

Discrete Event Control of Nonholonomic Mobile Robots

David J. Austin
Department of Engineering
Australian National University
Canberra, Australia
david@faceng.anu.edu.au
Fax: int + 61 2 6249 0506

Brenan J. McCarragher
Department of Engineering
Australian National University
Canberra, Australia
brenan@faceng.anu.edu.au
Fax: int + 61 2 6249 0506

ABSTRACT

A new, complete discrete event framework for path planning and control of mobile robots is presented. This framework permits path planning using a map of the environment as well as reactive approaches and considers the dynamics of the robot. Discrete event modelling is the natural paradigm for systems where motion is constrained by the presence of obstacles. As the mobile robot encounters obstacles the system dynamics change dramatically. These changes are represented by discrete events.

Most existing mobile robot control techniques neglect dynamic considerations. However, dynamic considerations are important for high speed operation. For example, all robots have a deceleration limit which must be considered when approaching an obstacle. This paper uses a complete discrete event model which considers both the kinematics and dynamics of the mobile robot when determining which obstacles presently constrain the motion of the robot.

Experimental results demonstrating the application of the discrete event model for path planning and control of a nonholonomic mobile robot in indoor office navigation tasks are presented.

INTRODUCTION

Mobile robotics has long been an area of interest with the goal of expanding the robotics industry from the limited manipulators that are presently used in factories. Two central problems are of interest in mobile robotics: the development of a model of the environment and the navigation between goals using the environmental model. Both of these problems can be effectively addressed using a discrete event framework. This paper develops a discrete event framework for path planning and control of mobile robot systems. In addition, we will consider the dynamics of the robot.

There has been a great deal of research into various techniques for mobile robot path planning and con-

trol. Recently, reactive or behaviours-based approaches inspired by Brooks [2] have been the focus of much research. For example, Ward and Zelinsky [14] propose a simple and effective method for learning velocity responses to sensory stimulus. Whilst reactive approaches possess excellent obstacle avoidance behaviours, they have difficulty moving towards a desired goal. The problem of integrating goal-seeking behaviour with obstacle avoidance behaviours appears to be very hard [12] and has yet to be addressed satisfactorily. Alternative approaches to mobile robot navigation rely upon a map of the environment and plan a path using the map which is then executed. However, such methods have poor obstacle avoidance characteristics. For example, Luo and Chen [11] proposed a map-based method which simply stops and waits when an unexpected obstacle is encountered.

Discrete event methods have been considered for mobile robots previously, but only by a few researchers. Kosecka *et al.* [8, 9, 7] considered discrete event modelling of navigation of mobile robots with various tasks defined as discrete states (e.g. `moving`, `steer_away`, `path_following`). Similarly, Feddema *et al.* [4] use the discrete states `Search`, `Rotate`, `Backup` and `Track` for the line following problem. However, these models are rather arbitrary without a mathematical basis for the state definitions. Instead, we propose a constraint-based method for defining the discrete states. As the mobile robot moves, it becomes constrained by the obstacles within the environment. We define the discrete state depending upon the constraints which are currently active.

In addition to path planning and obstacle avoidance abilities, it is desirable for a mobile robot controller to account for the dynamics of the robot to permit high speed movement without collision. Fox *et al.* [5] proposed a reactive control method which considers the robot dynamics whilst Alvarez *et al.* [1] include dynamics in a map-based path planner.

This paper proposes a framework for mobile robot control which allows for map-based control, reactive control and also considers the robot dynamics. In addition, the discrete event framework models the changes in dynamics

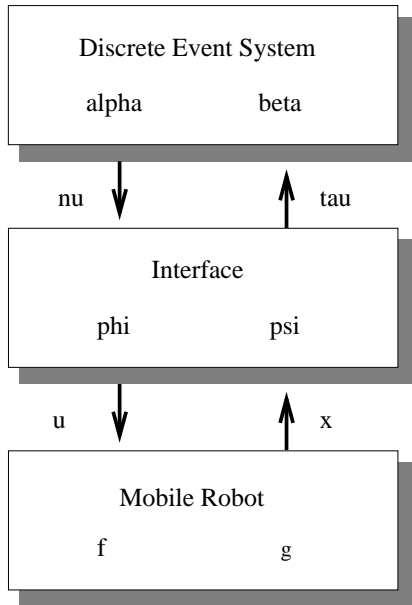


Figure 1: *Block Diagram Representation of System*

that occur as the robot becomes constrained by obstacles. In addition, the discrete event definition has a rigorous, mathematical basis.

DISCRETE EVENT MODELLING

Discrete event modelling is an ideal tool for mobile robotics as the gain and loss of constraints due to obstacles are fundamentally discrete events. Furthermore, discrete event models can be readily extended to deal with dynamic obstacles and reactive behaviours.

The structure of the adopted discrete event model is as shown in Figure 1. The system consists of three parts, which are the mobile robot, the discrete event controller, and the interface. Mathematically, the mobile robot is modelled by a set of differential equations describing the motion of the robot. The discrete event controller is modelled as an automaton describing task-level decision making, whereas the interface serves as the communication between the mobile robot and the decision maker.

The Mobile Robot System

Consider the general motion control of a mobile robot. The equation of motion of the robot in free space is given generally as:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \quad (1)$$

where $\mathbf{x}(t)$ is the continuous time state vector, and $\mathbf{u}(t)$ is the input vector. For the mobile robot, the state vector

is defined as:

$$\mathbf{x}(t) = [x \quad y \quad \theta \quad \dot{x} \quad \dot{y} \quad \dot{\theta}]^T \quad (2)$$

where x , y and θ are the position and orientation of the robot in some inertial reference frame.

The mobile robot that we will consider is nonholonomic and so we cannot control all of the state variables independently. The control of the robot allows the specification of a tangential velocity v_t and a turning speed or angular velocity ω . The ideal equations of motion for the robot are:

$$\dot{x} = v_t \cos \theta \quad (3)$$

$$\dot{y} = v_t \sin \theta \quad (4)$$

$$\dot{\theta} = \omega \quad (5)$$

Of course, the real world is not ideal so that many factors perturb the motion. Examples of disturbances that are difficult to detect are uneven terrain and wheel slip.

As the robot interacts with the environment, the free-space dynamics become constrained by the environment. For each obstacle in the environment, the constraint may be written as:

$$g_i(\mathbf{x}(t)) = 0 \quad (6)$$

where g_i is the constraint equation for the i^{th} obstacle. When the robot is constrained by multiple obstacles, the corresponding constraint equations are active. Note that the constraint depends upon the complete continuous state vector of the robot, including the velocity components. This dependence allows for the consideration of dynamic effects.

For example, the constraint imposed by a wall involves the velocity and the maximum deceleration of the robot as well as the position of the robot relative to the wall. This constraint may be written as:

$$v = \sqrt{2a_{max}\Delta x} \quad (7)$$

where v is the velocity of the robot towards the wall, a_{max} is the maximum deceleration of the robot and Δx is the distance between the robot and the wall. Figure 2 shows an example of a wall constraint, in which the constraint function is plotted. For velocities below the curve, the motion of the robot is not restricted by the wall whilst, for points above the curve, the robot will collide with the wall.

The Discrete Event System

The mobile robot is controlled by a task-level, discrete event controller modelled as an automaton. The automaton is a quintuple, (S, E, C, α, β) , where S is the finite set of states, E is the set of plant events, C is the set of controller events, $\alpha : S \times E \rightarrow S$ is the state transition function, and $\beta : S \rightarrow C$ is the output function. The

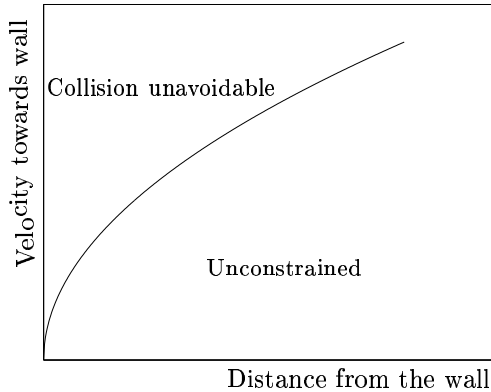


Figure 2: *Constraint as a function of distance and approach velocity. Points above the curve will result in collision, points on the line are constrained by the presence of the wall and points below the line are unconstrained.*

discrete state is a function of the constraint of the robot by obstacles in the environment. Each discrete state in S , denoted γ_k , represents the constraint of the robot by a different set of obstacles. Each *controller event* in C , denoted v_k , is generated by the discrete event controller, whereas each *plant event* in E , denoted τ_k , is generated by the activation of a constraint upon the robot. Plant events are generated when the mobile robot encounters an obstacle.

The dynamics of the discrete event controller are given generally by

$$\gamma_{k+1} = \alpha(\gamma_k, \tau_k) \quad (8)$$

$$v_k = \beta(\gamma_k) \quad (9)$$

where $\gamma_k \in S$, $\tau_k \in E$ and $v_k \in C$. The input and output signals of the discrete event controller are asynchronous sequences of events, rather than continuous time signals. The function α is functionally dependent on f and g as defined by the mobile robot system. Note that the state of the mobile robot system is \mathbf{x} , whereas γ is the state variable of the discrete event system and is dependent on \mathbf{x} .

The Interface

The interface is responsible for the communication between the plant and the controller, since these components cannot communicate directly due to the different types of signals being used. The interface consists of two maps ϕ and ψ .

The first map ϕ converts each controller event into a plant input. The input to the plant consists of a trajectory for the mobile robot to follow.

$$\mathbf{u}(t) = \phi(v_k, \mathbf{x}) \quad t_k \leq t < t_{k+1} \quad (10)$$

Note that the trajectory supplied to the robot is dependent upon the continuous state of the robot as well as the discrete controller event.

The second map ψ converts the state space of the plant into the set of plant events.

$$\tau_k = \psi(\mathbf{x}(t)) \quad (11)$$

Note that equation (11) does not imply that τ_k changes continuously as $\mathbf{x}(t)$ changes. The state space of the plant is partitioned into contiguous regions. The function ψ generates a new plant event only when the state first enters one of these regions. The map ψ is called a *process monitor*.

Example of Framework Operation

Example and picture here

Include how unknown obstacles and reactive behaviours can be implemented.

APPLICATION: OFFICE NAVIGATION

Trajectory Generation

There are two types of trajectory used by the robot to navigate to the goal. The first is a discrete state trajectory and the second type is the continuous trajectories used within each discrete state. We will now discuss the generation of both the discrete state and continuous trajectories.

The discrete state trajectory establishes the next desired discrete state. This is used by the discrete event controller to determine the controller event according to equation (9). Generation of the discrete state trajectory is fairly simple because there are a limited number of discrete states. In addition, the small number of discrete states permits automatic generation of the discrete state trajectory using a search technique. However, for this paper the discrete state trajectory is developed by the programmer.

For this paper, we will consider only constraints resulting from static obstacles within an office environment. More complex obstacles and moving obstructions can also be modelled and will be considered in future works. We will consider the constraints imposed by the office walls and also the supporting pillars within the office. In addition, we will consider intersections between walls (or corners in the office) as constraints. Corners can be modelled as a special case of the pillar constraint as corners are effectively a zero-radius pillar. Thus, there is a discrete state for each wall, pillar, and corner.

In response to each controller event, the interface supplies a continuous trajectory to the robot according to equation (10). When the robot becomes constrained by an obstacle, there are two possible behaviours. The first



Figure 3: *Nomad 200 robot used for experiments.*

is to move around the obstacle maintaining the constraint and the second is to back away avoiding the constraint. Maintaining the constraint is most useful as it provides wall-following trajectories. The method used for trajectory generation depends upon the constraint on the robot. There are four different types of constraint situation: unconstrained, wall constraint, pillar constraint and corner constraint. We will now consider trajectory generation for each of these.

Environment Map

The environmental map ψ (see equation (11)) is used to determine the discrete state of the robot as it interacts with the environment. Map making is a very complex issue and has been the subject of much research for a considerable period [3, 10, 13, 6]. However, the focus of this paper is discrete event control, so we have adopted a rather simplistic mapping strategy.

For this paper, the discrete state γ is determined solely from the continuous state information of the robot according to equation (11). We do not use any of the sensors to detect obstacles. Rather, the sonar sensors are used to correct for drift in the dead-reckoning position obtained from the wheel encoders. This is a simplistic method but is sufficient to demonstrate the discrete event control technique.

Experiments

The discrete event modelling technique was applied to experimental navigation tasks within an office environment. The experiments were conducted using the Nomad 200 robot shown in Figure 3. The Nomad 200 robot is subject to a nonholonomic constraint as discussed above. A map of the office environment was supplied to the robot and the position of the robot was obtained from the wheel encoders with sonar measurements used to correct for encoder errors.

Figure 4 shows the path taken through the office to achieve the goal. It can be seen that navigation within the office environment is highly constrained by obstacles and that the constraints change frequently. The discrete event framework provides a complete model for the discrete changes in constraint.

Figure 5 shows the discrete state of the robot as it performed the task, the breaks in the trajectory indicate where the discrete state of the robot changes. The discrete state of the robot changes whenever the robot encounters a new constraint. Figure 5 shows the effect of the dynamic constraints. For example, in the first state transition, from state 0 to state 1 it can be seen that the velocity of the robot towards the wall causes a transition into state 1 earlier than might otherwise be expected.

All of the discrete state transitions in the path are fairly clear with the possible exception of the transition from state 3 to state 0, shown in Figure 5 by the arrow. Here the robot has been instructed to cross the corridor by following the discrete state trajectory $\{...2, 3, 0, 4...\}$. To cross the corridor, the robot must move from constrained state 3 to the unconstrained state 0. To move into the unconstrained state the robot does not maintain the constraint in state 3 but finds a control command to reach free space, as was discussed in Section 3.

CONCLUSIONS

A rigorous discrete event modelling technique for nonholonomic mobile robots is proposed. This technique recognises that the dynamics of the mobile robot change in a discrete fashion as the robot becomes constrained by obstacles. Unlike previous discrete event models for mobile robots [8, 9, 7], the model proposed here is rigorous, using the presence of obstacles to define the discrete state. The discrete state also depends upon the dynamics of the robot so that an accurate model of the obstacle is provided for control purposes. A dynamic model permits greater speeds and so, results in faster completion of the task. Experimental results for an indoor office navigation task were presented, demonstrating the effectiveness of the discrete event model. These experiments also demonstrate obstacle modelling with constraints that incorporate the dynamic behaviour of the robot.

References

- [1] J. C. Alvarez, A. Shkel, and V. Lumelsky. Accounting for mobile robot dynamics in sensor-based motion planning: Experimental results. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, pages 2205–2210, May 1998.

- navigation. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, pages 964–969, May 1998.
- [12] A. D. Mali. Tradeoffs in making the behaviour-based robotic systems goal-directed. In *Proc. 1998 IEEE Intl. Conf. on Robotics and Automation*, May 1998. Leuven, Belgium.
- [13] S. Thrun and A. Bucken. Integrating grid-based and topological maps for mobile robot navigation. In *Proc. of the Conf. on Artificial Intelligence*, August 1996.
- [14] K. Ward and A. Zelinsky. Acquiring mobile robot behaviours by learning trajectory velocities with multiple fam matrices. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, pages 668–673, 1998.