My Lecture Notes

Overall Robot Navigation Paradigm

External World-> Sense -> Estimate -> Plan -> Act -> External World (Repeat)

- Sense+Estimate: estimate current and past pose
- Planning: Generate future robot pose
- Control: Stabilize robot pose

Main Questions to Solve

1. Where am I? (Sensing/Estimation)
2. Where am I going? (Planning)
3. How do I get there? (Control)

L2: Pose and Rotation

- Pose = position (translation) + orientation (rotation)
- Rigid Body: no deformation
  - Displacement:
    - Lengths preserved
    - Cross product presserved
  - Motion
    - include both translation and rotation
- Rotation
  - Right-handed coordinates
  - Can be represented with rotation matrices, Euler angles, Quaternions, etc.
    - Rotation matrix have no singularity and redundant parameters
    - Euler angles have gimble lock
    - Quaternion solves above problem

L3: Localization and Kinematics

- Solutions to "Where am I?" problem
  - external infrastructures (eg. GPS)
  - Odometry
    - Pose estimation only
    - Pros:
      - low computation complexity
      - high freq
    - Cons:
      - drift
      - no map
  - SLAM
    - Pose + local mapping

        * Pros:
            · builds local map
            · low drift
        * Cons:
            · higher complexity
            · map redundancy for long-term use
            · no global localization
    - Map-based Localization
        * Pose estimation
        * Pros
            · low drift with good map
            · lower complexity
            · good for long-term use
            · global localization
        * Cons
            · need good map
- Wheel Odometry
    - Locomotion: ability to transport/move itself place to place
        * issues: stability, contact, environment
    - wheels
        * standard wheel (2 dof (degree of freedom))
        * castor wheel (2 dof)
        * swedish wheel (3 dof)
        * ball wheel
    - estimate with kinematics (study of motion without regard to forces)

L4: Sensors

- Interoceptive: within body
    - Accelerometer
        * measures translational acceleration
    - Gyroscope
        * measures angular rate
    - IMU
        * measures linear acceleration and angular velocity
        * Pros
            · available and outlier free
            · high rate measurements
            · low cost
        * Cons
            · noisy
            · can't operate when inertial and visual measurements not in same frame (eg. on train)
- Exteroceptive Sensors
    - GNSS (satellite)
        * GPS

            - needs at least 4 satellites to estimate position
            - doesn't work well in denied areas (urban, forest, park, tunnel, etc)
    - Camera
        - Monocular
            - simplest
            - unknown depth
        - Uses perspective projection
            - closer object appear bigger
        - For solving depth
            - can use pictorial cues
            - stereo camera can compute depth
    - Lidars
        - Pros
            - wide fov
            - precise depth
            - illumination free
        - Cons
            - no RGB info
            - higher cost
            - sparse points per scan
    - Radar
        - Pros
            - weather robust
            - speed
        - Cons
            - noisy
            - less support
    - RGB-D Sensor
        - Lidar + Monocular
        - Pros
            - fast speed
            - relatively good depth
        - Cons
            - does not work outdoor
            - affected by illumination
- Multi-sensor Fusion
    - combines sensors
    - Pros
        - more robust
        - higher precision
    - Cons
        - high cost
        - time sync

L5: ICP

- 3D pose estimation with known correspondences
  - compute SVD
  - get rotation and translation, compute pose
- For unknown correspondenes, can use ICP
- ICP Idea: iteratively find alignment
- Algo:
  1. start with some guess
  2. find nearest neighbor
  3. refine rotation and translation based on latest data
  4. repeat step 2 until convergence
- Sampling/Filters
  - Can have uniform, random, feature-based sampling, etc.
- Weighting Correspondences
  - Noise: weight based on uncertainty
  - Outlier: lower weights for higher point-to-point distances
  - Determine transformation that minimizes weighted error function
- Data Association
  - closest point, point-2-plane, etc.

L6: Map

- Representations
  - Point Cloud
    - Pros
      - no discretization of data
      - map area not limited
    - Cons
      - unbounded memory usage
      - no direct representation of free or unknown space
  - Grid
    - Pros
      - volumetric representation
      - constant access time
      - friendly to planning
    - Cons
      - memory requirement
      - extent of map has to be known/guessed
  - Octree
    - tree-based data structure
    - Pros
      - full 3D model
      - memory efficient
    - Cons
      - tricky to implement

  - Elevation
    - 2D grid that stores estimated height for each cell
    - Pros
      - 2.5D representation
      - constant time access
    - Cons
      - no vertical objects
  - Features
    - Use features such as landmarks, objects, etc.
    - Pros
      - less memory loss
      - support localization
      - close to human eye perception
    - Cons
      - can't represent complex env
      - not planning oriented

L7: Bayes Filter

- Problem Overview:
  - Move (increases uncertainty)
  - Predict (use sensors, inputs, etc.)
  - Sense (get sensor measurements)
  - Update (decrease uncertainty)
- Bayes filter problem example:
  - Suppose robot obtains measurement z (eg. brightness)
  - What is P(door open|z)

L8: Particle Filters

- Particle filters are implementations of recursive Bayesian filtering
- Particles are propagated according to the motion model
- weighted according to likelihood of observations
- Pros
  - Easy to implement
  - able to handle nonlinear systems without linearization
  - able to represent arbitrary distribution
- Cons
  - particle degeneracy problem
  - need lots of particles to represent high dimensional state space, more computation complexity as well
- Applications
  - robot pose tracking, target tracking, etc.

L9: Kalman Filter and EKF

- Particle Degeneneracy Problem:
  - situation where a small number of particles in the filter carry a significant weight
  - eg. filter out 25% particles each time, after 4 operations particles originate from same source
- To solve problem, can use Kalman Filter (KF) and Extended Kalman Filter (EKF)
- Kalman Filter
  - weighted mean with Gaussians
  - Steps:
    1. Initialize system state
    2. Predict next step state
    3. Update with Kalman Gain
    4. Make correction
    5. Return to predict with new mean and variance
  - Everything stays Gaussian
  - variance never increases
  - prediction and update can happen in arbitrary order
  - Pros
    - simple
    - efficient
  - Cons
    - assumes linearity and Gaussian
- Kalman Gain
  - Intuition: How much to trust the sensor vs prediction
  - example: R=0 (perfect sensor)
    - gain = C^{-1}
    - variance is 0
  - example: R=∞
    - gain is 0
- EKF
  - Resolves non-linear functions by using local linearization
  - generates Gaussian approximation

L10: EKF SLAM

- Overall Goal: obtain both feature map and robot poses in real time
- SLAM Problem:
  - Given
    - control commands (odometry)
    - observations (measurements)
  - Want
    - Map of env
    - Path or current pose of robot

  - Difficulties
    - real world mapping between observations and landmarks unknown
    - picking wrong data associations can have large consequences
    - chicken-or-egg problem
      - a map is needed for localizatino
      - pose estimate needed for mapping
- EKF SLAM
  - Steps:
    - State prediction
      - state propagation with motion model
    - Measurement prediction: predict where map landmark should be if it appears
    - Obtained measurement: observe landmark with sensor
    - Difference: get difference between predicted and observed
    - Update step: update map and pose
  - Convergence results for linear case, diverge if nonlinearities large

L11: Place Recognition

- Loop Closing
  - recognizing already mapped area
  - uncertainties collapse
- Lidar Place Recognition
  - check if point cloud has been seen before
  - need retrieval scheme
    - eg. ring key
- Visual Place Recognition
  - check if image were taken around same location
  - difficult because challenges like information-rich, illumination changes, etc.

L12: Pose Graph SLAM

- Idea of Graph SLAM
  - use graph to represent problem
  - every node corresponds to pose during mapping
  - every edge is spatial constraint
  - build graph and find node config that minimizes error introduced by constraints

L13: Graph SLAM with Landmarks

- Rank of matrix H
  - 2D landmark-pose constraint
    - blocks Jij are 2x3 matrices

        * Hij cannot have more than rank 2
    - bearing-only constraint
        * Jij are 1x3 matrices
        * Hij has rank 1
- Questions
    - How many 2D landmark observations are needed to resolve robot pose?
        * Need min 2
    - How many bearing-only observations are needed to resolve robot pose?
        * Need min 3

L14: Visual Feature Detection

- Advantages of local features
    - locality
    - distinctiveness
    - quantity
    - efficiency
    - generality
- feature detection summary
    - compute gradient at each point in image
    - compute H matrix from gradient entries
    - compute eigenvalues
    - find points with large response
    - choose points where lambda is local maximum as features

L15: Visual Descriptor and Matching

- Transformational invariance
    - Want features to be same regardless of transformation
    - Need to have both of following
        1. Detector is invariant
        2. Design invariant feature descriptor
- Scale Invariant Feature Transform (SIFT)
    - basic idea
        * take 16x16 square window around feature
        * compute edge orientation for each pixel
        * throw out weak edges
        * create histogram of surviving edge orientations
    - properties
        * translation invariant
        * illumination invariant
        * rotation invariant

- Random Sample Concensus (RANSAC)
  - used to remove outliers from data
  - uses trial error approach to find best model

L16: Planning and Graph Construction

- motion planning (or piano mover's problem) finds optimal route to goal
- configuration
  - configuration is specification of every point on robot body
  - configuration space (c-space) is all possible robot configurations
  - use c-space instead of workspace for motion planning as robot is represented as point (easier for motion planning)
  - representing obstacle in c-space can be complicated, so use approximation
  - c-space is discretized for path planning
- two approaches to discretize c-spaces:
  - combinatorial planning
  - sampling-based planning
- combinatorial planning
  - visibility graph
    - construct path as polygonal line through vertices of obstacles
    - pros: simple
    - cons: path too close to obstacles (not safe)
  - voronoi diagram
    - take midpoint between obstacles to maximize distance between robot and obstacles
    - pros: safe, good clearance for uncertain robot
    - cons: complex calculation, not suitable for short-distance sensors
  - approximate cell decomposition
    - decompose into cells, construct connectivity graph, then search
    - pros: simple computations, simpler
    - cons: large storage space
- sampling based planning
  - probabilistic roadmap (PRM)
    - build graph to characterize free config space, use graph search to find path
    - pros: can cope with high dimensions, probabilistically complete (will contain a solution if it exists)
    - cons: collision detection takes majority of time, suboptimal if only limited samples given, need whole c-space as prior condition

L17: RTT, A-Star, Dijkastra

- rapidly exploring random trees (RTT)

  - aggressively probe and explore c-space by expanding incrementally from inital config
  - pros: balance between greedy search and exploration, easy to implement
  - cons: metric sensitivity, unknown rate of convergence
- Dijkastra's Algo
  - pros: complete and optimal
  - cons: can only see cost accumulated so far, no information about goal location
- A-Star
  - combines dijkstra and a heuristic