Design

# Nodes

**MapWalker** – Traverses the map identifying Targets that need to be moved to the jail. Sends Target Centroids to PushPlanner addTarget service. Stays in run mode until whole map has been traversed and all Targets identified. Then it goes into idle state releases control to the PushExecutor node.

**PushPlanner –** Initialized with coordinates of Jail. Receives Targets centroids and calculates path plans to push them to the jail. Offers services for adding Targets, updating Targets (used during pushing), listing known Targets, and getting path plans. Subscribes to /target\_loc topic for updating Target locations

**PushExecutor** – Node for executing the push plans. Stays in idle state until MapWalker node has finished identifying Targets. Recieves push plans from the PushPlanner node. Publishes updated Target location to /target\_loc topic. If push plan fails, requests a new or updated push plan.

# Functional Requirements

## MapWalker walk strategy

Given a known Map and a set of unidentified Targets  
When the MapWalker is started and configured to perform a COMPLETE walk  
Then is should perform localization  
And start to traverse the entire map  
And track the areas that it has visited

## MapWalker identifies centroid of Targets

Given a known Map and a set of unidentified Targets  
When the MapWalker identifies a Target  
Then the MapWalker should identify the center of the object as the centroid  
And send this information to the NewTarget service in the PushPlanner node

## MapWalker notifies system that walk is complete

Given a known Map with identified Targets  
When the MapWalker completes the walk  
Then it should notify the system so that the Target moving can begin

## Push Planner New Target

Given the push planner node NewTarget service is running  
When a new Target arrives  
Then the Target should be compared to see if it occupies the space of known Targets  
And when found to be unique it should be assigned a unique ID and added to the queue for a push plan  
And when it is found to be a known Target it should be discarded  
And when it is found to exist in the Jail it should be discarded

## Push Planner Update Target

Given the /target\_loc topic is being published to  
When a Target Update arrives  
Then the Target centroid coordinates should be updated

## Push Planner New Target Already Exists

Given the push planner node NewTarget service is running  
When a new Target arrives with a centroid < .5 meter from another single target  
Then the Target should be considered to be the same as the other one  
And no new Target should be added to the Target list

## PushPlanner Calculate Plan

Given the PushPlanner has at least on Target in the queue  
When the queue is checked for work  
Then the PushPlanner should calculate a start and end pose that will allow a robot to push the target to the jail  
And the PushPlanner should store this information for later retrieval in priority of shortest distance to jail

## PushPlanner Get Plan

Given the PushPlanner has calculated at least on PushPlan  
When the GetPushPlan service is called  
Then the PushPlanner should return the PushPlan with the shortest distance to the jail first

## PushExecutor wait for work

Given the MapWalker node is still running  
When the PushExecutor starts  
Then it should wait in a until the StartPushing service has been called

## PushExecutor execute push plan

Given the PushExecutor is in the Pushing state  
And there is at least one PushPlan availiable  
When the PushExecutor requests and receives a PushPlan from the GetPushPlan service  
Then the PushExecutor should navigate the the start pose specified in the plan  
And begin to push the target toward the end pose  
And send updates on the /target\_loc topic as to the centrod of the target as it moves  
And detect when the target is not heading toward the desired destination

## PushExecutor target off track

Given the PushExecutor is pushing a target  
When the PushExecutor detects that the target is no longer aligned to be pushed toward the end goal  
Then the PushExecutor should recalculate the centroid of the target  
And send an request for a new PushPlan to the PushPlanner GetPushPlan service  
And execute the new PushPlan

# Message and Services Needed

## Messages

Target  
- header : Header  
- centroid  
 - x : float  
 - y : float  
- id : string  
  
PushPlan  
- start : geometry\_msgs/Pose  
- goal : geometry\_msgs/Pose  
- target: Target

## Services

**MapWalker**

SetMapWalkerState – STOPPED=0, WALKING=1  
if walking map, stop, if not do nothing

**PathPlanner**

NewTarget – For adding new Targets to the list of known Targets  
- input: Target  
- returns: Target (with ID assigned)

GetPushPlan – Returns a plan for pushing a given Target to the Jail  
- input : n/a  
- returns PushPlan

**PushExecutor**

SetPushExecutorState – STOPPED=0, PUSHING=1  
if in idle state, wake up and request a new PushPlan, in not idle do nothing  
StopPushing – if pushing a Target stop, if not do nothing