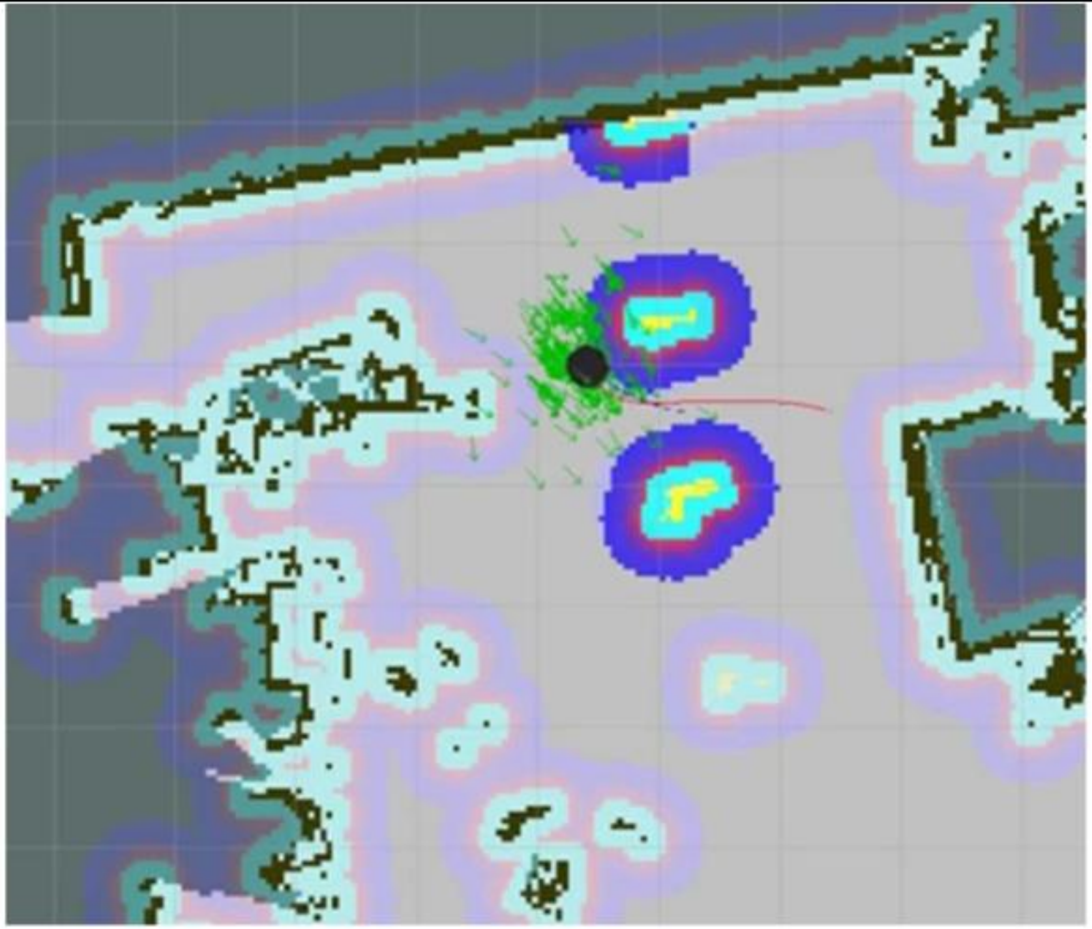


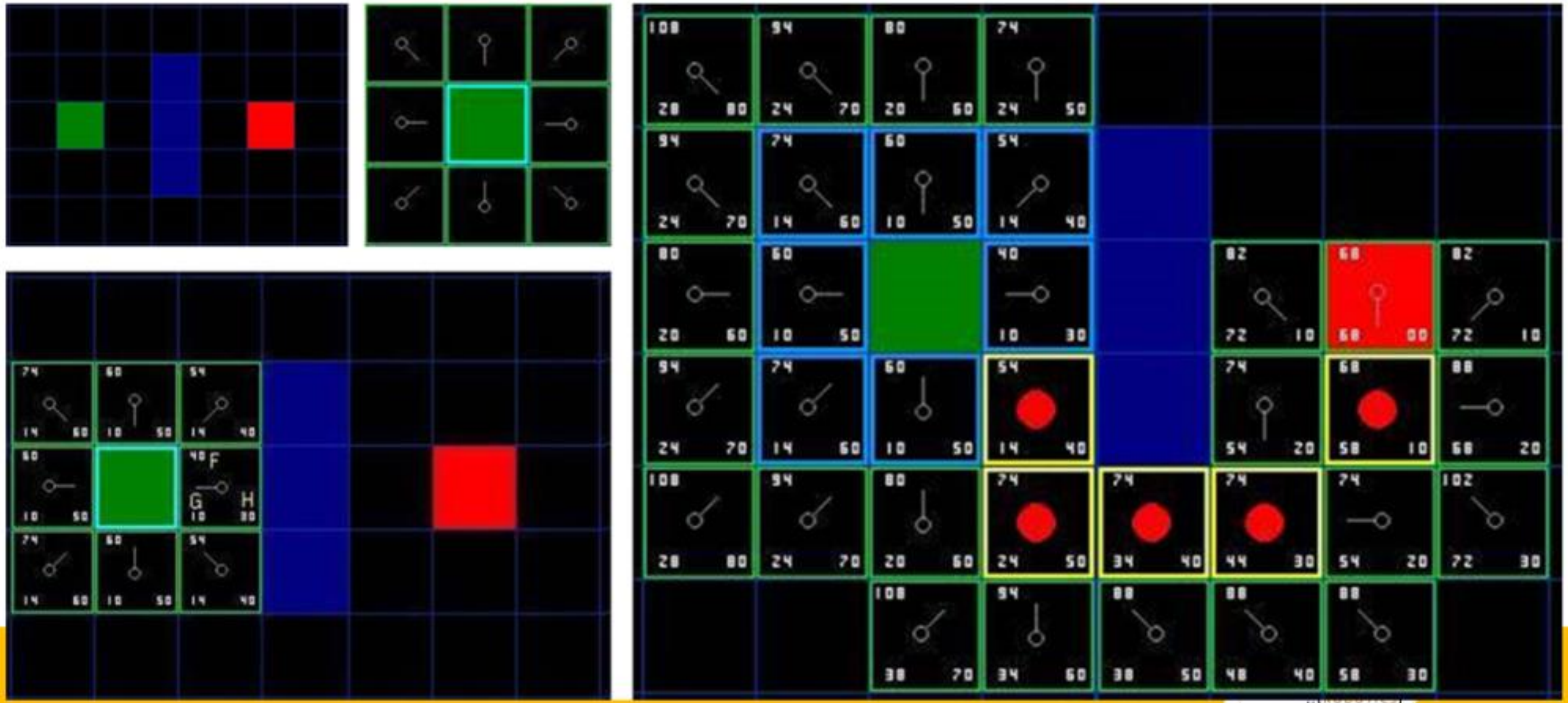
AUTONOMOUS NAVIGATION

Robot Motion Planning
Autonomous Motion Generation

[http://www.policyalmanac.org/games/aStarTutorial.htm]

A * algorithm

Path with shortest cost, $f(n) = \underbrace{g(n)}_{start \to n} + \underbrace{h(n)}_{n \to goal}$

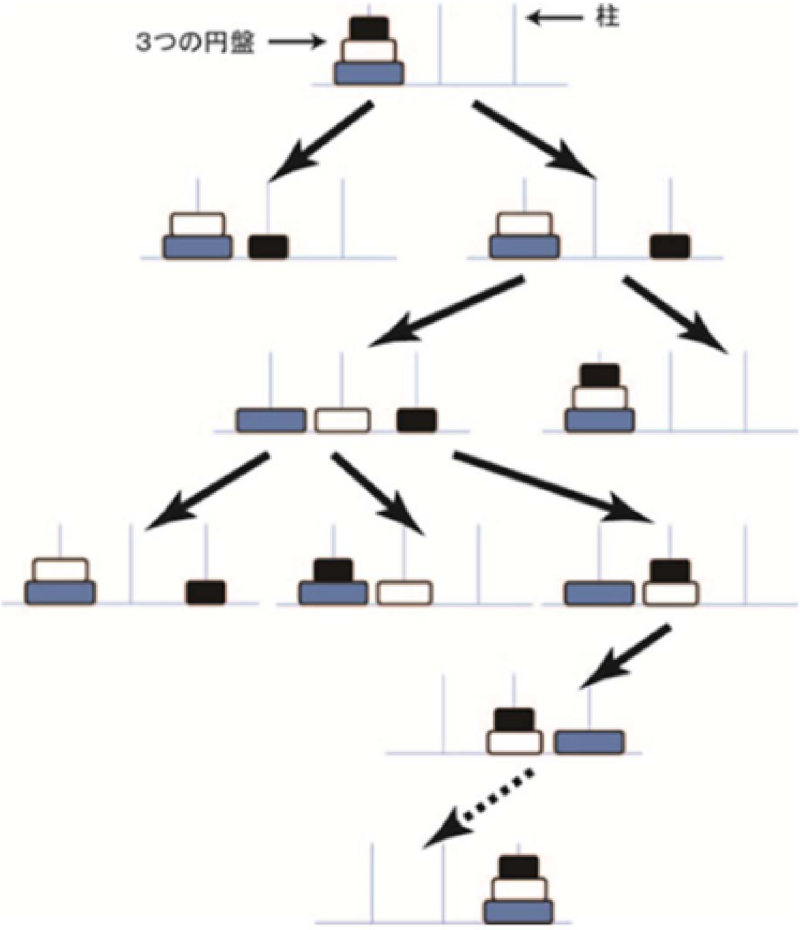


Robot Motion Planning
Autonomous Motion Generation

State Space & Search Problem

- Search Tree
- Each state, Node
- Connection between states, Link
- Start State
- Goal State

Tower of Hanoi



State space in Hanoi Tower

Autonomous Navigation

Adaptive Monte Carlo Localization

- https://en.wikipedia.org/wiki/Monte_Carlo_localization
- http://wiki.ros.org/amcl

B) Autonomous Navigation of a Known Map with TurtleBot

– http://wiki.ros.org/turtlebot_navigation/Tutorials/indigo/Autonomously%20navigate%20in%20a%20known%20map

Autonomous Navigation [Robot]

1. Bring up
   - $ roslaunch jupiterobot_bringup **jupiterobot_bringup.launch**
2. Launch AMCL with scanned map
   - [RGB-D] $ roslaunch jupiterobot_navigation **amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/test1.yaml
   - [Lidar] $ roslaunch jupiterobot_navigation **rplidar_amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/test1.yaml
3. Use RViz for navigation visualization
   - $ roslaunch turtlebot_rviz_launchers **view_navigation.launch**
4. Set robot current position in RViz
   - Click the "2D Pose Estimate" button
   - Click and drag on the map for the current robot location and orientation
5. Send a navigation goal with RViz
   - Click the "2D Nav Goal" button
   - Click and drag on the map for the goal location and orientation

Autonomous Navigation [Robot]

C) Source code implementation

– Node
  - /rc-home-edu-learn-ros/rchomeedu_navigation/scripts/**navigation.py**
  - Navigation target: **Location A**

1. Bring up
   - $ roslaunch jupiterobot_bringup **jupiterobot_bringup.launch**
2. Launch AMCL with scanned map
   - [RGB-D] $ roslaunch jupiterobot_navigation **amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/test1.yaml
   - [Lidar] $ roslaunch jupiterobot_navigation **rplidar_amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/test1.yaml
3. Use RViz for navigation visualization
   - $ roslaunch turtlebot_rviz_launchers **view_navigation.launch**
4. Determine and update the coordinate of the goal location
   - Click the "Publish Point" button and move the cursor to a desired goal location on map (do not click on the map)
   - Record the first two numbers (x, y) on the bottom left corner of RViz window
   - Update the numbers (x, y) as "Location A" in /rc-home-edu-learn-ros/rchomeedu_navigation/scripts/**navigation.py**
5. Launch the navigation code
   - $ roslaunch rchomeedu_navigation **navigation.launch**
   - Set robot current position with "2D Pose Estimate"

Jupiter Robot in Gazebo [Simulation]

- Launch robot in virtual world
  - $ roslaunch jupiterobot_gazebo **jupiterobot_world.launch** world_file:=/home/mustar/catkin_ws/worlds/Jupiter_Robot_Office.world

- Simulation model parameters
  - stacks: **h** (hexagon plates), **c** (circular plates) | default – **h**
  - lasers: **n** (none), **r** (rplidar), **h** (hokuyo) | default – **r**
  - arms: **n** (none), **5** (5 DOF arm), **7** (7 DOF arm) | default – **5**
  - heads: **n** (none), **1** (1 DOF head), **2** (2 DOF head) | default – **1**

Navigation: Autonomous Navigation [Simulation]

1. Launch AMCL with scanned map
   – $ roslaunch jupiterobot_gazebo **amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/JupiterOfficeSim.yaml
2. Use RViz for navigation visualization
   – $ roslaunch turtlebot_rviz_launchers **view_navigation.launch**
3. Set robot current position in RViz
   – Click the "2D Pose Estimate" button
   – Click and drag on the map for the current robot location and orientation
4. Send a navigation goal with RViz
   – Click the "2D Nav Goal" button
   – Click and drag on the map for the goal location and orientation

Navigation: Autonomous Navigation [Simulation]

C) Source code implementation

– Node

- /rc-home-edu-learn-ros/rchomeedu_navigation/scripts/**navigation.py**
- Navigation target: **Location A**

1. Launch AMCL with scanned map
   - $ roslaunch jupiterobot_gazebo **amcl_demo.launch** map_file:=/home/mustar/catkin_ws/maps/JupiterOfficeSim.yaml
2. Use RViz for navigation visualization
   - $ roslaunch turtlebot_rviz_launchers **view_navigation.launch**
3. Determine and update the coordinate of the goal location
   - Click the "Publish Point" button and move the cursor to a desired goal location on map (do not click on the map)
   - Record the first two numbers (x, y) on the bottom left corner of RViz window
   - Update the numbers (x, y) as "Location A" in /rc-home-edu-learn-ros/rchomeedu_navigation/scripts/**navigation.py**
4. Launch the navigation code
   - $ roslaunch rchomeedu_navigation **navigation.launch**
   - Set robot current position with "2D Pose Estimate"

Autonomous Vehicle

[https://www.youtube.com/watch?v=tiwVMrTLUWg]

Exercise: Waypoints Navigation

1. SLAM map building
2. Determine waypoints at key locations
   - Location examples: Kitchen, living room, bedroom, dining room.
3. Autonomous waypoints navigation
   - Send the goal location command by speech