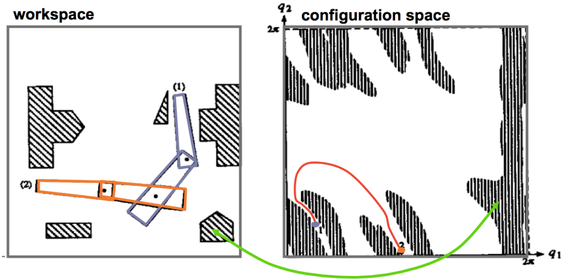
High Performance Computing Project Parallelization of the Probabilistic Roadmap Method with GPU Acceleration

Manuel Demmeler

Technische Universität München

Motivation



- Problem: find a feasible trajectory of a robot through a setting of obstacles
- Often necessary to pass feasible initial tajectories to optimization algorithms

Problem Statement

- Obstacles can not be assumed to have a closed form
- Therefore they are defined by an indicator function

$$I : \Omega \subset \mathbb{R}^d \to \{0, 1\}$$

- Collision for $q \in \Omega$, $I(q) = 1$

PRM Algorithm

PRM: Grow graphs from given start and endpoints $q_s, q_e \in \Omega$ until a connection is found, by repeating

- sample new node v randomly in environment of old nodes
- check for all possible neighbors w, if the linear connection is feasible, in this case add edge $\{v, w\}$ to the graph
- break, if the two graphs have been connected.

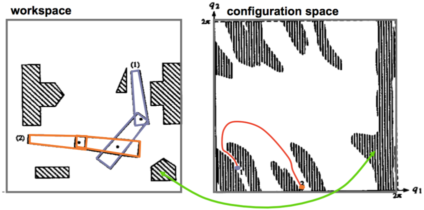
Definition

For two points v and w we define the connection with stepsize $h > 0$ as

$$[v, w]_h := \{q = \lambda v + (1 - \lambda) w \mid \lambda \in [0, 1], \|q - v\| \in \mathbb{N}_0 h\}.$$

We say, q_1 and q_2 are connected with stepsize $h > 0$, if $I(q) = 0$ for all $q \in [q_1, q_2]_h$.

Robotics Application

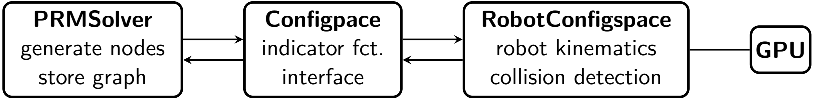


- For each linear connection, every intermediate configuration has to be checked for collisions
- $\Rightarrow$ Many independent and similar kinematics and collision calculations
- $\Rightarrow$ Well suited for usage of GPU
- Every GPU thread can make one intermediate test

Indicator Function Implementation

- Input: vectors of start and end nodes for each potential edge
- Output: vector of feasible edges
- CPU determines number of threads needed for each edge and passes nodes to GPU
- Each CUDA thread determines his intermediate configuration and checks for collsion
  - Kinematics with Denavit-Hartenberg transformations
  - Geometry parts are convex polytopes
  - Collision detection: Chung-Wang Collision Algorithm

Project Overview

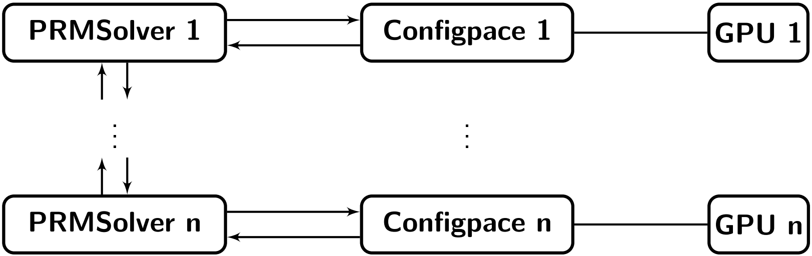


- PRM algorithm and Configuration space independent from each other
- Only connected through indicator function represented by Configspace interface

Second Level of Parallelization

Version 1:

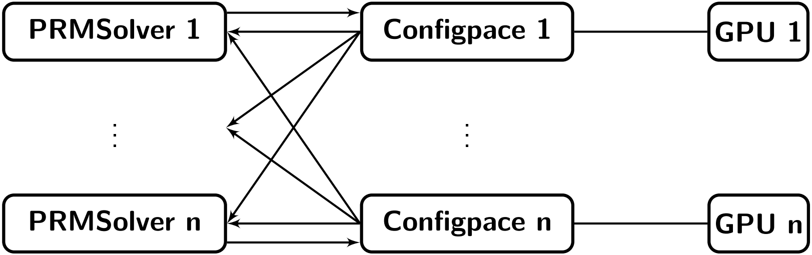
- Use multiple GPUs parallely
- Sample and connect new nodes on each processor
- Exchange nodes and edges



Second Level of Parallelization

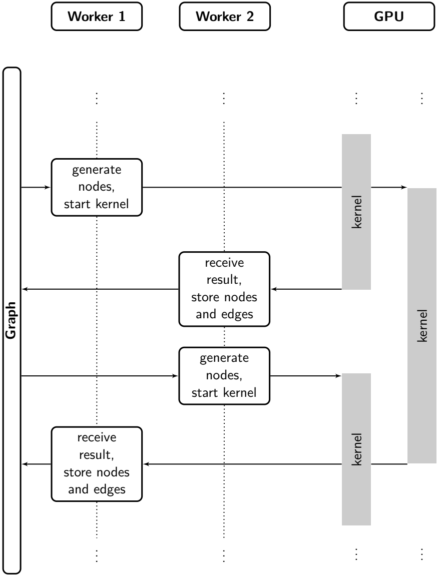
Version 2:

- Grow whole graphs on every processor
- Exchange only edges



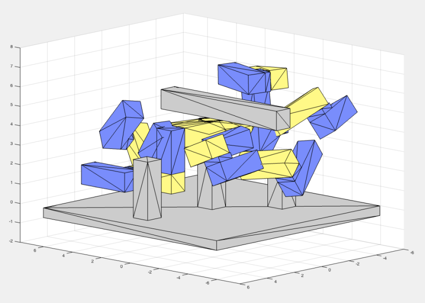
Second Level of Parallelization

Version 3: Asynchronous, overlapping kernel calls



Implementation and Testing

- GPU imlementation with CUDA
- CPU Parallelization with MPI
- Tested with a 4 axis robot scenario created with CAD/Matlab



Literatur

- D. Burschka, Lecture Robot Motion Planning, source: http://robvis01.informatik.tu-muenchen.de/courses/wegtraj/index.html
- H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki and S. Thrun, Principles of Robot Motion: Theory, Algorithms, and Implementation, MIT Press, 2005
- S. M. LaValle, Planning Algorithms, Cambridge University Press, 2006
- T. Buschmann, Skript zur Vorlesung: Roboterdynamik SS14, Lehrstuhl für Angewandte Mechanik, TU München, 2014
- K. Chung, Wenping Wang, Quick Collision Detection of Polytopes in Virtual Environments, Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 96), Mark Green (ed.), 1996
- C. Ericson, Real-Time Collision Detection, Elsevier, 2005