上海科技大学
ShanghaiTech University

# CS283: Robotics Spring 2023: Maps

Sören Schwertfeger / 师泽仁

ShanghaiTech University

# ARM PLANNING

# Kinematic Problems for Manipulation

- Reliably position the tip - go from one position to another position
- Don't hit anything, avoid obstacles
- Make smooth motions
  - at reasonable speeds and
  - at reasonable accelerations
- Adjust to changing conditions -
  - i.e. when something is picked up *respond to the change in weight*

# Planning Problem

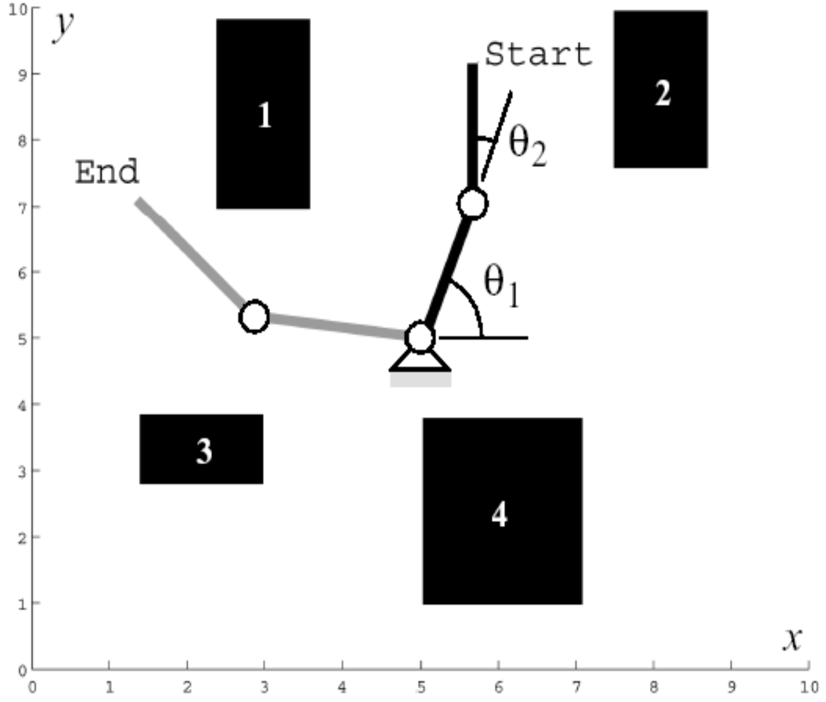
- (Arm) Pose: Set of join values
- (Arm) Trajectory:
  - Given a start pose and an end pose
  - A list of intermediate poses
  - That should be reached one after the other
  - With associated (desired) velocities and accelerations (maxima)
    - Without time (without velocity and acceleration): path! So:
    - Path: poses;   Trajectory: poses with speeds (and maybe accelerations)
- Constrains:
  - Don’t collide with yourself
  - Don’t collide with anything else (except: fingers with the object to manipulate!!!)
  - Additional possible constrains:
    - Maximum joint velocities or accelerations
    - Keep global orientation of a joint (often end-effector) within certain boundaries

# Planning Problem cont.

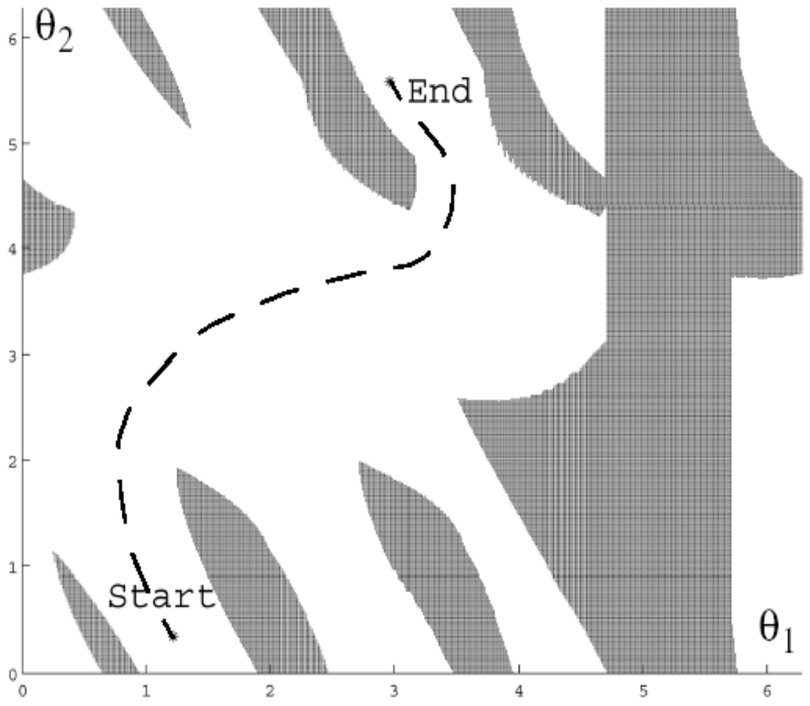
- Often the goal specified in Cartesian space (not joint space)
- => use IK to get joint space
- => often multiple (even infinitely many) solutions
  - Which one select for planning?
  - Plan for several solutions and select best!?

# Work Space (Map) → Configuration Space

- State or configuration $q$ can be described with $k$ values $q_i$




a)




b)

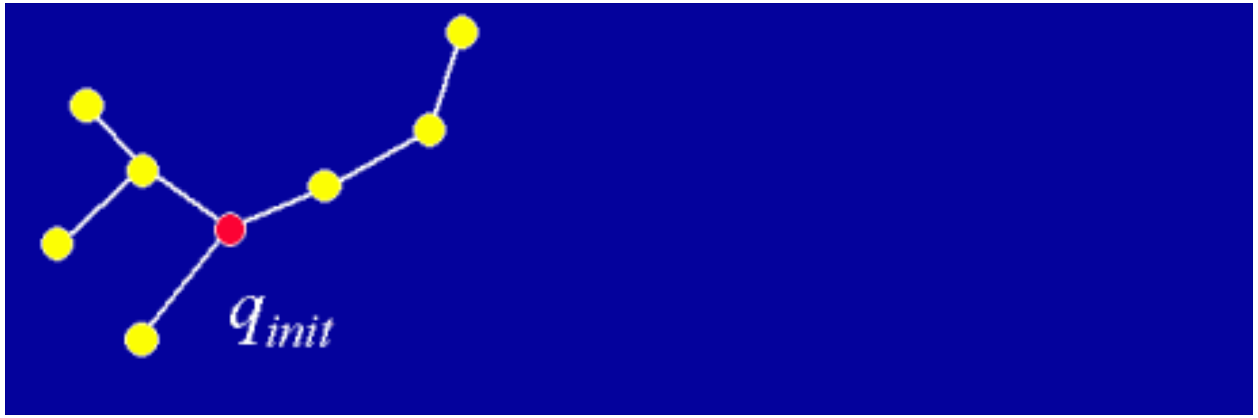
**Work Space**
Dimension depends on map
Dimension – typically 2D or 3D

**Configuration Space:**
the dimension of this
space is equal to the Degrees of Freedom (DoF)
of the robot

# RRT

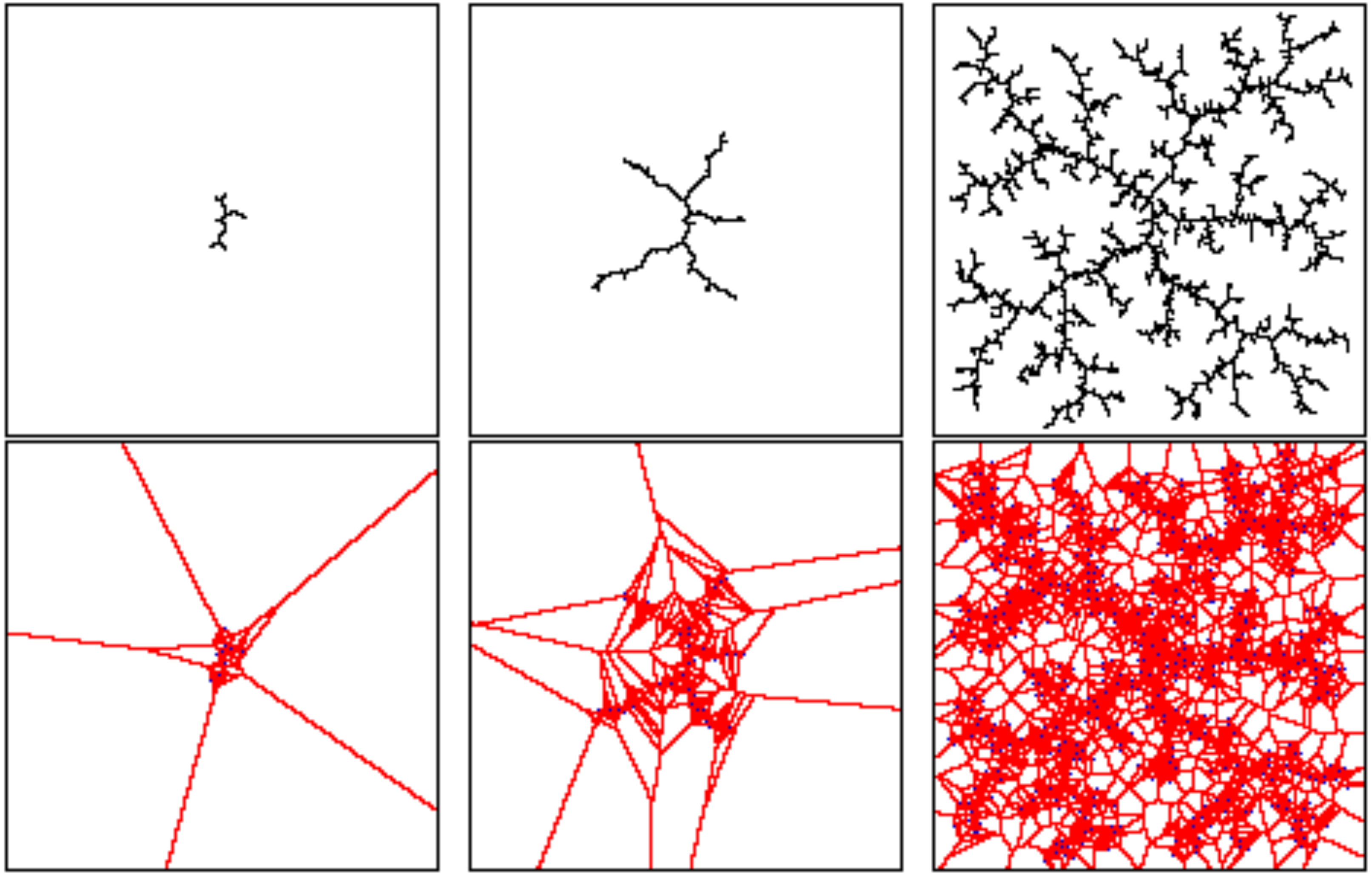
```
BUILD_RRT(q_init)
    T.init(q_init);
    for k = 1 to K do
        q_rand ← RANDOM_CONFIG();
        EXTEND(T, q_rand);
    Return T
```

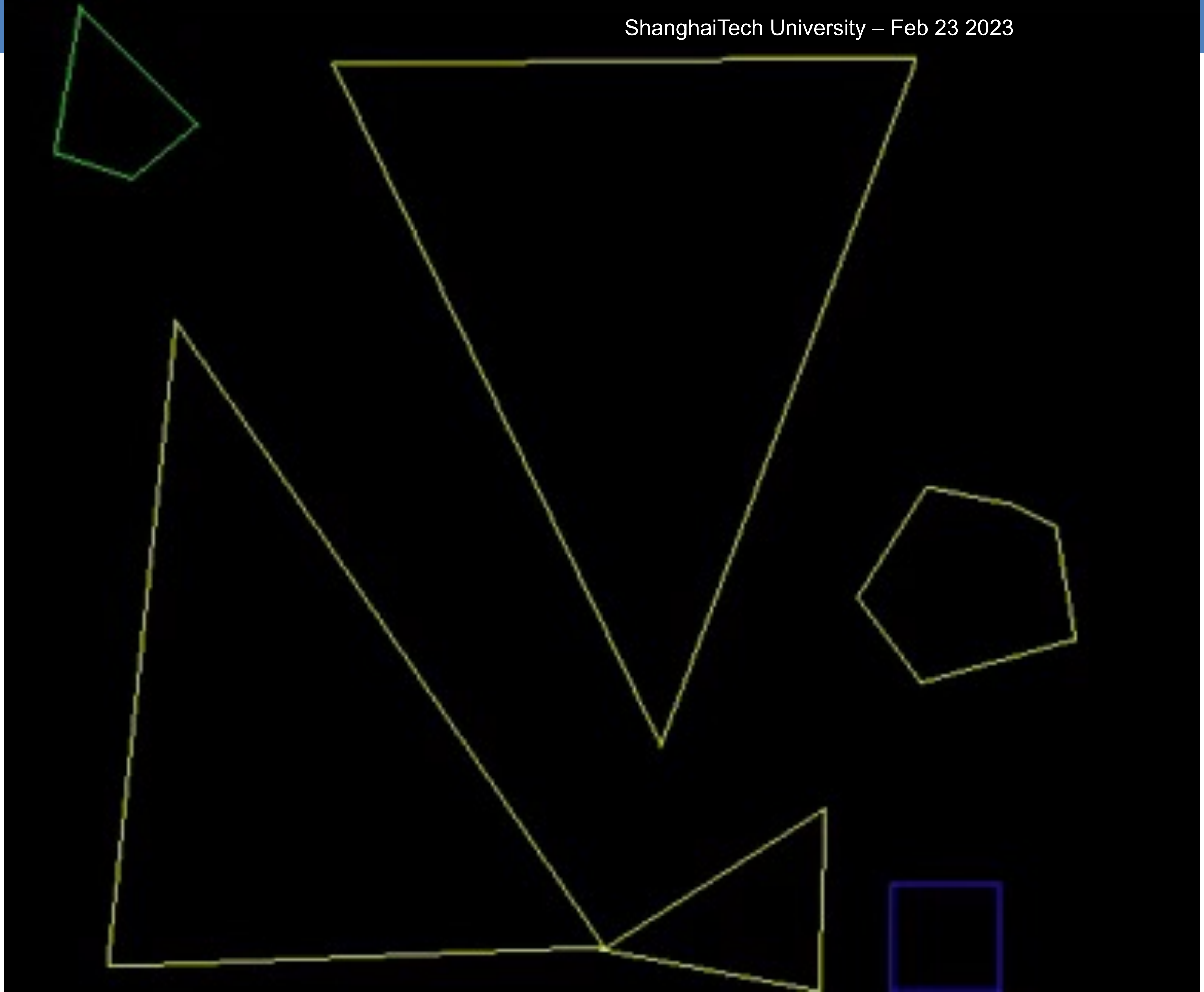
```
EXTEND(T, q)
    q_near ← NEAREST_NEIGHBOR(q, T);
    if NEW_CONFIG(q, q_near, q_new) then
        T.add_vertex(q_new);
        T.add_edge(q_near, q_new);
        if q_new = q then
            Return Reached;
        else
            Return Advanced;
    Return Trapped;
```

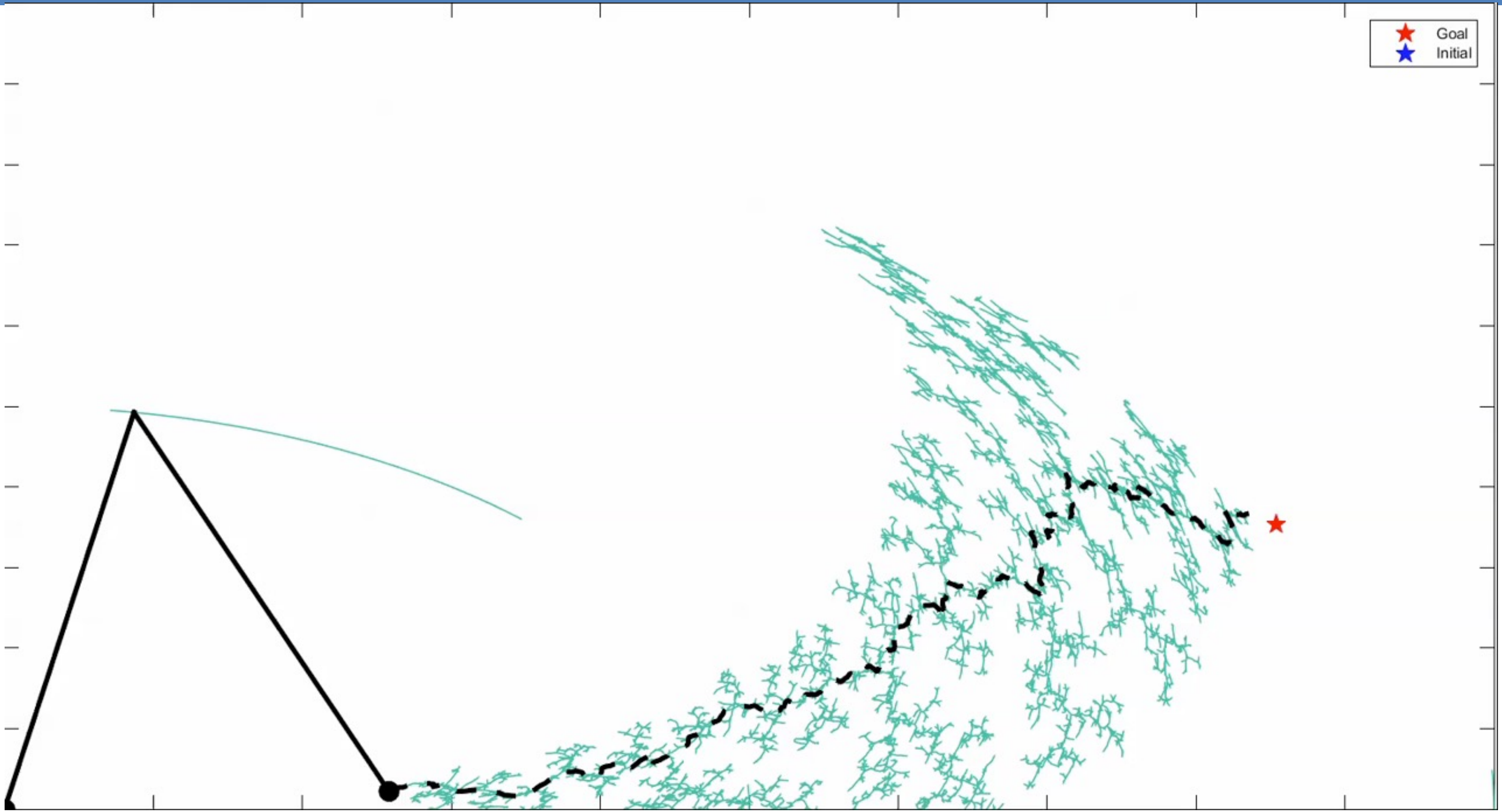


# Why are RRT's rapidly exploring?

The probability of a node to be selected for expansion is proportional to the area of its Voronoi region



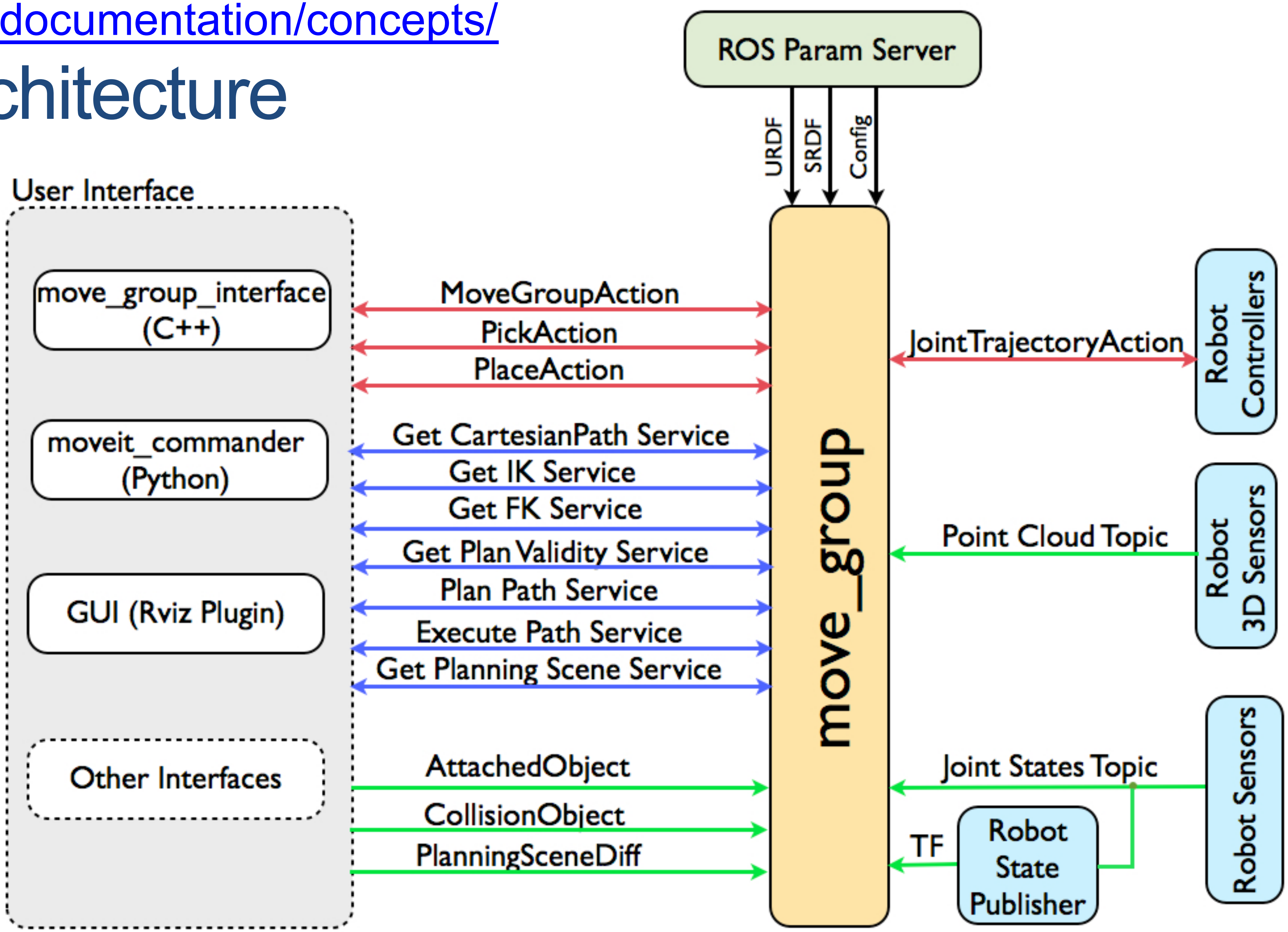



Goal
Initial

# MOVEIT & GRASPING

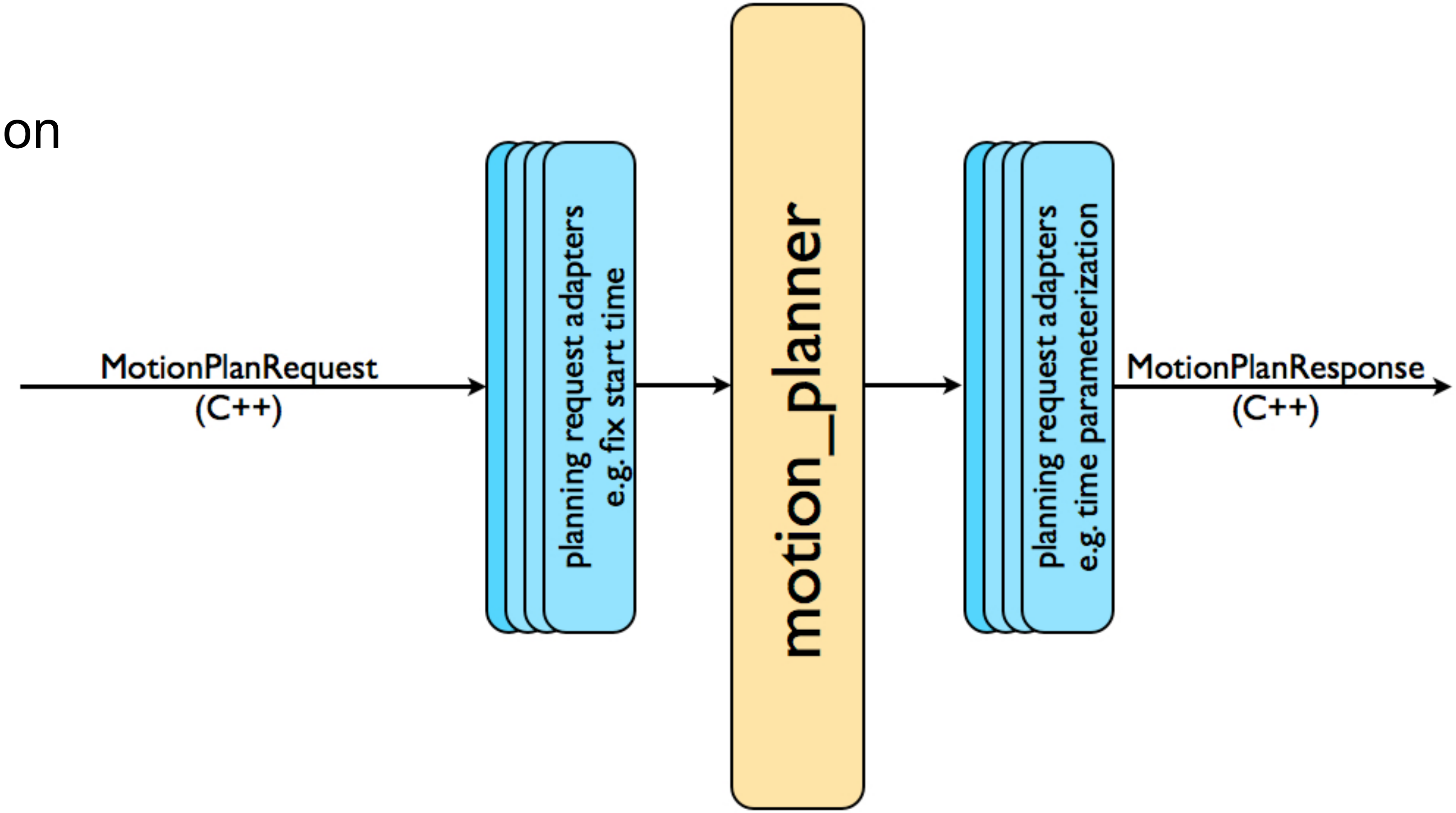
http://moveit.ros.org/documentation/concepts/

# System Architecture

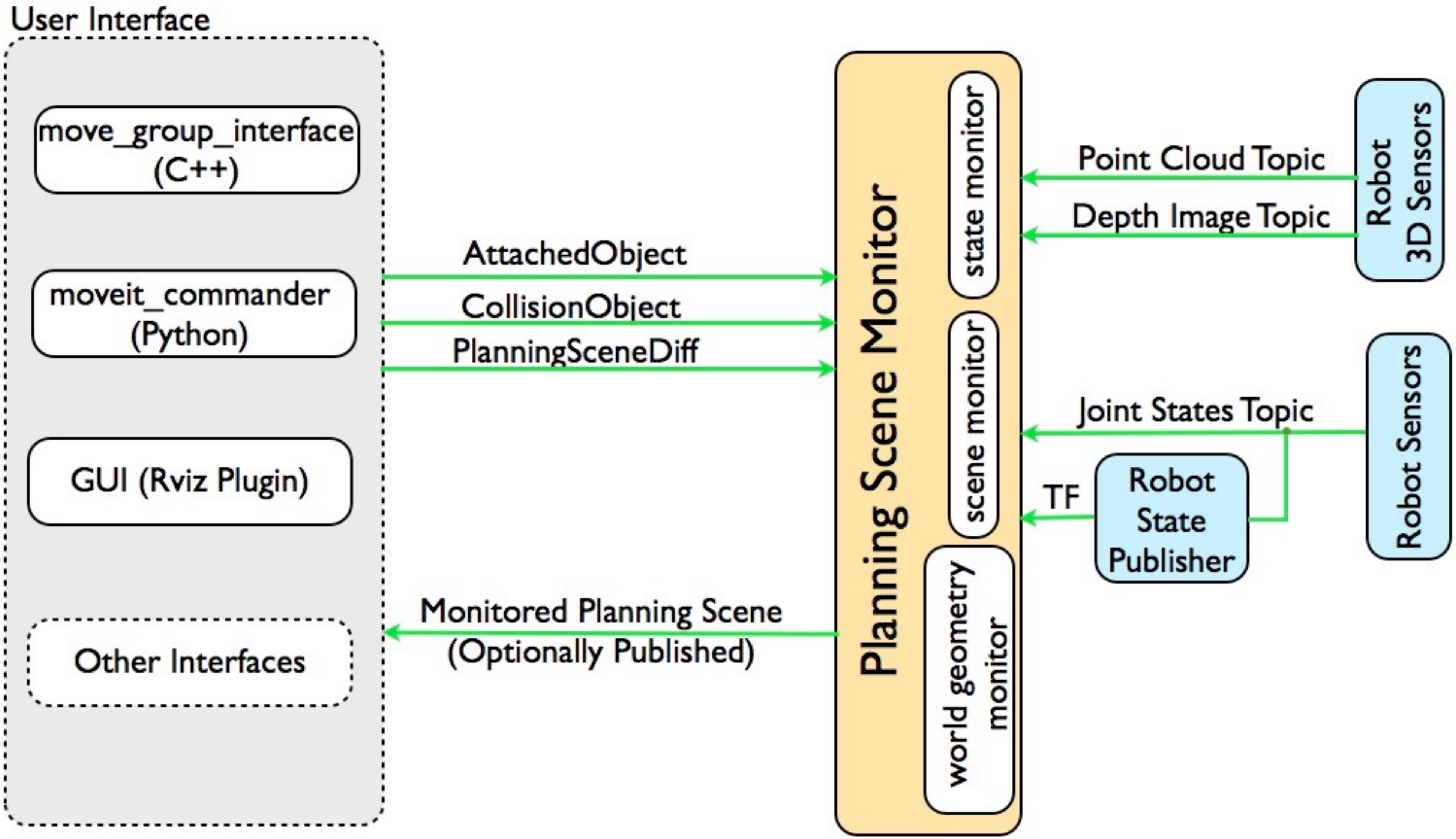


# Motion Planning

- Mainly:
- OMPL (Open Motion Planning Library)

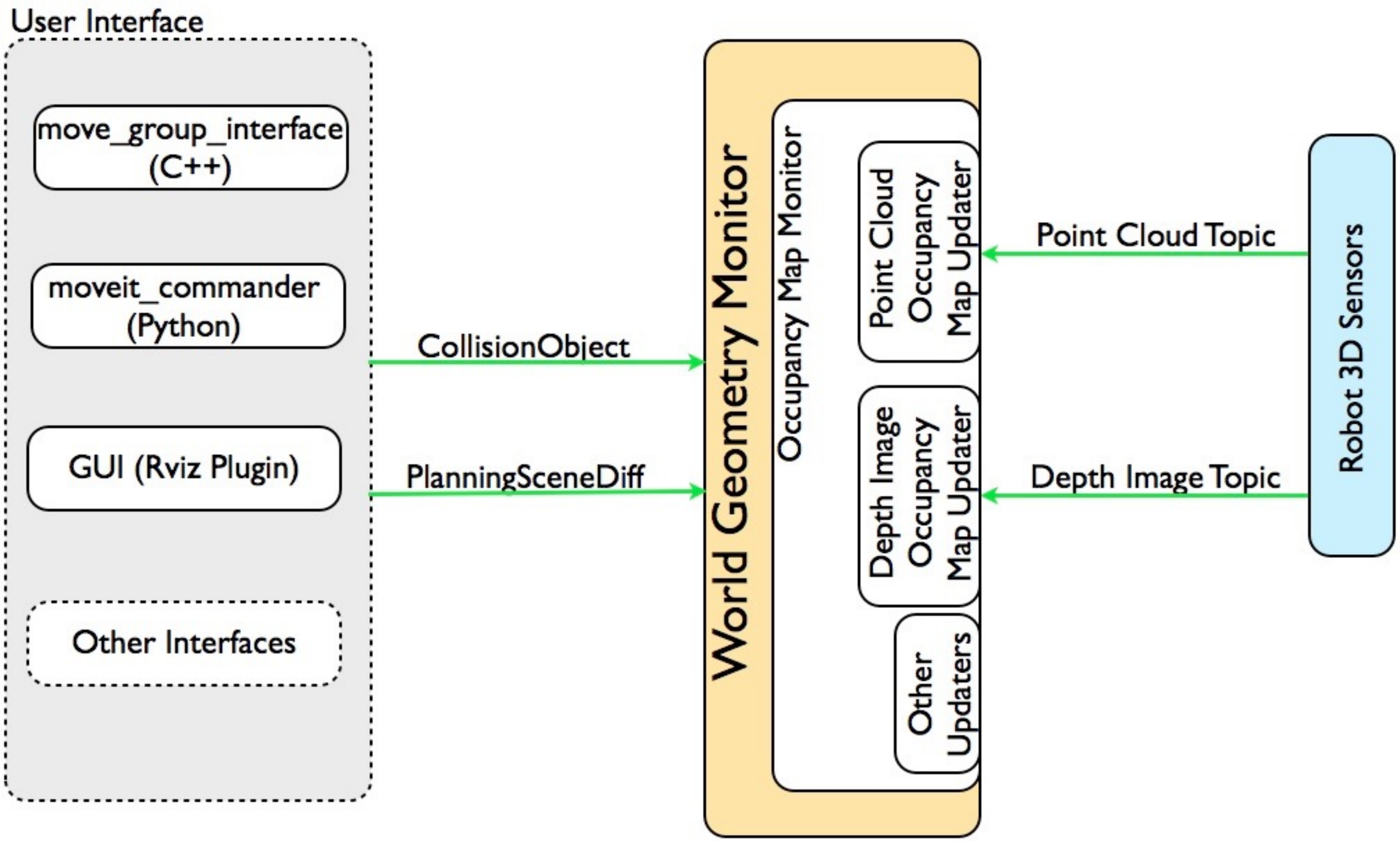


# Planning Scene



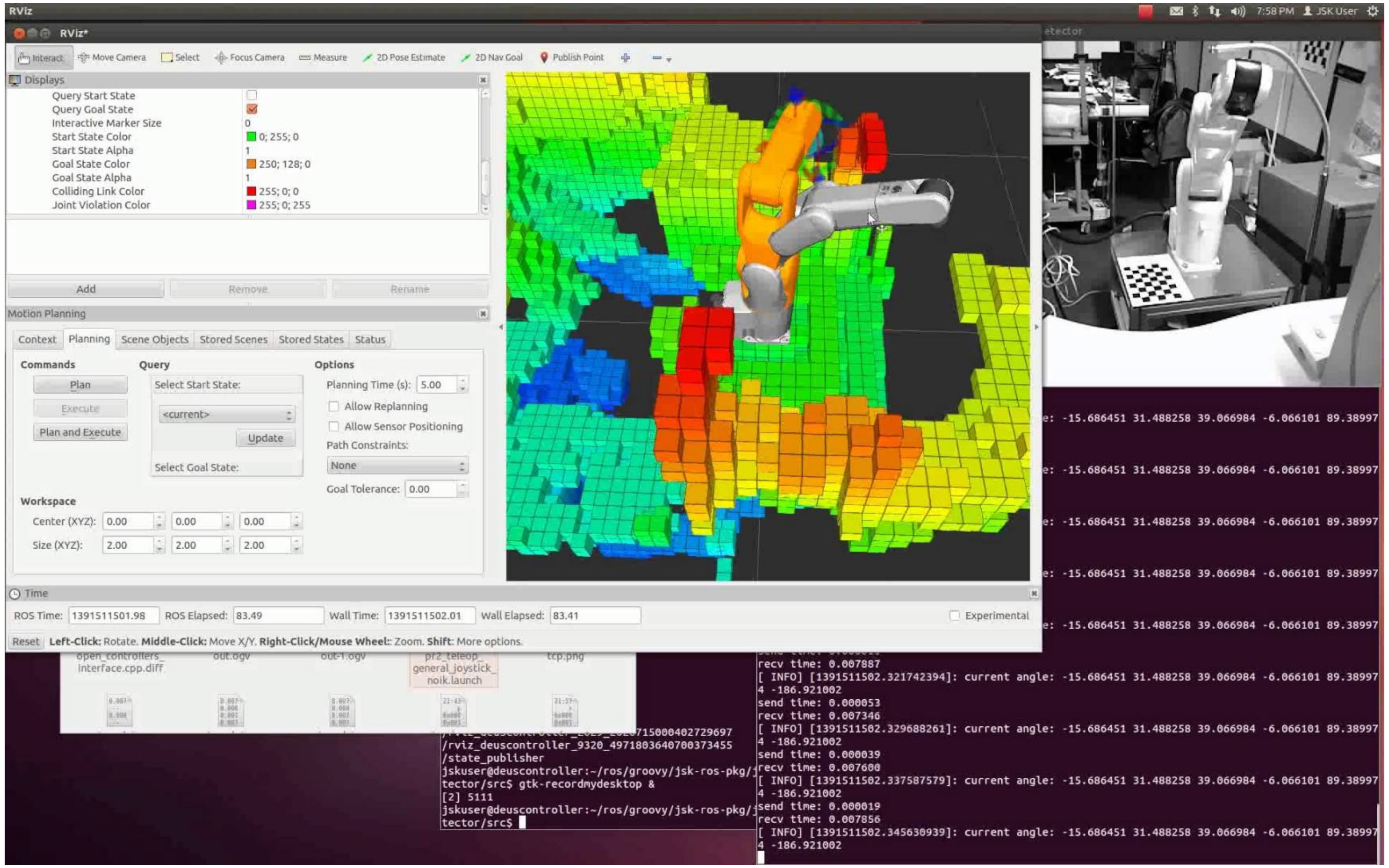
# 3D Perception

- Octomap



# Octomap / Octree

- Depth sensor (usually RGBD)
- http://wiki.ros.org/octomap

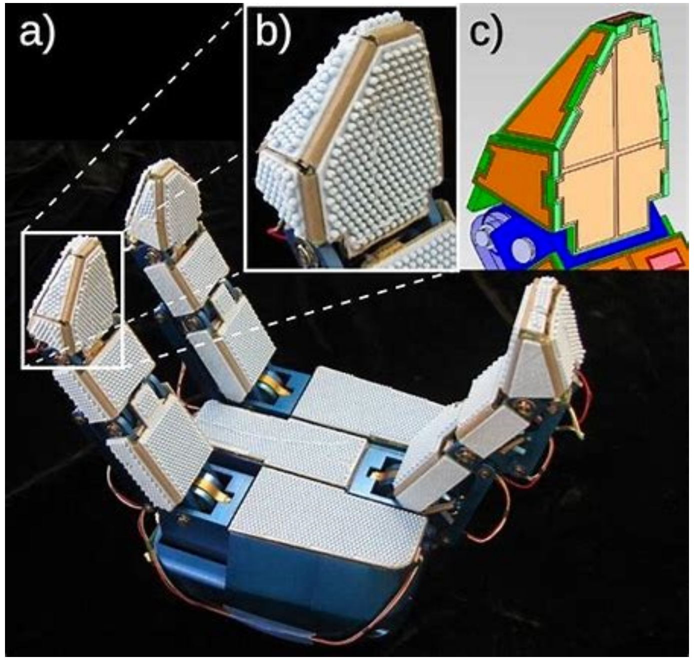


# Grasp an Object: Steps

1. Startup robot and sensors
2. Detect object & its pose
3. Select grasping points on the object
4. Scan the scene and environment (for collision checking later)
5. Use IK to check if grasping point can be reached – checks for collisions – may try thousands of possibilities (before concluding that there is always a collision)
6. Use motion planning to plan from current pose to goal pose: Lots of collision checks! Might realize that it is impossible after a long time
7. Execute that trajectory: Check if we reached the intermediate pose (within the time constraint) and command the next
8. Controller: take dynamics into account to move to the next intermediate pose
9. Once goal is reached close fingers.
10. Check if object is in fingers
11. Add the object to the collision description of the robot
12. Plan the path to the goal pose…

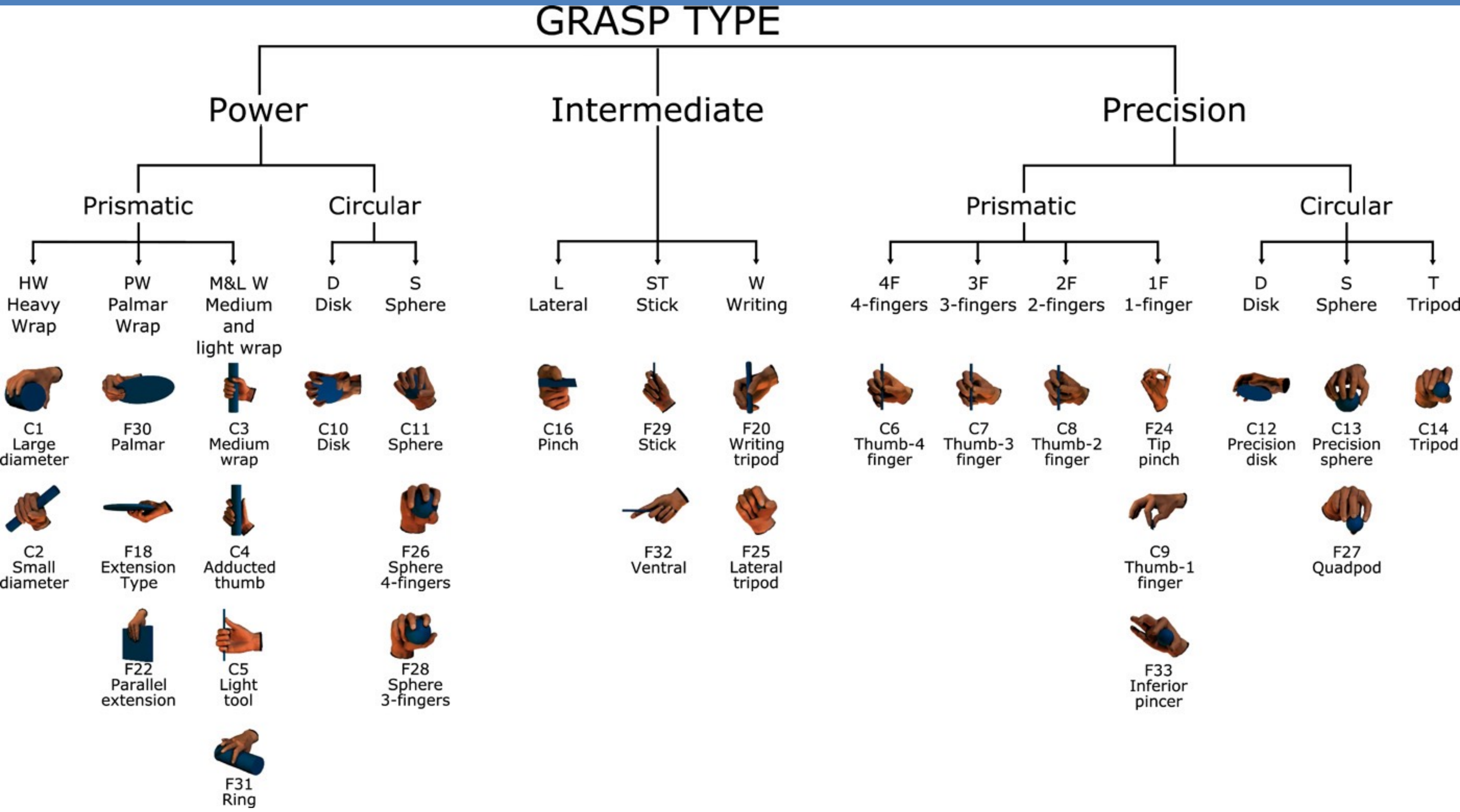
# Grasping …

- Gripper: Parallel; 3 Finger; Hand; Suction
- Force Sensors
- Tactile Sensors
- Grasp Types
- Grasp Planning

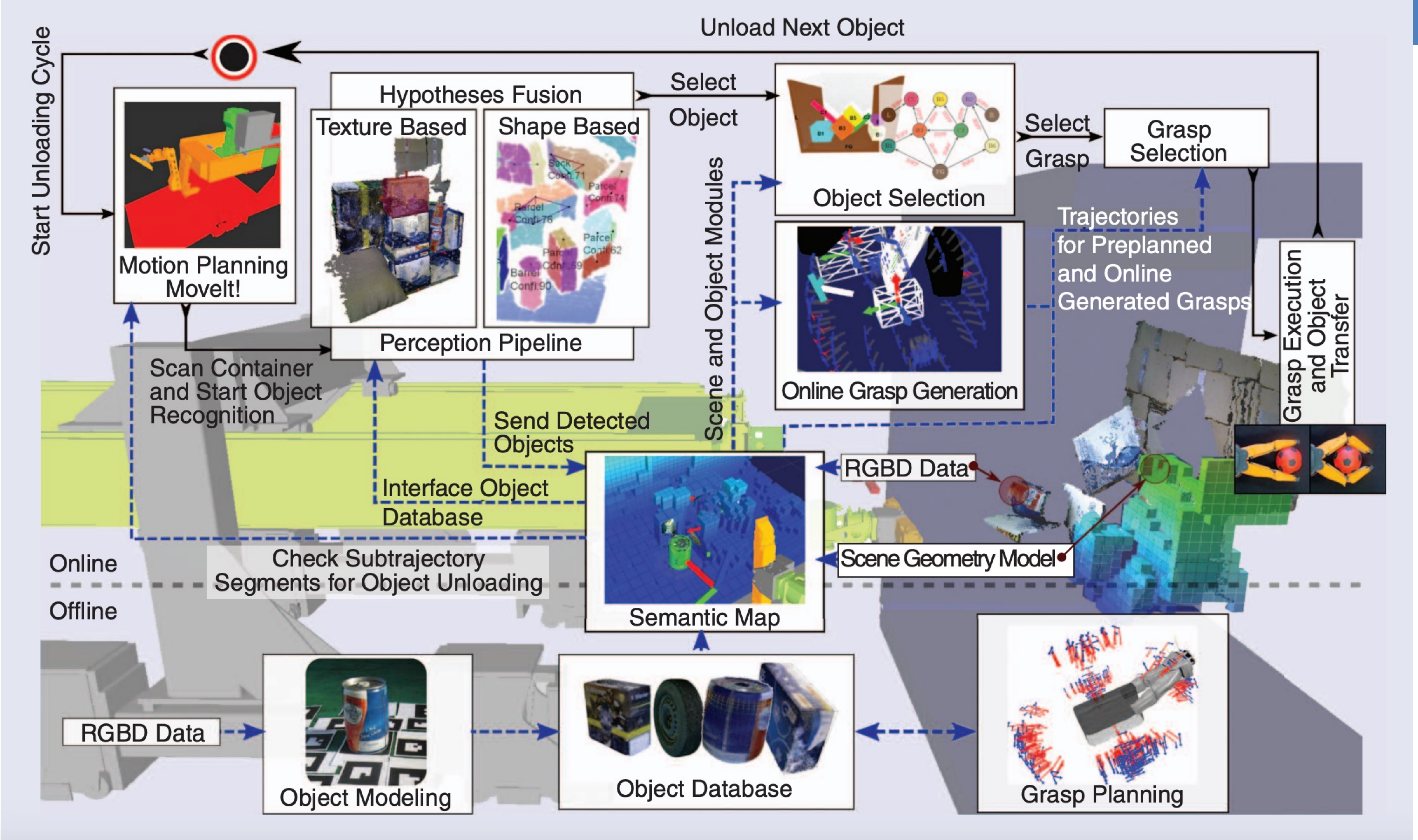



*"Biologically inspired tactile classification of object-hand and object-world interactions"*, by B. Heyneman and M. R. Cutkosky; Stanford University; Robio 2012

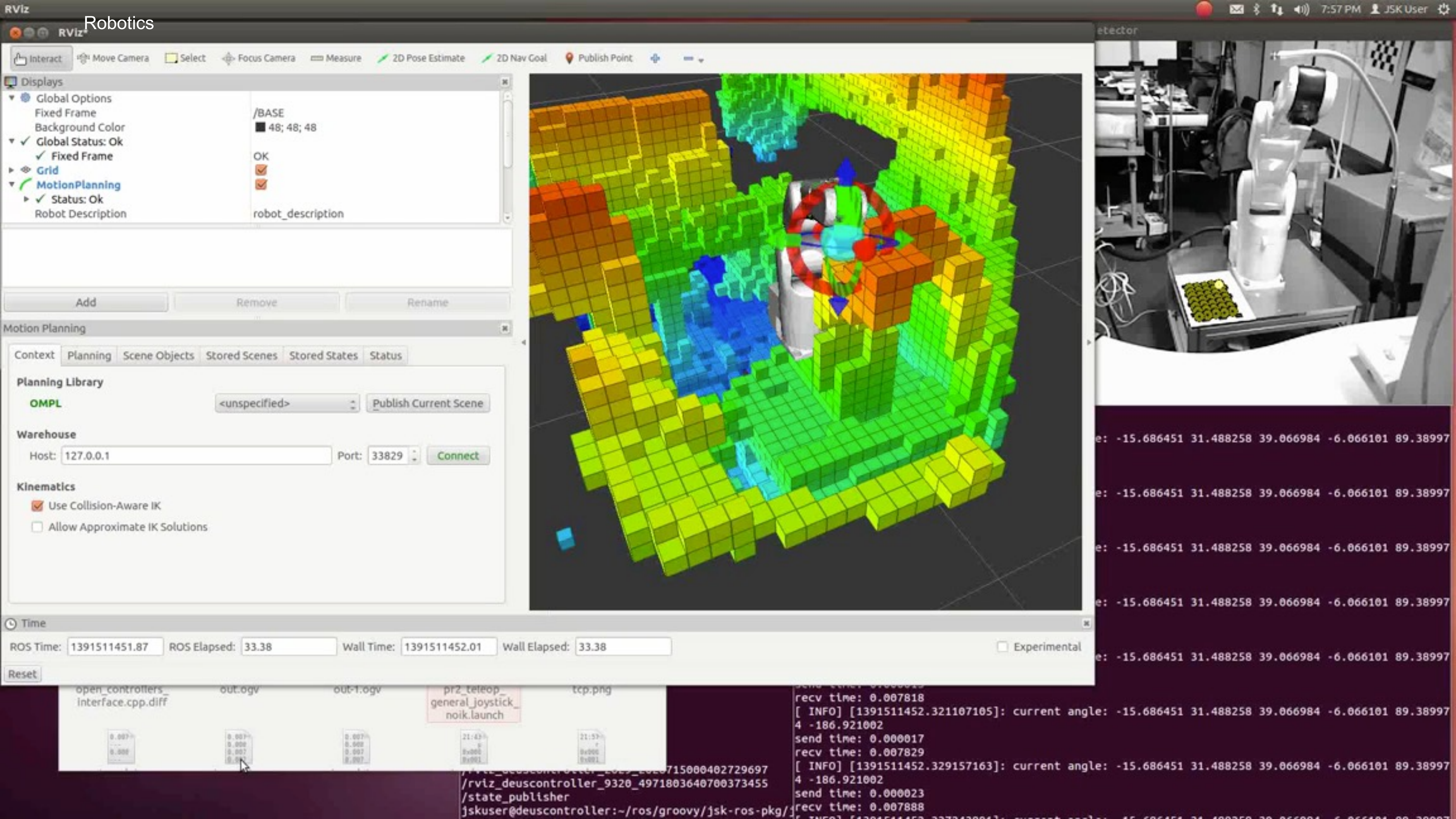
Citation for image on page +3: T. Stoyanov *et al.*, "No More Heavy Lifting: Robotic Solutions to the Container Unloading Problem," in *IEEE Robotics & Automation Magazine*, vol. 23, no. 4, pp. 94-106, Dec. 2016.


GRASP TYPE
Power
Intermediate
Precision
Prismatic
Circular
HW Heavy Wrap
PW Palmar Wrap
M&L W Medium and light wrap
D Disk
S Sphere
L Lateral
ST Stick
W Writing
Prismatic
Circular
4F 4-fingers
3F 3-fingers
2F 2-fingers
1F 1-finger
D Disk
S Sphere
T Tripod
C1 Large diameter
C2 Small diameter
F30 Palmar
F18 Extension Type
F22 Parallel extension
C3 Medium wrap
C4 Adducted thumb
C5 Light tool
F31 Ring
C10 Disk
C11 Sphere
F26 Sphere 4-fingers
F28 Sphere 3-fingers
C16 Pinch
F29 Stick
F32 Ventral
F20 Writing tripod
F25 Lateral tripod
C6 Thumb-4 finger
C7 Thumb-3 finger
C8 Thumb-2 finger
F24 Tip pinch
C9 Thumb-1 finger
F33 Inferior pincer
C12 Precision disk
C13 Precision sphere
F27 Quadpod
C14 Tripod

Lu, Qingkai & Hermans, Tucker. (2019). Modeling Grasp Type Improves Learning-Based Grasp Planning.


Unload Next Object
Start Unloading Cycle
Hypotheses Fusion
Texture Based
Shape Based
Select Object
Object Selection
Select Grasp
Grasp Selection
Motion Planning MoveIt!
Perception Pipeline
Scene and Object Modules
Online Grasp Generation
Trajectories for Preplanned and Online Generated Grasps
Grasp Execution and Object Transfer
Scan Container and Start Object Recognition
Send Detected Objects
Interface Object Database
RGBD Data
Scene Geometry Model
Semantic Map
Check Subtrajectory Segments for Object Unloading
Online
Offline
RGBD Data
Object Modeling
Object Database
Grasp Planning

# Robotics


RViz
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
/BASE
Background Color
48; 48; 48
Global Status: Ok
Fixed Frame
OK
Grid
MotionPlanning
Status: Ok
Robot Description
robot_description
Add
Remove
Rename
Motion Planning
Context
Planning
Scene Objects
Stored Scenes
Stored States
Status
Planning Library
OMPL
<unspecified>
Publish Current Scene
Warehouse
Host: 127.0.0.1
Port: 33829
Connect
Kinematics
Use Collision-Aware IK
Allow Approximate IK Solutions
Time
ROS Time: 1391511451.87
ROS Elapsed: 33.38
Wall Time: 1391511452.01
Wall Elapsed: 33.38
Experimental
Reset
recv time: 0.007818
[ INFO] [1391511452.321107105]: current angle: -15.686451 31.488258 39.066984 -6.066101 89.38997
4 -186.921002
send time: 0.000017
recv time: 0.007829
[ INFO] [1391511452.329157163]: current angle: -15.686451 31.488258 39.066984 -6.066101 89.38997
4 -186.921002
send time: 0.000023
recv time: 0.007888
/rviz_deuscontroller_9320_4971803640700373455
/state_publisher

# ADMIN

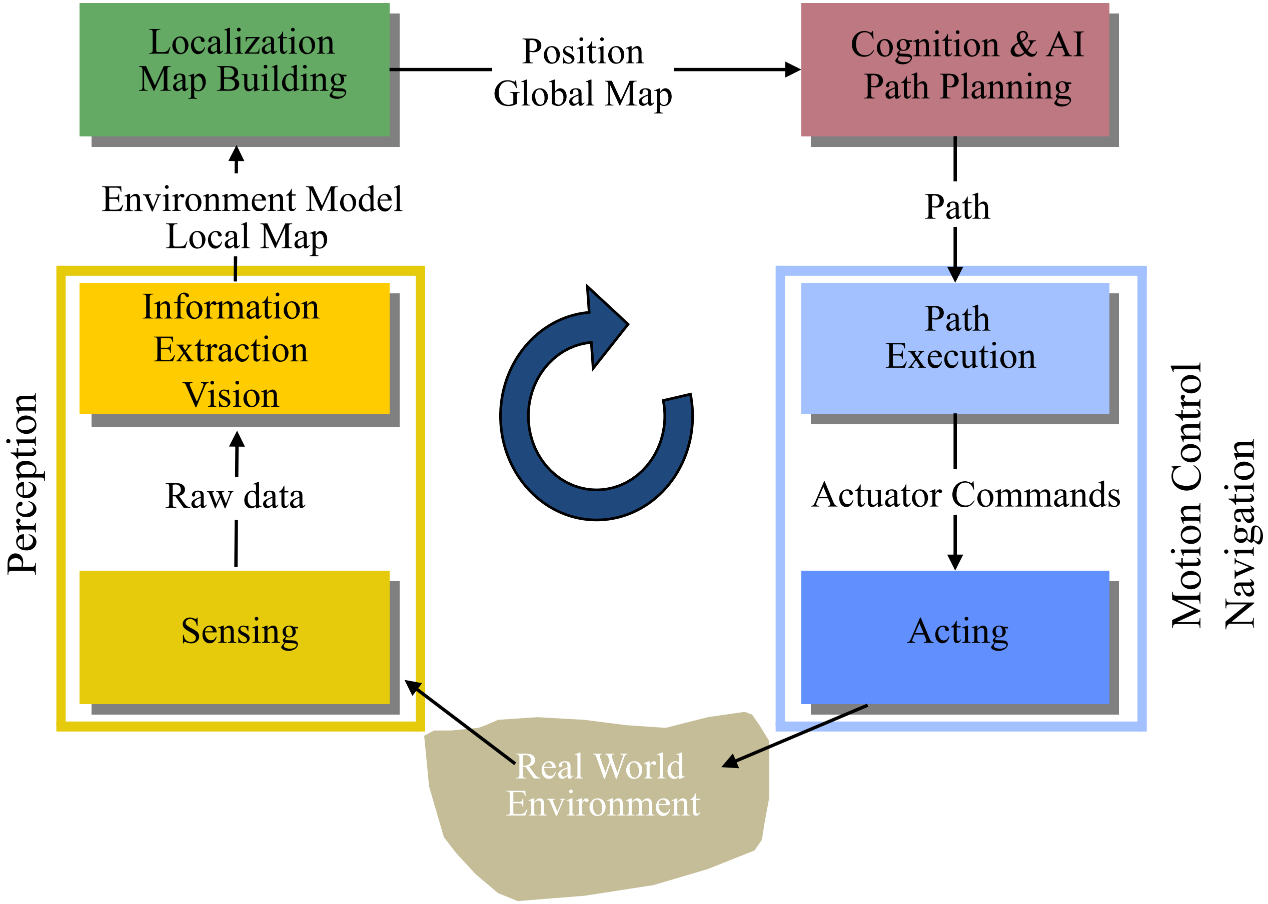
# Admin

- HW 2 published – due in 2 weeks (March 9).
  - Big homework – start early
  - Ask TA or Prof for help – we are happy to help! Don’t cheat by asking other students!
- Paper presentation & Project proposal instructions published
- Schedule a meeting with Prof. and graduate advisor for next week (via email)!!

| Project | Graduate Supervisor | Student |
|---|---|---|
| Campus Autonomy: Hunter SE self ch: | Chengqian Li | 刘志成 (Zhicheng Liu) |
| | | 庄凌颖 (Lingyin Zhuang) |
| Fetch Robot: Whiteboard Cleaning | Yinjie Li | 阳雅珣 (Yaxun Yang) |
| | | 孟相汀 (Xiangting Meng) |
| Learn to fold | Ziwen Zhuang | 林霄竹 (Xiaozhu Lin) |
| | | 李新龙 (Xinlong Li) |
| Calibration for Mapping Robot | Wenqing Jiang | 徐博文 (Bowen Xu) |
| | | 刘滔 (Tao Liu) |
| OminiWheg Robot | Prof. Schwertfeger | 辜俊 (Jun Gu) |

# MAP REPRESENTATION

# General Control Scheme for Mobile Robot Systems

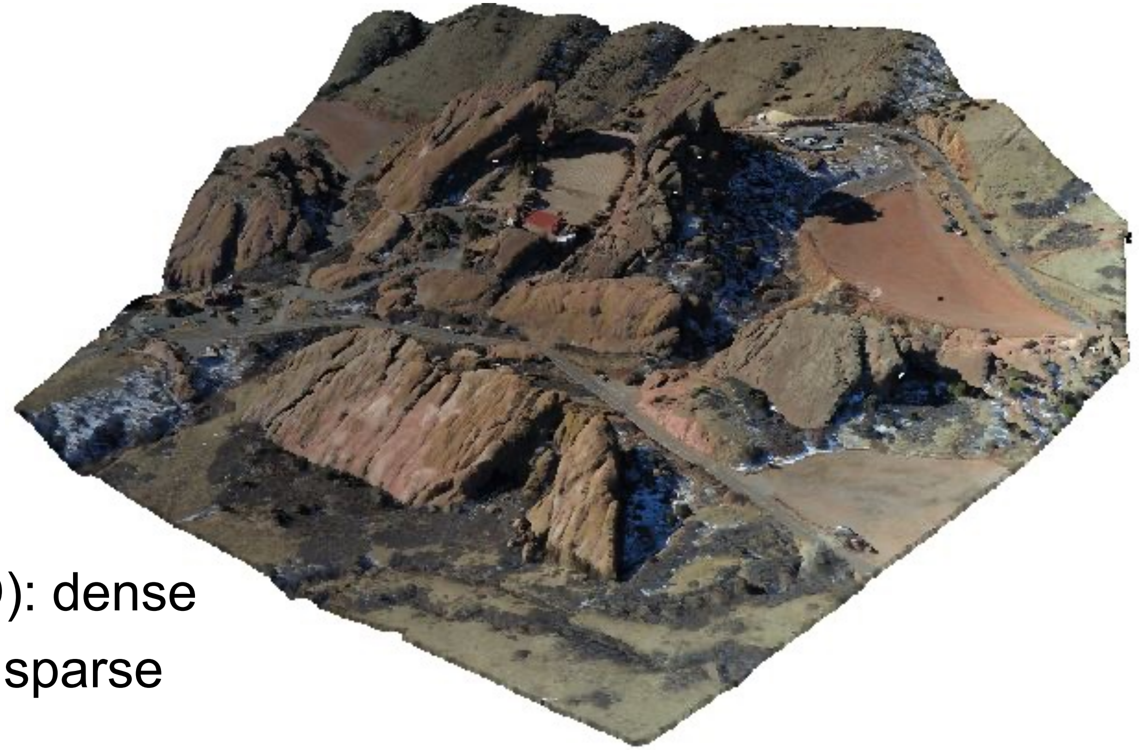



With material from Roland Siegwart and Davide Scaramuzza, ETH Zurich

# Map Representation: what is “saved” in the map

- Points (surface of objects, buildings): 2D or 3D
  - What: x,y or x,y,z coordinates; Optional: intensity; maybe RGB; maybe descriptor; temperature; …
  - From range sensors (laser, ultrasound, stereo, RGB-D): dense
  - From cameras (structure from motion; feature points): sparse
  - Variant: kd-tree
- Grid-map: 2D or 3D
  - Option: probabilistic grid map
  - Option: elevation map
  - Option: cost map
  - Option: Truncated Signed Distance Field
  - Option: Normal Distributions Transform (NDT)
  - Variant: Quad-tree; Oct-tree
- Higher-level Abstractions
  - Lines; Planes; Mesh
  - Curved: splines; Superquadrics
- Semantic Map
  - Assign semantic meaning to entities of a map representation from above
  - E.g. wall, ceiling, door, furniture, car, human, tree, …
- Topologic Map
  - High-level abstraction: places and connections between them
- Hierarchical Map
  - Combine Maps of different scales. E.g.:
  - Campus, building, floor
- Pose-Graph Based Map
  - Save (raw) sensor data in graph, annotated with the poses; generate maps on the fly
- Dynamic Map
  - Capture changing environment
- Hybrid Map
  - Combination of the above

# Point based map

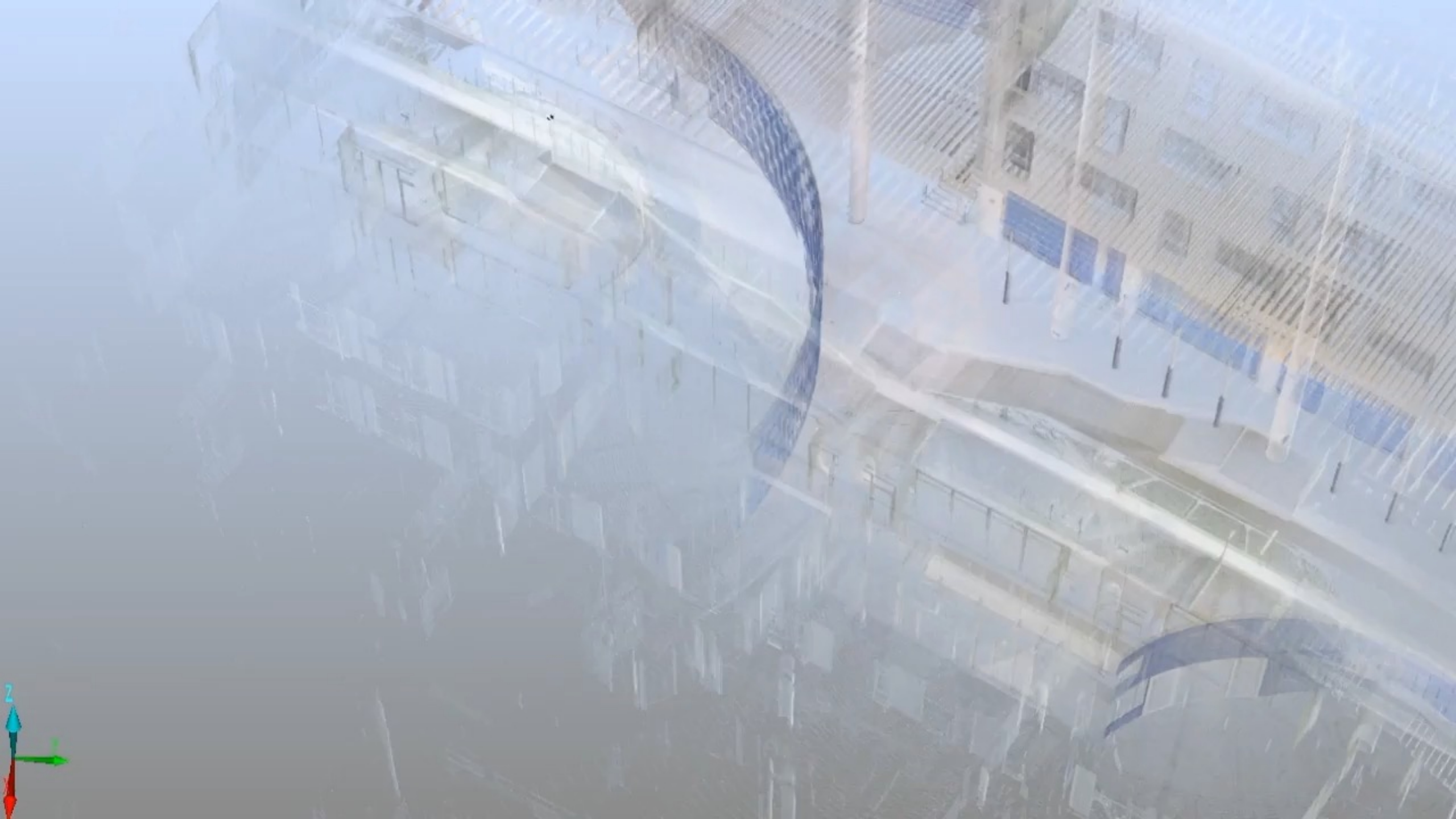


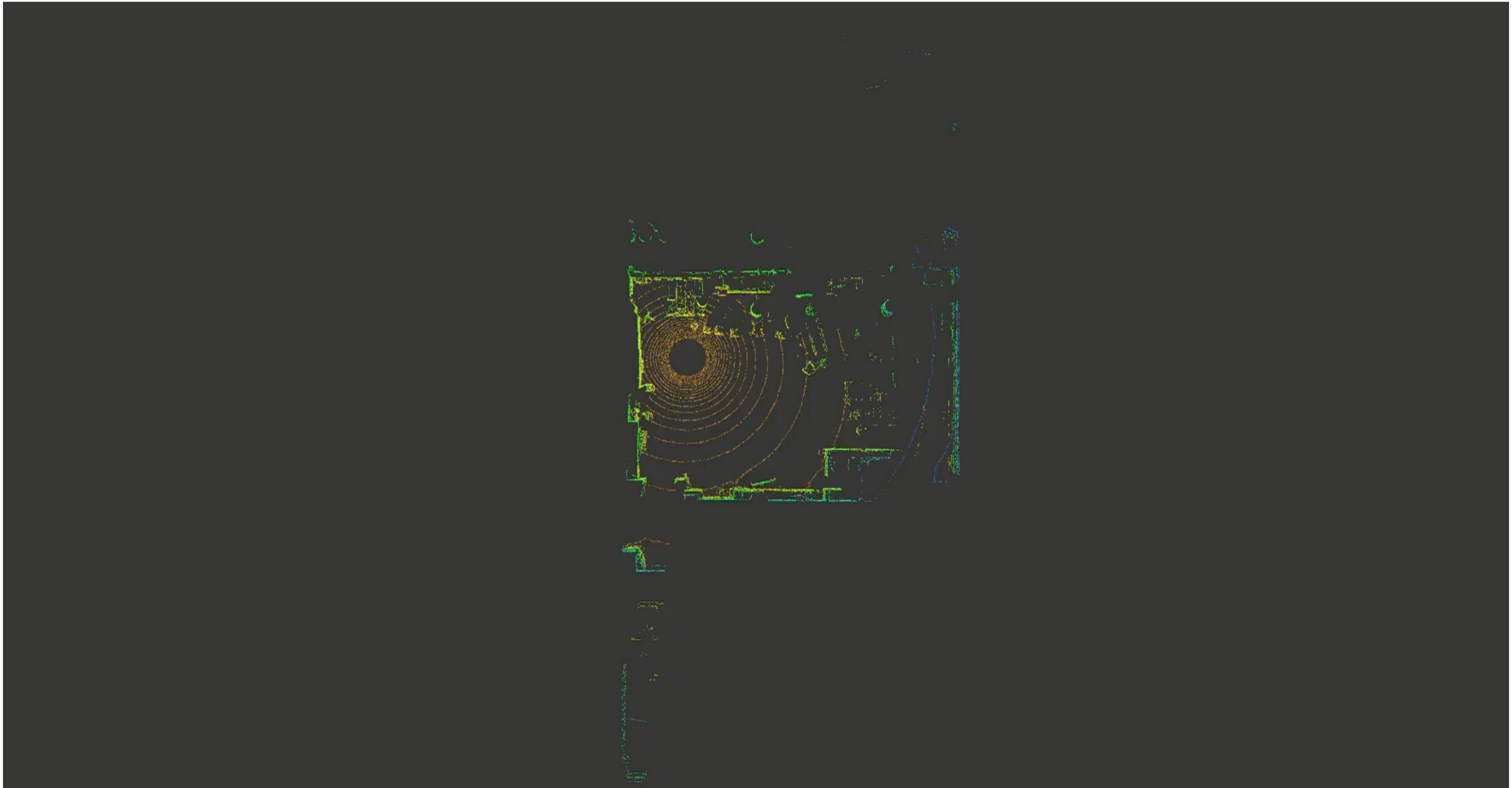
- Point Cloud
  - What: x,y or x,y,z coordinates; Optional: intensity; maybe RGB; maybe descriptor; temperature; …
  - From range sensors (laser, ultrasound, stereo, RGB-D): dense
  - From cameras (structure from motion; feature points): sparse
  - Variant: kd-tree
  - Structured point cloud: points are ordered in a 2D grid -> could calculate depth image from it
  - PCL: point cloud library: Used in ROS; https://pointclouds.org/
  - sensor_msgs/PointCloud sensors_msgs/PointCloud2
  - Excellent viewer: CloudCompare https://www.danielgm.net/cc/
  - File formats: PLY (bin & ASCII); XZY (ASCII); PCD (from pcl); LAS (terrestrial scanning); …

# RGBD-Inertial Trajectory Estimation and Mapping for Ground Robots

Zeyong Shan, Ruijian Li and Sören Schwertfeger

Source code: https://github.com/STAR-Center/VINS-RGBD

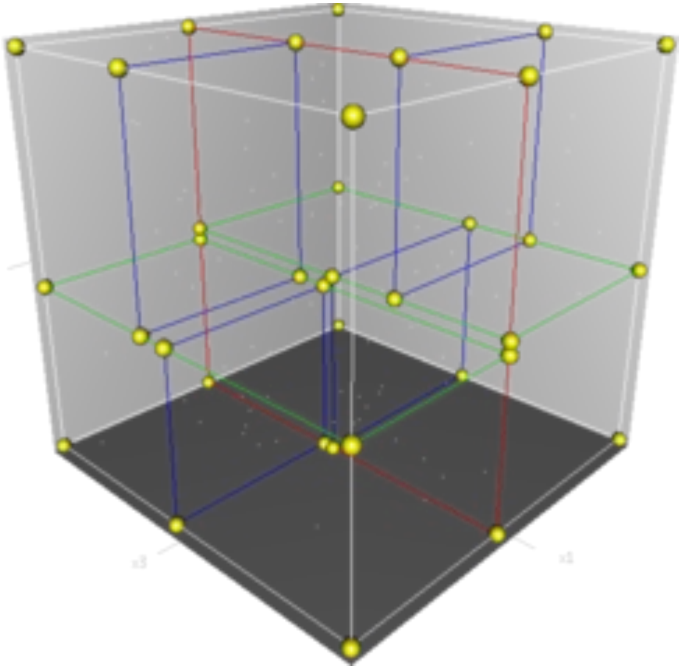


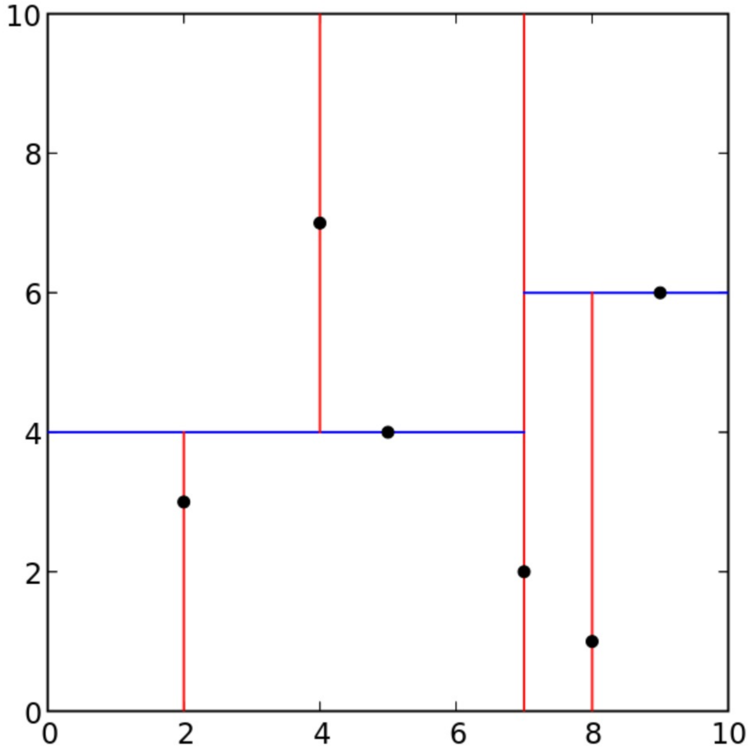


# k-d tree

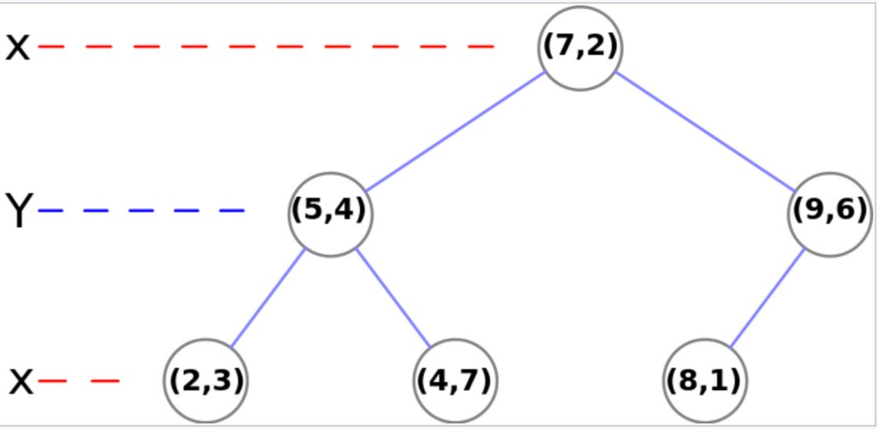
- k-dimensional binary search tree
- Robotics: typically 3D or 2D
- Every level of tree:
  - For a different axis (e.g. x,y,z,x,y,z,x,y,z) (=> split space with planes)
  - Put points in left or right side based on median point (w.r.t. its value of on the current axis) =>
  - Balanced tree
- Fast neighbor search -> ICP!

| Algorithm | Average | Worst case |
|---|---|---|
| Space | $O(n)$ | $O(n)$ |
| Search | $O(\log n)$ | $O(n)$ |
| Insert | $O(\log n)$ | $O(n)$ |
| Delete | $O(\log n)$ | $O(n)$ |



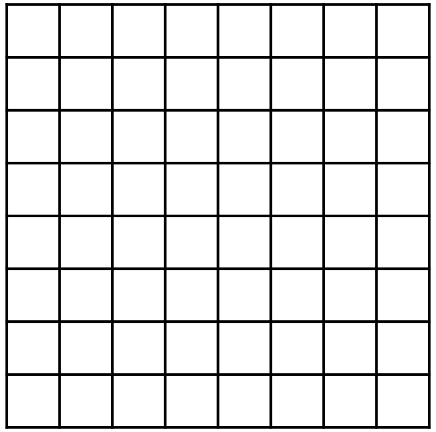


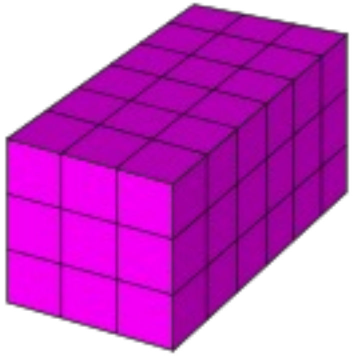

*k*-d tree decomposition for the point set `(2,3), (5,4), (9,6), (4,7), (8,1), (7,2)`.



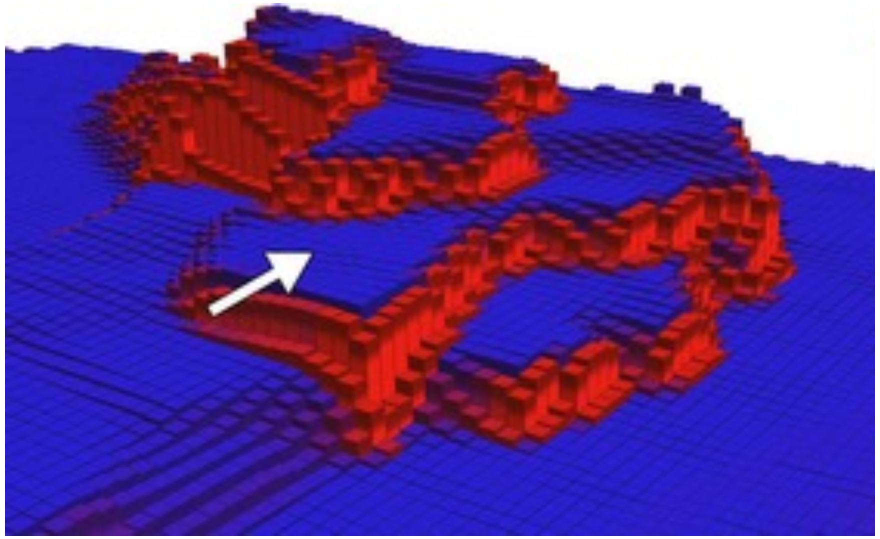

The resulting *k*-d tree.

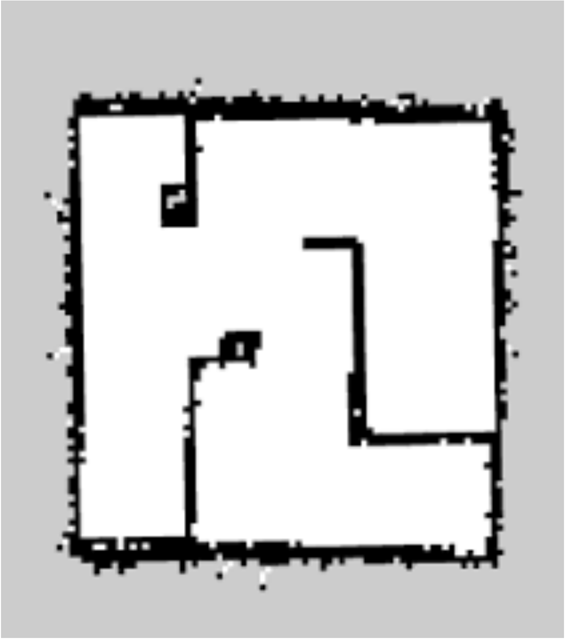
# Grid Maps





- Grid-map: 2D or 3D cartesian grid
- What to save in the cell:
  - Binary: Free; Occupied;
  - Colored: +Unknown; +Searched; +Path; …
  - Probability of being occupied 0…1.0
  - Height above ground: Elevation Map
  - Cost map: used for planning (covered later)





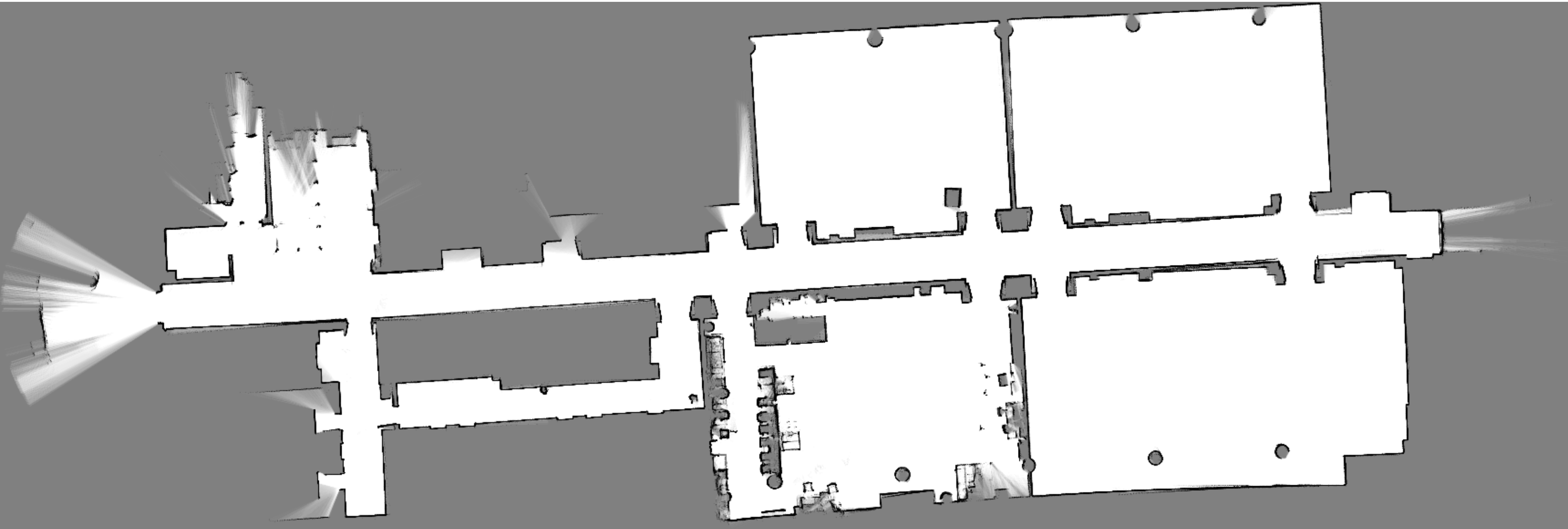
https://emanual.robotis.com/docs/en/platform/turtlebot3/slam/

# Grid Maps

- Resolution: e.g. 1 pixel == 5cm
- Size: e.g. 2000 pixel x 2000 pixel
- Scale: e.g. 1:100.000 (2DoF of 3 parameters resolution, size, scale)
- Frame
  - A map has a frame. Where? In the center of the image? Top left corner?
- Time stamp
- Formats:
  - Image: png, pgm (grey scale), tiff, jpeg (lossy!)
  - GeoTIFF: georeferenced TIFF (inlined tfw World File): map projection, coordinate systems, ellipsoids, datums (see GPS slide)
  - 3D: bt (octomap)
- QGIS: open source Geographic Information System for raster (image) and vector data http://qgis.org/en/site

# Probabilistic grid map

- 1: occupied; 0: free; 0.5: unknown
- Need error model of sensor (and of localization) to properly update cells with a scan
- Can remove dynamic (moving) objects (by observing the free space multiple times)



# Normal Distributions Transform (NDT)

- Sparse Gaussian mixture model
- Each cell has NDT Computed from covariance matrix of points inside the cell
- Useful for scan matching/ registration

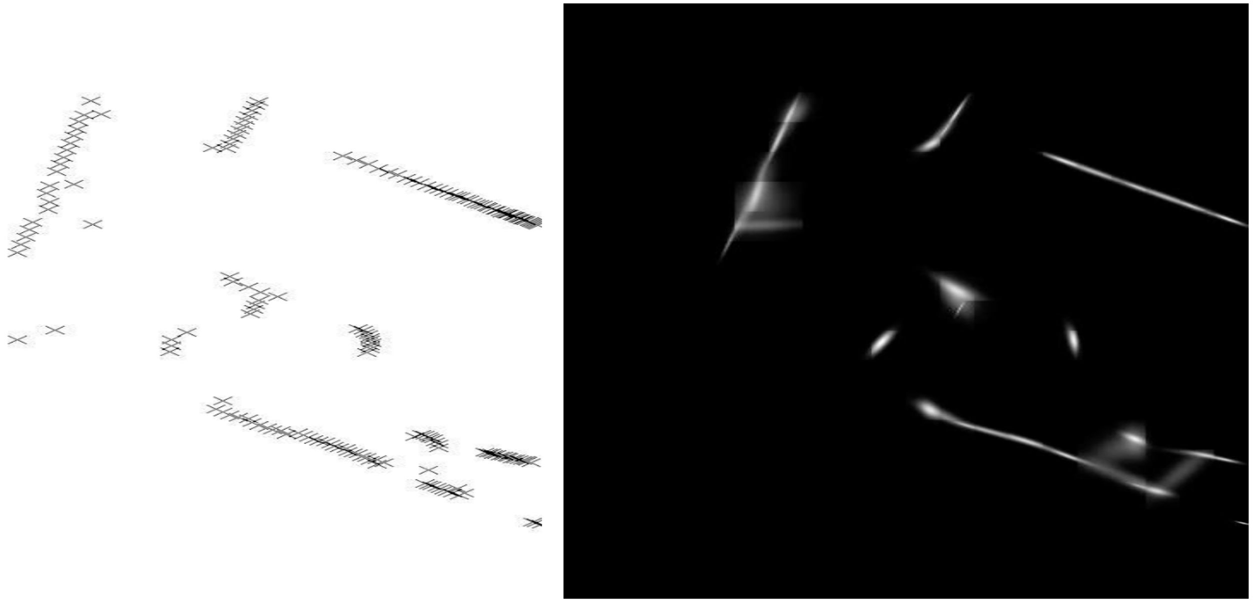
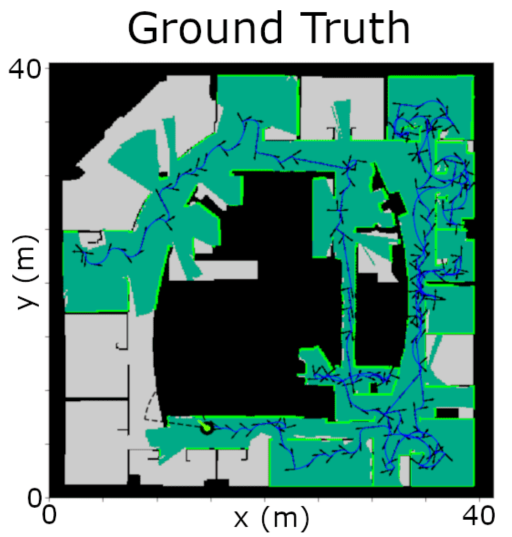


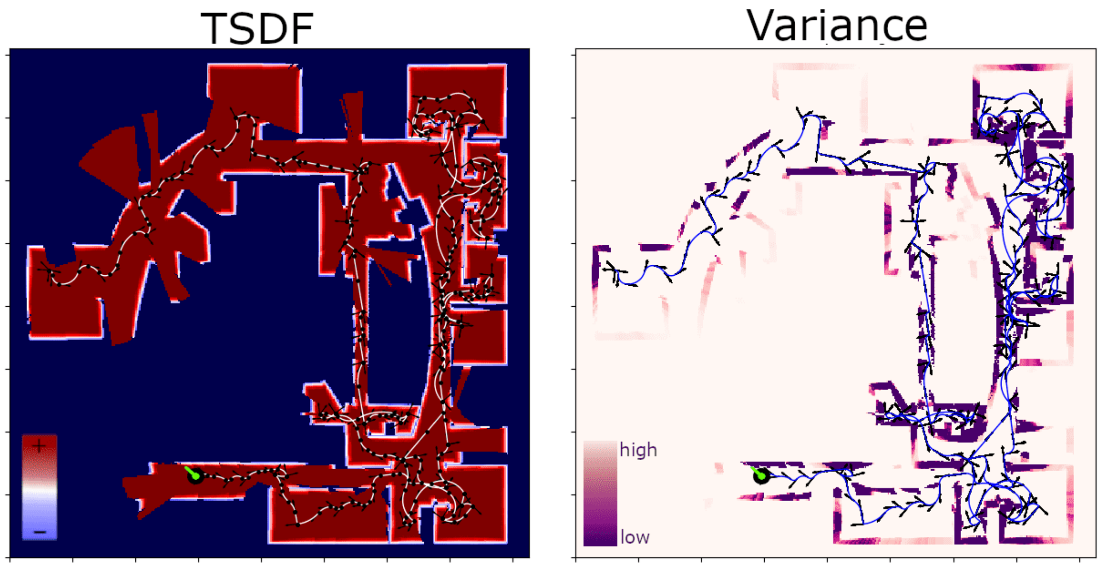
Fig. 1. An example of the NDT: The original laser scan and the resulting probability density.

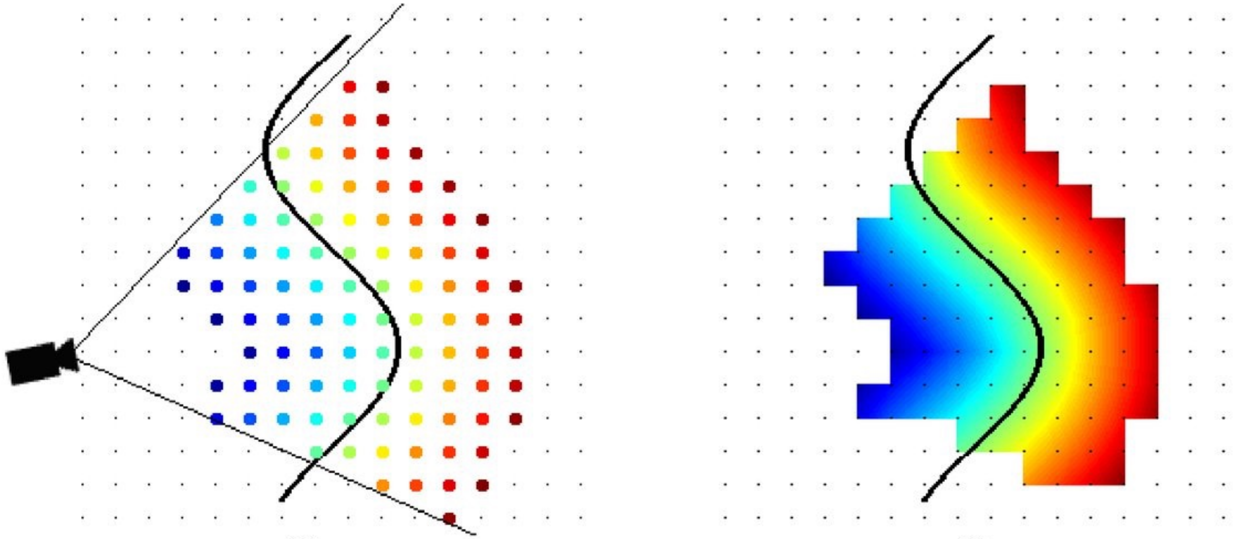
# Truncated Signed Distance Field

https://www.dcist.org/2020/03/20/active-exploration-in-signed-distance-fields/
https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9091/909117/Real-time-reconstruction-of-depth-sequences-using-signed-distance-functions/10.1117/12.2054158.short?SSO=1&tab=ArticleLinkFigureTable

- Distance to surface:
- positive in free space; negative behind
- Up to a certain distance (truncated)
- Useful for collision checking; planning; meshing
- Can be modeled using Gaussian random variable in each cell





(a) (b)