**ROBOT NAVIGATION:**

**Part #1 of Mini-Project:**

|  |  |  |
| --- | --- | --- |
| Gal Gilat |  |  |
| Alexander Vasilyev |  |  |
| Ofek Nachshoni | 212594527 | ofekn@campus.technion.ac.il |

Table of Contents:

[Question 1 3](#_Toc202477730)

[Question 2 6](#_Toc202477731)

[Question 3 9](#_Toc202477732)

# Question 1

We computed configuration space (C-space) obstacles for a convex polygonal robot navigating around a single convex polygonal obstacle. We implemented the slice-based method described in Latombe’s algorithm:

1. Defined the robot and obstacle as lists of vertices ordered counterclockwise.
2. For each of 32 regularly spaced orientation values , we:
   * Rotated the robot by
   * Reflected the robot about the origin
   * Computed the Minkowski sum of the reflected robot and the obstacle
   * Extracted the convex hull of the result to represent the obstacle in C-space for
3. Plotted the resulting polygonal C-obstacle slices in the plane for each , and printed the results for slices 1, 8, 16, and 32.

This allowed visualization of how the robot’s configuration constraints change with orientation relative to the obstacle.



figure 1: Map of Obstacles



figure 2: Layer 1



figure 3: Layer 8



figure 4: Layer 16



figure 5: Layer 32

# Question 2

In Question 2, we extended the solution from Question 1 to handle a realistic environment with multiple convex polygonal obstacles. The environment consists of outer walls (B01–B04), internal walls (B1–B5), and doors (B6–B7). We used the same algorithm as in Question 1 to compute the configuration space (C-space) obstacle slices for each obstacle individually over 32 discrete rotation angles of the robot.

For each orientation , we computed the Minkowski sum of the rotated robot with every obstacle and extracted the convex hull to obtain the C-obstacle slice. We then overlaid the slices for all obstacles in a single plot for θ-layers 1, 8, 16, and 32. This provided a layered visualization of the full configuration space and demonstrated how the robot’s collision constraints evolve with rotation in a complex environment.



figure 6: Layer 1



figure 7: Layer 8



figure 8: Layer 16



figure 9: Layer 32

# Question 3

In question 3, we took the printouts of layers 1,8,16,32 and put a “1” value for each cell with the c-obstacles boundaries. All other cells were filled with “0” value. Moreover, we chose to represent this via colors- “1” value is black and “0” value is white.

Here are the grids we got for each layer:



figure : Layer 1



figure : Layer 8



figure : Layer 16



figure : Layer 32