Path Planning and Obstacle Avoidance for Autonomous Mobile Robots: A Review

Voemir Kunchev[1], Lakhmi Jain[1], Vladimir Ivancevic[2], and Anthony Finn[2]

[1] School of Electrical and Information Engineering,
Knowledge Based Intelligent Engineering Systems Centre,
University of South Australia, Mawson Lakes SA 5095, Australia
kunvy001@students.unisa.edu.au,
Lakhmi.Jain@unisa.edu.au
[2] Defence Science and Technology Organisation
{Vladimir.Ivancevic, Anthony.Finn}@dsto.defence.gov.au

Abstract. Recent advances in the area of mobile robotics caused growing attention of the armed forces, where the necessity for unmanned vehicles being able to carry out the "dull and dirty" operations, thus avoid endangering the life of the military personnel. UAV offers a great advantage in supplying reconnaissance data to the military personnel on the ground, thus lessening the life risk of the troops. In this paper we analyze various techniques for path planning and obstacle avoidance and cooperation issues for multiple mobile robots. We also present a generic dynamics and control model for steering a UAV along a collision free path from a start to a goal position.

1 Introduction

In the past few decades there has been a great interest in the problem of motion planning for autonomous mobile robots. To better define motion planning problem we can decompose it into path planning and trajectory planning. Path planning is taking care of the generation of obstacle free path taking into consideration geometric characteristics of obstacles and the kinematic constrains of the robot. Trajectory generation deals with the robot's dynamics, moving obstacles or obstacles not known priori which are time dependent constrains [1]. The basic mobile robot navigation can be divided into the following tasks:

- Generate a model of the environment in the form of map.
- Compute a collision free path from a start to a goal position.
- Traverse the generated trajectory (with specified velocity and acceleration) and avoid collision with obstacles.

Among the mobile robot research society reactive behavior and planning behavior are often accepted as opposite approaches. The mobile robot must be able to act accordingly when unforeseen obstacles are found on the fly. If the robot rely only on pure path planning the robot is prone to physical collision with an unforeseen obstacle. On the other hand without path planning, with the use of reactive obstacle avoidance method only it will be impossible for the robot to reach its goal location.

B. Gabrys, R.J. Howlett, and L.C. Jain (Eds.): KES 2006, Part II, LNAI 4252, pp. 537 – 544, 2006.

Considering the robot environment motion planning can be either static or dynamic. We have static environment when the location of all the obstacles is known priori. Dynamic environment is when we have partial information about obstacles prior the robot motion. The path planning in a dynamic environment is done first. When the robot follows its path and locates new obstacles it updates its local map, and changes the trajectory of the path if necessary.

In his work Fox [2] divides the collision avoidance problem into “global” and “local”. The global techniques that involve path planning methods rely on availability of a topological map defining the robots workspace and obstacle location. He explains that the benefit from using path planning is that the entire path from start to goal can be planned, but this method is not suitable for fast collision avoidance due to its slowness cause by their complexity. On the other hand the local approaches of using pure obstacle avoidance methods suffer from the inability to generate an optimal solution. Another problem is that when using local approach only the robots often get ensnared into a local minimum. Because of these shortcomings, a reactive local approach representing obstacle avoidance cannot be considered feasible for dealing with robot navigation. Due to the reason that there is not a single universal method that can deal with both problems we need to combine both obstacle avoidance and path planning techniques to develop a hybrid system (combining reactive and deliberative approaches) overcoming the weakness of each of the methods.

The hybrid architecture unites the reaction with planning in a heterogeneous system by combining “low-level control” and “high-level reasoning” [3]. The most common hybrid systems are comprised of three layers [4]:

- Reactive layer uses low level sensor based decisions.
- Deliberative layer (planning layer) provides global planning. Its decisions can be based on predefined data (map) or data learned from sensors.
- Executive layer is the intermediate layer between the other two. It process com mands from the planning to the reactive layer.

In that sense we can divide robot navigation problem into two sub tasks:

- Obstacle avoidance
- Path planning

2 Review of Obstacle Avoidance Techniques

During the last decades scientists working in AI have contributed to development of planning methods and algorithms for the purpose of navigation of mobile robots. Considerable work has been done in the development of new motion planning and path planning techniques. This section is surveying common obstacle avoidance algorithms.

The purpose of obstacle avoidance algorithms is to avoid collisions with obstacles. Obstacle avoidance algorithms deals with moving the robot based on the feedback information from its sensors. An obstacle avoidance algorithm is modifying the trajectory of the mobile robot in real time so the robot can avoid collisions with obstacles found on its path.

- **Virtual Force Field**

Bornstein's research [5] on real-time obstacle avoidance for a mobile robot is based on Virtual Force Field (VFF) method. This method involves the use of histogram grid for representing the robots work area and dividing it into cells forming the grid. Any of these cells have a "Certainty Value C (i, j)" showing the measure of confidence that an obstacle is located in the cell. During its movement the robot maps the "range readings" into the Certainty Grid. In the same time the VVF method examines a frame area in the Certainty Grid for the occupied cells. Then the occupied cells repel the robot away. The extent of repellent force depends on the concentration of occupied cells in the examined frame, and it is "inversely proportional to the square of the distance between the cell and the robot".

- **Vector Field Histogram**

Even though the VFF method performs quite fast it has its shortcomings. The implemented test-bed shows that often the robot would not move between obstacles to close to each other due the repellent effect from both sides, causing the robots to repel away, a problem also experienced in the Potential Field method [6]. To solve the problems with VFF Borenstein and Koren [7] developed the Vector Field Histogram VFH technique. The method employs the use of 2D histogram grid to represent the environment, being reduced to single dimension "polar histogram" which is build around the position of the robot in a certain moment. The sectors presented in the polar histogram show the "polar obstacle density". The direction of the robot is computed by choosing the sector with least concentration of obstacles. The map represented by the histogram grid is renewed from the robot's sensors with data containing the distance between the robot and obstacles.

- **VFH+**

The VFH+ method [8] is similar to the VFH but introduces some novelties by employing "threshold hysteresis" to improve the shape of the trajectory, and the use of a cost function. The cost function is used to choose the best direction in between all candidate directions (which are free of obstacles) provided by the polar histogram. The selected direction is the one with the lowest cost. The new VFH+ method considers the vehicle width by enlarging the cells containing obstacles, which makes it easy to experiment with various vehicle dimensions. The trajectory of the vehicle is also considered with by "masking sectors that are blocked by obstacles in other sectors".

- **Dynamic Windows Approach**

The Dynamic Window Approach (DWA) [9] is another method for reactive obstacle avoidance dealing with the kinematical and dynamic constraints of the vehicle in contrast to VFF and VFH methods. The method might be described by a "search for commands" computing the velocities of the vehicle which are then passed to the velocity space. The robot's trajectory consists of a "sequence of circular arcs". The arcs are defined by a velocity vector (v_i, ω_i), in which v_i denotes the translational velocity and ω_i stands for the rotational velocity, together they represent the search space. The search space is being reduced to form a dynamic window, which takes into account the trajectory formed by the circular arcs and defined by the velocity vector.

$$V_r = V_s \bigcap V_a \bigcap V_d \tag{1}$$

The region V_r located in the dynamic window is intersected by the space of possible velocities represented by V_s, the area V_a in which the vehicle is able to stop and avoid collision, and the Dynamic window denoted by V_d.

- **Nearness Diagram**

The problem of obstacle avoidance in highly dense and troublesome environment is presented in [10]. The Nearness Diagram (ND) method uses “divide and conquer” approach splitting the environment into sectors to represent the location of obstacles. Experiments show that ND method can successfully avoid local minima trap only if it is completely visible to the sensors. The ND method utilizes the behavioral based “situated activity” paradigm. The concept uses predefined groups of condition states consisted of different problems and their corresponding actions. When algorithm is performed the current state based on information from sensors is defined and its corresponding action is executed as described in [11].

- **Curvature Velocity Method**

The Curvature Velocity Method (CVM) [12] takes into account the dynamic constrains of the vehicle allowing it to move fast in a dense environment. The velocity and acceleration constrains of the robot, and the presence obstacles presented as circular objects are added to a velocity space. The velocity space consists of translational and rotational velocity. The presumption is that the robot’s trajectory is based along arcs of circles $c=\omega/v$. The velocity is selected on the base of objective function that corresponds to the part of the velocity space that realizes the physical constrains of the robot and the obstacles.

- **Elastic Band Concept**

The Elastic Band Concept [13] works by deforming the original obstacles free path supplied by a path planner. The reason for that is that often the path planner computes a path that has sharp turns, which makes it impossible for the robot to steer. The path modified using the Elastic Band concept is shorter and smoother than the original path. This method can adapt to dynamic changes in the environment modifying the path if new obstacles are detected, avoiding the need for a new path preplanning. There are two forces that modify the form of the new path. A force that mimics the stretching of an elastic band eliminating the “slack” called “contraction force”, and an opposite force called “repulsion force” providing more room by repelling the robot away from obstacles.

The modern obstacle avoidance and algorithms reviewed in this section represent the synthesis of vector-field based techniques and agent-based AI techniques. In our future work we plan to further develop this approach by including some rigorous Lie groups and Lie algebras based methods (refer Section 4).

3 Review of Path Planning Algorithms

- **A* heuristic search**

A* is one of the most common path finding algorithm [14]. For its map representation A* utilizes a grid based search area divided into squares. Each square can be either a free space or an obstacle. In order to find the shortest path a collision free trajectory is calculated comprised of free space squares (also called nodes). To find the shortest path to the goal the A* algorithm uses heuristic approach. A* first adds its starting node A to OPEN set of free space nodes comprising a possible path. The next step is to look for free space nodes around node A and add them to its list and set node A as their parent node. The next step is to add node A to a CLOSED set and delete it from the OPEN set. The next node to be processed is determined by its minimum cost F towards the goal. The lowest cost F=G+H, where G is the cost for getting to the next node, and H is the estimated distance to the goal point. A* provides efficient and complete path finding, but one of its major weakness when dealing with large environments is the vast memory usage caused by the use grid representation of the map.

- **Visibility Graph**

The visibility graph method [15] consists of straight line segments joining at the obstacles visible vertices (but not crossing the obstacle) to define a roadmap from a start to a goal position. In this method the shape of an obstacle is represented as a polygon. The task of the visibility graph method is to connect the start and goal positions with all the vertices of the polygons that are visible. Then connect every single vertex of a polygon with the visible vertex of another polygon. In the created visibility graph any straight line can be a part of the path. The shortest possible path is then calculated using simple graph search technique. This method is prone to let the mobile robot collide with the edge of an obstacle due to the very close distance with its path. A solution of this problem is to artificially increase the size of the polygons before the path is planned so the robot can pass it from a safe distance. Due to increase the number of vertices the visibility graph method performs rapidly in areas when the number of obstacles (polygons) is low.

- **Generalized Voronoi Diagram**

The Voronoi diagram [16] consists of arcs (lines) which are equidistant from the two nearest obstacles. The obstacles in the Voronoi diagram are presented as polygons. The maximized clearance between the Voronoi arc segments and the polygons helps the robot maintain safe distance away from the obstacles. Another advantage of the this method is that it provides a complete path solutions (if possible path exists) based on the fact that if there is a gap between two obstacles there will be a Voronoi line in between. After the completion of the graph, two straight lines are added to the diagram. One line connects the graph with the start position, and a second line is used to connect the goal position [17]. Then graph search technique is used to calculate the roadmap based on the generated Voronoi diagram where the arcs are equivalent to the edges of a graph.

The path finding algorithms reviewed in this section represent the synthesis of computational geometry and AI techniques. In our future work we plan to further develop this approach by including some rigorous optimal control methods, like PMP (Pontryagin Maximum Principle).

4 Dynamics and Control of Steering a Generic UAV

Mechanically, every UAV represents a free-flying rigid body governed by the SE(3)-group, that is, the 6 DOF Euclidean group of 3D motions. Recall that the SE(3)-group couples 3D rotations with 3D translations; technically, it is defined as a non-commutative product of the rotational group SO(3) and the translational group $\boldsymbol{R}^3$. The corresponding nonlinear dynamics problem (that had been resolved mainly for aircraft, spacecraft and submarine dynamics) is called "dynamics on SE(3) group," while the associated nonlinear control problem (resolved mainly for general helicopter control) is called "control on SE(3)-group" [18].

We can represent a point in SE(3) by the 4×4 matrix g defined as:

$$g = \begin{bmatrix} R & x \\ 0 & 1 \end{bmatrix} \tag{2}$$

where $R \in$ SO(3) and $x \in \mathbf{R}^3$. The UAV-dynamics are now defined by:

$$\dot{g} = g \begin{bmatrix} 0 & -u_3 & u_2 & 1 \\ u_3 & 0 & -u_1 & 0 \\ -u_2 & u_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3}$$

where u_1, u_2 and u_3 – being 3 scalar inputs (UAV thrusts). The standard quadratic cost function for a particular trajectory is given by:

$$\tfrac{1}{2}\int_0^T \sum_{i=1}^{3} c_i u_i^2(t)\, dt \tag{4}$$

where c_i are the cost weights and T is desired final time. The optimal inputs are:

$$u_i = \frac{P_i}{c_i} \quad (i=1,2,3) \tag{5}$$

where P_i's are solutions of the Euler-like equations:

$$\begin{bmatrix} \dot{P}_1 \\ \dot{P}_2 \\ \dot{P}_3 \end{bmatrix} = \begin{bmatrix} \dfrac{c_2 - c_3}{c_2 c_3} P_2 P_3 \\ \dfrac{c_3 - c_1}{c_1 c_3} P_1 P_3 \\ \dfrac{c_1 - c_2}{c_1 c_2} P_1 P_2 \end{bmatrix} \tag{6}$$

If the cost weights are equal, then the system is (analytically) integrable and represents the so-called "Lagrange's top."

In case of a fixed altitude, the above dynamics and control problem reduces to the so-called "Hilare robot car," governed by the SE(2)-group of motions (which is a 2D subgroup of the SE(3) group). Given that this UAV always drives forward at a fixed speed, we need to find the steering controls so that the robot, starting from an initial position and orientation, arrives at some final goal position and orientation at a fixed time *T*. In this case the UAV-dynamics are simplified to:

$$\dot{g} = g\begin{bmatrix} 0 & -u & 1 \\ u & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{7}$$

with a single input *u* corresponding to the UAV's turning velocity. The quadratic cost function is simplified to: $\frac{1}{2}\int_0^T cu^2(t)dt.$ In this case, we have the constant of motion given by $l^2 = P_1^2 + P_2^2$, and optimal trajectory given by:

$$P_1(t) = l\cos\theta(t), \tag{8}$$

$$P_2(t) = l\sin\theta(t), \tag{9}$$

where $\theta(t)$ denotes the orientation of the UAV at any given point in time.

The model presented in this section can be further developed by including both modern Lie-derivative based control techniques and Hamiltonian optimal control.

5 Conclusion and Future Work

In this paper we have presented various algorithms and techniques for efficient obstacle avoidance and path planning for mobile robots. We have also presented a generic dynamics and control model for steering a UAV along a collision free path from a start to a goal position. In the case of fixed altitude, this model reduces to the "Hilare robot car." The major area of our future research involves motion planning for multiple mobile robots based on subsumption architecture. At the top level is a path planner which will work in tandem with a flocking behavior. The flocking action will be based on steering behaviors for a flock of mobile robots based on Boids [19] which will be overridden by reactive obstacle avoidance behavior. Our dynamical and control model will be further developed by including both modern Lie-derivative based control techniques and Hamiltonian optimal control.

References

1. Chean, S., L. "Path planning & high level control of an unmanned aerial vehicle", University of Sydney (2002)
2. Fox, D., W. Burgard and Thrun, S. "The Dynamic Window Approach to Collision Avoidance", IEEE Robotics and Automation Magazine, March (1997)

3. Coste-Manière, Ève; Simmons, Reid: “Architecture, the Backbone of Robotic Systems”. In Proceedings of the 2000 IEEE International Conference on Robotics & Automation, San Francisco, CA, (April 2000)
4. Russell, S. & Norvig, P. Artificial Intelligence: a Modern Approach. Prentice-Hall, 1995.
5. Borenstein, J. and Koren, Y. “Real-time Obstacle Avoidance for Fast Mobile Robots.” IEEE Transactions on Systems, Man, and Cybernetics, 19(5) (Sept/Oct 1989)
6. Khatib, O. “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots” The International Journal of Robotics Research, 5(1), (1986)
7. Borenstein, J. and Koren, Y. “The Vector Field Histogram- Fast obstacle avoidance for mobile robots.” IEEE Journal of Robotics and Automation 7(3), (June 1991)
8. Ulrich, I., and Borenstein, J. “VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots” IEEE International Conference on Robotics and Automation, p1572, Leuven, Belgium, (1998)
9. Brock, O., Khatib, O. “High-speed navigation using the global dynamic window approach.” In Proc. ICRA, pages 341-346, (1999)
10. Minguez, J., Montano, L. “Nearness Diagram Navigation (ND): Collision Avoidance in Troublesome Scenarios”. IEEE Transactions on Robotics and Automation, (2004)
11. Arkin, R. “Behavior-Based Robotics”. Cambridge, MA: MIT Press, (1999)
12. Simmons R., "The Curvature Velocity Method for Local Obstacle Avoidance," IEEE Int. Conf. on Robotics and Automation, Minneapolis, USA, (1996)
13. Quinlan S., Khatib O. "Elastic Bands: Connecting Path Planning and Control," IEEE Int. Conf. on Robotics and Automation, Atlanta, USA, (1993)
14. Wilson, N. J. “Principles of Artificial Intelligence” Springer Verlag. Berlin (1982)
15. Nilsson, N. J. “A Mobile Automaton: An Application of Artificial Intelligence Techniques” Proc. 1st Int. Joint Conf. on Artificial Intelligence, Washington D.C., 509-520 (1969)
16. Latombe, J. C. “Robot motion planning” Kluwer Academic Publishers, (1991)
17. Eldershaw, C. “Transfer Report: Motion planning” (1998), Unpublished.
18. Ivancevic, V. and Ivancevic, T. “Natural Biodynamics,” World Scientific Singapore, Series: Mathematical Biology, (2006)
19. Reynolds, C. "Steering Behaviors for Autonomous Characters," Proceedings of Game Developers Conference, (1999)