Basic Concepts of Robot Sensors, Actuators, Localization, Navigation, and Mapping

EL5223

Sensors and Actuators

- Robotic systems can utilize various types of sensors and actuators depending on the application.
- Some examples of sensors:
  - angle sensors (e.g., potentiometers, optical encoders) to measure joint angles
  - angular velocity sensors (e.g., tachometers on motors, gyros to measure three-axis angular velocities)
  - force and torque sensors
  - linear acceleration sensors (accelerometers)
  - distance/height sensors (e.g., ultrasonic range sensors)
  - sensors to sense objects in the environment, e.g., vision (electro-optic/infrared – EO/IR), LIDAR (laser range scanner), sonar, Radar, RGB-D (RGB color + depth)
- Some examples of actuators:
  - rotary and linear motors (e.g., DC motors)
  - hydraulic actuators, pneumatic actuators
  - piezoelectric actuators, shape memory alloys, etc.

Sensors and Actuators

- Sensor/actuator specifications/characteristics that are relevant when picking sensor and actuator components for a robotic application:
  - Range – max and min of the physical signal being measured (for a sensor) or being actuated (e.g., actuator force or torque); Field of View
  - Sampling rate (e.g., number of updates per second), latency/delay
  - Accuracy (e.g., how closely does a sensor measurement correspond to the actual physical quantity) and resolution (e.g., what is the smallest possible increment in sensor measurements). Note that accuracy and resolution are different; for example, a sensor might have very good resolution (very fine increments in sensor readings) but not very good accuracy.
  - Noise characteristics
- Gear train coupling: An actuator (e.g., a DC motor) is often coupled to a robot manipulator joint through a gear train. If the gear ratio is r, then the angular rotation θ_m of the motor and angular rotation θ_l of the link are related by $\theta_m = r\theta_l$. Also, if the angular velocities of the motor and the link are ω_m and ω_l, respectively, then $\omega_m = r\omega_l$. If the torques at the motor and the link are τ_m and τ_l, respectively, then $\tau_l = r\tau_m$. Hence, if we have a motor that spins faster than we need but provides lower torque than we need for a robotic application, then by picking r appropriately, we can convert the rotation rate to the desired range while also increasing the torque that we can apply on the robot's link.

Various Sources of Uncertainties in a Robotic System

- There are various possible sources of uncertainties in a robotic system that need to be kept in mind and, depending on the application, might need to be addressed in the path planning and control algorithms.
- Some examples of sources of uncertainties:
  - robot parameter uncertainties (e.g., link lengths, masses and inertias)
  - friction effects, gear train effects
  - sensory uncertainties/noise
  - actuator uncertainties/noise
  - delays/latencies in sensor measurements and actuator commands
  - environment uncertainties (e.g., properties of the objects that the robot is required to interact with, obstacle geometry of the environment, etc.)

Localization

- Localization is the problem of finding the robot's configuration (e.g., position and angular orientation for a mobile robot, joint angles/distances for a robotic manipulator, etc.) at each instant of time. If the robot geometry were perfectly known and actuators are perfect (no noise), then the robot configuration can be considered to be known if we know the commands sent to the actuators. However, in the presence of noise or uncertainties, then some real-time sensor measurements would be needed to find the robot configuration.
- Various sensors can be used for localization. For example,
  - GPS (Global Positioning System) for mobile robots in outdoor environments
  - Sensor measurements to known locations, e.g., RF (radio frequency) based triangulation based on a set of RF receivers/transmitters placed at known locations in the environment
  - Sensor measurements of motion relative to objects in the environment, e.g., using sensors such as LIDAR or vision
- Triangulation for localization: if distances or angles to multiple known locations (i.e., known 2D or 3D coordinates of positions in the environment) are measured, then the position of a moving point can be found using triangulation since each such measurement constrains the point to lie on a circle (for a distance measurement) or along a line (for an angle measurement) so that multiple measurements can together be used to find the complete position of the point. In general, with sufficient measurements, positions of multiple points on the robot can be found to thereby find the position and angular orientation of the robot (or each of its links).

Mapping

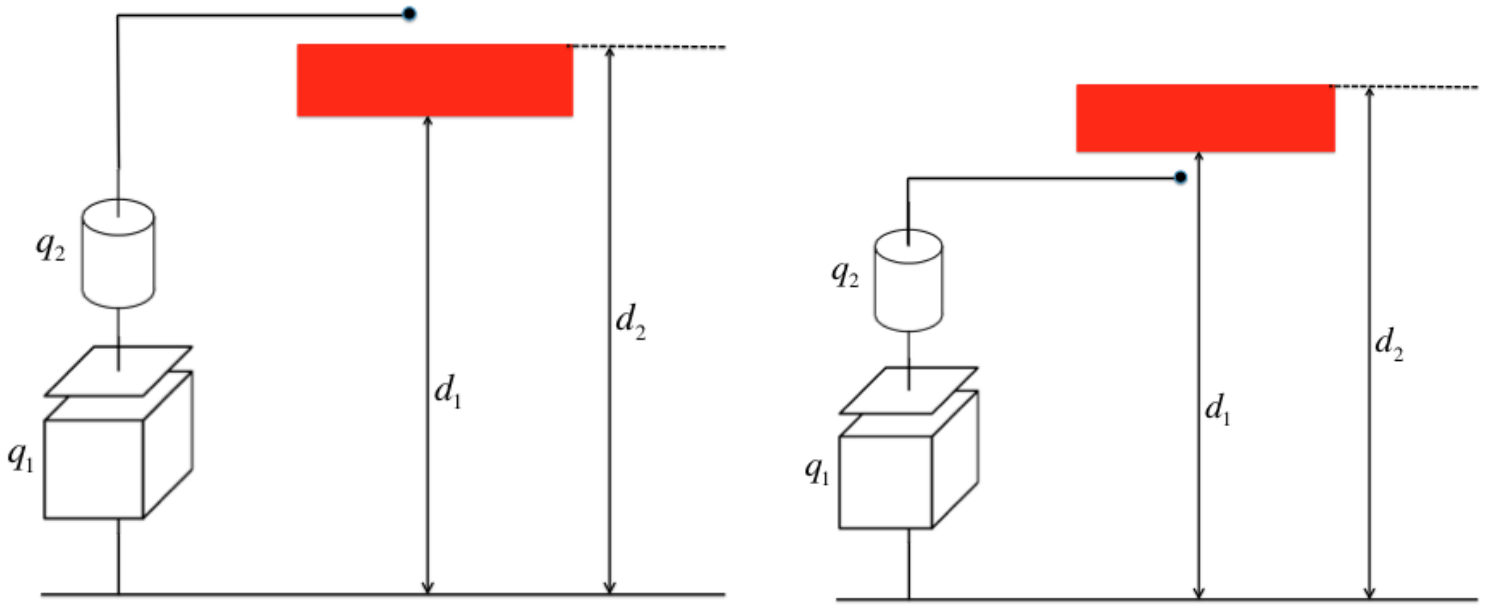
- Mapping is the problem of making a map of the environment using sensor measurements during the operation of the robot.
- The simplest type of map is a grid-based occupancy map in which the environment is represented as a grid (with some distance resolution) and each entry in the grid is set as 1 or 0 depending on whether an obstacle is detected at that location or not. Grid-based maps can be made in 2D (e.g., as a grid of (x, y) locations) or 3D (e.g., as a grid of (x, y, z) locations). For mobile robots operating over uneven terrain, the concept of a 2.5D map is useful wherein a 2D grid is used but each entry in the map is a height instead of 0 or 1 (i.e., the map stores the elevations of locations in the grid). The grid elements (i.e., cells of the grid) can be thought of as pixels (for a 2D map) or voxels (for a 3D map). More generally, each element of the grid can store not just a single value (e.g., 0 or 1), but a collection of properties of that location in the environment (e.g., a number between 0 and 1 to represent probability that there is an obstacle at that location, the color of the object at that location, etc.). To address noise/uncertainty in sensor readings and in the position and angular orientation of the robot, it is often better to keep obstacle probabilities in the grid instead of just 0 or 1, i.e., set a probability between 0 and 1 that there is an obstacle at that location (such a map is called a stochastic occupancy map) and then increase the probability each time an obstacle is seen at that location.
- If a point in the environment is seen using a sensor mounted on the end-effector of a robot, then the point representation needs to be converted to a world frame (e.g., the base frame or frame 0) to update the environment map. For example, if the homogeneous transformation matrix from the sensor-attached coordinate frame to the end-effector frame is H_s^e and the homogeneous transformation matrix from the end-effector frame to the base frame is H_e^0, then a point with coordinates p^s as seen in the sensor frame would correspond to coordinates $H_e^0 H_s^e p^s$ in the base frame.

Path Planning in Configuration Space

- Since a robot is not usually just a single point (but has physical extents comprising of all its links), specifying just the position of the end-effector is not sufficient to decide if a particular joint configuration is feasible given an obstacle geometry. However, specifying the joint configuration (i.e., all the joint variables) is indeed sufficient to decide if the configuration is feasible given an obstacle geometry. The set of all possible configurations is called the configuration space. For a robot manipulator with joint angles $q = [q_1, \ldots, q_n]^T$, the configuration space would be the set of all possible values of q (i.e., a subset of $\mathcal{R}^n$ taking into account the physically possible ranges of the joint variables). For a mobile robot characterized by position (x, y) and heading θ, the configuration space would be the set of all possible $[x, y, \theta]^T$ (i.e., a subset of $\mathcal{R}^3$). Once the configuration is specified, the locations of all points on the robot are known and hence it can be decided if there is any collision with an obstacle in the environment under that particular configuration.
- Hence, for path planning in an environment with obstacles, one general method is to represent the obstacles as an obstacle set in configuration space (i.e., as a set of configurations that are not feasible given the robot geometry and the obstacle geometry). Then, within the configuration space, the robot can be considered as a single point.
- A point in configuration space is considered as part of the obstacle set if for the particular robot configuration represented by that point, any part of the robot has a collision with an obstacle in the environment.

Path Planning in Configuration Space

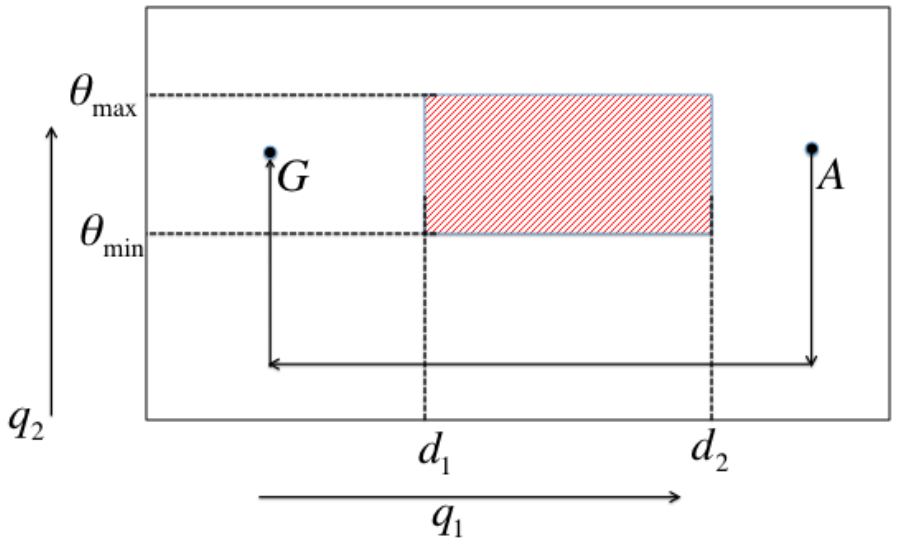
- For example, consider the prismatic-revolute manipulator shown below. The red object represents an obstacle in the environment.



- If the robot links are modeled as lines, then the entire set $\mathcal{S}$ of points occupied by the robot in three-dimensional space is the union of the set of points occupied by the first link and the set of points occupied by the second link:
  - the set of points occupied by the first link is the set $[0, 0, r_1]^T$ for all r_1 in the interval $[0, q_1]$.
  - the set of points occupied by the second link is the set $[r_2 \cos(q_2), r_2 \sin(q_2), q_1]^T$ for all r_2 in the interval $[0, l_2]$ where l_2 is the length of the second link.
- A configuration (q_1, q_2) is part of the obstacle set in configuration space if any of the points in the set $\mathcal{S}$ defined above is part of an obstacle in the environment.

Path Planning in Configuration Space

- For the obstacle shown in red in the figure on the previous range, if the angular range of the obstacle when seen from above is the interval $[\theta_{min}, \theta_{max}]$, then the obstacle set in configuration space is the set of all (q_1, q_2) such that $q_1 \in [d_1, d_2]$ and $q_2 \in [\theta_{min}, \theta_{max}]$. Hence, the obstacle set in configuration space is as shown below.



- If the robot is required to go from the configuration in the left-side figure on the previous page to the configuration in the right-side figure on the previous page, then the straight-line trajectory will definitely cause a collision with the obstacle. The initial and final configurations are shown as A and G in the figure above. One way to go from the initial configuration to the final configuration is the path shown in the figure wherein initially q_2 is reduced to clear the obstacle (i.e., rotating the second link so that the prismatic joint can be actuated downward without causing collisions), then q_1 is reduced, and then q_2 is moved back to the desired angular position.

Path Planning in Configuration Space

- Since the entire robot geometry is taken into account when forming the obstacle set in the configuration space, the robot can be considered to be a single point when planning a path in the configuration space, thus simplifying the path planning problem.
- There are many methods for path planning. For example,
  - Roadmap-based methods: Find a set of points that are in the obstacle-free part of the configuration space and then find pairs of points that can be joined by straight lines without intersecting the obstacle set. This gives a graph structure of the roadmap that can be used for path planning and graph search algorithms such as Dijkstra's algorithm, A^*, etc., can be applied to find a path. In general, various criteria such as distance in configuration space, proximity to obstacles, energy requirements, ranges of joint variables that provide more efficiency, etc., can be used to assign weights to edges in the graph.
  - Grid-based methods: The configuration space can be considered to be an n-dimensional grid within which a point robot is moving. Path planning within the grid is essentially a graph search problem and methods such as Dijkstra's algorithm, A^*, etc., can be applied.
  - Potential field methods: Consider the robot to be a particle with an electric charge (e.g., a positive charge) and consider the desired final configuration to be a particle with the unlike charge (i.e., negative charge) and consider the obstacles to have the same charge as the robot (i.e., a positive charge). Then, the robot is repelled from obstacles and attracted to the goal configuration.

Localization and Mapping

- Using triangulation, a set of distance or angle measurements to fixed locations in the environment can be utilized for localization. Using a motion model of the robot, a sequence of measurements over time can be used to improve the localization accuracy. For this purpose, a motion model is written in the form $\dot{x} = f(x, u, v); y = h(x, w)$ or as (in discrete time) $x_{n+1} = f(x_n, u_n, v_n); y_n = h(x_n, w_n)$. Here, x represents the motion state of the robot (e.g., position and angular orientation in a kinematic model; position, velocity, angular orientation, and angular velocity in a full dynamic model), u represents the input (e.g., velocity in a kinematic model; forces/torques in a dynamic model), v represents noise (or uncertainties), y represents measurements (e.g., position measurements, distance measurements to locations in the environment, etc.), and w represents sensor noise.
- For example, for a mobile robot, a kinematic motion model can be $\dot{p}_x = v\cos(\theta); \dot{p}_y = v\sin(\theta); \dot{\theta} = \omega$ where $[p_x, p_y]^T$ is the robot position, v is the speed, θ is the angular position, and ω is the angular turn rate. For a robotic manipulator, a dynamic motion model can be written using the dynamics equations $D(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \tau$ in terms of $x = [x_1^T, x_2^T]^T$ where $x_1 = q$ and $x_2 = \dot{q}$ as $\dot{x}_1 = x_2;\ \dot{x}_2 = -D^{-1}(x_1)C(x_1, x_2)x_2 - D^{-1}(x_1)g(x_1) + D^{-1}(x_1)\tau$.
- Depending on the type of sensor measurements, the measurement equations for y can be written. For example, if a sensor provides measurements of distance from the robot position p_r to a fixed location p_s in the environment, then the sensor measurement can be modeled as $||p_r - p_s||$.

Localization and Mapping

- A general method to estimate the state x given the motion model and measurement equations as described above is the Extended Kalman Filter (EKF). For example, given the motion model and measurement model $x_{n+1} = f(x_n, u_n, v_n); y_n = h(x_n, w_n)$, and a sequence of u_n and y_n over the time horizon $0 \leq n \leq N$, the EKF will provide an estimate of x_N, i.e., the state at time N.
- The problems of localization and mapping are coupled since localization assumes that the environment is (at least partially) known and tries to find the robot location whereas mapping assumes that the robot position is known and tries to map the environment. One way to address these problems together (i.e., Simultaneous Localization and Mapping or SLAM) is to consider an *augmented* state $\overline{x}$ which includes the robot state x and a feature state x_f (e.g., locations of points in the environment that are used for localization), write the motion model for the feature state as $\dot{x}_f = 0$, and use an EKF to find both x and x_f. If the environment is not known to begin with, then features in the environment would need to be found during robot operation and the detected features need to be matched between sensor measurements over successive time instants (i.e., feature extraction and feature association).