A Method for Solving Arbitrary-Wall Mazes by Computer

IVAN E. SUTHERLAND, MEMBER, IEEE

Abstract—A method for solving mazes with extended open areas and arbitrarily placed walls is described. This method reduces large open areas containing many possible paths to a small set of shortest paths. It is then possible to use Moore's algorithm of which the paper includes a summary. A computer simulation of a vehicle exploring an unknown maze is discussed. Crude navigation and measurement are sufficient for maze solving with the techniques described.

Index Terms—Area mazes, circuit board layout, curved walls, maze solving, Moore's algorithm.

Checkerboard Mazes

In the late 1940's Claude Shannon built a maze-solving mouse. Shannon's device consisted of a 25-square checkerboard maze with removable aluminum walls. A motorized carriage underneath the floor of the maze provided power to the mouse through magnetic coupling. Simple relay circuits provided logic and memory for exploring the maze and remembering a solution. Shannon's device is shown in Fig. 1.

Shannon's maze-solver used a very simple algorithm. He provided two bits of memory for each square to indicate the last direction in which the mouse left that square. These two bits can be thought of as an arrow showing which way the mouse went. When exploring the maze, the mouse always attempted to leave each square it entered in a direction 90° to the left of the recorded direction, updating the recording. If the mouse struck a wall in that direction, it returned to the center of the square and tried again another 90° to the left. With such an algorithm the mouse will eventually find the "cheese," but will appear to be very stupid in its exploration because it will sometimes leave a square by the very opening it used in entering. The behavior of the mouse can be made to appear more reasonable if, unless forced by three walls, it refrains from leaving a square by the same route it used when entering. Once the cheese has been found, all memory arrows for squares which were visited point towards it. For a second run, the mouse need only follow the pointers in each square.

It is easy to see that Shannon's algorithm works for any finite checkerboard maze. Suppose that it did not. For such a maze there would be some set of squares which the mouse does visit, and some that it does not. The set that it does visit contains at least the initial square, and the set that it does not visit contains at least the cheese square. If the maze is solvable, then there must be an opening between these two sets of squares. Consider the square on the visited side of such an opening. Each time this square is visited the mouse leaves by a route 90° different from its former exit route. Thus, in at most four visits, the mouse will leave the square towards "unvisited" territory. The notion of an "unvisited" cheese square therefore is seen to be absurd, and so we know that the mouse will eventually visit all squares.

Shannon's mouse is not guaranteed to find the shortest path to the cheese. In fact, in a maze with no walls at all, Shannon's mouse performs very badly indeed. It wanders around a very long time before striking the cheese, and thereafter remembers its meanders.

Moore's Algorithm: Point and Passageway Mazes

The meanderings of Shannon's mouse caused E. F. Moore to speculate about a method for finding the shortest path through a maze. Moore [1] reconstituted the maze problem in abstract terms as a number of points which could be visited and the available passageways between them, as shown in Fig. 2(b). In Moore's terms, Shannon's maze consisted of 25 points (the centers of the squares) connected by passageways wherever the walls were missing.

Moore's algorithm was also very simple. From the starting point, mark with the number 1 all points which can be reached in one step. From these places, mark...
with the number 2 all the as-yet-unmarked places which can be reached in one additional step. From these places mark with the number 3 all as-yet-unmarked places which can be reached in one additional step, and so on until the goal is reached. The shortest path may then be traced backwards by following any chain of decreasing numbers. The length of the shortest path is, of course, equal to the numeral that was marked on the goal square.

Moore also showed how his algorithm generalizes nicely for finding the lowest cost path through a maze which has a cost assigned to each passageway. Instead of marking places with successive integers, mark them with the lowest achievable cumulative cost. Cumulative cost for some location may be reduced if a longer path to it with lower cumulative cost is discovered. Activity should always proceed from the lowest cost position which has unmarked positions adjacent to it.

Mazes with Large Open Spaces

Moore's algorithm as described so far is most suitable for abstract mazes in which relatively few positions are connected by an irregular array of passageways. What about problems such as wire routing on circuit boards, however? Such problems do not lend themselves simply to Moore's algorithm because the available space consists of broad areas and the "passageways" are ill-defined. Nevertheless, Moore's algorithm can still be applied by breaking the available areas up into many many small squares. The maze represented by such a procedure consists of thousands of locations connected by a regular array of passages. Except where some real boundary such as another wire intervenes, each square is considered to be connected to its neighbor. Although such a representation is easy to understand, its solution requires a great deal of computing.

Several people have developed computer programs for dealing with such extensive mazes. Lee [2] has shown how to solve such mazes not only for the minimum length path, but also how to solve for the path which crosses the fewest number of other paths. Akers [3] has shown how to use only two bits of storage for each maze position, pointing out that the more significant bits of Moore's ordinal numbers are redundant. Lee also pointed out that the propagation of activity through the grid of points into which areas are broken is similar to the propagation of light through space. Lee shows pictures of various "diffraction" phenomena associated with edges in such a maze.

Lee's diffraction patterns show what the behavior of "light waves" would be in his maze. If we consider the behavior of "light rays" rather than waves, we can get a quite different (and novel) picture of how to find the shortest path through a maze defined by the geometry of its barriers. Light diffracts, we know, from the edges of things. Thus, to find the shortest path through such a maze we need to consider only the rays which go from the source to the edges of nearby barriers and the rays which reradiate from these edges to other edges, and so on until they reach the goal. Taking such a "ray view" of the maze formed by barriers in a printed circuit layout, for example, will result in far fewer vertices than the many thousands obtained by breaking up each open space into small areas.

A View from Inside the Maze

If the maze to be solved is completely known, the ray view described above can be used to reduce the maze to the point-and-passageway form for which Moore's algorithm is applicable. If, on the other hand, the maze-solving device is immersed in the maze and has knowledge only of those parts of the maze it has visited, the ray view can serve as a guide for exploring the maze. Let us think of the maze as consisting of open space with certain arbitrarily positioned and arbitrarily shaped walls. Such a maze is shown in Fig. 2(c). We will assume that the "subject" or mechanism which is to solve this maze can "see" the space immediately around itself. We will also assume that the walls are opaque so that in any direction the subject can see only to the first wall. We will further assume that the subject can form a rough idea of the distance to a wall in any direction, and that the subject has a rough sense of relative direction. These sensing abilities are patterned after our own visual sense.

It is my intention to describe in detail a method for exploring a maze with such senses, and a system for remembering a very modest amount of information sufficient to describe the path followed. The method described requires only relatively crude sensing of the maze walls and no absolute sense of position or direction. The method has been implemented in a computer program which will be described. Our interest here is primarily the practical problem of how to explore and remember a maze with arbitrary walls. The concern is not with the purely mathematical problem of proving that the method will work for all mazes. Indeed, because we assume a pseudorandom number generator (which must necessarily produce a repetitive sequence), one can imagine a maze which the method will not solve. The
chance of encountering such a maze in circuit layout or other practical work is very small.

**Obstructions and the Spaces Between Them**

The rooms and buildings in which we live form mazes of the type considered here which we succeed in solving every day. The key through which we explore and solve these mazes may lie in our ability to detect the edges of open doorways. Thus, looking out from a room, we can tell that there are parts of the adjoining room or hall which we cannot see from where we are. Beyond an open door there are hidden regions on each side. We describe how to go from one place to another in a building by advice such as, “Take the first left, go to the third door on your right, and you’re there.” Thus we seem to express our solutions to these everyday mazes in terms of the places we cannot see from where we are.

Some special terms can be used to better understand the heuristic notions suggested by our own descriptions of how to get around in a maze. The “scene” is that function of bearing angle which gives the approximate distance to the furthest wall visible at the given angle. The heuristic notions we are interested in are all associated with discontinuities in the scene, that is, they are all associated with angles at which the distance to the furthest visible wall changes abruptly. Each such discontinuity indicates that some area, not presently visible, may exist. We will call the opening into such an area a “spur,” as shown in Fig. 3. The “point defining the spur” is the point on the nearer wall at the angle of the discontinuity. The “bearing” of the spur is the specific angle at which it occurs. We will never need to know the actual numerical value of the bearing except to determine which of two spurs is to the right or left of another. The “range” to the spur is the distance from the observer to the point defining the spur. The “depth” of the spur is the distance from the point defining it to the wall behind it. We will never need to know the range or the depth of the spur accurately; they are used only to determine whether an opening between spurs or the spur itself is large enough for the subject to fit through.

A spur can be either right- or left-handed. A left-handed spur, such as the one shown in Fig. 3, is one which would be explored by turning left. An “opening” is the space between a left-handed and a right-handed spur if the left-handed one is to the left. An “obstruction” is indicated when a left-handed spur exists to the right of a right-handed one. The “width” of an opening or obstruction is the distance between the points defining the two spurs involved.

In the discussion which follows, it will be assumed that pathological wall conditions are not present. The walls must be free of cusps and knife edges. Roughnesses in the walls which create spurs whose depth is much less than the size of the subject will be ignored. Similarly, openings too small for the object to pass through will be ignored. Straight sections of the walls will be considered to be arcs of very large circles.

As the subject moves through the maze, the positions of the points defining the spurs may also move. The restrictions placed on the maze insure that the motion of the defining points will be continuous. The subject must keep track of this motion because spurs are identified and named, and continuous changes in a spur are allowed without changing its name. The restriction against straight walls can be removed if special logic is provided to properly identify a spur after the point defining it has moved discontinuously across the straight wall, as shown in Fig. 4.

Although the motion of spurs or the points defining them are continuous, the number of spurs will not be constant but will also change as the subject moves through the maze. These changes can occur in four ways.

1) A spur may appear where none existed before, as in Fig. 5(a), when the subject crosses the broken line in moving from point \( x \) to point \( y \).
2) A spur may disappear because the point defining it has reached a point of inflection causing its depth to become zero, as when moving from point \( y \) to point \( x \) in Fig. 5(a).
3) A spur may split when the subject crosses a line.
tangent in two places to the walls, as in Fig. 5(b),
when the subject crosses the broken line in going
from point x to point y.
4) Several spurs may merge under conditions similar
to 3), as in Fig. 5(b), when moving from point y
to point x.

It has been convenient to restrict the maze so that
each split is two-way; in such mazes, no three
to the walls may be collinear. This restriction can
usually be met by making very small changes to a maze
wherever it does not conform to the restriction. In the
computer simulation, nearly collinear tangents gave
considerable trouble.

To remember where it has been in the maze, the sub-
ject need remember only the history of changes in the
spurs. Given the two-way split restriction on the maze,
this record can take the form of a simple binary tree
which we can call a “split table.” A name is assigned
to each spur, and when it splits or merges, new names are
given to the resulting spurs. The names given to the two
results of a split are recorded with the nearer spur first.
If two spurs merge, the split table is checked to see if
they stemmed in the same order from a former split.
If so, the merged spur is given its former name; if not,
a new name is assigned and entered in the split table
with those two predecessors, so that if the result of a
merger should later on split, the resulting spurs will be
given their former names. When a spur disappears, this
fact is marked in the split table. Such a procedure will
assure that after having explored a certain passage and
having returned to its entrance, the subject will have
the same names assigned to the spurs as were assigned
to them before entering the passage, except for spurs
which have appeared or disappeared during the process.

The split table provides memory sufficient to solve
the maze only because splits and mergers of spurs are
unique. To be sure that splitting is unique, consider
the motion of the point defining a particular spur. It cannot
cross inflection points of the walls, nor can it cross points
of tangency of collinear tangents. The point defining a
particular spur is restricted to motion along a short
length of wall free of inflection points. A spur can split
only when the subject moves towards it and only when
its defining point is at one end of its short length of wall.
Moreover, a spur which merely disappears cannot split.
A split is therefore unique.

That merging is unique is more difficult to prove. If
spur H splits into spurs I and J, and some later motion
causes I and J to merge, can we be certain that the
result is properly called H? Recall that the point defin-
ing a spur is confined to a section of wall with constant
direction of curvature. Two such sections of wall have
at most four common tangents. By exhaustive examina-
tion of examples, one can show that if the relative ranges
of the merging spurs are considered, mergers of spurs
are unique. Thus, if a spur H splits into spurs I and J,
with I nearer to the observer, and later if I and J merge
with I nearer to the observer, then we can be certain
that we may indeed call the result H. But also, J and I
may merge with J closer to the observer and then we
certainly must not call the result H. The spur I may
perhaps merge with some other spur, say K, and produce
some result that we similarly must not call H.

Sensors Required for Maze Solving

The technique described above could be used in de-
signing a maze-solving device that will need only crude
equipment. The device must be able to detect spurs,
that is, angles at which the maximum range of vision
changes abruptly. With a radar-type sensor, such de-
tection is fairly easy. With an optical-type sensor simi-
lar to the human eye, abrupt changes in vision range
must be detected by the relative motion of texture on
the near and far walls, and this is still a relatively simple
task. The maze-solving device must also distinguish the
relative angle and range of spurs. That is, the device
must be able to announce which of two spurs close to-
gether in angle is closer to the observer and which is to
the left or right. The device does not need to sense either
the absolute range or the absolute bearing of a spur. It
is useful to distinguish between spurs with sufficient
depth to be explored, and minute ones which are merely
surface roughnesses in the walls. We should point out in
passing that the human visual system provides us only
with this rather crude information.

A device designed to explore a maze must be capable
of doing enough calculation to move appropriately. It
must be sensitive to contact with a wall. When explor-
ing a spur, it should move in such a way as to approach
the spur but miss the nearby walls. It may not be able
to move directly towards the spur. Rather, it should
move somewhat to the left or to the right of the point
defining the spur in order to avoid the walls. If in actual
contact with a wall, the device may have to move along
the wall towards its next goal. None of these computa-
tions is difficult; all that is required is a little geometry
and awareness of its own size.

Exploration Strategy

In the computer-simulation of maze-solving, a simple
exploration strategy was found to be satisfactory. If
after an initial complete scan of the horizon there are no
spurs, then the entire maze has been explored and any
goal will be visible. If there is only one spur, then it must
be examined. To examine a spur the subject moves towards it in such a way as to bring the hidden area behind the spur into view. In other words, if the spur is the right-hand side of an open doorway, the subject should move towards but slightly to the left of it so that when arriving in the doorway, the subject’s position will be sufficiently far from the edge of the doorway to avoid contact with the wall. If the initial scan discovers more than one spur, one of them may be chosen at random for exploration.

If the spur being explored or any other spur under observation disappears, then the area which it formerly concealed has been completely explored. The disappearance of a spur should be marked in the split table. All spurs which disappear without revealing the goal or “cheese” represent blind alleys in the maze. Likewise, all spurs which split only into blind alleys are themselves blind. If the spur being followed disappears, then some other spur not known to be a blind alley should be followed instead. When all spurs are known to be blind alleys, then the maze has been completely explored.

A subject can detect circular passages in a maze only if it marks the maze itself or has a sense of absolute position. From the split table alone, a circular passage with three openings transversed ten times is indistinguishable from a long, straight passage with thirty openings. By selecting at random the spur to explore, a subject will eventually solve even circular mazes. Where improved performance is desired, an absolute sense of position should be included.

So far we have assumed that the subject becomes instantly aware of all new spurs which appear as it moves. Equipment to implement such awareness would require regular complete scanning of the horizon. However, except for an initial scan, complete scanning is unnecessary. It is sufficient to scan only in the direction of known spurs. Newly appearing spurs (not produced by a split and thus not immediately sensed) represent territory which has already been viewed, and so they are of no immediate interest. On the other hand, a newly appeared spur may eventually come to have nearly the same bearing as some spur of current interest, and thus finally be detected. To a limited scan system, such an event seems very similar to a split, but can be distinguished from a true split by examining the ranges involved. Because of this fact, the sensing equipment can concentrate its attention in the few directions required to keep track of the current set of known spurs.

**Computer Study**

During the academic year 1960 to 1961, the author wrote a program for the TX-O computer at M.I.T. to simulate the motion of a subject moving through an arbitrary wall maze. The walls of the maze were represented as opaque areas on a sheet of plexiglass which was placed over the cathode ray tube display of the TX-O. The subject’s sensors were simulated by optical sensing of the opaque areas representing the walls. The examination was confined to that portion of the maze which would be “visible” from the subject’s current location in the maze. The subject was represented by a circle which was displayed on the cathode ray tube so that observers could watch the subject move through the maze.

The spurs were detected by means of a “vision line” which the computer plotted out from the subject. When a vision line reached an opaque maze wall, light from the CRT would no longer be visible. The photomultiplier shown in Fig. 6 sensed this, so that the computer would stop its probe. After a complete initial scan, the computer sent vision lines out only in directions corresponding to the spurs of current interest to it. Whenever a spur split, two new vision lines would appear where only one had been before. An observer could therefore easily see which spurs the subject was concerned with at any time.

The strategy for maze-solving used in the simulation was very simple. The simulated subject always moved towards some spur, or rather, slightly to the left or right of it so as to arrive next to it without actually hitting the wall. Once the subject chose a spur to explore, it followed that one until it either split or disappeared. If the spur being followed split, the subject would follow one of the two resulting spurs selected at random. If the spur being followed disappeared, the subject would follow a new spur chosen at random from those still visible, excluding any which were known to lead only to blind alleys. If all visible spurs were known to lead only to blind alleys, then exploration was complete.

The computer program was able to solve mazes with curved and straight walls and with passages placed anywhere within the confines of the available area. It was fairly difficult to make the mazes conform to the requirements of two-way splits. As a result, the actual mazes successfully solved were fairly simple. An example is shown in Fig. 7. More complicated mazes were tried, but the program was unsuccessful in solving them.
Some difficulty was experienced with sharp right-angle corners, and so their use was avoided. There was also difficulty with slightly concave walls, for it is difficult to make the distinction between a slightly concave wall and a doorway seen from the edge. As far as solution of the maze is concerned, the choice of configuration is of no consequence, but since the choice of doorway makes an entry in the split table slightly different from that made by the choice of concave wall, the decision must be applied consistently.

Conclusions

We have considered the solution of a complicated maze with large open spaces in terms of how rays of light might diffract in going from a source to a goal. Such a ray view of maze-solving can make it possible to select a small set of vertices (the reradiation points) and connect them with a small set of paths (those that the light rays would follow) sufficient to describe the maze. Such a procedure can reduce a maze with large open spaces to the abstract form suitable for solution by Moore’s algorithm. Moreover, the procedure suggested here will result in far fewer vertices than the procedures now in use.

A somewhat modified procedure can be used to explore arbitrary wall mazes when the full extent and layout of the maze is initially unknown. The computer study has shown some of the difficulties of implementing such a procedure. The major restriction that spurs may have only binary splits can probably be removed with only slight additional cost, and work to this end should be an early concern.

It is particularly important to note that spurs need not be discovered immediately after their appearance, for this means that after a full initial scan, the device for detecting spurs need track only already discovered spurs. The spur detector can, therefore, be a narrow beam device. The very lenient requirements on absolute accuracy of the spur detector are also appealing. A marked improvement in reliability could be expected if additional information about the range and location of spurs was stored and used.

In summary, it appears possible to make automatic devices capable of sensing and solving real physical arbitrary wall mazes.

References