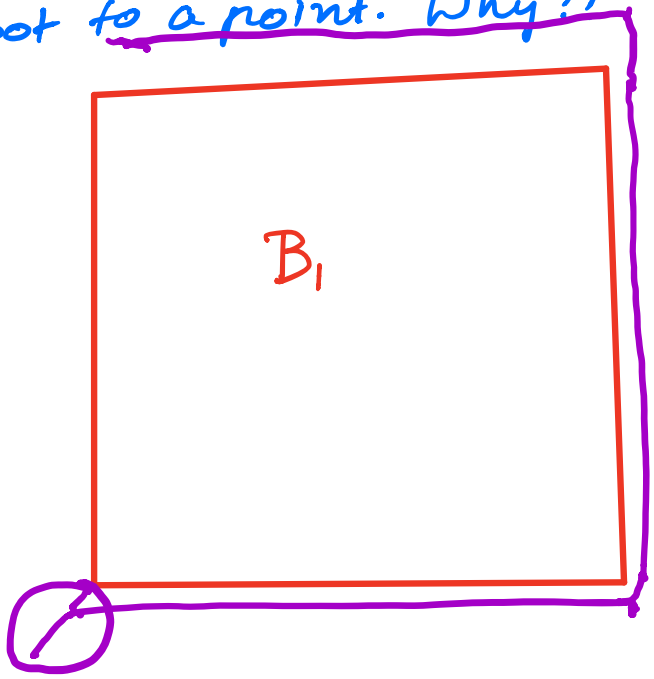
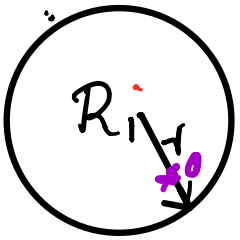
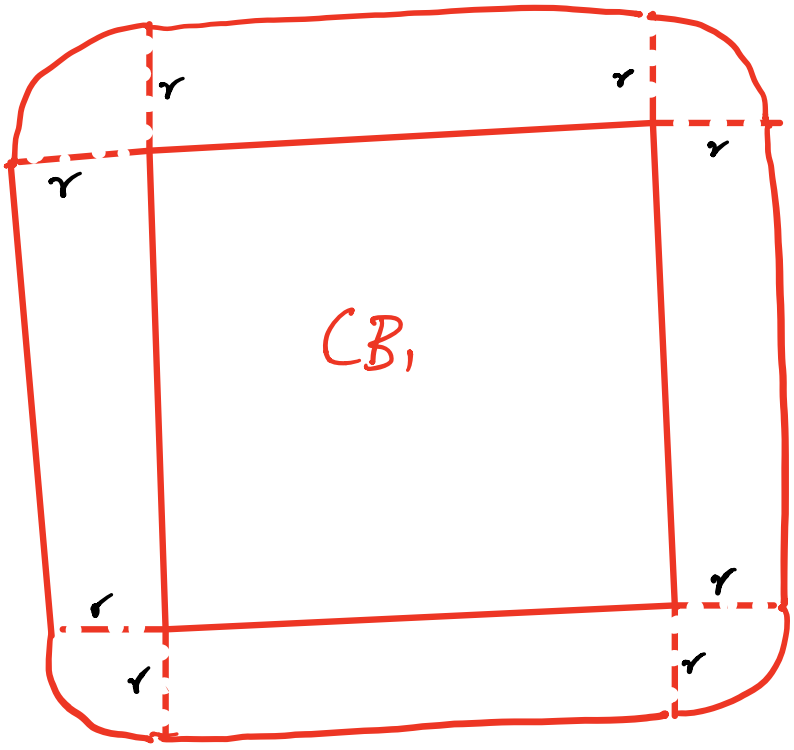
CONFIGURATION SPACE:

Main Idea: Grow the obstacle by the robot's size. Reduce the robot to a point. Why??







CB_1: The set of all points for which $B_1 \cap R_1 = \phi$.

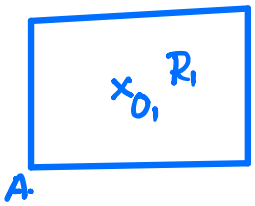
CB_1: The configuration space of the obstacle B_1

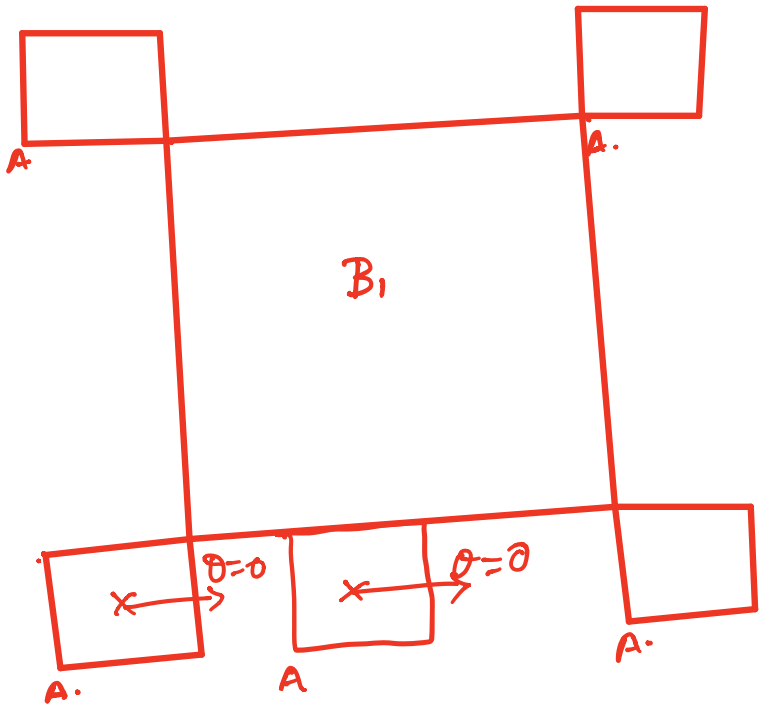
How to obtain CB_1: Trace the locus of all points for which the robot R_1 just grazes the obstacle B_1. The locus is always traced with respect to a fixed point in R_1. In

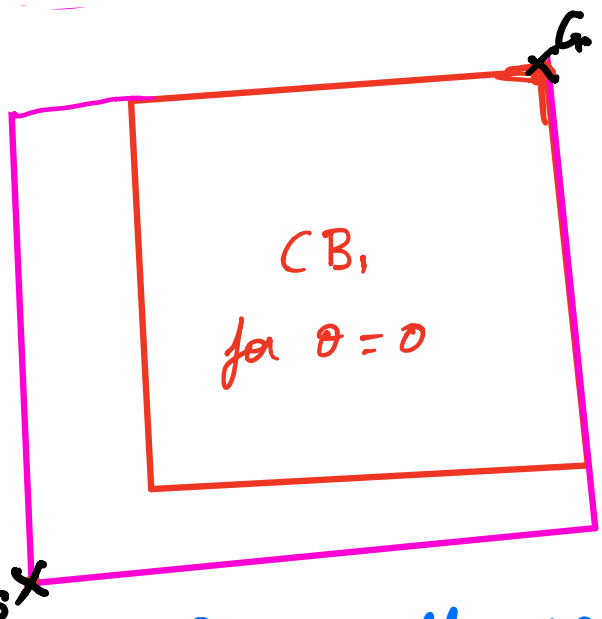
this case it is traced with respect to the center of R_1, say O_1.

How to obtain CB_1 for a non-circular robot.

Say rectangular :







The red quad is the original obstacle B_1

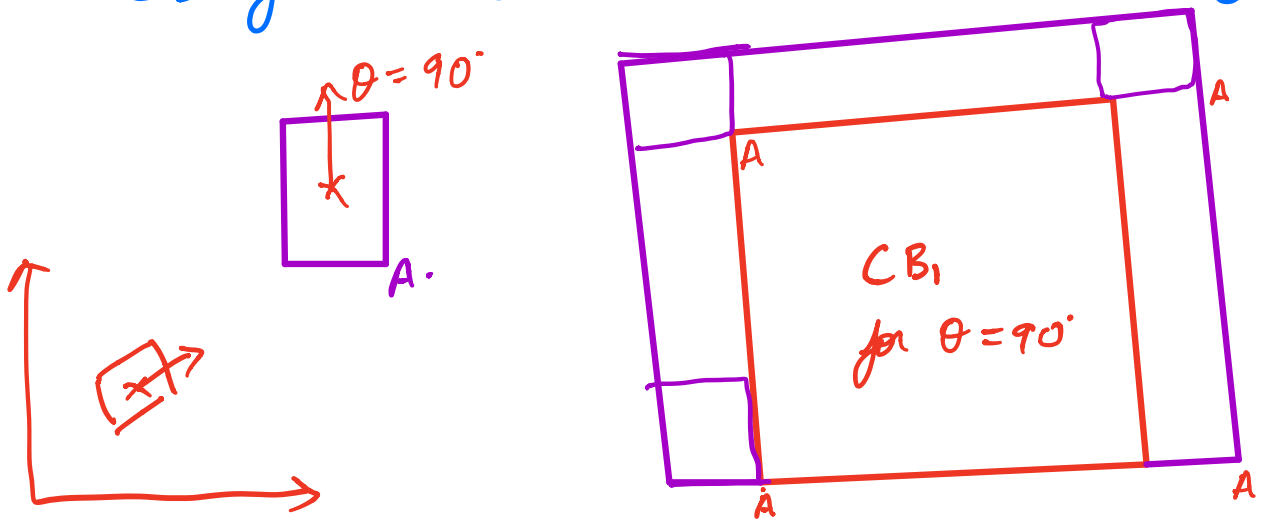
The pink lines trace the locus of the point A for which R_1 grazes B_1

CB_1 is the grown obstacle.

How is CB actually coded: There are libraries that compute CB for a two dimensional /planar robot that can be considered a circle or a polygon.

What is the dimension in which CB resides?

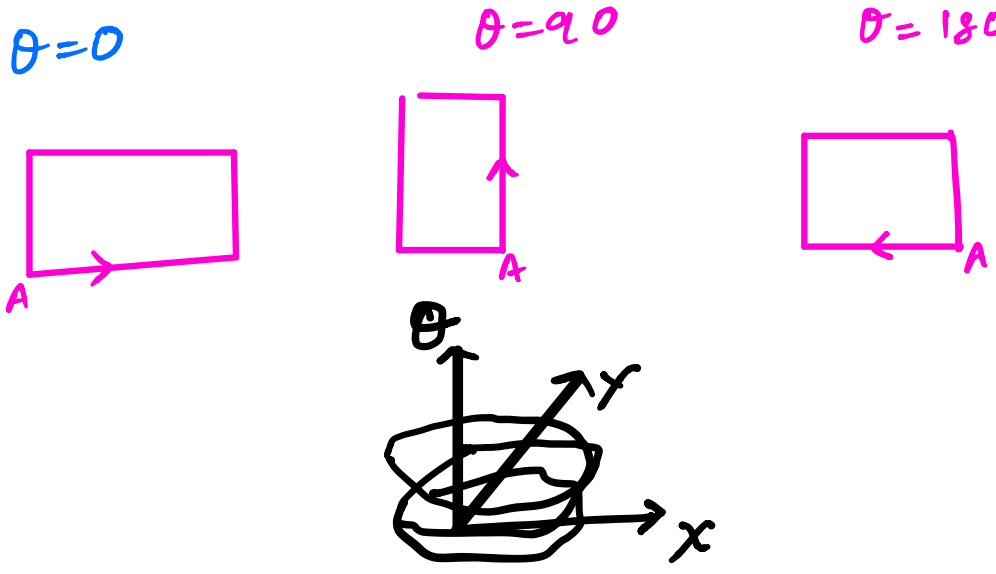
CB for a different orientation of R_1



Now what is the dimension of CB_1

- The set of all points of A for which R_1 grazes B_1.

- For different orientations of R_1 we get different $CB(\theta(R_1))$ where $\theta(R_1)$ represent the orientation of R_1 wrt a global reference frame



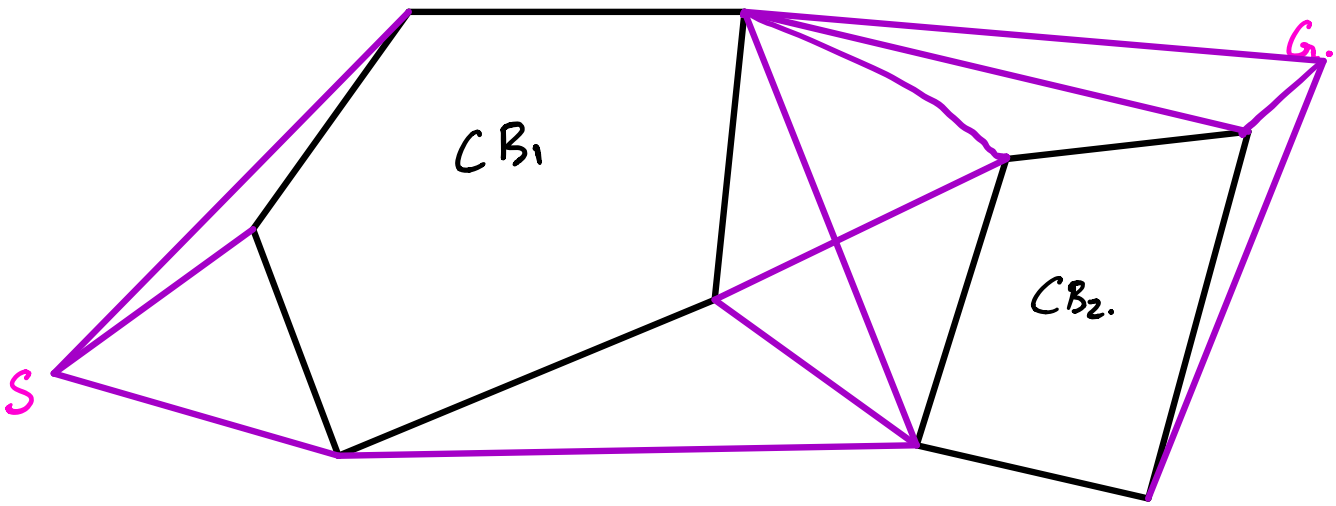
So what is the dimension of CB for a circular R_1. ?
polygonal (convex) R_1 ?

- The dimension of CB is the same as the number of degrees of freedom of R.

→ So what is the problem / are the problems due to the CS approach?

→ How do you make use of the CBs to compute a trajectory for a point robot?

The VISIBILITY GRAPH:



Connect all vertices of the CB's that are visible to one another.

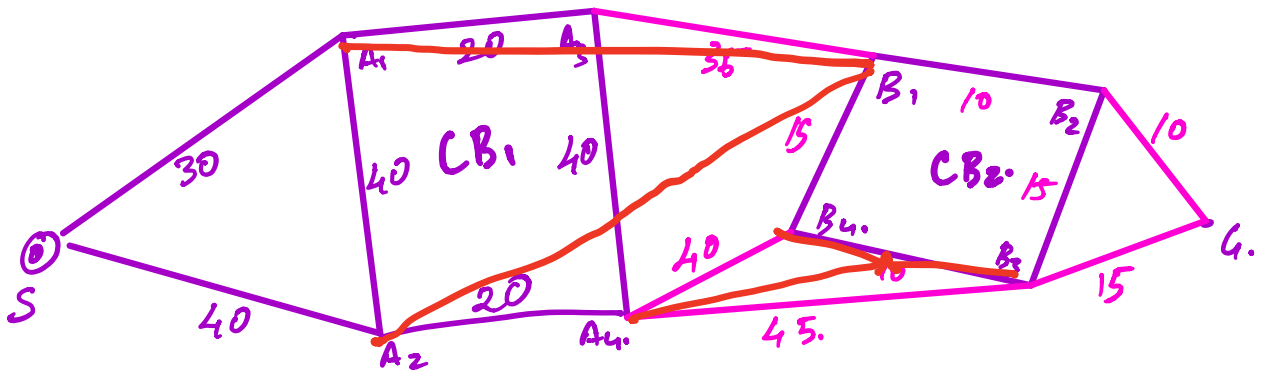
Connect the S & G vertices to the vertices of CB that are visible from S & G

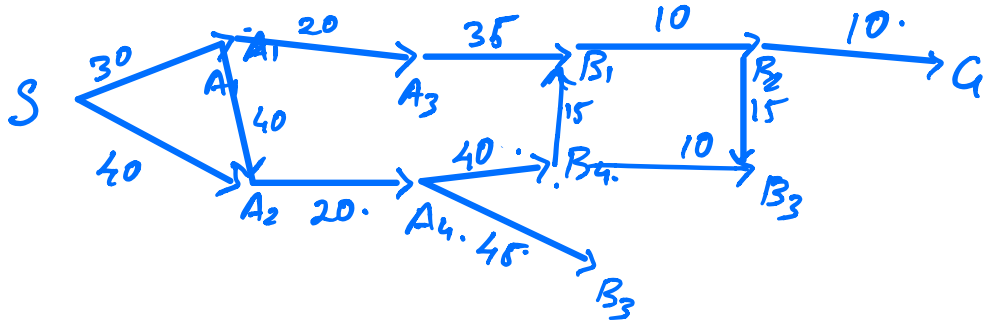
→ The resulting data structure is **Visibility Graph.**

→ Let the weight of the edges be the Euclidean distance between the two visible nodes (visible to one another).

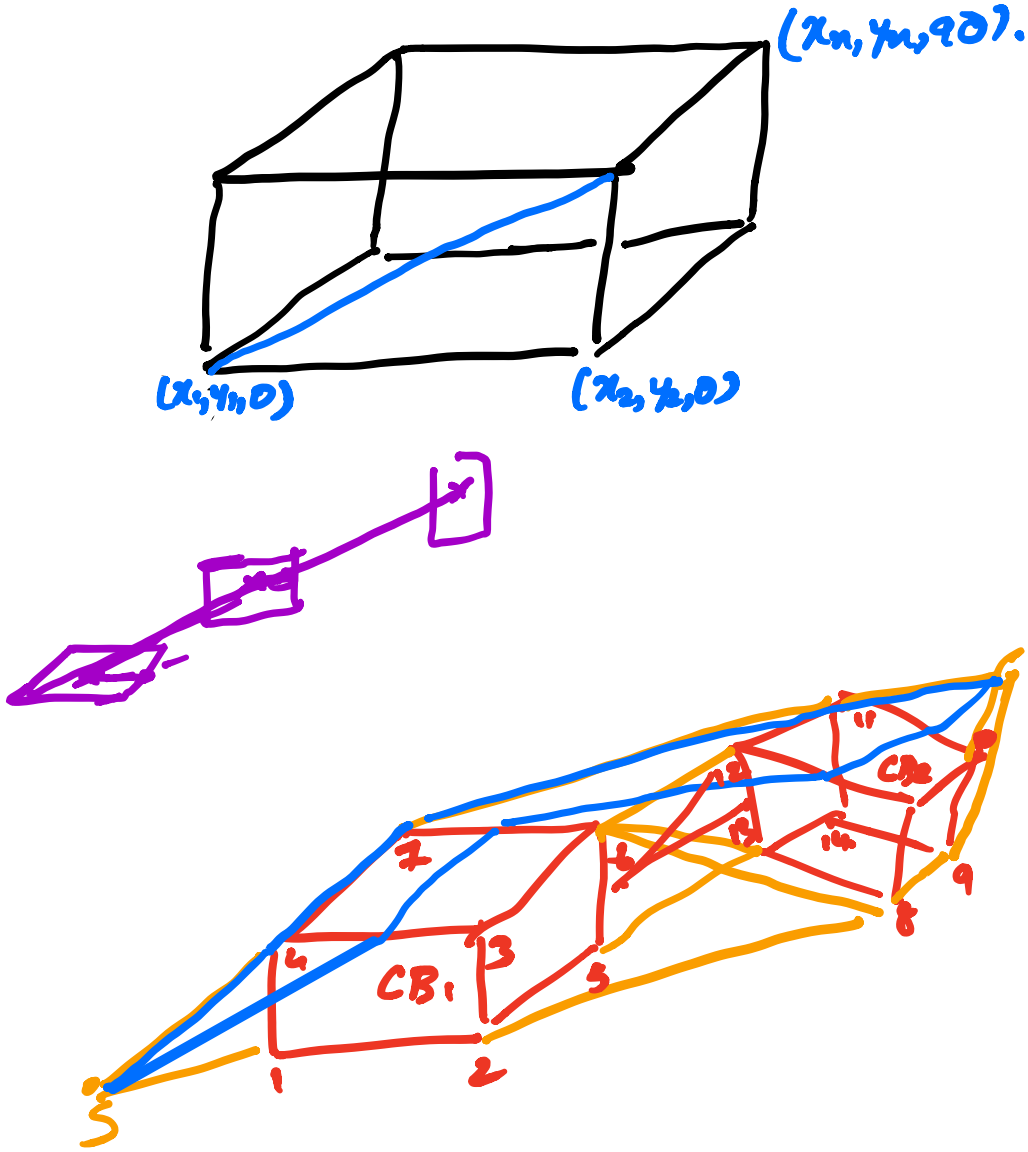
→ The shortest path lies on this graph.

→ Can be obtained by Djkstra's algorithm.





$S \rightarrow A_1 \rightarrow A_3 \rightarrow B_1 \rightarrow B_2 \rightarrow G.$


(x1, y1, 0)
(x2, y2, 0)
(xn, yn, 90).
CB1
CBn
1
2
3
4
5
6
7
8
9
10
11
12
13
14