Motion Planning and Proximity Computations for Industrial Robots

Dinesh Manocha
Department of Computer Science
UNC Chapel Hill

dm@cs.unc.edu

http://gamma.cs.unc.edu

Motion Planning

• Widely studied in academic for 40+ years

• Good algorithms and software tools

• Technology transfer (CAD/CAM, games, simulation)

• Limited use for industrial robots
Planning tools for Industrial Robotics: Challenges

- Limited capabilities
- Limited development tools
- Lack of portability and flexibility
- Slow technology adoption

Collision & Proximity Queries

*Geometric reasoning of spatial relationships among objects (in a dynamic environment)*

- Collision Detection
- Contact Points & Normals
- Closest Points & Separation Distance
- Penetration Depth
Collision & Proximity Computations

- A key component of motion planning algorithms (90% of total time)
- Widely used in CAD/CAM, simulation and virtual prototyping
- Studied in academia for 30+ years
- Supported in robot simulation and CAD systems

Our work on Proximity Computations

- Fast algorithms for convex polytopes (1991 onwards)
- Bounding volume hierarchies for general polygonal models (1995 onwards)
- Deformable models & self-collisions (2000 onwards)
- Use of GPUs and multi-core hardware (2005 onwards)
- Multi-robot planning and coordination (2008 onwards)
Prior work on Proximity Computations

Multiple software systems

- I-Collide, RAPID, PQP, DEEP, SWIFT, SWIFT++, DeformCD, PIVOT, Self-CCD, RVO, HRVO
  
  http://gamma.cs.unc.edu/software/#collision

- More than 110,000 downloads from 1995 onwards

- Issued more than 55 commercial licenses (Kawasaki, MSC Software, Ford, Honda, Sensable, Siemens, BMW, Phillips, Intel, Boeing, etc.)

Widely used, but not on industrial robots
Recent Work: FCL

- A new collision and proximity computation library
  - Flexible: different object types/queries
  - Extensible: adding new algorithms is easy
  - Efficiency: similar performance with the best libraries

- Provide many functions from state-of-the-art research, more in future

FCL Overview
FCL: Supported Functions (Apr. 2013)

<table>
<thead>
<tr>
<th></th>
<th>Rigid Objects</th>
<th>Point Clouds</th>
<th>Deformable Objects</th>
<th>Articulated Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Discrete) Collision Detection</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Continuous Collision Detection</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Self Collision Detection</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Penetration Estimation</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Distance Computation</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Broad-phase Collision</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

FCL: Usage

- Independent code, but ROS interface is provided
- Available at [http://gamma.cs.unc.edu/FCL](http://gamma.cs.unc.edu/FCL)
- Part of MoveIt: [http://moveit.ros.org/wiki/MoveIt!](http://moveit.ros.org/wiki/MoveIt!)
FCL Application: Optimal Inverse Kinematics for Complex Path Planning

Handling Sensor Data

- Human environments
  - Clutter, dynamic obstacles
- Data from 3D sensors
  - Large number of points (~10k for laser scans, ~20k for stereo)
- Real-time computation important for fast online reactive grasping, motion planning
- Proximity computation important for many useful heuristics in robotics

*Efficient collision and proximity computation is essential for any online robot operations in human environments*
Sensor Data

- Point cloud
  - Output from laser/Kinect, etc.
  - Cannot encode unknown regions
  - Very large

- Octree (octomap)
  - Store point cloud in a compact manner
  - Support multi-resolution
  - Encode occupied/free/unknown regions

Real-Time Planning with Sensor Data

http://moveit.ros.org/wiki/MoveIt!
Proximity & Planning with Industrial Robots

http://rosindustrial.org/

NVIDIA & AMD GPU Compute Accelerators

<table>
<thead>
<tr>
<th></th>
<th>NVIDIA GTX 680</th>
<th>AMD Radeon 7970</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Tflops</td>
<td>3.09</td>
<td>3.79</td>
</tr>
<tr>
<td>Double Gflops</td>
<td>1.1</td>
<td>947</td>
</tr>
<tr>
<td>Stream Cores</td>
<td>1536 CUDA Cores</td>
<td>2048 Stream Cores</td>
</tr>
</tbody>
</table>

Commodity Tera-Flop Processor (peak performance)
**Heterogeneous Processing**

CPU + GPU on the same chip

---

**GPU-based Sampling Planner**

OMPL Benchmarks

- **Easy Cubicle**
  - Absolute planning time: 0.028s
  - Speed up: 8.3x

- **AlphaPuzzle**
  - Absolute planning time: 1.314s
  - Speed up: 24.9x

- **Apartment**
  - Absolute planning time: 11.877s
  - Speed up: 12.8x
Real-Time Planning: High DOF Robots

• High-DOF robots (40 DOF for humanoids)

• Generate collision-free and smooth paths

• Dynamics constraints (e.g. dynamic stability)

• Moving obstacles (e.g. humans)

Real-Time Planning

• Use optimization-based techniques

• Formulate constraints

• Parallel computation on multi-core CPUs and many-core GPUs
Parallel Trajectory Optimization

- Parallel optimization of multiple trajectories
  - Use Multiple threads
    - Start from different initial trajectories
    - Trajectories are generated by quasi-random sampling
  - Exploits the multiple CPU cores (multi-cores) or GPU-based cores (many-cores)

Real-Time High DOF Planning
Parallel Trajectory Optimization

Performance improvement with number of cores

Going Forward

- Significant recent progress in algorithmic technology for motion planning and proximity computations
- Good software tools: FCL, ROS, MoveIT, OpenRAVE, etc.
- Applications
  - Advanced manipulation
  - Advanced perception
  - Flexible automation
- Major Challenge: Interface with robot devices and industrial use
Acknowledgements

• Army Research Office
• National Science Foundation
• Intel
• Willow Garage