU (Navlab)
    - Autonomous Highway system
    - Autonomous Urban system

I.3 Practical Mobile Robot Tasks

- Robots for survey and inspection
- Mining automation
- Space Robotics
  - NASA JPL, Soviet, LUNS Corp
- Autonomous Aircraft
- Military Reconnaissance
- Bomb and Mine Disposal
- Underwater inspection

I.3 Practical Mobile Robot Tasks

- Agriculture and Forestry
  - Forestry maintenance, Greenhouse Robot, lawn mowing robot, harvesting robots
- Aids for disabled
- Entertainment
- Cleaning Robots

Part II: Kinematics Models of Mobile Robots

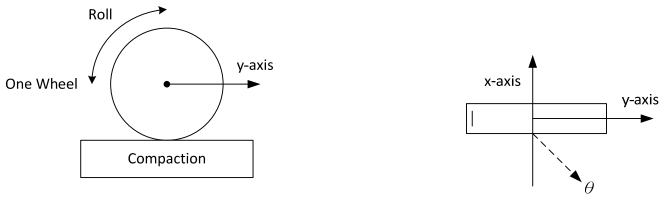
Kinematics of Wheeled Robots

Kinematics of Legged Robots

Kinematics of Other Mobile Robots

II.1 Kinematics of Wheeled Robots

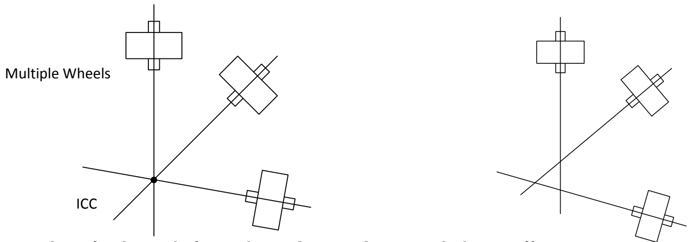
- Rolling Wheels



Measure wheel rotation to estimate motion
Control *(x, y,θ)* to obtain robot pose

II.1 Kinematics of Wheeled Robots

- For all wheels in contact with the ground to roll the motion of each of the vehicle's wheels must be along its own y axis.



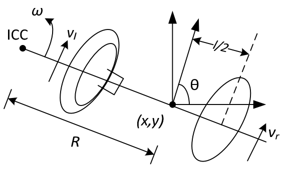
For a wheeled mobile robot (WMR) to exhibit rolling motion, a point must exist around which each wheel on the vehicle follows a circular course. This point is known as instantaneous centre of curvature (ICC) or the instantaneous centre of rotation (ICR).

II.1 Kinematics of Wheeled Robots

- Remarks:
    - If all of the wheels in contact with the ground are to exhibit rolling contact, then not only must the ICC exist, but each wheel's velocity must be consistent with a rigid rotation of the entire vehicle about the ICC.
    - For the vehicle to change its ICC. Some property of the wheels must be changed.
    - A vehicle located on a plane has three degree of freedom. An *(x,y)* position and a heading or orientation θ.
    - Mobile robots usually don't have complete independent control of all three pose parameters.

II.1 Kinematics of Wheeled Robots

- Differential drive: Robot consists of two wheels mounted on a common axis controlled by separate motors
    - A common system: 2 active drive wheels + castors
    - High resolution encoders to measure pos. v and a.
    - Dead reckoning navigation



l: distance along the axle between the center of two wheels
v_l: left wheel velocity
v_r: right wheel velocity
R: signed distance

II.1 Kinematics of Wheeled Robots

- Remark: ***R, ω, vl, vr, ICC*** are all functions of time

We have	$\begin{cases}\omega\left(R+{}^{l}/_{2}\right)=v_r\\ \omega\left(R+{}^{l}/_{2}\right)=v_l\end{cases}$	=>	$\begin{cases}R={}^{l}/_{2}\left(\frac{v_l+v_r}{v_r-v_l}\right)\\ \omega=\frac{v_r-v_l}{l}\end{cases}$

Special case: 1. $v_l=v_r$, $R\rightarrow\boldsymbol{\infty}$, $\omega=\mathbf{0}$

2. $v_l=-v_r$, $R=\mathbf{0}$ robot rotates about a point midway between two wheels.

3. others. ICC follow curved trajectory

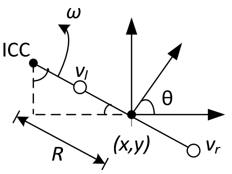
II.1 Kinematics of Wheeled Robots

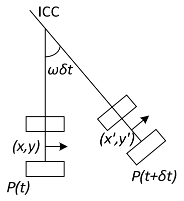
- Forward kinematics for differential drive robots

robot is at position *(x,y)*, making an angleθwith *x* axis.

We can find out $ICC=[x-R\sin\theta\,,y+R\cos\theta]$

$$\Rightarrow ICC_x=x-R\sin\theta, ICC_y=y+R\cos\theta$$





How to find out *x′, y′, ϑ′*

Easy to see $\theta'=\theta+\omega\delta t$

II.1 Kinematics of Wheeled Robots

$$ICC_x = x' - R\sin(\theta + \omega\delta t)$$

$$ICC_y = y' + R\cos(\theta + \omega\delta t)$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ICC_x \\ ICC_y \end{bmatrix} + \begin{bmatrix} R\sin(\theta + \omega\delta t) \\ -R\cos(\theta + \omega\delta t) \end{bmatrix}$$

$$= \begin{bmatrix} ICC_x \\ ICC_y \end{bmatrix} + \begin{bmatrix} R\sin\theta\cos\omega\delta t + R\cos\theta\sin\omega\delta t \\ R\sin\theta\sin\omega\delta t - R\cos\theta\cos\omega\delta t \end{bmatrix}$$

$$= \begin{bmatrix} ICC_x \\ ICC_y \end{bmatrix} + \begin{bmatrix} \cos\omega\delta t & \sin\omega\delta t \\ \sin\omega\delta t & -\cos\omega\delta t \end{bmatrix} \begin{bmatrix} R\sin\theta \\ R\cos\theta \end{bmatrix}$$

$$= \begin{bmatrix} ICC_x \\ ICC_y \end{bmatrix} + \begin{bmatrix} \cos\omega\delta t & -\sin\omega\delta t \\ \sin\omega\delta t & \cos\omega\delta t \end{bmatrix} \begin{bmatrix} R\sin\theta \\ -R\cos\theta \end{bmatrix}$$

Recall $\begin{cases} ICC_x = x - R\sin\theta \\ ICC_y = y + R\cos\theta \end{cases} \Rightarrow \begin{matrix} R\sin\theta = x - ICC_x \\ -R\cos\theta = y - ICC_y \end{matrix}$

So $\left.\begin{matrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ICC_x \\ ICC_y \end{bmatrix} + \begin{bmatrix} \cos\omega\delta t & -\sin\omega\delta t \\ \sin\omega\delta t & \cos\omega\delta t \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \end{bmatrix} \\ \theta' = \theta + \omega\delta t \end{matrix}\right\}$

$$\Rightarrow \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos\omega\delta t & -\sin\omega\delta t & 0 \\ \sin\omega\delta t & \cos\omega\delta t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{bmatrix}$$

- Forward kinematics for differential drive robots

II.1 Kinematics of Wheeled Robots

For Robot moves in direction *ϑ(t)* at given velocity *v(t)*

$$\begin{cases} x(t) = \int_0^t v(t)\cos[\theta(t)]\,dt \\ y(t) = \int_0^t v(t)\sin[\theta(t)]\,dt \\ \theta(t) = \int_0^t \omega(t)dt \end{cases}$$

For differential drive vehicle

$$v(t) = \frac{v_l(t) + v_r(t)}{2}$$

$$\theta(t) = \frac{v_r(t) - v_l(t)}{l}$$

$$\Rightarrow \begin{cases} x(t) = {}^1\!/_2 \int_0^t [v_r(t) + v_l(t)]\cos[\theta(t)]\,dt \\ y(t) = {}^1\!/_2 \int_0^t [v_r(t) + v_l(t)]\sin[\theta(t)]\,dt \\ \theta(t) = {}^1\!/_l \int_0^t [v_r(t) - v_l(t)]dt \end{cases}$$

- Forward kinematics for differential drive robots

II.1 Kinematics of Wheeled Robots

$v_l(t) = v_l \quad v_r(t) = v_r \quad v_l \neq v_r$ Find inverse kinematics.

$$\begin{cases} x(t) = \frac{1}{2}\int_0^t (v_r + v_l)\cos[\theta(t)]\,dt & (1) \\ y(t) = \frac{1}{2}\int_0^t (v_r + v_l)\sin[\theta(t)]\,dt & (2) \\ \theta(t) = \frac{1}{l}\int_0^t (v_r - v_l)\,dt \end{cases}$$

$\Rightarrow \theta(t) = \frac{t}{l}(v_r - v_l)$, substitute into (1) & (2)

$$\begin{cases} x(t) = \frac{1}{2}\int_0^t (v_r + v_l)\cos\left[\frac{t}{l}(v_r - v_l)\right]dt = \frac{1}{2}(v_r + v_l)\cdot\frac{1}{\frac{v_r - v_l}{l}}\sin\left[\frac{t}{l}(v_r - v_l)\right] \\ \quad = \frac{l}{2}\frac{v_r + v_l}{v_r - v_l}\sin\left[\frac{t}{l}(v_r - v_l)\right] \\ y(t) = \frac{1}{2}\int_0^t [v_r + v_l]\sin\left[\frac{t}{l}(v_r - v_l)\right]dt = -\frac{1}{2}\frac{v_r + v_l}{v_r - v_l}\cdot l\cos\frac{t}{l}(v_r - v_l)\Big|_0^t \\ \quad = \frac{l}{2}\frac{v_r + v_l}{v_r - v_l} - \frac{l}{2}\frac{v_r + v_l}{v_r - v_l}\cos\frac{t}{l}(v_r - v_l)\Big| \\ \theta(t) = \frac{t}{l}(v_r - v_l) \end{cases}$$

Given *x(t), y(t), t* to solve v_r and v_l

- Example 2.1 : Inverse kinematics for differential drive robots

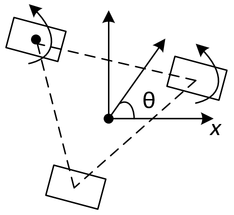
---

II.1 Kinematics of Wheeled Robots

- Synchronous drive (Synchro drive)

  Each wheel is capable of being driven and steered. Typical configurations involve three steered wheels arranged at the vertices of an equilateral triangle. All wheels point at same direction and turn at the same rate.

  (Two: one for direction, one for rollng)



How many motors?
Commercial Robot: Nomadics 200 and Robotics B21
Compare differential drive and synchro drive.
Both are sensitive to small variations in ground plane.
Differential drive hard to control pose $\theta + (x, y)$
synchro drive control θ independently.

II.1 Kinematics of Wheeled Robots

- Synchronous drive (Synchro drive)

Forward kinematics for synchronous drive

Rotate: ω, translational speed: v

Forward kinematics:

$$\theta(t) = \int_0^t \omega(t)dt$$

$$x(t) = \int_0^t v(t)\cos[\theta(t)]\,dt$$

$$y(t) = \int_0^t v(t)\sin[\theta(t)]\,dt$$

What is the ICC? Since all wheels point at same direction, ICC is at infinity.

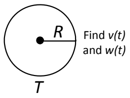
II.1 Kinematics of Wheeled Robots

- Synchronous drive (Synchro drive)

Inverse kinematics: Two special cases:

(1) $v(t) = \mathbf{0}$, $\omega(t) = \omega$ for some period δt , then robot rotate, in place by $\omega\delta t$

(2) $\omega(t) = \mathbf{0}$, $v(t) = v$ for some period δt , then the robot moves in the direction it is pointing a distance $v\delta t$



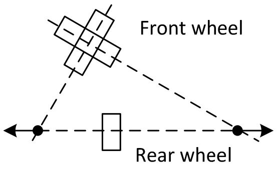
(1) Steered wheels:
a wheel for which the orientation of the rotational axis of the wheel can be controlled.

Differential drive: two wheels are steered wheels
Synchronous drive: all wheels are steered wheels

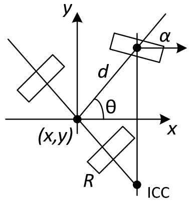
II.1 Kinematics of Wheeled Robots

- Tricycle, bogey, and bicycle drive: similar kinematics

Example: bicycle



Example: Tricycle



Forward kinematics for steered vehicle

$$R = d\tan\left(\frac{\pi}{2}-\alpha\right)$$

$$v = \omega\cdot\sqrt{d^2+R^2}$$

$$\begin{cases} ICC_x = x + R\sin\theta \\ ICC_y = y - R\cos\theta \end{cases}$$

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos\omega\delta t & -\sin\omega\delta t & 0 \\ \sin\omega\delta t & \cos\omega\delta t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{bmatrix}$$

II.1 Kinematics of Wheeled Robots

- Tricycle, bogey, and bicycle drive: similar kinematics

Inverse kinematics:

Two special cases: (1) $\alpha = 0$. Robot drives straight ahead

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x + v\cos\delta t \\ y + v\sin\delta t \\ \theta \end{bmatrix}$$

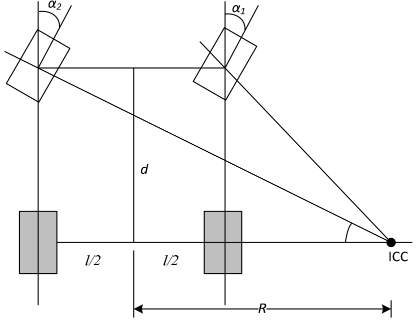
(2) $\alpha = 90^\circ$

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \pm \frac{v\delta t}{d} \end{pmatrix}$$

II.1 Kinematics of Wheeled Robots

- Car Drive (Ackerman steering)

  Ackerman steering: also known as kingpin steering.
  Steering wheels: front wheels.
  The front wheels each rotate on separate arms so as to be able to rotate different amount to point at the ICC.



$\alpha_1 \neq \alpha_2$, why?

$$R + {}^{l}/_{2} = d\tan({}^{\pi}/_{2} - \alpha_2) = d\cot\alpha_2$$

$$R - {}^{l}/_{2} = d\tan({}^{\pi}/_{2} - \alpha_1) = d\cot\alpha_1$$

$$\Rightarrow {}^{l}/_{d} = \cot\alpha_2 - \cot\alpha_1$$

l : lateral wheel separation
d : congitudinal wheel separation
α_1 : relative angle of inside wheel
α_2 : relative angle of outside wheel

II.1 Kinematics of Wheeled Robots

- Complex wheels

Complex wheels are wheels that exhibit more than one preferred rolling direction. Such wheel can result in omnidirectional robots.

[120] N/A One wheel robot: Xu Yangsheng
[257] Six wheeled omni-directional robot. USU
[316] No Photo

Remarks: omnidirectional robot are easier to control than robots based on simple wheels because the motion of the robot is independent of the (x, y, θ) pose of the robot.

II.1 Kinematics of Wheeled Robots

- Tracked Vehicles

Remarks: Similar to differential drive vehicles in terms of their kinematics but are more robust to terrain variations.

Remarks: hard to determine the motion

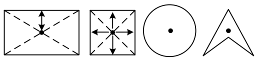
Solution: Add castor or omnidirectional wheel to measure motion.

II.2 Kinematics of Legged Robots

- Introduction
  - Why legged robots?
    wheeled or tracked vehicle: need continuous ground support,
    legged: rough terrains, planetary, natural, ...
  - Vehicle stability

Static stability: If a limbed robot is designed so that its balance is maintained at all times even if all its legs were to freeze in position.

Stability margin: measure of the current stability of the robot. It is defined as the minimum distance from the vertical projection to the robot’s center of gravity to the boundary of the vertical projection of the convex hull of the support polygon.



II.2 Kinematics of Legged Robots

- Dynamic stability:
  - If the center of gravity of the robot is allowed to move outside of the convex hull of the support polygon and the robot moves is a controlled manner, the robot is said to exhibit dynamic stability.

Example: One leg, two legged, four legged machine.
How about three legged?
Number of legs: least one leg.
Common: four, six and eight, twelve
For statically stable robot: How many? Minimum four legs, three to support, one to move.
For dynamic stability: How many legs are required? Minimum: one leg.
Limb design and control
Simple joint: single degree of freedom
Three-dimensional spare: minimum three joints
Joint classes: (1) Prismatic: introduce translation
(2) Rotational: Introduce rotation

II.2 Kinematics of Legged Robots

- Forward and inverse kinematics

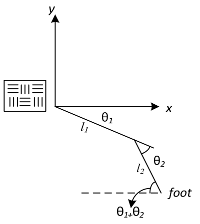
Forward kinematics:

$$\theta(\text{joint space}) \Rightarrow \begin{matrix}x\\y\\z\end{matrix}(\text{Cartesian space})$$

Inverse kinematics:

$$\begin{matrix}x\\y\\z\end{matrix}(\text{Cartesian space}) \Rightarrow \theta(\text{joint space})$$

Example: Forward kinematics: know $\theta_1\ \ \theta_2\ \ l_1\ \ l_2 \Rightarrow (x, y)$



(1) $x = l_1 \cos\theta_1 + l_2(\cos(\theta_1 + \theta_2))$

(2) $y = l_1 \sin\theta_1 + l_2(\sin(\theta_1 + \theta_2))$

Forward Kinematics

$(1)^2+(2)^2$ $\Rightarrow \cos\theta_2 = \dfrac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}$

Sub into (1) and get $\theta_1 = f(x, y, l_1, l_2)$

Inverse kinematics

Remarks: For inverse kinematics, solution may be not unique.

II.2 Kinematics of Legged Robots

- Forward and inverse kinematics

$\dot{x}$ velocity of end-effectors, $\dot{\theta}$ velocity in joint space, $(\theta_1\ \theta_2)$

then $\dot{x} = J(\theta)\dot{\theta}$ $J(\theta)$ is the Jacobian matrix

$$J(\theta) = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} \end{bmatrix} \qquad \dot{x} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \qquad \dot{\theta} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

So if we know the velocity of $\dot{\theta}$ we can derive $\dot{x}$

Singularities:

when a limb loses one or more degree of freedom, such as the boundary of workspace, singularities occurs, which means it is impossible to move the limb in an arbitrary direction from this point.

II.2 Kinematics of Legged Robots

- Forward and inverse kinematics

Recall $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} l_1 \cos\theta_1 + l_2(\cos(\theta_1 + \theta_2)) \\ l_1 \sin\theta_1 + l_2(\sin(\theta_1 + \theta_2)) \end{pmatrix}$

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

$$|J| = \begin{vmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{vmatrix}$$

$$= -l_1 l_2 \sin\theta_1 \cos(\theta_1 + \theta_2) + l_1 l_2 \cos\theta_1 \sin(\theta_1 + \theta_2) = l_1 l_2 \sin\theta_2$$

when $|J| = 0,$ $\sin\theta_2 = 0, \theta_2 = k\pi$ J is not full rank.

What is the position of the robot?

Stretched out or bent in.

Kinematics of Robotic Arm

- Ref: "Introduction to Robotics"
- "John J. Craig"
- Addison-Wesley Publishing Company

II.3 Kinematics of Kinematics of other Mobile Robots

- Aquatic vehicle:
  - thrusters, propeller
- Fly vehicle:
  - Helicopter control, fixed-wing control, Buoyant vehicle
- Space robot:
  - robot manipulator: advantage? micro gravity

Part III: Sensor Fusion of Mobile Robots

Brief Review of Sensor Fusion (slides)
Non-visual Sensors and Algorithms
Visual Sensors and Algorithms
Introduction to Kalman Filtering

III.1 Brief Review of Sensor Fusion

- What is sensor fusion?
- Why need sensor fusion?
- Example of sensor fusion?
- Visual sensing:
    - Use light reflected from objects to reason about structure
- Non visual sensing:
    - Use audio, inertial and other modalities to sense the environment

III.1 Brief Review of Sensor Fusion

- Classification

- Internal state sensors: internal parameters: battery level, wheel positions, joint angles
- External state sensors: aspects of the world: humidity, color
  - Contact sensors
  - Non contact sensors

- Active sensors: make observations by emitting energy into the environment or by modifying the environment
- Passive sensors: passively receive energy to make their observations (preferred, why: nonistrusive)

III.1 Brief Review of Sensor Fusion

Properties of sensors:

- Real sensors are noisy
- Real sensors return an incomplete description of the environment
- Real sensors cannot usually be modeled completely

Four main classes of sensors based on return data

- 1)Range sensor: return measurement of distance between the sensor and objects
- 2)Absolute position sensors: return position of robot
- 3)Environmental sensors: return properties of the environment:
  - Temperature
  - Color
  - humidity
- 4)Inertial sensors: differential properties of robot's position:
  - Acceleration
  - Velocity

III.1 Brief Review of Sensor Fusion

- Important sensor properties:
    - Speed of operation: measurement rate (continuously), or delay
    - Cost: price of sensors: gyroscope (several thousand), emitter-detector (a few cents)
    - Error rate: average error, number of outliers (wildly incorrect measurements) missed measurement
    - Robustness: sensor tolerates various deviations: Physical disturbance, environmental noise
    - Computational requirements: contact sensor (simple), computer vision (huge computation)
    - Power, weight and size requirements

III.2 Non-visual Sensors and Algorithms

- Tactile sensors (contact sensors: bumpers)
    - Concept: Detect physical contact with a object
    - Micro switches: on and off binary value
    - Whisker sensors: diagram
    - Strain gauges: give more information than binary micro switches (change in resistance) (force?)
    - Piezoelectric transducers: force voltage output

III.2 Non-visual Sensors and Algorithms

- Infrared sensors: (proximity detectors)
  - Concept: simplest type of non contact sensors to detect obstacles
  - Concepts or mechanism: emit an infrared pulse and detect the reflected signal
  - Returned signal is a function of distance and the reflective properties of the surface

Remarks: (1)To avoid ambient light interference, encode the emitted signal
(2)Intensity of IR is proportional to d^{-2}

IR sensors are inherently short range sensors. Typical maximum ranges are 50 to 100cm

---

III.2 Non-visual Sensors and Algorithms

- Laser rangefinders (laser radar or lidar)
  - Concept: same as for sonar, instead of short sound pulse, short pulse of light is emitted
  - Range: several hundred meters (maximum). Lower one: several tens of meters.
  - Accuracy: 50mm for single scan. 16mm for average of nine scans. Sometime for 180 degree

Methodologies: 1)Triangulation: the use of geometric relationship between the outgoing light beam the incoming ray and its position on film plane.
2)Time of flight: the measure of the time delay for ongoing light ray to hit a target and return.

$$d = \frac{1}{2}vt$$

3)Phase-based: based on the difference between the phase of the emitted and reflected signals

Problem: risk of eye dame to observer

III.2 Non-visual Sensors and Algorithms

- Radar (radio detection and ranging)
  - Concept: same as for laser rangefinder, instead of light. Radio pulse is emitted (GHz range)
  - Problem: too costly and too bulky

III.2 Non-visual Sensors and Algorithms

- GPS (satellite-based positioning) [Russia: GLONASS]
  - Concept: the system operates by allowing a user to measure the difference in time-of-flight for a radio signal to arrive from a combination of satellites
  - Measurement: latitude, longitude elevation, current time
  - SPS: standard positioning system: accuracy of 100m horizontally156m vertically
  - Civilian applications
    - PPS: precise positioning system: GPS uses a sophisticated model of signal error and other factors to produce a measurement
    - Advantage: passive system, no other observation necessary
    - Disadvantage: environment limited, not indoors, underground, or underwater, forest, between high buildings
    - DGPS: Differential GPS: accuracy 1m to 10m
    - DCGPS: Differential carrier GPS: accuracy 1mm
    - For mobile robot: a combination of GPS for coarse localization and an alternative sensor-based method for precise positioning within a local neighborhood.

III.2 Non-visual Sensors and Algorithms

- Compass sensors:
  - Concept: measure the horizontal component of the earth's natural magnetic field, just like ordinary hiking compass does (vertical component, bird)

$$vx^2 + vy^2 = r^2 \qquad \begin{aligned} vy &= r\cos\theta \\ vx &= r\sin\theta \end{aligned} \qquad \theta = \arctan\left(\frac{vx}{vy}\right)$$

Direction of magnetic north:

Indoor robot: distortion due to metal objects are unavoidable

---

III.2 Non-visual Sensors and Algorithms

- Shaft Encoders:
  - Concept: to measure rotation to perform path integration, they are devices mounted on the rotating shaft
- Absolute shaft encoders: a disc showing a gray code (a binary code in which only one bit change between successive words) is read by an optical sensor and indicates the shaft's position. It gives current position of the shaft, but are not well suited for integrated movement over time
- Incremental encoders: disc is read by two photo receptors, A and B. So both direction and angle turned can be obtained
  - Digress: To measure lateral movement. LVDT (Linear variable differential transform)

  - Integrate turns of the robot's wheel turns yield current position
  - This is called path integration or dead reckoning
  - Dead reckoning is very unreliable overall but short distance
  - Improvement: combine with compass, landmarks

III.2 Non-visual Sensors and Algorithms

- Accelerometers
  - Concept: spring-mounted masses whose displacement under acceleration can be measured
  - Newton's Law:

$$F = ma \qquad F = kx^2 \Rightarrow a = {kx^2}/{m}$$

  - By mounting three accelerometers orthogonally to one another, an omnidirectional acceleration sensing device can be fabricated

Gyroscopes:

Concept: measure angular acceleration by exploiting basic Newton mechanics

Problem: small measurement error accrue over time (drift)

III.2 Non-visual Sensors and Algorithms

- Sonar sensors (problem: can't be used in mars and moon?)
  - Sonar units are typically installed in mobile robots by mounting emitter-receiver
  - Units in fixed position on the base of the vehicle. Typical configuration is to mount the sonar emitter-receiver on a rotating platform and thus achieve omnidirectionality

$$\text{Robot} \xrightarrow{d} \text{object} \qquad d = \frac{1}{2}ct \qquad c = c_o + \frac{0.6T\text{m}}{\text{s}} \qquad c_o = \frac{331\text{m}}{\text{s}}$$

T is the temperature in degrees Celsius, c = speed of sound, t = time delay

III.2 Non-visual Sensors and Algorithms

- Sonar sensors

Method:
1) Send high voltage, high-frequency electrical signal through the transducer to produce a brief acoustic pulse.
2) Switch transducer to "listening" mode and act as microphone, because residual oscillation in the transducer immediately after the acoustic pulse is generated

Blanking interval 6cm (can't detect anything less than 6cm far away
Sonar range 40-50kHz

Problem of sonar sensors:
the sensitivity of a sonar transmitter/receiver is not uniform,
but consists of a main lobe and side lobes.

III.3 Visual Sensors and Algorithms

- CCD camera (standard charge-coupled device) obtain 640X480 at 30Hz
- (optical mice, 1500 times a second)

III.3 Visual Sensors and Algorithms

- Calibration
- One camera (pin-hole model): direction of object, no depth information

Approach: 1)Restrict possible distance value
2)Multiple camera
3)Temporal integration

III.3 Visual Sensors and Algorithms

$\tilde{p}$: projection matrix
Given an arbitrary point in homogenous coordinates $x = [x_1 \quad x_2 \quad x_3 \quad 1]^T$

Through projection matrix to project to 2-D pixel $({}^{x_1}/_{x_3} \quad {}^{x_2}/_{x_3})$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \tilde{p} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{bmatrix} \qquad \tilde{p}_{3,4} = 1 \quad \begin{bmatrix} X & X & X & X \\ X & X & X & X \\ X & x & x & 1 \end{bmatrix}$$

Extrinsic camera parameters: define the rigid portion of transformation
Intrinsic parameters: parameters depend upon the nature of the camera

$\tilde{p}$ can be decomposed into intrinsic and extrinsic parameters

III.3 Visual Sensors and Algorithms

Camera calibration: the process of obtaining $\tilde{p}$

For each known 3-D point $\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \end{bmatrix}$ we will use $\tilde{p}$

to transform it to camera coordinate system $\begin{bmatrix} x_m \\ x_m \\ x_m \end{bmatrix}$ use $\tilde{p}\begin{bmatrix} x_m \\ y_m \\ z_m \\ 1 \end{bmatrix} = \tilde{p}\begin{bmatrix} x_\omega \\ y_\omega \\ z_\omega \\ 1 \end{bmatrix}$

Then through perspective projection to get undistorted image coordinate

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} f\dfrac{x_m}{z_m} \\ f\dfrac{y_m}{z_m} \end{bmatrix}$$

III.3 Visual Sensors and Algorithms

so each pair $\begin{bmatrix} x_n \\ y_n \end{bmatrix}$ generates 2 linear equations

$\tilde{p}$ has 11 unknown variable, need

$5\frac{1}{2}$ pairs, or 6 points for calibration.
If more point available -> LMS method

III.3 Visual Sensors and Algorithms

➢Signal and sampling

➢Image features and their combination
1)Edge detection
2)Fourier
3)Image filter

Remarks: (1)Vision is a powerful sensor for mobile robot.

(1)It can be used to recover environmental structure or to localize the robot in its environment without emitting additional energy into the environment.

III.4 Introduction to Kalman Filtering

- The Discrete Kalman Filter

$x_{k+1} = \phi_k x_k + \omega_k$ $x_k = (n \times 1)$ Process state vector at time *tk*

$z_k = H_k x_k + v_k$ $\phi_k = (n \times n)$ State transition matrix

$\omega_k = (n \times 1)$ White sequence $E[\omega_k \omega_i^T] = \begin{cases} Qk & i=k \\ 0 & i \neq k \end{cases}$

$z_k = (m \times 1)$ Vector measurement at time t_k

$H_k = (m \times n)$ Measurement matrix

$v_k = (m \times 1)$ Measurement error $E[v_k v_i^T] = \begin{cases} R_k & i=k \\ 0 & i \neq k \end{cases}$

Remarks: $E[\omega_k \omega_i^T] = 0$ for all *k* and *i*

III.4 Introduction to Kalman Filtering

- Assume, we have initial estimate of the process at some point in time ***tk***, this prior (or a priori) estimate will be denoted as $\hat{x}_k^-$

$\hat{x}_k^-$: “hat” denotes estimate, and “super minus” is a reminder that this is our best estimate prior to assimilating the measurement t_k.

$e_k^- = x_k - \hat{x}_k^-$ and the associated error covariance matrix is

$$P_k^- = \mathbf{E}[e_k^- e_k^{-T}] = E\left[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T\right]$$

III.4 Introduction to Kalman Filtering

With $\hat{x}_k^-$, we seek to use z_k to improve prior estimation

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k \cdot \hat{x}_k^-)$$

$\hat{x}_k^-$ updated estimate, K_k (blending factor)

Use minimum mean-square error as the performance criterion to find *Kk*.

First form the expression for covariance matrix:

$$P_k^- = \mathbf{E}[e_k e_k^T] = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k^T)]$$

Next substitute $z_k = H_k x_k + v_k$ into $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k \cdot \hat{x}_k^-)$

So $\hat{x}_k = \hat{x}_k^- + K_k(H_k x_k + v_k - H_k \cdot \hat{x}_k^-)$

III.4 Introduction to Kalman Filtering

and $P_k = E\left[(x_k - [\hat{x}_{\bar{k}} + K_k(H_k x_k + v_k - H_k \cdot \hat{x}_{\bar{k}})])\,(x_k - [\hat{x}_{\bar{k}} + K_k(H_k x_k + v_k - H_k \cdot \hat{x}_{\bar{k}})]^T)\right]$

$$= E\left\{[(x_k - \hat{x}_{\bar{k}}) - K_k(H_k x_k + v_k - H_k \cdot \hat{x}_{\bar{k}})][(x_k - \hat{x}_{\bar{k}}^T) - K_k(H_k x_k + v_k - H_k \cdot \hat{x}_{\bar{k}})]^T\right\}$$

$$= E\left\{[(I - K_k H_k)(x_k - \hat{x}_{\bar{k}}) - K_k v_k][(I - K_k H_k)(x_k - \hat{x}_{\bar{k}}) - K_k v_k]^T\right\}$$

$$= (I - K_k H_k) E(x_k - \hat{x}_{\bar{k}})(x_k - \hat{x}_{\bar{k}})^T (I - K_k H_k)^T + K_k \mathbf{E}(v_k v_{\bar{k}}^T) K_k^T$$

$$= (I - K_k H_k) P_{\bar{k}} (I - K_k H_k)^T + R_k K_k^T$$

III.4 Introduction to Kalman Filtering

- Regroup:

$$P_k = P_{\bar{k}} - K_k H_k P_{\bar{k}} - P_{\bar{k}} H_k^T K_k^T + K_k (H_k P_{\bar{k}} H_k^T + R_k) K_k^T$$

Rewrite: $P = \bar{P} - KH\bar{P} - \bar{P}H^T K^T + K(H\bar{P}H^T + R)K^T$

First assume $H\bar{P}H^T + R$ to be symmetric and positive definite
→ It can be written in factored form

$$SS^T = H\bar{P}H^T + R$$

$$P = \bar{P} - KH\bar{P} - \bar{P}H^T K^T + KSS^T K^T$$
$$= \bar{P} - (KS - A)(KS - A)^T - AA^T$$

III.4 Introduction to Kalman Filtering

We require $KSA^T + AS^TK^T = KH\bar{P} - PH^TK^T$

where $A = \bar{P}H^T(S^T)^{-1}$

$P = \bar{P} - (KS - A)(KS - A)^T - AA^T$

Only term $(KS - A)(KS - A)^T$ involves *K*, it is the product of a matrix and its transpose, which ensure that all terms along that major Diagonal will be non negative, clearly to adjust *K* to minimize *P* is to make $(KS - A)(KS - A)^T = 0$

Thus we choose $KS = A \Rightarrow K = AS^{-1}$

Since $A = \bar{P}H^T(S^T)^{-1}$

$= \bar{P}H^T(S^T)^{-1} \cdot S^{-1}$

$= \bar{P}H^T(SS^T)^{-1}$ Where $SS^T = H\bar{P}H^T + R$

$= \bar{P}H^T(H\bar{P}H^T + R)^{-1}$

III.4 Introduction to Kalman Filtering

- So we have our final expression for the optimum ***K***

$K_k = P_{\bar{k}}H_k^T(H_kP_{\bar{k}}H_k^T + R_k)^{-1}$ (Kalman gain)

Recall $P_k = P_{\bar{k}} - K_kH_kP_{\bar{k}} - P_{\bar{k}}H_k^TK_k^T + K_k(H_kP_{\bar{k}}H_k^T + R_k)K_k^T$

Leads to $P_k = P_{\bar{k}} - P_{\bar{k}}H_k^T(H_kP_{\bar{k}}H_k^T + R_k)H_kP_{\bar{k}}$

$= (I - K_kH_k)P_{\bar{k}}$

$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(z_k - H_k\hat{x}_{\bar{k}})$

$K_k = P_{\bar{k}}H_k^T(H_kP_{\bar{k}}H_k^T + R_k)^{-1}$

$P_k = P_{\bar{k}} - K_kH_kP_{\bar{k}} - P_{\bar{k}}H_k^TK_k^T + K_k(H_kP_{\bar{k}}H_k^T + R_k)K_k^T = (I - K_kH_k)P_{\bar{k}}$

(1)Thus $\hat{x}_{\overline{k+1}} = \phi_k\hat{x}_k$

$e_{\overline{k+1}} = x_{k+1} - \hat{x}_{\overline{k+1}}$

$= (\phi_kx_k + \omega_k) - \phi_k\hat{x}_k$

$= \phi_k(x_k - \hat{x}_k) + \omega_k = \phi_ke_k + \omega_k$

$P_{\overline{k+1}} = \mathbf{E}(e_{\overline{k+1}}e_{\overline{k+1}}{}^T) = \mathbf{E}[(\phi_ke_k + \omega_k)(\phi_kx_k + \omega_k)^T]$

$= \phi_kP_k\phi_k^T + Q_k$

Part IV: Pose and Maps

Algorithms for Pose Maintenance

Map Building

IV. 1 Algorithms for Pose Maintenance

- When robot is navigating, it needs to know “where it is?”
- Localization, pose estimation, or positioning
    - Strong localization: estimate the robot’s location with respect to some global representation of space
    - Weak localization: knowing if current location has been visited before (using qualitative maps)
    - Global localization: infer the robot’s position without an a priori estimate of its location

IV. 1 Algorithms for Pose Maintenance

- Pose maintenance or local localization: refine an estimate of the robot's pose continually.
- Pose estimation problem is also known as the drop-off problem (ex. People is walking blind folded)
- Dead reckoning:
- keeping track of how much one moves by observing internal parameters without reference to the external world is known as dead reckoning
- Errors: electrical noise, quantization, digitization artifacts, wheel slip, gear backlash...

IV. 1 Algorithms for Pose Maintenance

- Key step in localization involves matching current observation to established map.

Matching

1)Data-data matching: directly matching the current raw data with predicted raw data

2)Data-model matching: matching the observed data to more abstract models stored in the map.

3)Model-model matching: matching models stored in the map to models generated from current observations.

IV. 1 Algorithms for Pose Maintenance

- Dead Reckoning:
- Proprioception: In the context of biological systems, this observation of internal parameters (for example, how many steps are taken) is referred to as proprioception.

Assume error free, mobile robot with velocity $v = \frac{dx}{dt}$

Position of the robot $x = \int_{t_0}^{t_f} f v dt = \int_{t_0}^{t_f} \frac{dx}{dt} \cdot dt$

($t0$ = start time, tf = stop time)

IV. 1 Algorithms for Pose Maintenance

- Landmark
- Landmark can be used to improve pose estimation
- Problems about landmark
  - Landmarks are unlabeled
  - Landmarks are hard to detect
  - Wrong or incorrect measurement
- Additional factors about landmark
  - Are the landmarks passive or active
  - What kind of sensor is used (vision, sonar…)
  - Properties of landmark

IV. 1 Algorithms for Pose Maintenance

- Landmark classes
  - Artificial and Natural
- Sensing modalities:
  - Video-based sensing (doors, walls)
  - Laser-transmission
  - Active-radio beacon
  - Sonar
- Triangulation
  - Refer to the solution of constraint equations relating the pose of an observer to the position of a set of landmarks

---

IV. 1 Algorithms for Pose Maintenance

Triangulation for two landmarks
Diagram skipped
Robot with unknown location *x* senses two landmarks P_1, P_2
d_1, d_2 can be measured for robot

$$x(x, y, \theta)$$

we can get

$$x = {(a^2 + d_1{}^2 + d_2{}^2)}/{2a}$$

$$y = \pm(d_1{}^2 - x^2)^{1/2}$$

IV. 1 Algorithms for Pose Maintenance

Triangulation with uncertainty: sensor measurements have associated uncertainty.

GDOP: Geometric Dilution of Precision (GDOP): variation in the output estimate *X* with variations in the input parameters S

$$GDOP = {}^{\Delta x}/_{\Delta S}$$

Jacobian *J* of the measurement equation $= \lim_{\Delta S \to 0} GDOP = \lim_{\Delta S \to 0} \frac{\Delta x}{\Delta S}$

IV. 1 Algorithms for Pose Maintenance

Example for GDOP

$$x = {}^{(a^2 + d_1{}^2 + d_2{}^2)}/_{2a}$$

$$y = \pm\left(d_1{}^2 - x^2\right)^{1/2}$$

$$J = \begin{bmatrix} \frac{\partial x}{\partial d_1} & \frac{\partial x}{\partial d_2} \\ \frac{\partial y}{\partial d_1} & \frac{\partial y}{\partial d_1} \end{bmatrix} = \begin{bmatrix} \frac{d_1}{a} & \frac{d_2}{a} \\ \frac{d_{1(a-x)}}{ay} & \frac{xd_2}{ay} \end{bmatrix}$$

IV. 1 Algorithms for Pose Maintenance

- Servo Control
- Object recognition is hard due to: prospection distortion, image noise, light and shadow changes, and specular reflection.
- Solution: Visual servoing (image-based servoing): using vision data to relate a robot’s pose to that of a landmark.
- Servoing:
  - Allow a robot to move to a specific target position using observed sensor measurements.

IV. 1 Algorithms for Pose Maintenance

- Servo Control Method
- I(qgoal) image is associated with qgoal position.
- Difference between I(qgoal) and I(qc) to move robot.
- qc: Current position

Assumption: The distance between the current robot location and the target position is monotonically related to the distance in sensor space between the target sensor reading and the current sensor measurement.

IV. 1 Algorithms for Pose Maintenance

- Problem related to sensor-based servo control:
  - The mapping from pose to signal is not usually convex over large regions
  - The feature space may not be robust or stable.
  - No quantitative position is produced, except at “home” location
  - Servoing can only be used to return to previously visited poses.

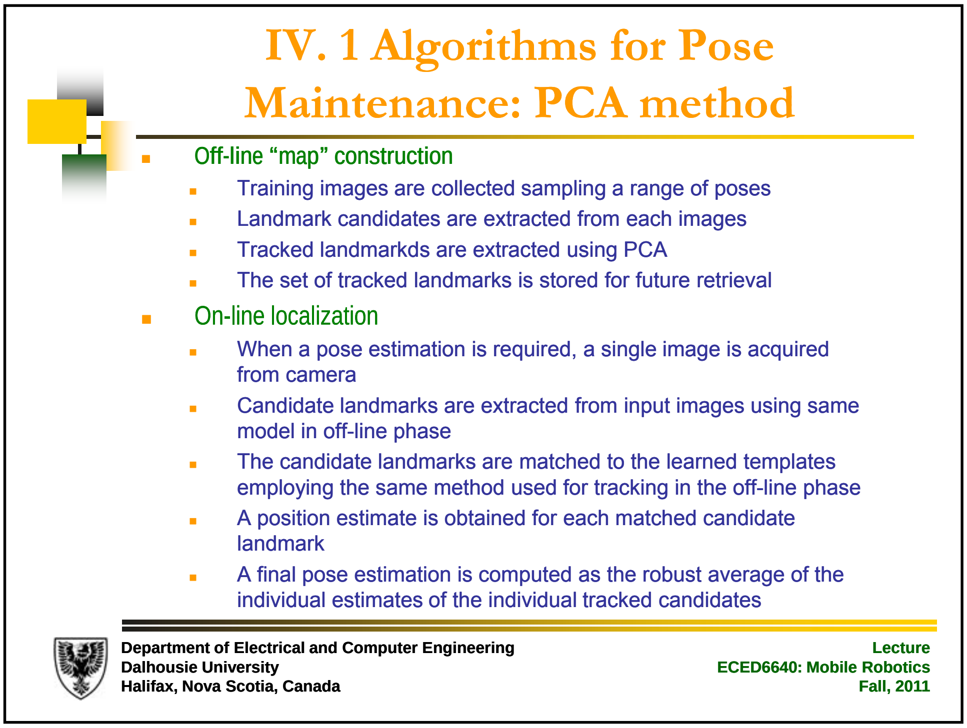
IV. 1 Algorithms for Pose Maintenance

- Nongeometric Methods: perceptual structure
- Eigen Landmarks
- Landmark learning is to train a system to recognize small sub images with recognizable characteristics (PCA methods)

PCA-based Landmark Extraction

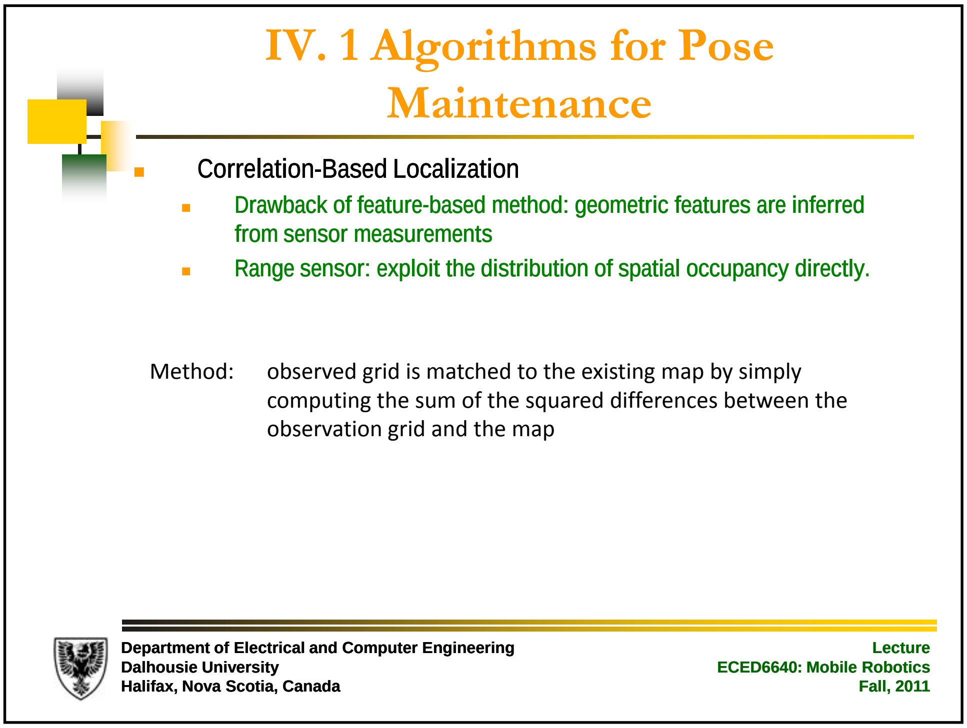
$\Rightarrow \lambda_1, \lambda_2, \dots, \lambda_n \rightarrow$ chose highest $\lambda_1, \lambda_2, \dots, \lambda_k \rightarrow v_1, v_2, \dots, v_k$

$v = [v_1 \quad v_2 \quad v_k]$ is the feature of training example



IV. 1 Algorithms for Pose Maintenance: PCA method

- Off-line “map” construction
  - Training images are collected sampling a range of poses
  - Landmark candidates are extracted from each images
  - Tracked landmarkds are extracted using PCA
  - The set of tracked landmarks is stored for future retrieval
- On-line localization
  - When a pose estimation is required, a single image is acquired from camera
  - Candidate landmarks are extracted from input images using same model in off-line phase
  - The candidate landmarks are matched to the learned templates employing the same method used for tracking in the off-line phase
  - A position estimate is obtained for each matched candidate landmark
  - A final pose estimation is computed as the robust average of the individual estimates of the individual tracked candidates



IV. 1 Algorithms for Pose Maintenance

- Correlation-Based Localization
  - Drawback of feature-based method: geometric features are inferred from sensor measurements
  - Range sensor: exploit the distribution of spatial occupancy directly.

Method: observed grid is matched to the existing map by simply computing the sum of the squared differences between the observation grid and the map

IV. 1 Algorithms for Pose Maintenance

- Global Localization
- Global localization is necessary when the robot is restarted, or its ongoing localization estimate process has failed.
- Exist solution
  - Localization from one or a few possible vantage points.
- Problem
  - If the environment is sufficiently self-similar.
- Improvement
  - Reduce this set of possible locations to a single one by moving about the environment.

IV. 2 Map Building

- Introduction
- Does the robot need a map?
- Two kinds of maps
  - Metric maps: based on absolute reference frame and numerical estimate of where objects are in space. (tourist scale map of city)
  - Topological maps: (relational maps) only explicitly represent connectivity information, typically in the form of a graph

IV. 2 Map Building

- Five layers of representation of map data
  - Sensorial: raw data signals
  - Geometric: two- or three-dimensional objects inferred from sensor data
  - Local relational: Functional, structural, or semantic relations between geometric objects that are near one another
  - Topological: large-scale relational links that connect object and locations across the environment as a whole
  - Semantic: functional labels associated with the constituents of the map

IV. 2 Map Building

- Sensorial maps
- Maps based on direct sensor readings offer the possibility of coupling the environmental representation as directly as possible.

Idea: couple measurements with odometry information→servo control

A) Image-based mapping
Assume perfect odometry: $I_t(x_t, y_t, \theta_t)$ can be collected continuously, then servo-control can be used

B) Spatial occupancy representations

IV. 2 Map Building

- Geometric Maps
  - Different geometric representation can result in very different exploration and search algorithms.
  - Map constructing: perfect odometry + triangulation
  - Limitation: It is hard to know if all space has been explored
- Spiral search: key technique is geometric exploration.

Example: search for a goal on a line. Assume goal is distance *d* from starting location move in one direction d if not goal, move in opposite direction 3*d*. Worst case scenario: 3*d* distance average 3d/2.

IV. 2 Map Building

- Spiral search: key technique is geometric exploration.

General: On trip *i,* the robot moves distance *f(i)* to the left and then return to the origin.

On trip *i+1*, the robot moves a distance *f(i+1)* to the right and then return to the origin (if it doesn't find the goal).

Robot must explore new territory on each trip. $\forall i \geq 1 \; f(i) > f(i-2) \; (f(0) = f(-1) = 0)$

Linear spiral search : $\forall i \geq 1 \; f(i) = 2^i$

Thus, given the goal is distance *n* from starting location,
the total distance traveled is a combination of a series of two-short trips of size 2^i , where $2^i < n$ followed by a final trip of length *n* that reaches the goal.

IV. 2 Map Building

- Spiral search: key technique is geometric exploration.

The total length is given by

$$2\sum_{i=1}^{\lceil \log n \rceil +1} 2^{i} + n$$

IV. 2 Map Building

- Topological Maps
  - To avoid the obvious difficulties in maintaining a long-term metric map of an environment, an alternative class of representation is based on graphs.

Graph-based map is given by $G = (V, E)$

with set of *N* vertices *V* and set of *M* edges *E*.
The vertices are denoted by $V = \{v_1, v_2, \dots v_n\}$

and the edge by $E = \{e_1, e_2, \dots e_n\}$

edge $e_{ij} = \{v_i, v_j\}$

IV. 2 Map Building

- Topological Maps
  - Signature: refer to specific observable characteristics associated with a location.
  - Problem: If signature is not unique, representation can't be constructed.
- Marker-Based Exploration
  - With ones signature _ are not sufficiently distinctive, the robot can disambiguating the node by using some mechanism (Marker)

IV. 2 Map Building

- Multiple Robots
  - Multiple robot can team up to do map building.
- Properties of multiple robot:
  - Improved robustness: a multiple robot system can function even if one robot fails.
  - Improved efficiency: it's possible to accomplish a search or exploration task more quickly then single robot
  - Alternative algorithms: catching an elusive object in a graph.

IV. 2 Map Building

- Multiple Robots
  - Issus of multiple robots:
    - Where are the other robots
    - Partitioning the work
    - Multi-robot planning
    - Data merging

IV. 2 Map Building

- Exploration with multiple robots:
  - Start point know sub graph Plan partition of work and rendezvous schedule
  - Each robot explores using the single-robot algorithm
  - At the predetermined time, the robots return to the agreed point in Harmonization begin
  - If unexplored edges remain in the single harmonized map, continue the algorithm starting with step 0

Part V: Representing and Reasoning

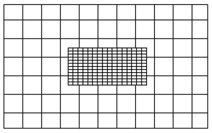
Algorithms for Space Representing
Algorithms for Robot Representing
Algorithms for Path Planning

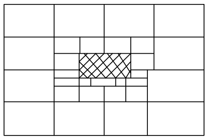
V.1 Algorithms for Space Representing

- A robot's internal representation of its space is typically required for at least three different classes of task:
  - To establish what parts of the environment are free for navigation
  - To recognize regions of locations in the environment
  - To recognize specific objects within the environment
- Spatial decomposition: to sample discretely the two -or higher- dimensional environment to be described.
  - Samples are binary: two dimension grids are called bitmaps
  - Otherwise: occupancy grids or pixel maps
  - In three dimensions: sampling elements are known as voxels or volume elements.
  - Advantage: general, no assumption made regard object type
  - Disadvantage: grid resolution is limited by grid size is empty or occupied and representation is storage intensive if much environment

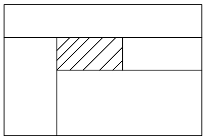
V.1 Algorithms for Space Representing

- A robot's internal representation of its space is typically required for at least three different classes of task:


Uniform

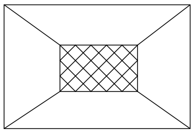

Quadtree

Recursive cells that are not neither uniformly empty or full are subdivided into four equal parts


BSP

Each new boundary divide a cell into two region

V.1 Algorithms for Space Representing


Exact

Free space is simply broken down into nonoverlaping region by planes such that the union of the parts is exactly the whole

Remarks:
1) Uniform & guadtree is unique solution
2) for BSP and Exact solution is not unique

V.1 Algorithms for Space Representing

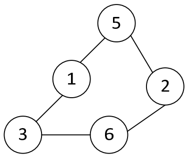
- Geometric Representations.
  - Geometric maps are made up of discrete geometric primitives: lines, polygons or polyhedral, points, polynomial functions and so forth.
- Remarks: primary shortcoming of geometric model-based representations relates to the fact that they can be difficult to infer reliably from sensor data.
  - Lack of stability: the representation may change in a drastic way given a small variation in the input
  - Lack of uniqueness: many different environments may map to same representation.
  - Lace of expressive power: it may be difficult to represent the salient features of the environment within the modeling system.

V.1 Algorithms for Space Representing

- To address instability (as well as uniqueness): introduce stabilizer
  - Discarding data points that don’t have neighbors
  - Preferring line segment models that are straight, aligned with preferred directions or parallel to other models
  - Preferring line segment models are long
  - Preferring models that have uniforms or low curvature
  - Preferring models with uniform data coverage

V.1 Algorithms for Space Representing

- Topological Representations
  - Topological representations: nonmetric
  - Explicit representations of connectivity between regions or objects. Vertices correspond to know landmarks and edges to the path between them.



V.2 Algorithms for Robot Representing

- Representing the robot
- Configurations space: key construction and formalism for motion planning.
- Configuration q of the robot A is a specification of the physical state of A with respect to a fixed environmental frame Fw.

Example: 2-d. rigid mobile robot $q = [x, y, \theta]$

Limbed robot: more complex.

V.2 Algorithms for Robot Representing

- Representing the robot

Physical construction of robot
Presence of obstacle

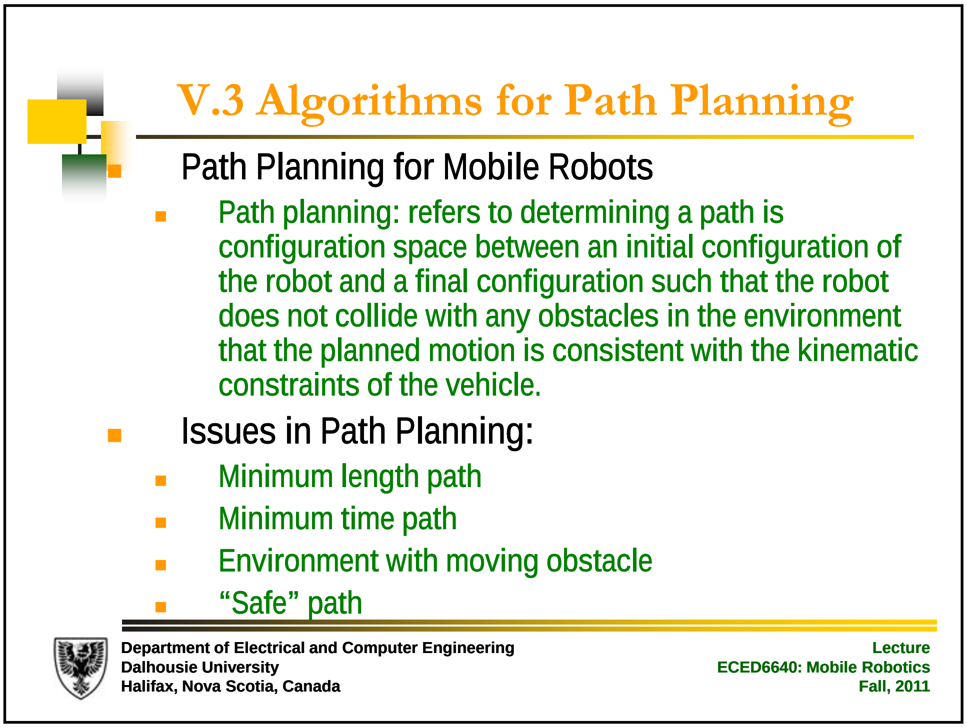
} Constraints of configuration

$G(q) = 0$: holonomic constraints

$G\left(q, \frac{dq}{dt}, \frac{d^2q}{dt^2}, \dots\right) = 0$: non holonomic constraints
very complex example: car parking

V.2 Algorithms for Robot Representing

- Simplification
  - Point robot assumption: robot can represent as a point
  - Problem: if nonholonomic and size has to be considered.

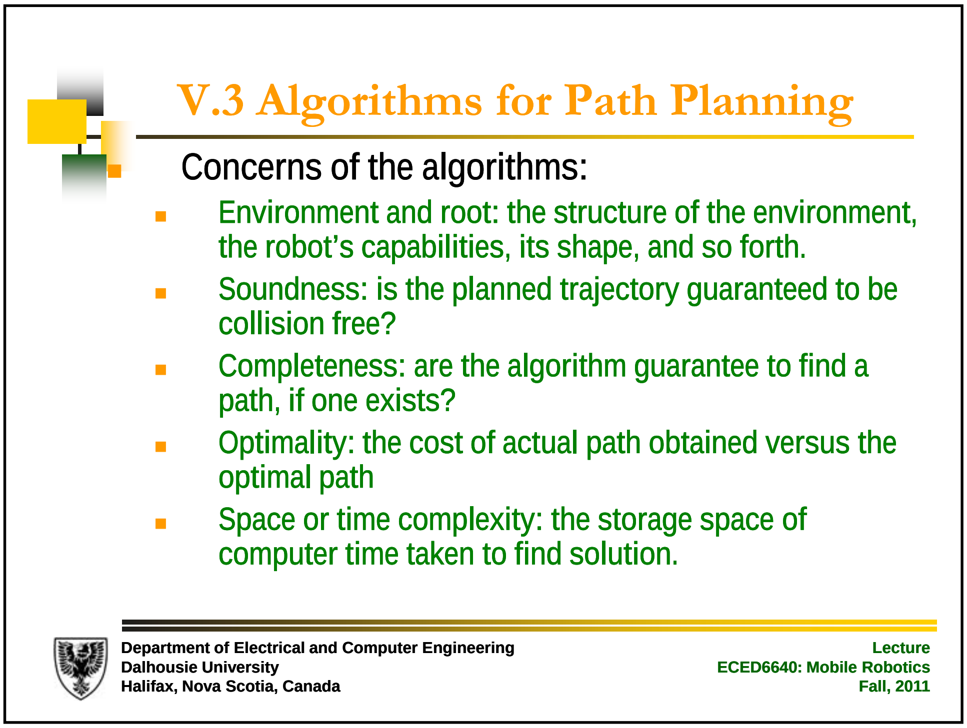


V.3 Algorithms for Path Planning

Path Planning for Mobile Robots

- Path planning: refers to determining a path is configuration space between an initial configuration of the robot and a final configuration such that the robot does not collide with any obstacles in the environment that the planned motion is consistent with the kinematic constraints of the vehicle.

- Issues in Path Planning:
  - Minimum length path
  - Minimum time path
  - Environment with moving obstacle
  - “Safe” path



V.3 Algorithms for Path Planning

Concerns of the algorithms:

- Environment and root: the structure of the environment, the robot’s capabilities, its shape, and so forth.
- Soundness: is the planned trajectory guaranteed to be collision free?
- Completeness: are the algorithm guarantee to find a path, if one exists?
- Optimality: the cost of actual path obtained versus the optimal path
- Space or time complexity: the storage space of computer time taken to find solution.

V.3 Algorithms for Path Planning

Searching a discrete state space: Graph search

```
Procedure graph search (s, goal)
        Open := {s}
        closed := { }
        found := false

    while (open!=0) and (not found) do
    select a node n from open
    open := open – {n}
    closed := closed v{n}
    if n E goal then
    found := true

        else begin
        let M be the set of all nodes directly accessible from n which are not in closed
        open := open U M
        end
        end while
```

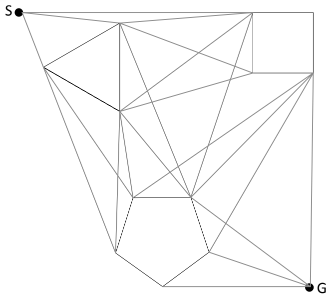
Remarks: search algorithm only determines if a path exists

V.3 Algorithms for Path Planning

Searching a discrete state space: Dynamic programming

  - Recursive procedure for evaluating the minimum cost path to any point in the environment from some source.
- Constructing a Discrete search space.
  - Straight forward approach: take a geometric representation of the free space and discretize it; using uniform, quadtree, BSP, exact method. Then construct the graph in which adjacent cells are connected.
  - Other mechanisms: visibility graph & Voronoi graph

V.3 Algorithms for Path Planning



- The set of vertices is made up of union of all of the verticals of the polygonal obstacles in the environment as well as the start and end points. (The edges of the graph connect all vertices that are visible to each other. The straight line connecting them does not intersect any obstacle.)

The task of search a graph for a path from one node to another is discussed in Graph Search.
Remark: path is very close to obstacle.

V.3 Algorithms for Path Planning

- Voronoi diagrams: to solve visibility graph problem
  - The generalized Voroni diagram is the locus of points equidistant from the closest two or more obstacle boundaries.
  - Property: maximizing the clearance between the points and obstacles.
  - Remarks: too conservative, long path length

V.3 Algorithms for Path Planning

- Searching a continuous state space
- Potential field:
  - Robot and obstacle are modeled as charged point so they are avoided by the repulsive force between them.

Attraction toward the goal is modeled by an additive field, which in the absence of obstacles, draws the charged robot towards the goal. Artificial potential field U(q) is constructed from components associated with the goal (Ugoal(q)) and any obstacles (Uobstacle(q)).

V.3 Algorithms for Path Planning

Potential field:

The net artificial potential filed

$$U(q) = U_{\mathrm{goal}}(q) + \sum U_{\mathrm{obstacle}}(q)$$

This can produce artificial force

$$F = -\nabla U(q) = -\begin{pmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{pmatrix} \rightarrow$$

$$U_{\mathrm{goal}}(q) = \mathrm{a\ dist}(q, \mathrm{goal})^2$$

drive the robot

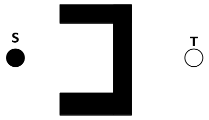
dist() is Euclidean distance between the state vector *q and goal state.*

$$U_{\mathrm{obstacle}}(q) = \beta\ \mathrm{dist}(q, \mathrm{obstacle})^{-1}$$

V.3 Algorithms for Path Planning

Potential field:

- Problem with potential field: local minima



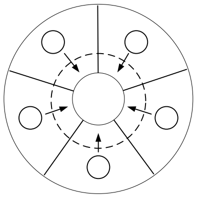
V.3 Algorithms for Path Planning

- Common approaches:
  - Backtracking when a local minimum is encountered.
  - Taking some random steps when local minimum is detected
  - Invoking a procedure planner, such as well follower when
  - Increasing potential when the robot visit a region to cause it to be repelled by previously visited regions.

V.3 Algorithms for Path Planning

Vector field histogram

- In VFF, detailed information concerning the local distribution of obstacle is lost.
- VfH: overcome VFF's problem by constructing a local polar representation of obstacle density.



Circular window of width w around robot is divided into angular sectors. Obstacle in each sector are combined to measure that sector's traverstability. Threshold is set.

V.3 Algorithms for Path Planning

The orientation of the obstacle is given by

Vector direction $\beta_{i,j} = \frac{\arctan(y_j - y_i)}{x_i - x_r}$

$(x_i, x_r) \rightarrow$ robot

$(y_j, y_i) \rightarrow$ obstacle

Vector magnitude $m_{i,j} = c_{i,j}^2 (a - b d_{i,j}^2)$

$c_{i,j}$ is the certainty of cell $c_{i,j}$ in the grid

$d_{i,j}$ is the distance of cell $c_{i,j}$ from robot

a and *b* are constant which satisfy $a - b\left(\frac{w-1}{2}\right) = 1$

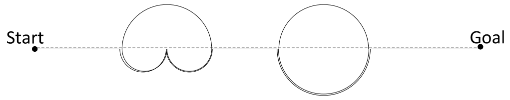
V.3 Algorithms for Path Planning

Bug algorithm

Two behaviors: 1) moving directly toward the goal location
2) circum-navigating an obstacle

Two improvements:

1) move to the target directly as soon as a straight-line path is feasible [distbug]
2) move towards a point on the line SG closer to G than the intersection point with current obstacles. [visbug]



V.3 Algorithms for Path Planning

- Spatial Uncertainty
  - Pose problem
- Complex Environment
  - Dynamic
  - Outdoor
  - Unknown
- Planning for Multiple Robot
  - Individual robot avoid to each other
  - Coordination for efficiency
  - Central planner + decentralize planner

Part VI: Control Algorithms of Mobile Robots

Introduction

Feedback and Close-loop Control

Control Law Partitioning and Tracking Control

Non Linear Control

Adaptive Control

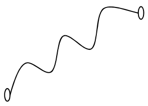
VI.1 Introduction

- Some mobile robot control method:
  - Neural network and fuzzy logic
  - Adaptive control
  - Classic PID control
  - Other control algorithms

VI.1 Introduction

Problem: Give desired trajectory $\theta d(t), \dot{\theta} d(t), \ddot{\theta} d(t)$

or $xd(t), \dot{x}d(t), \ddot{x}d(t)$, find the required joint



torques. $\tau(t)$ that will cause the desired motion.

$\textcircled{H}\ \dot{\textcircled{H}}\ \ddot{\textcircled{H}} \rightarrow \tau$

VI.1 Introduction

Mobile robot dynamics:

$$M(\theta)\ddot{\theta}(t) + V(\theta,\dot{\theta})\theta(t) + F(\theta,\dot{\theta}) = \tau$$

$M(\theta)$: inertia of the system

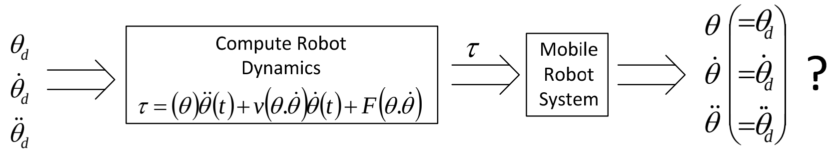
$V(\theta,\dot{\theta})$: related to the speed of robot

$F(\theta,\dot{\theta})$: disturbance

VI.2 Feedback and Close-loop Control

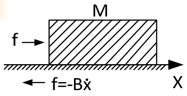
- Can we use dynamic equation to control the motion
- i.e. using

$$\tau = M(\theta)\ddot{\theta}(t) + V(\theta,\dot{\theta})\dot{\theta}(t) + F(\theta,\dot{\theta})$$



θ_d, $\dot{\theta}_d$, $\ddot{\theta}_d$ $\Longrightarrow$ Compute Robot Dynamics $\tau = (\theta)\ddot{\theta}(t) + v(\theta.\dot{\theta})\dot{\theta}(t) + F(\theta.\dot{\theta})$ $\xrightarrow{\tau}$ Mobile Robot System $\Longrightarrow$ $\theta\,(=\theta_d)$, $\dot{\theta}\,(=\dot{\theta}_d)$, $\ddot{\theta}\,(=\ddot{\theta}_d)$?

VI.2 Feedback and Close-loop Control



$$m\ddot{x}(t) + B\dot{x} = f \quad x(0) = 0 \quad \dot{x}(0) = 0$$

Assume $xd(t)$ is given, to cause a motion $x(t) = xd(t)$

We use control $f(t) = \hat{M}\ddot{x}d(t) + \hat{B}\dot{x}d(t)$

$\hat{M}$ and $\hat{B}$ are estimate model

Apply control $\underbrace{m\ddot{x} + B\dot{x}}_{\text{System}} = \underbrace{\hat{M}\ddot{x}d(t) + \hat{B}\dot{x}d(t)}_{\text{Model}}$

VI.2 Feedback and Close-loop Control

Define $e(t) = x(t) - xd(t)$ as tracking error

Then:

$$m(x - xd) + mxd + B(x - xd) + Bxd = \hat{M}xd + \hat{B}xd$$

$$\Rightarrow m\ddot{e} + B\dot{e} = (\hat{M} - M)\ddot{x}d + (\hat{B} - B)\dot{x}d$$

If $\hat{M} = M, \hat{B} = B$ (exact model)

Then $m\ddot{e} + B\dot{e} = 0$

If $e(0) = \dot{e}(0) = 0 \Rightarrow e(t) = 0$

But if $\hat{M} = M, \hat{B} - B = 0.1$ and assume $xd(t) = {}^{1}/_{2}\, t^2$

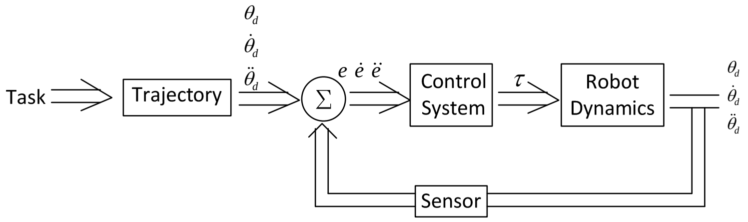
Then $\ddot{e} + \dot{e} = 0\ddot{x}d + 0.1\dot{x}d = 0.1t$

$\ddot{e}(t) + \dot{e}(t) = 0.1t, \quad e(0) = 0 \quad \dot{e}(0) = 0$

$$e(t) = 0.05t^2 - 0.1t + 0.1 - e^{0.1-t} \Rightarrow t \to \infty \quad e(\infty) \to \infty$$

VI.2 Feedback and Close-loop Control

- The key issue in designing a useful controller has been the use of feedback control



VI.3 Control Law Partitioning and Tracking Control

The mass-spring-friction system model:

$$m\ddot{x} + b\dot{x} + kx = f$$

The model based portion of the control law:

$$f = \alpha f' + \beta$$

With $\alpha = m$ and $\beta = b\dot{x} + kx$

$\Longrightarrow$ Error dynamics

$$m\ddot{x} + b\dot{x} + kx = \alpha f' + \beta = mf' + b\dot{x} + kx \Longrightarrow \ddot{x} = f'$$

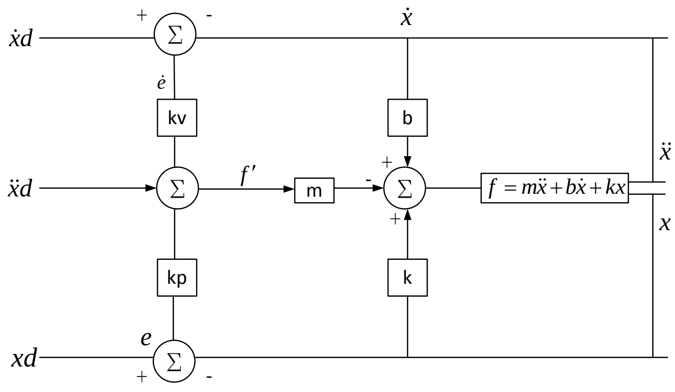
The servo portion of control law f' is chosen as

$$f' = \ddot{x}d + kv(\dot{x}d - \dot{x}) + kp(xd - x)$$

$$\ddot{x} = \ddot{x}d + kv(\dot{x}d - \dot{x}) + kp(xd - x)$$

$$e = xd - x \Longrightarrow \ddot{e} + kv\dot{e} + kpe = 0$$

VI.3 Control Law Partitioning and Tracking Control



VI.4 Nonlinear Control

- Robot Model $\tau = M(\theta)\ddot{\theta}(t) + V(\theta,\dot{\theta})\dot{\theta}(t) + F(\theta,\dot{\theta})$

Apply partition control law: $\tau = \alpha\tau' + \beta$

Where $\alpha = M(\theta)$ and $\beta = V(\theta,\dot{\theta})\theta(t) + F(\theta,\dot{\theta})$

Error dynamics:

$$\tau = M(\theta)\ddot{\theta}(t) + V(\theta,\dot{\theta})\dot{\theta}(t) + F(\theta,\dot{\theta}) = \alpha\tau' + \beta = M(\theta)\tau' + V(\theta,\dot{\theta})\dot{\theta}(t) + F(\theta,\dot{\theta})$$

$$\Rightarrow \ddot{\theta} = \tau'$$

With the servo law

$\tau' = \ddot{\theta}d + kv\dot{E} + kpE$ and $E = \theta d - \theta$

$$\ddot{\theta} = \ddot{\theta}d + kv\dot{E} + kpE$$

Error dynamics $\Rightarrow \ddot{E} + kv\dot{E} + kpE = 0$

VI.4 Nonlinear Control

- Joint PID control

Control law with $\alpha = I$, $\beta = \mathbf{0}$

$$\tau' = \ddot{\theta}d + kv\dot{E} + kpE + ki\int Edt$$

$$\Rightarrow \tau = \alpha\tau' + \beta = \tau' = \ddot{\theta}d + kv\dot{E} + kpE + ki\int Edt$$

$$M(\theta)\ddot{\theta}(t) + V(\theta,\dot{\theta})\dot{\theta}(t) + F(\theta,\dot{\theta}) = \ddot{\theta}d + kv\dot{E} + kpE + ki\int Edt$$

VI.4 Nonlinear Control

- Using Lyapunov Stability Analysis to prove the above system is stable
- Lyapunov Stability Analysis
  - It can be used to consider the stability without solving for solution of differential equations. While Lyapunov method is useful for examine stability, it does not provide any information about the transient response or performance of the system. It is an energy based method. It is one of the few techniques that can be applied directly to nonlinear systems to investigate their stability.

Lyapunov's method is concerned with determining the stability of a differential equation

$\dot{x} = f(t)$ where $x \in R^{m\times 1}$

$f(\cdot)$ could be linear or non linear

VI.4 Nonlinear Control

- Remarks: higher-order differential equations can always be written as a set of first-order differential equations.
  - Example $\ddot{x} + b\dot{x} + qx^3 = 0$

Let $x = \begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}x\\\dot{x}\end{bmatrix} \Rightarrow \dot{x} = \begin{bmatrix}\dot{x}\\\ddot{x}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -qx^2 & -b\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}$

$$\Rightarrow F(x) = \begin{bmatrix}0 & 1\\ -qx^2 & -b\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix} = \begin{bmatrix}x_2\\ -qx_1^3 - bx_2\end{bmatrix}$$

VI.4 Nonlinear Control

- To prove a system stable by Lyapunov method,
- one is required to propose a generalized energy function, $v(x)\epsilon R'$ that has the following properties:

$v(x)$ has continuous first partial derivative and $v(x) > 0$ for all x except $v(0) = 0$

$\dot{v}(x) \leq 0$ here $\dot{v}(x)$ means the change in $v(x)$ along all system trajectories.

VI.4 Nonlinear Control

- Example1 : Spring-damper system

$x + b(x) + kx = 0$ The functions $b(\cdot)$ and $k(\cdot)$ satisfy $xb(x) > 0$ for $x \neq 0$

$xkx > 0$ for $x \neq 0$

Find out the stability of the system

VI.4 Nonlinear Control

- Solution

Construct an energy function

$$v(x,\dot x) = {}^1/_2\dot x^2 + \int_0^x k(\lambda)d\lambda$$

$v(x,\dot x) > 0$ for all *x* except $v(0) = 0$

$$\dot v(x,\dot x) = \dot x\ddot x + k(x)\cdot\dot x$$

$$= \dot x[-b(\dot x) - k(x)] + k(x)\dot x$$

$$= -\dot x b(\dot x) - k(x)\dot x + k(x)\dot x$$

$$= -\dot x b(\dot x) \le 0 \ (\dot x \ne 0)$$

So the system is stable

VI.4 Nonlinear Control

- Example 2
  - Mobile robot control

Dynamics $\tau = M(\theta)\ddot\theta(t) + V(\theta,\dot\theta)\dot\theta(t) + F(\theta,\dot\theta)$

Control: $\tau = kpE + kd\dot E + F(\theta,\dot\theta)$

Define: $\theta \rightarrow \theta d$ with $\dot\theta d = \ddot\theta d = 0$

$$\Rightarrow \tau = kpE - kd\dot\theta + F(\theta,\dot\theta)$$

Error dynamics:

$$m(\theta)\ddot\theta + v(\theta,\dot\theta) + kd\dot\theta + kp\theta = kp\theta d$$

VI.4 Nonlinear Control

- Example 2

Contract Lyapunov function

$$v = \tfrac{1}{2}\dot{\theta}^T m(\theta)\dot{\theta} + \tfrac{1}{2}E^T kpE > 0 \quad (x \neq 0)$$

$$\dot{v} = \tfrac{1}{2}\ddot{\theta}^T m(\theta)\dot{\theta} + \tfrac{1}{2}\dot{\theta}^T \dot{m}(\theta)\dot{\theta} + \tfrac{1}{2}\dot{\theta}^T m(\theta)\ddot{\theta} + \tfrac{1}{2}\dot{E}^T kpE + \tfrac{1}{2}E^T kp\dot{E}$$

Since $X^T AY = Y^T AX$

$$\dot{v} = \dot{\theta}^T m(\theta)\ddot{\theta} + \tfrac{1}{2}\dot{\theta}^T \dot{m}(\theta)\dot{\theta} + E^T kp\dot{E}$$

$$= \tfrac{1}{2}\dot{\theta}^T \dot{m}(\theta)\dot{\theta} + \dot{\theta}^T m(\theta)\ddot{\theta} - E^T kp\dot{\theta}$$

We have $\tfrac{1}{2}\dot{\theta}^T \dot{m}(\theta)\dot{\theta} = \dot{\theta}^T v(\theta, \dot{\theta})$

$$\dot{v} = \dot{\theta}^T\left(v(\theta, \dot{\theta}) + m(\theta)\ddot{\theta} - kpE\right)$$

$$= -\dot{\theta}^T kd\dot{\theta} \leq 0$$

System stable

VI.4 Nonlinear Control

- Example 2

If $\dot{v} = 0$

$$-\dot{\theta}^T kd\dot{\theta} = 0 \Rightarrow \dot{\theta} = 0 \Rightarrow \ddot{\theta} = 0$$

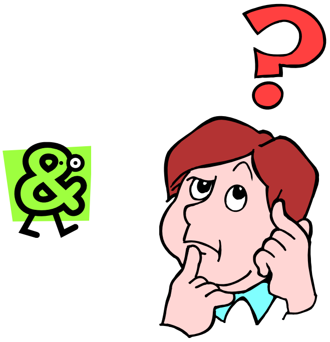
$$\Rightarrow kp\theta = kp\theta d \Rightarrow kpE = 0 \Rightarrow E = 0$$

$\Rightarrow$ globally asymptotically stable

VI.5 Adaptive Control

Research on Robotics

Questions and Comments



Jason.gu@dal.ca
http://myweb.dal.ca/jgu