**Objective:**

Design and implement an autonomous navigation algorithm for a robot. The robot, given the coordinates of the four vertices of a rectangular pillar, must find the center point of the pillar and navigate to this center. The robot must follow a path that avoids diagonal movements, calculating its displacements accordingly.

**Setup:**

* The rectangular pillar is defined by its four corners.
* The robot starts at a known position outside the boundary of the pillar.

**Tasks:**

* Calculate the center point of the rectangular pillar using the given corners.
* Develop an algorithm that allows the robot to navigate from its starting position to the center point of the pillar.
* The robot can only move in straight lines parallel to the x or y axes (no diagonal movements allowed).
* The robot must avoid any collision with the pillar while navigating to the center.

**Constraints:**

* The robot cannot move diagonally to the center point.
* The robot must determine an optimal path that avoids the pillar.
* The robot should minimize the total distance traveled to the center point.
* The robot must calculate its displacement and move step-by-step, updating its position iteratively.

**Deliverables:**

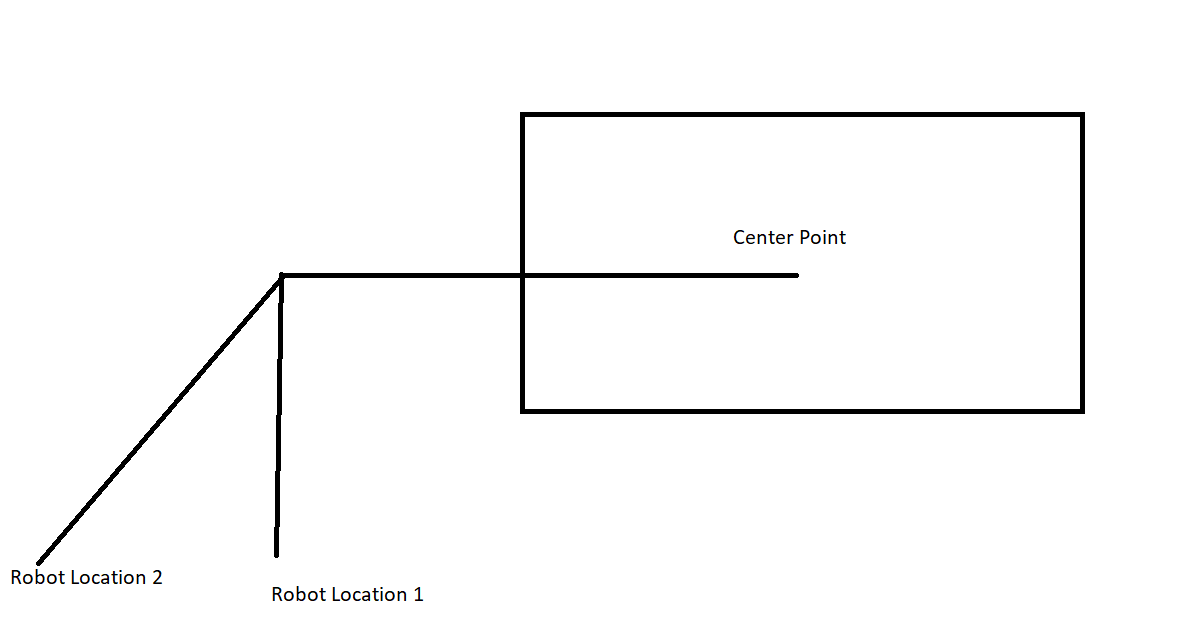
* A simulation or real-time demonstration showing the robot successfully navigating to the center of the pillar.
* A documented algorithm explaining the logic and steps taken to calculate the center and navigate to it.
* A presentation detailing the approach, challenges faced, and solutions implemented.

**Evaluation Criteria:**

* Accuracy in calculating the center point of the pillar.
* Efficiency and correctness of the navigation algorithm.
* Successful avoidance of diagonal movements and pillar collision.
* Innovation in approach and algorithm design.
* Clarity and comprehensiveness of the documentation and presentation.

**Tools and Technologies:**

* Any programming language suitable for robot navigation (e.g., Python, C++).
* Real-world implementation can use ROS-compatible robots like TurtleBot for demonstration purposes or you can use PyGame simulation but ROS is well and better .



Example of Robot Movement