PATH PLANNING

Agenda

Topics to be touched upon briefly...

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Using Deep Reinforcement learning

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Machine learning
- Velocity planning

"Process of constructing a shortest possible path connecting destination and avoiding the detected obstacles

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Using Deep Reinforcement learning

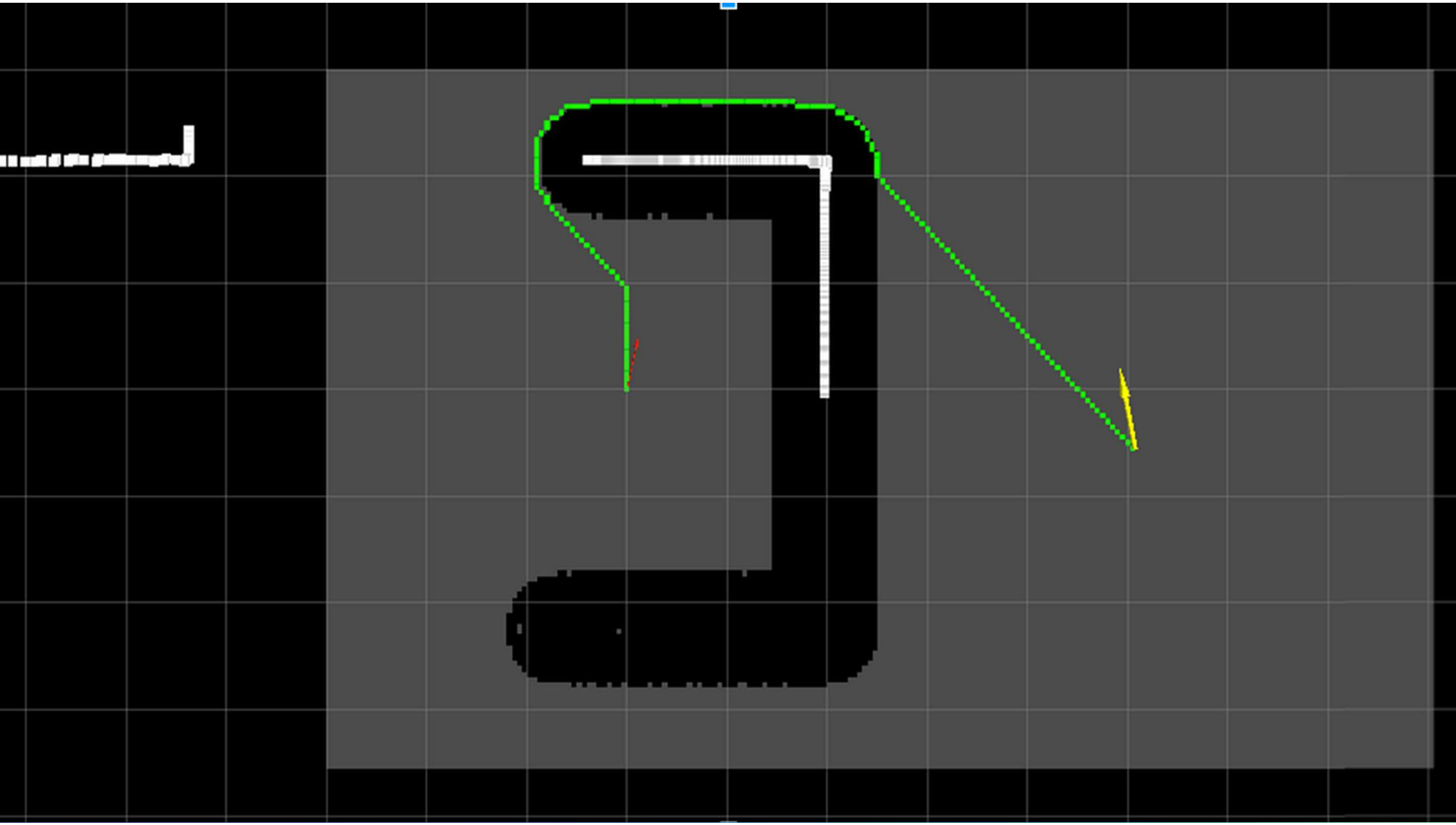
Setting Local and Global goal

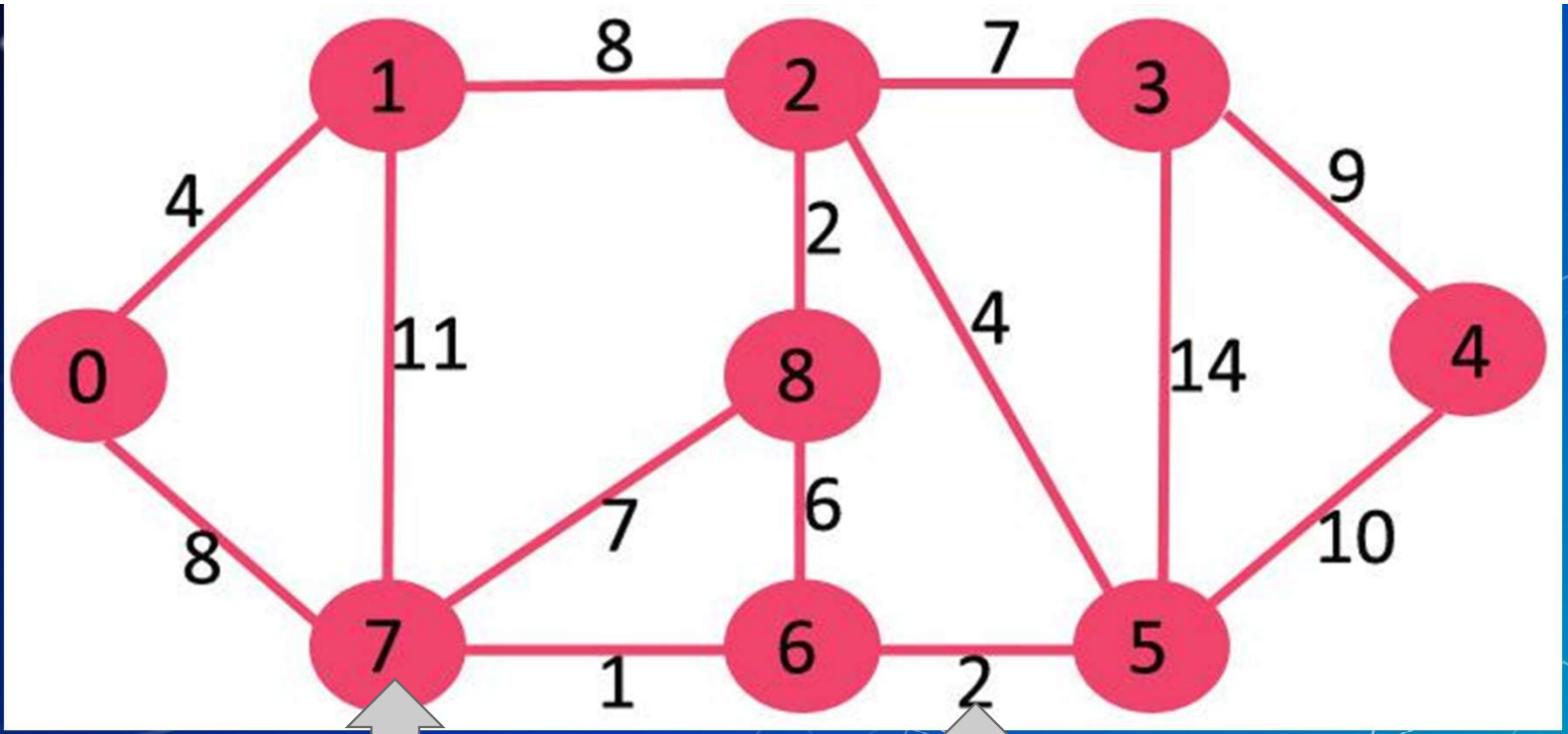
Finding shortest path connecting destination

Velocity Planning

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Using Deep Reinforcement learning

- Obstacles → occupancy grid
- Cells of occupancy grid as vertices of a graph
- Cost for changing cells (vertices)
- Cost ~ Distance to be travelled
- Algorithms required to find shortest path





Node / Vertex

Cost / Weights

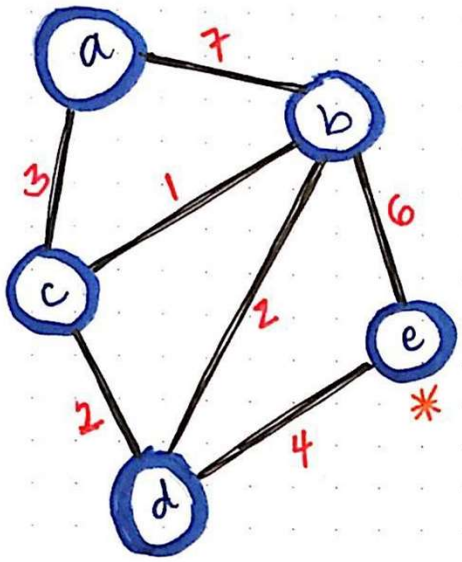
Real life environment visualization as a graph

- Objects, lanes and other vehicles in occupancy grid
- Each pixel has value ~ it's "occupancy".

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Using Deep Reinforcement learning

Dijkstra's algorithm

Uses 'greedy method' to find least cost path to any vertex on the Occupancy Grid



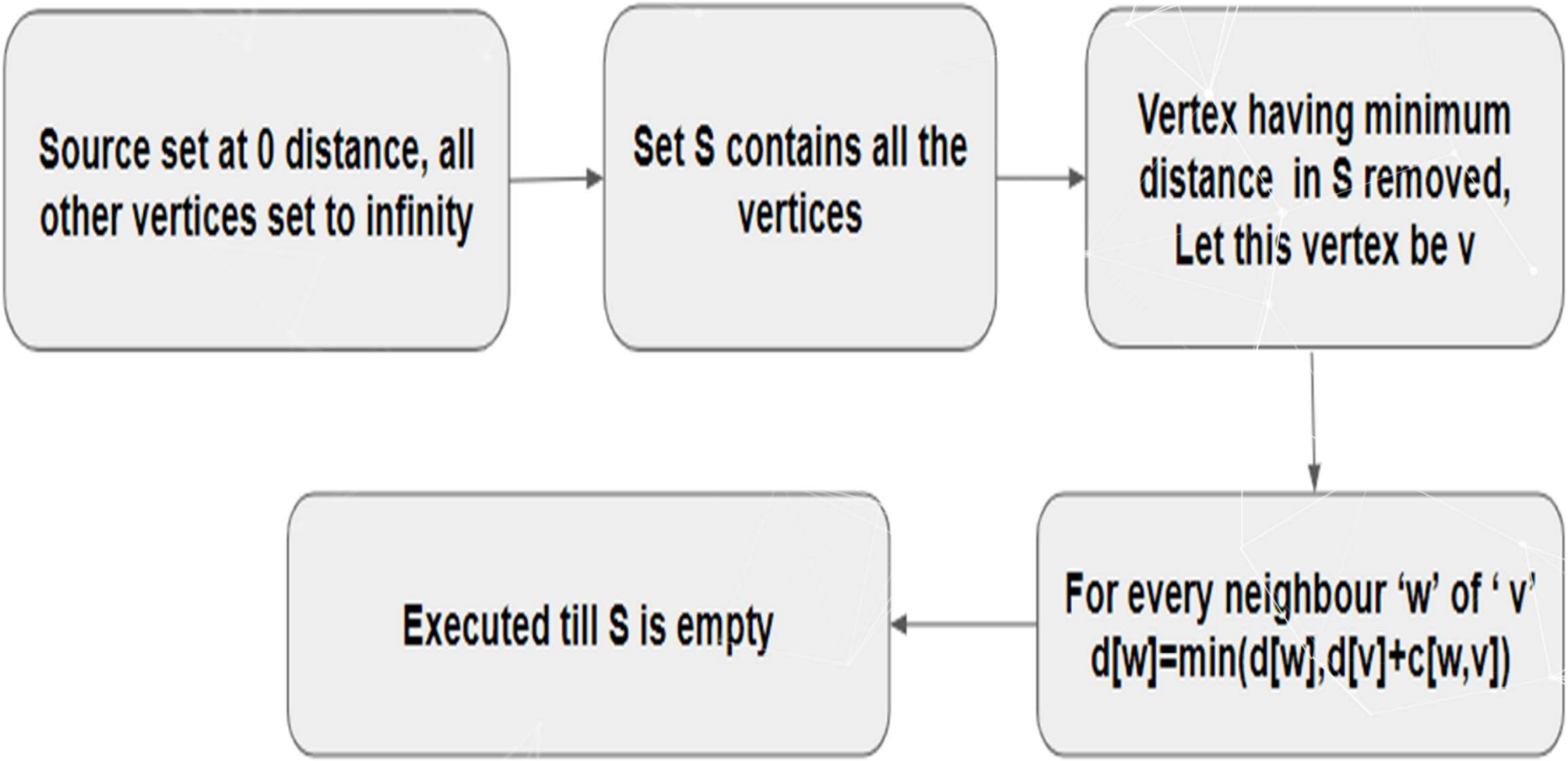
VERTEX	SHORTEST DIST. FROM (a)	PREVIOUS VERTEX
a	0	
b	~~0~~ ~~7~~ 4	~~a~~ c
c	~~0~~ 3	a
d	~~0~~ 5	b
e	~~0~~ ~~10~~ 9	~~b~~ d

Visited = [a, c, b, d]

Unvisited = [(e)]

current vertex

*The only node left to visit is (e). However, all of its neighbors have already been visited, so there's nothing for us to examine or update here!


Source set at 0 distance, all other vertices set to infinity
Set S contains all the vertices
Vertex having minimum distance in S removed, Let this vertex be v
For every neighbour 'w' of ' v' d[w]=min(d[w],d[v]+c[w,v])
Executed till S is empty

ALGORITHM

Why Dijkstra's Algorithm?

Advantages

- Simple
- Computationally light
- Good for blind search

Disadvantages

- Very large processing time for dynamic planning
- Doesn't consider smoothness of path
- Ambiguity with negative weights
  - (Alternative is Bellman-Ford)

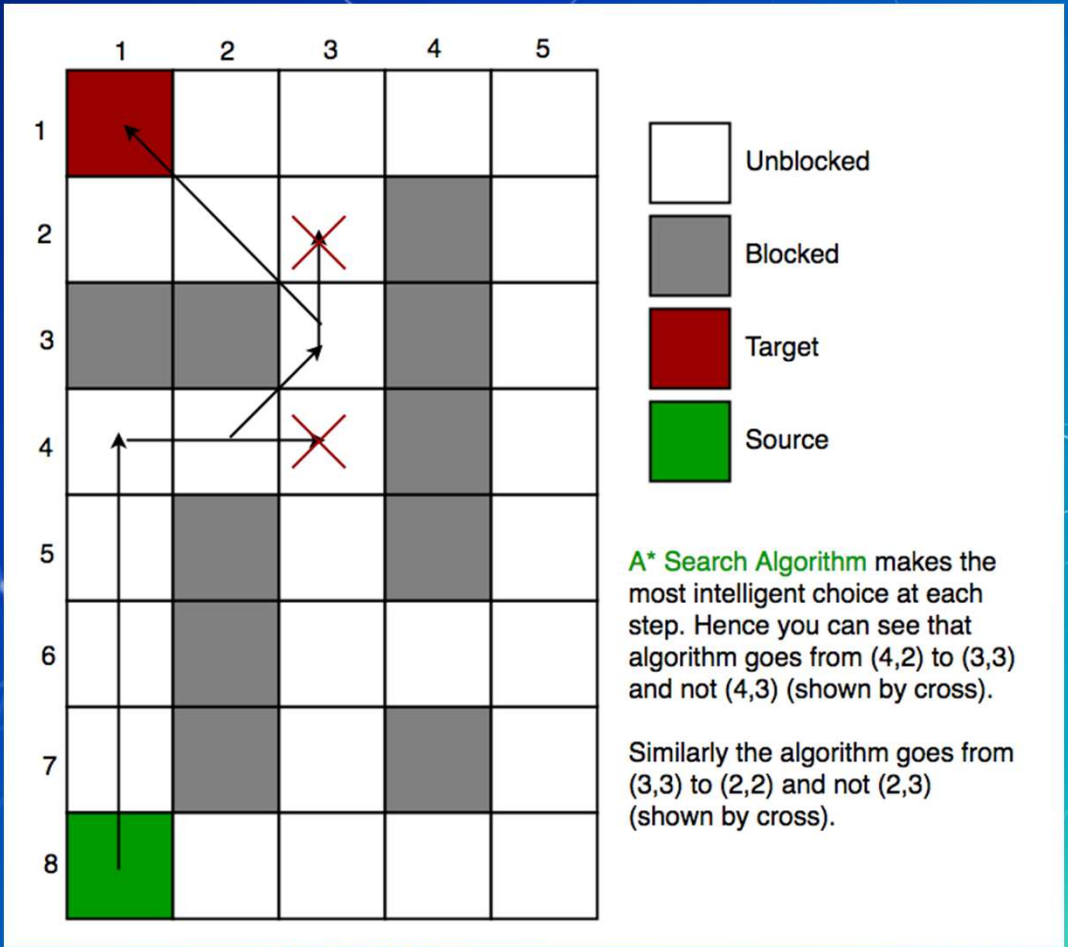
A* algorithm

Minimizes ‘f-value’.

f-value = g-value + h-value

g-value = Movement cost to reach current node

h-value = Estimated movement cost to reach the destination node




GOAL
I got you!
START

Ways to calculate H-value

Exact Calculation

- High processing time
- All distances required apriori

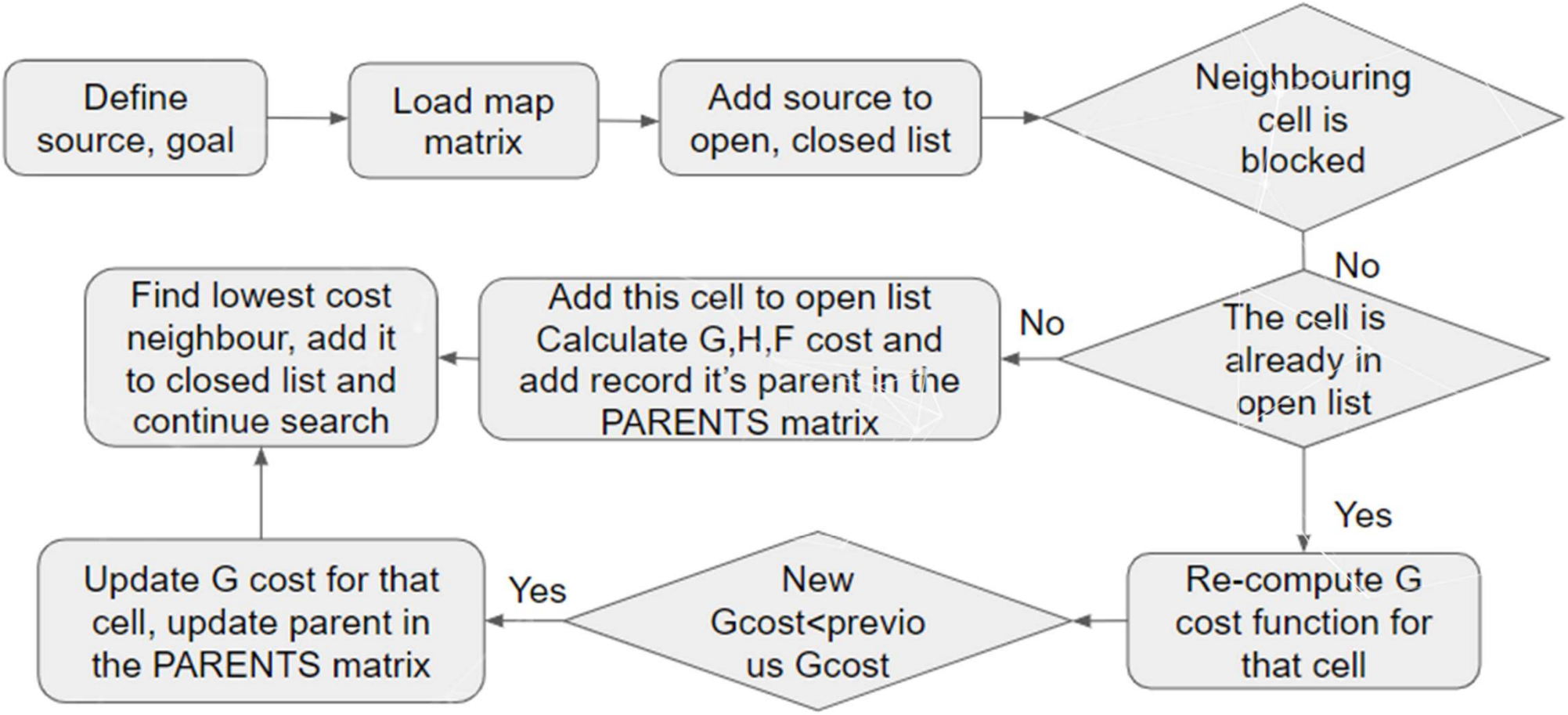
Approximate calculation

- Calculate Euclidean distance
- Time and resource saving

Then we define 2 sets namely open set, closed set and a Parents matrix.

Parents is a 2D matrix, which should map all the nodes in the open list to it's parent node i.e. the node from which we came to the current node.

A* algorithm rough flowchart



Why A* Algorithm?

Advantages

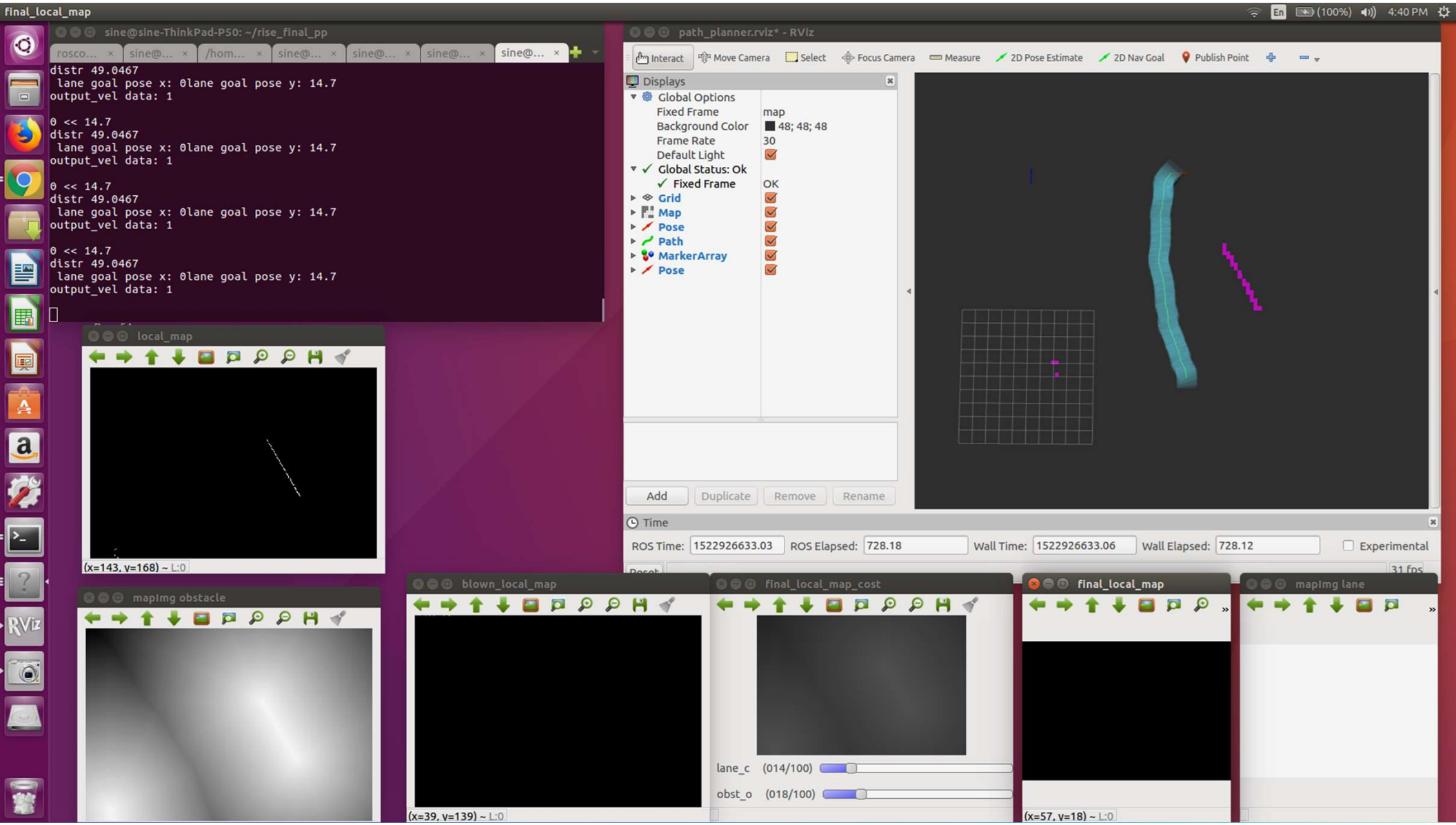
- Intelligent Choice
- Less time complexity

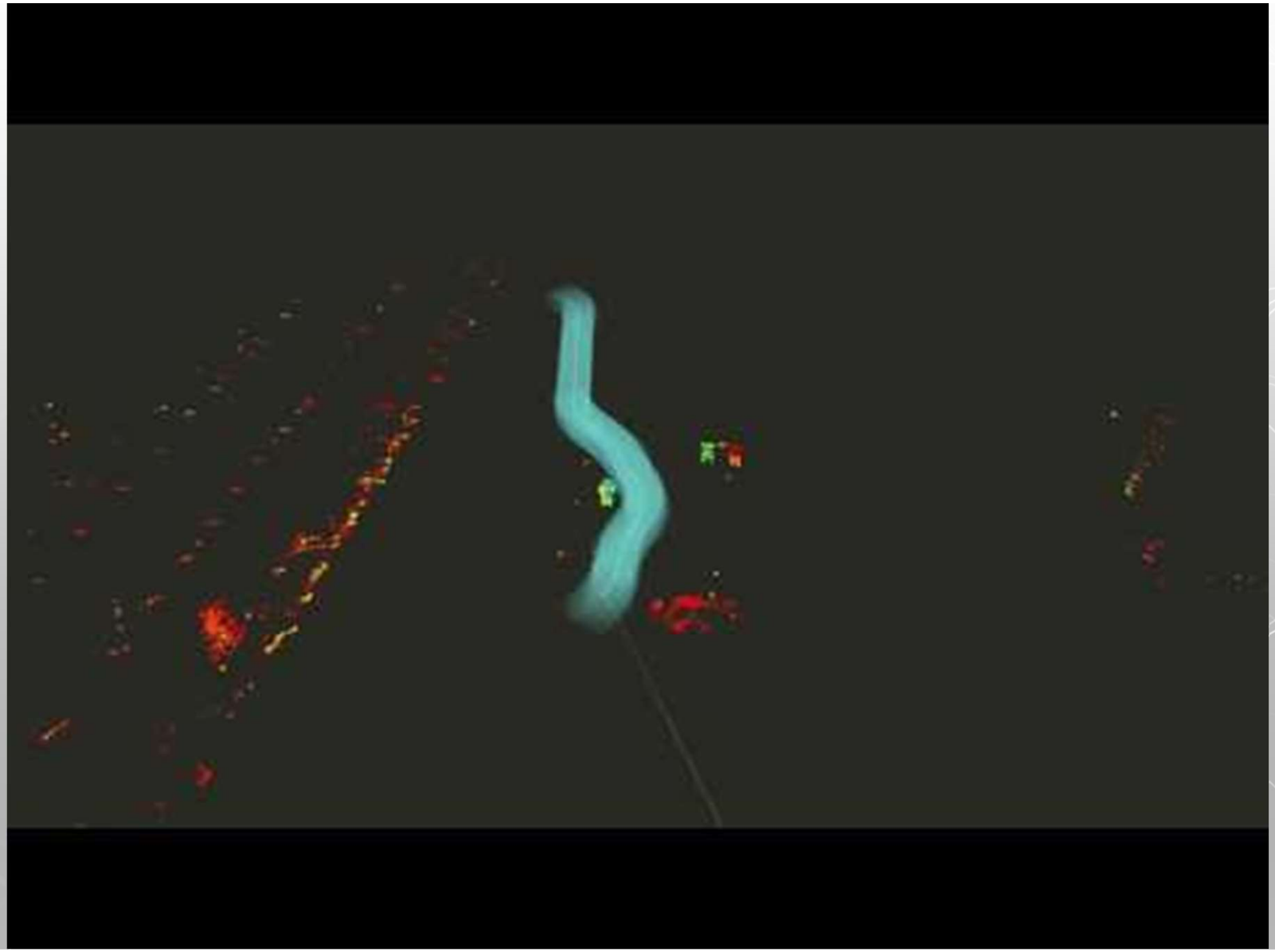
Disadvantages

- Doesn’t consider smoothness of path

Hybrid A* algorithm

Smoothness filter over A* algorithm

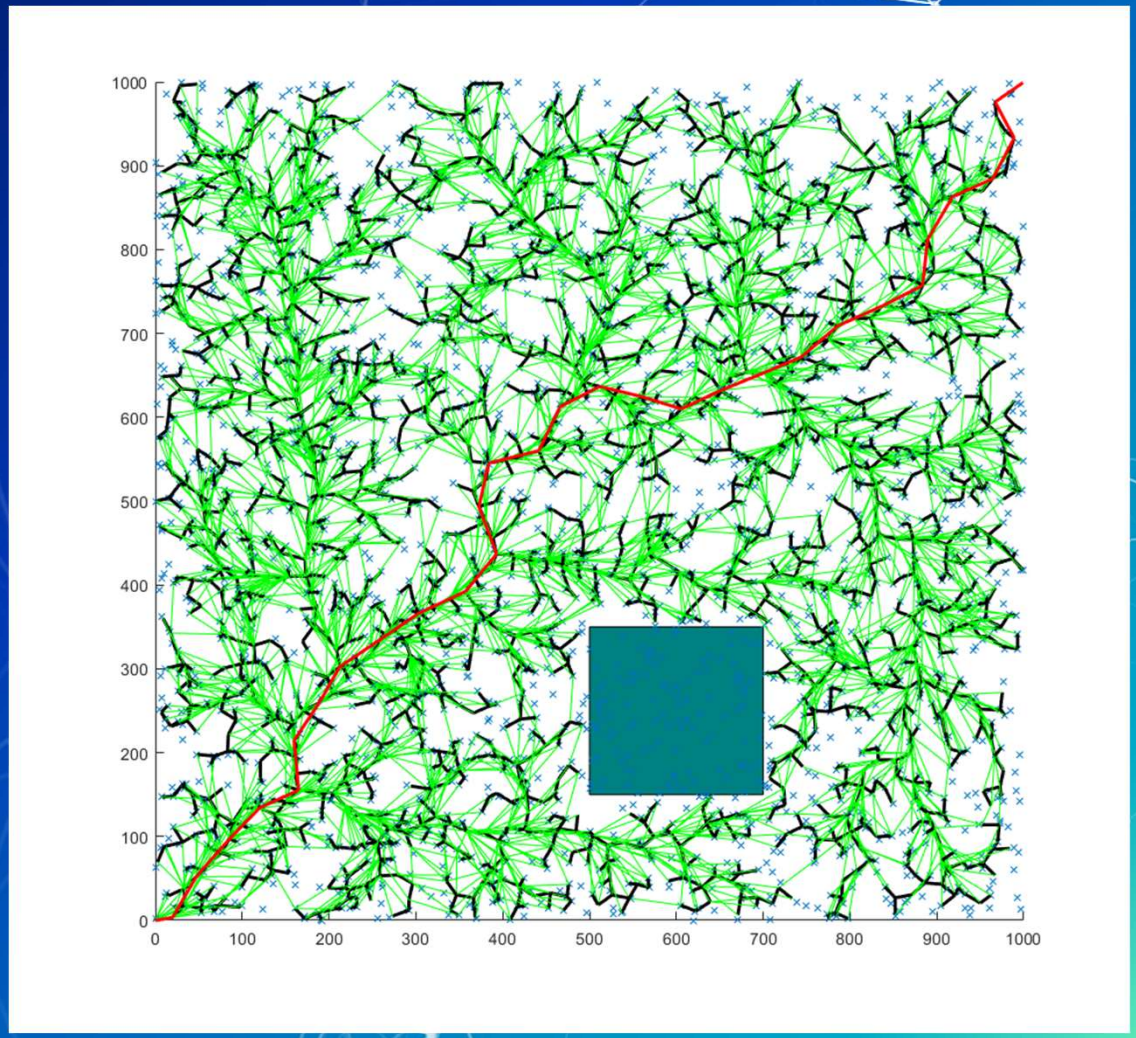

final_local_map
sine@sine-ThinkPad-P50: ~/rise_final_pp
distr 49.0467
lane goal pose x: 0lane goal pose y: 14.7
output_vel data: 1
0 << 14.7
distr 49.0467
lane goal pose x: 0lane goal pose y: 14.7
output_vel data: 1
0 << 14.7
distr 49.0467
lane goal pose x: 0lane goal pose y: 14.7
output_vel data: 1
0 << 14.7
distr 49.0467
lane goal pose x: 0lane goal pose y: 14.7
output_vel data: 1
path_planner.rviz* - RViz
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
map
Background Color
48; 48; 48
Frame Rate
30
Default Light
Global Status: Ok
Fixed Frame
OK
Grid
Map
Pose
Path
MarkerArray
Pose
Add
Duplicate
Remove
Rename
Time
ROS Time: 1522926633.03
ROS Elapsed: 728.18
Wall Time: 1522926633.06
Wall Elapsed: 728.12
Experimental
local_map
(x=143, y=168) ~ L:0
mapImg obstacle
blown_local_map
(x=39, y=139) ~ L:0
final_local_map_cost
lane_c (014/100)
obst_o (018/100)
final_local_map
(x=57, y=18) ~ L:0
mapImg lane



RRT* algorithm

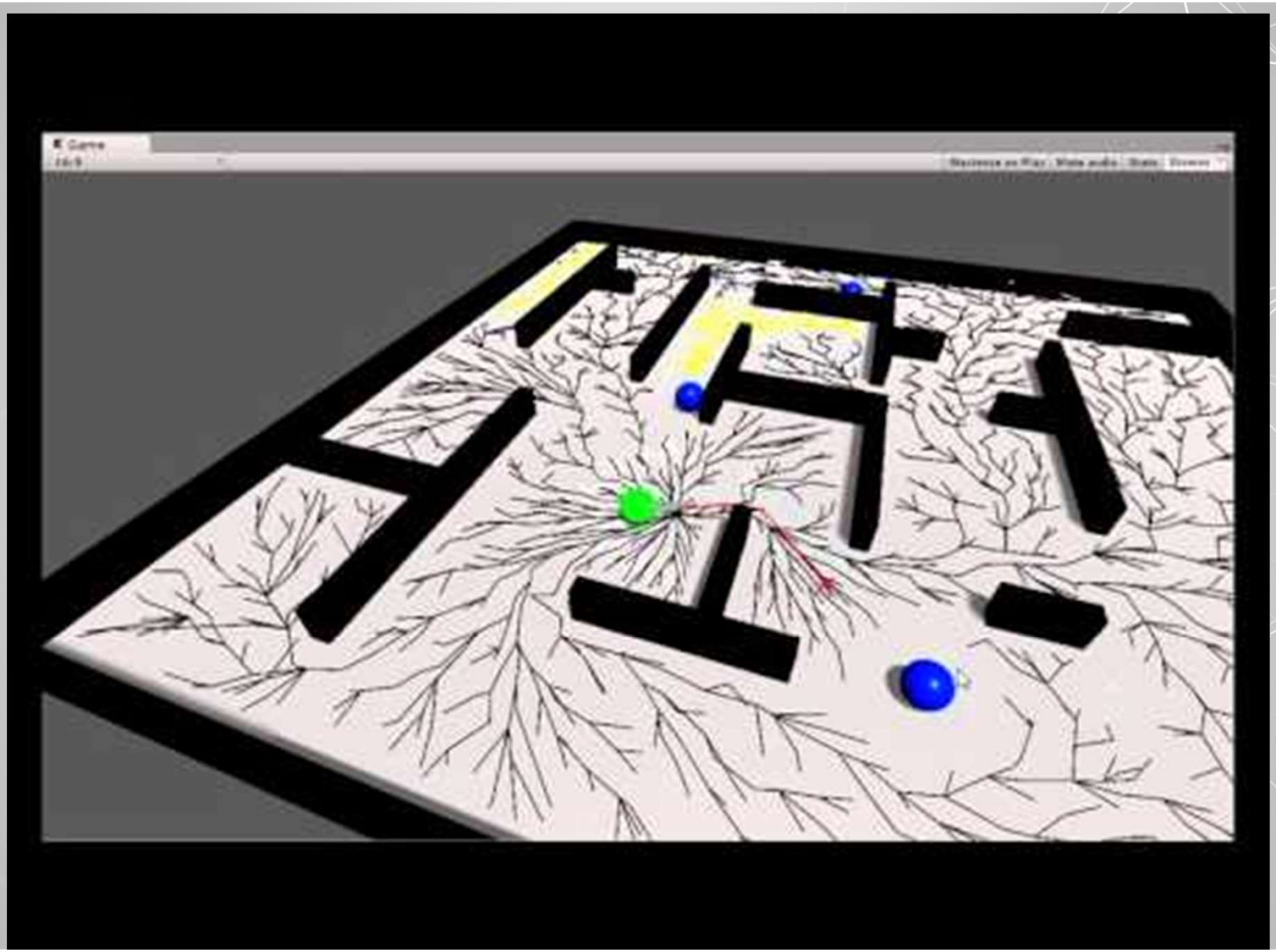
Rapidly exploring random tree

- Adaptation



Some information

- Avoids obstacles, minimizes cost (length of path).
- Records the distance each vertex has traveled relative to its parent vertex.
- Rewiring of the tree.



RRT* informed algorithm

Rapidly exploring random tree

- More Adaptation!

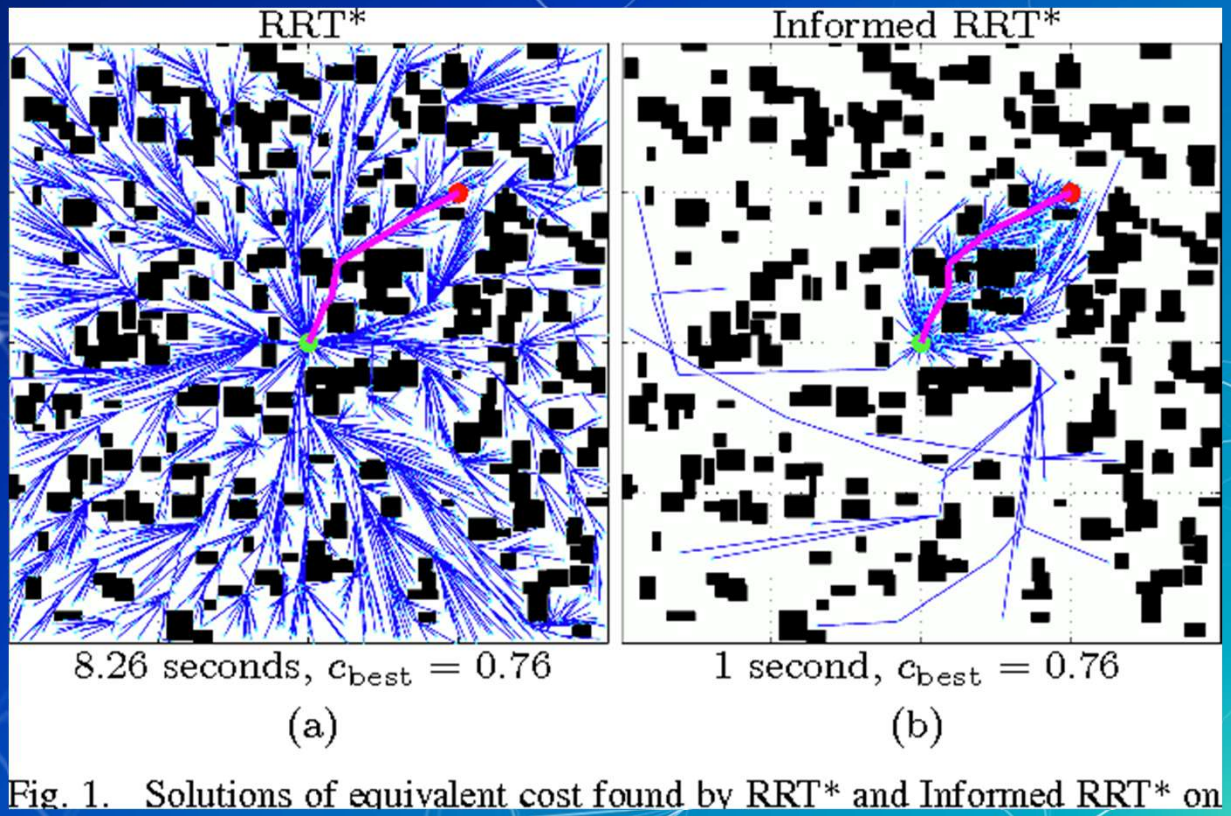
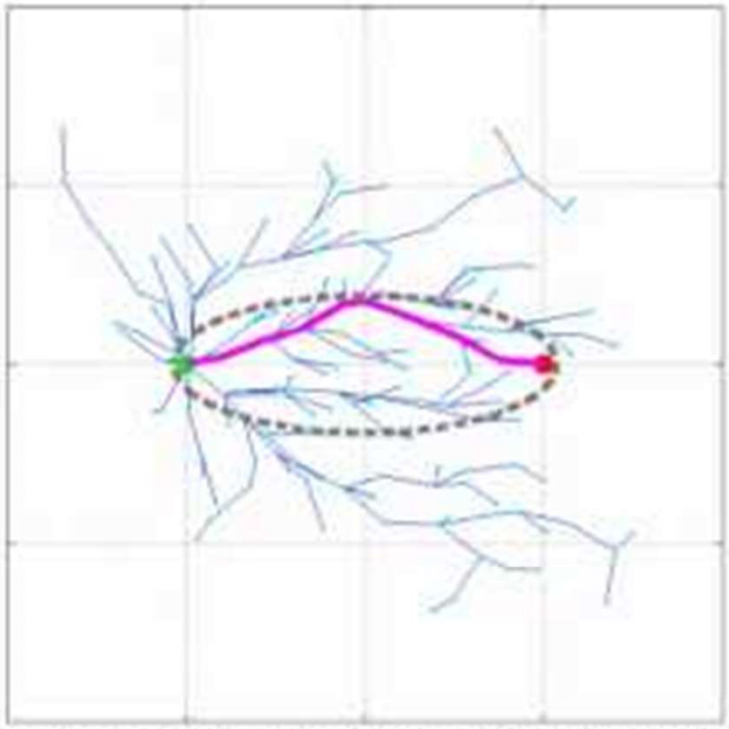


Fig. 1. Solutions of equivalent cost found by RRT* and Informed RRT* on

Some information

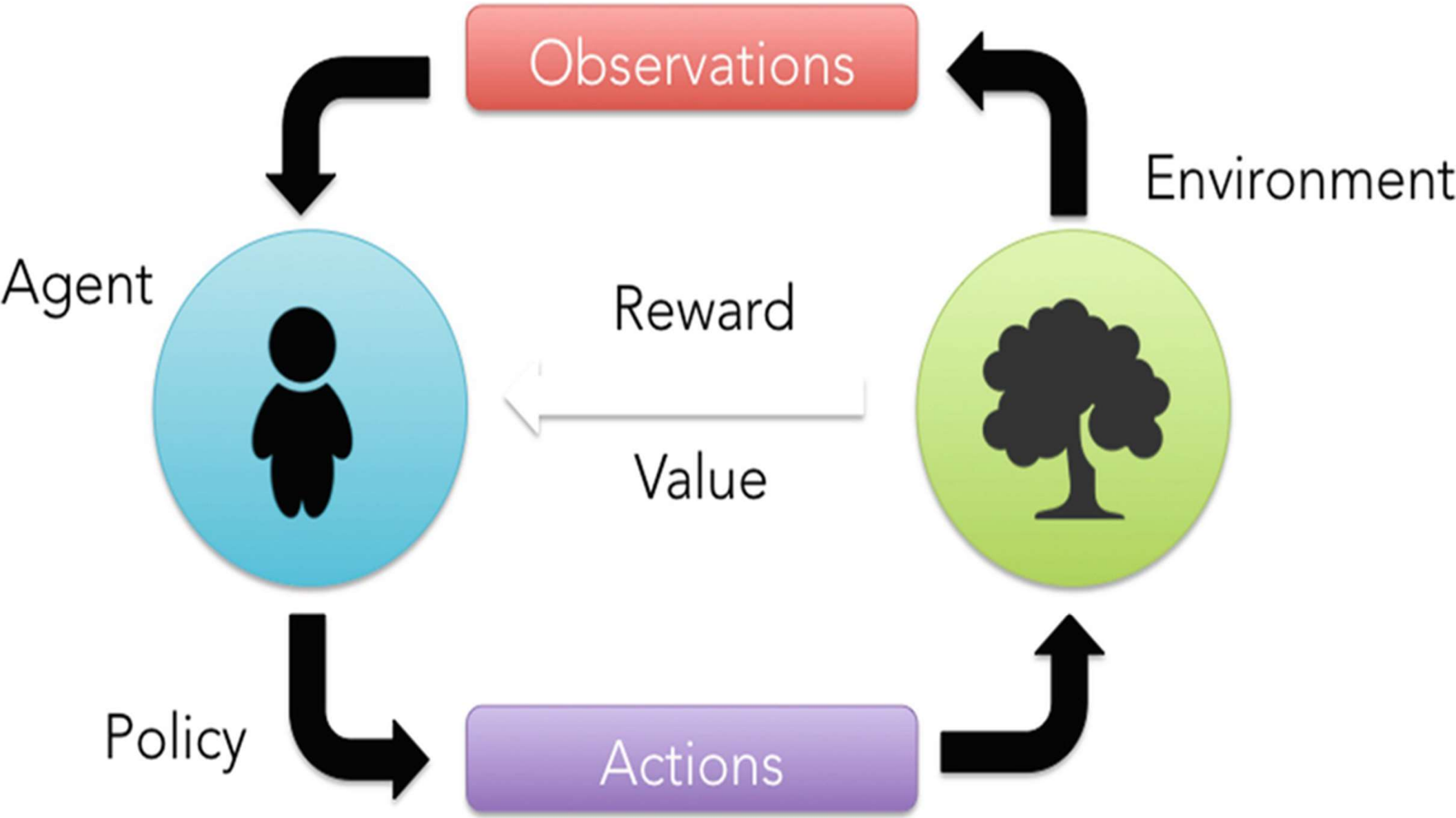
- Avoids obstacles, minimizes cost (length of path).
- Records the distance each vertex has traveled relative to its parent vertex.
- Rewiring of the tree.
- Sampling of points after path is planned for cost reduction

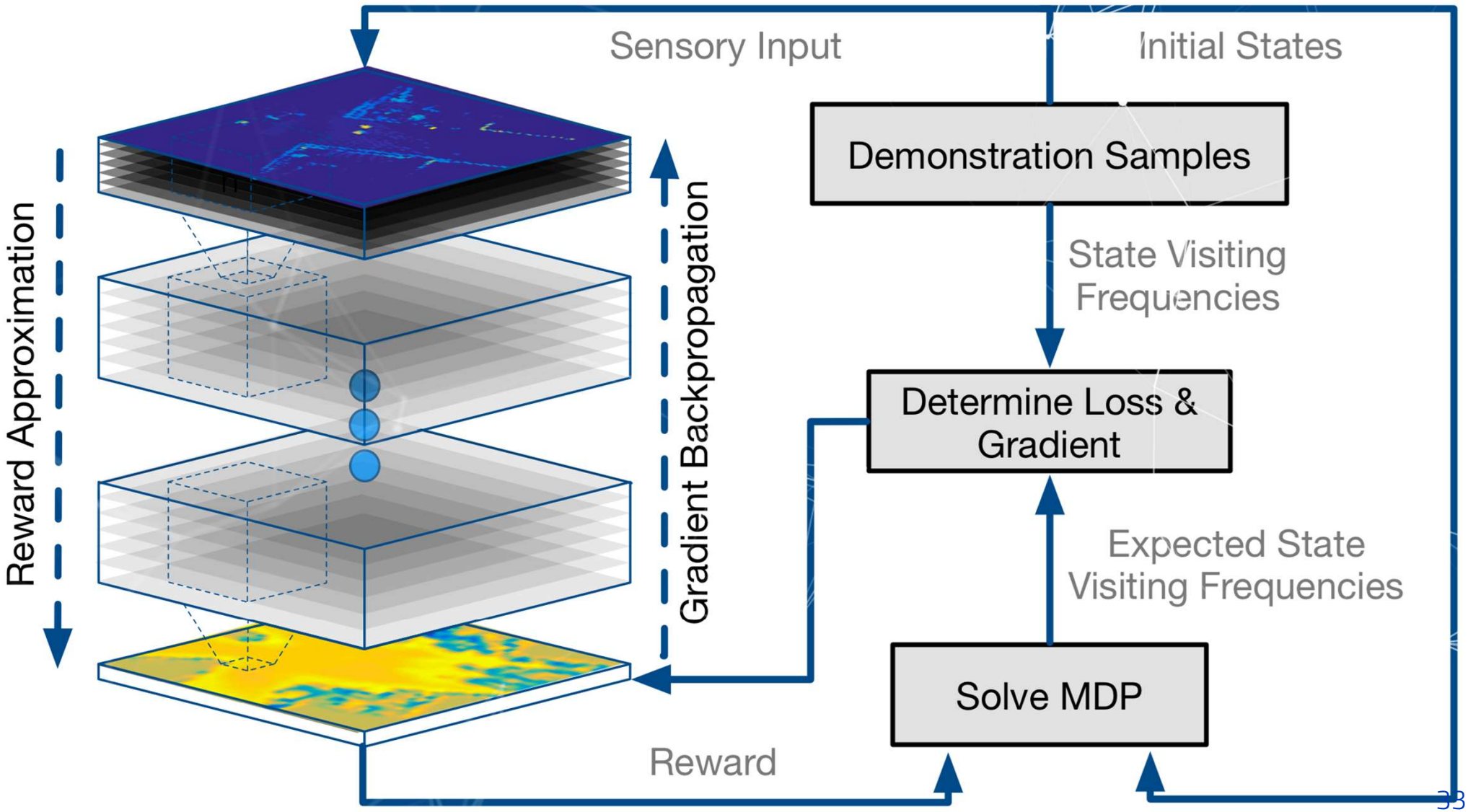
000177



By *directly* sampling the ellipse, we focus the search
to only the states that have the possibility of improving the solution.

- What is path planning?
- Dynamic Path planning
- Graph method
- Algorithms:
  - Dijkstra's algorithm
  - A* planning
  - Hybrid A* planning
  - RRT* planning
  - RRT* informed planning
- Using Deep Reinforcement learning


Observations
Environment
Agent
Reward
Value
Policy
Actions


Sensory Input
Initial States
Demonstration Samples
State Visiting Frequencies
Determine Loss & Gradient
Expected State Visiting Frequencies
Solve MDP
Reward
Reward Approximation
Gradient Backpropagation

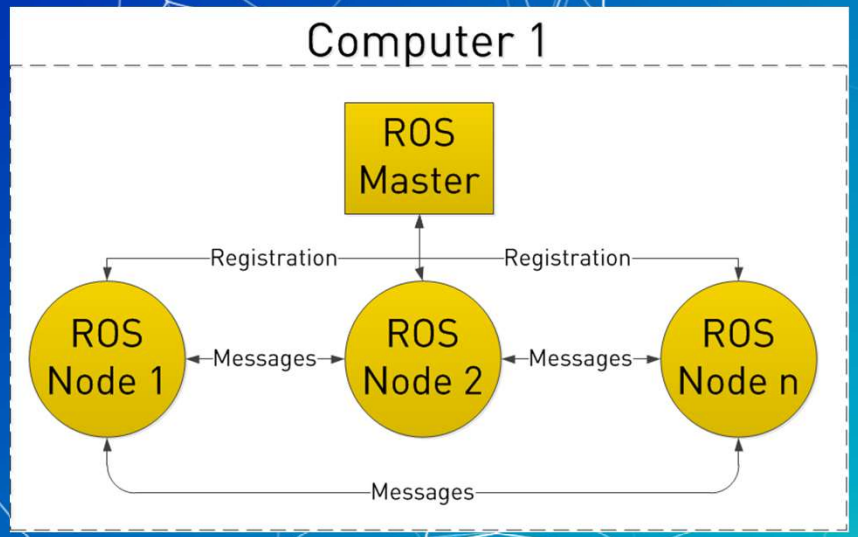

Neural network activations
DNN Control

ROS

ROBOT OPERATING SYSTEM

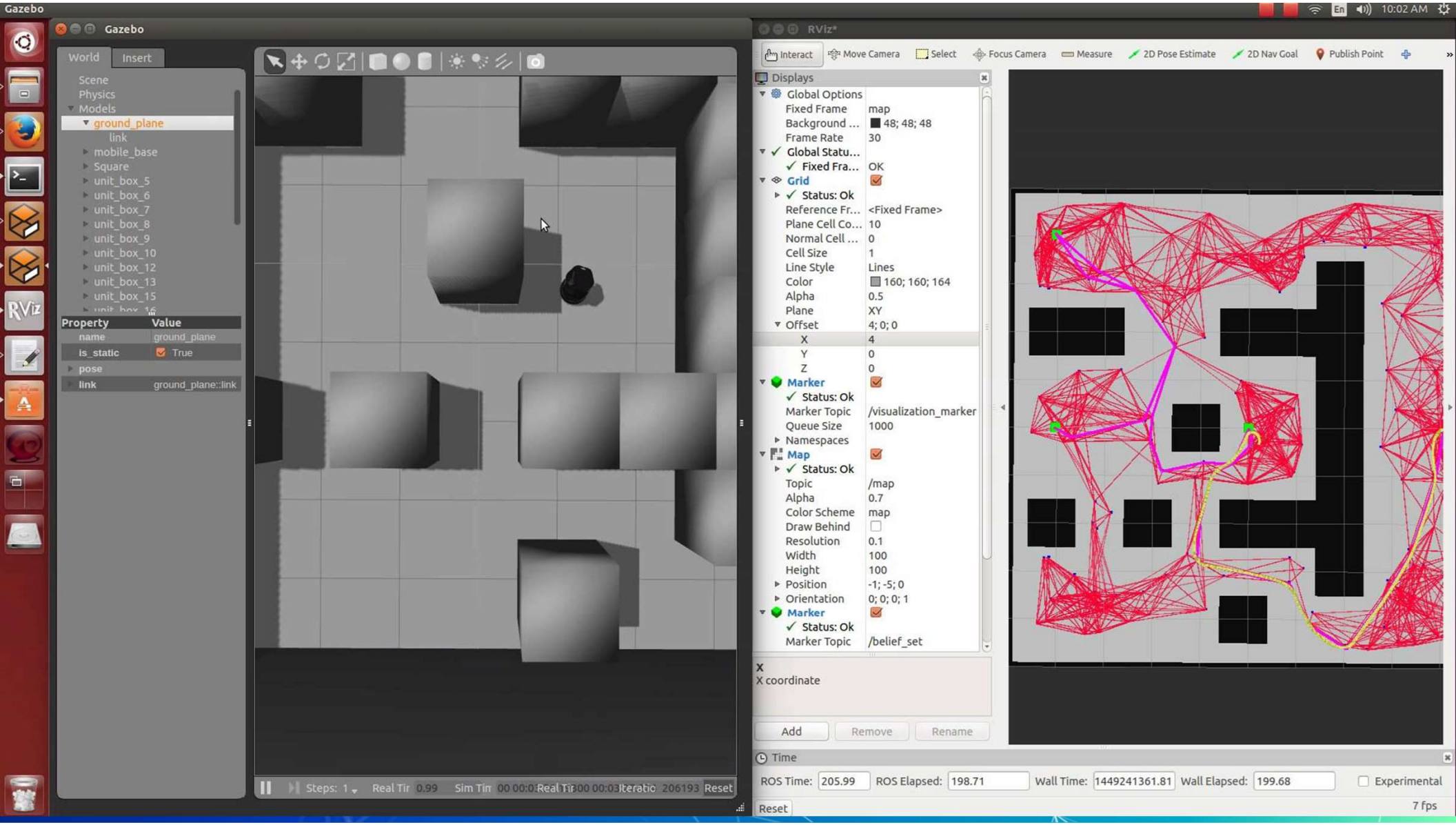
What is ROS? Why to use it?

- Set of software libraries and tools to helps us build robot applications.
- Interface for communication
- A lot of reusability and possibilities of coworking



What is ROS? Why to use it?

- Great tools for simulation and visualisation such as **Rviz** and **Gazebo**
- **Lots** of debugging tools
- Many packages already available on ROS platform
- Many useful pre-built libraries and routines
- No language boundaries


Gazebo
World
Insert
Scene
Physics
Models
ground_plane
link
mobile_base
Square
unit_box_5
unit_box_6
unit_box_7
unit_box_8
unit_box_9
unit_box_10
unit_box_12
unit_box_13
unit_box_15
Property
Value
name
ground_plane
is_static
True
pose
link
ground_plane::link
Steps: 1
Reset
RViz*
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
map
Background ...
48; 48; 48
Frame Rate
30
Global Statu...
Fixed Fra...
OK
Grid
Status: Ok
Reference Fr...
<Fixed Frame>
Plane Cell Co...
10
Normal Cell ...
0
Cell Size
1
Line Style
Lines
Color
160; 160; 164
Alpha
0.5
Plane
XY
Offset
4; 0; 0
X
4
Y
0
Z
0
Marker
Status: Ok
Marker Topic
/visualization_marker
Queue Size
1000
Namespaces
Map
Status: Ok
Topic
/map
Alpha
0.7
Color Scheme
map
Draw Behind
Resolution
0.1
Width
100
Height
100
Position
-1; -5; 0
Orientation
0; 0; 0; 1
Marker
Status: Ok
Marker Topic
/belief_set
X
X coordinate
Add
Remove
Rename
Time
ROS Time:
205.99
ROS Elapsed:
198.71
Wall Time:
1449241361.81
Wall Elapsed:
199.68
Experimental
Reset
7 fps

Common terminologies

Node Processes that perform computation.

Master Provides name lookup and access to rest of the computational graph

Message Data structure which is output of a node

Common terminologies

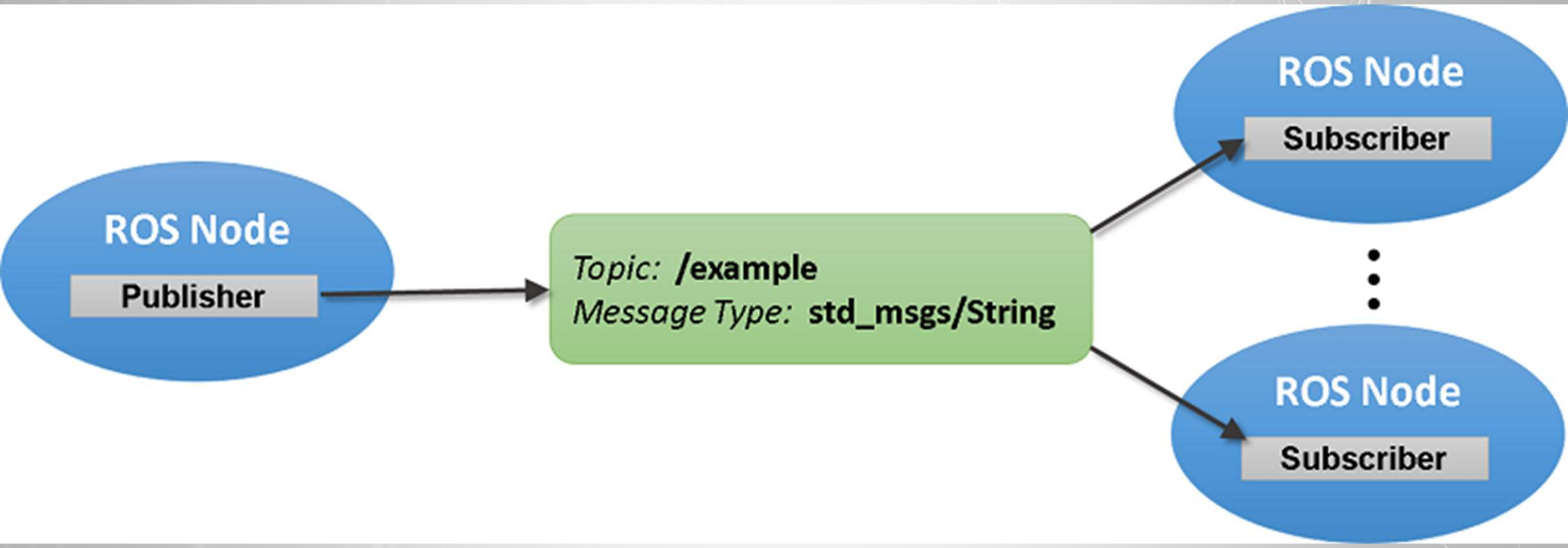
Topics Name used to identify the message handled out of the nodes

Bags Formats for saving and playing back ROS messages and data

Publisher and Subscriber

Publisher Used by node to publish data to ROS network on a particular topic

Subscriber Used by node to subscribe data from ROS network from a particular topic


ROS Node
Publisher
Topic: /example
Message Type: std_msgs/String
ROS Node
Subscriber
ROS Node
Subscriber

THANKS!

Any questions?

CREDITS

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash
- All images and videos are taken from different resources.
- All credits to the owners.