Forward Chaining for Robot and Agent Navigation using Potential Fields

Graeme Bell **Michael Weir**

School of Computer Science
University of St Andrews
North Haugh, St Andrews, Fife, Scotland.
Email: gbb@dcs.st-and.ac.uk, mkw@dcs.st-and.ac.uk

Abstract

The ability to navigate successfully is a crucial part of the behaviour of many agents and systems, ranging from robots and computer game characters to neural networks. Navigation in robotics is addressed here using an approach that is extensible to other areas.

Potential fields are acknowledged to be a very powerful representation of robot navigation environments. This representation has been largely abandoned though, due to its susceptability to premature termination of progress caused by local minima.

We seek to encourage the reopening of research into this method by introducing a new approach called Forward Chaining. This technique avoids premature termination of progress by dynamically reshaping the potential field using subgoals which chain forwards towards the goal. A number of increasingly competent and robust navigation heuristics yielding efficient paths are demonstrated. Various avenues for future research are given.

Keywords: Robotics, Navigation, Potential Fields.

1 Introduction

The problem addressed in this paper is that of navigating a robotic agent through 2-D space from a starting position S to some realisable goal position G, in the presence of intervening regions of unrealisable space (such as physical obstacles).

The purpose of this paper will be to expose problematic obstacle configurations for potential field based navigation that produce local minima and cause premature termination of progress. Mechanisms will then be provided that overcome these obstacle configurations, without needing a supporting mechanism that explicitly recognises the problematic obstacle structure a priori. Sections of the paper that are particularly useful in guiding an implementation of the mechanisms have been placed in boxes.

Throughout this paper, it is assumed that we are addressing this problem in a situation where positional information about the robot and obstacles is being provided by lower level algorithms and can therefore be taken for granted. The goal position is provided by some other algorithm or by a human controller. It is not assumed that a global map of obstacles will be possessed a priori. Only the details of those obstacles near the robot during its journey will necessarily be available to the robot through its sensors.

This work was supported in part by funding from the Carnegie Trust for the Universities of Scotland.

1.1 Context

To date there have been a number of approaches to solving this problem (with the above assumptions), featuring differing levels of pre-planning, reactivity, and world-modelling, with varying computational expense and success. The main categories that have emerged are:

- BUG algorithms (Lumelsky & Stepanov 1987, Lumelsky et al 1990), that employ sub-algorithms to carry out line or wall following behaviour.
- Approaches that model the world as a discrete graph, either by placing a grid over the world, or generating a visibility graph, or using a Voronoi diagram. A graph search algorithm (e.g. depth first or A*) is then used to determine a directed subgraph of nodes that connect from the start to the goal. An overview can be found in each of (Latombe 1991, Dudek 2000).
- Potential field-based approaches that use the metaphor of virtual attractive forces pulling the robot towards the goal, and virtual repulsive forces that push the robot away from obstacles, in order to model a summed force that directs the robot. The first example can be found in (Khatib 1986) and overviews can be found in (Dudek 2000, Latombe 1991).

  Previous research in neural networks involving one of the authors (Weir and Fernandes 1994, Lewis and Weir 1999, Lewis and Weir 2000, Weir, Lewis & Milligan 2000) has provided a novel approach for steering using potential fields, and so it is this category that will be developed further here.

1.2 Navigation using Potential Fields

Potential fields provide a simple, intuitive and extensible metaphor that models the effect of obstacles and goals upon the robot as being forces that push and pull the robot around in a continuous space. The robot is usually modelled as a point object.

A mathematical function is constructed to generate these fields. This is typically comprised of a goal function and an obstacle function, which together provide a vector field that is designed to

draw the robot towards the goal and away from obstacles. Equation 1 shows the potential U as a function of q, where q represents a position, $U_{goal}(q)$ the contribution of the goal function at position q, and $\sum U_{obstacles}(q)$ the sum of the contributions of potential from each obstacle function at position q:

$$U(q) = U_{goal}(q) + \sum U_{obstacles}(q) \quad (1)$$

In the case of a two-dimensional navigation problem, this becomes

$$U(x,y) = U_{goal}(x,y) + \sum U_{obstacles}(x,y) \quad (2)$$

Traditionally, $U_{goal}(x,y)$ is modelled as being a quadratic bowl or a cone, and for each obstacle, $U_{obstacle}(x,y)$ is modelled as the the inverse of the distance between the obstacle and the robot. $U_{obstacle}(x,y)$ therefore increases as the robot gets nearer to an obstacle. The resulting field $U(x,y)$ has at least one minimum basin, which lies around the goal, and several high potential areas corresponding to obstacles.

In a manner analogous to that of a ball rolling downhill under the influence of a gravitational potential field, the robot traverses the navigational potential field using gradient descent. With luck, or given a very simple environment, the robot will arrive at the goal's position of global minimum potential. If this happens, navigation has been successful and a path has been generated from the start to the goal. If this does not happen, and the robot has no opportunity to restart navigation, then navigation has failed.

The potential field approach has always been plagued by the difficulty of overcoming the local minimum problem. In the context of robotic navigation, this problem is illustrated by the 'Robot Trap' (Dudek 2000) example in Figure 1 below.

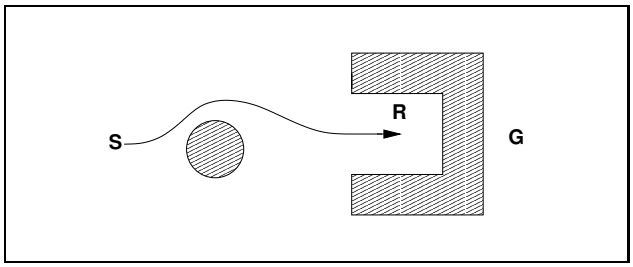

Figure 1: A robot R navigating from S to G is able to navigate around a circular obstacle, but becomes stuck upon entering the robot trap obstacle.

The potential field contribution from the goal causes the robot initially to travel directly towards G. As the robot approaches an obstacle, it begins to be increasingly affected by the obstacle's contribution to the potential field. For many obstacle shapes, the summed potential continues to decrease along the sides of the obstacle in the direction of the goal, which directs the robot to travel around the obstacle.

However, at some point inside the robot trap, overall potential is higher everywhere locally around the robot, and so the robot ceases to move. This stopping point is a local minimum. This is similar to a ball rolling down a hill and getting stuck in a temporary dip in the hill.

Attempts to overcome the local mimimum problem in navigation have included:

- Trying to generate special potential fields that lack local minima, e.g. based on harmonic functions (Koditschek 1987).
- Trying to recognize when progress has been terminated by a local minimum, so that an escape mechanism can be engaged. Overviews can be found in (Latombe 1991, Dudek 2000) as well as a recent overview in (Xi-yong & Jing 2002).

However, none of these attempts has been seen as an outright solution to the problem. In what follows, a new methodology for tackling the *local minimum problem* for navigation will be presented.

1.3 Our Modelling of the Problem

1.3.1 Goal potential

Here, the goal potential $U_{goal}(x,y)$ is modelled as a cone (Latombe 1991) pp 299.

Goal Potential Function

$$U_{goal}(x,y) = \sqrt{((x-x_c)^2 + (y-y_c)^2)}/a \quad (3)$$

- (x,y) is the position at which the potential is being measured.
- (x_c, y_c) is the goal position.
- a is a problem independent parameter that varies the steepness of the cone. We set $a = 0.1$.

1.3.2 Obstacle potential

The total obstacle potential is a summation of the potential of every local obstacle that the robot is aware of, where we define the potential of each obstacle as follows.

The potential inside each obstacle is set to infinity. It is important to ensure that if we are selecting a position to move to based on its lower potential, we will always choose (realisable) positions outside obstacles rather than (unrealisable) positions inside obstacles.

The obstacle field potential outside the obstacle falls from infinity (at the edges of the obstacle) to zero in all derivatives at some finite distance $O_{falloff}$ from the obstacle. This design enables obstacles to be disregarded outside a local area of influence surrounding their boundary. This allows calculations of potential and gradient to be optimised by excluding far off obstacles.

We use circular elements for building obstacles as it is simple for them to meet the above requirements. First, all points within a distance O_{radius} of the center of the obstacle are defined as having infinite potential. Then, the falloff is modelled using a smooth non-analytic function that falls from infinity to zero as the distance from the center of the object varies from O_{radius} to $O_{radius} + O_{falloff}$.

Obstacle Potential Function

$$r = \sqrt{((x-x_c)^2 + (y-y_c)^2)} \quad (4)$$

$$U_{obs}(r) = \begin{cases} 0, & r \geq (O_{radius} + O_{falloff}) \\ \text{falloff(r)}, & r < (O_{radius} + O_{falloff}), \\ & r > (O_{radius}) \\ \infty, & r \geq 0, r \leq O_{radius} \end{cases} \quad (5)$$

$$\text{falloff(r)} = b(1/(r^2 - (O_{radius})^2)) \times e^{(-1/(-(r^2)+(O_{radius}+O_{falloff})^2))} \quad (6)$$

- (x, y) is the position at which the potential is being measured.
- (x_c, y_c) is the center of the obstacle.
- r is a function derived from x, y, x_c, y_c that gives the distance between the obstacle center and the robot.
- b is a problem independent parameter that scales the falloff function. We set $b = 3$.
- O_{radius} is the radius of the obstacle.
- $O_{falloff}$ is the distance over which the function falls off from infinity to zero.

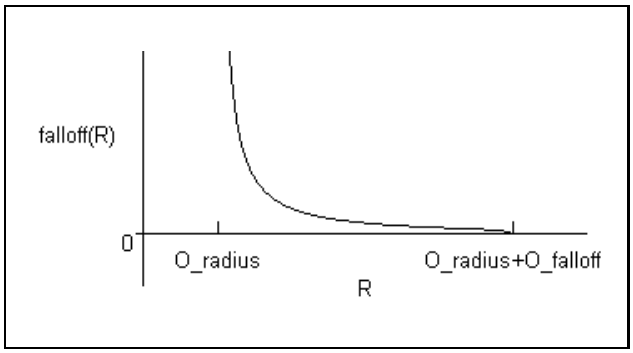

Figure 2: falloff(R) Potential Falloff Function, plotted for $O_{radius} < R < O_{radius} + O_{falloff}$.

The limits of this falloff function are as follows, given the assumptions $O_{radius} > 0$, $O_{falloff} > 0$.

$$\lim_{r \to (O_{radius})+} \text{falloff(r)} = +\infty \quad (7)$$

$$\lim_{r \to (O_{radius}+O_{falloff})-} \text{falloff(r)} = 0 \quad (8)$$

This establishes the continuity of the obstacle function.

1.3.3 Visualising the resultant potential field

Summing the object and goal potential fields as described in section 1.3.1 and 1.3.2 and illustrated in Figure 3 produces a field such as that shown in figure 4.

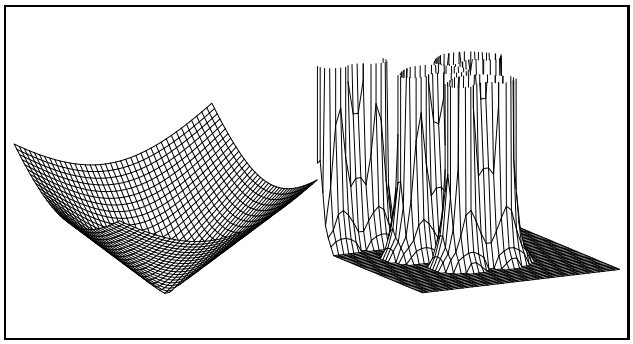

Figure 3: $U_{goal}(x, y)$ and $U_{obstacle}(x, y)$ Surfaces.

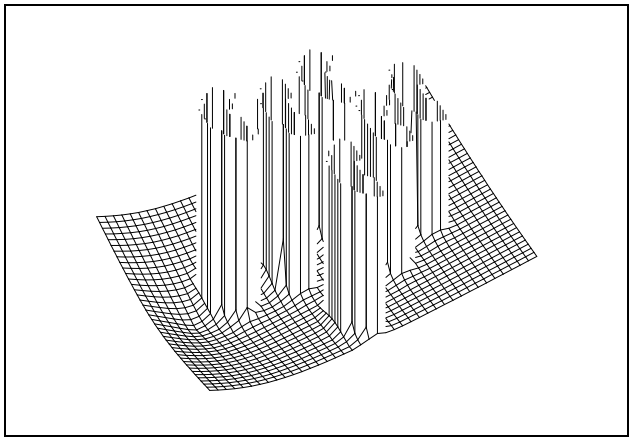

Figure 4: $U(x, y)$ Goal + Obstacle Surface.

2 The Development of Forward Chaining

2.1 Plasticity and Persistence

When gradient descent is used on potential fields such as those generated by the equations in Section 1, and which often contain a number of local minima, it is quite likely that it will not succeed in generating a path that reaches the intended goal.

Gradient descent produces relatively direct shapes of trajectory in continually following the steepest direction downwards given locally by the potential field. As already described, this can cause it to become trapped in local minima basins. On those occasions when gradient descent is able to find a path to a goal though, its directness in steering down the surface tends to produce a path that is reasonably efficient, provided the potential surface is relatively benign and without ravines.

Simulated Annealing (Kirkpatrick et al 1983) is a different potential field approach that is more successful than gradient descent in seeking a state of lowest potential in the presence of local minima. The path that it generates is often extremely indirect, as simulated annealing relies on choosing at random whether or not to follow the potential field downwards. Its initial behaviour at least may be described as highly *plastic* in that it can easily deviate from the gradient of the potential field. The paths generated by using simulated annealing on a navigational potential field could be successful in reaching the goal and not suffering premature termination of progress. However, they are likely to be highly indirect and inefficient, making them relatively useless for efficient travel. Generally, simulated annealing theoretically requires infinite time to ensure success in finding the goal's global minimum. This can make it very inappropriate for either real-time or pre-planned navigation path generation.

In summary, it would be beneficial if an approach could be found that combined the properties of *directness* and *plasticity* in the design of a navigation heuristic. Directness coupled with plasticity would enable a desired position to be reached despite being pushed off course. The path would be able to *persist* towards the goal in a relatively direct and efficient manner. The path is then *goal-directed* (Weir 1984) more properly.

2.2 Subgoal Chaining

In order to model goal-directed plasticity and persistence, we turn to natural agents for inspiration. When human agents describe navigation to a goal,

they do not tend to specify all of the points in the path that they intend to navigate through. Instead, typically a series or *chain* of *subgoals* are chosen that, in combination with some interpolation mechanism to move between them, will result in successful navigation. For example:

> "To get to the hotel, take the third street on your left, go down to the Bank and turn left".

By splitting the task into sub-tasks, stepping stones are created, each of which is easier to achieve than attempting the original task in one go. These subgoals may be chosen ahead of time, or they may be chosen dynamically. When driving in a car, you may pick out in advance a series of cities and towns to travel to in order to reach your destination. In the event that unexpected heavy traffic is encountered, you may dynamically reshape your list of subgoals in order to bypass the heavy traffic - perhaps by deliberately travelling to a town that is off course. This is done in the expectation that you will be able to successfully reshape your travel back towards the final destination later, using your skills at navigating.

To travel along the chain, the agent does not attempt to navigate to the long term goal and tries to navigate instead to the next short term subgoal. The original goal therefore becomes the final subgoal that marks the end of the chain. By the time this final subgoal is to be attempted, the agent will have been positioned close to the goal by the rest of the subgoals in the chain, so that it will be possible to navigate successfully to the goal in the final step.

Subgoal chaining is well known and frequently used in symbolic AI in the form of agendas (Rich and Knight 1991). It is less well known in other fields of heuristic search such as neural networks and robotics, though some work has been done (Weir & Fernandes 1994, Gorse et al 1997, Lewis & Weir 1999, Lewis & Weir 2000, Weir et al. 2000, Xi-yong & Jing 2002).

A mechanism will now be described that is capable of joining these subgoals into a continuous path that the robot can physically traverse. Gradient descent is used here for this purpose, with the proviso that the target is a subgoal which is nearby. To create the subgoal potential field that gradient descent will traverse, it is necessary to remove the original goal potential contribution from the field, and replace it with a subgoal potential field. The subgoal potential field is similar in nature but has the subgoal co-ordinates at the centre of its basin rather than the goal co-ordinates. Gradient descent will tend to produce a reasonably direct and successful path to a subgoal on a potential field if the subgoal and the current position are close together. The reason for this is that the descent is likely to take place within a basin that contains a global rather that a local minimum. It is important to remain aware that although gradient descent is much more likely to reach a nearby subgoal than a far-off goal, it is still not fully guaranteed. Ideally, the subgoal selection mechanism should be capable of dealing with a degree of failure in travel to an individual subgoal.

With inter-subgoal movement being carried out in this manner, the task of navigation has now become that of selecting subgoals to be attempted, in such a way that the path produced by traversing the subgoal chain *plastically* deviates round obstacles and then *persists* in heading for the goal.

A template navigation heuristic for dynamically constructing a chain and traversing between its subgoals is therefore:

Template Subgoal Chaining Heuristic

1. Pick a subgoal to be attempted that should produce progress towards the goal.
2. Generate a potential field for the subgoal.

$$U_{sgo}(x,y) = U_{subgoal}(x,y) + \sum U_{obstacles}(x,y) \quad (9)$$

3. Perform gradient descent on $U_{sgo}(x,y)$ starting from the current position. Continue until the current position becomes equal to the subgoal co-ordinates, or until some timeout condition is reached (perhaps a reasonable number of steps taken without success) that indicates failure to reach the subgoal.
4. If the final goal has been reached, then exit. Otherwise, return to step 1.

In summary, the merit of the subgoal chaining approach is that the task is broken down into a sequence of subgoals that are for the most part readily achievable, certainly compared with the task as an indivisible whole. In particular, the danger of the process becoming stuck in a local mimimum on a fixed potential surface is replaced by the relative safety of being carried in a global minimum basin. This global minimum basin is varied dynamically using a series of subgoal potential surfaces until the basin reaches the goal.

Heuristics for the single aspect of subgoal selection which are proposed hereafter are implementations of step 1 of the whole template navigation heuristic.

2.3 Defining a Subgoal Selection Mechanism

Dynamic surface variation is not all plain sailing though, and a cautionary note is needed. As a simple example of a possible mechanism, consider a linear subgoal chaining mechanism similar to that proposed by a paper from the neural network research community (Gorse et al 1997). This linear mechanism sets subgoals at a fixed distance *in a direct line towards the goal* from the current position. By attempting each subgoal on the linear chain in succession, the technique requires at each stage that progress is made in the compass, i.e. straight, direction towards the goal. Such travel is not possible when at a position of local minimum potential for the goal as travel must move temporarily *away* from the goal. Consequently, the process in linear subgoal chains gets stuck[1].

Techniques such as ERA fail to reach their goals when faced with local minima (Lewis & Weir 1999).

Robot motion suffers in a similar manner. When a robot using linear chaining to select subgoals enters a minimum such as that associated with the robot trap problem, it will repeatedly try to place subgoals in the same direction as the goal. Since these subgoals do not allow direction away from the goal and out of the robot trap, the robot remains stuck. Linear subgoal chaining mechanisms have insufficient plasticity to overcome the local minimum problem. This leads us to make the following conjecture.

Conjecture 1 *A subgoal selection mechanism can only be highly successful if it uses a non-linear chain, in order to gain sufficient plasticity.*

[1] The basin being travelled in becomes one of a local rather than a global minimum.

A number of further conjectures, and the problems they allow to be tackled, follow in the remaining parts of this section. The broad theme is to show that the local minimum problem can be turned into a *repeating cycle problem* through subgoal chaining. That is, instead of the danger being that of getting stuck at a local minimum, the danger becomes one of being sucked into a repeating cycle of motion that does not include the goal. The repeating cycle problem identifies the fact that regions associated with particular minima for the goal-obstacle field remain problematic for subgoal chaining, and that more sophisticated subgoal selection is needed to avoid repeating cycles. However, the necessary sophistication can be provided through incrementally adding relatively simple mechanisms, thus making the problems tractable.

We transform the local minimum problem into another problem, and begin the development of subgoal chaining, with the following.

Conjecture 2 *To avoid getting stuck in one state, the next state should be taken from a set of states at a significant distance from the current state.*

This prevents the current state being selected as the next state . In practice, the set of states is taken to be on a circle surrounding the current state, and the next state for the following heuristic is the point of lowest potential for the goal-obstacle field on the circle.

LPCIRCLE subgoal selection heuristic

1. If the distance between the current position and the long term goal is less than distance D, set the subgoal to be the goal. Otherwise:
2. Scan the circle at a distance D surrounding the current point for the point of lowest potential and use it as the subgoal.

   To do this:

   (a) Calculate the value of the goal-obstacle $U(x, y)$ potential field, at a distance D from the current position for a selection of angles in the range 0-360 degrees around the current position.

   (b) Pick the angle at distance D which resulted in the lowest potential, and use the corresponding position as the subgoal for gradient descent.

The heuristic works well for problems without local minima. The next step is to evaluate the performance for a problem with a local minimum.

2.4 The Shallow C Problem

This first problem consists of a shallow C shape lying between the start and the goal, so that a local minimum basin is produced. This is represented by the rightmost obstacle of Figure 5.

Consider a robot entering this local minimum. For shapes with shallow curvature, the robot will soon find points of lowest potential on the chosen circle surrounding the robot to lie along a line in front of, parallel and close to the obstacle boundary. It will consequently set subgoals along this line and transit to points close to the line through gradient descent on the subgoal-obstacle potential field.

The local minimum for the goal-obstacle field lies along the line mid-way along the front of the shape. After passing through this point, the subgoal will be set back towards the point as it represents a point of locally lowest potential for the goal-obstacle field. This sets up a repeating cycle in the form of a local oscillation between a pair of states that prevents further progress. The local minimum is thus associated with a repeating cycle problem for subgoal chaining.

To ensure progress, a successful mechanism should be able to override the goal-obstacle field's reversal of direction and instead continue to place subgoals forwards. By forward, we are suggesting a direction that is in the worst case 90 degrees to the left or right from the direction in which subgoals were previously being placed.

Conjecture 3 *To avoid oscillation between a pair of states, the next subgoal should be set in the forwards direction defined by the tangent to the current subgoal path.*

This ensures that the subgoals and transitions are not set in a back and forth pattern. It also reduces the main residual threat, to being one of repeat cycles of length > 2.

Based on this conjecture, the first prototype of the subgoal picking mechanism (FWDS1) operates as follows. FWDS1 works by setting a subgoal at the point of lowest goal-obstacle potential along the forwards semi-circle. The point of lowest potential for such a subgoal in conjunction with the obstacle field is then transited to (Figure 6).

FWDS1 works well for obstacles with shallow curvature by choosing points along the near-boundary line described above, which runs parallel and just in front of the obstacle boundary. This allows the boundary to be circumnavigated closely and effectively with continuous forwards movement.

FWDS1 Subgoal Selection Heuristic

1. If the distance between the current position and the long term goal is less than distance D, set the subgoal to be the goal. Otherwise:
2. The vector from the second last attempted subgoal to the last attempted subgoal indicates the *forward direction*. If there are no previous subgoals, then initialise the *forward* direction to point directly at the goal.
3. Scan the *forward* semicircle at a distance D for the point of lowest potential and use it as the subgoal. To do this:

   (a) Calculate the value of the goal-obstacle $U(x, y)$ potential field, at a distance D from the current position. Do this for a selection of angles at most up to +/- 90 degrees from the *forward direction* (see Figure 6).

   (b) Pick the angle at distance D which resulted in the lowest potential, and use the corresponding position as the subgoal for gradient descent.

2.5 The Moderate C problem

FWDS1 fails when faced with obstacles such as a C shape with large enough total curvature across the C (such as the obstacle second from the right in Figure 5). We call these C's "Moderate C" shapes. The failure of FWDS1 occurs after the path has begun to follow a line parallel to the obstacle boundary.

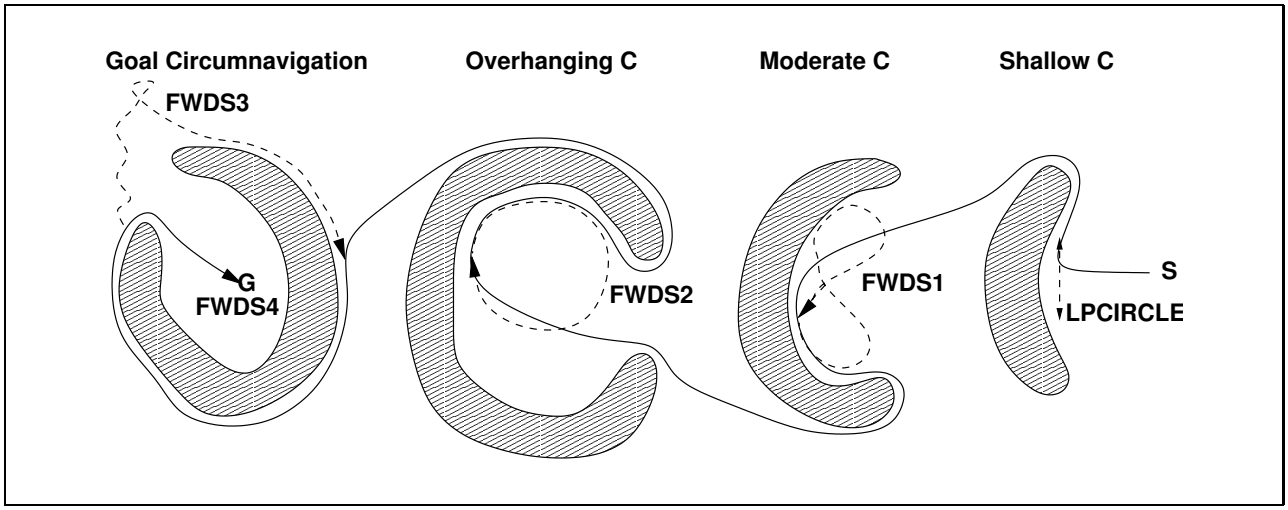

Figure 5: Informal representation of the problematic local obstacle structures and the typical paths taken by each subgoal selection heuristic. Repeating cycles are shown with a dashed line where they deviate from the path of FWDS4. The heuristic suffering from each repeating cycle is marked next to the dashed line. Each obstacle is built from circular elements in the experimental implementation.

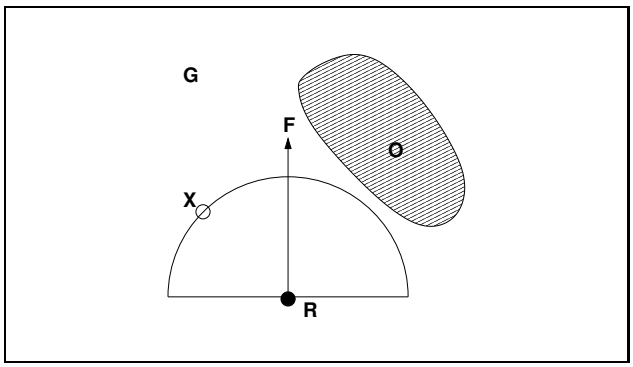

Figure 6: Selection of a FWDS1 subgoal. F is the tangential forwards direction, R the robot's current position, G the goal, O an obstacle and X the selected position of lowest potential on the forward semicircle and hence the chosen subgoal.

For obstacles of moderate total curvature, the point of lowest potential, and hence the subgoal, comes away from being along a line parallel and close to the boundary to being in a local closed loop more deeply in *free space* - i.e. space that is free of obstacles. This loop in fact contains the local minimum for the goal-obstacle field. The closure of the loop means that a repeating cycle sets in with no further progress.

An extra mechanism is now required to ensure the line of lowest potential for the subgoal-obstacle field flows around the obstacle and on towards the goal.

Conjecture 4 *To avoid a cycle of n states locally occurring in a region of free space, (n > 2), the next subgoal should be set at or behind the obstacle boundary.*

The new location of the subgoal ensures that the line of lowest potential more closely follows the obstacle boundary, rather than a local loop in free space. Fully implemented, it will reduce the threat to being that of more global loops of states, i.e. those occuring due to global features of the environment.

The implementation of Conjecture 4 for Moderate C shapes is done as follows.

Conjecture 5 *For Moderate C shapes, placing a subgoal to the goal-side of the forwards direction results in subgoals being placed behind the boundary.*

This provides a mechanism by which the boundary can be followed in the case of an obstacle of moderate curvature being encountered.

In particular, FWDS2 is designed to encourage the subgoal to be set in a position at or behind the obstacle boundary by rotating the subgoal from the tangential forwards direction towards the goal direction by a set amount. This makes the point of lowest local potential in the subgoal-obstacle field return to being a point close to and along the obstacle boundary (for moderate curvature). Success with shallow curvature is preserved as well. The threat is now effectively reduced to that of obstacles of high total curvature and more global loops.

FWDS2 Subgoal Selection Heuristic

1. If the distance between the current position and the long term goal is less than distance D, set the subgoal to be the goal. Otherwise:
2. The *forward* direction is defined as before.
3. The goal direction is defined as being the vector that points towards the long term goal from the current position.
4. A temporary subgoal SG_1 is placed at distance D in the current *forward* direction. This subgoal is then swung round the *forward* semicircle towards the goal, up to a limit of B degrees either clockwise or anti-clockwise. The direction in which the subgoal is swung should be the direction which moves the subgoal round to the goal direction most quickly. The range B limits the extent to which the goal can bias movement away from the *forward* direction. If the goal direction occurs within the range, the rotation is simply to fall in line with the goal direction.
5. Scan the *forwards* semicircle at a distance D for the point of lowest potential on the subgoal-obstacle field and use it as the subgoal SG_2. To do this:
   (a) Calculate the value of the subgoal-obstacle $U_{sglo}(x, y)$ potential field, at a distance D from the current position. Do this for a selection of angles at most up to +/- 90

degrees from the *forwards direction*.

(b) Pick the angle at distance D which resulted in the lowest potential, and use the corresponding position as the subgoal SG_2 for gradient descent.

2.6 The Overhanging C Problem

FWDS2 fails when presented with a C shape that has such a high total curvature across it, that the ends of the C form an overhang for navigating round the C (see the second from left obstacle of Figure 5). We call these C's "Overhanging C's". The reason for failure is that at the overhang, turning towards the goal puts the subgoal into free space rather than into the obstacle. The situation of putting the subgoal into free space when turning towards the goal also occurs once an obstacle has been successfully circumnavigated, at which point the robot correctly heads for the goal by turning towards it. However, turning towards the goal and putting the subgoal into free space is unsafe at an overhang as it leads the robot back into the centre of the C, and into a repeating cycle of motion within the C.

Conjecture 6 *To avoid a cycle of n states occurring while circumnavigating an obstacle of high total curvature, the obstacle should only be left when it is safe to head to the goal.*

In practice the safety required in Conjecture 6 is achieved by tracking the total angle through which the agent's path has been *twisted*.

If the twist is severe enough, the agent will find itself looping back into its initial direction towards the goal. Moving off an overhang is an example of this. The agent should instead place the subgoal in the opposite direction, so as to untwist itself by continuing to follow the obstacle boundary.

The mechanism we add here works by calculating a cumulative angle that the robot's path has twisted through. This can be done by keeping track of the last two positions occupied by the robot and using them to determine the angle between the robot's last two average direction vectors. This angle is added to a running total called the *path twist*. If at any point during navigation the *path twist* exceeds a threshold value (we have experimentally determined 135 degrees is a suitable threshold), then the path has become too twisted and will soon be in danger of turning back on an earlier path. The correct behaviour in this situation is to automatically place the biased subgoal as far as possible in the opposite direction to the twist, so as to try and untwist. The line of lowest potential then flows successfully around overhangs as well as being able to successfully flow around the earlier problem shapes. The main residual threat has now been reduced to one of non-local loops in free space.

FWDS3 Subgoal Selection Heuristic

As FWDS2, but with the following new steps.

2A. Calculate the angle between the vector from the second last position to the last position, and the vector from the last position to the current position. Add this angle to the cumulative *path twist* variable.

4A. If the *path twist* exceeds 135 degrees, then the *goal-bias* subgoal should be swung to the edge of the range in the clockwise direction, to untwist. If the *path twist* is less than minus 135 degrees, the *goal-bias* subgoal should be swung to the edge of the range in the anti-clockwise direction, to untwist.

2.7 The Goal Circumnavigation Problem

FWDS3 selects subgoals so as to circumnavigate obstacles and return the agent back towards its initial goal direction. It can therefore fail if this direction is no longer progressive towards the goal. This case can occur if an obstacle is large or near enough to the goal to surround it, since in circumnavigating the obstacle, the robot has significantly circumnavigated the goal (e.g. the leftmost obstacle in Figure 5 and the examples in Figure 7). The initial goal direction no longer points the agent towards the goal and so may not be progressive. To correct for this effect, another property is measured that tracks this effect - the *goal twist*. This is measured by calculating the angle between the vectors pointing from the goal to the previous and current positions, during each iteration of subgoal selection. The relative angle thus generated is then summed over many iterations to generate a cumulative *goal twist* angle.

Consider the situation where the agent has rotated by perhaps 60 degrees anti-clockwise around the goal. The correct direction for FWDS3 to resume (relative to the original goal direction) will then also be rotated by 60 degrees anti-clockwise, in order to continue pointing at the goal. This example shows that the application of *goal twist* to the FWDS3 heuristic is straightforward. Where previously a decision was made on the cumulative *path twist* value, it will now instead be made on the value of the cumulative *path twist* relative to the cumulative *goal twist*.

FWDS4 Subgoal Selection Heuristic

As FWDS3, but adding 2B and replacing 4A.

2B. Calculate the angle of rotation about the goal by finding the angle between the vectors (goal $\rightarrow$ last position) and (goal $\rightarrow$ current position). Add this angle to the cumulative *goal twist* angle rotated about the goal.

4A. The cumulative *path twist* minus the cumulative *goal twist* is calculated and called the *relative path twist*. If the *relative path twist* exceeds 135 degrees, then the *goal-bias* subgoal should be swung to the edge of the range in the clockwise direction, to untwist. If the *relative path twist* is less than minus 135 degrees, the *goal-bias* subgoal should be swung to the edge of the range in the anti-clockwise direction, to untwist.

FWDS4 is the last mechanism in the development of subgoal chaining as far as the paper is concerned. The Future Work section contains descriptions of further problems that suggest where further development is needed. The series of subgoal chaining mechanisms given so far is nevertheless sufficient to show how the local minimum problem can be transformed into a series of tractable repeating cycle problems. We now turn our attention to proof of concept demonstrations.

3 Experiments

These experiments are intended to verify that the mechanisms given in this paper are capable of dealing with the problem situations they were designed to overcome, while retaining the ability to succeed on simpler problems.

3.1 Arena Configuration

There are 4 arenas: Shallow C, Moderate C, Overhanging C and Goal Circumnavigation. Each arena contains a single instance of the corresponding obstacle type. The shapes of the obstacles roughly correspond to those seen in Figure 5.

In the three types of C arenas, goals will be placed so that success is only possible by navigating past the obstacle structure and then on towards the goal. We would expect each heuristic to fail as described in Section 2, for example FWDS1 should not be able to navigate past a Moderate or Overhang C.

The Goal Circumnavigation arena is slightly different as the task is not to navigate past the obstacle, but rather round and into the obstacle, then on to the goal. Therefore we would expect that the FWDS3 approach, because it uses path twist relative to the initial goal direction should fail on this arena. We would expect FWDS4 to succeed due to the change to relative path twist. It might be expected that FWDS1 and FWDS2 could succeed as path twist does not feature in these heuristics.

Measurement of distance in these arenas will be in abstract units. In the animations of experimental results, 1 unit corresponds to a distance of 10 pixels.

3.2 Experimental Configuration

The robot must navigate from a starting position S to a goal position marked G on each arena. The experiments have been conducted using a variety of randomly chosen S and G positions spaced approximately 30 units apart, subject to the constraint that the start and the goal are positioned so that the obstacle structure is faced at some point during navigation. Navigation is considered successful if the goal is reached within some reasonable number of steps (here, 50 steps). Navigation is considered to have failed if the step limit is reached without successfully reaching the goal.

3.3 Heuristic Parameters

The following heuristics will be tested: Standard Gradient Descent on a goal-obstacle field (GD), Linear Chaining (LC), LPCIRCLE Chaining (LPC), FWDS1 Chaining (F1), FWDS2 Chaining (F2), FWDS3 Chaining (F3), FWDS4 Chaining (F4).

The choice of the subgoal step size D requires some domain knowledge of the sort of arena that will be faced. An ant sized robot for example would require a small D to be able to navigate round the obstacles it is likely to encounter - a human sized robot would require a correspondingly larger step size. The size has to be set small enough to be able to roughly follow the obstacle boundary, and large enough to avoid being thrown around by small scale fluctuations in the potential field. The step size is thus dependent upon the problem domain, but is independent of particular problems within the domain. The fixed distance D used here for all subgoal chaining heuristic experiments will be set to 2 units.

Wherever gradient descent is used, the step size will be set to 0.2 (on this scale of arena, this will ensure that obstacles are not stepped into).

If gradient descent is being carried out on a subgoal-obstacle surface, it will be terminated if it reaches within 1 step of the known subgoal position (i.e. successful navigation to the subgoal) or after reaching a limit of 15 steps (i.e. failure to navigate to the subgoal) as this allows enough time to reach a realisable subgoal placed 2 units away. After gradient descent is terminated for a subgoal, a new subgoal will be chosen as per the subgoal chaining heuristic, and gradient descent will then begin to move towards the new subgoal.

The *goal-bias* range is set here to be up to +/- 45 degrees. This is not problem dependent.

3.4 Results

Table 1 below shows the results of the experiments after being carried out on a variety of starting/goal combinations. The result is marked as ✗ in the event that all of the navigation attempts failed, and ✓in the event that navigation was successful in all experiments.

Problem	GD	LC	LPC	F1	F2	F3	F4
Shallow C	✗	✗	✗	✓	✓	✓	✓
Moderate C	✗	✗	✗	✗	✓	✓	✓
Overhang C	✗	✗	✗	✗	✗	✓	✓
Circum'tion	✗	✗	✗	✓	✓	✗	✓

Table 1. Table of Experimental Results

The results are in line with the expectations described in Section 3.1. AVI animations generated as part of the results shown in Table 1 are available at `http://www.dcs.st-and.ac.uk/~icg/resources/local_research/animations/forwards-chaining/`.

3.5 Analysis of Results

- Gradient descent fails when presented with shallow C obstacles as a result of premature termination of progress due to a local minimum. Linear subgoal chaining is as (un)successful as gradient descent, as expected. LPCIRCLE oscillates as expected.
- FWDS1 and FWDS2 prove to be adept at solving the Goal Circumnavigation problem as anticipated earlier.
- FWDS3 fails on Goal Circumnavigation, due to the addition of path twist (relative to the initial path) into the heuristic. When the path twist reaches its threshold value, the robot tries to untwist, resulting in behaviour that causes it to miss its opportunity to head for the goal via the entrance in the obstacle (See figure 5).
- Forward Chaining using the prototype FWDS4 subgoal selection heuristic is most successful and can overcome the full range of tested problems insurmountable by gradient descent.
- In general, each mechanism added to FWDS has allowed success in the presence of the obstacle configuration it was designed to overcome. Additionally, robust success has been maintained on simpler problems. The exception to this is FWDS3 which gains success on Overhang C at the expense of its ability to navigate around the Goal Circumnavigation problem.
- The paths taken by Forward Chaining were relatively direct as well as plastic. This can be seen in the animations and in Figure 5.

4 Future Work

4.1 Comparison with Other Approaches

Future work may involve comparing this technique quantitatively and qualitatively against other navigation techniques (besides standard gradient descent) that involve potential fields, and against techniques in other domains, such as BUG or graph-based approaches.

4.2 3-D and n-D Navigation

Currently, we are investigating how to translate the 2-D Forward Chaining approach into a 3-dimensional and ultimately an n-dimensional navigation mechanism, for arbitary n. A simple and well defined heuristic for 3-D robotic navigation would be of great benefit to anyone seeking to operate a robot in an 3-D environment. To envisage the advantages of n-D navigation, consider a robot arm with 10 degrees of freedom, each provided by a single motor. Each possible position of the arm has a representation as a combination of settings for each of the 10 actuators involved. This representation corresponds to a state in an n-D space[2] where n=10. Having a heuristic that is able to navigate from one 10-D configuration to another 10-D configuration in an efficient and successful way could make controlling complex robots significantly simpler. Impossible combinations of actuator settings would correspond to unrealisable regions on the 10-D navigational potential field, and would be readily bypassed by Forward Chaining treating them as 10-D obstacles.

The main issues requiring resolution for n-dimensional navigation are those of selecting a point of lowest potential, and twist. In the present 2-D approach, potential values are sampled along the forward semi-circle so that a good approximation to the lowest value can be found. The computational costs of such a discrete sampling method are still feasible for 3-D (using a forward hemispherical surface), but become infeasible for large enough n-D spaces. A method based on steepest gradient descent is envisaged to yield a more feasible method for low potential position selection.

As far as twist is concerned, it may be possible to treat it as a combination of independent twist angles in higher dimensions. For example, the two angles in spherical co-ordinates may provide independent twist angles for 3-D and generalise to (n-1) angles in n-D.

4.3 Global Problems

With Conjectures 3 to 5, we have tried to cater for the threat of repeating loops that could otherwise occur locally. With Conjecture 6, we have begun to also address the issues raised in situations where the path has the task of significantly circumnavigating the goal as well as local obstacles. We would like to ensure successful navigation for at least a broad range of these global cases as well. We have two problems that show next steps in what has to be tackled. These are:

1. The mushroom problem (left, Figure 7).

   Here, each side of the base of the 'mushroom' has the effect of reversing the robot's direction and twist, causing a large global cycle of motion through the arena to be set up.

2. The spiral problem. (right, Figure 7).

   The robot is trapped in a long term global cycle, which circumnavigates the goal entirely.

[2]See (Russell and Norvig 1995) for a description of robot configuration spaces.

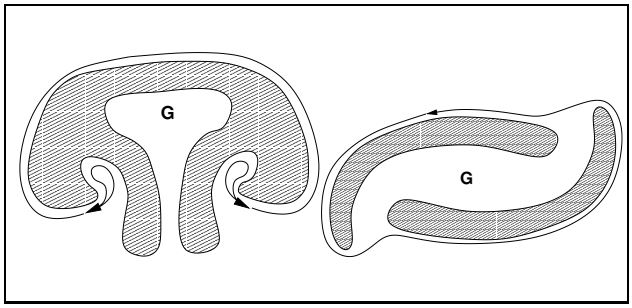

Figure 7: The Mushroom and Spiral Problems

It is therefore necessary to recognise that under special circumstances it is still possible for FWDS4 to fail to navigate to the goal on some maps. Although the situations causing this to happen are likely to appear on a sufficiently large, dense and mazelike map, we do not consider this failing to devalue the present Forward Chaining heuristics. They have already markedly expanded upon the capabilities of gradient descent, allowing successful navigation on most, if not all, small arenas, and we believe they can be expanded further.

4.4 Applications for Forward Chaining

Forward Chaining clearly has immediate application to robotic navigation in a 2-D plane, for use with either simulated or physical robots.

An interesting secondary application lies in the field of computer games. Several genres of game are based upon a player interacting with a 2-D map. In these genres, it is often necessary for a human or computer player to direct their units across a map to some far off position near an opponents base. The command is usually given by selecting the unit at its current location, and clicking on the part of the map that you require the unit to navigate to.

A traditional consequence of sending units to navigate their way across the map is that the units become stranded in situations directly akin to the C problem described in this paper. These C problems are caused in game by the presence of decorational or strategic map features that units cannot cross, such as rocks or water. It can be frustrating for players to discover in the heat of battle that the cavalry they expected to save the day were unable to navigate past a C shaped beach near their home base[3].

Approaches currently used to overcome the fallibility of existing navigation heuristics include:

- Keeping maps small, and avoiding creating arrangements of impassable features that have shapes such as the C problem.
- Waypoint mechanisms. Instead of specifying a single point to navigate to, the player specifies a number of discrete points along a path that are each intended to be easily reached from the previous point, by even a poor navigation heuristic. The unit navigates to each point in turn until it reaches the goal.

The similarity of waypoint mechanisms to subgoal chaining is clear. To compensate for the deficiencies of the unit navigation heuristic, the human is being forced to select subgoals that allow the interpolating navigation heuristic to succeed. One application of Forward Chaining is as an alternative to forcing the player to manually select all waypoints. Instead, perhaps a set of waypoints could be automatically

[3]One of the authors has personally experienced several such C-shape related gaming catastrophes!

suggested to the player when they click on a location. Another application would involve removing waypoints entirely from the game, so that units can be left to navigate for themselves, leaving only high level navigation instructions to the player.

It is hoped that extending Forward Chaining to 3-D will produce similar spinoff applications in computer games based on 3-D environments in addition to the 3-D robotic navigation applications.

5 Conclusion

This paper has described the problem of robotic navigation and some of the approaches that have been taken to address it. In particular, potential field based approaches were considered and the difficulties posed by local minima were highlighted. Navigational potential fields were modelled using novel functions that allow obstacles to have only local influence.

The concepts of *plasticity* and *persistence* were introduced, and the idea of using *subgoal chaining* to dynamically reshape the potential field was suggested as a way to allow potential field based navigation to overcome local minima. Subgoal chaining was then shown to provide a way of transforming the local minimum problem into the tractable repeating cycle problem. A number of increasingly difficult obstacle configurations were presented, and corresponding steering mechanisms were devised to allow success through appropriate subgoal placement.

Forward Chaining creates continuous forward motion along a curved line of lowest potential that circumnavigates obstacles, and persistently heads for the goal after each circumnavigation. Simulated experiments confirm that each of the mechanisms integrated into the Forward Chaining navigation heuristic has improved its competence and robustness. The success of Forward Chaining on all of the experimental problem configurations underlines this fact. The approach is uniformly successful across the wide range of obstacle types tested, using parameters which are robust and largely problem independent. The range of problems solvable first-time has gone beyond those of fixed surface descent and on to a large and increasing number of problems that are solved using a small number of simple features.

In conclusion, this work has developed a potential field based heuristic that is capable of successfully navigating on a far wider range of problems than previous gradient descent based approaches. New avenues of research have been opened and it is hoped that future work will improve the level of success in the presence of more global problems and allow successful navigation in spaces with higher dimensionality.

References

Kirkpatrick, S., Gelatt, C. & Vecchi, M., (1983), 'Optimisation by Simulated Annealing', *Science* (200),671-680.

Lumelsky, V. & Stepanov, A., (1987), 'Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitary shape', *Algorithmica*, 2(4):403-440.

Lumelsky, V., Mukhopadhyay, S. & Sun, K., (1990), 'Dynamic path planning in sensor-based terrain aquisition', *IEEE Trans. Robotics and Automation*, 6(4):462-472.

Gorse, D., Shepherd, A. J. & Taylor, J. G. (1997), 'The new era in supervised learning', *Neural Networks*, 10(2):343-352.

Khatib, O., (1986), 'Real-Time Obstacle Avoidance for Manipulators and Mobile Robots', *The International Journal of Robotics Research*, 5(1).

Xi-yong, Z. & Jing, Z., (2002), 'Virtual local target method for avoiding local minimum in potential fields based navigation', *Journal of Zhejiang University SCIENCE*, 4(3):264-269 2003.

Koditschek, D., (1987), Exact robot navigation by means of potential functions: Some topological considerations, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', Mar 1987, pp 1-6.

Weir, M. & Fernandes, A., (1994), Tangent hyperplanes and subgoals as a means of controlling direction in goal finding, *in* 'Proceedings of the World Conference on Neural Networks', Volume 3, pp 438-443.

Lewis, J. & Weir, M. (1999), Subgoal Chaining and the Local Minimum Problem, *in* 'Proceedings of the International Joint Conference on Neural Networks', 1999.

Lewis, J. & Weir, M. (2000), Using Subgoal Chaining to Address the Local Minimum Problem, *in* 'Proceedings of the International ICSC Symposium on Neural Computation', 2000.

Weir, M., Lewis, J., Milligan, G. (2000), Using Tangent Hyperplanes to direct Neural Training, *in* 'Proceedings of the International ICSC Symposium on Neural Computation', 2000.

Dudek, G. (2000), *Computational Principles of Mobile Robotics*, Cambridge University Press.

Latombe, J. (1991), *Robot Motion Planning*, Kluwer Academic Publishers.

Weir, M. (1984), *Goal-directed Behaviour*, Gordon and Breach Science Publishers.

Rich, E. & Knight, K. (1991), *Artificial Intelligence*, McGraw-Hill Education.

Russell S. & Norvig, P. (1995), *Artificial Intelligence*, Prentice Hall.