POTBUG: A Mind's Eye Approach to Providing BUG-Like Guarantees for Adaptive Obstacle Navigation Using Dynamic Potential Fields

Michael Weir, Anthony Buck, and Jon Lewis

School of Computer Science, University of St Andrews
North Haugh, St Andrews, Fife, Scotland
{mkw, arb11, jpl3}@st-and.ac.uk
http://www.dcs.st-and.ac.uk

Abstract. The problem we address is adaptive obstacle navigation for autonomous robotic agents in an unknown or dynamically changing environment with a 2-D travel surface without the use of a global map. Two well known but hitherto apparently antithetical approaches to the problem, potential fields and BUG algorithms, are synthesised here. The best of both approaches is attempted by combining a Mind's Eye with dynamic potential fields and BUG-like travel modes. The resulting approach, using only sensed goal directions and obstacle distances relative to the robot, is compatible with a wide variety of robots and provides robust BUG-like guarantees for successful navigation of obstacles. Simulation experiments are reported for both near-sighted (POTBUG) and far-sighted (POTSMOOTH) robots. The results are shown to support the theoretical design's intentions that the guarantees persist in the face of significant sensor perturbation and that they may also be attained with smoother paths than existing BUG paths.

1 Introduction

1.1 The Problem

The general problem we address is how to get a robot to navigate from A to B where there are intervening obstacles.

In static and familiar environments, a path between a particular A and B that circumvents obstacles may be known before travel commences. If not, a global map showing the location of intervening obstacles may allow a path between A and B to be computed [1]. Knowledge of a connecting path coupled with knowledge of a reliable mapping of robotic action into motion then enables a sequence of actions moving the robot from A to B to be readily computed.

However, such knowledge may not be available for autonomous robots working in unknown or dynamically changing environments. An important consequence is that the unknown throws up unexpected aspects of the environment resulting in perturbation of the robot motion. The motion task then is a challenging one of finding a route based on more limited knowledge through adaptive behaviour. A common

S. Nolfi et al. (Eds.): SAB 2006, LNAI 4095, pp. 239 – 250, 2006.

scenario is where knowledge is restricted to goal direction and local obstacle data provided through sensors. This scenario in autonomous obstacle navigation is addressed here for a 2-D travel surface.

1.2 Existing Approaches

There are numerous techniques for robotic navigation in 2-D space that do not rely on the existence of a global map providing a priori knowledge of the environment. They feature varying levels of pre-planning, reactivity and world modelling and differ with respect to their computational expense and success. Major general approaches as described in [2] are:

- BUG based algorithms (e.g. [3], [4]) which provide geometrical paths connecting to the goal. For other non-heuristic algorithms, see [5].
- Methods which develop a discrete model of the environment that may, for example, be searched via an A* algorithm to establish an optimal connection from start to goal. Overviews can be found in [6] and [7].
- Potential field based systems in which the motion of a robotic agent is directed by a combination of repulsive obstacle potential and attractive goal potential [8]. Overviews can be found in [6] and [7].

We focus here on the two approaches in the above that employ a direct travel path without search, i.e. potential fields and BUG algorithms.

1.2.1 Potential Field Based Navigation

The potential field approach views the robot as analogous to a charged particle attracted to the goal and repelled by the obstacles [8]. These virtual forces are used to guide the robot in navigating its environment. By converting sensor information found in the field into a combination of attractive and repulsive potential, motion may be generated towards the goal through the direction of the potential field gradients. The goal distance and the inverse of the local obstacle distance are commonly taken to represent the degree of attractive goal potential and repulsive obstacle potential respectively at each location. Provided the goal is reasonably clear of obstacles, it is identifiable as a unique location of lowest combined potential. The combined potentials may be visualised abstractly as a surface of varying height over the travel environment. Standardly, the locally steepest gradient downwards on the potential surface points the way to the goal so that travel proceeds like a ball rolling to the bottom of a hill. Such an approach offers the possibility of robust travel since increasing error between actual and intended motion will result in steeper gradients coming into play to guide the motion back on course. The underlying potential field framework also offers compatibility with a wide range of robot designs and a relatively seamless integration with robot features such as finite size and sensor-based perception.

While there has been substantial enthusiasm for the potential field approach in robotics [7], [9], progress has been blocked by the commonality of features such as the well known Local Minimum Problem [10]. In the latter case, obstacles containing

common shapes of many varieties such as C shapes cause the robot to become stuck upon entering the convex inside of a region bordering the obstacle, e.g. the inside of the C. The robot cannot escape the region except by temporarily going further away from the goal, which would entail going upwards in potential against the downward flow in the static field. The behaviour is equivalent to a ball getting stuck in a local pit part way down the hill.

1.2.2 BUG Algorithms

There are many algorithm variants belonging to the BUG family of algorithms beyond those initially developed in [3], [4], that also provide a guarantee for their geometric paths to reach a realisable goal. The family commonly assumes a point robot analogous to a bug which proceeds directly towards the goal when it can, and steadily goes forwards along intervening obstacle boundaries until the boundaries can be left to carry on directly towards the goal.

The guarantee comes about because, in 2-D, going forwards along an obstacle boundary after hitting the obstacle always eventually results in a point being reached where the obstacle can be left *safely*. That is, the boundary traversed is left permanently with no danger of an infinite loop developing in the path through returning to the point. If leaving takes place at such a point, the obstacle is then only a temporary diversion. As the travel between obstacles always reduces the distance to the goal and the number and size of obstacles are finite, reaching a realisable goal is guaranteed. There are many BUG algorithms, each with different conditions for leaving an obstacle safely enough to preserve the guarantee, but all have the above basic modus operandi. Our own method has its own improvements for leaving safely.

The BUG algorithms also have their drawbacks though. The earlier versions were relatively inefficient in the degree to which they went round obstacles before leaving them. Subsequent variants attempt to reduce the inefficiency of their predecessors' paths so they leave an obstacle earlier, e.g. [11]. More recently DistBUG [12] and 3D Bug [13] feature a more prevalent use of real sensor data.

However, a central issue is that while providing a theoretically sound geometric path for a point robot, they require extra support, that is unspecified in detail, to realise their paths. There is no specification in BUG's path planner for how to follow an obstacle edge or how to cope with a finite sized robot or with perturbation of the robot's perception and action. In robotics, it is not trivial to provide such support given the uncertain nature of a robot's environment and idiosyncrasies of the robot itself. BUG paths are also inherently inefficient by closely clinging to the obstacle boundaries. There is no opportunity to curve a smoother course to go round obstacles at a greater distance from the obstacles, in the way natural agents with extended sensing such as vision can do.

In this paper, we aim to provide the best of both approaches, i.e. to provide robust and smooth paths through a potential field approach that has a more integral specification than BUG, but also to provide BUG-like guarantees of reaching realisable goals.

2 Previous Work on Overcoming Local Minima Using Potential Fields

The local minimum problem has been a serious problem for potential field methods. While there have been attempts to overcome this problem in a variety of ways [6], [7], [14], [15], none of the attempts have been seen to be an outright solution. Obstacle avoidance on its own using potential fields does not guarantee convergence to the goal. Franceschini et al [16] show that robots become stuck in local minima or wander aimlessly without a target. As described earlier, the use of a target induces local minima in common convex shapes for a static field. Varying a potential field dynamically during the behaviour may move the process on from local minima in a static field [17], but then introduces cyclic paths for various common convex obstacle shapes due to being attracted back onto a previously visited obstacle edge. BUG algorithms provide theoretical guarantees that obstacle edges are not returned to as described above. A potential field approach thus needs to provide more practical guarantees equivalent to BUG if success is to be assured on a universal basis.

Forward chaining is a technique recently developed by one of the authors [2] that aims to provide smooth plasticity and persistence for robotic agent navigation towards a goal. It is a relatively inexpensive dynamic potential field based approach capable in principle of traversing static local minima in combined obstacle and goal potential. Smooth adaptation of the robot's travel path while maintaining persistence towards a goal is provided by the use of intermediate subgoal attractors that dynamically form temporary stepping stones connecting to the goal. Other work which has made use of subgoals in order to address the local minimum problem for potential fields can be found in [18], [19], [20] and more recently in robotics in [21].

A subgoal as an individual acts as an attractor on a combined subgoal and obstacle potential surface just like the goal acts as an attractor on the combined goal and obstacle potential surface in traditional static potential field methods. The difference is to repeatedly replace the subgoal with one further forwards to generate a simple and dynamically changing potential surface. The dynamics carries the robot position within a moving *dip* that corresponds to the moving attractor basin at each stage of the process. That is, each new subgoal generates a different potential surface containing a single global minimum forwards of the previous location that is readily approachable through gradient descent from the current robot position. A suitable sequence of subgoals, set to track the goal and nearby obstacle edges in a forward direction as they are sensed, allows the robot to travel from start to goal without getting stuck in static local minima short of the goal.

Forward chaining has been shown in [2] to reach the goal successfully for a variety of obstacle courses including local minima for the traditional static method. However, the design was limited in being for a point robot, in its obstacle leaving condition restricting the type of obstacle course navigable, and by not using direct or perturbable sensor information. We now cater for these features in the present paper.

3 A Mind's Eye Approach to Providing a Goal Reachability Guarantee Using Dynamic Potential Fields

In this section we present POTBUG and POTSMOOTH flavours for Forward Chaining robots that have evolved to remove the limitations outlined above, and extend the goal reachability guarantee to arbitrary obstacle shapes by chaining subgoals along BUG-like paths.

3.1 Sensor and Subgoal Based Potential Evaluation

In the following sections, range sensors are used to repeatedly establish obstacle distances and directions relative to the robot throughout the behaviour. The direction to the goal is also repeatedly established through sensor readings and is not blocked by obstacles at any time. Repeated estimates of the distance to the goal are established by triangulation.

The sensor readings are used to construct a varying potential field relative to the robot's current position and orientation. The obstacle potential function has an infinite value on the edge of an obstacle and falls off to a value of 0 over a finite *fall-off* range from the obstacle. The obstacle potential function is given by

$$U_{\text{Obstacle}}(d_i) = \begin{cases} 1/d_i \cdot e^{-1/(s^2-d_i^2)} & , 0 \le d_i \le s \\ 0 & , d_i > s \end{cases} \quad (1)$$

where d_i is the sensed distance of the robot from an edge point i of an obstacle, and s is the *fall-off* range which is generally set to be 3r, where r is the robot radius. [For the given fall-off function to have a monotonic decrease in gradient size, $s \le 1$. Consequently, as s = 3r, the robot radius needs to be expressed in terms of units that result in such a fall-off range.] Setting the fall-off range in terms of the robot radius will be shown to be important for making the robot navigation suitable for its size.

Sensor readings of obstacle distances d_i are taken in various forwards linear directions that impact on obstacles within sensor range. The potential function used to model subgoals as attractors for the robot is a quadratic bowl in terms of the distance d_a to that attractor

$$U_{Attractor}(d_a) = d_a^{\,2} \quad (2)$$

The combined obstacle and attractor potential may then be calculated as:

$$U_{Combined} = U_{Attractor}(d_a) + \sum_i U_{Obstacles}(d_i) \quad (3)$$

The robot uses the equivalent of a mind's eye to focus attention on a circle set concentrically around itself with a radius that is the distance of the nearest sensed obstacle point, or the sensor range itself if no obstacles are sensed, or the goal if this is within sensor range with no obstacles inbetween. The mind's eye is used by the robot to monitor and position the subgoal on the forwards semi-circle either relative to the

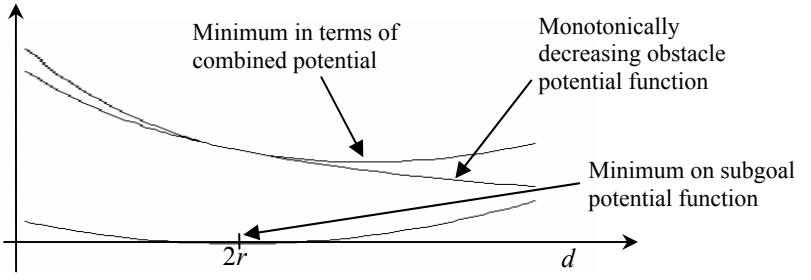

Fig. 1. Summation of obstacle potential and subgoal potential to form combined potential. The distance *d* from the obstacle increases from left to right.

goal or to any intervening obstacles. The robot performs steepest descent down the dip of the combined obstacle and subgoal potential surface at each stage.

3.2 Subgoal Setting and Reachability Guarantee

Each subgoal is set forward in free space using the sensors to ensure there are no obstacles between the current robot position and the subgoal. In the absence of any sensed obstructions, subgoals are set on the forward semi-circle in the goal direction so as to pull the robot along a straight line to the goal. In the presence of sensed obstructions, subgoals are set on the forward semi circle at a safe yet close distance from the obstacle edge to pull the robot around the obstacle. By selecting positions that have a summed obstacle potential value of $B=U_{Obst}(2r)$, where r is the robot radius, we effectively set a safe distance corresponding to $2r$ from a single obstacle sensor sample, and slightly more for multiple sensed obstacle points. The robot uses its mind's eye to set subgoals targeting an obstacle potential contour of this B-value, a *B-contour*, surrounding obstacles.

For each subgoal the combined subgoal and obstacle potential surface is the summation of monotonically changing functions and a quadratic bowl. Starting at an obstacle edge, the combined potential decreases monotonically until the minimum of the subgoal function is reached (Fig. 1). After this point, the obstacle function gradients continue to decrease monotonically in size towards zero while the subgoal function gradient now becomes positive and increases monotonically. The combined potential will therefore eventually start to turn and rise (monotonically). A single global minimum attractor on the combined surface occurs when the subgoal's positive function gradient matches the obstacle functions' negative total gradient, see Fig. 1. This always occurs between $2r$ and the edge of the fall-off range, $3r$, from a single obstacle sample point. Consequently the simple nature of the obstacle and subgoal surfaces ensure that monotonic descent towards the minimum is always possible with the global minimum basin containing the current robot position at each stage.

3.3 Subgoal Chaining Around Obstacles

The targeting of a B-contour will set a subgoal steadily to the left or right of the current robot trajectory when approaching an obstacle to make the robot turn to fall

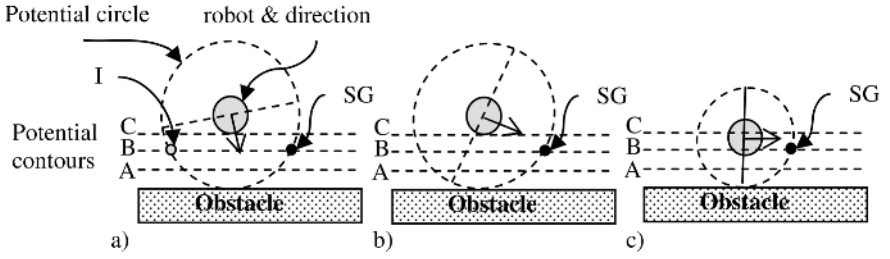

Fig. 2. (a) A subgoal setting SG on the forward half of potential circle (forward semi-circle) at the most forward intersection with a potential value of B (compare with I). (b) The robot body (inner circle) turns when targeting SG for subsequent travel towards the minimum on subgoal and obstacle surface. (c) Continued movement towards successive subgoals set on the B-contour pulls robot centre into and along the BC-band of obstacle potential.

into line along the obstacle edge as depicted in Fig. 2. The most forward B-contour intersection with the forwards semi-circle is chosen as the subgoal. In the pathological case when multiple intersections are exactly equally forward, one of them is chosen at random. Subgoals are continually set and replaced forwards of the robot along the B-contour of potential thus pulling the robot forwards towards the minimum on the combined subgoal and obstacle potential surface. This minimum lies in a BC-band of potential where $C=U_{Obst}(3r)$ as described earlier. The A-contour is associated with a potential value of $A=U_{Obst}(r)$ and represents a boundary of safety which is not crossed by the edge robot during travel.

We deem passages between obstacles to be too narrow for safe navigation when there is significantly less than a robot width either side of the robot. The setting of the obstacle potential *fall-off* range to $3r$ is sufficient to achieve occlusion of such passages without losing the ability to navigate other passages. Occlusion will occur for some passages that are less than $6r$, and all those less than $4r$, wide when the *B*-contour flows past the passage. The sets of obstacles surrounded by a BC-band can be treated as though they are just one virtual obstacle since the passages between them are unsafe to navigate.

The safe leaving of a BC-band to go to the goal or another obstacle requires satisfaction of the condition of being clear to leave with no immediate obstruction in the goal direction. If a BC-band has had to be followed to any extent, it is also required that finite progress has been made towards the goal since the first point on the band, the *hit point*, was reached by the robot. The progress ensures that in theory each traversed part of a BC-band is left permanently to go to another one nearer the goal. In this way, the chain connects to the goal with a BUG-like guarantee.

3.4 Modes of Operation

The robots chain their subgoals to reach the goal through three main modes of operation, two of which are much like those found in BUG. The two similar modes are *free* which entails moving directly towards the goal when there are no obstructions in the way, and *engaged*, when having to circumnavigate an obstacle to any extent until *safe to leave*. The third mode is a transitional *approach* mode following *free* mode.

Once an obstacle has been detected and deemed to be an obstruction on the way to the goal, the robot enters the *approach* mode. In this mode, it approaches the *B*-contour surrounding the obstacle to establish a hit point in the BC-band, turning while it does so. When the robot achieves this hit point, if it is then at a corner of the obstacle edge and is clear to leave immediately towards the goal it does so. Otherwise it enters the *engaged* mode in which it follows the BC-band further round the obstacle boundaries. This carries on until it is deemed safe to leave the BC-band and recommence travel towards the goal again in *free* mode. The process terminates at any stage if the robot centre comes within $2r$ of the goal where r is the robot's radius.

3.5 Travel Algorithm

The top level algorithm may be described as follows:

1. Initialise
   1.1 set the sensor range and the radius of the robot
2. **while** (not reached goal)
   2.1 scan for goal, rotate robot to face goal, scan for obstacles
   2.2 **while** (no *obstruction* in goal direction on forwards semi-circle arc of width $2r$ and not reached goal, *free mode*)
   2.2.1 make a move (towards goal)
   end while
   2.3 **while** (hit point not flagged and not reached goal, *approach mode*)
   2.3.1 if the robot's position w.r.t. obstacle is deemed a hit point,
   then flag hit point reached and record robot's current distance to goal as hit point distance
   else make a move (towards B-contour)
   2.3.2 if (hit point flagged and not reached goal and clear to leave obstacle)
   then safe to leave obstacle, make a move (towards goal)
   end while
   2.4 **while** (not *safe to leave* obstacle and not reached goal, *engaged mode*)
   2.4.1 make a move (along B-contour)
   2.4.2 if (not reached goal and progress attained and clear to leave obstacle)
   then safe to leave obstacle
   end while
   end while

make a move procedure

1. set the radius of the forwards semi-circle to minimum of (sensed distance to obstacles, sensor range, distance to goal)
2. set subgoal on semi-circle towards goal or towards or along B-contour
3. make a step down the subgoal obstacle potential surface through robot motion
4. test for goal reached

3.6 POTBUG and POTSMOOTH

The POTBUG robot has a near-sighted mind's eye due to the sensor range being 4*r*. In the POTSMOOTH robot, the only difference is that the sensor range is extended. With a more far-sighted mind's eye, the robot is able to detect obstacles at a greater distance and turn earlier. The earlier turning makes for a smoother path around the obstacle with less of an abrupt turn near to the obstacle. This generates paths qualitatively different to straight BUG-like paths in their approach to obstacles. For both flavours, the algorithm's targeting of the desired B-contour ensures that, while turning, the robot continues to move closer to the desired sensed obstacle edge to register a hit point.

3.7 Example Run

Fig. 3. shows the POTBUG and POTSMOOTH robots on an obstacle course containing a variety of shapes that present local minima for static potential field methods. The local minima are overcome by the robots and the goal is reached.

The jagged edged rectangle demonstrates how, in contrast to BUG, summation of potential is able to smooth the robot's perception of an obstacle edge to allow for a smooth plastic path around an obstacle and persistent travel further towards the goal. The POTSMOOTH robot has a less BUG-like travel path due to turning earlier than POTBUG using its extended sensor range.

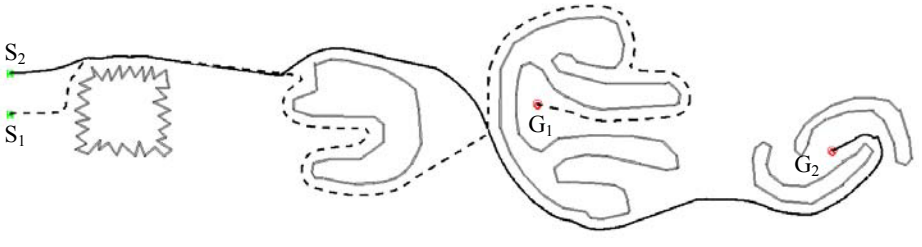

Fig. 3. POTBUG (dashed path from start S_1 to goal G_1) and POTSMOOTH (continuous path from start S_2 to goal G_2) on a difficult obstacle course with multiple local minima and a jagged edged rectangle

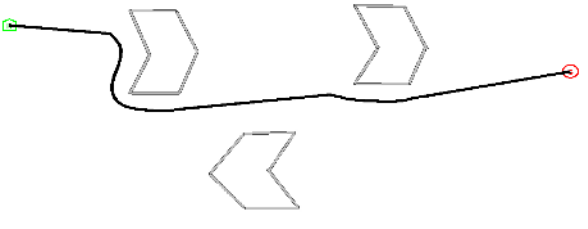

Fig. 4. Test arena for simulation experiments containing 3 chevron shapes with a typical POTBUG path (without sensor perturbation)

4 Simulation Experiments

In the following we present simulation experiments conducted on POTBUG and POTSMOOTH robots. The test arena for the experiments is an obstacle course containing 3 chevron-shaped obstacles (see Fig. 4). All the above tests are run on a set of 241 pairs of initial and goal positions thoroughly distributed throughout the test arena and with intervening obstacles. The robot radius and the sensor range are the only variables used to parameterise the robot, besides the degrees of perturbation experimented with. All robots have 30 sensors.

Table 1. Average ratio of unperturbed to perturbed path curvature (RPC) and length (RPL)

		gp											
	%	**0**		**5**		**10**		**20**		**30**		**40**	
	0	1.00	1.00	2.30	1.01	3.66	1.01	6.88	1.08	10.92	1.17	16.41	1.35
	5	1.07	1.00	2.43	1.02	3.73	1.01	7.03	1.08	10.93	1.19	16.43	1.37
op	**10**	1.21	1.00	2.51	1.01	3.84	1.02	7.22	1.09	10.90	1.16	16.20	1.35
	20	1.67	1.01	2.94	1.01	4.31	1.03	7.60	1.09	11.44	1.18	17.11	1.35
	30	2.18	1.02	3.34	1.03	4.65	1.04	7.81	1.07	11.79	1.19	17.65	1.37
	40	2.75	1.03	3.87	1.04	5.08	1.04	8.26	1.08	12.57	1.21	18.78	1.45
		RPC	**RPL**	**RPC**	**RPL**	**RPC**	**RPL**	**RPC**	**RPL**	**RPC**	**RPL**	**RPC**	**RPL**

4.1 Robust Smoothness (POTBUG)

Robustness here consists of smoothly maintaining plasticity and persistence of the travel path in the face of unreliable sensor readings. BUG on its own has no integrated method for addressing this issue. By contrast, adaptation of robot motion through the corrective potential field gradients allows the POTBUG robot to smoothly get back on course if the intended path is not realised at any stage.

The extent of successful adaptation possible may be tested through introducing perturbation repeatedly into the sensed distance to any detected object and sensed direction to the goal. The experimental aim is to show the extent to which relatively inaccurately sensed obstacles and goals leads to failures or departures from the path produced with relatively accurate sensing. Such robust smoothness may be quantified by comparing the ratio of overall curvature (RPC) and path length (RPL) of a POTBUG forward chaining path under various degrees of perturbation to the unperturbed counterparts for the tested pairs of initial and goal positions. The sensor range is small ($4r$).

A random percentage of the sensed obstacle distance and the sensed goal direction and distance is added to or subtracted from unperturbed sensor readings within a maximum given by each value of *op* and *gp* respectively in Table 1. In the unperturbed case there were no failures. In the perturbed cases, there was at most 1% failure. Table 1 shows that there is a broad trend of RPC and RPL increasing with increasing perturbation. RPL increases by a relatively much smaller amount compared to RPC for up to 40% random perturbation in both individual obstacle and goal sensors. This is in line with our visual inspection of typical travel paths that

shows a slightly drunken walk around, but close to, the true goal directions and B-C bands. There is considerable individual fluctuation about the mean curvature in the perturbed cases, most probably due to each transition moving the robot from side to side. The standard deviations are on average 86% of the mean. The path length for individual cases is again much less variable with standard deviations that are on average 13% of the mean.

Beyond the 40% random perturbation shown in Table 1, the goal is still achieved with the path length and curvature continuing to increase relative to the unperturbed case. The high degree of success in attaining the goal in the face of perturbation is quite possibly due to the recalibrating nature of the potential used. More specifically, the random effects of the goal and obstacle sensor perturbation tend to cancel out with repeated triangulation and polling of multiple obstacle sensors respectively.

4.2 Extended Smoothness (POTSMOOTH)

The empirical aim in this section is to investigate the extent to which the undulatory smoothness of travel paths may be enhanced by extending the sensor range. We extend the range from $4r$ to $12r$ in 4 steps of $2r$ to try out various types of POTSMOOTH robot and evaluate the ratios of path length and curvature for each range relative to those for the $4r$ counterpart set the same initial and goal positions. The sensors are unperturbed for this experiment.

Table 2 shows the path length and curvature decreasing as the sensor range is extended. There were no failures. This is consistent with our visual inspection of typical travel paths that shows the travel path curving earlier but still reliably joining the B-contour further to the right or left of the last goal direction taken in the *free* travel mode. There is consequently much less time spent in closely following the obstacle boundaries as was shown in Fig 3.

Table 2. Average ratio of path curvature (RPC) and path length (RPL) for sensor range (SR) compared to path curvature and path length for sensor range $4r$

SR	*4r*	*6r*	*8r*	*10r*	*12r*
RPC	1.000	0.975	0.955	0.946	0.932
RPL	1.000	0.995	0.992	0.989	0.986

5 Conclusion

A dynamic potential field method has been shown by design and empirically to achieve effective guarantees for goal realisability in the face of intervening obstacles. This was done by setting achievable and continually replaced subgoal attractors in the robot mind's eye as targets and descending towards them using potential field gradients. BUG-like travel modes were found to guarantee dynamically emerging connections of the subgoal chains to the goal based solely on significantly perturbable sensor readings without the aid of a global map or other prior knowledge.

The POTBUG and POTSMOOTH flavours of robot were developed with small and extended sensor ranges respectively. POTBUG's adaptive resilience to unreliable sensor readings and POTSMOOTH's adaptive smoothing of the global travel path have been

empirically tested in simulations and found to be reliable to significant degrees. The initial results are promising for offering a relatively seamless integration with real world robotics and the implementation of the algorithms in physical robots is currently underway.

The authors thank G. W. Lucas for support through the Rossum Project Simulator.

References

1. Russell S. J., Norvig P., Artificial Intelligence, a Modern Approach, Prentice Hall. 2002.
2. Bell, G., Weir, M., Forward chaining for robot and agent navigation using potential fields, Twenty-seventh Australasian Computer Science Conference (ACSC2004), Vol 26, 2004.
3. Lumelsky, V. & Stepanov, A., Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, Algorithmica, 2(4):403-440, 1987.
4. Lumelsky, V., Mukhopadhyay, S. & Sun, K., Dynamic path planning in sensor-based terrain acquisition. IEEE Trans. Robotics and Automation, 6(4):462-472, 1990.
5. Rao, N.S.V. Kareti, S. Shi, W. and Iyenagar, S.S. Robot Navigation in Unknown Terrains: Introductory Survey of Non-Heuristic Algorithms. Oak Ridge National Laboratory Technical Report, ORNL/TM-12410:1--58, July 1993.
6. Latombe, J., Robot Motion Planning, Kluyver Academic Publishers, 1991.
7. Dudek, G., Jenkin, M., Computational Principles of Mobile Robotics, Cambridge University Press, 2000.
8. Khatib, O., Real-time obstacle avoidance for manipulators and mobile robots, The International Journal of Robotics Research, 5(1), 1986.
9. Arkin, R., Behavior-based robotics, MIT Press, 1998.
10. Koren, Y., Borenstein, J., Potential field methods and their inherent limitation for mobile robot navigation, IEEE Conference on Robotics and Automation, pp. 1398-1404, 1991.
11. V.J.Lumelsky and T.Skewis, Incorporating range sensing in the robot navigation function, IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 5, pp. 1058-1068, 1990.
12. Kamon, I., Rivlin, E., Sensory-based motion planning with global proofs, IEEE Transactions on Robotics and Automation, 13(6), 1997.
13. Kamon, I., Rivlin, E., Range-sensor-based navigation in three-dimensional polyhedral environments, The International Journal of Robotics Research, 20(1), 2001.
14. Koditschek, D., Exact robot navigation by means of potential functions: Some topological consideration, IEEE Conference on Robotics and Automation, pp. 1-6, 1987.
15. Alvarez, D., Online motion planning using Laplace potential fields, IEEE International Conference on Robotics and Automation, pp. 3347-3352, 2003.
16. 16.Franceschini, N., Pichon, J. M., and Blanes, C., From insect vision to robot vision, Philosophical Transactions of the Royal Society B, 337, 283-294, 1992.
17. 17.Huang, W.H., Fajen, B.R., Fink, J.R., and Warren, W.H., Visual navigation and obstacle avoidance using a steering potential function, Robotics and Autonomous Systems, 54(4), 288-299, 2006.
18. Gorse, D. et al., The new era in supervised learning, Neural Networks, 10(2):343-352, 1987.
19. Lewis, J., Weir M., Subgoal chaining and the local minimum problem, IEEE International Joint Conference on Neural Networks, 1999.
20. Lewis, J., Weir M., Using subgoal chaining to address the local minimum problem, International ICSC Symposium on Neural Computation, 2000.
21. Xi-yong, Z., Jing, Z., Virtual local target method for avoiding local minimum in potential fields based navigation, Journal of Zhejiang University SCIENCE, 4(3):264-269, 2002.