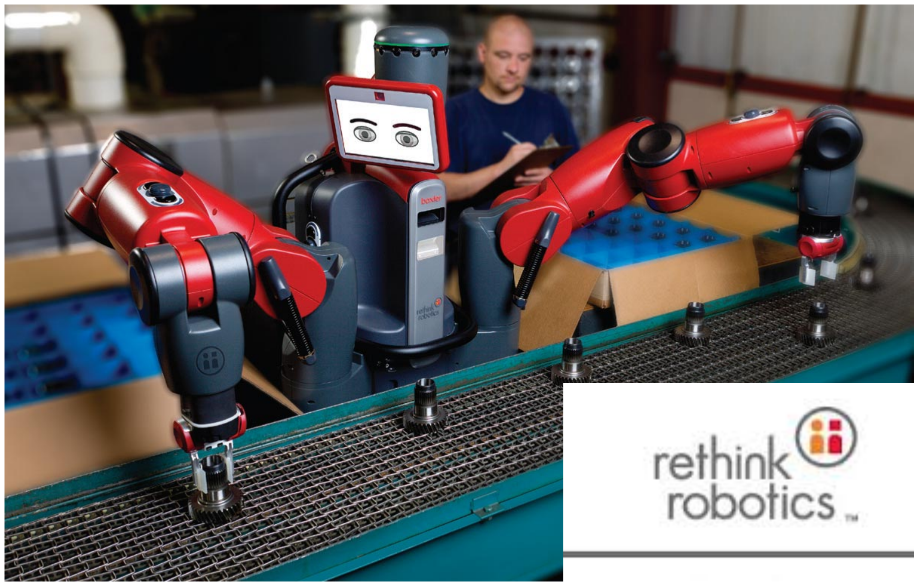
Real-Time GPU-Based Motion Planning for Task Execution

Chonhyon Park, **Jia Pan**, **Ming Lin**, **Dinesh Manocha**
University of North Carolina at Chapel Hill

Task Executions of Robots

- Advances in technology allow robots to perform complex tasks

<PR2: fetching a beer from the fridge>



<Baxter: $22k robot needs no programming>

Task Execution with Multiple Components

- A task is decomposed into many primitive subtasks

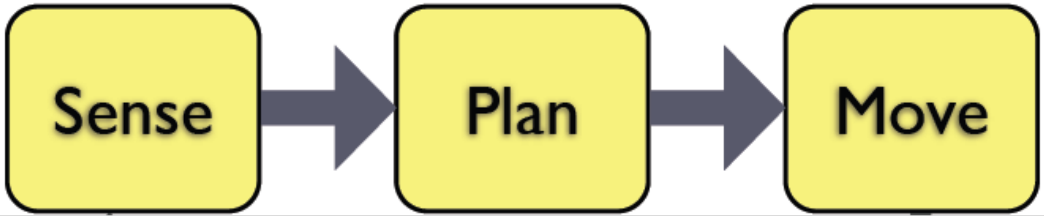
<PR2: taking out a beer from the fridge>
(From Willow Garage)

1. Move the body to the fridge.
2. Move the left arm to the handle.
3. Move the body to open the fridge door.
4. Move the body to in front of the fridge.
5. Move the left arm to hold the door.
6. Move the right arm to the beer.
7. Grasp the bottle.
8. Move the right arm to the basket.
9. Release the bottle.

Most of the subtasks are moving the robot to the next desired pose

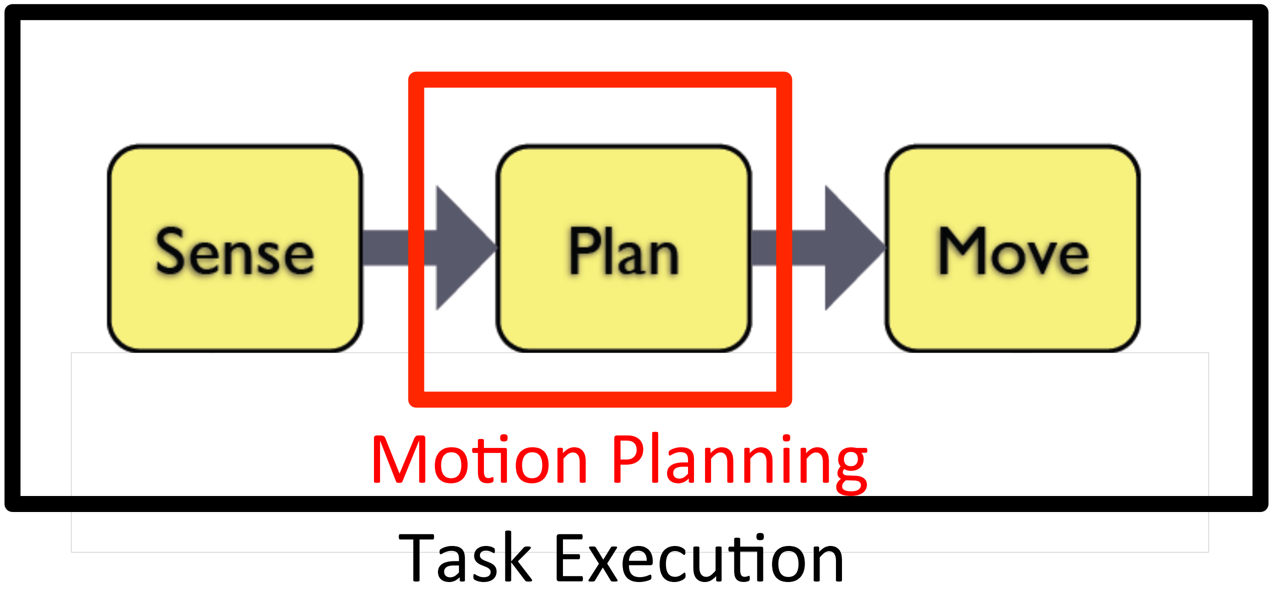
Subtask Execution

- ‘Move the body to the fridge’ subtask
  - Use sensors to recognize the objects and obstacles in the environment
  - Compute a collision-free path to the pose close to the fridge
  - Control motors to execute the computed motion



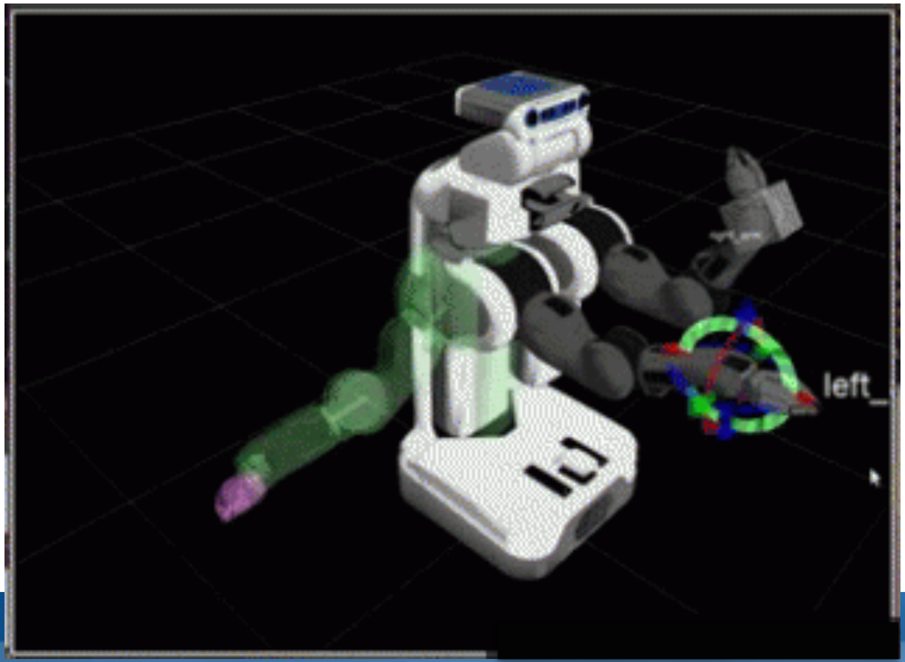
Task Execution vs. Motion Planning

- Sense – get the environment information
- Plan – compute motion to the desired state
- Move – execute the planned motion



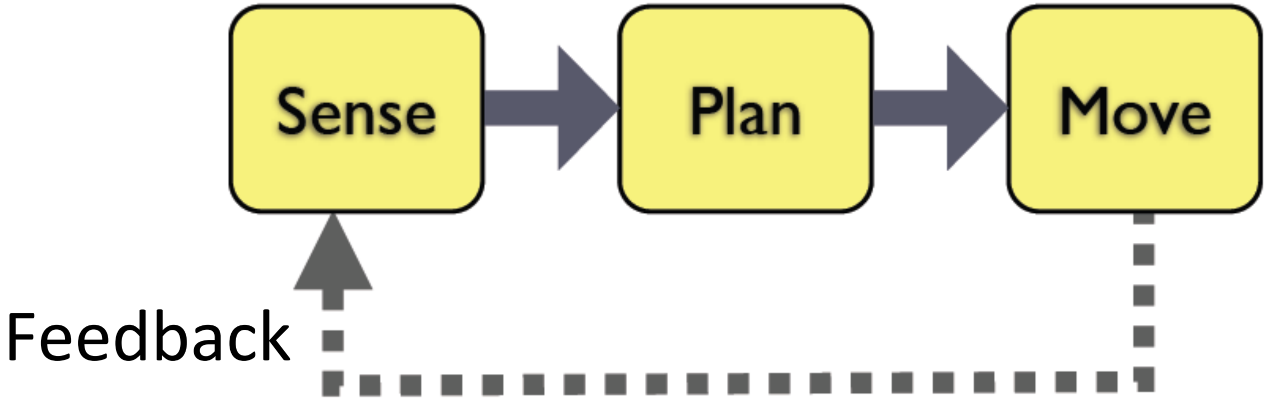
Motion Planning

- Find a continuous, collision-free motion trajectory from an initial pose to a goal pose
- An important problem in robotics, gaming, virtual prototyping, CAD/CAM, etc.



Why Real-time Motion Planner?

- Dynamic, uncertain environments or control uncertainty
- Complex task execution needs real-time feedback
- Combine with active sensing

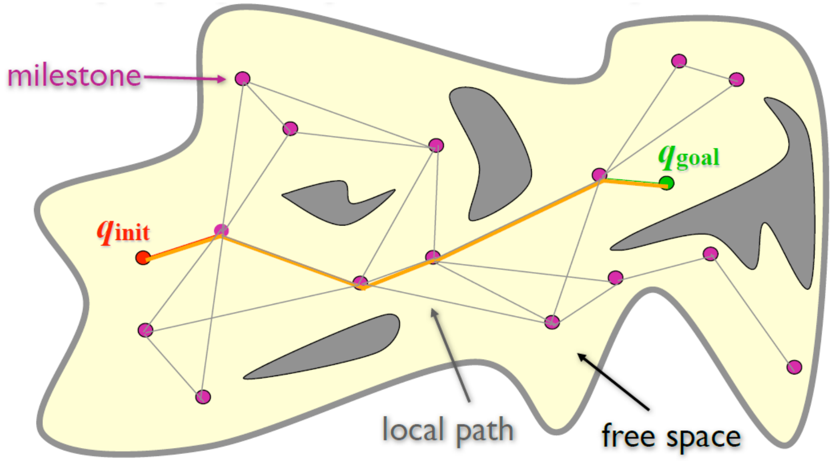


Outline

- **Sampling-based Motion Planning**
- Poisson-disk Sampling
- Parallel Poisson-RRT Motion Planning
- Experimental results
- Conclusion

Sampling-based Planning Algorithms

- PRM [Kavraki et al. 1996]
  - Construct complete roadmap
  - Graph search on roadmap to process query



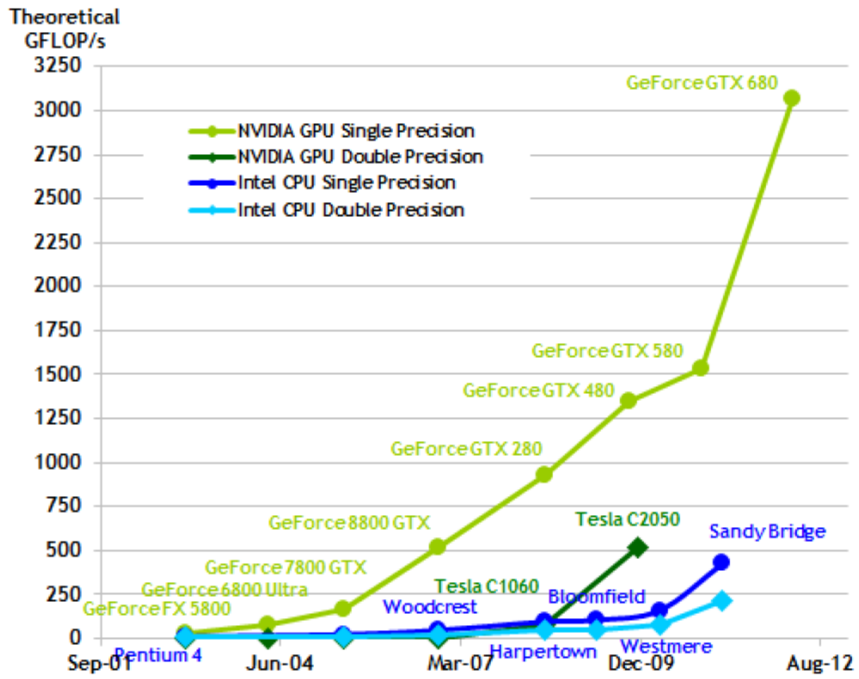
  - Efficient for multiple queries

Sampling-based Planning Algorithms

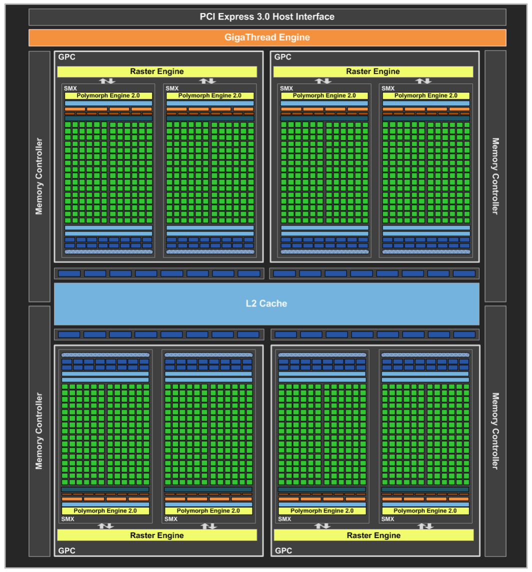
- RRT [Kuffner and LaValle 2000]
    - No construction phase
    - Expand tree on the fly
    - Efficient for single query

Parallel Planning Algorithms

- Parallel algorithms can benefit from the high computational power of GPUs



<Floating-Point Operations per Second for the CPU and GPU>

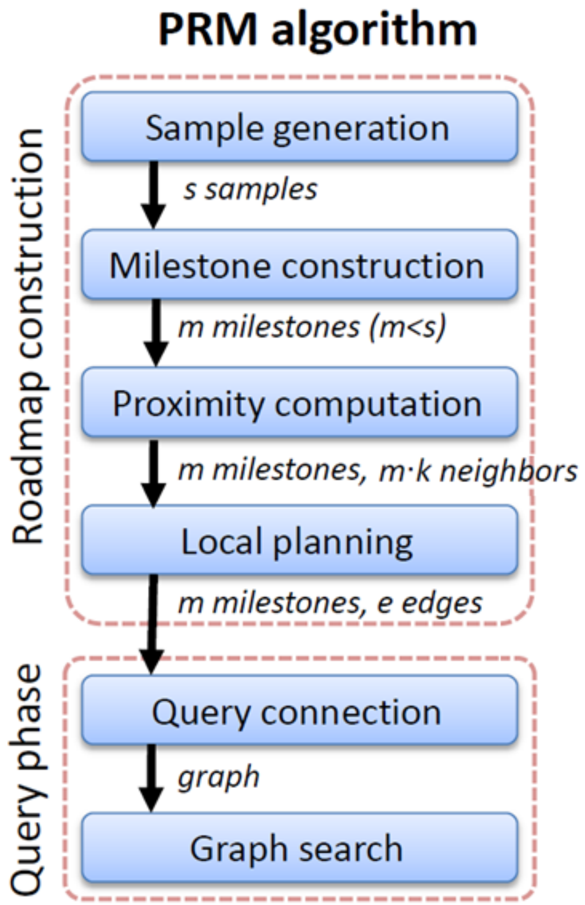


<Nvidia Kepler Architecture>

- 1536 CUDA cores

Parallel Planning Algorithms

- Parallel PRM algorithm on GPUs
  - G-Planner [Pan et al. 2010]
  - 10-100x speed-up from single-thread CPU algorithm
  - PRM is GPU-friendly
    - A large number of samples
    - Independent computations

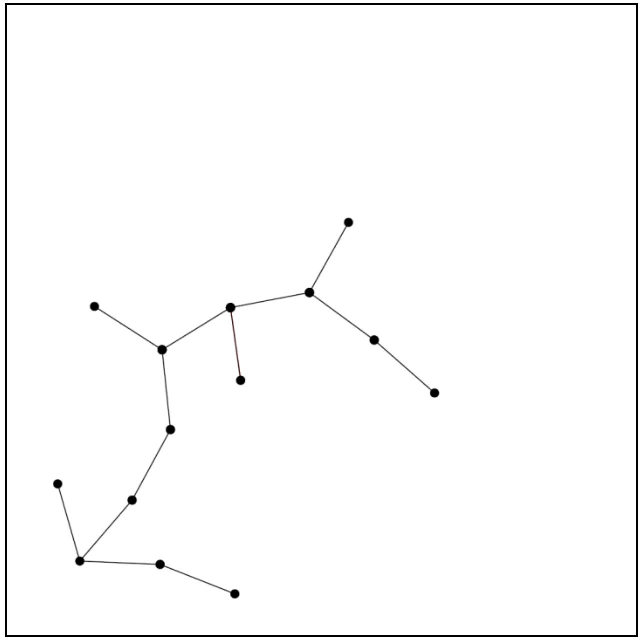


Parallel RRT Algorithms

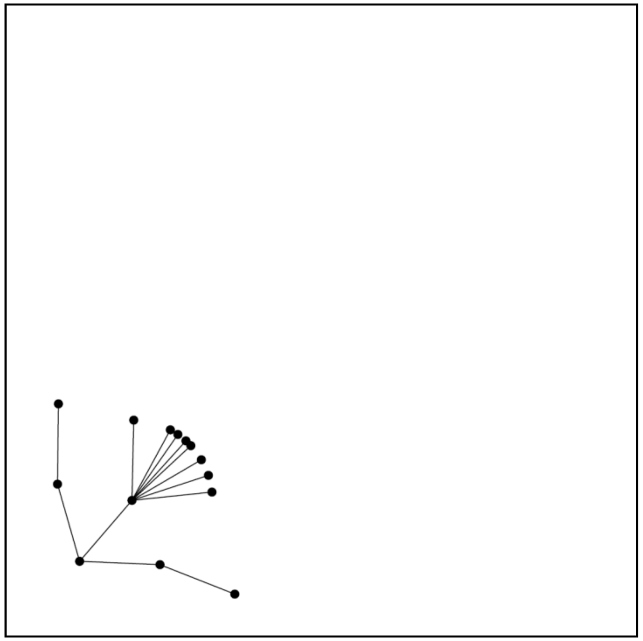
- Two categories of parallel RRTs [Carpin and Pagello 2002]
  - OR Parallel RRT
    - Multiple threads grow multiple trees
  - AND Parallel RRT
    - Multiple threads grow 1 tree

AND Parallel RRT Algorithm

- Serial RRT
  tree expansion



- AND Parallel RRT
  tree expansion



  - Generate nodes
    which are too close
  - Worse effect on GPU algorithm

Parallel RRT Algorithms

- RRT with GPU collision checking [Bialkowski et al. 2011]
  - Collision checking is the most time-consuming part
  - Multi-link robotic manipulator in 2D

- Parallel RRT with space partitioning [Ichnowski and Alterovitz 2012]
  - Partition the configuration space
  - 10 DOF Nao robot

Outline

- Sampling-based Motion Planning
- **Poisson-disk Sampling**
- Parallel Poisson-RRT Motion Planning
- Experimental results
- Conclusion

Poisson-Disk Sampling

- Widely used in computer graphics
  - Rendering, imaging, and animations

< A field of plants>

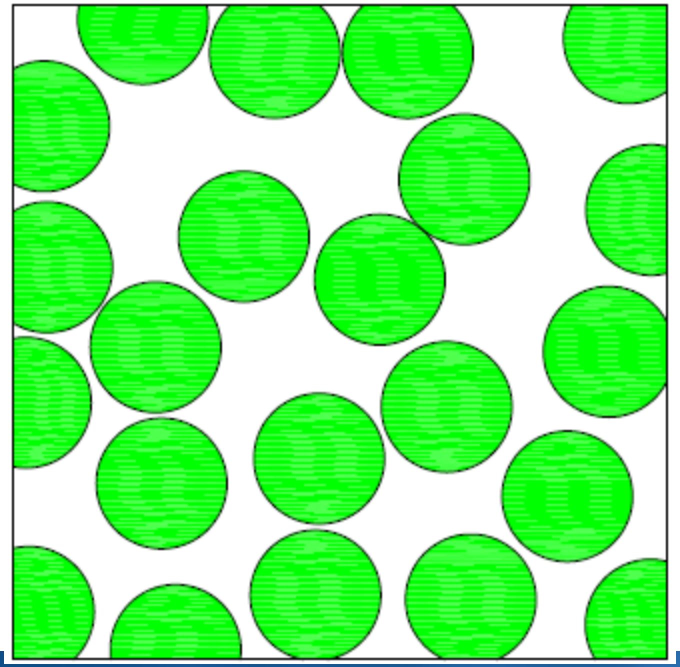
< Droplets on the glass>

Poisson-Disk Sampling

- Empty disk property

$$\forall x_i, x_j \in X, x_i \neq x_j : \|x_i - x_j\| \geq r$$

  - Samples are at least a minimum distance *r* apart from others

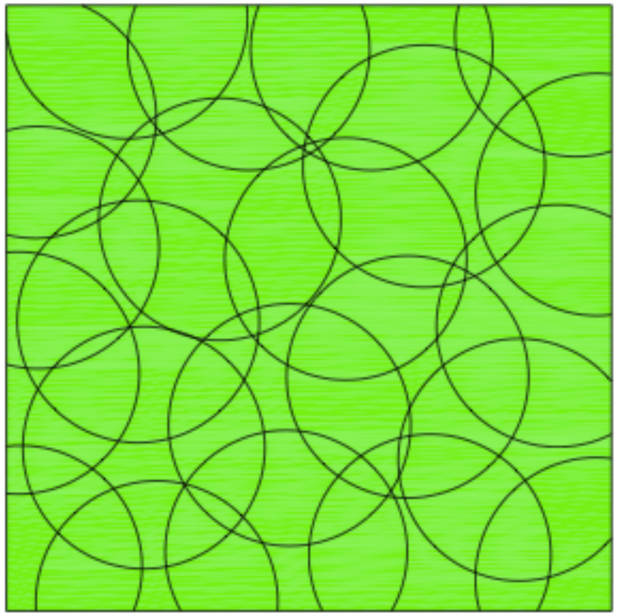


Maximal Poisson-Disk Sampling

- Maximal property

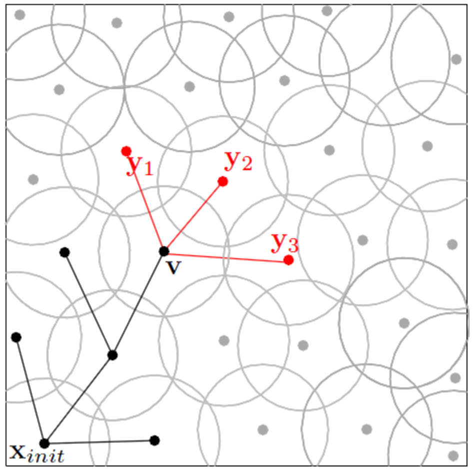
$$\forall x_i \in \mathcal{D}, \exists x_i \in X : \|x - x_i\| < r$$

  - No uncovered region in the domain by disks of radius *r*



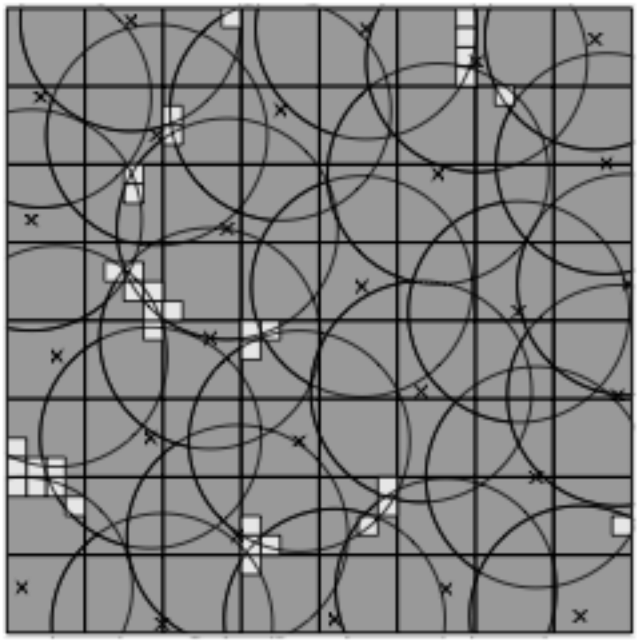
Our Approach

- Use maximal Poisson-disk samples in parallel RRT tree expansion
  - Empty-disk property
    - Ensure nodes are not too close
  - Maximal property
    - Ensure samples cover the entire space



Maximal Poisson-Disk Sampling

- 'State of the art' sampling algorithm
  [Ebeida et al. 2012]
- Generate samples in high-dimensions
- Not real-time performance



Outline

- Sampling-based Motion Planning
- Poisson-disk Sampling
- **Parallel Poisson-RRT Motion Planning**
- Experimental results
- Conclusion

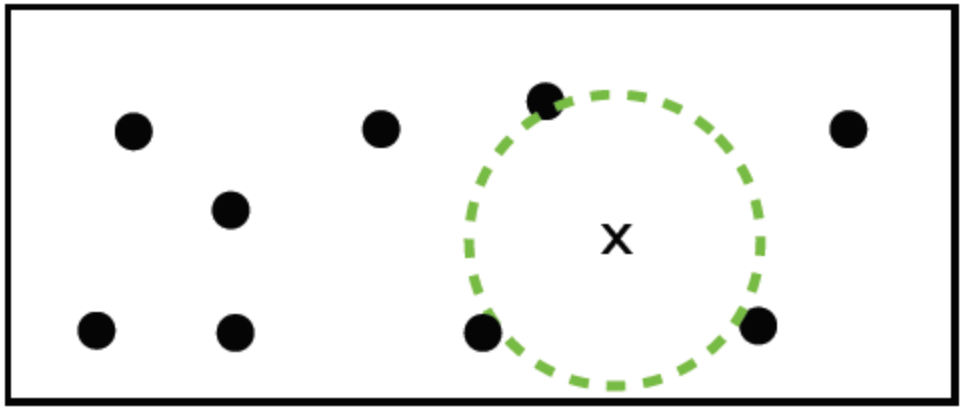
Poisson-RRT Planning Algorithm

- AND Parallel RRT
- Use GPU many-cores
- Use precomputed maximal Poisson-disk samples

Precomputed MPS for Planning

- Dispersion
  - The largest empty "ball" of unoccupied space

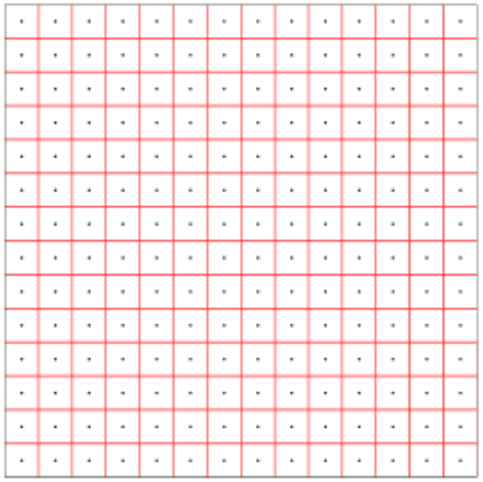
$$\delta(P,\rho) = \sup_{x \in X} \min_{p \in P} \rho(x,p)$$

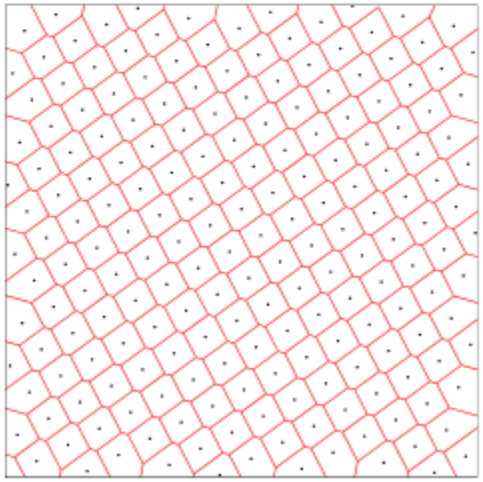


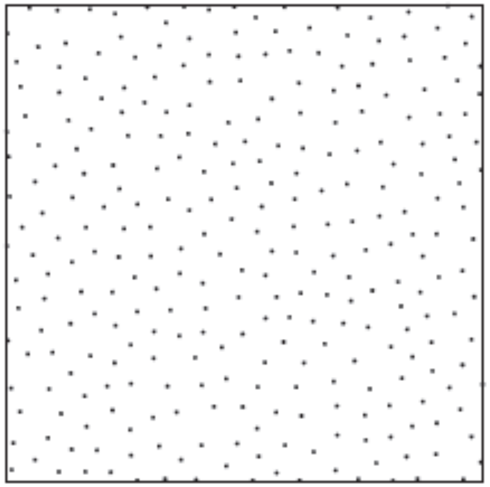
- Dispersion of MPS < r

Precomputed MPS for Planning

- Low-dispersion samplings


Sukharev grid


A nongrid lattice

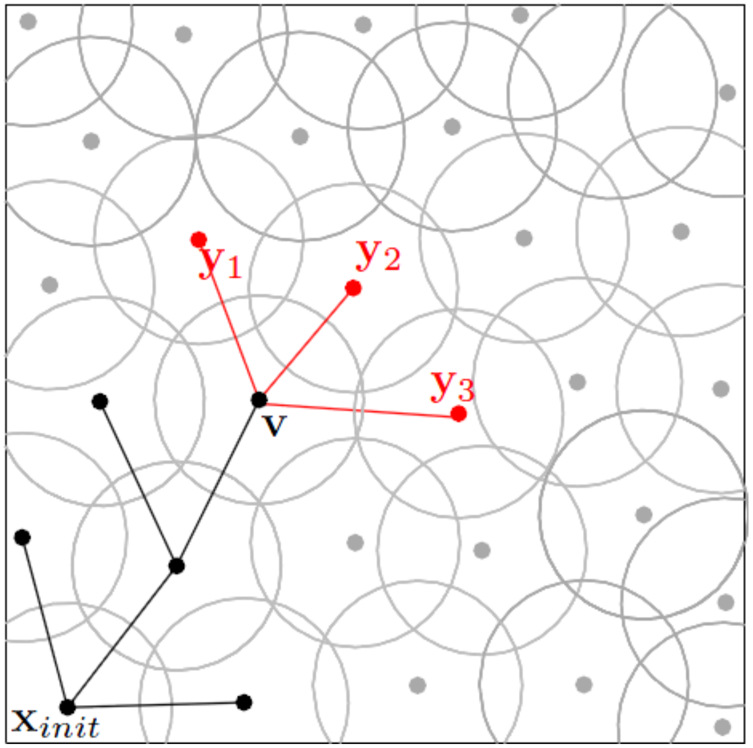

MPS

- Maximum Poisson-disk sampling
  - Low-discrepancy : Not aligned to certain axis
  - Less sensitive to the environment

Poisson-RRT Planning Algorithm

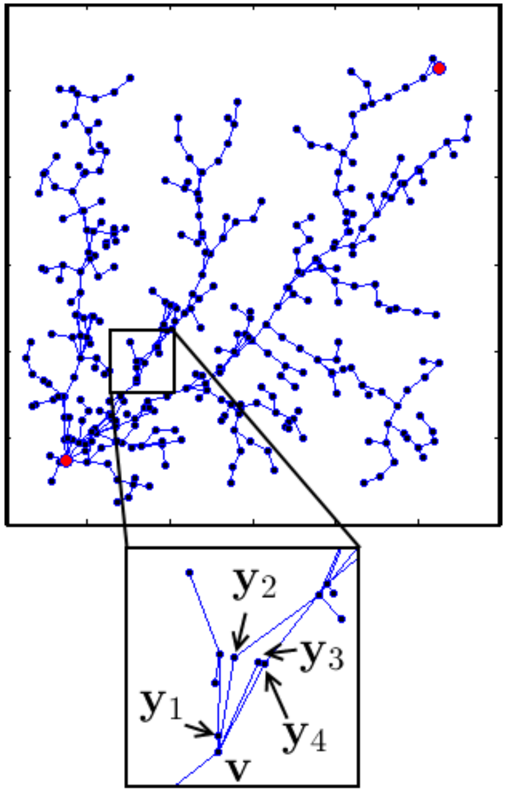
- Poisson-RRT Algorithm

0. Precomputed MPS
1. Sampling
2. Find neighbor
   Poisson-disk samples
3. Expand tree

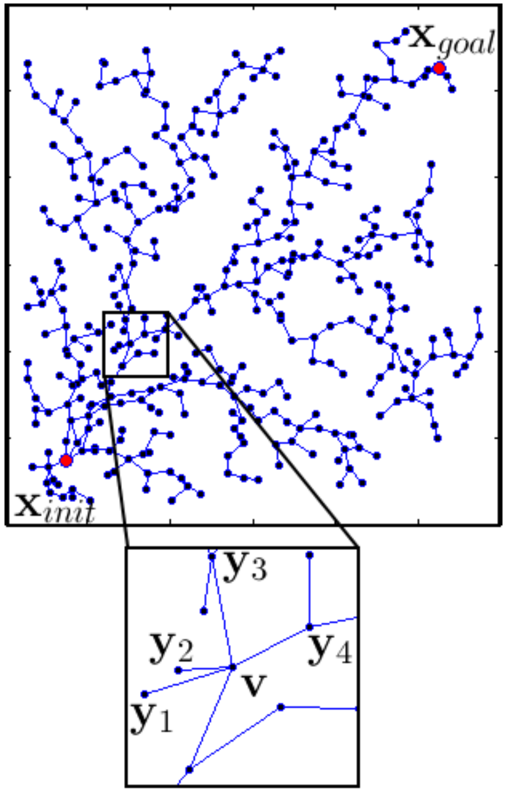


Parallel Poisson-RRT Algorithm

- AND Parallel RRT Tree



- Poisson-RRT Tree



No nodes which are too close to each other

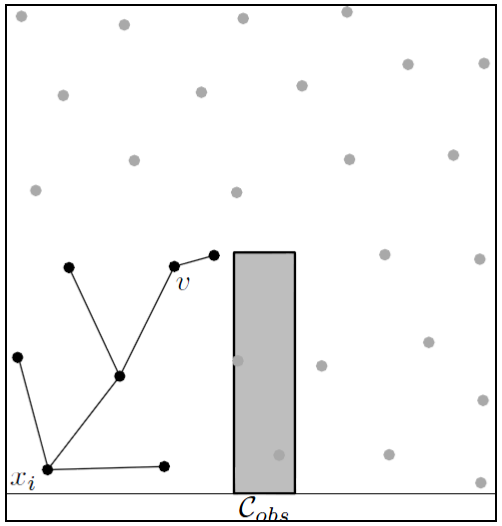
Adaptive Sampling

- No guaranteed solution with precomputed samples

- Samples may be in collision
- Edges connecting samples may be in collision

Adaptive Sampling

- When a collision check fails
  - Apply adaptive sampling template
    - Template has MPS samples of half radius in a MPS disk
    - Add the closest template sample to the collision point



Adaptive Sampling

- Add randomness to the tree
  - Node is chosen by random sample
  - Different order of tree expansion makes the adaptive sampling generates different samples

Poisson-RRT Planning Algorithm

- Primitive procedures in RRT
  - GPU parallel computation improves the performance
  - Collision checking using BVH with OBB trees
  - Nearest neighbor search using Locality-Sensitive Hashing

Outline

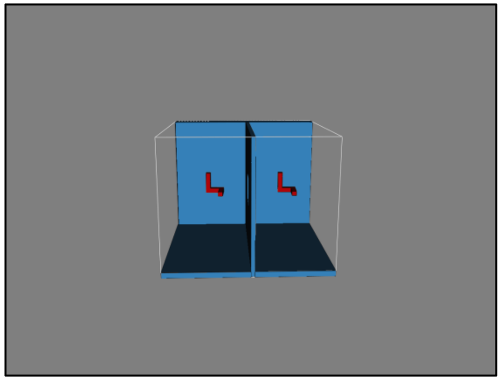
- Sampling-based Motion Planning
- Poisson-disk Sampling
- Parallel Poisson-RRT Motion Planning
- **Experimental results**
- Conclusion

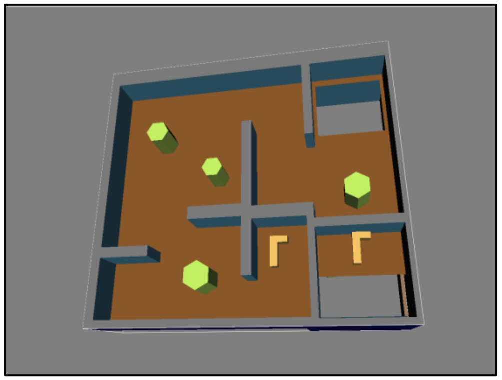
Experimental Results

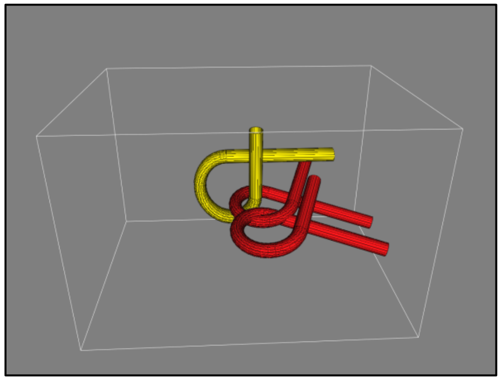
- Implementation with CUDA
- Integrated in OMPL and ROS simulator
  - 3D(6-DOFs) OMPL benchmarks
  - HRP-4 robot planning(23-DOFs)
- System
  - CPU: Intel Sandy Bridge i7-2600 (Single thread)
  - GPU: NVIDIA Geforce GTX580

Experimental Results

- OMPL Benchmarks (6 DOFs)


<Easy>

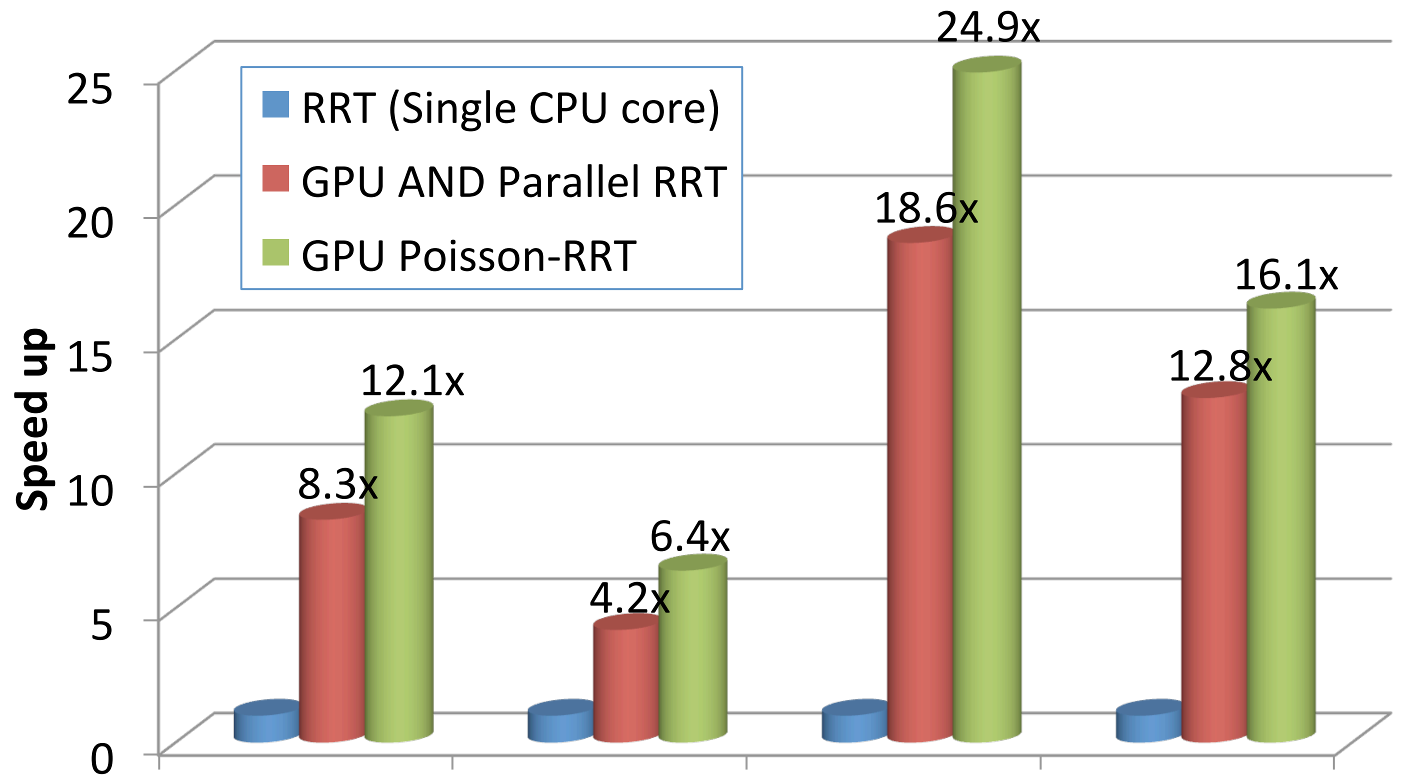

<Cubicle>


<AlphaPuzzle>

<Apartment>

Experimental Results

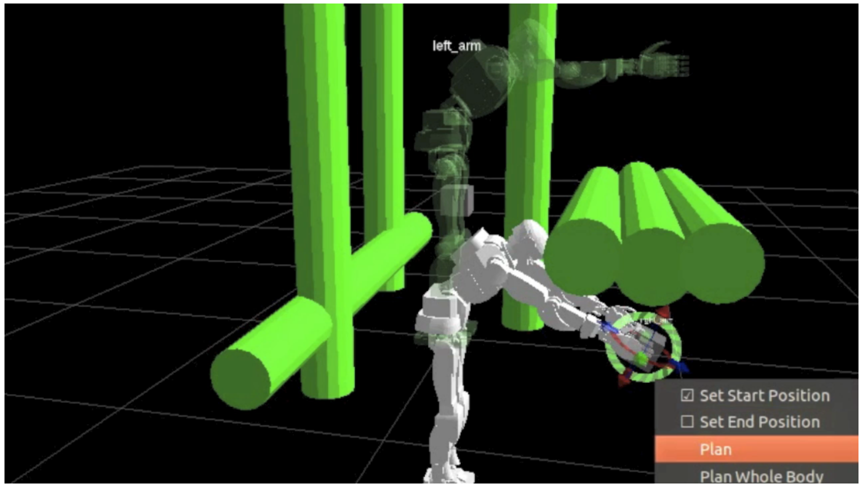
- Planning Time for OMPL Benchmarks

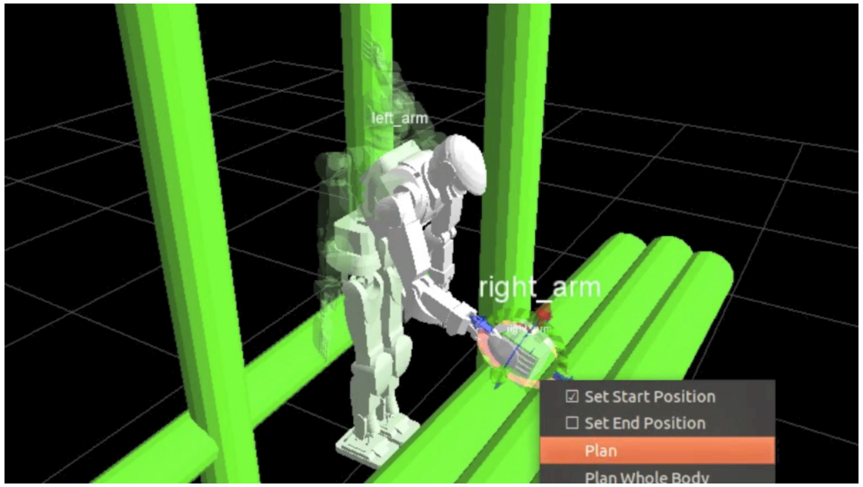


Absolute planning time for GPU Poisson-RRT	Easy	Cubicle	AlphaPuzzle	Apartment
	0.028s	0.361s	1.314s	11.877s

Experimental Results

- HRP-4 robot planning (23 DOFs)





Experimental Results

- Performance of Sample Processing

	RRT with GPU collision checking [Bialkowski et al. 2011]	Parallel RRT with space partitioning [Ichnowski and Alterovitz 2012]	GPU Poisson-RRT
Platform	GPU	CPU	GPU
Processor	Nvidia Quadro FX1800M	Intel 2.0GHz 8 core x 4	Nvidia Geforce GTX580
Time	25s	11.6s	6.9s
Processed Samples	40K	100K	160K
Time/Sample	0.625ms	0.116ms	0.043ms
Algorithm	RRT*	RRT*	RRT

Conclusion

- A GPU-friendly RRT planning algorithm
  - Parallel algorithm to exploit GPUs
  - Poisson-disk sampling improves the performance
    - 20x better than CPU-based algorithm
    - 50-100% better than prior GPU-based RRT algorithm
  - Application to task planning
- Applied for 23 DOFs robot task planning & active sensing

Future Work

- Integration with task planning applications
- Apply to complex task execution problems in challenging scenarios
  - Dynamic environment
  - Multiple constraints

Acknowledgements

- This research is supported by
  - Army Research Office
  - National Science Foundation
  - Willow Garage

Thank you

Questions?