WRITE SOMETHING CLEVER

951926 1001231 1024072 1028907

***Abstract*—In this report, we present a system which performs the 1996 AAAI Mobile Robotics Competition task "Call a meeting"[1]. We provide some background information about the areas of robotics that are relevant to the problem, with brief reference to literature. We then provide a detailed description of our system and evaluate its performance. Finally, we discuss the experimental results and suggest areas for improvement.**

I. Background

Stuff about the task we are supposed to do goes here

A. Robot Motion

holonomic robots, PID control, reactive control

B. Localisation

- sensor model, update
- motion model

The aim of localisation is to obtain an estimate of the robot's pose at any given time. While it is possible to use odometry data to do so, this data is inherently noisy and prone to error. In particular, odometry errors can arise from changes in the surface being traversed and the robot's weight, among others. Many advanced localisation techniques are based on the Bayes filter, which uses a belief distribution $bel(x_t)$ to represent the state x_t. The calculation of $bel(x_t)$ at time t is dependent on $bel(x_{t-1})$ at time $t-1$, the last action u_t, and the last measurement z_t. In the first step, called the prediction step, the prior belief $\overline{bel}(x_t)$ is calculated. This step merges two probability distributions; the prior belief over the previous state x_{t-1}, and the probability of transitioning from that state to the posterior state x_t given that the action u_t was taken. This step does not take into account any measurement taken in the posterior state, predicting based solely on the knowledge of the action taken. In the second step, the measurement update, the posterior belief is calculated by multiplying the prior belief with the probability of being in the posterior state given that the measurement z_t was observed. The result of this multiplication is generally not a probability, and therefore requires normalisation using some constant α. As there are usually multiple posterior states, x_t is usually a state vector rather than a single state, and so the two steps will be applied multiple times in order to update the belief for each state being considered. The filter is recursive, requiring some idea of the initial belief $bel(x_0)$ at time $t = 0$. The initial belief is either a distribution centred on x_0, in the case where the initial position is known, or a uniform distribution over the space otherwise. As the Bayes filter is not restricted to finite state spaces, it is not possible to implement it for anything other than very simple problems. There are two families of algorithms for localisation, known as *recursive state estimators*, with various properties that permit the use of the Bayes filter in more complex estimation tasks.

Algorithm 1 Bayes filter[2]

1: **Algorithm Bayes_filter**$(bel(x_{t-1}), u_t, z_t)$
2: **for** all x_t **do**
3: $\quad \overline{bel}(x_t) = \int P(x_t \mid u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$
4: $\quad bel(x_t) = \alpha P(x_t \mid z_t) \overline{bel}(x_t)$
5: **end for**
6: **return** $bel(x_t)$

1) Gaussian Filters: The basic principle of the family of Gaussian filters is the use of multivariate normal distributions, which can be formulated from a mean μ and covariance Σ, to represent belief. As a result, the assumption that the system is a linear Gaussian system is made; the initial belief must be a Gaussian, and both the state transition function and measurement probability must be linear functions. Although Gaussian filters can be extended to non-linear systems, they perform best when the system meets the assumptions made. One of the main advantages of such filters is the computational complexity, which is polynomial with respect to the dimensionality of the state space. The main disadvantage is that Gaussians are unimodal and therefore cannot represent situations in which there are multiple hypotheses; a situation that is often encountered in robotics. Examples include the Kalman filter and the information filter, which are derived from two different ways of representing Gaussians. Both of these filters can be extended to non-linear systems by using a Taylor expansion to produce linear approximations of non-linear functions. Mixtures of Gaussians can also be used to extend the Kalman filter to encompass situations in which multiple hypotheses are required, but each extension increases the complexity of the algorithm. This extensibility is one of the reasons for the popularity of the Kalman filter in state estimation problems. With some extension, the information filter is particularly suited to multi-robot systems where information from multiple sources must be integrated, but issues with complexity have led the Kalman filter to become the more popular of the two for the majority of problems.

2) Nonparametric Filters: In contrast to Gaussian filters, nonparametric filters

Bayes filter, Kalman filter, particle filter (MCL), small section about mapping—still an active area of research in robotics. Mention SLAM, which has been pretty much solved.

C. Route Planning

PRM (sampling methods, graph search), RRT

D. Exploration

frontier based techniques

Algorithm 2 Basic Monte Carlo Localisation[2]

Algorithm MCL($\mathcal{X}_{t-1}, u_t, z_t, m$)
$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
for $m = 1$ to M **do**
  $x_t^{[m]} =$ **sample_motion_model**$(u_t, x_{t-1}^{[m]})$
  $w_t^{[m]} =$ **sensor_model**$(z_t, x_t^{[m]}, m)$
  $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
end for
for $m = 1$ to M **do**
  draw i with probability $\propto w_t^{[m]}$
  add $x_t^{[i]}$ to $\bar{\mathcal{X}}_t$
end for
return $\bar{\mathcal{X}}_t$

Algorithm 3 Probabilistic Road Map Generation

Algorithm generate_PRM(map)
$V =$ **sample_vertices**(map)
for $v_c \in V$ **do**
  while connections$(v_c) < C$ **do**
    $v_t =$ **get_closest**(V)
    if $d(v_c, v_t) < D_n$ **then**
      if $\neg\phi(v_c, v_t) \wedge \gamma(v_c, v_t)$ **then**
        connect(v_c, v_t)
      end if
    else
    end if
  end while
end for

E. Robot Vision

II. Design

A. System Structure

MENTION ALGORITHM COMPLEXITY! brief ROS description, callback based system, finite state automaton

B. Platform

Stuff about the pioneer—available sensors, some data about its size, specifications, our additions to it. Kinect specs. Include a picture of the robot with the kinect on it.

Algorithm 4 Path Flattening

Algorithm flatten_path(P, I, map)
for $i := 0$ **to** I **do**
  for $j := 0$ **to** $|P| - 2$ **do**
    $A \leftarrow$newpath(i)
    $B \leftarrow$newpath($i + 1$)
    $C \leftarrow$newpath($i + 2$)
    if freely_connected(map,A, C) **then**
      $P \setminus B$
    end if
  end for
end for
return newpath

III. Experimentation

A. PRM

inflated map - show inflated map superimposed onto the original map Redo experiment for sampling methods. short, medium, long path length. Display image of map with one of the routes displayed and show the difference between the sampling methods. Find the optimum route by sampling a massive number of vertices on to the space and then finding a route using that—the flattened path is then the most optimal route, and we compare the other routes to this route for each experiment.

Redid sampling experiments - now have comparison for neighbourhood, threshold and nearestN strategies on 4 different paths, including trying to get into the corridor. 5,10,20,30 maxconn for each strategy, inflation radius of 5 so that you can just about get into the corridor. random vertices:25,50,100,200,400,800, grid step 0.5,1,2,4, cell size 1,2,4,6, target per cell 2,5

REDO THIS ONCE THE ABOVE DATA IS COMPILED Best sampling: (approximate) optimum path found for each path by using grid sampling with a step of 0.2m and 50 max connections and neighbourhood connection strategy. Screenshots available in screenshots dir. Repeated experiments five times for cell and random, once only for grid. 5,10,20,30 maxconnections. Used results from previous experiments on the sampling strategy, but compared the best parameters for each. Also checked whether the corridor which causes issues was accessible when using each strategy as a measure of its ability to populate tight spaces. Not necessarily a good evaluation, since grid may be good on this map by accident, but terrible on others. new_prmlogs contains data.

B. Vision

C. Exploration

Coverage experiments info: cell and grid done for cell size and grid spacing of 10,8,6,4,2,1, with cell repeated three times for each. The fov max distance was 3.5, angle 57, minimum distance 0.3. Started at 2.80,18.97,40 in stage. Max move speed 0.7m/s, max rotation speed 0.4 rad/sec. Ran until the end of the exploration path was reached.

IV. Discussion

A. Performance

B. Potential Improvements

the path generated to do exploration could be improved by using heuristic techniques

C. Conclusions

References

[1] D. Kortenkamp, I. Nourbakhsh, and D. Hinkle, "The 1996 aaai mobile robot competition and exhibition," in *AI Magazine*, vol. 18, 1997.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents Series, Mit Press, 2005.