Final Project: Asteroid Planning

Robot Intelligence: Planning 4649/7649

Group 2: Kyle Volle, Luke Tornquist, Steffan Slater, Xinyan Yan

December 10, 2013

Abstract

In this paper, planning and machine learning concepts are applied to the design of an autonomous agent to play an "asteroids" type computer game.

1 Introduction

The "asteroids" genre of games poses several interesting planning problems. In this game, the player controls a spaceship in an environment that is filled with fast moving obstacles called asteroids. Collision with an asteroid causes the player to lose a life. Points are scored by shooting the asteroids and breaking them into two smaller, faster obstacles. Asteroids progress from large to medium to small. If a small asteroid is hit, it is removed from the game. As the game goes on more and more obstacles are spawned. The goal for the player then is to stay alive as long as possible to keep shooting asteroids.

This game is interesting for several reasons. For starters there is no final goal, but rather the planner seeks to stay alive for as long as possible. This differentiates this problem from more traditional motion planning problems. Additionally, in this game there are no static states. Conservation of momentum keeps the ship moving at a constant velocity if there is no player input. Combined with the dynamic nature of the obstacles, the game state can change rapidly requiring frequent replanning.

The planner utilizes several processes in order to maximize the potential future score. An array of safe trajectories is first calculated and from this list another process selects one of these trajectories based on the anticipated rewards and costs. Finally, a third process determines the game inputs corresponding to following this trajectory and places these inputs in an executor queue to be performed as the replanning occurs. Factors for selecting a trajectory include, but are not limited to, time required to achieve that trajectory, density of asteroids along the trajectory, distance to obstacles along the trajectory and the ability to destroy asteroids along the way. The Methods section will go into greater detail about the techniques and strategies employed.

2 Related Work

A significant amount of literature exists for the problem of navigation with mobile obstacles. This literature tends to fall into one of two categories - local methods or global methods, based on how obstacles influence robot behavior [6]. Local methods include bug algorithms and the Vector Field Histogram, while global methods include artifical potential fields and visibility graphs. Local methods tend to run rapidly but may not find feasible paths, while global methods will find a valid solution if there is one but tend to be slow. Given that our environment can be frequently changing as asteroids collide or are struck by bullets and break apart, a local method would appear to be a better choice

to allow for rapid replanning; however, including some ability to look ahead is critical in an environment with moving obstacles.

The selected approach for navigation planning is based on the concept of velocity obstacles, which takes into account all future obstacle/robot future relative positions, assuming constant velocities [5]. The constant velocity assumption is acceptable for our problem as the plan will be rebuilt frequently, which will account for any asteroid-asteroid or asteroid-bullet interactions. While the exact approach used will be described in detail later, the concept is to convert the space and time domain into a purely spatial representation which can be easily searched for collisions. This results in a velocity space representation of the robot and obstacles, which is then used for planning. This approach, unlike some navigation methods, generalizes well to real-world applications of robots with limited sensors and domain knowledge, and has been used for autonomous surface vessel navigation in the crowded Singapore harbor [4].

3 Methods

3.1 Game and Planner Interface

3.2 Navigation Planning

The navigation planning approach is based upon the velocity obstacle method described in [5]. The motion of the robot and a single obstacle is transformed into relative motion of the robot and a stationary obstacle at the same current position. The asteroid is then expanded by the radius of the robot plus a safety factor. This safety factor prevents this method from being complete, as it can cause the planner to miss potential feasible passages, but compensates for inaccuracies in the turning maneuvers and the slight lag of the planner behind the actual game execution. This expansion of the asteroid collapses the spacecraft to a point, moving the problem into the more tractable configuration space.

After expanding the asteroid, the exclusion cone of the obstacle is computed. This cone consists of two rays from the current robot position through the tangency points of the obstacle. Since the velocity obstacle approach transforms a moving obstacle to a stationary one, any robot velocity relative to the obstacle which falls into the exclusion cone will at some point result in a collision. The current relative velocity of the robot to the obstacle is then placed in the space, and may or may not be within the exclusion cone of the obstacle. It is then necessary to compute the range of possible relative velocities after some amount of time.

The computation of the possible relative velocities is accomplished by computing the potential change in velocity in each direction based on the robot's current attitude. In Asteroids this presents an interesting problem, as the space of potential velocity changes is not a simple shape. While a two-degree-of-freedom actuated effector such as the print head on a 3D printer has a rectangular velocity change space (as the two directions are independent), and a vehicle which

can apply a thrust linearly in any direction has a circular velocity change space, the craft in Asteroids can only accelerate forward and has a finite turn speed. This means that for a given amount of time, the craft can accumulate more change in velocity in a forward direction than in any other direction, as it must first turn before accelerating in that direction. This results in a heart-shaped velocity change space given by the equation:

$$\Delta v(\theta) = a\left(\Delta t - \frac{\theta}{\omega}\right)$$

where a is the acceleration of the craft and ω is the angular velocity while turning. This velocity change space is then rotated by the craft attitude and translated by the relative velocity to obtain the set of reachable relative velocities within the time step. The time step used is the time required to perform a 180-degree turn and accelerate to 25% of the ship maximum velocity or the shortest time to impact, whichever is smaller.

The set of reachable relative velocities is then intersected with the exclusion cone to find which directions (headings) are safe for the craft to move along. This is done by calculating the angle from the current position to each point on the reachable velocity curve and searching for all points where that angle is equal to the angle of the edges of the exclusion cone, as those tangent lines are lines of constant angle (radial lines in polar coordinates). Which side of the intersection heading is safe is determined by picking a point on either side and taking the sum of the angles between the tangent lines and a line though the current position and that point. If the sum is equal to the angle between the two tangent lines, it is between the two and therefore in the exclusion zone. If that sum is greater than angle between the two tangents, it is outside the exclusion zone. This strategy will give the safe and unsafe heading ranges for the craft based on a single obstacle. The results for all obstacles can then be superimposed to get the overall safe and unsafe ranges.

To ensure that not all headings are excluded, only asteroids within some distance of the craft are considered. This distance can be modified by the planner, for example by reducing the search radius to open up more safe headings. From the set of safe headings, one heading is selected as the target and a rotation and acceleration maneuver to that heading is computed. The selection of the heading is the true job of the navigation planner. Most of the extant literature focuses on using the velocity obstacle approach to calculate a path to a goal point, but Asteroids does not have a goal point. The selection of the target heading must therefore be chosen in a different manner. One method that was considered, for example, was selecting the safe heading closest to the current direction, thereby minimizing the amount of turning that was to be carried out.

3.3 Machine Learning

4 Experiments

5 Analysis

6 Discussion

References

[1] Mike Stilman. Task constrained motion planning in robot joint space. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3074–3081. IEEE, 2007.

[2] Tobias Kunz and Mike Stilman. Manipulation planning with soft task constraints. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1937–1942. IEEE, 2012.

[3] John Reif and Micha Sharir. Motion planning in the presence of moving obstacles. *Journal of the ACM (JACM)*, 41(4):764–790, 1994.

[4] Tirthankar Bandyophadyay, Lynn Sarcione, and Franz S Hover. A simple reactive obstacle avoidance algorithm and its application in singapore harbor. In *Field and Service Robotics*, pages 455–465. Springer, 2010.

[5] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17 (7):760–772, 1998.

[6] Seyed Mohammad Khansari-Zadeh and Aude Billard. A dynamical system approach to realtime obstacle avoidance. *Autonomous Robots*, 32(4):433–454, 2012.