2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

High Quality Goal Connection For Nonholonomic Obstacle Navigation Allowing For Drift Using Dynamic Potential Fields

Michael K. Weir and Matthew P. Bott

Abstract— **The problem we address in this paper is how to plan and execute high quality paths for robots subject to nonholonomic constraints while navigating obstacles in 2D space. The navigation is to be carried out continuously at speed and may be subject to drift that is not predictable a priori. The problem raises the challenge of adaptively maintaining a smooth robust path of low computational cost. The algorithm is complete in providing feasible paths connecting to the goal in cluttered environments without global maps or positioning while also optimising the path curvature in free space. The approach is a generic gradient-based methodology set in dynamic potential fields that are not subject to fixed local minima or other misdirecting surface features of static fields. Multiple planning and execution cycles are interleaved to allow frequent updates for dealing with unanticipated obstacles and drift. We present our methodology and demonstrate experimental results for simulated robots. The results show that low curvature paths are found that robustly connect to the goal under perturbation through a sequence of fast adaptive replanning.**

I. INTRODUCTION

A. Problem and Hypothesis

Many tasks such as navigation under tight constraints in space and time require a high quality goal-connecting path as well as simply reaching a desired configuration. The contribution of this paper is to provide a tested generic methodology for high quality solutions to nonholonomic obstacle navigation problems. High quality here means smooth, robust, and relatively fast to replan. Also, although this is yet to be tested, the methodology may be capable of extension to other tasks requiring high quality solution.

The problem we address in this paper is how to plan and execute these high quality paths for robots subject to nonholonomic constraints while navigating obstacles in 2D space. The navigation is to be complete and carried out continuously at speed without the use of global obstacle maps or positioning. It may also be subject to drift that is not predictable a priori.

Many planners opt for a shortest path to reduce the time taken [1]. We instead opt for a smoother low realizable curvature solution that is less distance-greedy. Such a solution is still reasonably short while also promoting more even distribution of minimal curvature that is useful for both quick replanning and traveling at speed.

The hypothesis is that low realisable curvature paths can be found that robustly connect to the goal under perturbation through a sequence of fast adaptive replanning.

School of Computer Science, University of St Andrews
St Andrews Fife Scotland{mkw, mb1}@cs.st-andrews.ac.uk

B. Multi-step and Single-Step Planning

Holonomic constraints are constraints on the robot's realisable positions, of which obstacles are examples. Nonholonomic constraints are non-positional constraints on the paths in a given configuration space for a particular robot. They are derivative constraints that cannot be integrated out, i.e. turned into holonomic constraints [2].

The presence of nonholonomic constraints means that only some differentially smooth geometric goal-connecting paths are realisable and knowing the set which is realisable does not have a priori solution. Our approach, which gives dynamic solutions, handles all robots whose only limitation is a minimum turning circle. An example will be provided for individual differential drive and car-like robots navigating from and to (x,y,ϕ) states where x and y are cartesian planar coordinates and ϕ is the robot's horizontal orientation in (x,y). These wheeled robots are well known to have nonholonomic constraints on the differentially smooth paths that are realisable between end poses in (x,y,ϕ) space, though not in (x,y) (except for turning circle considerations). For example, they cannot move in many linear directions in (x,y,ϕ) space, even though they can in (x,y) space.

When nonholonomic (NH) constraints restrict the paths as just described, an M-step planner (where step = arc), $M > 1$, may be needed to ensure constraint satisfaction by the whole path [3]. A 1-step planner may suffice in other cases [4].

Designing planners of high quality paths, especially generic ones coping with drift, is a substantial part of the major challenge presented by nonholonomic motion planning (NMP) over the last two decades [2,3,5].

C. Smooth Nonholonomic Motion Planning

In common with many approaches [3,6,7] we will use obstacle potential to promote smooth travel around obstacles. In planning a goal-connecting path, we will plan a path that assumes free space lies ahead unless a sensed obstacle across the path proves otherwise. In addition we will optimize the curvature of the path while meeting NH constraints.

Smoothness in the sense of optimal low curvature may be defined as minimal path *strain*, i.e. $Min\left(\int \sigma^2 dp\right)$ where σ is the curvature along the path *p* [8], which is a property of the path *as a whole*. Splines provide minimal path strain for a given collection of spline knots [8]. We go one step further and dynamically optimize the planned positions of the knots in line with NH constraints during travel so the low curvature path is nonholonomically realizable (Fig. 1).

There are approaches that enable nonholonomic robots to plan whole paths through decoupling control from geometry to more easily find a geometric path first and corresponding controls afterwards e.g. [9]. The decoupling approach may be seen as inherently limited in suitability for a generic approach though when trying to meet NH constraints. This is because, as described above, not all paths may be realizable, so the geometric path found first may possibly not be realizable.

Consequently, planning realizable whole paths under NH constraint benefits from attention to control and geometry simultaneously. Our methodology plans a whole M-step control sequence that connects a current configuration state to the goal configuration state through appropriate forward kinematics to meet the NH constraints for the current robot. This is not to say that a perfect forwards kinematics and control sequence has to be assumed. Imperfection resulting in drift is catered for here through feasible replanning. Earlier work has laid out the symbolic detail for how this may be done for travel solely in free space [10].

D. *Obstacle Navigation*

Obstacle navigation without global knowledge of obstacle shapes and locations presents a general challenge for guaranteeing being able to find high quality paths that connect to the goal configuration state. This is to be done armed with sensors providing only goal data relative to the robot, actuator movement, and local obstacle distance data.

Obstacle navigation poses problems for NMP in particular, as NMP may rely on correct sequences of controls to attain the goal and obstacles may disrupt trajectories planned a priori. Sensor error or other causes leading to unpredictable drift are further sources of irregular perturbation that are difficult for NMP to handle [3].

A generic nonholonomic planning system ought to be able to provide M-step as well as 1-step plans. In our examples, we will view planning that assumes free space as solving an M-step travel problem in (x,y,ϕ). Obstacles will be viewed as holonomic (x,y) constraints that temporarily interrupt the M-step solution and for which a 1-step planner suffices to attain various (x,y) positions that safely and smoothly avoid collision while rounding the obstacle. The 1-step planner uses the same forward kinematics as the M-step planner to ensure the NH constraints are always met.

E. *Related Approaches*

There are existing nonholonomic approaches for obstacle navigation that offer some of the properties needed for quick low curvature travel. These include a version of Tanbug using splines in free space [6], Euclidean minimization [3], Rapidly Exploring Random Trees (RRTs) [9,11], and Path deformation [12]. One major issue is that the related approaches each possess at least one different feature that undermines generic fast low curvature goal connection.

Tanbug using free space splines for example creates target paths of relatively low curvature. However, the associated controls are for (x,y) paths only and as such are not capable of generic extension to other spaces such as (x,y,ϕ).

Euclidean minimization provides generic nonholonomic goal connection in free space. However, if it is set in a static potential field, it is subject to fixed local minima and gets stuck when navigating a variety of obstacle shapes, and so is not complete. The path shapes are also only designed to greedily shorten Euclidean distance to the goal. Consequently they can create significant unnecessary curvature, including loops [10].

RRTs provide complete as well as generic goal connection given sufficient nodes [11], and can be enhanced to optimize dynamically and remove at least some of the initial unnecessary curvature [9]. However, the random nature of path creation and modification means they either provide an unoptimized path relatively quickly or a low curvature path relatively slowly.

Lamiraux's path deformation technique is similar to ours in using dynamic optimization through potential fields and sets paths that follow the smooth obstacle potential contours to enable low curvature navigation around obstacle boundaries. However, it presently lacks a number of key features for our purposes. One is that the technique is incomplete in the sense of not making goal connection generically and through sensor-based deformation alone. The internal parameters have to be tuned differently to make the method work in different situations [13]. It relies on a global map both initially and when the path deformation is insufficient to remove collisions [13,14]. A second is that there is no design for maintaining low curvature across the whole path. A third is that the overall time for basic deformation in the replanning process is relatively slow (over 10 times slower than our method on an equivalent machine) with the travel also stopping at times [14].

F. *Our Approach*

Our own approach extracts four key features of genericity, completeness, overall low curvature, and relatively quick replanning from the above approaches and sets them in *dynamic potential fields*, i.e. fields with associated travel surfaces and attractors that vary during the behaviour. The key properties provided and not provided by the various approaches are summarized in Table I.

The travel needs to be locally optimal at each stage and yet also has to connect to the goal. We also need to replan quickly to counter drift within space-time constraints. Consequently we use a dynamic attractor in the form of a mobile local minimum moved along an adaptive target path to reach the goal.

TABLE I
TABLE OF METHODS AND PROPERTIES

Planning Method	G	C	LC	Q
Tanbug With Free space Splines	×	√	√	√
Euclidean Minimisation	√	×	×	√
Quick RRT	√	√	×	√
Smooth RRT	√	√	√	×
Path Deformation	√	×	×	×
Dynamic Fields	√	√	√	√

G: genericity. C: completeness; LC: low curvature; Q: quick replanning

II. Methodology

A. Dynamic field method

In this paper, we develop a method for nonholonomic robots that generalises previously separate approaches for nonholonomic robots travelling in free space [10] and holonomic robots travelling around obstacles [7].

Two planners are derived from the approaches. One is an M-step (x,y,ϕ) path planner that is updated continuously but only executed in free space. Its execution is interrupted by a second 1-step (x,y) path planner while rounding obstacles and then leaving to join an M-step path.

Robot localisation relative to the goal is achieved through triangulation using sensors able to detect distance and angle to goal and actuator movement.

B. M-step planning

The M-step planner needs to find a path of minimal strain that connects to the goal as described earlier. To begin the design, an M-step path *without* strain (or drift) considerations can be found using a static potential field [3]. That is, planning a sequence of M controls and V control variables that ends in the goal configuration may be done using the control-configuration mapping provided by the forwards kinematics to enable descent in potential over control space $<C_{11}, \ldots C_{MV}>$ where potential is a goal disconnection penalty such as Euclidean distance to goal. (For the example solutions, M is initially set to 3 and V is 2.) The use of the forward kinematics in the mapping ensures the robot's NH constraints are obeyed.

Simply then adding strain to the potential in a static field to also minimize strain leaves the global minimum short of the goal if the penalty is set too low relative to the strain or makes the global minimum too difficult to access from a random initial state if it is set too high due to distorting the travel surface [10].

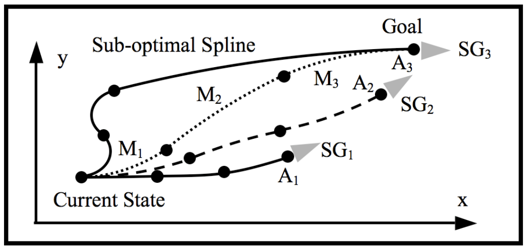

Fig. 1. Subgoals SG_i in (x,y,ϕ) are moved during planning to instigate attempts A_i that stretch and reshape the configuration path resulting from controls M_1-M_3 towards the goal (x,y,ϕ) state (SG_3). The resulting path has a locally optimal curvature spline relative to other paths in the vicinity.

Consequently instead of this, the algorithm starts with a subgoal that is a nearby local minimum using a low penalty and then moves the subgoal to bend the path towards the goal (Fig. 1) by gradually increasing the penalty. In this way the robot is able to plan a goal-connecting path with locally optimized curvature. Locally optimal means no unnecessary undulations, kinks, or loops relative to other realizable paths in the vicinity.

The subgoal acts as a moving nearby configuration attractor and moves from near the initial configuration in a series of dynamically evolved subgoals that end in the goal configuration. At each stage, the subgoal is set in a new locally accessible position further towards the goal. This method enables the subgoal to generate a moving dip in the dynamic potential field's travel surface (Fig. 2) that carries the configuration and control sequence to the goal.

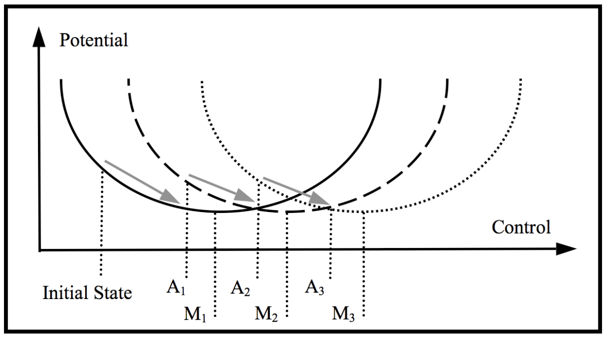

Fig. 2. Minima M_i in control corresponding to configuration subgoals SG_i are moved forwards after each attempt A_i to create a forwards moving dip in potential.

In detail, an (x,y,ϕ) spline is initially placed through the configuration states corresponding to the beginning and end of each of M small random controls. The spline shape is evaluated for its potential including strain and other elements in (1). Descent takes place to change the controls and hence the spline's knot configurations. Descent proceeds until a configuration path is found obeying the NH constraints whose spline has locally optimised low realisable curvature relative to those of other splines in the vicinity. The resulting end state is short of the goal such as A_1 in Fig. 1. The path is then stretched and reshaped by descending towards a new nearby subgoal end state in (x,y,ϕ) that both decreases the gap with the goal and maintains locally optimal curvature for the extended curve. This is done using a potential $U_{Attractor}$ for an attractor configuration subgoal α of

$$U_{Attractor}(\sigma_\alpha) = \sum_{i=1}^{n} \int \sigma_i^2 \, dp + \gamma_\alpha + \mathrm{B}(c) \qquad (1)$$

where n is the number of configuration variables and $\int \sigma_i^2 \, dp$ is the strain of the i^th of n component cubic splines of the variable n-D spline curve p. (For the example solutions, n is 3.) The attractor subgoal α is set through the non-negative term γ_α which increasingly penalizes disconnection between the M^{th} attained state and the goal state. The value of γ_α is automatically increased gradually for each non-zero degree of disconnection throughout plan formation to keep the subgoal moving to the goal as described above. (For the earlier Euclidean distance example, a scaling factor for the distance may be increased to increase γ_α.)

B is a boundary condition on configuration states c to prevent any invalid solution configuration paths associated with the M controls from being considered, e.g. those containing backward motion or curvatures greater than that of the minimum turning circle. B is composed of potential fall-off functions in control space that fall away to zero inside valid regions of configuration space. This enables the final planned solution path to have locally minimal strain that is realizable. See [10] for more details.

The process is iterated until a plan emerges for a whole M-step goal-connecting path as shown for an (x,y,ϕ) goal state in Fig. 1. If the path is completely clear of obstacles, the M-step plan's 1st control is executed. Replanning occurs during intervals allocated to each control to keep the plan up to date regardless of whether it is executed or not. This is to take account of events that have occurred to undermine the plan such as drift or having to go round obstacles, and also to further optimise smoothness. It is relatively quick to replan due to similarity between successive smooth plans. Obstacle rounding is done using a separate holonomic planner described below.

The top level algorithm for planning and executing an M-step goal connecting path *P* with inputs of goal direction and distance, sensed obstacle edges and actuator motion is:

1. Initialize a sequence of *M* controls.
2. Initialize a corresponding n-D configuration spline.
3. Compute the initial cost gradients of the configuration spline from the initial configuration to the goal configuration – for an initially low penalty for the forward end being disconnected from the goal.
4. Descend through control until descent slows to indicate closeness to the current subgoal of local minimum cost potential.
5. Increase the penalty for goal disconnection to move the subgoal towards the goal and then descend again.
6. Repeat 4,5 until a locally optimized control sequence is planned to reach the goal configuration within 1% of the initial distance to the goal.
7. If in free space then execute the plan's 1st control.
8. Re-plan 1 to 6 at regular intervals.
9. Repeat 7,8 until the goal configuration is reached within the tolerance described in 6.

If the number of controls proves insufficient for keeping the trajectory flexed and descent slows despite γ_α increasing then *M* is automatically increased until there is sufficient flexibility [10].

C. 1-step planning

As alluded to above, when obstacles are sensed to lie across the planned path, a 1-step path planner interrupts execution of the M-step planner controls and temporarily takes over execution. The interrupting system also uses descent over a dynamic field but with changes to the subgoal, M value and potential. The mobile local attractor subgoal is changed to be a target configuration in (x,y) set continually ahead of the current location clear of obstacle collision. The subgoal travel follows a BUG algorithm set in dynamic potential fields [7]. That is, the subgoal and executed attempt on it *approach* each obstacle and then *engage* at a *hit point* on a contour of obstacle potential to round the obstacle along the contour. Once progress has been made, and can be made, towards the goal in free space, the obstacle is *left*. M is changed to 1 and the potential is changed so that single step controls are planned for moving to the single mobile (x,y) subgoal using a Euclidean minimization technique based on a fixed fall-off function for obstacle potential and Euclidean distance.

This function has an infinite value on the edge of an obstacle and falls off to a value of 0 over a finite range from the obstacle. The fall-off function *U* is given by

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

where d_i is the sensed distance of a point from an edge point *i* of an obstacle, and *s* is the fall-off range. The total fall-off at the point is summed over all sensed edge points and the parameter *s* is set to be Max(r, 3w) where r is the robot's minimum turning circle radius, and w is robot width. Passages narrower than $2s + 2r$ are occluded using local sensor information so that U-turns in passages can be safely made if required. These features enable navigation plans around obstacles and through passages to suit the robot.

The Euclidean distance potential is given by

$$U_{Attractor}(d_\alpha) = d_\alpha^2 \qquad (3)$$

where d_α is the Euclidean distance from the end configuration resulting from a single control to the attractor configuration subgoal α. The attractor α is positioned on a contour of obstacle potential at a distance *s* close to but safely away from the obstacle edge [10]. Descent towards the attractor is continued using the cost potential metric defined as in (3) until the attempt is within an acceptable fraction (e.g. 1%) of the initial subgoal distance.

The Euclidean minimiser is designed to effectively counter drift in all 3 approach, engage, and leave modes. Moves made in attempting any given subgoal are small, and the updates are frequent (every 50 ms). Consequently only a small adjustment is required to bring the robot back on course to correct the drift.

Travel goes forwards along a chain of subgoal positions even if this means temporarily being at a greater Euclidean distance from the goal configuration, such as can occur when going around the c-shape obstacle in Fig. 3. In this way, the moving dip in the dynamic field passes through locations such as the centre of the c-shape that would be fixed local minima using a Euclidean metric in static fields.

Leaving an obstacle requires progress to the goal to be assured. Such progress is measured here as Euclidean distance to goal relative to the hit point, and also involves the plan for the future path if the robot were to leave the obstacle. This plan is made up from the 1-step planner's leave path and the M-step plan for free space travel that it joins. A leave path that is quick to plan is composed of two end arcs, with curvature that of the minimum turning circle, joined by a straight line. In the actual system, we use a smoother (x,y) spline whose peak curvature could breach the minimum turning circle constraint in theory but which does not in practice. The arc-based version is kept as a fail-safe.

Progress is taken to be sufficient when the planned future path has an initial direction pointing inside a circle centered at the goal with a radius to the hit point (Fig. 3). The planned future path will then keep turning inside the circle to its center. Since the path all lies within the circle, this guarantees that traveling along this path would diminish Euclidean distance to goal. When the robot is also clear to leave the obstacle forwards along the planned future path away from the nearest obstacle edge, the robot is thus *safe to leave* with further convergence to the goal assured as per all BUG algorithms [7].

The top-level obstacle-rounding algorithm with input of sensed obstacle edges may be described as follows:

1. When nearest obstacle sensed to lie across planned M-step path, enter *approach mode.*
2. Make moves (in (x,y)) towards obstacle potential contour until robot position smoothly *hits* the contour.
3. If not clear to leave immediately, enter *engage mode* else go to 5. and enter *leave* mode.
4. Make moves forwards along obstacle potential contour until safe to leave obstacle.
5. Enter *leave mode*.
6. Make moves until robot position merges smoothly with the latest planned M-step path.

For more detail on the holonomic aspects, see [10].

D. Completeness

Each of the 4 travel modes ensures any path planned for the mode satisfies the turning constraint. Paths with excess curvature breaking this condition do not occur in plans for free and leave modes as they are not solutions with locally optimised curvature under the safeguards in either mode's scheme as given earlier. For the approach mode, paths with excess curvature do not occur because targets are set at a minimal distance of the minimum turning circle radius. For the engage mode, they do not occur because the obstacle potential contour followed is set at a distance of *s* away from the obstacle edge so that it is not asked to curve round any obstacle point(s) with excess curvature.

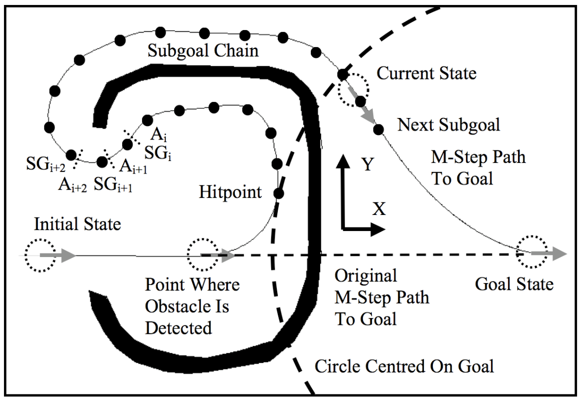

Fig. 3. Subgoals SG_i are set one at a time to pull executed attempts A_i on them around obstacles and then to join with an M-step path in free space connecting smoothly to the goal (x,y,ϕ) configuration state.

The two planners always reduce the Euclidean distance between each hit or leave point and the goal. Hence the planners combine so that, in the absence of drift, where there is a possible goal-connecting path, such a path is found. By "possible" path is meant one that satisfies the turning circle constraint, as well as the narrow passage constraint mentioned earlier, so that the goal is approachable by the robot without being less than *s* away from the obstacles. Illustrations of overall travel to a goal (x,y,ϕ) state using the planners are given in Figs. 3, 4, and 5.

III. Results

The system's completeness has been tested on a series of simulated differential drive and car-like robots, with each type having a different forwards kinematics, in a series of simulated 2D environments. These consisted of ten obstacle courses each containing between one and twenty obstacles of significantly different shapes and sizes. Each environment used 4 pairs of initial and goal (x,y,ϕ) configurations, over which all areas of the course were travelled. These courses were deemed a representative subset of all possible obstacle environments as all conceived robot-obstacle interactions are present. This is because testing included obstacles of regular and irregular shapes of sizes larger and smaller than both the robot and its sensing capabilities, as well as closely packed and well-spaced obstacles. C-Shaped obstacles were also tested to confirm that the method is not subject to traditional local minima problems (Figs. 3&4). Initial and goal positions close to and far from obstacles have also been tested, as have initial and goal headings which face the robot towards or away from obstacles and each other. Overall 3000 tests were conducted during which none of the internal parameters (see §II.B&C) were altered thus showing that different tuning is not required for each test, environment, or robot. Throughout testing the results were found to be the same for differential drive and car-like robots of the same size and turning circle within the test granularity.

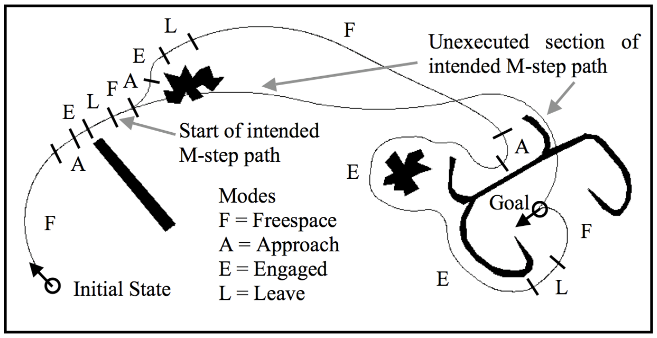

Fig. 4. One of the environments and the 4 travel modes in a typical solution path, together with an unexecuted path segment.

One of the environments and the 4 travel modes in a typical solution path is shown in Fig. 4 together with a path segment which is unexecuted due to the detection of an obstacle during motion. Another is shown in Fig. 5 along with the area reached by the obstacle range sensors. All courses were tested for turning circles and robot radii at regular values between 0.15 and 1m. These values are representative of commonly used sizes of robots and their turning circles. Although turning circles are not necessary for differential drive robots they were applied with the same values as applies to a car-like robot of the same size.

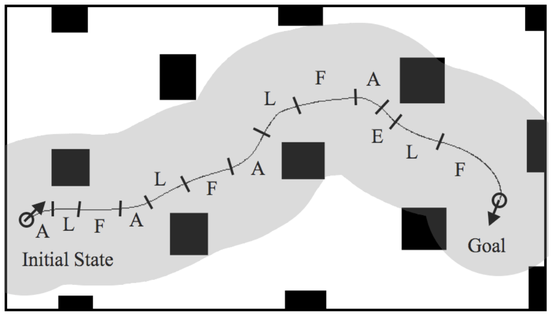

Fig. 5. One of the environments and the travel modes in a typical solution path, together with the sensor range used (10x robot radius – grey area)

Without perturbation through sensor error and with perfect forwards kinematics, all tasks were completed successfully with solution paths conforming to their theoretical expectations. These were: locally minimal curvature in free space and tight circumnavigation of obstacles without any unnecessary undulation, kinks or loops in the path. Accepting that a system with no global map may turn right or left somewhat arbitrarily upon initial sensing of an obstruction, no unnecessary circumnavigation of obstacles was observed, i.e the paths were tightly focused towards the goal. In the tests, an (x,y) spline for the leave path was used and the arc-based alternative was unused.

Perturbation was then put into effect. Table II shows the range of error artificially introduced to sensors and actuators and the limits at which the tests depicted in Figs. 4&5 suffered collision or connection failure. All % errors were started at 0% and incremented by 10% for each subsequent test. The radian increments were 0.04.

As perturbation is increased, although sensor error may be averaged out to a large extent, significant drift occurs. The drift makes travel go inside the intended obstacle potential contour, solution passages become unnecessarily passed by, and the robot steers away from the goal direction. It nevertheless takes substantial perturbation to prevent replanning from maintaining goal connection without collision as Table II shows.

TABLE II.
PERTURBATION AND SUCCESS LIMITS FOR FIGS. 4&5

Error Application	Range Tested	Limit Fig 4	Limit Fig 5
Goal distance sensor	-50% to 50%	±40%	±40%
Goal angle sensor	-1 to 1 radian	±0.12 radians	±0.8 radians
Obstacle sensors	-50% to 50%	±40%	±20%
Actuator sensors	-50% to 50%	±40%	±30%
Control executions	-60% to 60%	±50%	±40%

TABLE III.
COMPUTATIONAL PERFORMANCE

Operation	Time Taken
M-step re-plan	< 1 second
1-step re-plan	< 50ms

The algorithms are coded in Java 5, computational performance running on Mac OS 10.5 with a 2Ghz Intel core 2 duo processor with 2GB of ram is shown in Table III. These times have been found to be the same when testing both differential drive and car-like robots, suggesting that they will not change significantly for other robots with 2 controls and 3 configuration variables.

IV. Conclusion

In order to provide high quality paths for nonholonomic obstacle navigation, a generic path planner with smooth low curvature solutions has been presented based on coordinated dynamic potential fields.

The methodology is designed so that obstacles may be navigated without global maps or positioning systems. It is also designed to provide complete goal connection in the driftless case, and resist being broken when there is drift through quick replanning based on search with the low costs of local gradient descent.

The system has been tested on simulated differential drive and car-like robots with various limited turning circles and widths. Various simulated environments have been successfully navigated without and with various forms of substantial perturbation.

Future work is intended to test other approaches on the same obstacle courses and examine further generic coordinated organisation of dynamic potential fields.

References

[1] Y. Guo, and T. Tang, "Optimal Trajectory Generation for Nonholonomic Robots in Dynamic Environments", *IROS*, 2008.

[2] J. P. Laumond, S. Sekhavat and F. Lamiraux, "Guidelines in Nonholonomic Motion Planning for Mobile Robots", *Lecture Notes in Control and Information Sciences 229*, Springer, 1998, pp. 2-53.

[3] H. Choset, "Principles of Robot Motion : Theory, Algorithms, and Implementation", MIT Press, 2005.

[4] A. A. Masoud, "Dynamic Trajectory Generation for Spatially Constrained Mechanical Systems Using Harmonic Potential Fields", *ICRA*, 2007, pp. 1980-1985.

[5] L. Zexiang and J. F. Canny, "Nonholonomic Motion Planning", *The Kluwer International Series in Engineering and Computer Science*, Vol. 192, 1993.

[6] S. Wei and M. Zefran, "Smooth Path Planning and Control for Mobile Robots", *IEEE International Conference on Networking, Sensing and Control*, 2005, pp. 894-899.

[7] M. K. Weir, A. Buck, J. Lewis, "POTBUG A Mind's Eye Approach to Providing BUG-like Guarantees For Adaptive Obstacle Navigation Using Dynamic Potential Fields", *In Proceedings of International Conference on Simulation of Adaptive Behaviour (SAB)*, 2006.

[8] C. de Boor, "A Practical Guide to Splines", *Applied Mathematical Sciences*, Vol. 27, Springer-Verlag, 1978, pp. 66,67.

[9] K. Yang, S. Sukkarieh, "3D Smooth Path Planning for a UAV in Cluttered Natural Environments", *IROS*, Nice, France, 2008.

[10] M. Weir, J. Lewis, and M. Bott, "Enabling Nonholonomic Smoothness Generically Allowing For Unpredictable Drift", *International Conference on Control, Automation, Robotics and Vision*, 2008.

[11] D. Ferguson, and A. Stentz, "Anytime, Dynamic Planning in High-dimensional Search Spaces", *ICRA*, 2007.

[12] F. Lamiraux, D Bonnafous and Oliver Lefebvre, "Reactive Path Deformation for Nonholonomic Mobile Robots", *IEEE Transactions on Robotics*, 2004, Vol. 20, No. 6, pp. 967-977.

[13] F. Lamiraux, D. Bonnafous and C. Van Geem, "Path Optimisation for Nonholonomic Systems: Application to Reactive Obstacle Avoidance and Path Planning", *Control Problems in Robotics*, Springer 2002, pp. 1-16.

[14] D. Bonnafous and F. Lamiraux, "Sensor based trajectory following for Nonholonomic systems in highly cluttered environment", *IROS*, 2003.