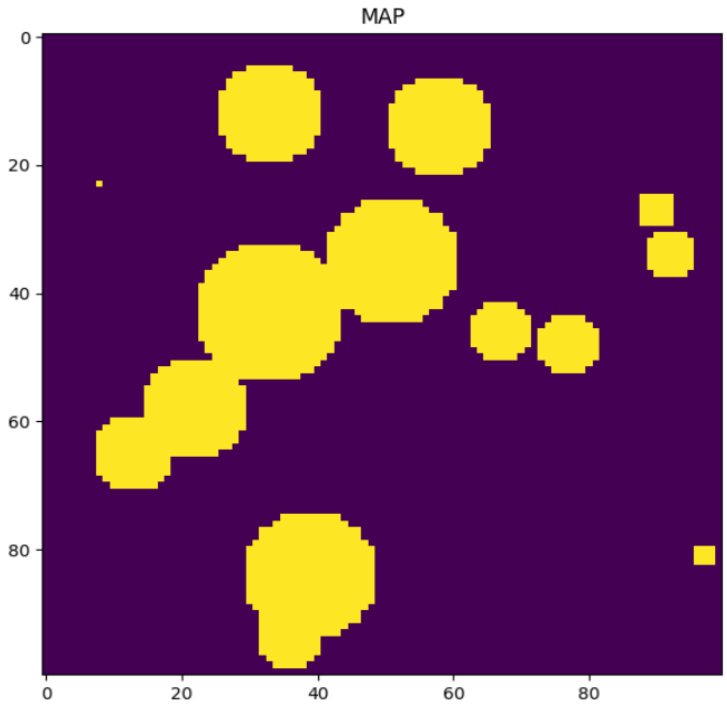
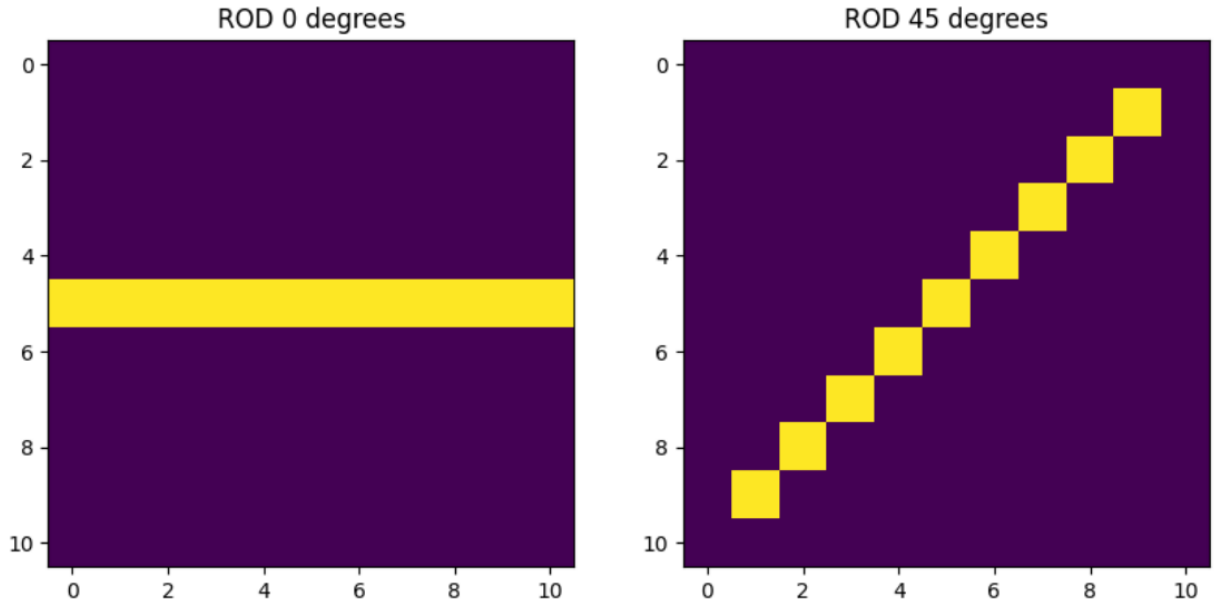
Task 1: Configuration Space

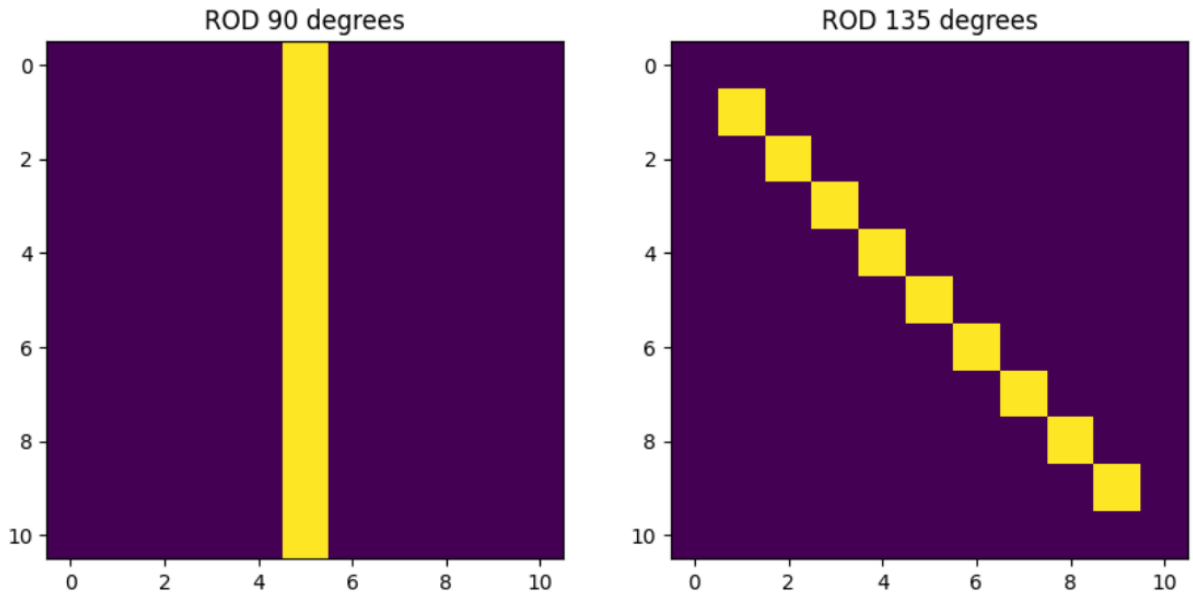
A. Visualize from the given data the workspace and the different rod configurations for each discretized orientation. Comment on the given discretized values for orientation.

A1. Visualize from the given data the workspace

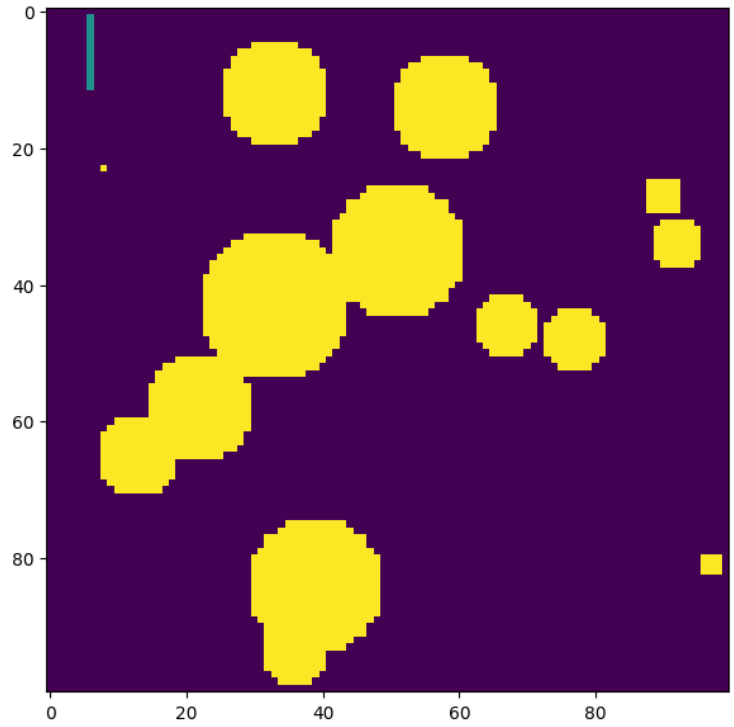


A2. Visualize from the given data the different rod configurations for each discretized orientation. Comment on the given discretized values for orientation.



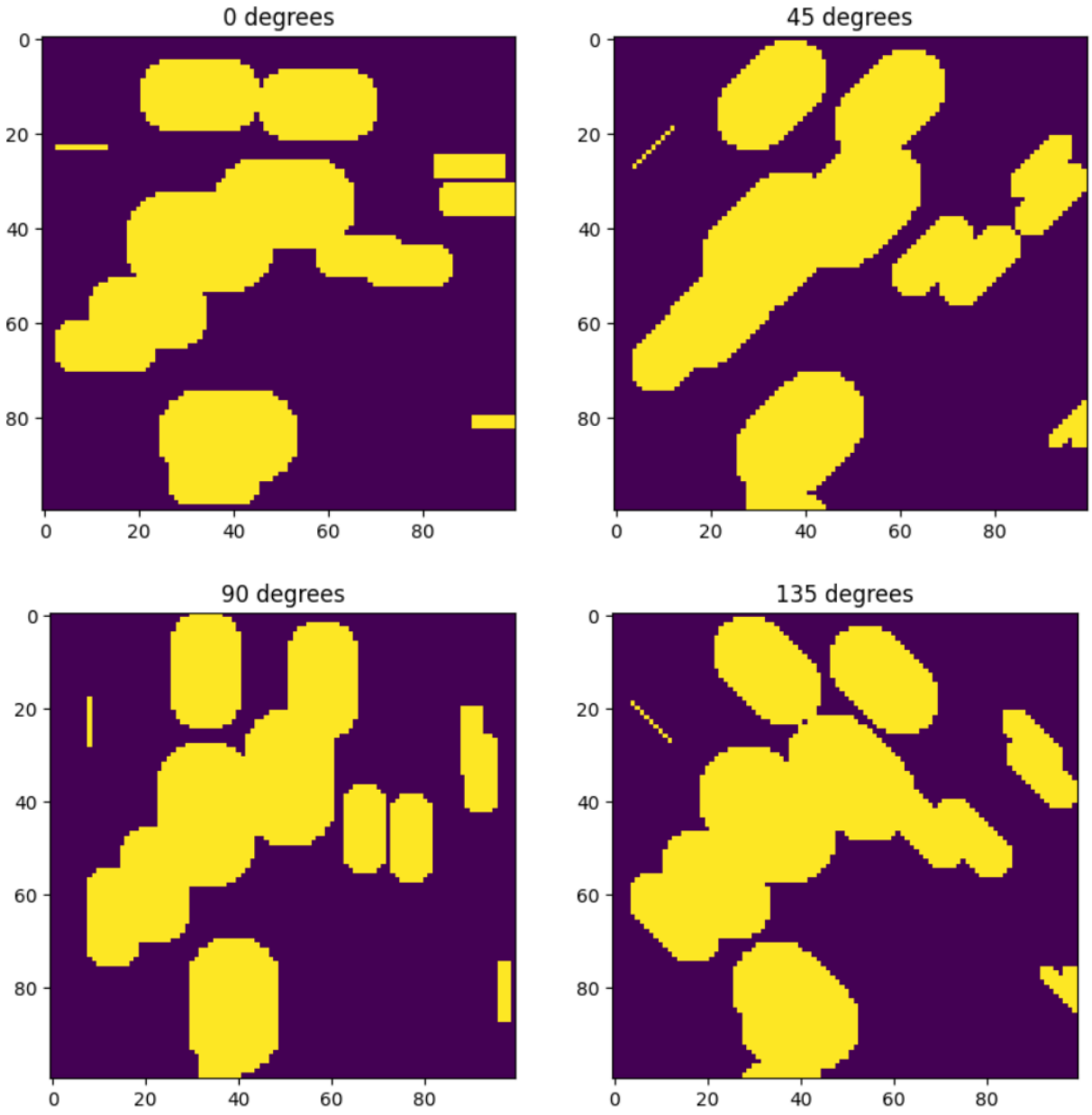


B. Visualize the environment together with the object.



Position of object is (6,6,2)

C. Create the C-space for the 2D environment map. For this, plot all the images corresponding to each of the orientations by using collision checking.



D. Comment on the obtained C-space with the previous method. What is the size of the C-space?

The obtained C-space is used to check explicitly for collisions on all configurations, with 2D convolutions of the object against the obstacle. The size of C-space is 100 × 100 with 4 dimensions.

Task 2: A star Algorithm

A. Implement the A star algorithm and plan in the generated discrete C-space from the previous task. The starting configuration of the agent is (6,6,2) and the goal configuration

is (55,55,0). On this first iteration, use an heuristic function h(q, qG) = 0, which is equivalent to the Dijkstra algorithm.

Plan is:

```
[(6, 6, 2),(6, 7, 2),(6, 8, 2),(6, 9, 2), (6, 10, 2), (6, 11, 2), (6, 12, 2),
(6, 13, 2), (6, 14, 2), (6, 15, 2), (6, 16, 2), (6, 17, 2), (6, 18, 2),
(6, 19, 2), (6, 20, 2), (6, 21, 2), (6, 22, 2), (6, 23, 2), (6, 24, 2),
(6, 25, 2), (7, 25, 2), (8, 25, 2), (9, 25, 2), (10, 25, 2), (11, 25, 2),
(12, 25, 2), (13, 25, 2), (14, 25, 2), (15, 25, 2), (16, 25, 2), (17, 25, 2),
(18, 25, 2), (19, 25, 2), (20, 25, 2), (21, 25, 2), (21, 26, 2), (22, 26, 2),
(23, 26, 2), (23, 27, 2), (24, 27, 2), (24, 28, 2), (24, 29, 2), (25, 29, 2),
(25, 30, 2), (25, 31, 2), (25, 32, 2), (25, 33, 2), (25, 34, 2), (25, 35, 2),
(25, 36, 2), (25, 37, 2), (25, 38, 2), (25, 39, 2), (25, 40, 2), (25, 41, 2),
(25, 42, 2), (25, 42, 1), (25, 43, 1), (25, 44, 1), (25, 45, 1), (25, 46, 1),
(25, 46, 0), (25, 47, 0), (25, 48, 0), (25, 49, 0), (25, 50, 0), (25, 51, 0),
(25, 52, 0), (25, 53, 0), (25, 54, 0), (25, 55, 0), (25, 56, 0), (25, 57, 0),
(25, 58, 0), (25, 59, 0), (25, 60, 0), (25, 61, 0), (26, 61, 0), (26, 62, 0),
(27, 62, 0), (27, 63, 0), (27, 64, 0), (28, 64, 0), (29, 64, 0), (29, 65, 0),
(30, 65, 0), (30, 65, 1), (30, 65, 2), (30, 64, 2), (30, 63, 2), (30, 62, 2),
(31, 62, 2), (31, 61, 2), (31, 61, 2), (33, 61, 2), (34, 61, 2), (35, 61, 2),
(36, 61, 2), (37, 61, 2), (38, 61, 2), (39, 61, 2), (40, 61, 2), (41, 61, 2),
(42, 61, 2), (43, 61, 2), (44, 61, 2), (45, 61, 2), (46, 61, 2), (47, 61, 2),
(48, 61, 2), (49, 61, 2), (50, 61, 2), (50, 60, 2), (51, 60, 2), (51, 59, 2),
(52, 59, 2), (52, 58, 2), (53, 58, 2), (53, 57, 2), (54, 57, 2), (54, 56, 2),
(55, 56, 2), (55, 55, 2), (55, 55, 1), (55, 55, 0)]
```

B. Change the heuristic function now to be h(q, qG) = L1 norm of the x, y components. Comment on the changes, how many states have been visited compared to Dijkstra? What is the final cost? Comment on the results.

Dijkstra and A* algorithm return the same path, but Dijkstra takes longer time than A*.

The final cost is 125.

C. Propose an heuristic function h(q, qG) that includes orientation. Compare metrics with the previous results. Comment on the results

C1. Propose an heuristic function h(q, qG) that includes orientation

The heuristic function can be changed to $h(q, q_G) = L2norm$

$$h(q, q_G) = L2norm = \sqrt{(x_{cur} - x_{goal})^2 + (y_{cur} - y_{goal})^2}$$

where

- $h(q, q_G)$ is a heuristic function;
- x_{cur}, y_{cur} - coordinates of current point
- x_{goal}, y_{goal} - coordinates of goal point

The same path is received. Using heuristic function with L2 norm decreases the running time. Conclude, heuristic function plays an important role in the A* algorithm, helping to find the goal faster. Of the three types h = 0, h = L1 norm and h = L2 norm, L2 norm runs the fastest and most efficiently in this problem.

Beside that, we can consider this method:

$$h(m,n,t,goal,p) = [w_0 + w_1 h_1(m,n,t) + w_2 h_2(p,m,n)](h_0(n,goal))$$

where

- h(m,n,t,goal,p) is a heuristic function;
- m - starting grid point;
- n - current neighbor;
- t steps explored in the forward direction;
- goal - goal grid point;
- p - previous explored node
- $h_0(n,goal) = \|n - goal\|^2$
- $h_1(m,n,t) = 1$ or 0 - is a binary variable that is 1 if moving from m for t steps along the direction of m contains an obstacle.
- $h_2(m,n,t) = 1$ or 0 - is a binary variable that is 0 if p, m and n of the vehicle's previous heading and the next are in the same direction, and 1 if they are not aligned.
- The weights w_0, w_1 predict the possibility for a turn and therefore avoids that path altogether.