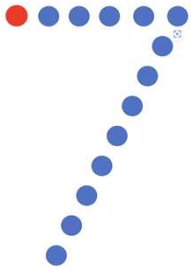
SEMTRON

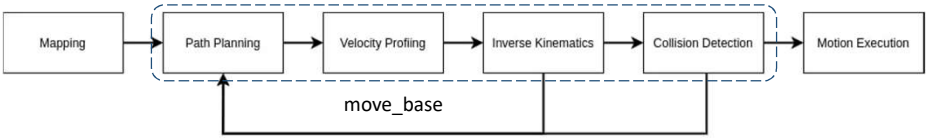
SEVEN

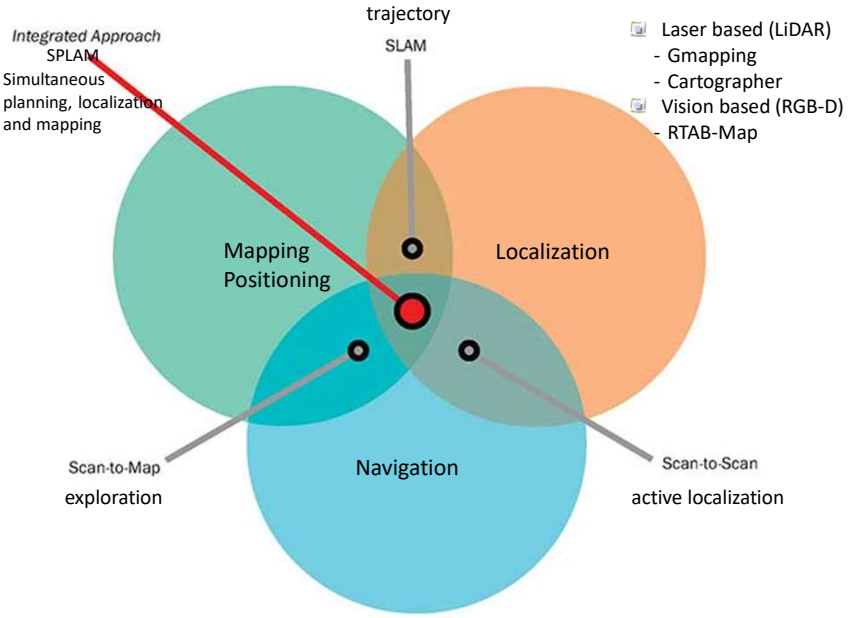
Autonomous Exploration



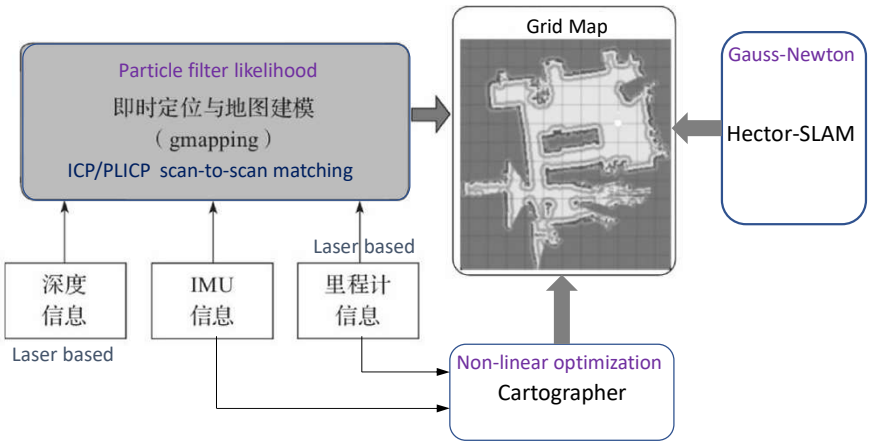
SEMTRON

SLAM (simultaneous localization and mapping)

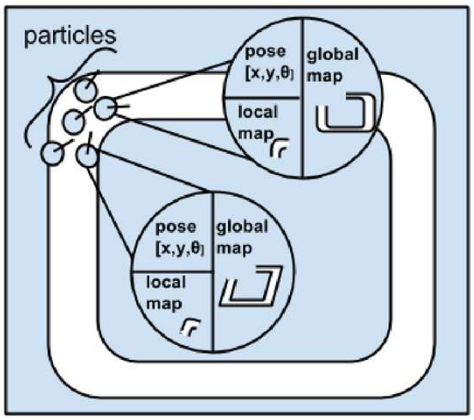


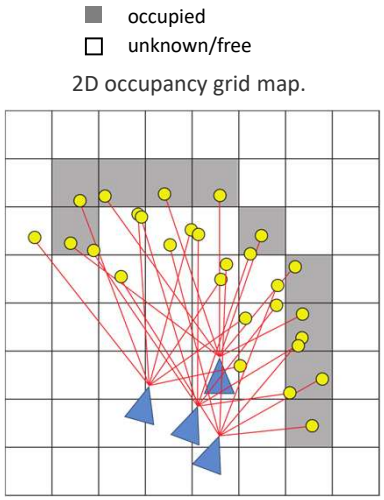


The purpose of the LiDAR odometry is to produce a local map by creating an estimate of the motion between two neighboring point cloud frames



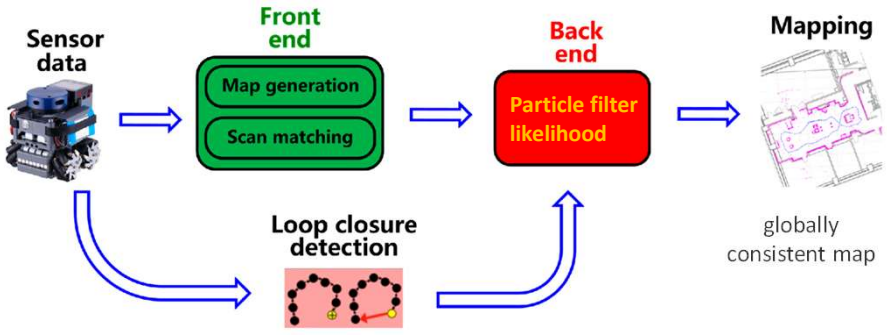
Laser-based SLAM



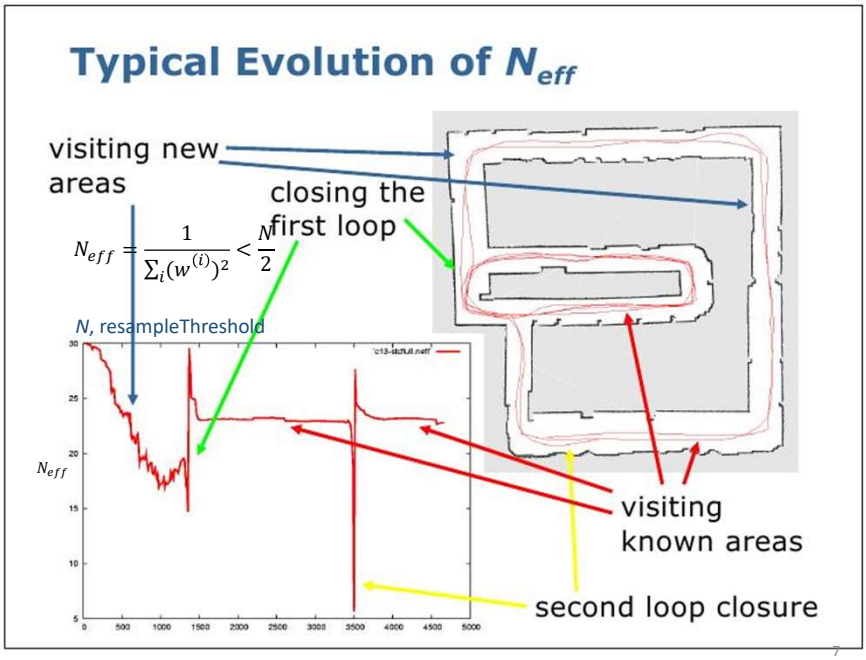


Grid-based scan-to-scan matching

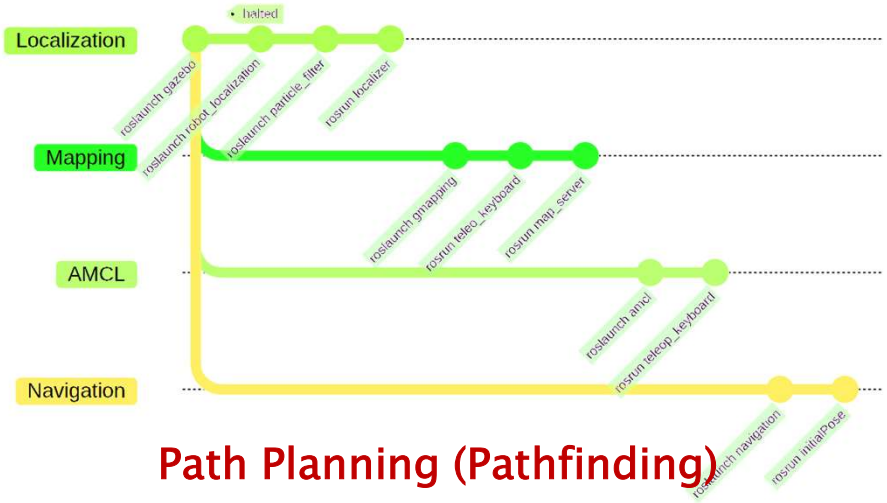
Global data association corrects cumulative mistakes by recognising if the robot has reached the location it has arrived at in the historical instant



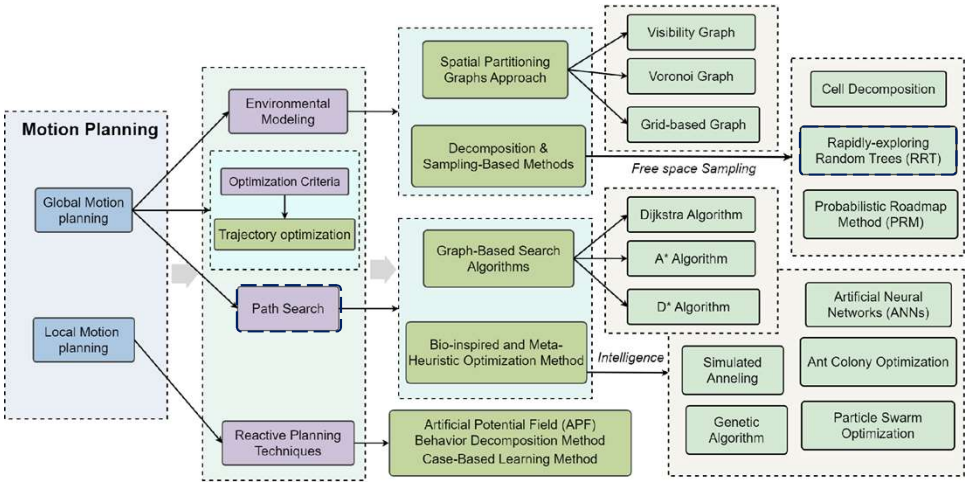
Combined particle filter and scan-matching SLAM



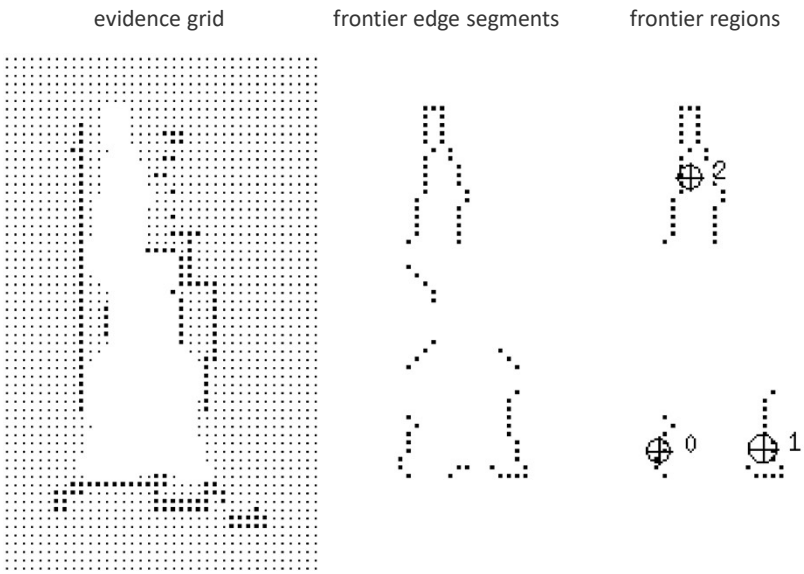
SEMTRON



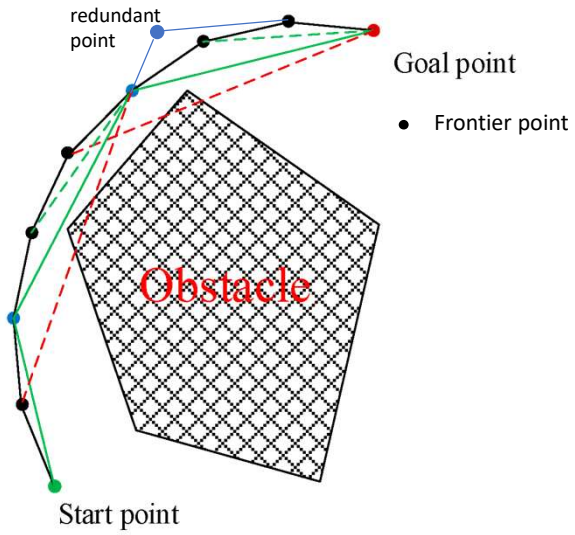
Path Planning (Pathfinding)



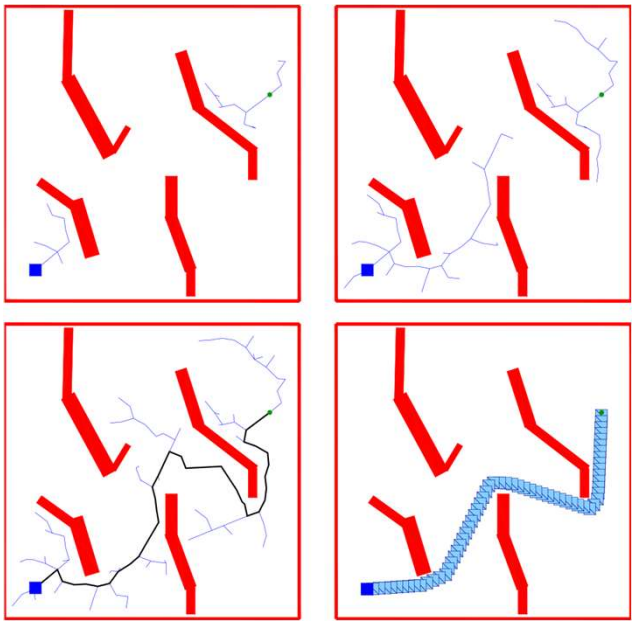
Motion planning techniques in robotics



Frontier detection in pathfinding

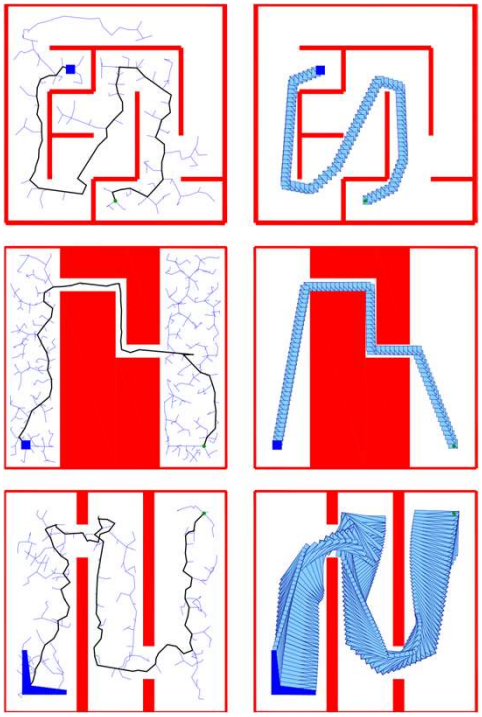


Frontier detection in RRT



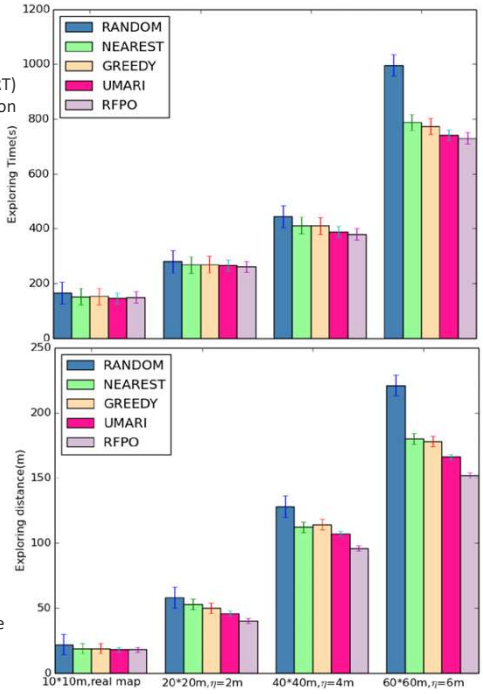
RRT pathfinding

RRTs

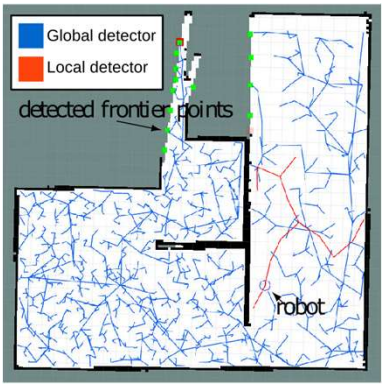


Final Path

Umari and Mukhopadhyay (RRT) random frontier points' optimization

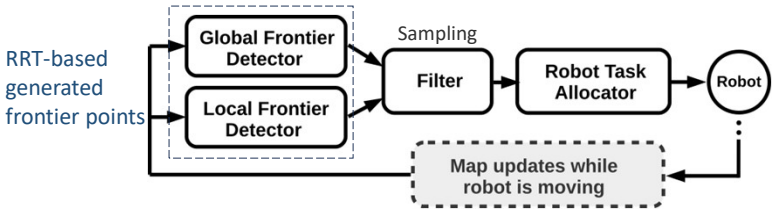


η represents the growth rate of the tree



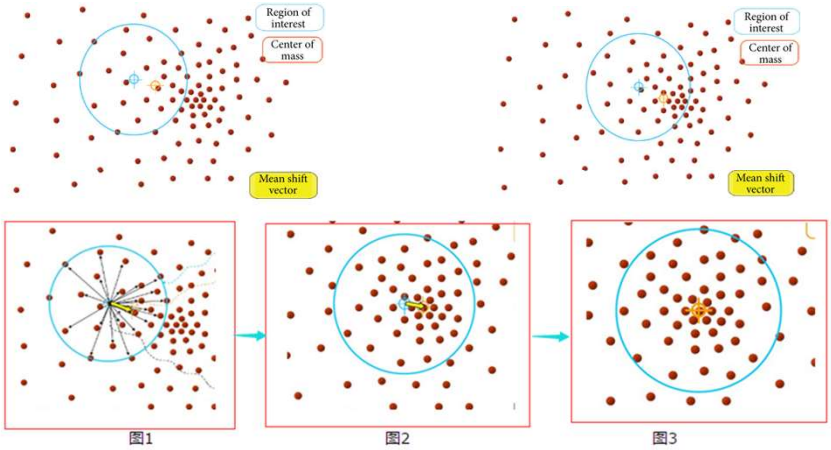
selection of the frontier point with the highest evaluation value (information gain and exploration cost) - the key to efficient exploration

path planning to the selected clustered frontier points (frontier region) from the filter module



RRT autonomous exploration

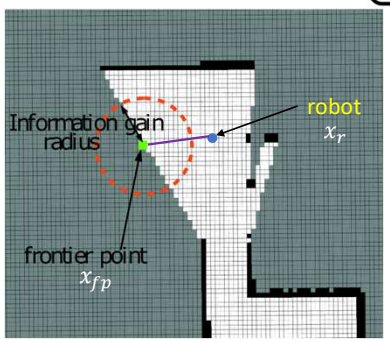
The filter module clusters the frontier points with mean shift clustering algorithm and stores them. The module also deletes invalid and old frontier points.

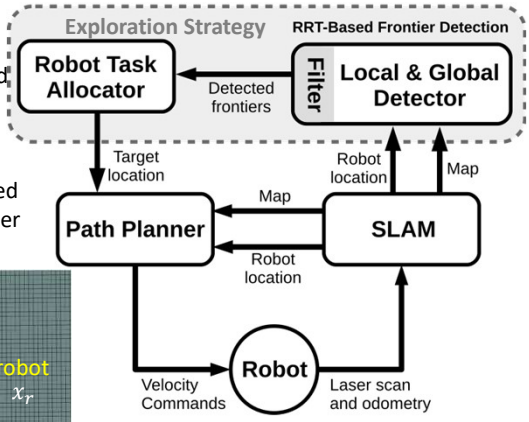


Sampling mean shift algorithm

expected distance to be traveled by a robot to reach a frontier point (*N*)

area of unknown region expected to be explored for a given frontier point (*I*)

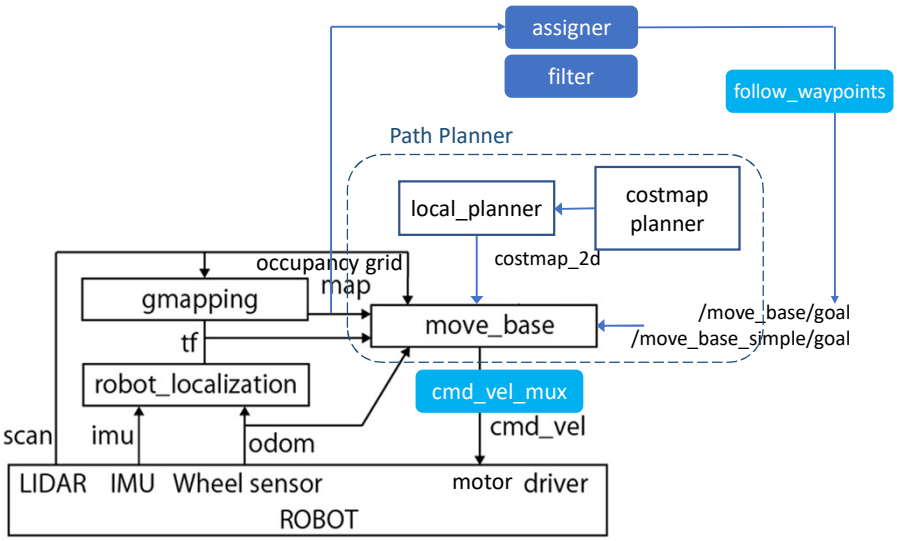




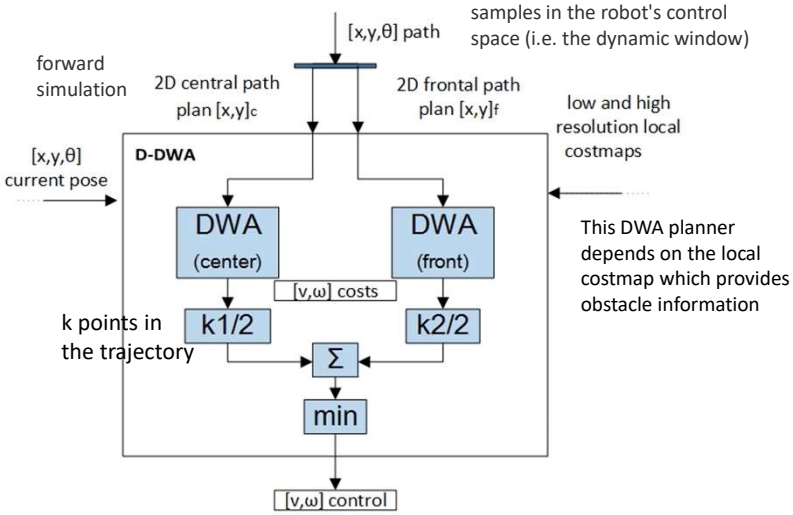
$$R(x_{fp}) = \lambda h(x_{fp}, x_r) I(x_{fp}) - N(x_{fp})$$

$$h(x_{fp}, x_r) = \begin{cases} 1, & \|x_r - x_{fp}\| > h_{rad} \\ h_{gain}, & h_{gain} > 1 \end{cases}$$

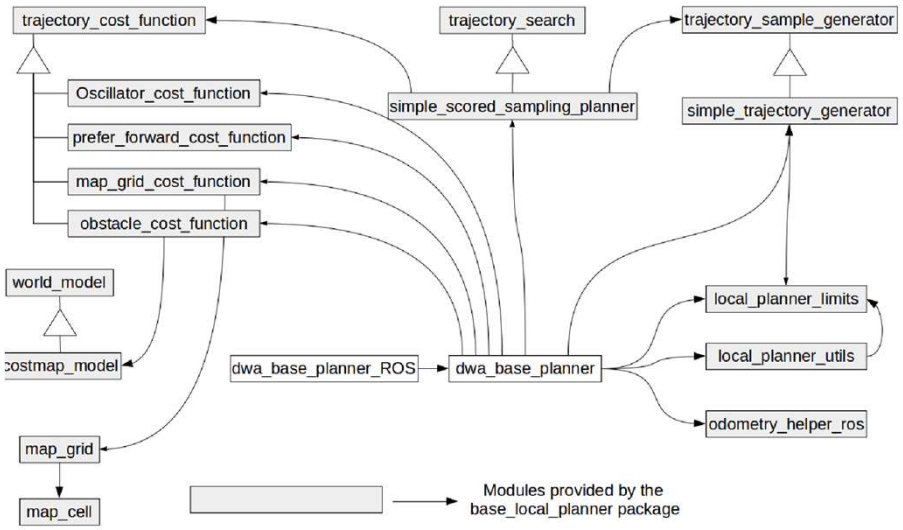
RRT autonomous exploration



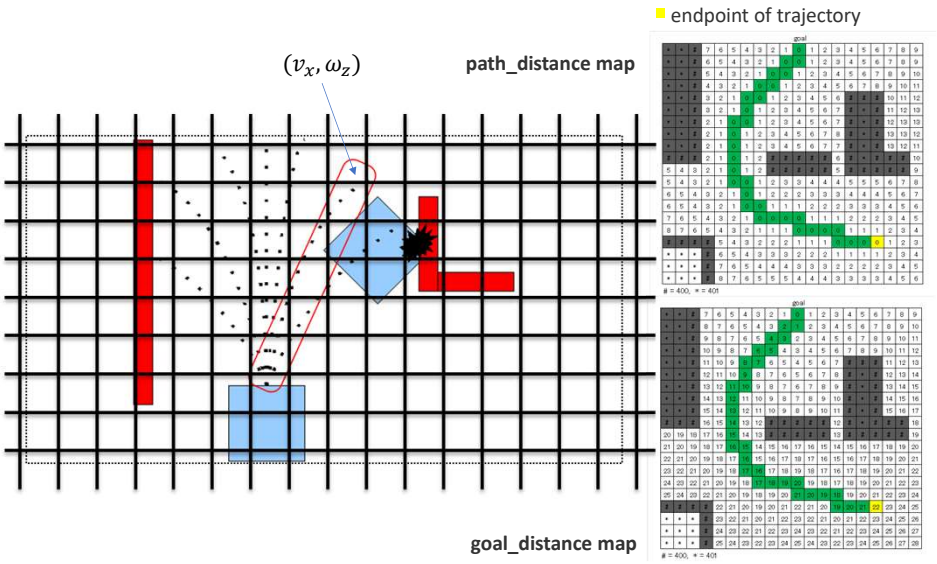
RRT autonomous exploration



Dynamic Window Approach local planner



dwa_local_planner Plugin



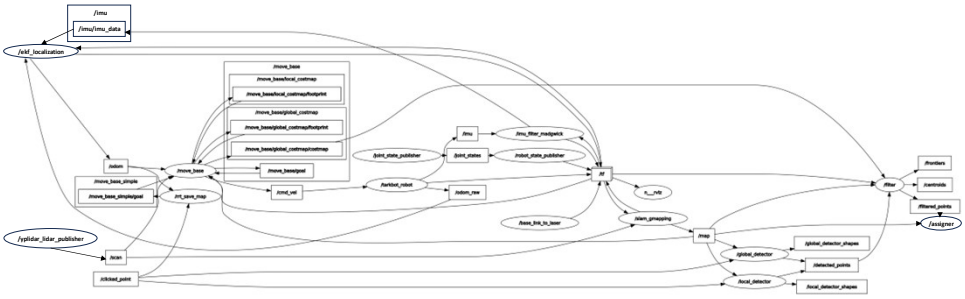
dwa_local_planner Plugin

DWA maximizes an objective function that depends on

(1) the progress to the target,
(2) clearance from obstacles, and
(3) forward velocity to produce the optimal velocity pair.

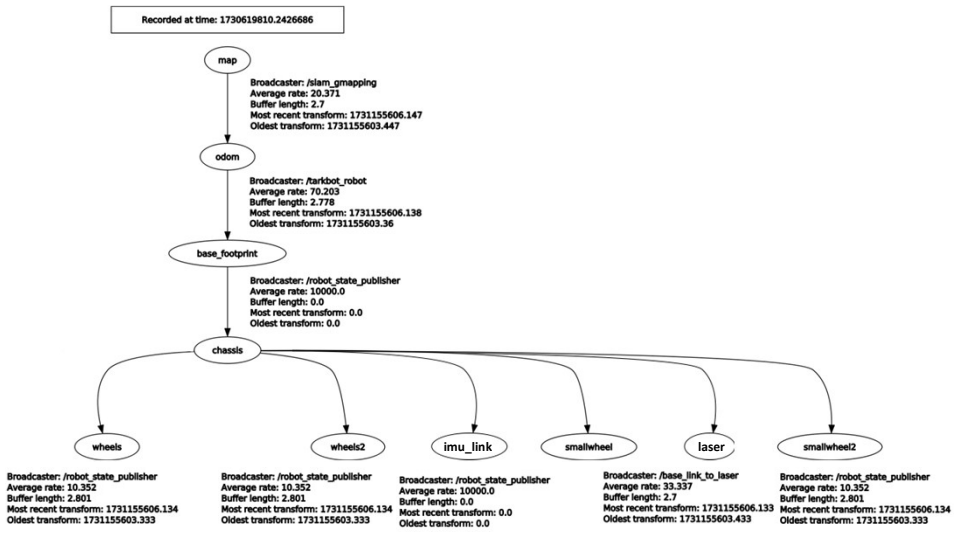
cost = path distance bias ∗ (distance(*m*) to path from the endpoint of the trajectory)
+ goal distance bias ∗ (distance(*m*) to local goal from the endpoint of the trajectory)
+ occdist scale ∗ (maximum obstacle cost along the trajectory in obstacle cost (0-254))

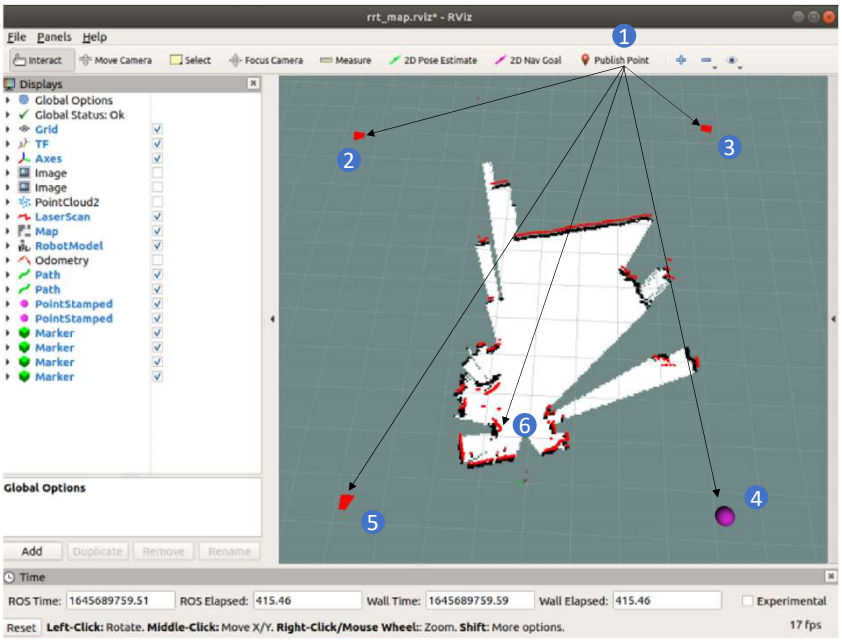
dwa_local_planner Plugin



* /move_base/DWAPlannerROS/global_plan [nav_msgs/Path]
* /move_base/DWAPlannerROS/local_plan [nav_msgs/Path]
* /move_base/DWAPlannerROS/cost_cloud [sensor_msgs/PointCloud2]
* /move_base/DWAPlannerROS/trajectory_cloud [sensor_msgs/PointCloud2]

* /move_base/NavfnROS/plan [nav_msgs/Path]




rrt_map.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Global Status: Ok
Grid
TF
Axes
Image
Image
PointCloud2
LaserScan
Map
RobotModel
Odometry
Path
Path
PointStamped
PointStamped
Marker
Marker
Marker
Marker
Global Options
Add
Duplicate
Remove
Rename
Time
ROS Time: 1645689759.51
ROS Elapsed: 415.46
Wall Time: 1645689759.59
Wall Elapsed: 415.46
Experimental
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
17 fps
1
2
3
4
5
6