Programming by Demonstration: Removing Suboptimal Actions in a Partially Known Configuration Space

J. Chen and A. Zelinsky

Systems Engineering, Research School of Information Sciences and Engineering
The Australian National University
Canberra, Australia
What is Programming by Demonstration (PbD)?

- Robot programming based on showing rather than coding
- Human provides a demonstration of the task to be programmed
- PbD interface interprets the demonstration to derive low-level control commands
Why Programming by Demonstration?

Advantages:

- simple and natural for humans
- powerful because makes use of human’s high level of skill

Limitations:

- human can often include unimportant, inconsistent, or sub-optimal actions in the demonstration
What is Configuration Space?

Configuration of object represented as a point in an n-dimensional space (Lozano-Perez ’83)

Task Space

distinct contact formations

Configuration Space

distinct C-surfaces
Why a Partial Knowledge of Configuration Space?

- Our method uses a partial knowledge of C-space to remove sub-optimal actions
- Use partial knowledge of C-space determined in related work (published in ICRA ’01)
- Related work derives C-space in regions visited in the demonstration
The Problem - Removing Sub-Optimal Actions

Find an efficient, obstacle-free path between two points on a C-surface
Removing Sub-Optimal Actions - Our Solution

Identify that C-surface is obstacle-free at two types of points:
- those in demonstrated paths (α)
- those exactly beside demonstrated paths on neighboring C-surfaces (β)

Method involves 4 steps:
- Create boundary segments
- Grow likely-free regions from boundary segments
- Create interior segments from demonstrated paths
- Create connectivity graph representing connectivity between obstacle free points
Creating Boundary Segments

- demonstrated paths on boundary C-surfaces identified
- projection of raw points in these paths onto boundary C-surfaces required
Generating Likely-free Regions

- grow likely-free regions in front of boundary segments on our C-surface
- generate points at a maximum distance of “md” away from boundary segments
- points generated within known boundaries of the C-surface
Generating If-Regions: a specific example

- specific example of generating points in a likely-free region from a boundary segment
- known boundaries on C-surface not violated, i.e. spindle body does not pass into right support
Creating Interior Segments

- interior segments: project raw demonstrated points onto the C-surface
Creating and Searching a Connectivity Graph

- capture connectivity of points in lf-regions and interior segments in a connectivity graph
- Assign costs to arcs in graph equal to distance between points they connect
- search graph for least cost path to obtain sub-optimality free path on C-surface
Results

- Original demonstrate path too long i.e. suboptimal

- our method produces a shorter, more efficient path than the one demonstrated

- not simply a matter of deriving the shortest path on the C-surface between the start and end points
Conclusion

Compared to similar work, eg. Delson-West ’96, Kaiser-Dillmann ’96, De Schutter et. al. ’96:

Advantages:

• More advanced than simply removing loops in a path
• No pre-assumed demonstration topology - start and end points of different paths do not have to coincide
• Applicable to C-space (task dof) of any dimension

Limitations:

• appropriate tuning of parameters in the method required to ensure obstacle free path derived
• further research into automatics setting of parameters required