# Problem Specification

A robot path planning problem P in this domain is defined as P = {S, s0, g, L, f, O, D, c}, where:

• S is the state space is represented as a 5-tuple 6 s = (x, y, θ, v) of x and y location, heading, speed, and time.

• s0 is our starting state.

• g is our agent’s goal pose as a 4-tuple g = (x, y, θ, v)

• L is the set of all actions a that are possible.

• The function f is defined as f : S → A, such that ∀s ∈ S, f(s) = Lfeas where Lfeas ⊆ A is the set of dynamically feasible actions such that ∀a, a ∈ Lfeas , the action a can be executed given the input state s.

• O is the set of stationary obstacles identifying whether a given coordinate is blocked or not. The entire map is represented in x,y coordinates

• D is the set of dynamic obstacles represented as a series of Gaussian distributions indexed by time. 7

• c is the cost function.

# Objects in World:

The world is defined as a 2D world in which . The world is described using:

* Position (P): Set of all points (x,y) in W.
* Orientation (Or):
* Velocity(V):
* Acceleration(a):
* CircleModel(C): A circle approximation of the robot to maintain safe distance from other objects.
* Trajectory(T): A set of points in state space that are the future states resulting from a given initial state.
* Obstacles (O): Areas in the 2D world that are occupied by any obstacle. This can be defined as such that the obstacle region i.e. O is the set of all points that lie in W and lie in one or more obstacles. O is represented as an m-side polygon which can be defined using vertices and edges. Hence the polygon can be defined as a series of vertices, along with an intersection of m half-planes. Each half-plane corresponds to the points that lie to one side of a line. Given Hi as a half plane, a convex m-sided polygon is expressed as
* Robot(R): Bodies that are modeled as polygons and are able to move in the given world based on a certain plan and motion model.

Here, both obstacles and robots are subset of the world W.

# Motion Model:

# State Variables:

Location(R)= robot’s current location. Range(Location(R))=Position

Pose(R)=robot’s current orientation/pose. Range(Pose (R))=Orientation

Acceleration(R)=robot’s current acceleration. Range(Acceleration (R))=Acceleration

Velocity(R)= robot’s current velocity. Range(Velocity(R))= Velocity

# Methods:

Change Pose

Locate Robot  
GenerateTrajectory

CheckTrajectory

For all(x,y,theta) check x-x + y-y + r theta-theta < d

Move

DetectLocation

Change Velocity

Stop

m-opendoor(r, d, l, h):

task: opendoor

pre: loc(r) == l && adjacent(l, d) && handl(d, h)

body:

while !reachable(r, h) do

move-close(r, h)

end

monitor-status(r, d)

if door-status(d) == closed then

unlatch(r, d)

end

throw-wide(r, d)

end-monitor-status(r, d)