Online Algorithms for Searching and Exploration in the Plane

Subir Kumar Ghosh

School of Technology & Computer Science
Tata Institute of Fundamental Research
Mumbai 400005, India
Overview

1. What is online algorithm?
2. Efficiency of online algorithms.
5. Continuous and discrete visibility.
7. Searching for a target in an unknown star-shaped polygon.
What is offline algorithm?

- Starting from $s$, a point robot is searching for the point $t$ in $R$.
- If the robot has the complete geometric information (or map) of $R$ and also knows the exact location of $t$, then the robot can choose a path inside $R$ to move from $s$ to $t$.
- In many situations, it is expected that the robot follows the Euclidean shortest path from $s$ to $t$ inside $R$.
- In some situation, the robot may be asked to follow a minimum link (or, turn) path from $s$ to $t$ inside $R$. 
There are known efficient sequential algorithms for computing such paths.

Thus, the robot can compute an optimal path, depending upon the optimization criteria, using its on-board computer system and then follows the path from $s$ to $t$.

Such algorithms are called *offline algorithms* of a robot path planning for a target searching problem in a known environment.

What is online algorithm?

- Suppose, a robot does not have the complete knowledge of the geometry of $R$ apriori.
- The robot also does not know the location of the target $t$, but the target can be recognized by the robot.
- In such a situation, the robot is asked to reach $t$ from its starting position $s$ using its sensory input provided by acoustic, visual, or tactile sensors of its on-board sensor system.
- The problem here is to design an efficient *online algorithm* which a robot can use to search for the target $t$.
- Observe that any such algorithm is ‘online’ in the sense that decisions must be made based only on what the robot has received input so far from its sensor system.
One of the difficulties in working with incomplete information is that the path cannot be pre-planned and therefore, its global optimality can hardly be achieved.

Instead, one can judge the online algorithm performance based on how it stands with respect to other existing or theoretically feasible algorithms.
The efficiency of online algorithms for searching and exploration algorithms is generally measured using their competitive ratios.

$$\text{Competitive ratio} = \frac{\text{Cost of the online algorithm}}{\text{Cost of an optimal offline algorithm}}$$

Searching for a target on a line

- Suppose, the target point $t$ is placed on a line $L$ in an unknown location.
- Starting from a given position $O$ on $L$, the problem is to design an online algorithm for a point robot for locating $t$.
- It is assumed that the robot can detect $t$ if it stands on top of $t$ or reaches $t$.
- The problem may be viewed as an autonomous robot is facing a very long wall and it wants go to the other side of the wall through a door on the wall but it does not known whether the door is located to the left or right of its current position.
Suppose the robot knows that \( t \) is located exactly \( d \) distance away from \( O \).

Then the robot first walks \( d \) distance to the right.

If \( t \) is not found, then the robot returns to \( O \) and then walks \( d \) distance to the left.

So, the competitive ratio of this straightforward on-line algorithm is 3.

What is the competitive ratio of the search if \( d \) is not known apriori?
Alternate walk

The robot walks one unit to the right along $L$. If $t$ is not found, then it returns to its starting point $O$.

In the next step, the robot walks two units to the left of $O$ along $L$. If $t$ is not found again, the robot returns to $O$.

In the next step, the robot walks four units to the right along $L$ and if it is again unsuccessful to locate $t$, it returns to $O$.

After some steps, the robot locates $t$.

The process of doubling the length is known as *doubling strategy*.
Assume that $t$ is located at a distance $d$ from the origin on the positive axis.

Assume that $2^{k-1} < d \leq 2^{k+1}$ for some $k$.

The total distance traveled during the alternative walk is
\[(2.1 + 2.1| - 2| + 2.4 + 2.1| - 8| + \ldots + 2.2^{k-1} + 2.1| - 2^k| + d = 2.2^{k+1} + d)\].

If the location of $t$ is known apriori, then it is a straight walk of length $d$ from the origin to $t$.

So, the competitive ratio of the alternate walk is $(2.2^{k+1} + d)/d = 1 + 2.2^{k+1}/d$ which is at most $1 + (2.2^{k+1}/2^{k-1}) = 9$. 
A beautiful young cow Ariadne is at the entrance of a simple labyrinth which branches in \( m \geq 2 \) corridors. She knows that the handsome Minotaur is waiting somewhere in the labyrinth. What is the best searching strategy for Ariadne to locate Minotaur?

Visit \( m \geq 2 \) rays in a cyclic order starting with an initial walk of length one.

Increase the length of the walk each time by a factor of \( m/(m - 1) \) till \( t \) is located.

This strategy gives the competitive ratio of \( 1 + 2m^m/(m - 1)^{m-1} \), which is optimal.


Assume that the point robot knows the exact location of $t$ but does not know the positions of unknown polygonal obstacles $h_1, h_2, \ldots, h_k$.

The robot starts from $s$, and moves towards $t$ following the segment $st$ till the robot detects by its tactile sensor that it has hit a polygonal obstacle (say, $h_i$) at a some point $u_i$.

Then the robot goes around the boundary of $h_i$ to locate the boundary point of $h_i$ (say, $v_i$) which is closest to $t$.

Then the robots moves from $u_i$ to $v_i$ following the shorter of the two paths from $u_i$ to $v_i$ along the boundary of $h_i$. 

\[ R \]
Then the robots moves from $u_i$ to $v_i$ following the shorter of the two paths from $u_i$ to $v_i$ along the boundary of $h_i$. 

Treating $v_i$ as $s$, the robot repeats the same process of moving towards $t$ following the segment $v_i t$ till $t$ is reached. 

The length of the path traversed by the robot is bounded by the length of $st$ and 1.5 times the perimeters of those polygonal obstacles that are hit by the robot.


Algorithms for target searching in an unknown unbounded region


The visibility polygon of $P$ from a point $p$ (denoted as $VP(P, p)$) is the set of all points of $P$ that are visible from $p$.

In other words, for every point $z \in P$, if the line segment joining $z$ and $p$ lies inside $P$, then $z$ belongs to $VP(P, p)$.

Continuous and discrete visibility

If the robot computes visibility polygons from each point on its path, we say that $P$ is explored under continuous visibility.

If the robot computes visibility polygons from a selected set of points on its path, we say that $P$ is explored under discrete visibility.
Let $u_1, u_2, \ldots, u_{n/4}$ be the nearest points of $s$ in the alleys of a simple polygon $P$ of distance $d$ such that if the robot moves from $s$ to $u_i$ for each $i$, the robot can see the alley completely.

In order to search $t$, the robot moves from $s$ to $u_i$ in each alley and then returns to $s$ if it does not locate $t$.

For every unsuccessful search, the robot travels $2d$ distance.
In the worst case, the robot locates \( t \) in the last alley.

So, the total distance travelled by the robot is at least 
\[ 2d(n/4 - 1) + d. \]

Hence, the lower bound of the competitive ratio for this problem is \( n/2 - 1 \).


A simple polygon $P$ is said to be a street (also called LR-visibility polygon) if there exists two points $s$ and $t$ on the boundary of $P$ such that every point of the clockwise boundary from $s$ to $t$ of $P$ (denoted as $L$) is visible from some point of the counterclockwise boundary of $P$ from $s$ to $t$ (denoted as $R$) and vice versa.

Observe that if a point robot moves along any path between $s$ and $t$ inside the street $P$, it can see all points of $P$. 
Algorithms for target searching in an unknown street


The left and right constructed edges of $VP(P, s)$ decide the movement of the robot initially. If $\theta < \pi/2$, then the robot follows the bisector of $\theta$ till it reaches a point where $\theta$ becomes $\pi/2$. Then the robot follows a curve path toward $v_l v_r$ which is define by an algebraic expression based on positions of current $p$, $v_l$ and $v_r$. 
Target searching using link paths

Another problem for searching $t$ in an unknown street $P$ is find a path such that the number of links (or, turns) in the path is as small as possible.

Walking into the kernel in an unknown star-shaped polygon with continuous visibility

Starting from the initial position $s$, the problem is to design a competitive strategy to walk into the kernel of $P$.

Algorithms for walking into the kernel


Exploring unknown polygons: continuous visibility

Starting from a point $s$ inside $P$, the exploration problem is to design an online algorithm which a point robot can use for moving inside $P$ such that every point of $P$ becomes visible from some point on the exploration path of the robot.

However, if $P$ contains holes, the exploration problem does not admit competitive strategy.

Exploring simple polygons: continuous visibility

Observe that if both edges of every reflex vertex $u_i$ of $P$ are seen by the robot, then the entire $P$ has been explored by the robot.

Exploring unknown polygons: discrete visibility

In the remaining part of the lecture, we present exploration algorithms and their competitive ratios from the following papers.


Motivation for discrete visibility

Many on-line computational geometry algorithms for exploring unknown polygons assume that the visibility region can be determined in a continuous fashion from each point on a path of a robot. Is this assumption reasonable?

1. Autonomous robots can only carry a limited amount of on-board computing capability.
2. At the current state of the art, computer vision algorithms that could compute visibility polygons are time consuming.
3. The computing limitations suggest that it may not be practically feasible to continuously compute the visibility polygon along the robot’s trajectory.
4. For good visibility, the robot’s camera will typically be mounted on a mast and such devices vibrate during the robot’s movement.
5. Hence for good precision the camera must be stationary while computing visibility polygons.

It seems feasible to compute visibility polygons only at a discrete number of points.
Exploration cost

Is the cost associated with a robot’s physical movement dominate all other associated costs?

The essential components that contribute to the total cost required for a robotic exploration can be analyzed as follows. Each move will have two associated costs as follows.

1. There is the time required to physically execute the move. If we crudely assume that the robot moves at a constant rate, \( r \), during a move, the total time required for motion will be \( r D \), where \( D \) is the total path length.

2. In an exploratory process where the robot has no apriori knowledge of the environment’s geometry, each move must be planned immediately prior to the move so as to account for the most recently acquired geometric information. The robot will be stationary during this process, which we assume to take time \( t_M \).

3. Since the robot is stationary during each sensing operation, we assume that it takes time \( t_S \).
Let $N_M$ and $N_S$ be respectively the number of moves and the number of sensor operations required to complete the exploration of $P$. Hence, the total cost of an exploration is equated to the total time $T$ required to explore $P$: $T(P) = t_M N_M + t_S N_S + r D$.

Now, $(t_M N_M + t_S N_S)$ can be viewed as the time required for computing and maintaining visibility polygons by computer vision algorithms, which is indeed a significant fraction of $T(P)$ because computer vision algorithms consume significant time on modest computers in a relatively cluttered environment.

Therefore, we assume that the overall cost of exploration is proportional to the cost for computing visibility polygons.

The criteria for minimizing the cost for robotic exploration is to reduce the number of visibility polygons that the on-line algorithms compute.


We present an exploration algorithm that a point robot can use to explore an unknown polygonal environment $P$ under discrete visibility.

In order to explore $P$, the robot starts from a given position, and sees all points of the free space incrementally.

It may appear that it is enough to see all vertices and edges of $P$ in order to see the entire free-space. However, this is not the case.

Three views from $p_1$, $p_2$ and $p_3$ are enough to see all vertices and edges of $P$ but not the entire free-space of $P$. 
(i) Let $S$ denote the set of viewing points that the algorithm has computed so far. (ii) The triangulation of $P$ is denoted as $T(P)$. (iii) The visibility polygon of $P$ from a point $p_i$ is denoted as $VP(P, p_i)$.

**Step 1:** $i := 1$; $T(P) := \emptyset$; $S := \emptyset$; Let $p_1$ denote the starting position of the robot.

**Step 2:** Compute $VP(P, p_i)$; Construct the triangulation $T'(P)$ of $VP(P, p_i)$; $T(P) := T(P) \cup T'(P)$; $S = S \cup p_i$;

**Step 3:** While $VP(P, p_i) - T(P) = \emptyset$ and $i \neq 0$ then $i := i - 1$;
Step 4: If $i = 0$ then goto Step 7;
Step 5: If $VP(P, p_i) - T(P) \neq \emptyset$ then choose a point $z$ on any constructed of $VP(P, p_i)$ lying outside $T(P)$;
Step 6: $i := i + 1; p_i := z; \text{goto Step 2};$
Step 7: Output $S$ and $T(P);$  
Step 8: Stop.
The algorithm needs $r + 1$ views. Competitive ratio is $(r + 1)/2$, where $r$ denotes the number of reflex vertices of the polygon.

**Open Problem:** Can the bound be improved?
We wish to design an algorithm that a convex robot $C$ can use to explore an unknown polygonal environment $P$ (under translation) following the similar strategy of a point robot.

$C$ needs more than $r + 1$ views for exploration.

**Open problem**

Can one derive an upper bound on the number of views for a convex robot exploration?
Exploring an unknown polygon: Bounded visibility

Computer vision range sensors or algorithms, such as stereo or structured light range finder, can reliably compute the 3D scene locations only up to a depth $R$. The reliability of depth estimates is inversely related to the distance from the camera. Thus, the range measurements from a vision sensor for objects that are far away are not at all reliable.

Therefore, the portion of the boundary of a polygonal environment within the range distance $R$ is only considered to be visible from the camera of the robot.

Vertices of restricted visibility polygon from $p_i$ with range $R$ are $u_1, u_2, \ldots, u_{12}$. 
An exploration algorithm using restricted visibility

- The algorithm starts by computing the restricted visibility polygon $RVP(P, p_1)$ from the starting position $p_1$.

- It chooses the next viewing point $p_i$ on a constructed edge or a circular edge of $RVP(P, p_{i-1})$ for $i \geq 1$ till a boundary point $z$ of $P$ becomes visible.
Taking $z$ as the next viewing point $p_i$, $RVP(P, p_i)$ is computed. Taking viewing points along the boundary of $P$ in this fashion, restricted visibility polygons are computed till all points of this boundary of $P$ become visible.

The process of computing restricted visibility polygons ends once the entire $P$ is explored.
Competitive ratio

The maximum number of views needed to explore the unknown polygon $P$ with $h$ obstacles of size $n$ is bounded by

$$\left\lfloor \frac{8 \times \text{Area}(P)}{3 \times R^2} \right\rfloor + \left\lfloor \frac{\text{Perimeter}(P)}{R} \right\rfloor + r + h + 1.$$ 

The competitive ratio of the algorithm is

$$\left\lfloor \frac{8\pi}{3} + \frac{\pi R \times \text{Perimeter}(P)}{\text{Area}(P)} + \frac{(r+h+1)\pi R^2}{\text{Area}(P)} \right\rfloor.$$ 

Open problem

Can one improve the competitive ratio of the algorithm?
Exploration and Coverage Algorithms


Thank You.