Mobile Robots

Wei Qi Yan

Auckland University of Technology

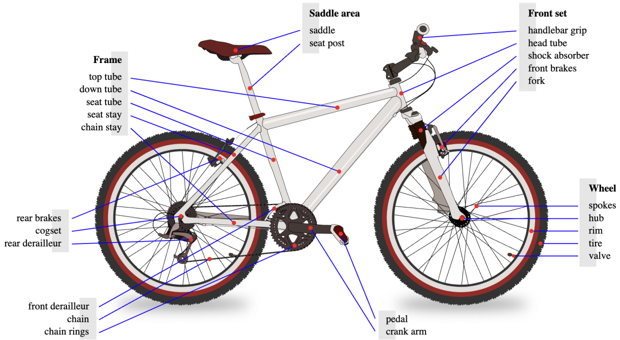
Table of Contents

Mobile Vehicles

- Mobile robots, a class of robots that are able to move through the given environment.
- The robot will take a path to reach its destination which faces challenges such as obstacles that might block its way or having an incomplete map, or no map at all.
- One strategy is to have very simple sensing of the world and to react to what is sensed.
- An alternative to create a map of its environment and then reasoned about the map to plan a path to its destination.
- These free-ranging mobile wheeled vehicles typically are use of the fixed infrastructure for guidance.

Mobile Vehicles

Bicycle model.

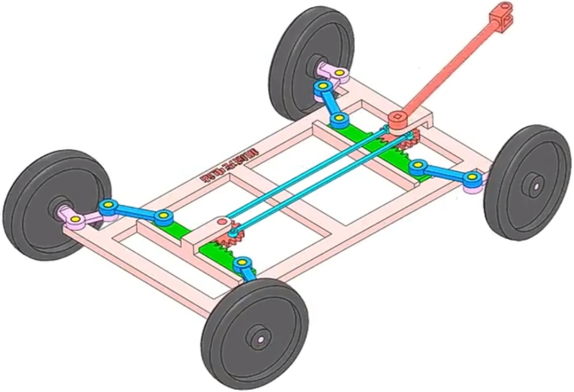


https://en.wikipedia.org/wiki/Bicycle

Mobile Robots

Mobile Vehicles

A simple proportional controller turns the steering wheel so as to drive the robot toward the target.



https://au.mathworks.com/help/robotics/ug/mobile-robot-kinematics-equations.html

Mobile Vehicles

- Following a line, two controllers are required to adjust steering.
- One controller steers the robot to minimize the robot's normal distance from the line: $ax + by + c = 0$.
- The second controller adjusts the heading angle, or orientation, of the vehicle to be parallel to the line $\theta^* = tan^{-1}(-\frac{b}{a})$.
- The combined control law turns the steering wheel so as to drive the robot toward the line and move along it.

Questions?

Mobile Robots

Questions?

In mobile robots, following a line, the two controllers are required to adjust steering.

1. The first controller steers the robot to minimize the robot's normal distance from the line. The second controller adjusts the heading angle, or orientation, of the vehicle to be parallel to the line.
2. Both controllers steer the robot to minimize the robot's normal distance from the line.
3. Both controllers adjust the heading angle, or orientation, of the vehicle to be parallel to the line.
4. None of the given options

The right answer is:___

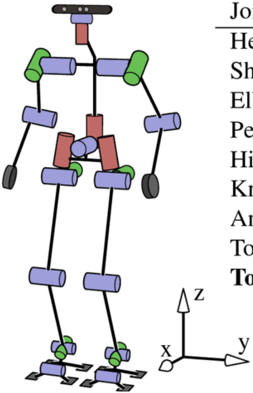
Questions?

Humanoid Robots

Introduction

- A humanoid robot is a robot resembling the human body in shape.
- In general, humanoid robots have a torso, a head, two arms, and two legs, though some humanoid robots may replicate only part of the body.
- Androids are humanoid robots built to aesthetically resemble humans.
- Sensors sense the position, orientation, and speed of the humanoid's body and joints, along with other internal values.
- Actuators are the motors responsible for motion in the robot.

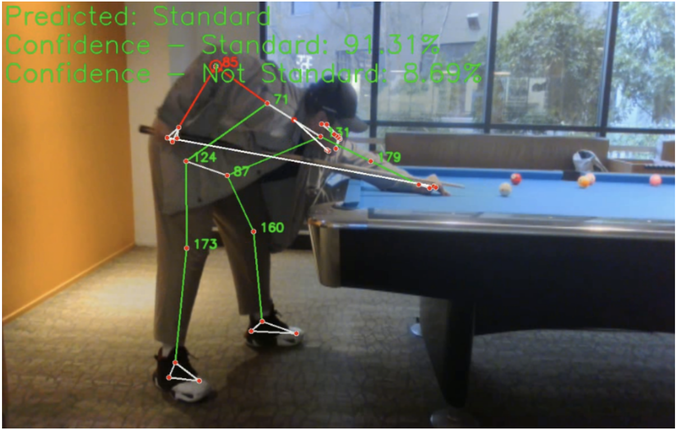
DoF



Joint	DoF
Head	2
Shoulder	2
Elbow	1
Pelvis	2
Hip	3
Knee	1
Ankle	2
Toe	1
Total	**24**

Hildebrandt, A. et al (2019) Kinematic optimization for bipedal robots: A framework for real-time collision avoidance. Autonomous Robots 43(2):1-19

Pool Player



https://mediapipe-studio.webapps.google.com/studio/demo/object_detector

Questions?

Humanoid robots

Questions?

Androids in robotics are:

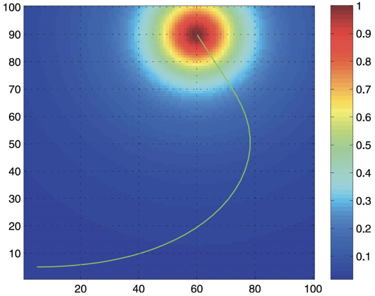
1. Humanoid robots built to aesthetically resemble humans
2. Mobile phones
3. Female robots
4. None of the given options.

The right answer is:___

Questions?

Reactive Navigation

- A robot is a goal-oriented machine that can sense, plan and act, like insects.
- A very simple class of goal achieving robots are known as Braitenberg vehicles which are characterised by the direct connection between sensors and motors.
- Braitenberg vehicle is moving toward (and past) the maximum of a 2D scalar field.



Simple Automata

- Multiple behaviours and ability to switch between them lead to an approach known as behaviour-based robotics or behavioral robotics.
- The behavior robots simply react to their environment and problems within that environment.
- Simple automata shares the ability to sense when they are in proximity to an obstacle.
- Simple automata includes a finite state machine and other logic in between the sensor and the motors.
- The simple automata performs goal seeking in the presence of non-driveable areas or obstacles.
- The automata have memory.

https://en.wikipedia.org/wiki/Behavior-based_robotics

Assumptions for Map-Based Planning

- The key to achieve the best path is to use a map.
- In a map, the driveable regions or obstacles are presented as polygons, each comprising lists of vertices or edges.
- This is potentially a very compact format but determining potential collisions between the robot and obstacles.
- A simpler and very computer-friendly representation is the occupancy grid, the memory is required to hold the occupancy.

Navigation

Assumptions for Map-Based Planning

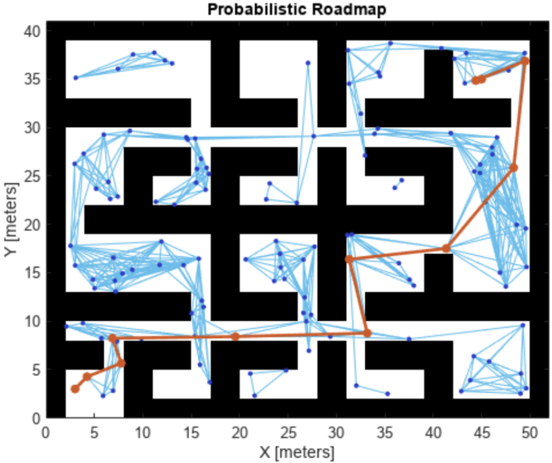
- The robot operates in a grid world and occupies one grid cell.
- The robot does not have any non-holonomic constraints and can move to any neighbouring grid cell.
- It is able to determine its position on the plane.
- The robot is able to use the map to compute the path it will take.

Assumptions for Map-Based Planning

- This navigation algorithm has exploited its global view of the world and has, through exhaustive computation, found the shortest possible path.
- The penalty for achieving the optimal path is computational cost.
- A fairly complex planning problem has been converted into one that can now be handled by using a Braitenberg-class robot that makes local decisions based on the distance to the goal.
- The roadmap methods provide an effective means to find paths in large maps at greatly reduced computational cost.

MATLAB Robotics System Toolbox

MATLAB: Probabilistic Roadmap



https://au.mathworks.com/help/robotics/ug/path-planning-in-environments-of-difference-complexity.html

D* Algorithm

D* algorithm also supports computationally cheap incremental replanning for small changes in the map.

- D* algorithm generalizes the occupancy grid to a cost map which represents the cost of traversing each cell in the horizontal or vertical direction.
- D* algorithm finds the path which minimizes the total cost of travel.
- D* algorithm supports incremental re-planning which has a lower computational cost than completely replanning.
- D* algorithm allows updates to the map to be made at any time while the robot is moving.
- After re-planning, the robot simply moves to the adjacent cell with the lowest cost which ensures continuity of motion even if the plan has been changed.

Voronoi Roadmap Method

In mathematics, a Voronoi diagram is a partition of a plane into regions close to each of a given set of objects. The steps in the creation of a Voronoi roadmap are:

- A free space is indicated by white cells.
- The skeleton of the free space is a network of adjacent cells no more than one cell thick.
- The skeleton with the obstacles overlaid in red and roadmap junction points indicated in blue.
- Regarding distance transform of the obstacles, pixel values correspond to distance to the nearest obstacle.

PRM: Probabilistic Roadmap Method

PRM reduces the computational burden by probabilistic sampling but at the expense of less optimal paths.

- The advantage of PRM is that relatively few points need to be tested to ascertain that the points and the paths between them are obstacle free.
- Each edge of the graph has an associated cost which is the distance between its two nodes.
- The color of the node indicates which component it belongs to and each component is assigned a unique color.
- Travel along the roadmap involves moving toward the neighbouring node which has the lowest cost, that is, closest to the goal.
- We repeat the process until we arrive at the node in the graph closest to the goal, and from there we move directly to the goal.

PRM: Probabilistic Roadmap Method

The important tradeoffs in achieving computational efficiency:

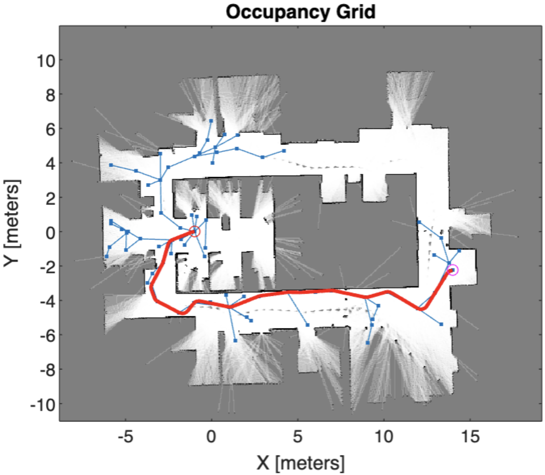
- The underlying random sampling of the free space means that a different graph is created every time the planner is run, resulting in different paths and path lengths.
- The planner can fail by creating a network consisting of disjoint components.
- The long narrow gaps between obstacles are unlikely to be exploited since the probability of randomly choosing points that lie along such gaps is very low.

RRT: Rapidly-Exploring Random Tree

RRT uses a kinematic model of the vehicle to create paths which are feasible to drive, and can readily account for the orientation of the vehicle as well as its position.

- An RRT computed for the bicycle model with a velocity, steering angle limits, integration period, and initial configuration.
- RRT algorithm is computing the control input that moves the robot from an existing point in the graph.
- This is repeated multiple times and the control input with the best performance is chosen.

MATLAB: Plan Mobile Robot Paths Using RRT



https://au.mathworks.com/help/nav/ug/plan-mobile-robot-paths-using-rrt.html

Navigation

Questions?

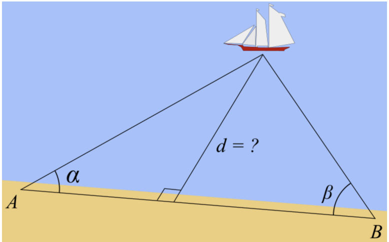
Dead Reckoning

Dead reckoning is the estimation of location based on estimated speed, direction and time of travel with respect to a previous estimate.

- GPS signals are also extremely weak and can be easily jammed.
- Location estimation by using dead reckoning is based on its position and the estimated distance travelled since, and the average compass heading.
- Resectioning is the estimation of position by measuring the bearing angles to known landmarks.
- Triangulation (Surveying) is the estimation of position by measuring the bearing angles to the unknown point from each of the landmarks.
- Computer vision can be employed to create a visual odometry system based on observations of the world moving past the robot.

Dead Reckoning

Triangulation (Surveying) is the estimation of position by measuring the bearing angles to the unknown point from each of the landmarks.



$$\because l = 1_1 + l_2, \tan(\alpha) = \frac{d}{l_1}, \tan(\beta) = \frac{d}{l_2},$$

$$\therefore l = d\left(\frac{\cos(\alpha)}{\sin(\alpha)} + \frac{\cos(\beta)}{\sin(\beta)}\right) = d\frac{\sin(\alpha+\beta)}{\sin(\alpha)\sin(\beta)}; d = l\frac{\sin(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}.$$

Localization

Dead Reckoning

Dead reckoning is the estimation of location based on estimated speed, direction and time of travel with respect to a previous estimate.

- Most robotic platforms have proprietary motion control systems that accept motion commands from the user (speed and direction) and report odometry information.
- An odometer is a sensor that measures distance travelled, typically by measuring the angular rotation of the wheels.
- The direction of travel can be measured by using an electronic compass, or the change in heading can be measured by using a gyroscope or differential odometry.

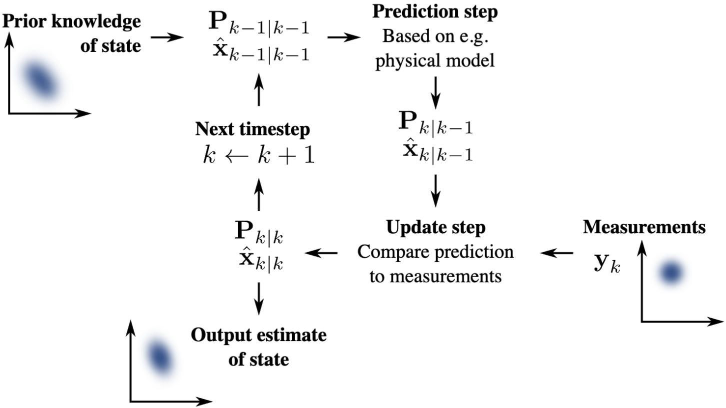
Kalman Filtering

- Kalman filter provides the optimal estimate of the system state and position *assuming that the noise is zero-mean.*
- Kalman filter is a recursive algorithm that updates, at each time step, the optimal estimate of the unknown true configuration and the uncertainty associated with that estimate based on the previous estimate and noisy measurement data.
- Kalman filter allows data from many varied sensors to update the state.
- Kalman filter provides the best estimate of where robots are.

https://au.mathworks.com/help/control/ref/ss.kalman.html

Localization

Kalman Filtering

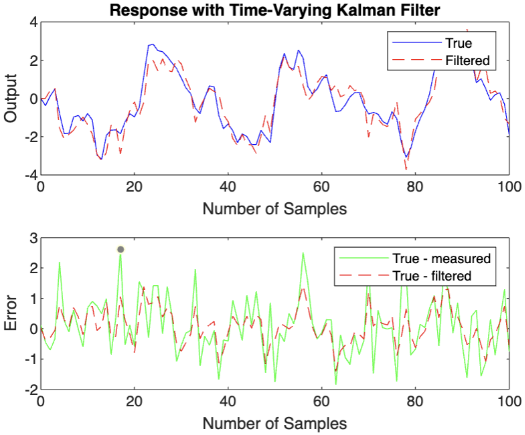


https://au.mathworks.com/help/control/ref/ss.kalman.html

Kalman Filtering

- We create a map of robot locations while it is moving in an environment with landmarks.
- The state vector comprises an estimated coordinates of the landmarks that have been observed as $\hat{\mathbf{x}}$.
- The corresponding estimated covariance is $\mathbf{P}$, $\mathbf{P}(k|k) = cov[\mathbf{x}(k) - \hat{\mathbf{x}}(k|k)]$ and $\mathbf{P}(k|k-1) = cov[\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1)]$.
- The prediction equation is $\hat{\mathbf{x}}(k+1|k) \leftarrow \hat{\mathbf{x}}(k|k)$, while the covariance matrix is $\mathbf{P}(k+1|k) \leftarrow \mathbf{P}(k|k)$.
- The updated state estimate is $\hat{\mathbf{x}}(k+1|k+1) \leftarrow \hat{\mathbf{x}}(k+1|k)$ and $\mathbf{P}(k+1|k+1) \leftarrow \mathbf{P}(k+1|k)$.
- Invariants of expectation: $\mathbf{E}[\mathbf{x}(k) - \hat{\mathbf{x}}(k|k)] = \mathbf{E}[\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1)] = 0$. That means all estimates have a mean error of zero.

MATLAB Kalman Filtering



https://au.mathworks.com/help/control/ref/ss.kalman.html

Questions?

Questions?

Like insects, a robot is a goal-oriented machine that can:

1. sense
2. plan
3. act
4. None of the given options.

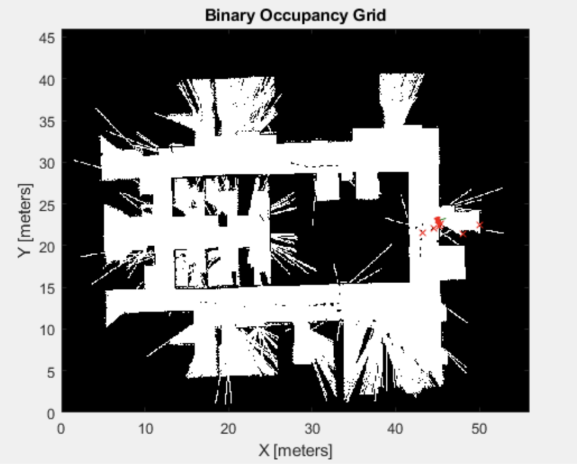
which answer is wrong?___

Questions?

MATLAB Automated Driving Toolbox

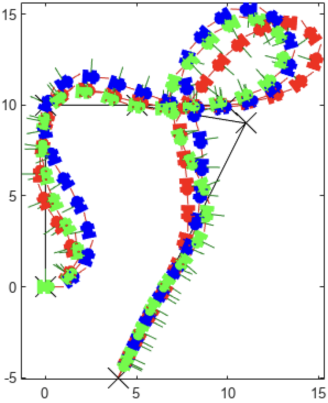
- Automated Driving Toolbox provides algorithms and tools for designing, simulating, and testing ADAS (Advanced Driver Assistance Systems).
- The Toolbox can be applied to design and test LiDAR perception systems, as well as sensor fusion, path planning, and vehicle controllers.
- Visualization tools include a bird's-eye-view plot and scope for sensor coverage, detections and tracks, and displays for video, LiDAR, and maps.
- Automated Driving Toolbox provides reference application examples for ADAS and automated driving features, including forward collision warning, autonomous emergency braking, adaptive cruise control, lane keeping assist, and parking valet.

MATLAB: Avoid Obstacles Using Reinforcement Learning



https://au.mathworks.com/help/robotics/ug/avoid-obstacles-using-reinforcement-learning-for-mobile-robots.html

MATLAB: Unicycle Kinematic Model



https://au.mathworks.com/help/robotics/ug/simulate-different-kinematic-models-for-mobile-robots.html

Questions?

Questions?

MATLAB Automated Driving Toolbox provides reference application examples for ADAS and automated driving features, including:

1. forward collision warning
2. autonomous emergency braking
3. lane keeping assist
4. none of the given options

which answer is wrong?___

Questions?

Learning Objectives

- Derive solutions for particular robotic vision and visual control tasks characterised by specifics of image data and deep learning algorithms.
- Demonstrate critical thinking in designing problem solutions for tasks in robotics.