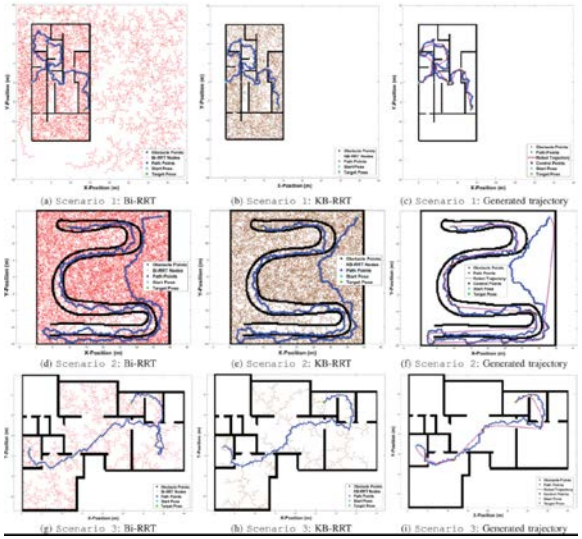
Recap: Sampling based Kinodynamic Planning

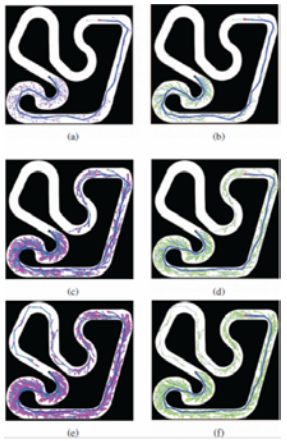
CS686

Seungwook Lee, 11/12/2019

- **Kinematic Constraints Based Bi-directional RRT: KB-RRT (ICRA 2019)**
  - For efficient computation are needed for real time operation of autonomous mobile robots
  - Bidirectional-RRT(Bi-RRT) based path planning algorithms + kinodynamic constraints
  - They proposed a kinematically constrained Bi-RRT (KB-RRT) algorithm
    - In order to restrict the number of nodes



- **Sampling-based Algorithms for optimal motion planning using closed-loop prediction (ICRA 2017)**
  - Leverages ideas from the RRT# algorithm
  - RRT# + closed-loop prediction = CL-RRT#



Occupancy Mapping using two different methods

CS686 Paper Presentation

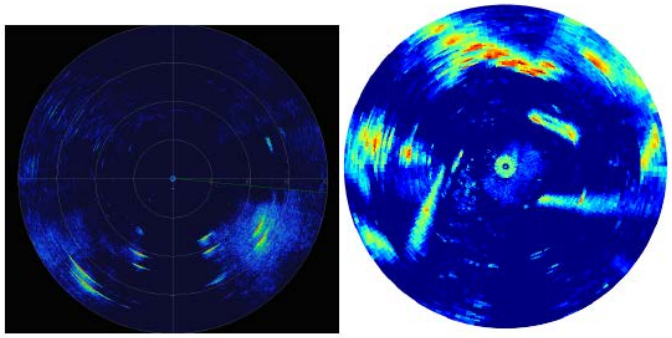
20195487
Wonho Song
송원호

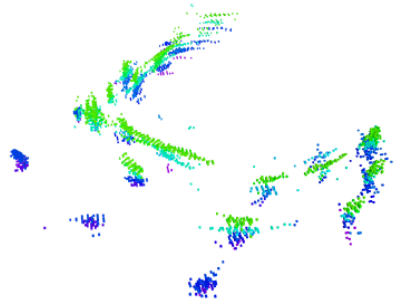
2019. 11. 14.

KAIST 한국과학기술원
Korea Advanced Institute of Science and Technology

Motivation

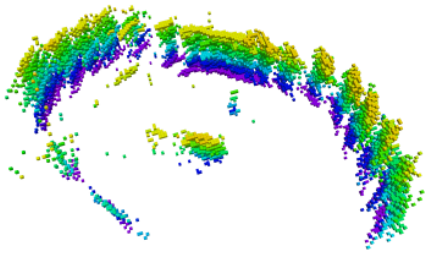
- **Applications of mobile robots warrant reliable planning and navigation, but sensors often give sparse and noisy readings. Additionally it should be real-time, taking place online**
  - Underwater navigation using sonar
    - Environmental disturbances
    - Low resolution
    - Multiple returns



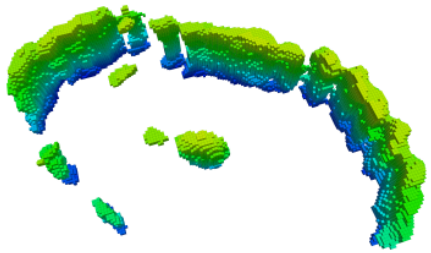


(a) Pier 84 OctoMap

(b) Pier 84 GP OctoMap



(a) USMMA OctoMap



(b) USMMA GP OctoMap

- Traditional occupancy grid mapping is limited by strong spatial independence assumptions

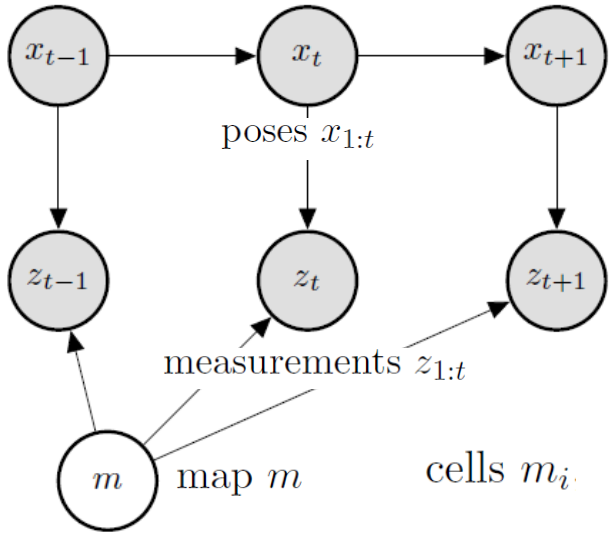
J. Wang, S. Bai, and B. Englot \Underwater Localization and 3D Mapping of Submerged Structures with a Single-Beam Scanning Sonar," Proceedings of the IEEE International Conference on Robotics and Automation, Accepted, To Appear, May 2017.

Background

- **Applications of mobile robots warrant reliable planning and navigation, but sensors often give sparse and noisy readings. Additionally it should be real-time, taking place online**
  - We can derive the recursive update as follows:

$$
\begin{aligned}
p(m_i|z_{1:t}, x_{1:t}) &= \frac{p(z_t|m_i, z_{1:t-1}, x_{1:t})p(m_i|z_{1:t-1}, x_{1:t})}{p(z_t|z_{1:t-1}, x_{1:t})} && \text{(Bayes)} \\
&= \frac{p(z_t|m_i, x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(z_t|z_{1:t-1}, x_{1:t})} && \text{(Markov)} \\
&= \frac{p(m_i|z_t, x_t)p(z_t|x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i)p(z_t|z_{1:t-1}, x_{1:t})} && \text{(Bayes)} \\
p(\neg m_i|z_{1:t}, x_{1:t}) &= \frac{p(\neg m_i|z_t, x_t)p(z_t|x_t)p(\neg m_i|z_{1:t-1}, x_{1:t-1})}{p(\neg m_i)p(z_t|z_{1:t-1}, x_{1:t})} \\
\frac{p(m_i|z_{1:t}, x_{1:t})}{p(\neg m_i|z_{1:t}, x_{1:t})} &= \frac{p(m_i|z_t, x_t)}{p(\neg m_i|z_t, x_t)} \frac{p(m_i|z_{1:t-1}, x_{1:t-1})}{p(\neg m_i|z_{1:t-1}, x_{1:t-1})} \frac{p(\neg m_i)}{p(m_i)}
\end{aligned}
$$

$$
\begin{aligned}
&\log \frac{p(m_i|z_t, x_t)}{1 - p(m_i|z_t, x_t)} + \log \frac{p(m_i|z_{1:t-1}, x_{1:t-1})}{1 - p(m_i|z_{1:t-1}, x_{1:t-1})} - \log \frac{p(m_i)}{1 - p(m_i)} \\
&\quad = l_{t,i} + l_{1:t-1,i} - l_0
\end{aligned}
$$



  - Why not just use Gaussian process occupancy maps?
    - Prediction requires inversion of a matrix the size of the training data; in O(N3)
    - Too slow for real-time mapping scenarios

Background

- **Paper 1 : Overlapping Hilbert Maps**
  - K. Doherty, J. Wang, and B. Englot, "Probabilistic map fusion for fast, incremental occupancy mapping with 3D Hilbert maps," Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1011-1018, 2016.

- **Paper 2: Gaussian Process Occupancy Maps**
  - J. Wang and B. Englot, "Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion," Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1003-1010, 2016.

Overlapping Hilbert Maps (ICRA 2016)

CS686

[Paper 1] Historically

- **Original Hilbert Maps**
  - Need some modifications on parameters

- Nonlinear logistic regression

$$p(y_* = 1|x_*, x, w) = \frac{1}{1 + \exp(w^T k(x_*, x))}$$

- Approximate radial basis function kernel (Nyström method)

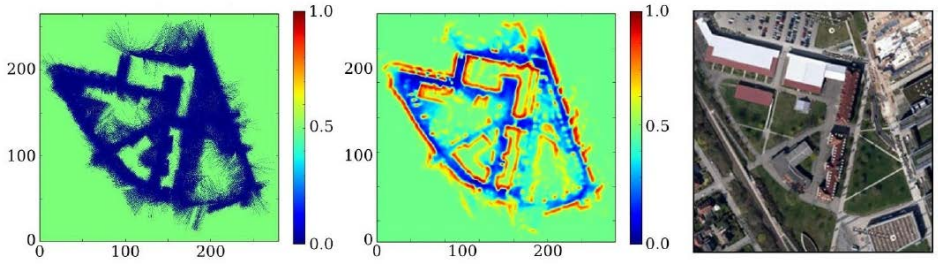
$$k(x, x') = \exp(-\gamma\|x - x'\|^2)$$

$$\phi_{\text{Nyström}}(x) = D^{-1/2}V^T(k(x, \hat{x}_1), \ldots, k(x, \hat{x}_m))$$

- Train iteratively with stochastic gradient descent: minimize

$$E(w, b) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)) + \alpha\|w\|_1$$

$$L(y_i, f(x_i)) = \log(\exp(-y_i(x_i^T w + b)) + 1)$$



Pros:

- Faster than GPOM (Linear time complexity, but big coefficient attached)
- Comparable accuracy

Cons:

- No uncertainty estimates
- Still intractable for real-time 3D mapping

Overlapping Hilbert Maps (ICRA 2016)

[Paper 1] Overall plan

- **Overall plan**

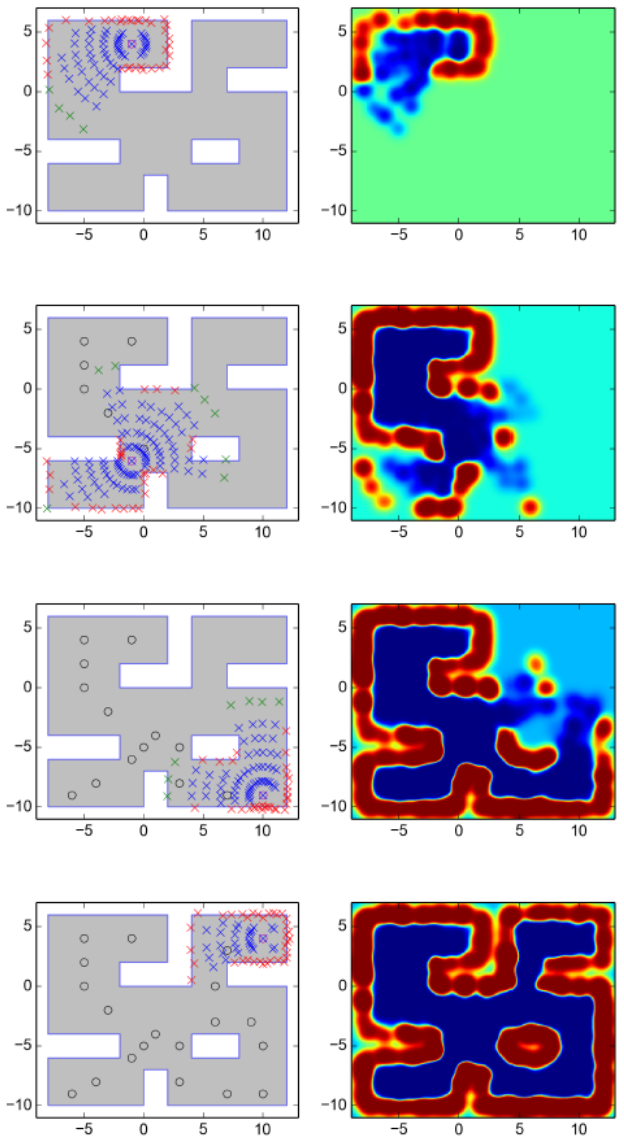
Probabilistic Local Map Fusion

↓

Combining Logistic Regression Classifiers

↓

Computational Results



[Paper 1] Step 1

Probabilistic Local Map Fusion

- Assume discretized grid map $m_i \in \mathcal{M}$
- Estimate the probability of occupancy of map cell m_i given the data by combining logistic regression outputs $p(m_i|\mathcal{D}_{1:t})$

$$p(m_i|\mathcal{D}_{1:t}) = \left[1 + \frac{1 - p(m_i|\mathcal{D}_{1:t-1})}{p(m_i|\mathcal{D}_{1:t-1})} \frac{1 - p(m_i|\mathcal{D}_t)}{p(m_i|\mathcal{D}_t)} \frac{p(m_i)}{1 - p(m_i)}\right]^{-1}$$

- Doesn't this assume grid cell independence?
- Spatial dependence accounted for by logistic regression classifier
- We don't *need* to discretize this; treat the logistic regression as spatial interpolation, combine predictions at query time

[Paper 1] Step 2

Combining Logistic Regression Classifiers

$$p(m_i|\mathcal{D}_1) = \frac{1}{1+\exp(w_1^T k(X_1, x_*))}$$

$$p(\neg m_i|\mathcal{D}_1) = 1 - p(m_i|\mathcal{D}) = \frac{\exp(w_1^T k(X_1, x_*))}{1+\exp(w_1^T k(X_1, x_*))}$$

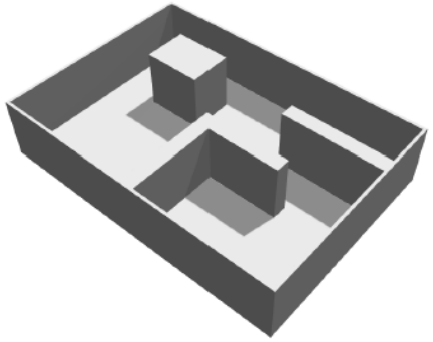
$$\begin{aligned} p(m_i|\mathcal{D}_{1:2}) &= \left[1 + \frac{1-p(m_i|\mathcal{D}_1)}{p(m_i|\mathcal{D}_1)} \frac{1-p(m_i|\mathcal{D}_2)}{p(m_i|\mathcal{D}_2)} \frac{p(m_i)}{1-p(m_i)}\right]^{-1} \\ &= \left[1 + \exp(w_1^T k(X_1, x_*)) \exp(w_2^T k(X_2, x_*))\right]^{-1} \\ &= \frac{1}{1+\exp(w_{new}^T k(X_{new}, x_*))} \end{aligned}$$

where $w_{new} = [w_1, w_2]$ and $X_{new} = [X_1; X_2]$.

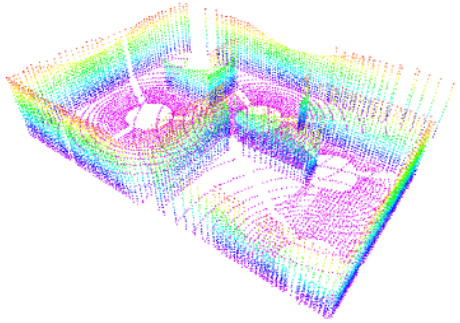
Overlapping Hilbert Maps (ICRA 2016)

[Paper 1] Result

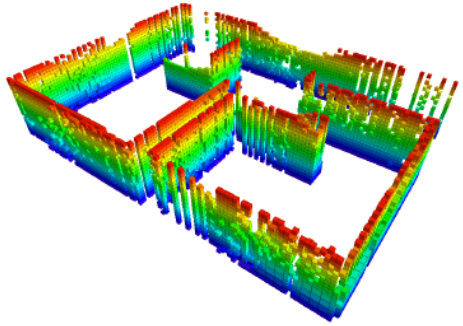
- **Computational Results**
  - Simulated datasets 1: "structured" in Gazebo



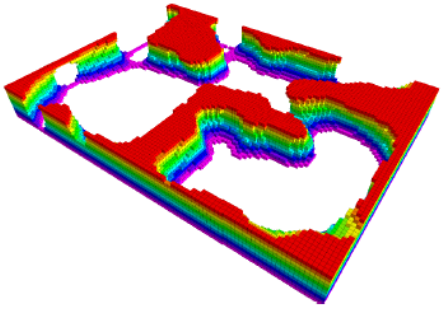
(a) The ground truth map



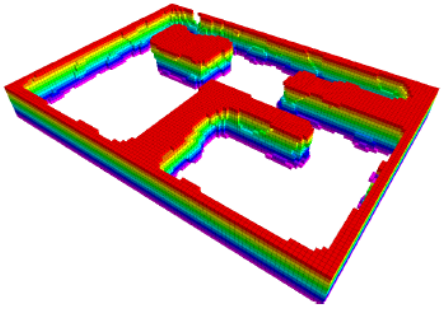
(b) The raw sensor data



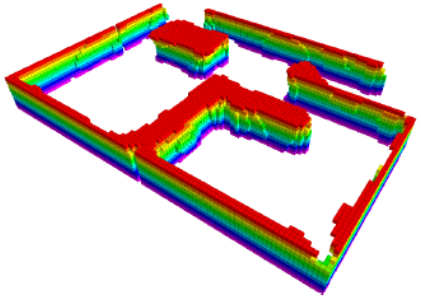
(c) The result of OctoMap



(d) The result of global Hilbert maps ($m = 100$)



(e) The result of global Hilbert maps ($m = 1000$)

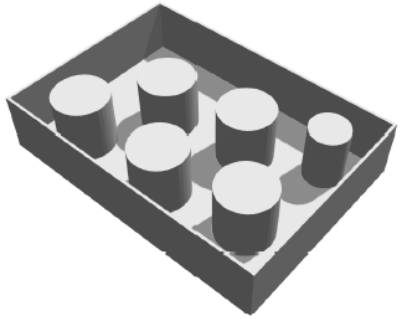


(f) The result after incrementally applying overlapping Hilbert maps

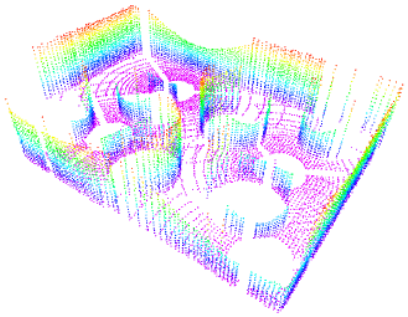
Overlapping Hilbert Maps (ICRA 2016)

[Paper 1] Result

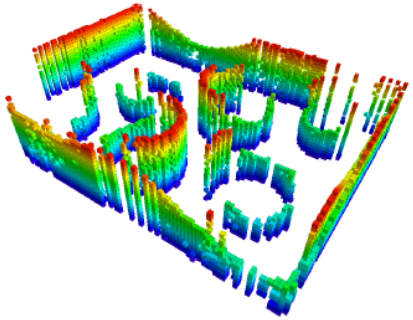
- **Computational Results**
  - Simulated datasets 2: "unstructured" in Gazebo



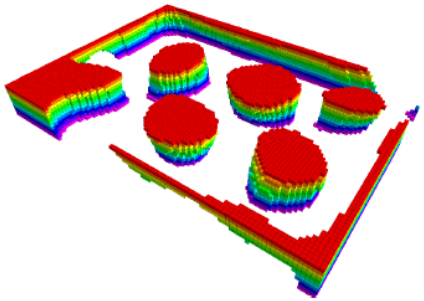
(a) The ground truth map



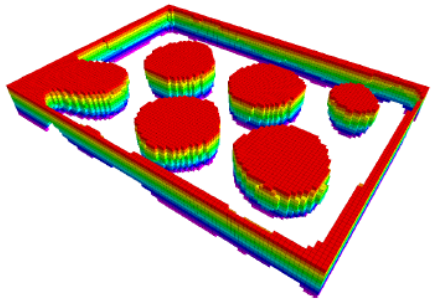
(b) The raw sensor data



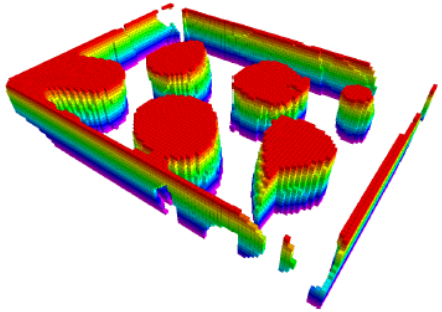
(c) The result of OctoMap



(d) The result of global Hilbert maps ($m = 100$)



(e) The result of global Hilbert maps ($m = 1000$)



(f) The result after incrementally applying overlapping Hilbert maps

[Paper 1] Result

- **Computational Results**
  - Computation times for mapping the three environments examined

Dataset	Scans	Points/Scan	Test Points/Scan	Method	Time/Scan (s)	Time (s)
Structured Simulation	12	3500	29892	Overlapping Hilbert Maps	1.89	22.68
				Global Hilbert Map ($m = 100$)	N/A	8.70
				Global Hilbert Map ($m = 1000$)	N/A	331.64
				OctoMap	0.02	0.2
Unstructured Simulation	12	3500	29892	Overlapping Hilbert Maps	1.83	21.96
				Global Hilbert Map ($m = 100$)	N/A	8.52
				Global Hilbert Map ($m = 1000$)	N/A	332.82
				OctoMap	0.01	0.14
Freiburg Corridor FR-079	66	4943	371170	Overlapping Hilbert Maps	14.6	963.6
				Global Hilbert Map ($m = 100$)	N/A	1656.95
				Global Hilbert Map ($m = 1000$)	N/A	11914.96
				OctoMap	0.1	6.7

Overlapping Hilbert Maps (ICRA 2016)

[Paper 1] Conclusion & future work

- **Conclusion**
  - Local approximation to Hilbert maps
  - Combine predictions from multiple classifiers
  - Faster, but not quite there (!!)
    - Implemented pretty naively in Python

- **What next?**
  - How can we do this in real-time?

[Paper 2] Overall plan

- **Overall plan**

SPATIAL PARTITIONING WITH TEST-DATA OCTREES

↓

NESTED UPDATES WITH BCMS

↓

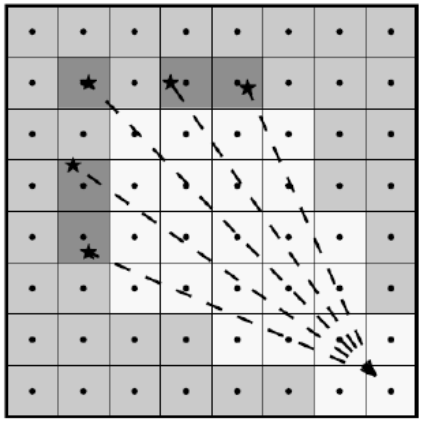
Computational Results

Algorithm 1 GPOctoMap-NBCM-P

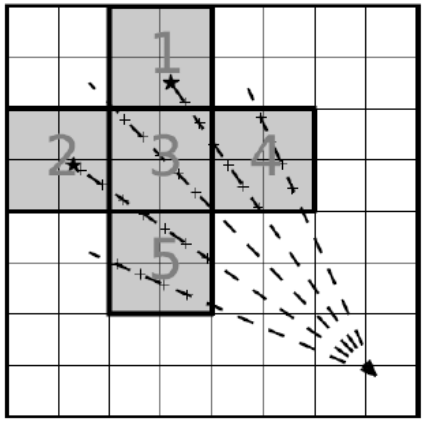
$\mathcal{M} \leftarrow \{\mathcal{B}_i = \emptyset\}, i = 1, \ldots, K$;
$\mathcal{G} \leftarrow \{g_i = \emptyset\}, i = 1, \ldots, K$;
while $!MappingComplete$ **do**
  $\mathcal{Z}_n \leftarrow SensorHits()$;
  $\mathcal{M}_n \leftarrow \emptyset$; # local blocks need to be updated
  for $\mathcal{B}_i \in \mathcal{M}$ **do**
    if $Intersect(\mathcal{B}_i, \mathcal{Z}_n)$ **then**
      $\mathcal{M}_n \leftarrow \mathcal{M}_n \cup \mathcal{B}_i$
    end if
  end for
  for $\mathcal{B}_i \in \mathcal{M}_n$ **do** # training in blocks
    if $IsEmptyOctree(\mathcal{B}_i)$ **then**
      $\mathcal{B}_i \leftarrow CreateOctreeInBlock(\mathcal{B}_i)$;
    end if
    $X, \mathrm{y} \leftarrow GetTrainingPoints(\mathcal{Z}_n)$;
    $g_i \leftarrow GPRegression(X, \mathrm{y})$;
  end for
  for $\mathcal{B}_i \in \mathcal{M}_n$ **do** # BCM fusion in extended blocks
    for $\mathcal{B}_e \in FindExtendedBlocks(\mathcal{B}_i)$ **do**
      $X_* \leftarrow GetOctreeLeafNodes(\mathcal{B}_e)$;
      $\mathrm{f}_*, \mathrm{var}_* \leftarrow GPPredict(g_e, X_*)$;
      $\mathcal{B}_i \leftarrow BCMUpdate(\mathcal{B}_e, \mathrm{f}_*, \mathrm{var}_*)$;
    end for
  end for
  for $\mathcal{B}_i \in \mathcal{M}_n$ **do**
    $PruneNodes(\mathcal{B}_i)$;
  end for
end while
$OccupancyProbs \leftarrow \emptyset$;
for $\mathcal{B}_i \in \mathcal{M}$ **do**
  if $!IsEmptyOctree(\mathcal{B}_i)$ **then**
    $\mathrm{x}_*, \mathrm{f}_*, \mathrm{var} \leftarrow GetOctreeLeafNodes(\mathcal{B}_i)$;
    $\{\mathrm{x}_*, \mathrm{p}\} \leftarrow BinaryClassification(\mathrm{f}_*, \mathrm{var}_*)$;
    $OccupancyProbs \leftarrow OccupancyProbs \cup \{\mathrm{x}_*, \mathrm{p}\}$;
  end if
end for
return $OccupancyProbs$;

[Paper 2] Contribution

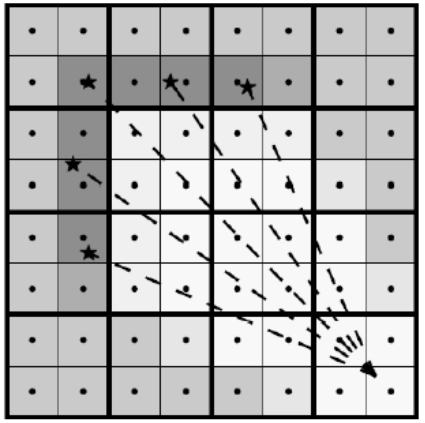
- **Test-data Octrees**
  - Consider only points in "extended block"
  - Quickly retrieve all relevant grid cells (and all relevant training data
  - Prune neighboring grid cells sharing the same state



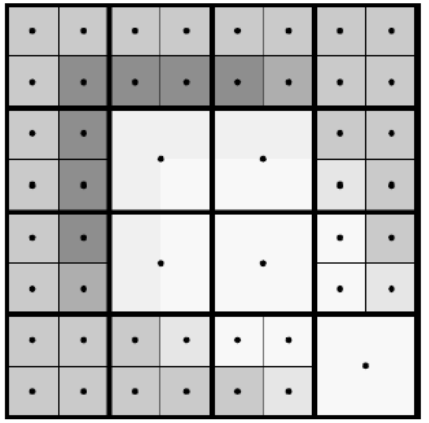
Standard occupancy grid mapping



predicting the occupancy probability



GP occupancy mapping with test -data octrees before pruning

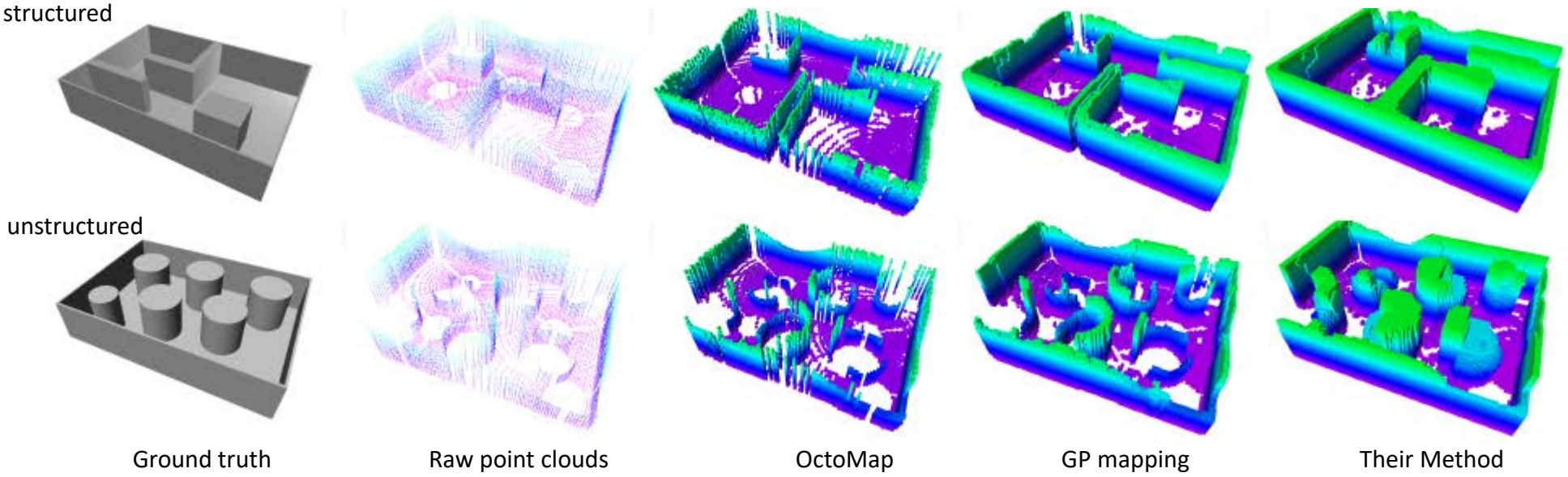


Final map after pruning

Gaussian Process Occupancy Maps (ICRA 2016) CS686

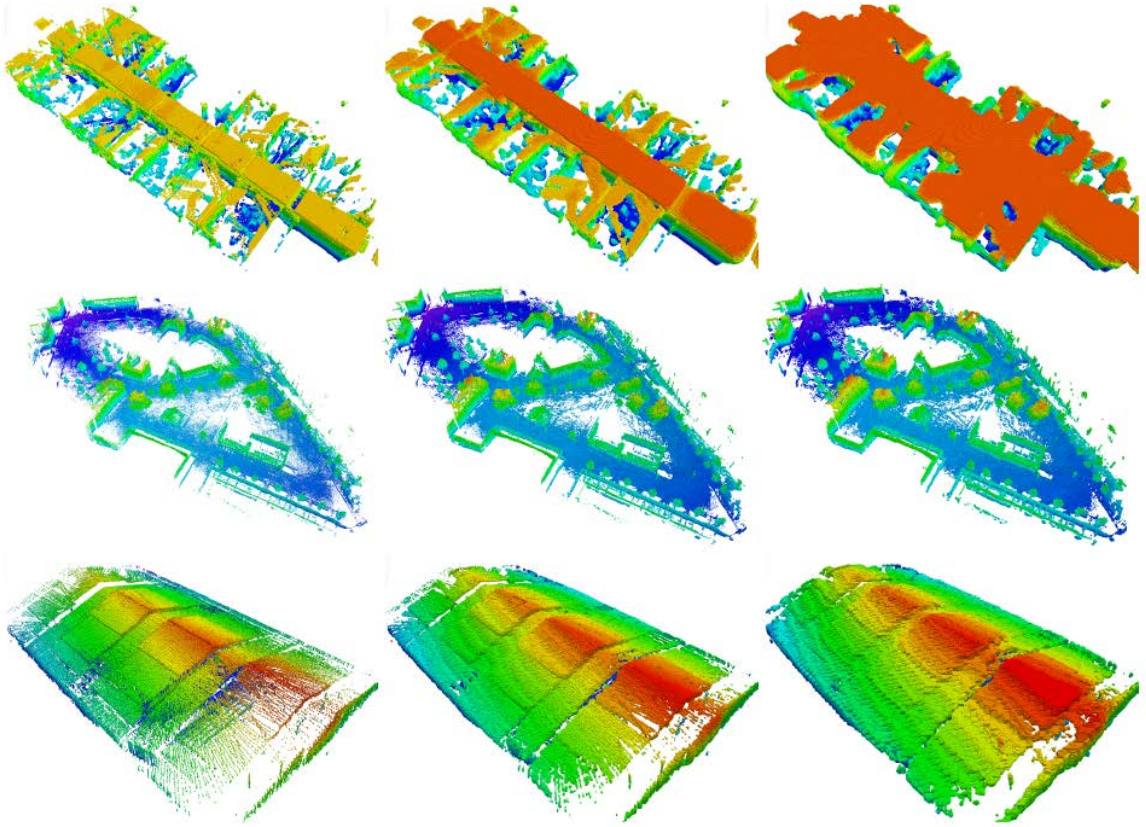
[Paper 2] Result

- **Computational Results**
  - Simulated datasets in Gazebo



[Paper 2] Result

- **Computational Results**
  - Other datasets, 0.1m resolution



Freiburg-079 corridor dataset
(43:7m x 182:m x 3:3m)

Freiburg campus dataset
(292m x 167m x 28m)

Spacebot Arena dataset
(72:3m x 71:4m x 12:9m)

OctoMap GP mapping Their Method

[Paper 2] Result

- **Computational Results**
  - Computation times for mapping the four environments examined

Dataset	Size (m^3)	Scans	Points/Scan	Sampled Points/Scan	Method	Time (s)	Time/Scan (s)
Sim Structured Data	$10.0 \times 7.0 \times 2.0$	12	3500	1506	BCM-NP	6.57	0.55
					NBCM-P	0.44	0.04
					NBCM-P+	0.37	0.03
					OctoMap	0.20	0.02
FR-079	$43.8 \times 18.2 \times 3.3$	66	89445	7601	BCM-NP	724.10	10.97
					NBCM-P	10.96	0.17
					NBCM-P+	9.57	0.15
					OctoMap	6.71	0.10
FR Campus	$292.0 \times 167.0 \times 28.0$	81	247817	76009	BCM-NP	N/A	N/A
					NBCM-P	329.95	4.07
					NBCM-P+	294.62	3.64
					OctoMap	242.88	3.00
Spacebot Arena	$72.3 \times 71.4 \times 12.9$	8	296734	64950	BCM-NP	1189.42	148.68
					NBCM-P	33.06	4.13
					NBCM-P+	27.35	3.42
					OctoMap	33.65	4.21

Summary

- **Paper 1 : Overlapping Hilbert Maps**
  - Substantial time improvement over Hilbert maps
  - Comparable accuracy to HM
  - Still not real-time ready
  - No uncertainty estimates

- **Paper 2: Bayesian Generalized Kernel Occupancy Maps**
  - Real-time viable
  - Comparable accuracy to GPOM
  - Provides uncertainty estimates
  - BCM is unstable when mapping the same region

Small Quiz

Summary

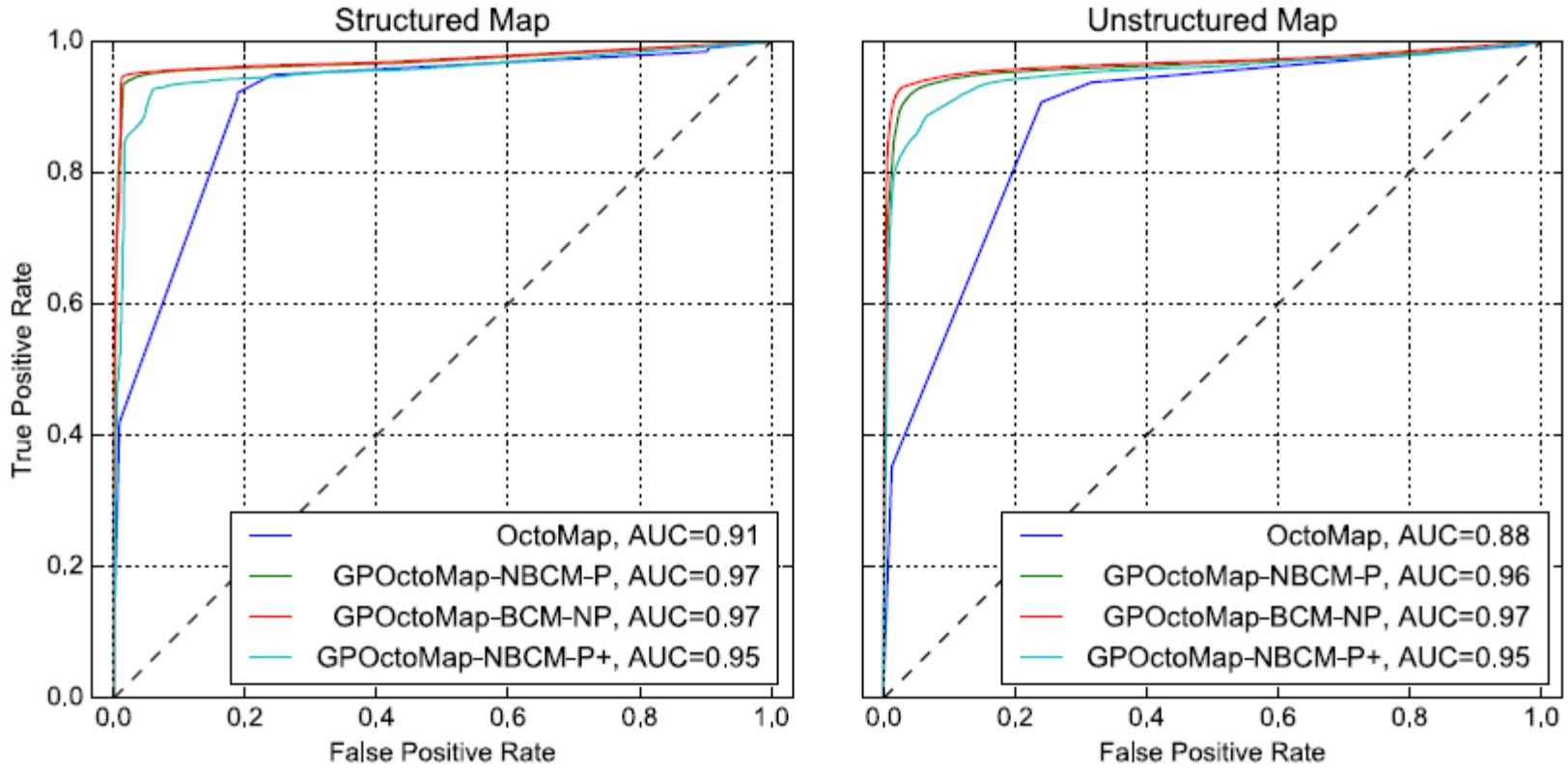
- Applications of mobile robots warrant reliable planning and navigation, but sensors often give sparse and noisy readings. OctoMap can handle them. (T/F)

- Gaussian Process Occupancy Maps are available online. (T/F)

Gaussian Process Occupancy Maps (ICRA 2016) CS686

[Paper 2] Result

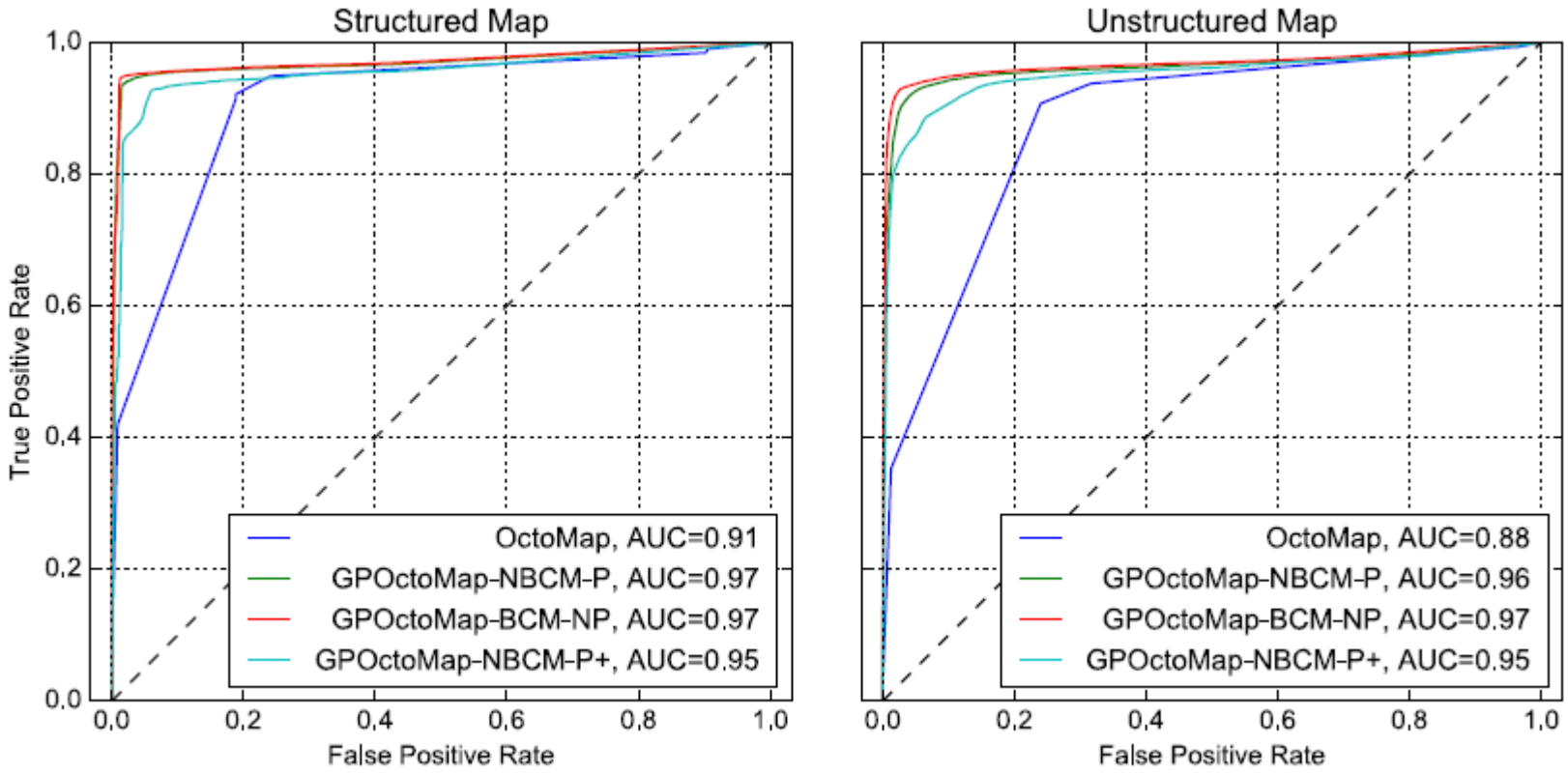
- Computational Results



Gaussian Process Occupancy Maps (ICRA 2016) CS686

[Paper 2] Result

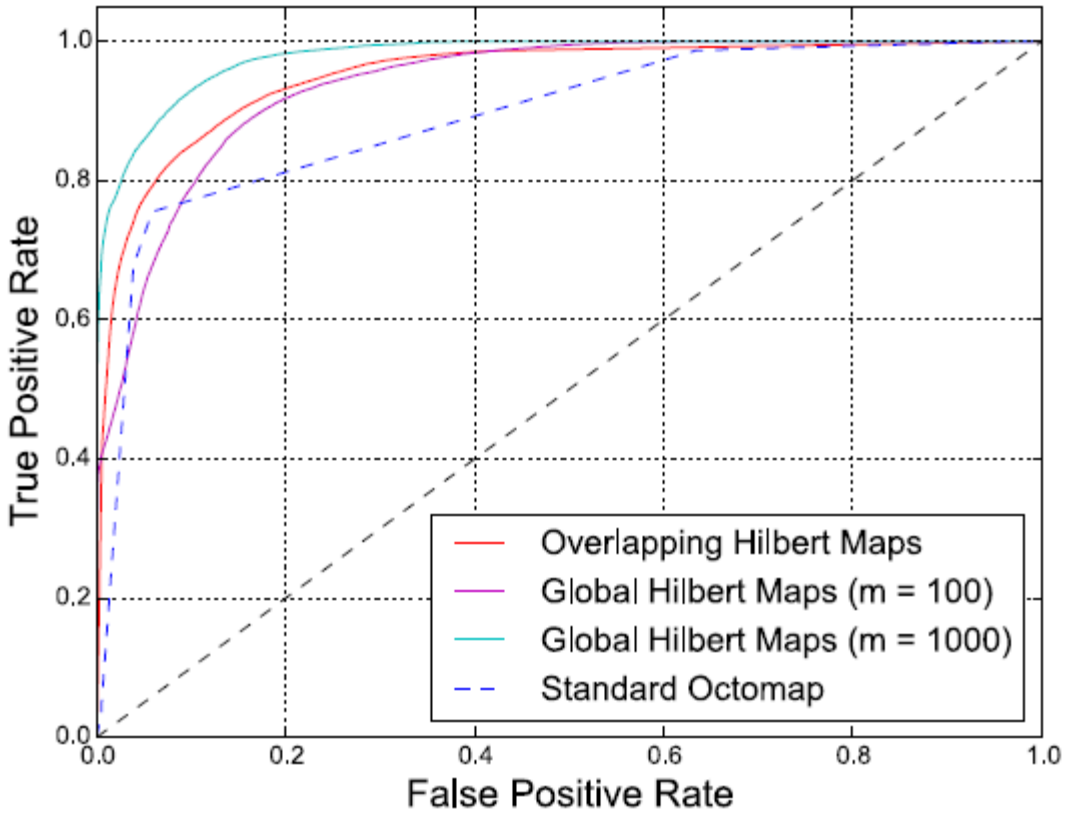
- **Computational Results**



Overlapping Hilbert Maps (ICRA 2016)

[Paper 1] Result

- **Result graph**
  - Computation



Overlapping Hilbert Maps (ICRA 2016)

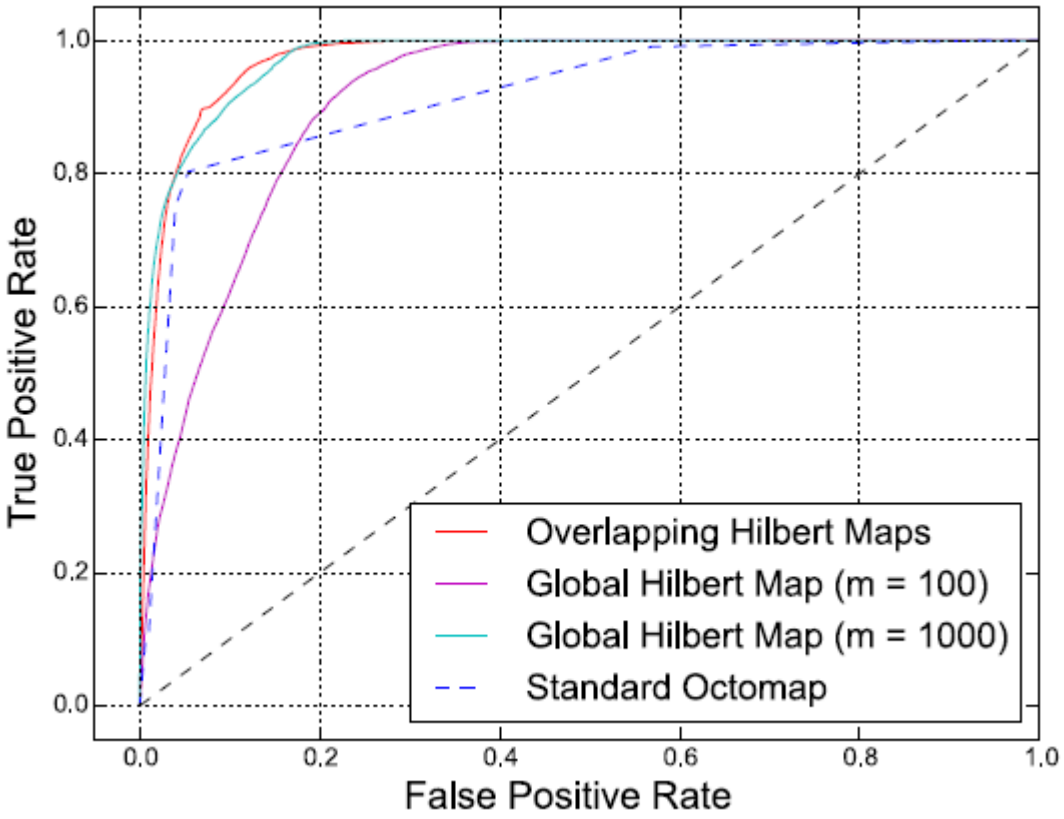
CS686

[Paper 1] Result

- **Result graph**
  - Computation

Method	AUC	Precision	Recall
OctoMap	0.92	0.320	0.097
Global Hilbert Map ($m = 100$)	0.91	0.792	0.108
Global Hilbert Map ($m = 1000$)	0.97	0.661	0.877
Overlapping Hilbert Maps	0.97	0.709	0.868

Method	AUC	Precision	Recall
OctoMap	0.89	0.418	0.042
Global Hilbert Map ($m = 100$)	0.94	0.999	0.248
Global Hilbert Map ($m = 1000$)	0.98	0.990	0.629
Overlapping Hilbert Maps	0.95	0.797	0.832



Source: "Local Path Planning of Driverless Car Navigation Based on Jump Point Search Method Under Urban Environment", *Future Internet 2017*

Overlapping Hilbert Maps (ICRA 2016)

CS686

[Paper 1] Result

- **Result**
  - Computation

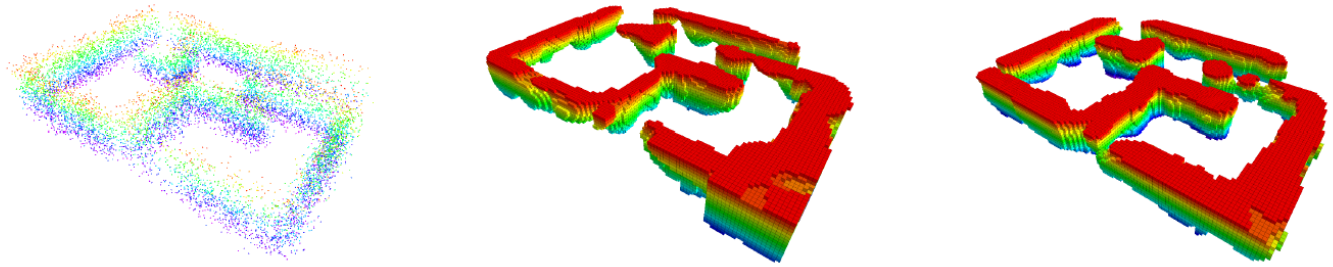


Fig. 8: Incrementally mapping the "structured" environment from Figure 2 when the range sensor is corrupted by noise. The first image (left) shows the raw sensor data. The second (center) shows the map produced using the Nyström method, as before. The rightmost image shows the same map produced using overlapping Hilbert maps with RKHS.

Source: "Local Path Planning of Driverless Car Navigation Based on Jump Point Search Method Under Urban Environment", *Future Internet 2017*