Robotic Information Garthering - Exploration of Unknown Environment

Jan Faigl

Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague

Lecture 05

B4M36UIR – Artificial Intelligence in Robotics

Overview of the Lecture

- Part 1 – Robotic Information Gathering - Robotic Exploration
  - Robotic Information Gathering
  - Robotic Exploration
  - TSP-based Robotic Exploration
  - Robotic Information Gathering

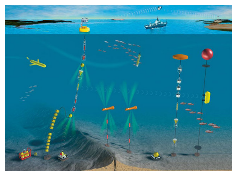
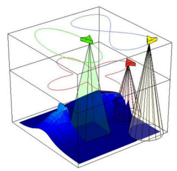
Part I

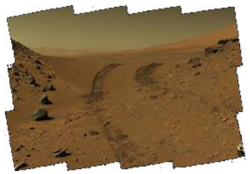
Part 1 – Robotic Exploration

Robotic Information Gathering

Create a model of phenomena by autonomous mobile robots performing measurements in a dynamic unknown environment.



Challenges in Robotic Information Gathering

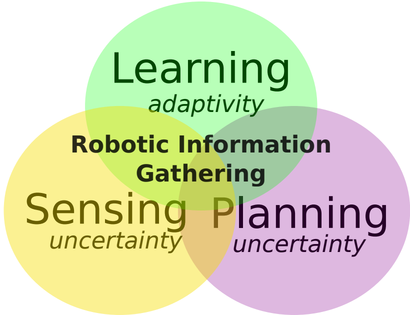
- **Where to take new measurements?**

  To improve the phenomena model
- **What locations visit first?**

  On–line decision–making
- **How to efficiently utilize more robots?**

  To divide the task between the robots
- **How to navigate robots to the selected locations?**

  Improve Localization vs Model



How to address all these aspects altogether to find a cost efficient solution using in–situ decisions?

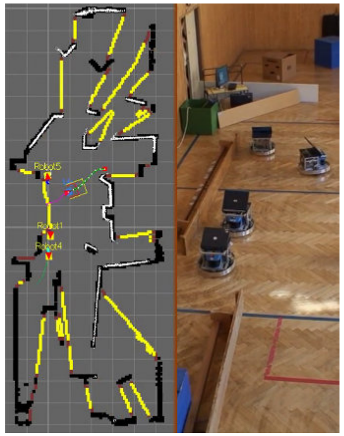
Robotic Exploration of Unknown Environment

- Robotic exploration is a fundamental problem of robotic information gathering
- The problem is:

  How to efficiently utilize a group of mobile robots to autonomously create a map of an unknown environment
  - Performance indicators vs constraints

    Time, energy, map quality vs robots, communication
  - Performance in a real mission depends on the on-line **decision-making**
  - It includes the problems of:
    - Map building and localization
    - Determination of the navigational waypoints

      Where to go next?
    - Path planning and navigation to the waypoints
    - Coordination of the actions (multi-robot team)



Courtesy of M. Kulich

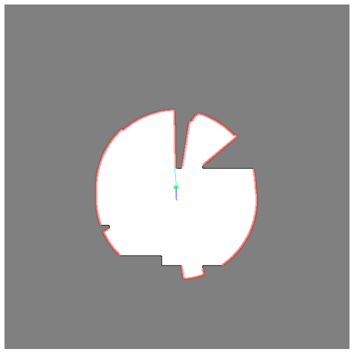
Mobile Robot Exploration

- Create a map of the environment
- **Frontier**-based approach

  Yamauchi (1997)
- Occupancy grid map

  Moravec and Elfes (1985)
- Laser scanner sensor
- Next-best-view approach

  Select the next robot goal



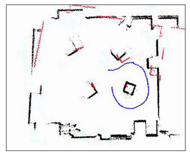
Performance metric:

Time to create the map of the whole environment

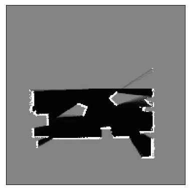
search and rescue mission

Environment Representation – Mapping and Occupancy Grid

- The robot uses its sensors to build a map of the environment
- The robot should be localized to integrate new sensor measurements into a globally consistent map
- **SLAM** – Simultaneous Localization and Mapping
  - The robot uses the map being built to localize itself
  - The map is primarily to help to localize the robot
  - The map is a "side product" of SLAM

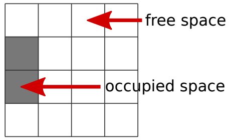


- **Grid map** – discretized world representation
  - A cell is **occupied** (an obstacle) or **free**
- **Occupancy grid map**
  - Each cell is a binary random variable modeling the occupancy of the cell

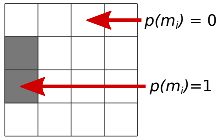


Occupancy Grid

- **Assumptions**
  - The area of a cell is either completely free or occupied
  - Cells (random variables) are indepedent of each other
  - The state is static



- A cell is a binary random variable modeling the occupancy of the cell
  - Cell m_i is occupied $p(m_i) = 1$
  - Cell m_i is not occupied $p(m_i) = 0$
  - Unknown $p(m_i) = 0.5$



- Probability distribution of the map m

$$p(m) = \Pi_i p(m_i)$$

- Estimation of map from sensor data $z_{1:t}$ and robot poses $x_{1:t}$

$$p(m|z_{1:t}, x_{1:t}) = \Pi_i p(m_i|z_{1:t}, x_{1:t})$$

Binary Bayes filter – Bayes rule and Markov process assumption

Binary Bayes Filter 1/2

- Sensor data $z_{1:t}$ and robot poses $x_{1:t}$
- Binary random variables are indepedent and states are static

$$p(m_i|z_{1:t}, x_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(z_t|m_i, z_{1:t-1}, x_{1:t})p(m_i|z_{1:t-1}, x_{1:t})}{p(z_t|z_{1:t-1}, x_{1:t})}$$

$$\overset{\text{Markov}}{=} \frac{p(z_t|m_i, x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(z_t|z_{1:t-1}, x_{1:t})}$$

$$p(z_t|m_i, x_t) = \frac{p(m_i, z_t, x_t)p(z_t, x_t)}{p(m_i|x_t)}$$

$$p(m_i, z_{1:t}, x_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(m_i|z_t, x_t)p(z_t|x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i|x_t)p(z_t|z_{1:t-1}, x_{1:t})}$$

$$\overset{\text{Markov}}{=} \frac{p(m_i|z_t, x_t)p(z_t|x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i)p(z_t|z_{1:t-1}, x_{1:t})}$$

Binary Bayes Filter 2/2

- Probability a cell is occupied

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(m_i|z_t, x_t)p(z_t|x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i)p(z_t|z_{1:t-1}, x_{1:t})}$$

- Probability a cell is not occupied

$$p(\neg m_i|z_{1:t}, x_{1:t}) = \frac{p(\neg m_i|z_t, x_t)p(z_t|x_t)p(\neg m_i|z_{1:t-1}, x_{1:t-1})}{p(\neg m_i)p(z_t|z_{1:t-1}, x_{1:t})}$$

- Ratio of the probabilities

$$\frac{p(m_i|z_{1:t}, x_{1:t})}{p(\neg m_i|z_{1:t}, x_{1:t})} = \frac{p(m_i|z_t, x_t)p(m_i|z_{1:t-1}, x_{1:t-1})p(\neg m_i)}{p(\neg m_i|z_t, x_t)p(\neg m_i|z_{1:t-1}, x_{1:t-1})p(m_i)}$$

$$= \frac{p(m_i|z_t, x_t)}{1 - p(m_i|z_t, x_t)} \frac{p(m_i, z_{1:t-1}, x_{1:t-1})}{1 - p(m_i|z_{1:t-1}, x_{1:t-1})} \frac{1 - p(m_i)}{p(m_i)}$$

sensor model z_t, recursive term, prior

Logs Odds Notation

- Log odds ratio is defined as

$$l(x) = \log \frac{p(x)}{1 - p(x)}$$

- and the probability $p(x)$ is

$$p(x) = 1 - \frac{1}{1 - e^{l(x)}}$$

- The product modeling the cell m_i based on $z_{1:t}$ and $x_{1:t}$

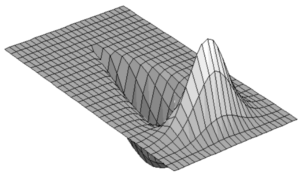
$$l(m_i|z_{1:t}, x_{1:t}) = \underbrace{l(m_i|z_t, x_t)}_{\text{inverse sensor model}} + \underbrace{l(m_i, |z_{1:t-1}, x_{1:t-1})}_{\text{recursive term}} - \underbrace{l(m_i)}_{\text{prior}}$$

Occupancy Mapping Algorithm

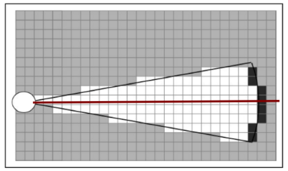
Algorithm 1: OccupancyGridMapping($\{l_{t-1,i}\}, x_t, z_t$)

foreach m_i *of the map m* **do**
  if m_i *in the perceptual field of* z_t **then**
    $l_{t,i} := l_{t-1,i} + \text{inv\_sensor\_model}(m_i, x_t, z_t) - l_0$;
  else
    $l_{t,i} := l_{t-1,i}$;
return $\{l_{t,i}\}$

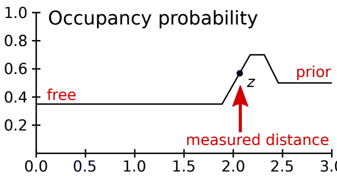
- Occupancy grid mapping developed by Moravec and Elfes in mid 80'ies for noisy sonars



Inverse sensor model for sonars range sensors



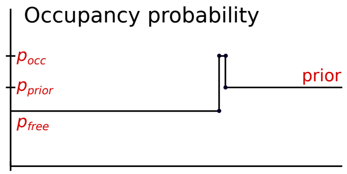
Field of view of the sonar range sensor



Occupancy value depending on the measured distance

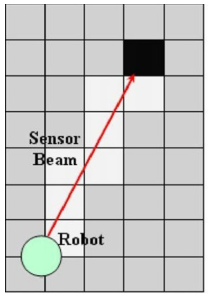
Model for Laser Sensor

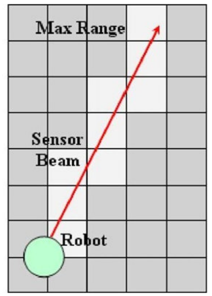
- The model is "sharp" with a precise detection of the obstacle
- For the range measurement d_i, update the grid cells along a sensor beam



Algorithm 2: Update map for $\mathcal{L} = (d_1, \dots, d_n)$

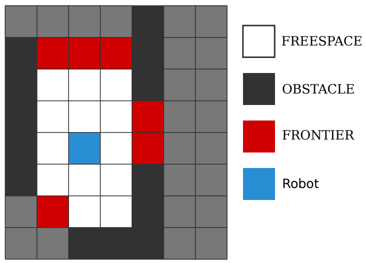
foreach $d_i \in \mathcal{L}$ **do**
  foreach *cell* m_i *raycasted towards* $\min(d_i, range)$ **do**
    $p := grid(m_i) p_{free}$;
    $grid(m_i) := p/2p - p_{free} - grid(m_i) + 1$;
  $m_d :=$ cell at d_i;
  if *obstacle detected at* m_d **then**
    $p := grid(m_d) p_{occ}$;
    $grid(m_i) := p/2p - p_{occ} - grid(m_i) + 1$
  else
    $p := grid(m_d) p_{free}$;
    $grid(m_i) := p/2p - p_{free} - grid(m_i) + 1$





Frontier-based Exploration

- The basic idea of the **frontier** based exploration is navigation of the mobile robot towards unknown regions
  Yamauchi (1997)
- **Frontier** – a border of the known and unknown regions of the environment
- Based on the probability of individual cells in the occupancy grid, cells are classified into:
  - FREESPACE – $p(m_i) < 0.5$
  - OBSTACLE – $p(m_i) > 0.5$
  - UNKNOWN – $p(m_i) = 0.5$
- **Frontier cell** is a FREESPACE cell that is incident with an UNKNOWN cell
- Frontier cells as the navigation waypoints have to be reachable, e.g., after obstacle growing



Use grid-based path planning

Frontier-based Exploration Strategy

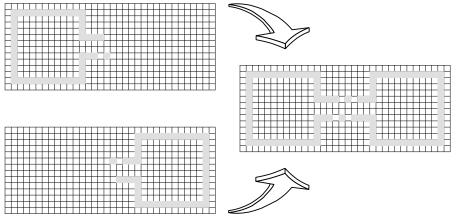
Algorithm 3: Frontier-based Exploration

map := init(*robot*, *scan*);
while *there are some reachable frontiers* **do**
  Update occupancy *map* using new sensor data and Bayes rule;
  $\mathcal{M}$:= Created grid map from *map* using thresholding;
  $\mathcal{M}$:= Grow obstacle according to the dimension of the robot;
  $\mathcal{F}$:= Determnine frontier cells from $\mathcal{M}$;
  $\mathcal{F}$:= Filter out unreachable frontiers from $\mathcal{F}$;
  f := Select the closest frontier from $\mathcal{F}$, e.g. using shortest path;
  path := Plan a path from the current robot position to f;
  Navigate robot towards f along *path* (for a while);

Multi-Robot Exploration – Map Marge

- The individual maps can be merged in a similar way as integration of new sensor measurements

$$P(occ_{x,y}) = \frac{odds_{x,y}}{1 + odds_{x,y}},$$
$$odds_{x,y} = \prod_{i=1}^{n} odds^{i}_{x,y},$$
$$odds^{i}_{x,y} = \frac{P(occ^{i}_{x,y})}{1 - P(occ^{i}_{x,y})}.$$

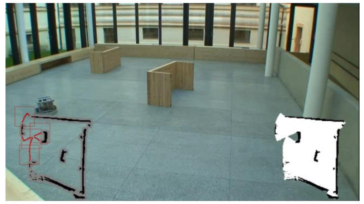


$P(occ^{i}_{x,y})$ is the probability that grid cell on the global coordinate is occupied in the map of the robot.

We need the same global reference frame (localization).

Multi-Robot Exploration – Overview

- We need to assign navigation waypoint to each robot, which can be formulated as the **task-allocation problem**
- Exploration can be considered as an **iterative procedure**



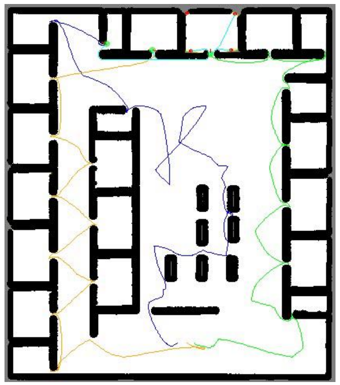
1. Initialize the occupancy grid Occ
2. $\mathcal{M} \leftarrow$ create_navigation_grid(Occ)
   cells of $\mathcal{M}$ have values {freespace, obstacle, unknown}
3. $\boldsymbol{F} \leftarrow$ detect_frontiers($\mathcal{M}$)
4. Goal candidates $\boldsymbol{G} \leftarrow$ generate($\boldsymbol{F}$)
5. **Assign next goals to each robot** $r \in \boldsymbol{R}$, $(\langle r_1, g_{r_1}\rangle, \ldots, \langle r_m, g_{r_m}\rangle) =$ assign($\boldsymbol{R}, \boldsymbol{G}, \mathcal{M}$)
6. Create a plan $\boldsymbol{P_i}$ for each pair $\langle r_i, g_{r_i}\rangle$
   consisting of simple operations
7. Perform each plan up to s_{max} operations
   At each step, update Occ using new sensor measurements
8. If $|G| == 0$ exploration finished, otherwise go to Step 2

- There are several parts of the exploration procedure where important decisions are made regarding the exploration performance, e.g.
  - How to determined goal candidates from the the frontiers?
  - How to plan a paths and assign the goals to the robots?
  - How to navigate the robots towards the goal?
  - When to replan?
  - etc.

Exploration Procedure – Decision-Making Parts

1. Initialize – set plans for m robots, $\mathcal{P} = (P_1, \ldots, P_m)$, $P_i = \emptyset$.
2. `Repeat`
   - 2.1 **Navigate robots** using the plans $\mathcal{P}$;
   - 2.2 Collect new measurements;
   - 2.3 Update the navigation map $\mathcal{M}$;

   `Until` **replanning condition is met.**
3. **Determine goal candidates $\boldsymbol{G}$** from $\mathcal{M}$.
4. `If` $|\boldsymbol{G}| > 0$ **assign goals to the robots**
   - $(\langle r_1, g_{r_1}\rangle, \ldots, \langle r_m, g_{r_m}\rangle)$=assign($\boldsymbol{R}, \boldsymbol{G}, \mathcal{M}$), $r_i \in \boldsymbol{R}, g_{r_i} \in \boldsymbol{G}$;
   - **Plan paths** to the assigned goals $\mathcal{P} =$ plan($\langle r_1, g_{r_1}\rangle, \ldots, \langle r_m, g_{r_m}\rangle, \mathcal{M}$);
   - Go to Step 2.
5. Stop all robots or navigate them to the depot

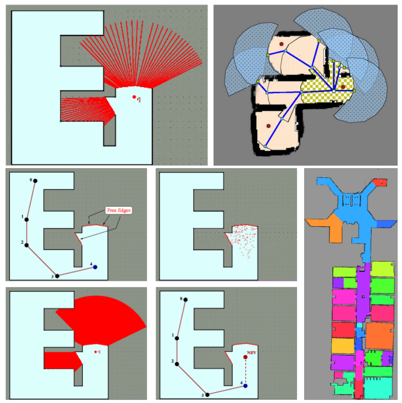


All reachable parts of the environment are explored.

Improvements of the basic Frontier-based Exploration

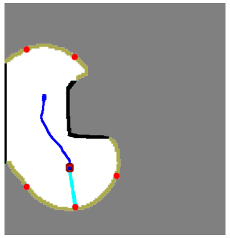
Several improvements have been proposed in the literature

- Introducing utility as a computation of expected covered area from a frontier
  González-Baños, Latombe (2002)
- Map segmentation for identification of rooms and exploration of the whole room by a single robot
  Holz, Basilico, Amigoni, Behnke (2010)
- Consider longer planning horizon (as a solution of the Traveling Salesman Problem (TSP))
  Zlot, Stentz (2006), Kulich, Faigl (2011,2012)
- Representatives of free edges
  Faigl, Kulich (2015)

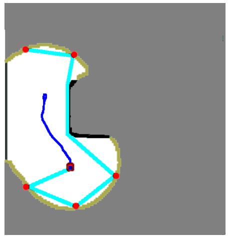


Distance Cost Variants

- **Simple robot–goal distance**
  - Evaluate all goals using the robot–goal distance
    A length of the path from the robot position to the goal candidate
  - Greedy goal selection – the closest one
  - Using frontier representatives improves the performance a bit



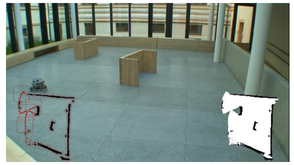
- **TSP distance cost**
  - Consider visitations of all goals
    Solve the associated traveling salesman problem (TSP)
  - A length of the tour visiting all goals
  - Use frontier representatives
  - the TSP distance cost improves performance about 10-30% without any further heuristics, e.g., expected coverage (utility)



Kulich, M., Faigl, J, Přeučil, L. (2011): On Distance Utility in the Exploration Task. ICRA.

Multi-Robot Exploration Strategy

- A set of m robots at positions $\boldsymbol{R} = \{r_1, r_2, \ldots, r_m\}$
- At time t, let a set of n goal candidates be $\boldsymbol{G}(t) = \{g_1, \ldots, g_n\}$
  I.e, frontiers



- The exploration strategy (at the planning step t):

  Select a goal $g \in \boldsymbol{G}(t)$ for each robot $r \in \boldsymbol{R}$ that will minimize the required time to explore the environment.

The problem is formulated as the **task-allocation problem**

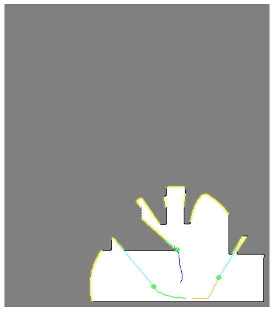
$$(\langle r_1, g_{r_1}\rangle, \ldots, \langle r_m, g_{r_m}\rangle) = \text{assign}(\boldsymbol{R}, \boldsymbol{G}(t), \mathcal{M}),$$

where $\mathcal{M}$ is the current map

Multi-Robot Exploration – Problem Definition

A problem of creating a grid map of the unknown environment by a set of m robots $\boldsymbol{R} = \{r_1, r_2, \ldots, r_m\}$.

Exploration is an iterative procedure:



1. Collect new sensor measurements
2. Determinate a set of goal candidates
   $$\boldsymbol{G}(t) = \{g_1, g_2, \ldots, g_n\}$$
   e.g., frontiers
3. At time step t, select next goal for each robot as the task-allocation problem
   $$(\langle r_1, g_{r_1}\rangle, \ldots, \langle r_m, g_{r_m}\rangle) = \text{assign}(\boldsymbol{R}, \boldsymbol{G}(t), \mathcal{M}(t))$$
   using the distance cost function
4. Navigate robots towards goal
5. If $|\boldsymbol{G}(t)| > 0$ go to Step 1; otherwise terminate

Goal Assignment Strategies – Task Allocation Algorithms

1. **Greedy Assignment**
   Yamauchi B, Robotics and Autonomous Systems 29, 1999
   - Randomized greedy selection of the closest goal candidate
2. **Iterative Assignment**
   Werger B, Mataric M, Distributed Autonomous Robotic Systems 4, 2001
   - Centralized variant of the broadcast of local eligibility algorithm (BLE)
3. **Hungarian Assignment**
   - Optimal solution of the task-allocation problem for assignment of n goals and m robots in $O(n^3)$
   Stachniss C, C implementation of the Hungarian method, 2004
4. **Multiple Traveling Salesman Problem – MTSP Assignment**
   - ⟨cluster–first, route–second⟩, the TSP distance cost
   Faigl et al. 2012

MTSP-based Task-Allocation Approach

- Consider the task-allocation problem as the **Multiple Traveling Salesman Problem (MTSP)**
- MTSP heuristic $\langle$*cluster–first, route–second*$\rangle$
  1. Cluster the goal candidates $\boldsymbol{G}$ to m clusters
     $$\boldsymbol{C} = \{C_1, \dots, C_m\}, C_i \subseteq \boldsymbol{G}$$
     using K-means
  2. For each robot $r_i \in \boldsymbol{R}, i \in \{1, \dots m\}$ select the next goal g_i from C_i using the TSP distance cost
     Kulich et at., ICRA (2011)
     - Solve the TSP on the set $C_i \cup \{r_i\}$
       the tour starts at r_i
     - The next robot goal g_i is the first goal of the found TSP tour

Faigl, J., Kulich, M., Přeučil, L. (2012): Goal Assignment using Distance Cost in Multi-Robot Exploration . IROS.

Statistical Evaluation of the Exploration Strategies

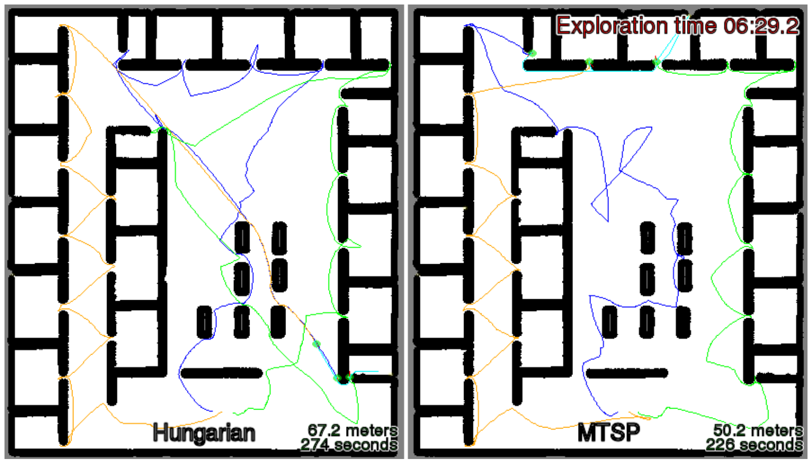
- Evaluation for the number of robots m and sensor range ρ

ρ	m	Iterative vs Greedy	Hungarian vs Iterative	MTSP vs Hungarian
3.0	3	+	=	+
3.0	5	+	=	+
3.0	7	+	=	+
3.0	10	+	+	–
4.0	3	+	=	+
4.0	5	+	=	=
4.0	7	+	=	+
4.0	10	+	+	–
5.0	3	+	=	+
5.0	5	+	=	+
5.0	7	+	=	+
5.0	10	+	+	–

Total number of trials 14 280

Performance of the MTSP vs Hungarian Algorithm

- Replanning as quickly as possible; $m = 3, \rho = 3$ m



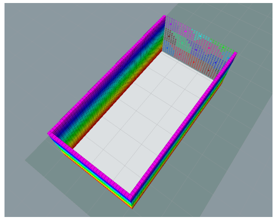
The MTSP assignment provides better performance

Information Theory in Robotic Information Gathering

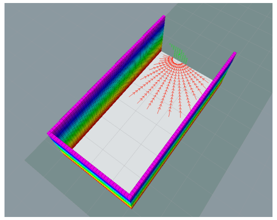
- Employ information theory in control policy for robotic exploration
  - **Entropy** – uncertainty of x: $H[x] = -\int p(x) \log p(x) dx$
  - **Conditional Entropy** – expected uncertainty of x after learning unknown z; $H[x|z]$
  - **Mutual information** – how much uncertainty of x will be reduced by learning z; $I_{MI}[x; z] = H[x] - H[x|z]$
- Control policy is a rule how to select the robot action that reduces the uncertainty of estimate by learning measurements:
  $$\text{argmax}_{a \in A} I_{MI}[x; z|a],$$
  where A is a set of possible actions, x is a future estimate, and z is future measurement
- Computation of the mutual information is computationally demanding
- Cauchy-Schwarz Quadratic Mutual Information (CSQMI) defined similarly to mutual information
  - A linear time approximations for CSQMI
    Charrow, B. et al., (2015): Information-theoretic mapping using Cauchy-Schwarz Quadratic Mutual Information. ICRA.
- Compute CSQMI as Cauchy-Schwarz divergence $I_{CS}[m; z]$ – raycast of the sensor beam and determine distribution over the range returns

Actions

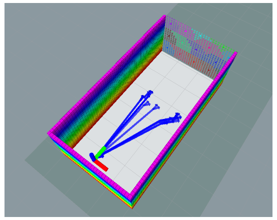
- Actions are shortest path to cover the frontiers



Detect and cluster frontiers

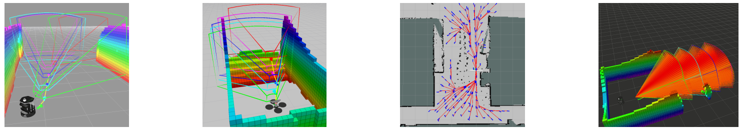


Sampled poses to cover a cluster

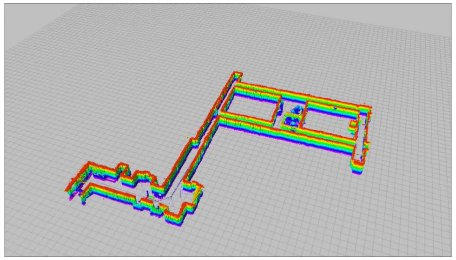


Paths to the sampled poses

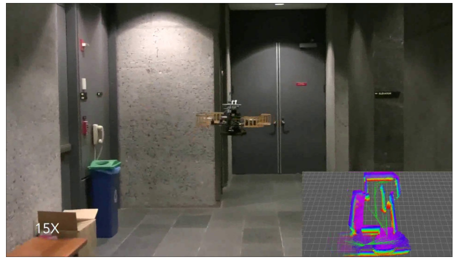
- Select an action (a path) that maximizes the rate of Cauchy-Schwarz Quadratic Mutual Information



Example of Autonomous Exploration using CSQMI



Ground vehicle



Aerial vehicle

- Planning with trajectory optimization – determine trajectory maximizing I_{CS}

Charrow, B. et al., (2015): Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. RSS.

Robotic Information Gathering

- Robotic information gathering can be considered as the **informative motion planning** problem to a determine trajectory $\mathcal{P}^*$ such that

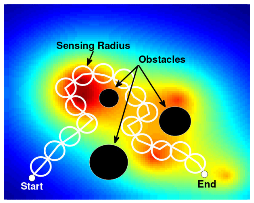
$$\mathcal{P}^* = \text{argmax}_{\mathcal{P}\in\Psi} I(\mathcal{P}), \text{ such that } c(\mathcal{P}) \leq B, \text{ where}$$

  - Ψ is the space of all possible robot trajectories,
  - $I(\mathcal{P})$ is the information gathered along the trajectory $\mathcal{P}$
  - $c(\mathcal{P})$ is the cost of $\mathcal{P}$ and B is the allowed budget

- Searching the space of all possible trajectories is complex and demanding problem
- A discretized problem can solved by combinatorial optimization techniques

Usually scale poorly with the size of the problem

- A trajectory is from a continuous domain



- **Sampling-based motion planning techniques** can employed for finding maximally informative trajectories

Hollinger, G., Sukhatme, G. (2014): Sampling-based robotic information gathering algorithms. IJRR.

Summary of the Lecture

Topics Discussed

- Robotic information gathering
- Robotic exploration of unknown environment
- Occupancy grid map
- Frontier based exploration
- Exploration procedure and decision-making
- TSP-based distance cost in frontier-based exploration
- Multi-robot exploration and task-allocation
- Mutual information and informative path planning *informative and motivational*

- Next: Randomized sampling-based motion planning methods