On the following $n$ lines are $n$ comma-delimited four-bit coding numbers describing which edges of the square are open to movement. The coding method is described as below. Each number represents a four-bit number that has a bit value of $0$ if an edge is closed (walled) and $1$ if an edge is open (no wall); the $1$s register corresponds with the upwards-facing side, the $2$s register the right side, the $4$s register the bottom side, and the $8$s register the left side.

There are three subsections in this section. They are: project overview, problem statement, and metrics.

After understand the sensing method of robot, let’s take a look at maze01 more detail. The start location is (0,0), and the goal is a square with four positions, they are (5,5), (5,6), (6,5), (6,6). For solving such a maze, we can identify at least two kinds of hazards. I named them as: dead-end hazard and loop hazard. When a robot encounters a dead-end hazard, it has no way for any direction except moves backwards. For example, when the robot get into the position (1,7), (4,9), (8,1), or (10,9), it encounters a dead-end hazard. The other hazard is loop hazard, which is caused by isolated walls. For example, the right-hand-side wall of position (0,9) and (0,10) may causes loop hazard. Our robots should avoid these hazards in the second run.

There are many paths for a robot to reach the goal. I just list three of them as shown in Figure . Because the robot can move up to three units, the moving steps can be reduced. When the steps of a path are reduced, I call this path as an optimized path.

The shortest path of maze01 is path1 (Figure ); it takes 30 steps. After optimization, only 17 steps are required (Figure ).