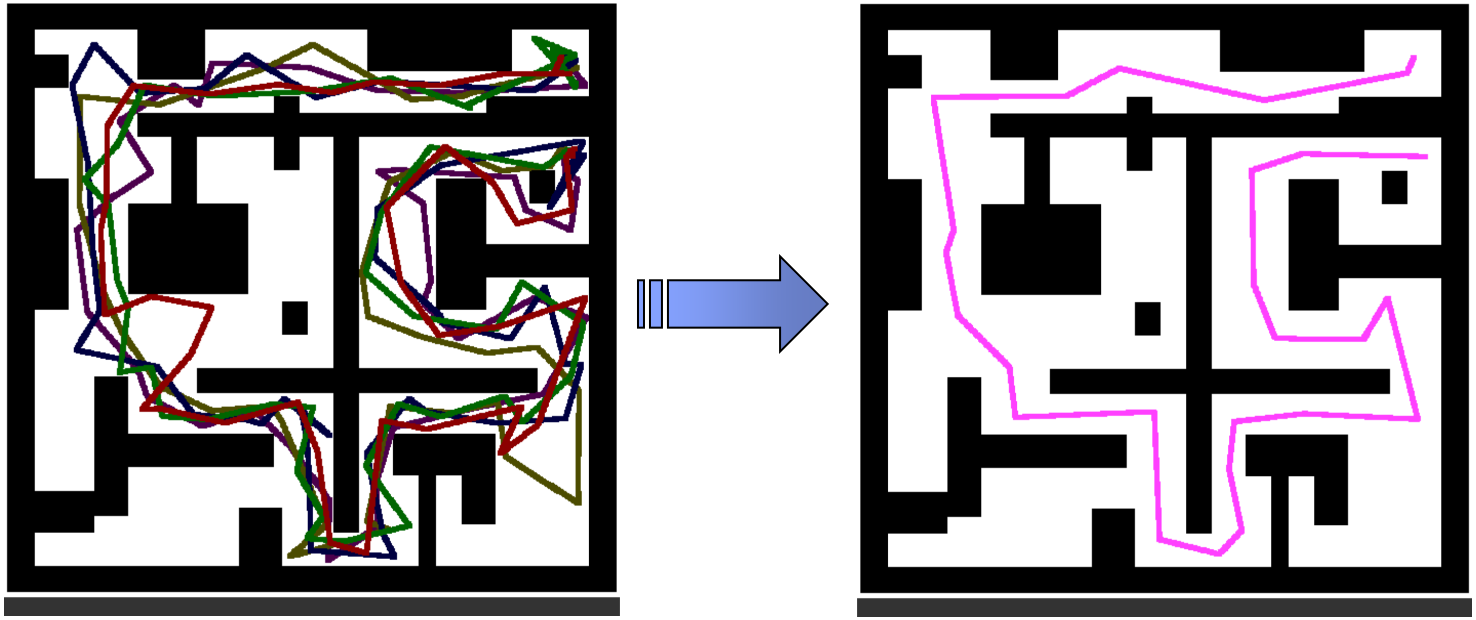
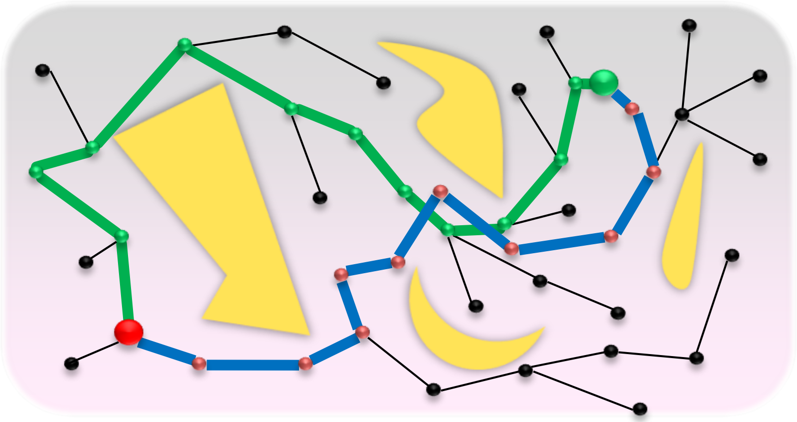
# A Little More, a Lot Better: Clustering, Comparing and Merging Motion Paths

Barak Raveh, Angela Enosh and Dan Halperin
Tel-Aviv University, School of Computer Science
Hebrew University, Jerusalem, Institute for Medical Research



# Talk Outline

- **Improving Path Quality by Path Hybridization**
- Path Alignment and Clustering

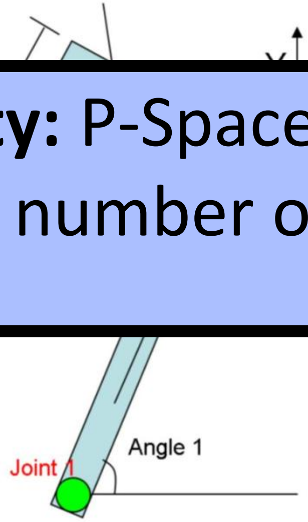


# The Motion Planning Problem

Planning the motion of a robot (or a moving object) in a *k-dimensional* C-space among obstacles

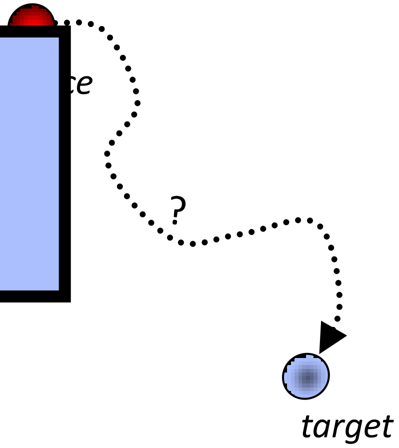
## The world

Workspace with (static or moving) obstacles



## Robot configuration

Defined by *k* degrees of freedom



## Motion Query

From source configuration to target configuration

**Complexity:** P-Space with respect to number of *dofs*

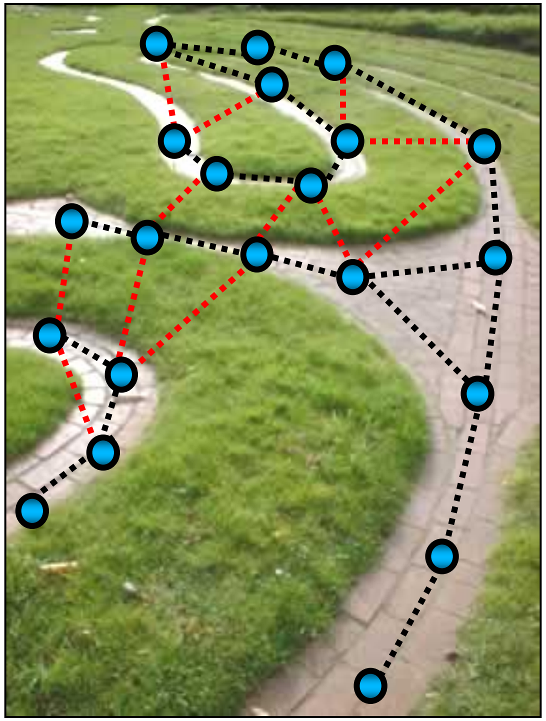
*(Reif, 79')*

# Sampling-based Algorithms that Create *Roadmaps* in High-Dimensional C-spaces

- Probabilistic Roadmap (PRM, *Kavraki et al., 96'*)
- Rapidly-exploring Random Trees (RRT, *LaValle and Kuffner, 01'*)
- Expansive-Space Trees (EST, *Hsu et al. 99'*)

## PRM Algorithm – example in two-dimensional configuration space:

- Randomly sample *n* valid robot configurations
- Connect close-by configurations by dense sampling ("local-planning")
  - Discard invalid edges ( = collisions)

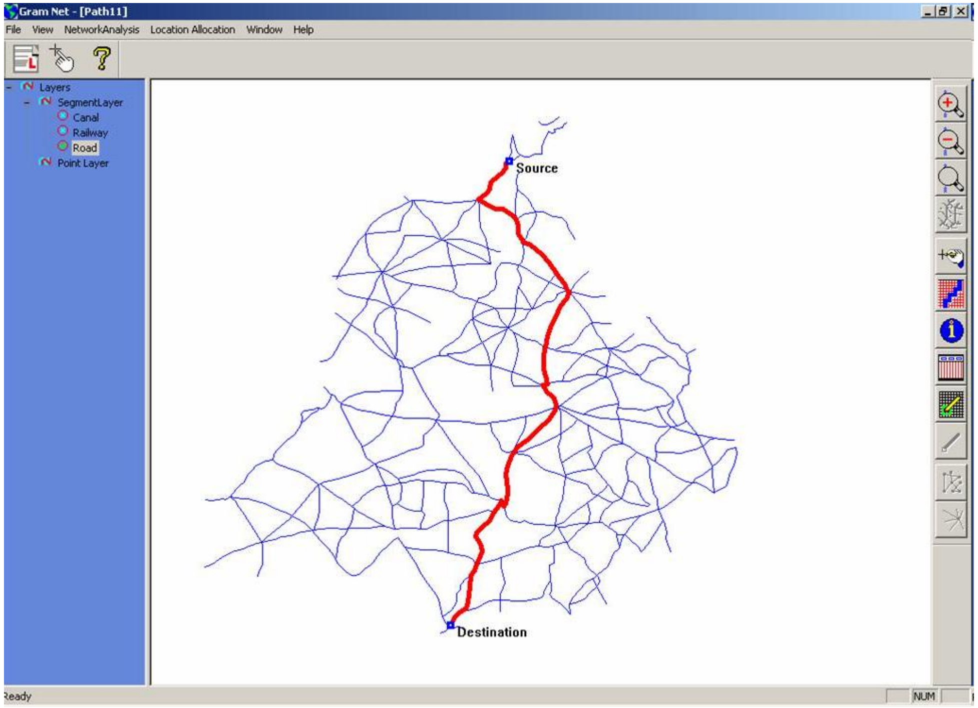


**The Basic Query :** Collision Detection

(reviewed – Lin & Manocha, handbook of Discrete and Comp. Geom. 2004)

# High-Quality Paths

- **Short** paths
- **High-clearance** paths
  (away from obstacles)
- **smooth** paths
- **low-energy** paths
  (in physical systems):



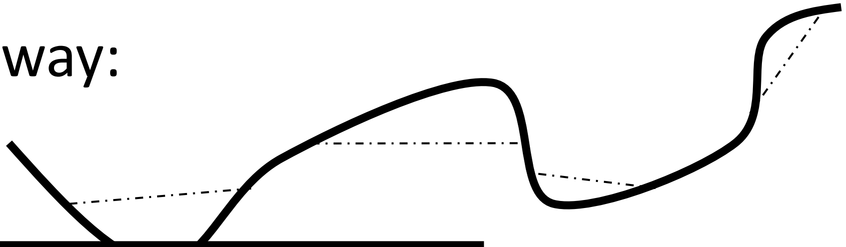
NP-complete even in very simple settings

*(e.g., Canny and Reif, 87')*

# Related Work

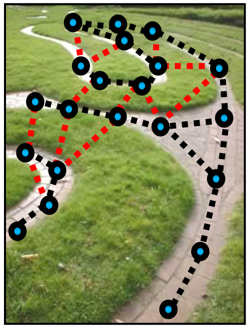
## Path Length:

- Self-shortcuts of output pathway:



- PRM with c
  **slow –** road-m

All are ad-hoc solutions to a specific quality criterion



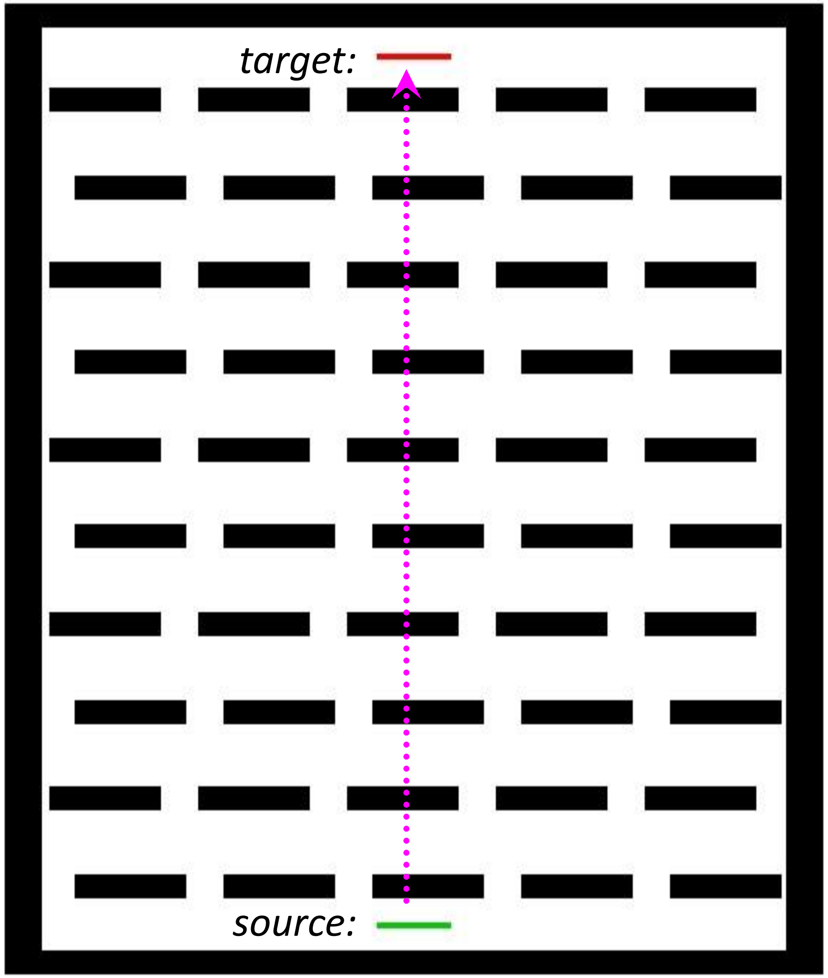
- PRM with *useful* cycles – adding only significant short-cuts to road-map *(Nieuwenhuisen et al., 04')*

## Path Clearance:

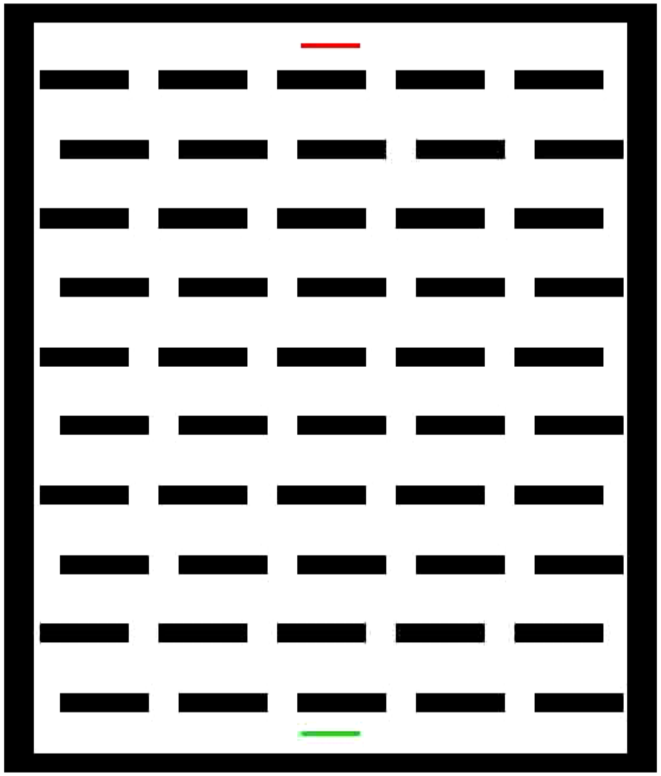
- Improving paths clearance by iteratively retracting into the medial-axis

  *(Wilmarth et al. 97', Geraerts et al. 07')*

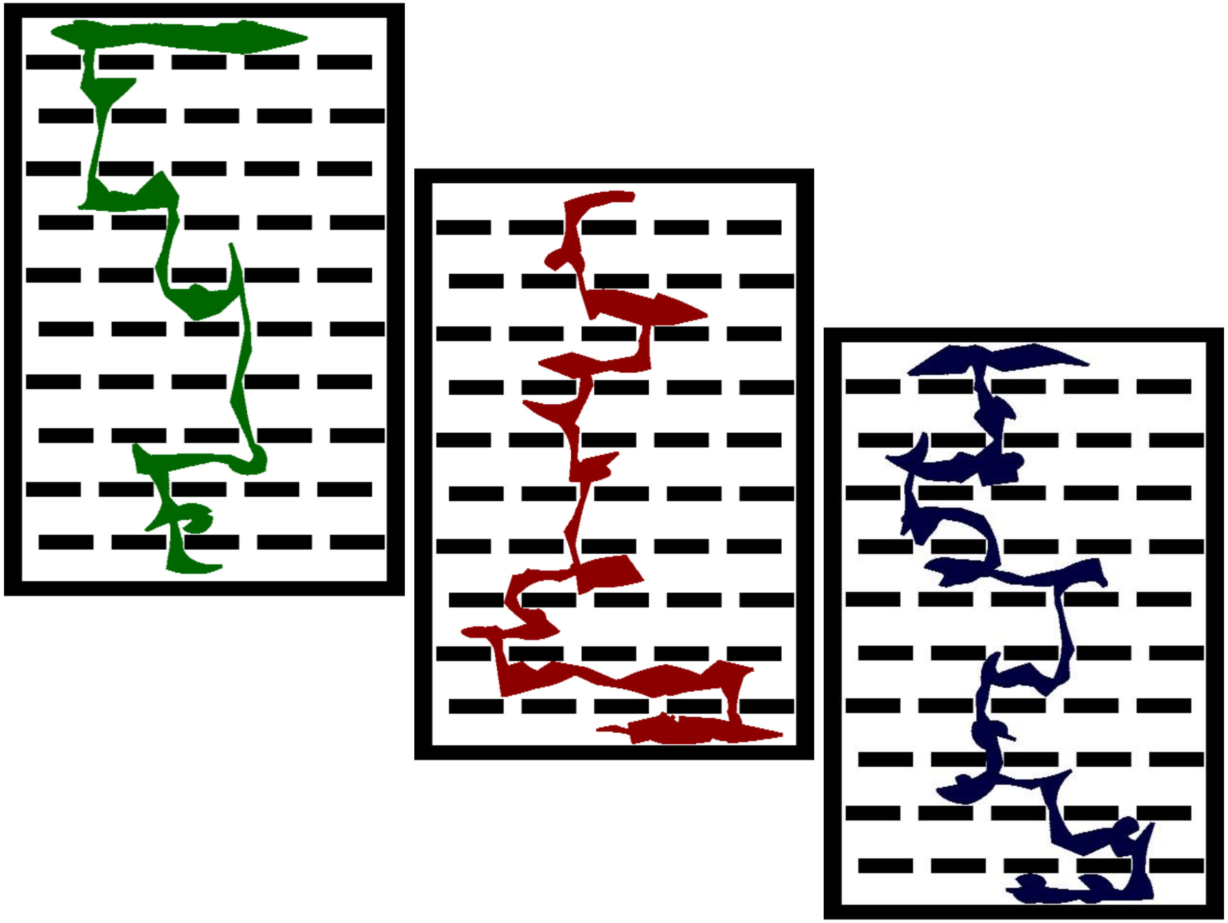
# **Example:** Move the Rod from to the Top of a 2D grid *(rotation + translation)*



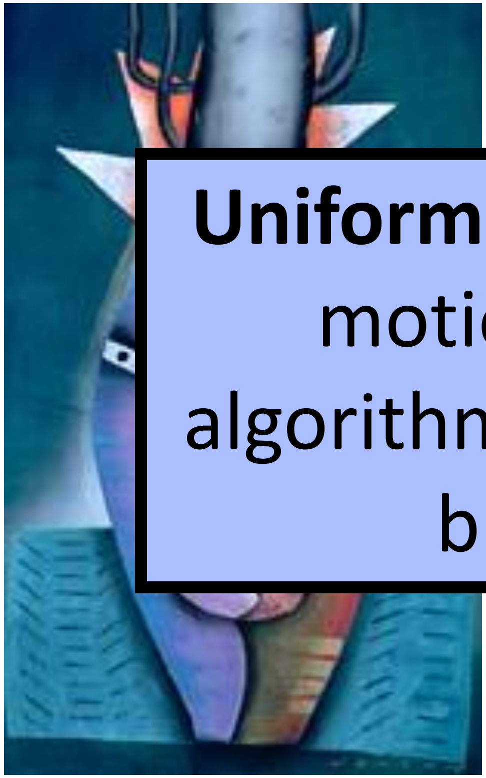
# Randomly Generated Motion Path

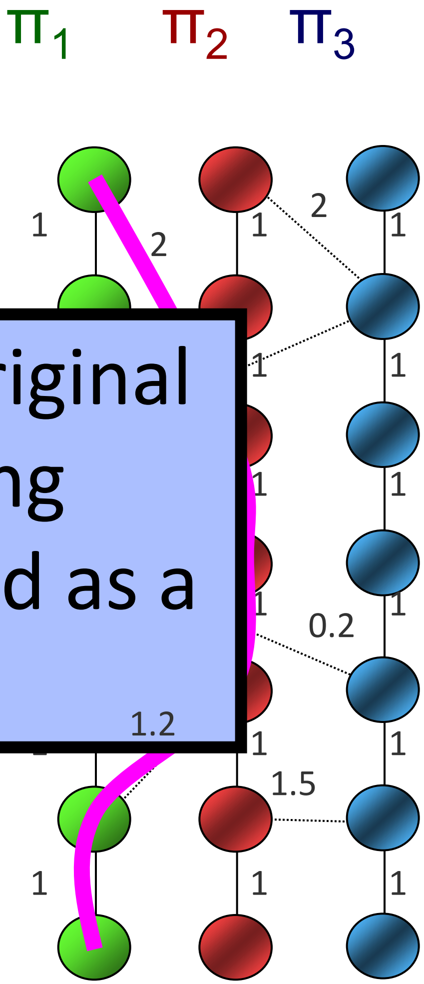


# 3 Randomly Generated Motion Paths:



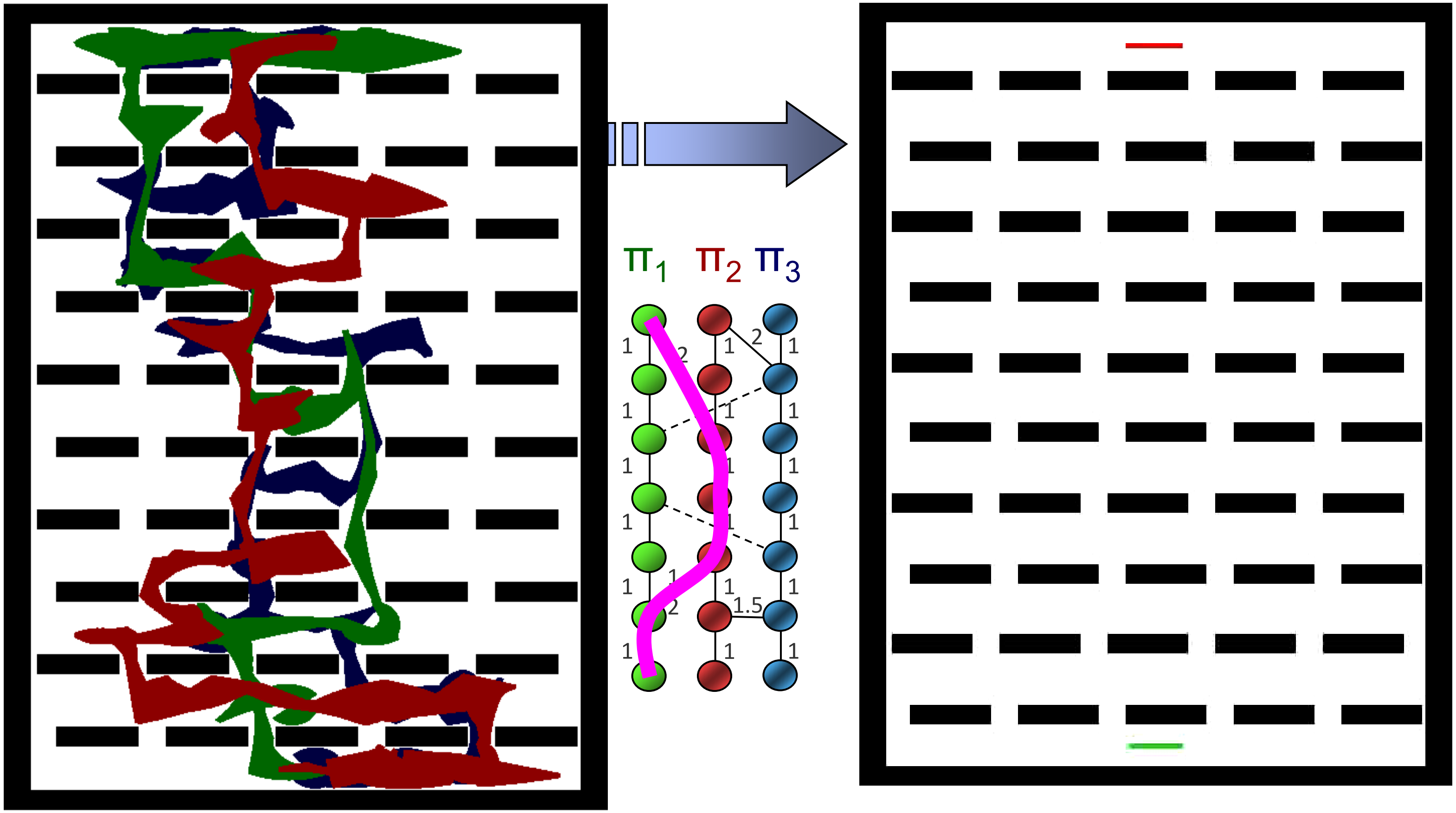
# H-Graphs: Hybridizing Multiple Motion Paths = looking for shortcuts *(Raveh, Enosh and Halperin, 2011)*



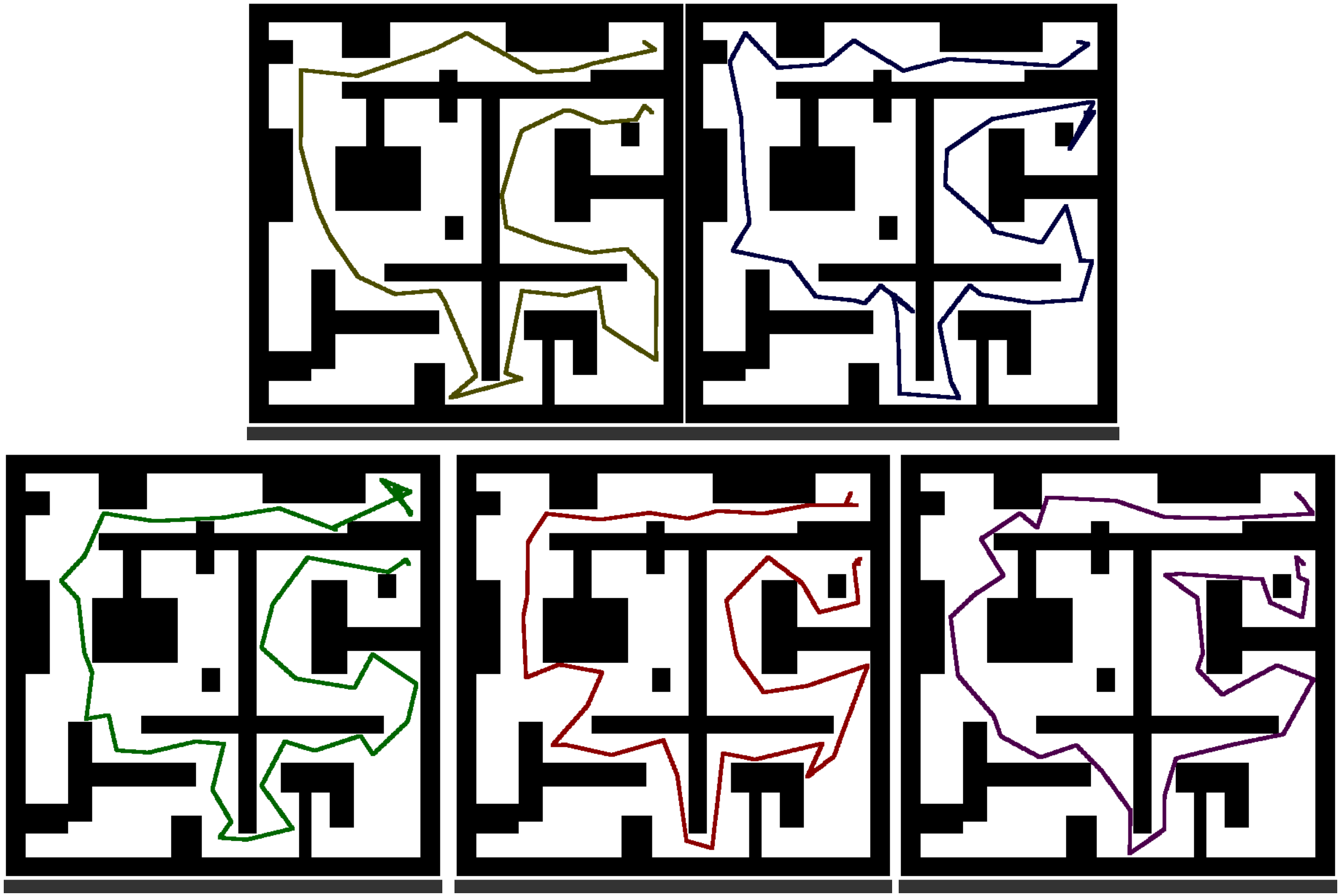


**Uniformity:** the original motion planning algorithm is treated as a black-box

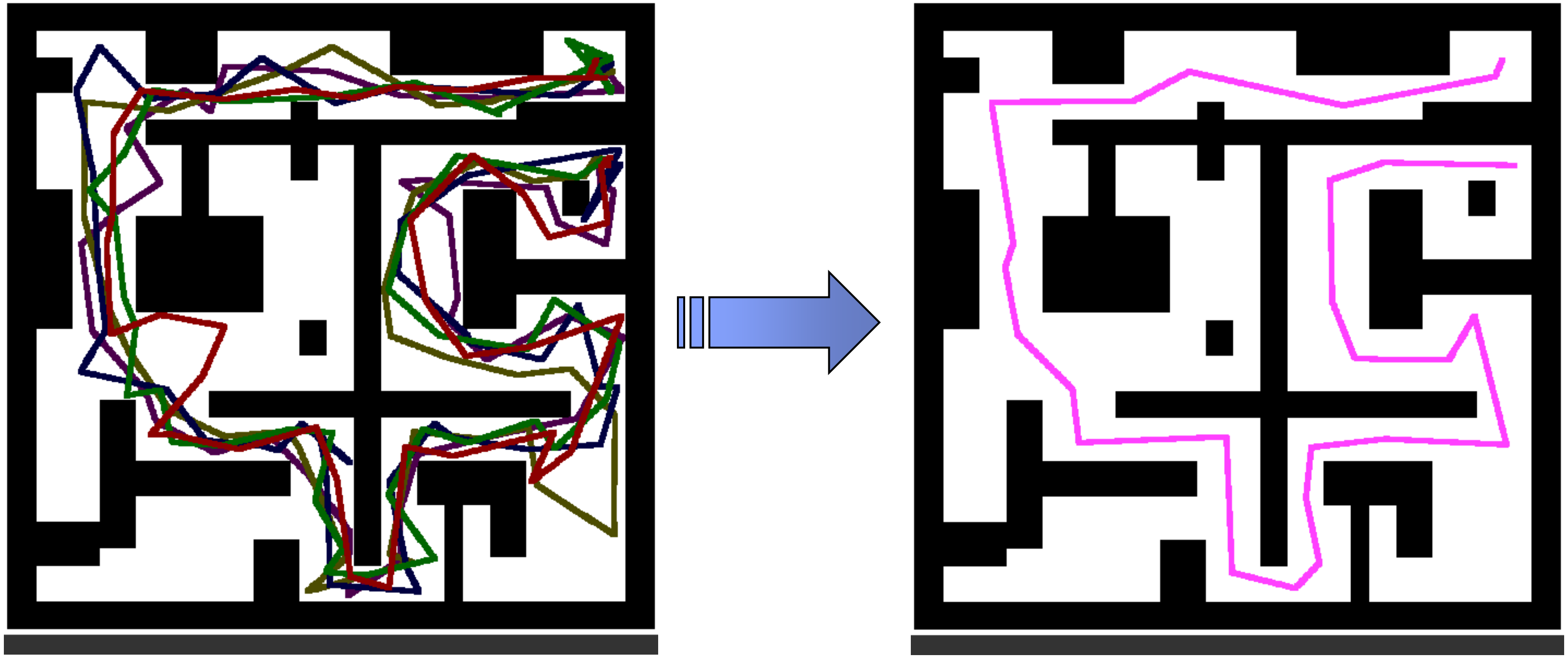
# Hybridizing Three Random Motion Paths



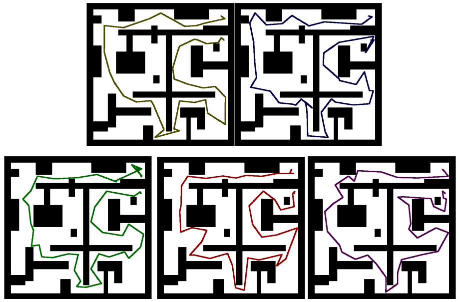
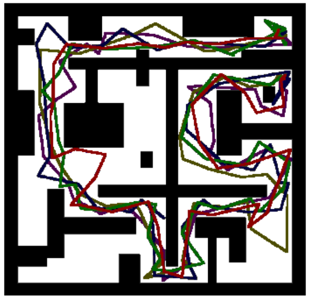
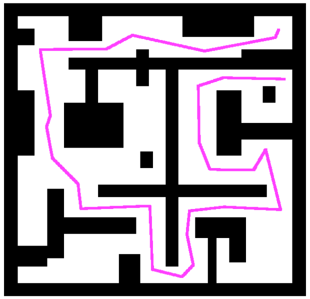
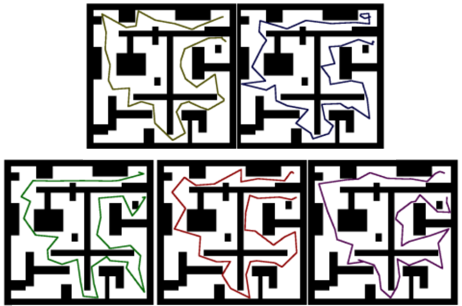
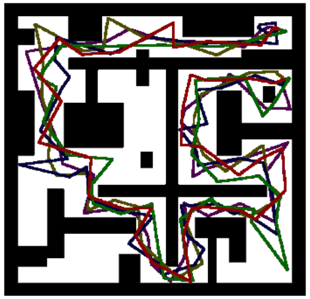
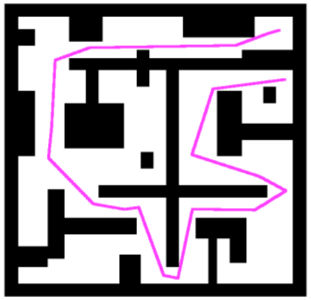
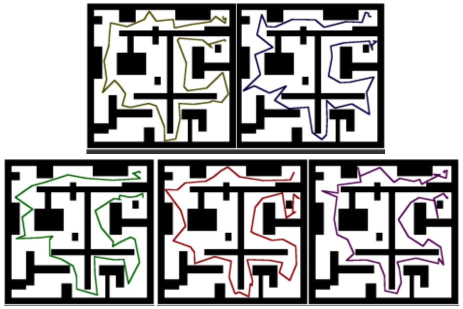
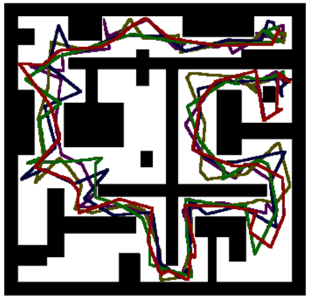
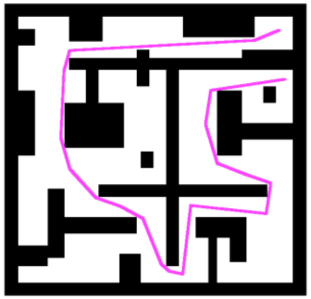
# General Treatment of Quality Criterion: Path Clearance



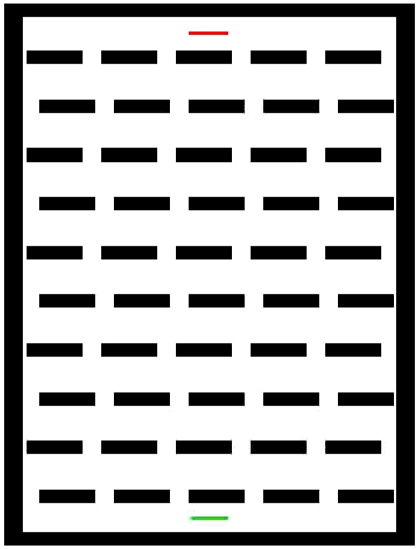
# General Treatment of Quality Criterion: Path Clearance

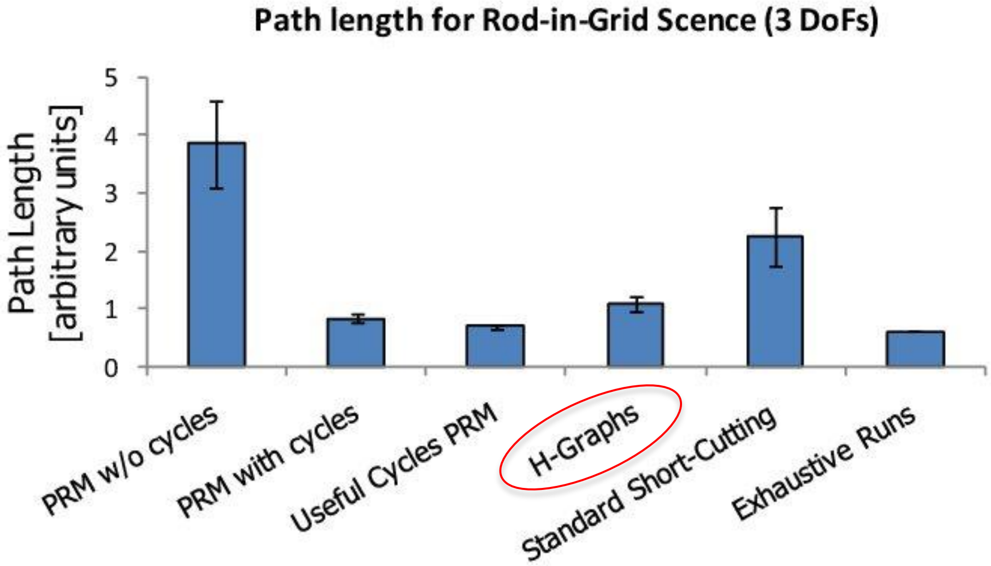


# General Treatment of Quality Criterion

| **Quality Measure** | **The Input Paths** | **H-Graph** | **Output Path** |
|---|---|---|---|
| *Clearance and length (emphasis on clearance)* |  |  |  |
| *Clearance and length (emphasis on length)* |  |  |  |
| *Path length* |  |  |  |

# **Rod-in-Grid Scene:** 3 Degrees of Freedom

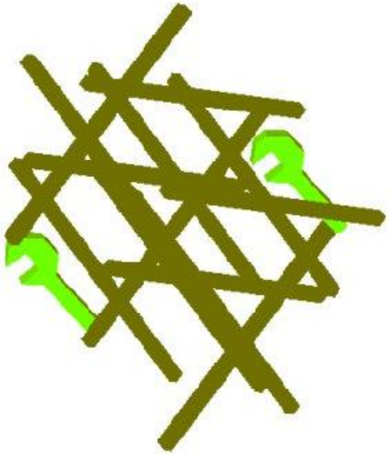


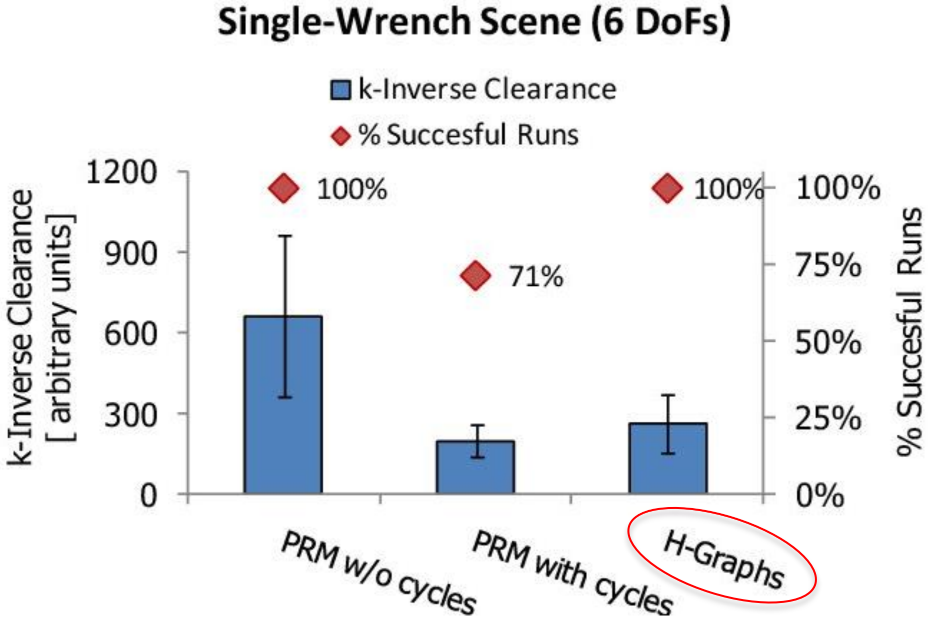


Implemented in the **OOPSMP package** (Plaku, Moll and Kavraki), collision detection – **PQP** (Lin and Manoch)

# **Wrench Scene:** 6 Degrees of Freedom

Moving the wrench among the metal beams (rotation + translation)

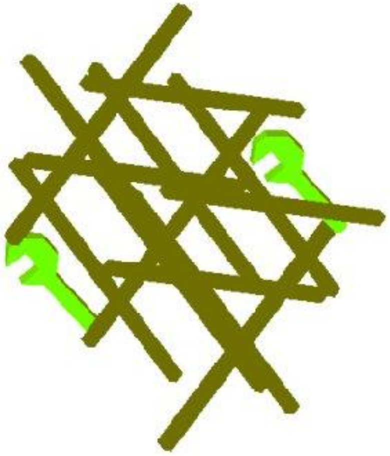


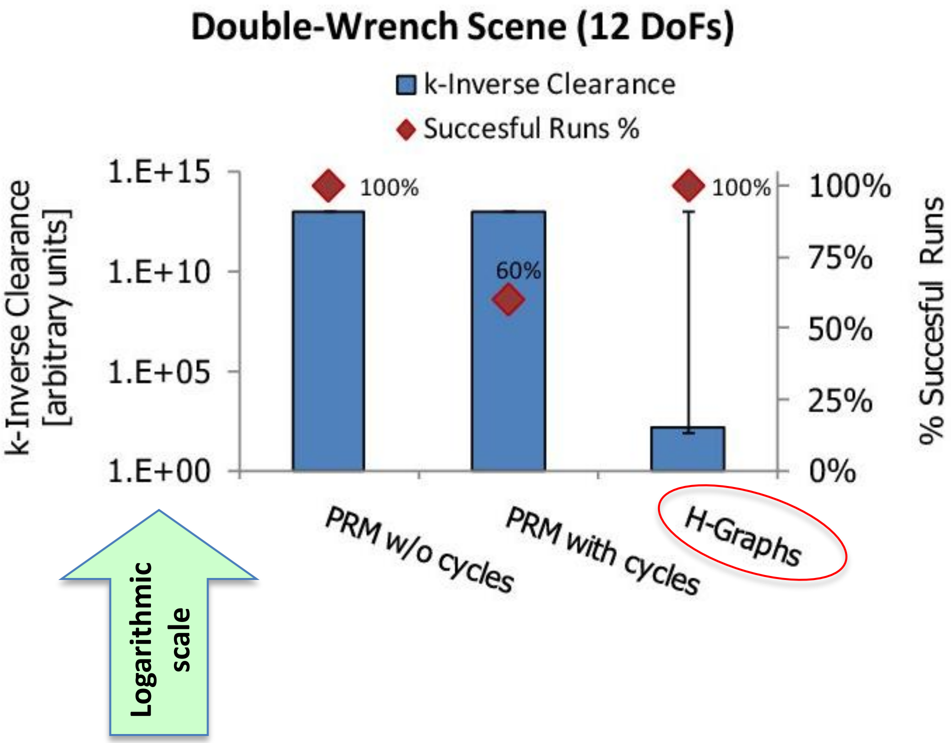


Scene adapted from Nieuwenhuisen et al., ICRA 04'

# **Double-Wrench:** 12 Degrees of Freedom

Switching the two wrenches (rotation + translation x 2)



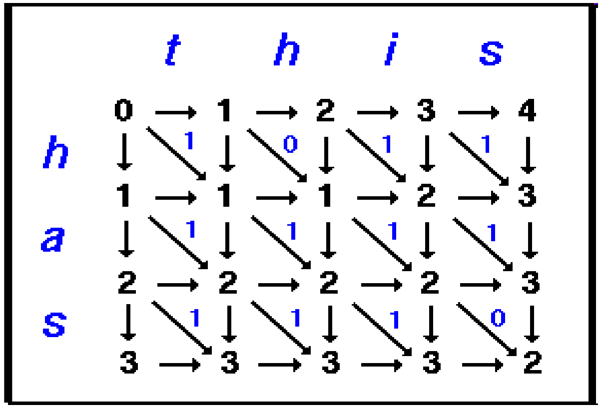


**Conclusion** ➔H-Graphs become particularly useful for high-dimensional problems (at least in this example)

Scene adapted from Nieuwenhuisen et al., ICRA 04'

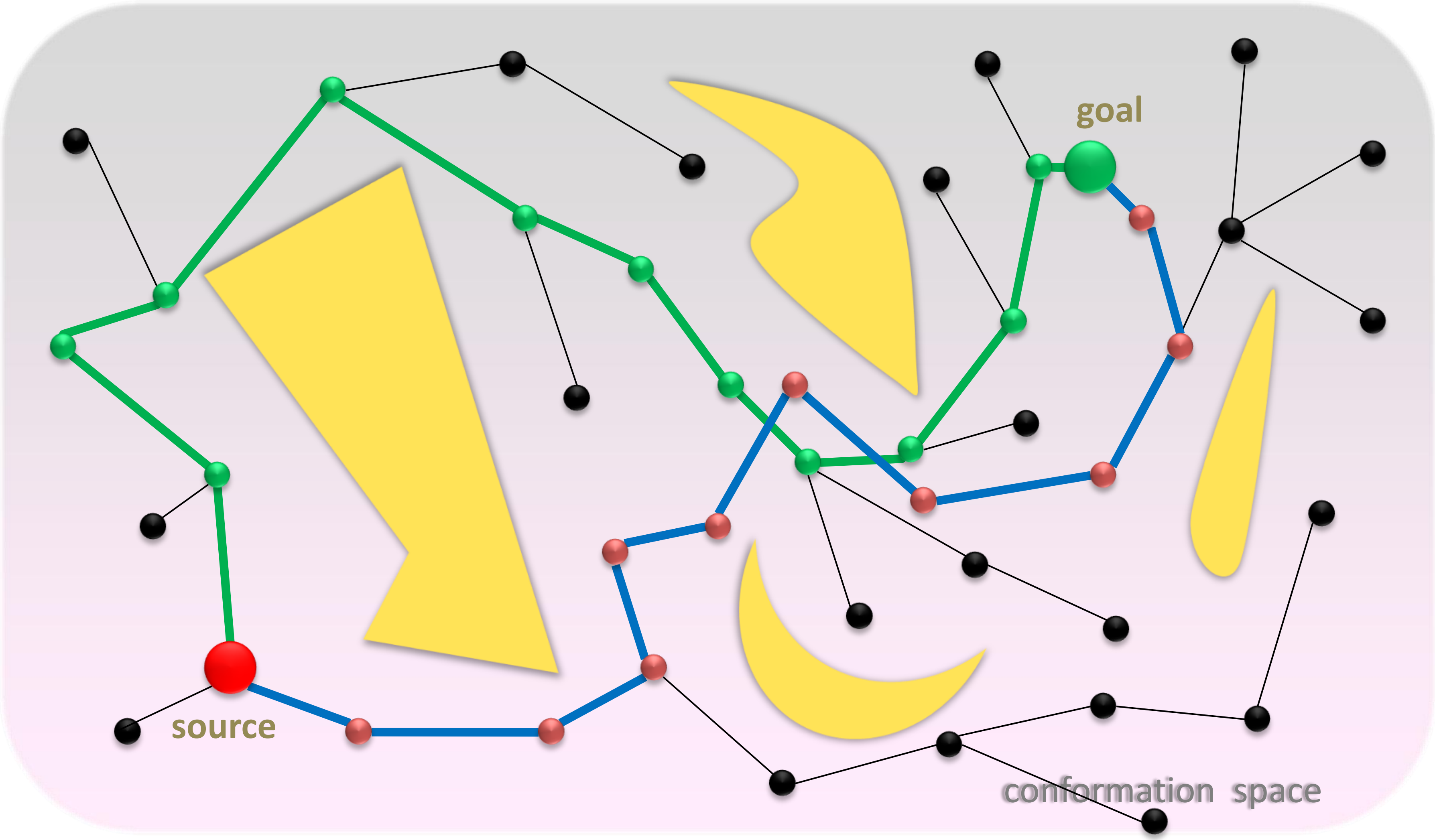
# Talk Outline

- Improving Path Quality by Path Hybridization
- **Path Alignment and Clustering**

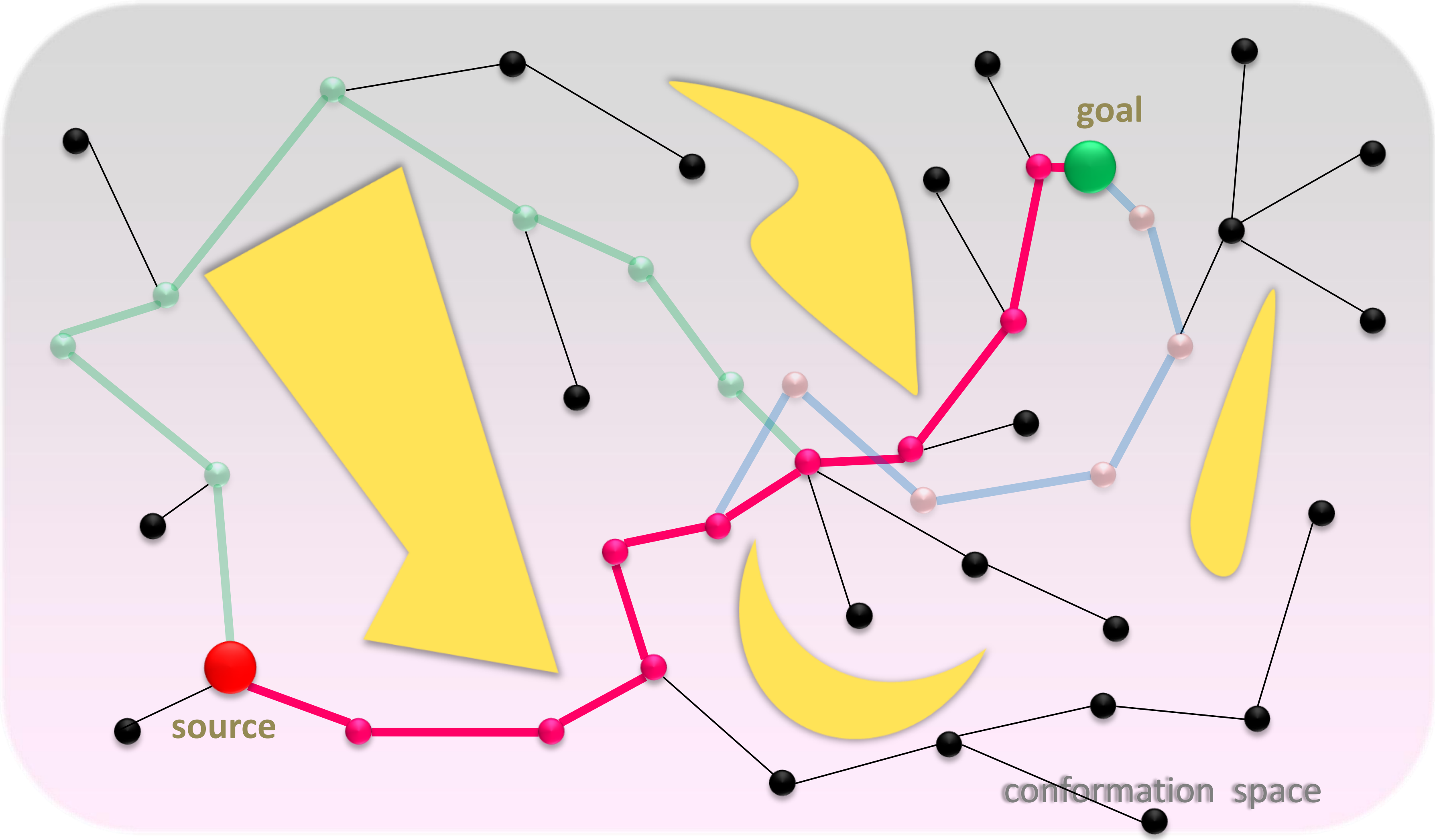


| | | t | h | i | s |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| h | 1 | 1 | 1 | 2 | 3 |
| a | 2 | 2 | 2 | 2 | 3 |
| s | 3 | 3 | 3 | 3 | 2 |

# Path Hybridization in C-space



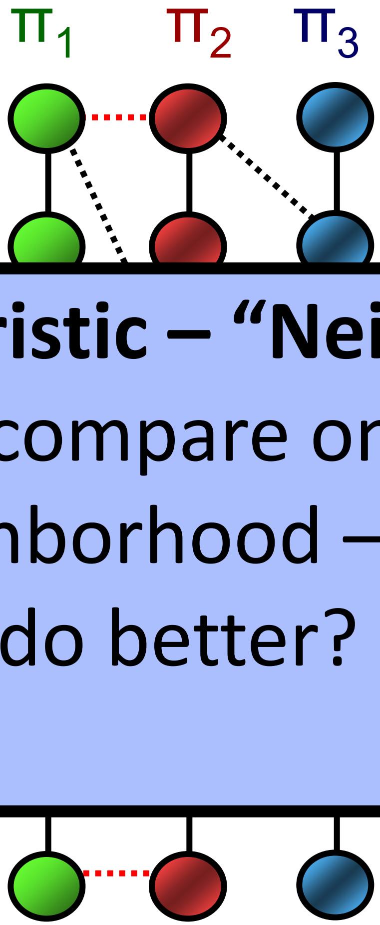
# Path Hybridization in C-space

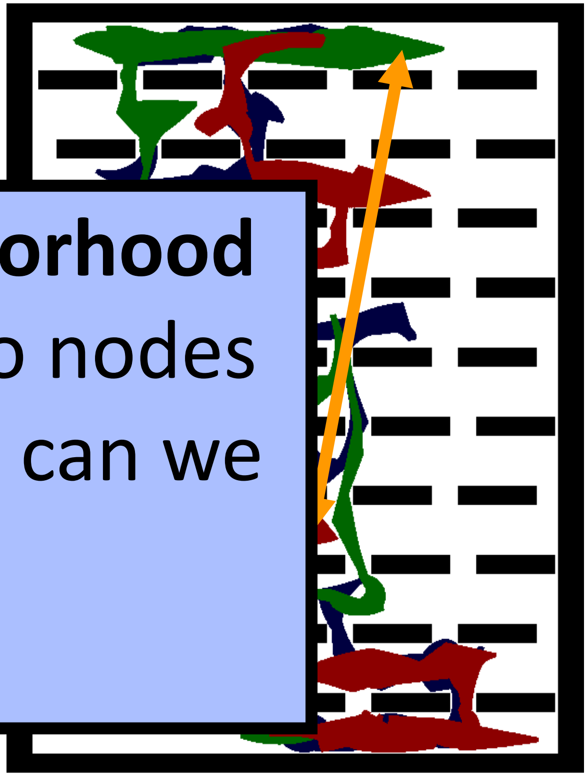


# Running-Time Bottleneck for Hybridization:

## Trying to Connect Nodes from Different Paths

In a naïve implementation: $O(n^2)$ potential edges need to be tested





**Simple Heuristic – “Neighborhood H-Graphs”:** compare only to nodes in local neighborhood – but can we do better?

# Edit Distance String Matching
# ➔ Linear Alignments of Motion Paths

(Enosh, Raveh *et al.*, Biophysical J 2008)

Comparing “This dog” and “That Dodge” with insertion / deletions / replacement:

T H I – S D O – G –

T H A T – D O D G E

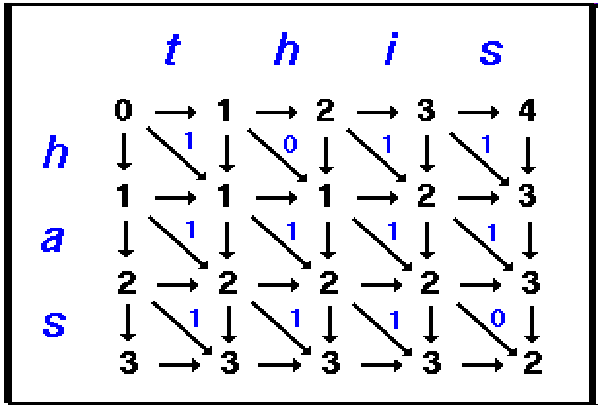
---

**Classical dynamic-programming algorithm:**

→ insertion

↓ deletion

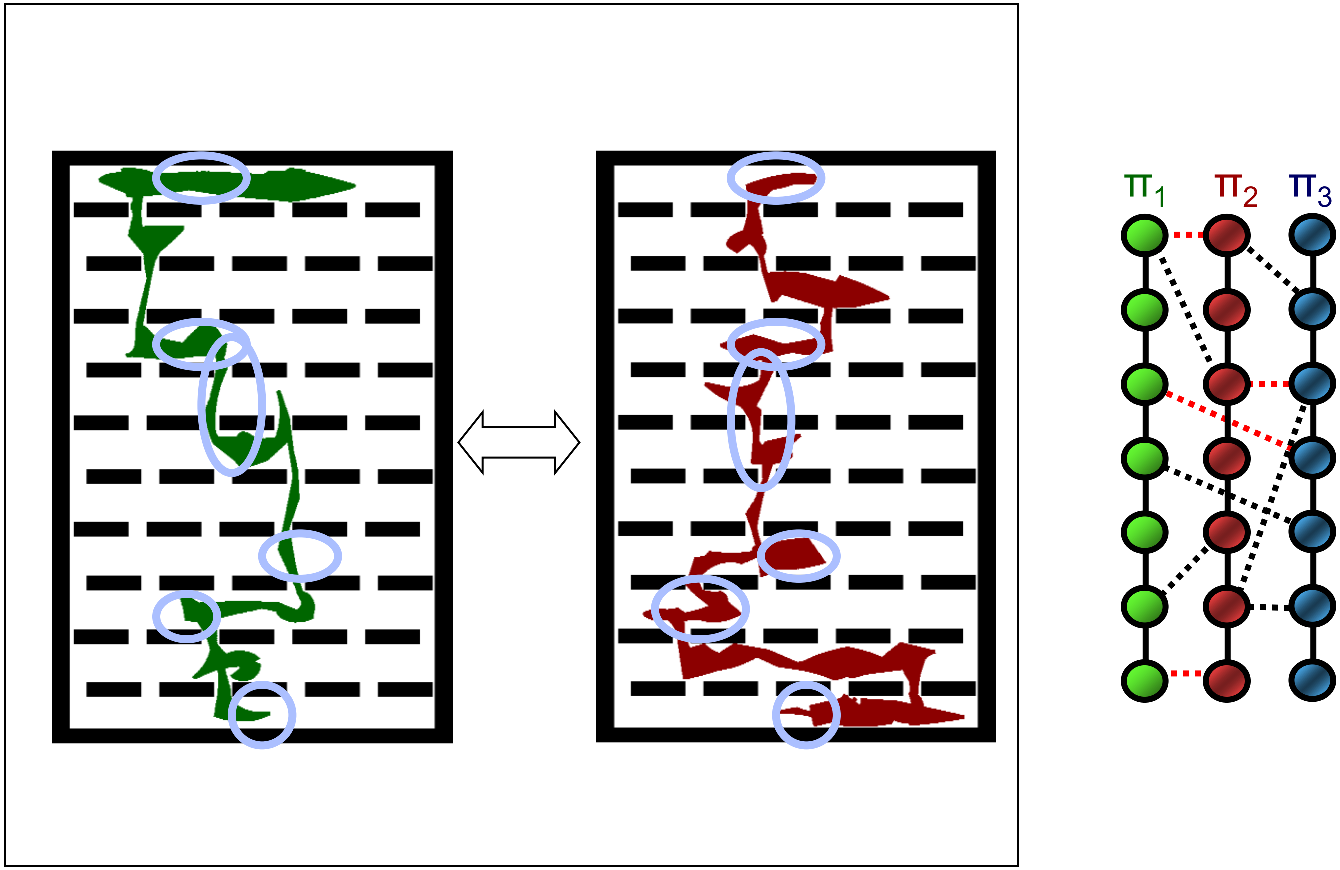
↘ replacement



|   | t | h | i | s |
|---|---|---|---|---|
|   0 | 1 | 2 | 3 | 4 |
| h 1 | 1 | 1 | 2 | 3 |
| a 2 | 2 | 2 | 2 | 3 |
| s 3 | 3 | 3 | 3 | 2 |

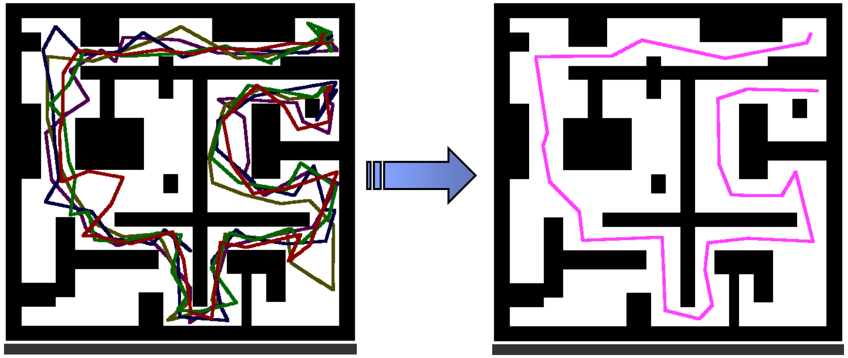
# Alignment Length is Linear

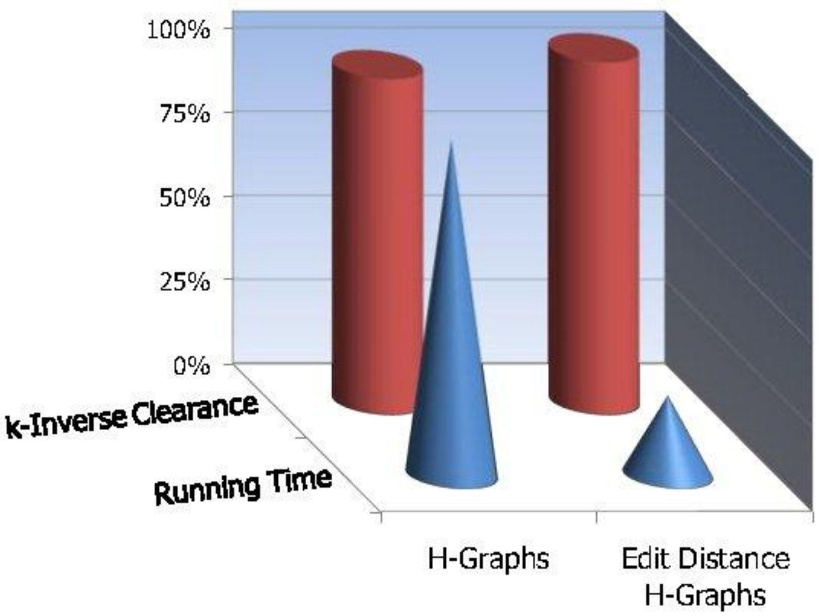
## Now testing only O(n) edges along the alignment



# Comparison of Running Times

- Hybridizing five motion paths in a 2-D maze:
  - From 3.52 seconds to 0.83 seconds on average (75% decrease), with comparable path quality





# Application: Clustering via Path Alignment

- Fast path alignment (reminiscent of curve matching) is extremely useful for clustering a large body of motion paths into sub-classes
- Useful in high-dimensional, hard-to-visualize configuration spaces

# Summary

- H-Graphs effectively produce high quality motion paths
- Uniform treatment of:
  - Motion planning algorithms
  - Path optimality criteria
- Edit-distance H-graphs
  - reduce computation time by alignment of input motion path (quadratic → linear)
  - Path clustering

# **In a Nutshell:** Incorporating Partial Information and Imposing External Cues

*(Raveh, Enosh et al., PLoS Computational Biology, 2009)*

Why use partial information?

- ✓ Reduce (vast) combinatorial search space
- ✓ Reduce random noise
- ✓ Incorporate experimental / expert knowledge
  - → common language

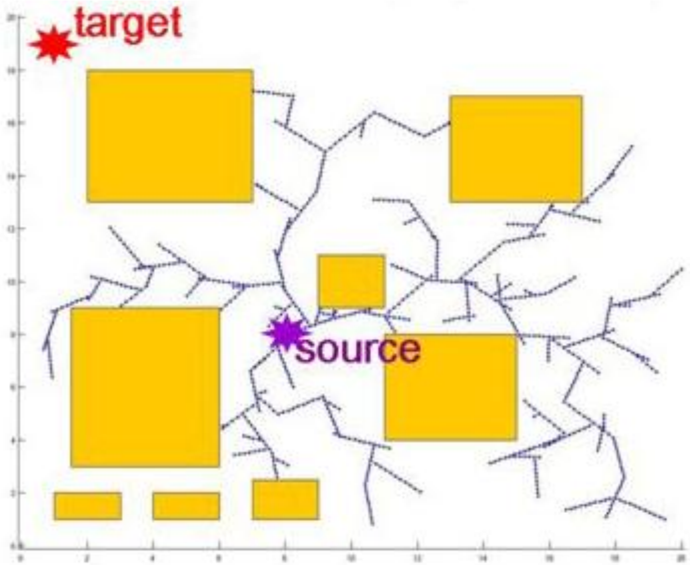
Types of information:

- Type I: constraints (cues) on motion and target
- Type II: restriction of DoFs

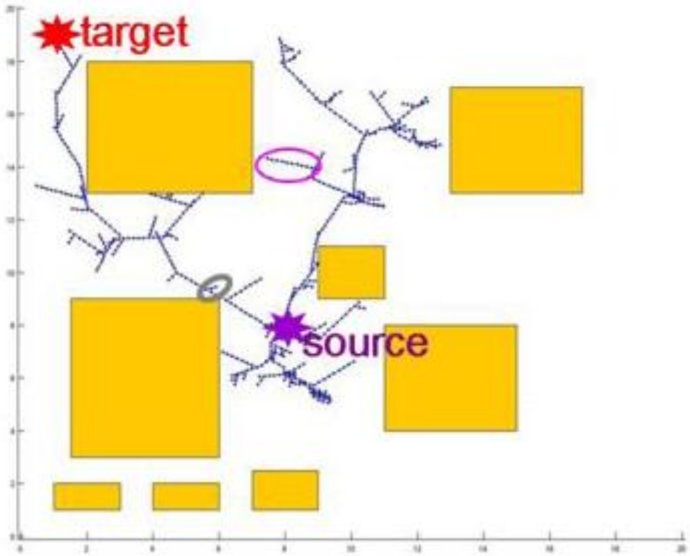
# **In a Nutshell:** Incorporating Partial Information and Imposing External Cues

*(Raveh, Enosh et al., PLoS Computational Biology, 2009)*

2D toy model **without** Partial Information



2D toy model **with** Partial Information



experimental knowledge / expert intuition / etc.

# THE END