IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2016

BEng Honours Degree in Computing Part III
BEng Honours Degree in Electronic and Information Engineering Part III
MEng Honours Degree in Electronic and Information Engineering Part III
BEng Honours Degree in Mathematics and Computer Science Part III
MEng Honours Degree in Mathematics and Computer Science Part III
MEng Honours Degrees in Computing Part III
MSc in Computing Science
MSc in Computing Science (Specialist)
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C333

ROBOTICS

Wednesday 23 March 2016, 14:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1 a A sonar sensor is mounted facing 90° to the left of the front of a robot to enable wall following. The aim is that the robot should control its steering to track a straight or gently curving wall at a safe distance $d = 30$cm. Write a short but clear program in Python pseudocode to control the robot using simple proportional servoing. The robot should maintain a constant forward speed 10cm/s as it makes symmetric left and right turns. Assume that the robot has a differential drive configuration where the linear velocities of the left and right wheels can be set reliably with functions `SetLeftVelocity()` and `SetRightVelocity()` respectively. These functions each take a floating point argument in cm/s units up to a maximum of 20cm/s. The sonar value in cm can be read with `GetSonarDepth()`. The system function `time.sleep()` is available, and takes an argument in seconds and should be used such that the control loop has a maximum update rate of 20Hz.

b This is the standard statement of the PID control law which can be used across a wide range of robotics and other control problems:

$$u(t) = k_p e(t) + k_i \int_{t_0}^{t} e(\tau) d\tau + k_d \frac{de(t)}{dt} .$$

Explain in general terms the role of each term in this expression, and briefly describe the effect of adjusting the values of each of the gain constants k_p, k_i and k_d.

c The wall following program is to be altered to replace proportional control with the use of PID control in the sonar to steering control loop. As before, the robot should retain a constant forward velocity.

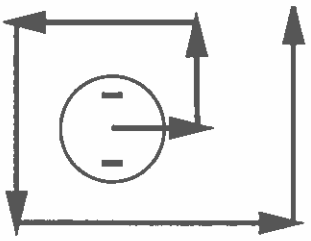
Write a new modified version of your program to achieve this. You should introduce variables in order to calculate the differential and integral errors from the sequence of measurements obtained by the sonar.

d Explain briefly why an omnidirectional ring of sonar sensors would be likely to be a better set-up for wall following than a single side-facing sonar. What would be a simple strategy for making use of the multiple measurements available?

The four parts carry, respectively, 30%, 30%, 30%, and 10% of the marks.

2a Using centimetre units for distances and radians for angles, a robot's motion can be programmed using position control with functions `DriveForward(D)`, (positive is forward), and `Rotate(alpha)` (positive is left). Write a function `NavigateToWaypoint(Wx, Wy, currentState)` in Python pseudocode such that the robot will turn and drive from any current location (x, y, θ) in the world coordinate frame to a waypoint at (W_x, W_y). Arguments `Wx` and `Wy` are floating point waypoint coordinates. Argument `currentState` is an object of class `State` which has data members `State.x`, `State.y` and `State.theta`, and these represent the current position and orientation of the robot. Your function should return an object `newState` with the robot's updated position and orientation after the movement. Assume here that the robot is very well calibrated and makes all commanded movements precisely. The angular orientation with which it reaches each waypoint does not matter.

b The robot is to be used as a floor cleaner, and is placed in the centre of a large room, where it defines a coordinate frame and zeroes its state to $(0, 0, 0)$. Write a program which defines a suitable sequence of waypoints and uses the `NavigateToWaypoint` function to move in a never-ending spiral cleaning pattern which starts as in the figure below. The robot has width `W` and should navigate such that it leaves no gaps in the floor surface covered.



c The robot runs fairly accurately on the hard floor surface where it was originally calibrated, but when placed (without any change to its program) on a thick carpet, it is found that it moves with systematic error such that it tends both to travel less than the commanded distance and to under-rotate on turns. Also it experiences significant zero-mean 'noise' errors in both distance and rotation.

Make a copy of the diagram of the start of the desired spiral trajectory, and on top sketch five example trajectories which the robot might execute on the carpet, having been placed precisely back at the start position and orientation at the beginning of each trial.

The three parts carry, respectively, 50%, 30%, and 20% of the marks.

3 A wheeled robot must localise within a cave with a flat floor. The cave is very large and the robot is far from any walls, but it has a low but uneven ceiling whose 2D height field has previously been precisely surveyed and is made available to the robot. The ceiling height above world coordinate (x, y) in integer centimetre units can be accessed at `heightMap[x][y]`. The robot has wheel odometry and a sonar sensor which is mounted in the centre of the robot, 10cm from the ground and pointing vertically upwards.

The robot is to perform Monte Carlo Localisation (MCL) as it moves, using a cloud of $N = 100$ weighted particles to represent a probabilistic estimate of its state. Each particle consists of a point estimate $\mathbf{x}_i$ of the parameters (x, y, θ) representing the robot's position and orientation, and a scalar weight w_i. Upon initialisation the robot is placed precisely at location $(x, y, \theta) = (100.0, 100.0, 0.0)$, with units of centimetres and radians.

Write clear and precise Python-like pseudocode for the following:

a Suitable global data structures for the storage of the particle set, and a function `InitialiseParticleSet()` to initialise the set to the correct values.

b A function `MotionPrediction(D, alpha)` which updates the particle set in response to a controlled motion of the robot either to drive forward through distance `D` or rotate on the spot through angle `alpha`. Either `D` or `alpha` will be zero on every call to this function. Make reasonable assumptions about the robot's motion uncertainty and include parameters which could be calibrated. You can use library function `random.gauss(mean, sigma)` to generate a random sample from a Gaussian distribution.

c A function `MeasurementUpdate(z)` which uses an unnormalised Gaussian likelihood function to alter the weights of all particles in response to a sonar measurement `z`. Python's library function `math.exp()` can be used for exponentiation.

d A function `void NormaliseParticleSet()` which normalises the particle set.

e A function `ResampleParticleSet()` which resamples the normalised particle set to replace it with particles which have equal weights but a spatial distribution representing the robot's position and orientation estimate. The library function `random.random()` returns a uniformly sampled random number in the range 0.0 to 1.0.

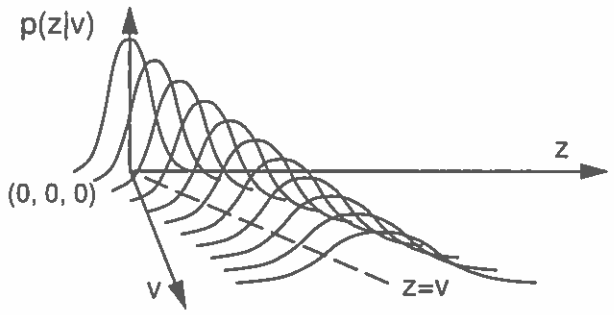
The five parts carry, respectively, 20%, 30%, 20%, 10%, and 20% of the marks.

4 a A new model of sensor, able like sonar to report the depth of the closest obstruction, is to be fitted to a mobile robot. To use this sensor in a probabilistic localisation algorithm, it must be calibrated to model its performance in detail.

i) Explain the meaning of the likelihood function $p(z|v)$ characterising the sensor, where z is a depth measurement and v is the ground truth distance.

ii) Describe the essentials of a calibration procedure to enable full determination of $p(z|v)$. Bear in mind that the accuracy of the sensor may depend on depth.

iii) Careful calibration reveals a likelihood function with shape illustrated by the following plot of a surface which has height $p(z|v)$ above the zv plane. The line $z = v$ is shown for reference.



Comment on the presence of systematic or zero-mean errors, the shape of their distribution and how the sensor's accuracy depends on depth.

iv) Sketch clearly a similar likelihood surface for another depth sensor which has constant uncertainty as a function of depth, but a systematic error meaning that it reports measurements on average 5cm greater than ground truth.

b Real sensors sometimes give 'garbage' measurements which are very far from ground truth. Sketch and explain a 'robust' likelihood function which characterizes a sonar sensor which provides bias-free measurements with a standard deviation of 2cm from ground truth (independent of depth) 90% of the time, but which in 10% of attempted measurements returns garbage values. The sensor has a minimum range of 10cm and a maximum range of 200cm, and the garbage values are uniformly distributed along this range. You need only sketch a 'slice' through the likelihood function to show a plot of $p(z|v)$ for a fixed ground truth value v =100cm.

c A robot is equipped with an array of depth sensors and is to map a previously unvisited area using a probabilistic occupancy grid. It has very accurate

odometry such that perfect localisation can be assumed. As it moves and makes measurements, the robot must repeatedly update the probability $P(O_i)$ that each cell i in its grid map is occupied. $P(E_i)$ is the corresponding probability that the cell is empty, where $P(O_i) + P(E_i) = 1$. Each time a new set of measurements Z is made, the robot has a given, pre-calibrated function which allows it to calculate for each cell the likelihoods $P(Z|O_i)$ and $P(Z|E_i)$ that those measurements would have been made given that the cell is really either occupied or empty. Show how the probability of occupancy of cell i should be updated after measurement Z, and that a 'log odds' representation of occupancy leads to a simple additive update.

The three parts carry, respectively, 50%, 20%, and 30% of the marks.