Introduction to Robotics

Amitabha Mukerjee

IIT Kanpur, India

What is a Robot?

Robot properties:

- Flexibility in Motion
  - Mobile robots

daksh ROV: de-mining robot
20 commissioned in Indian army 2011.
100+ more on order
built by R&D Engineers, Pune

daksh platform derived
gun mounted robot (GMR)

Want your personal robot?

Roomba vacuum
Cleaning robot

By i-robot
Price: ~ rs. 15-30K

How to vacuum a space?

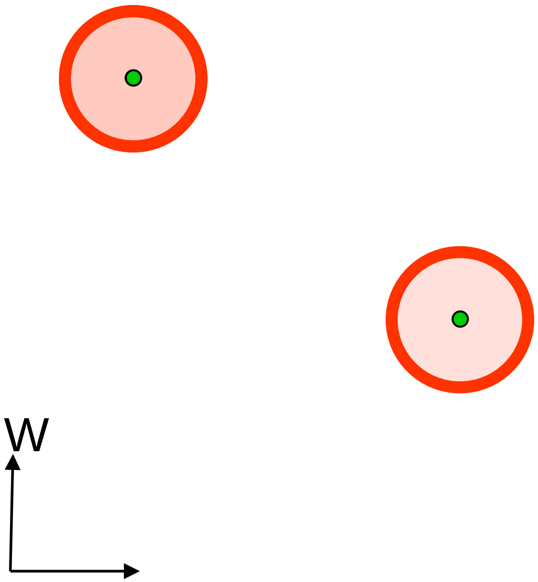
Roomba vacuum
Cleaning robot

By i-robot
Price: ~ rs. 30K

https://www.youtube.com/watch?v=dweVBqei9LA

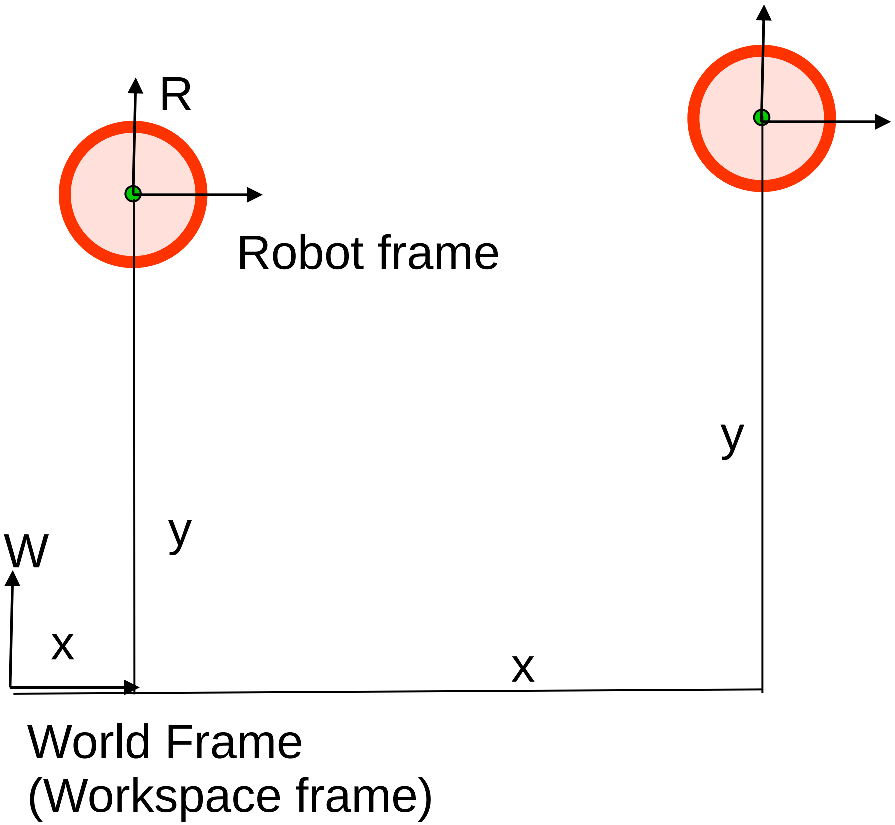
Models of Robot Motion

Circular robot



World Frame
(Workspace frame)

Models of Robot Motion



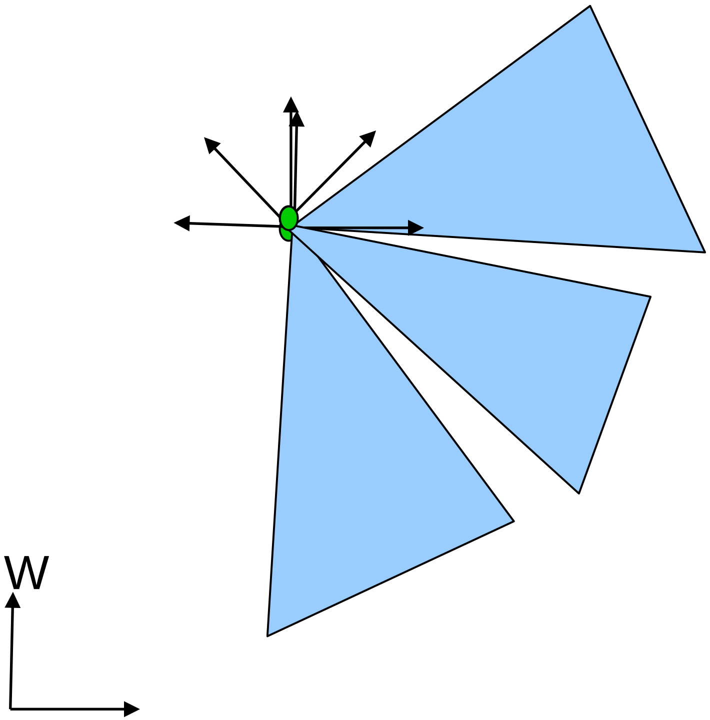
DEFINITION:
degrees of freedom:
number of parameters needed to fix the robot frame R in the world frame W

NOTE:
Given robot frame R, every point on the robot is known

(x,y) = **configuration** (vector **q**)

given configuration **q** for a certain pose of the robot, the set of points on the robot is a function of the configuration: say R(**q**)

Non-Circular Robot



DEFINITION:
degrees of freedom:
number of parameters needed to fix the robot frame R in the world frame W

How many parameters needed to fix the robot frame if it can only translate?

How many if it can rotate as well?

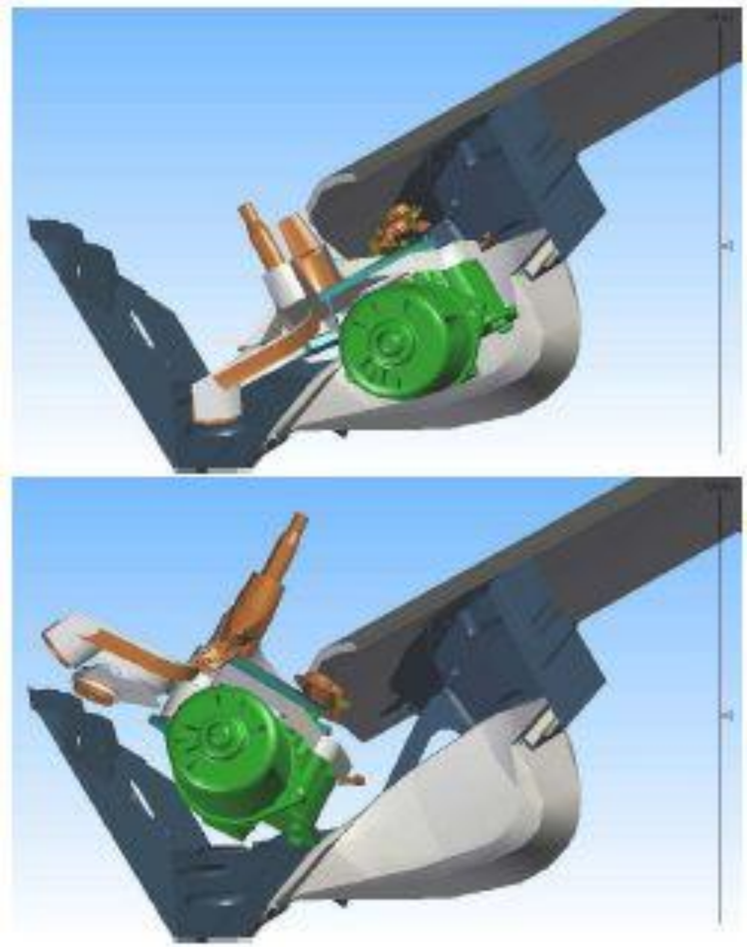
Motion in 3-D: Piano movers problem

General 3D motion:

How many parameters needed to fix the pose?

Can a design be assembled?

Test based on CAD models



Research mobile robot

Turtlebot

Based on i-robot (roomba) platform
(with kinect RGB-D sensor)

ROS (open-source) software

Price: ~ 75K

Articulated robots

What is a Robot?

Robots properties:

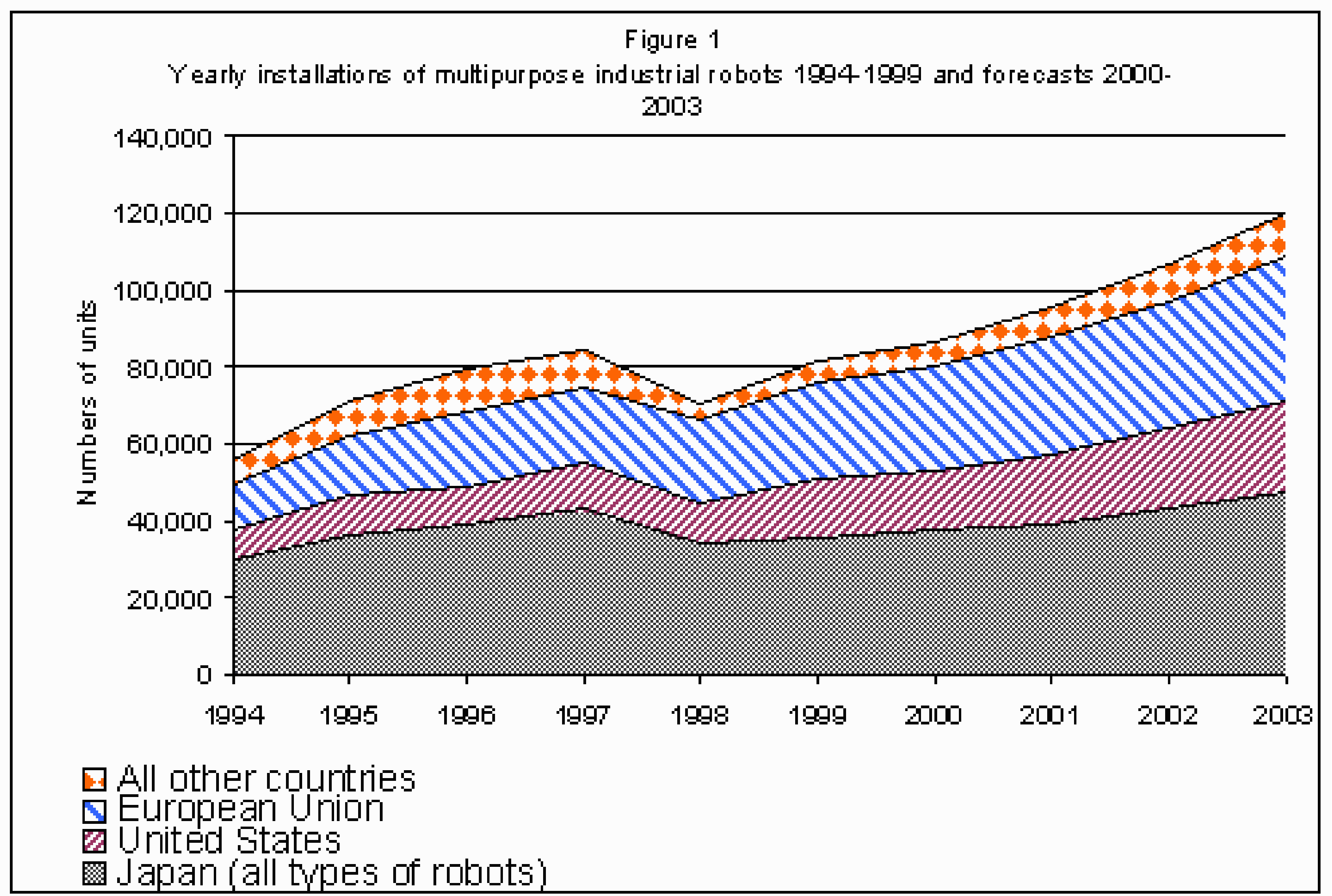
- Flexibility in Motion
  - Mobile robots
  - Articulated robots

SCARA 4-axis arm
(4 degrees-of-freedom)

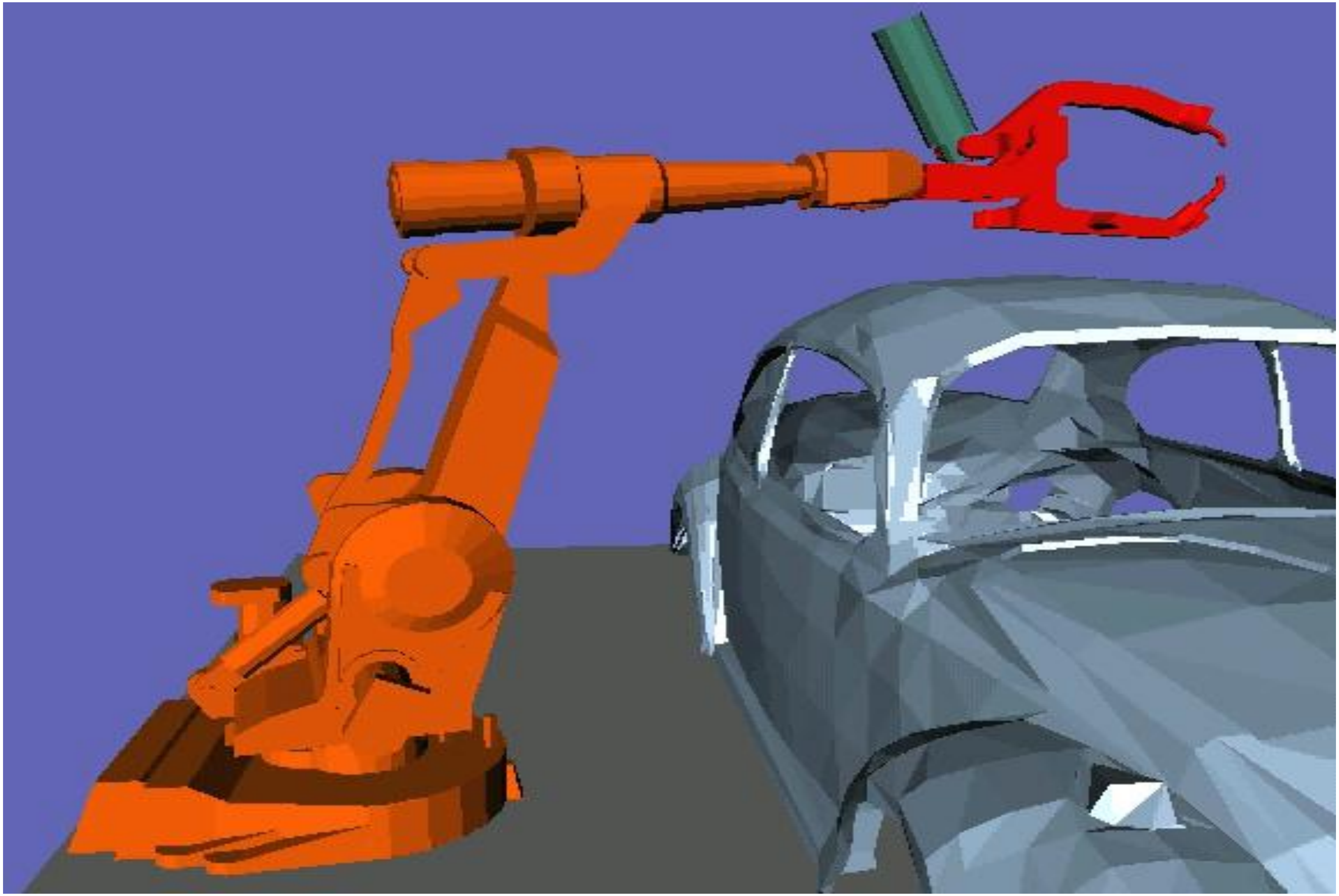
by Systemantics
Bangalore

Industrial Robot

Industrial Robots



How to program a welding robot?



What is a Robot?

Robot properties

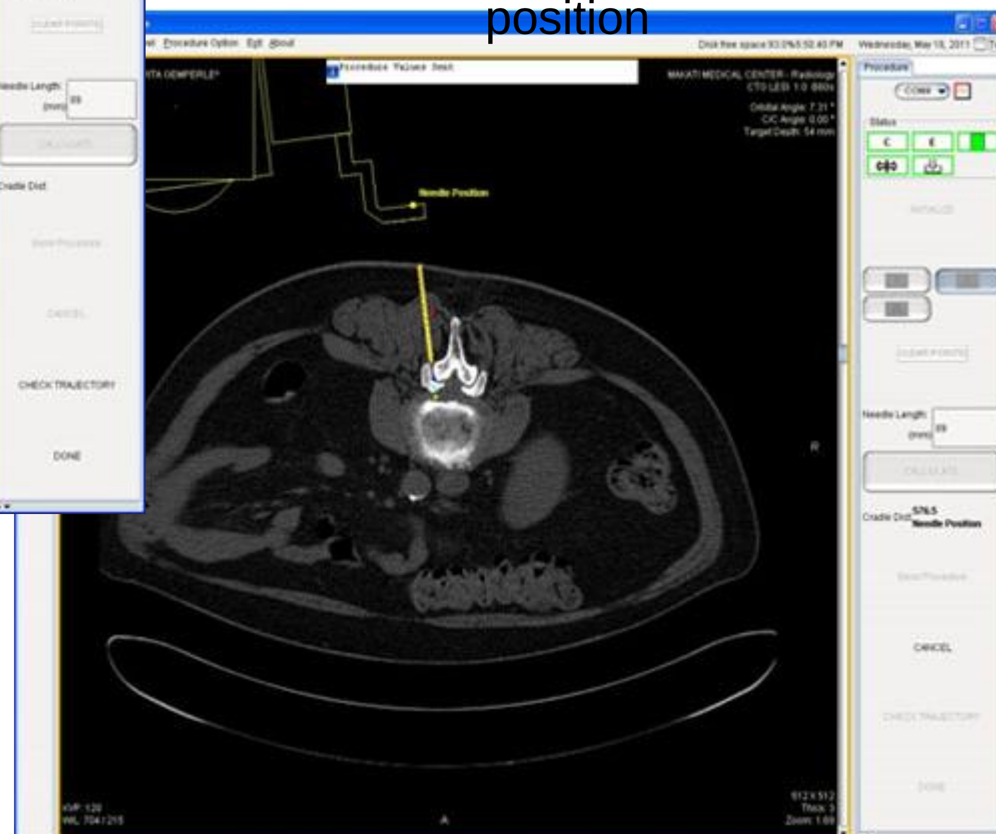
- Flexibility in M
  - Mobile robot
  - Articulated r
    - Industrial r
    - Surgical robots

perfint healthcare, chennai

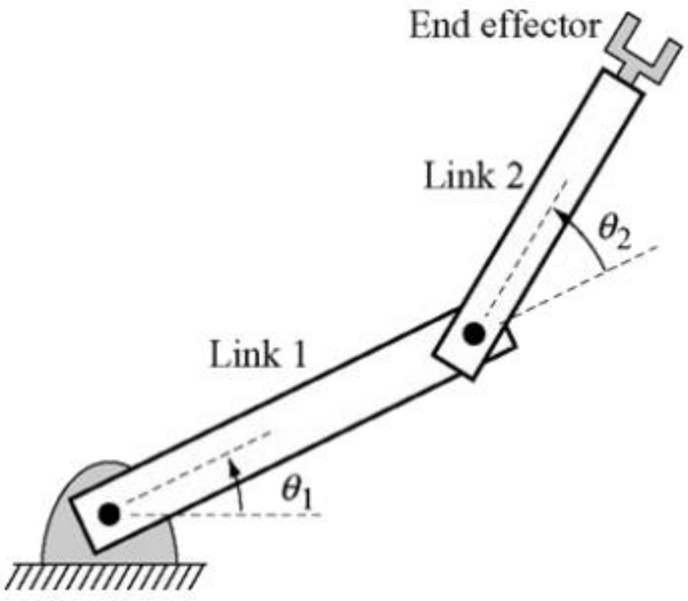
Surgical Robot : Lumbar biopsy

needle path as planned on CAT scan

inserted needle position



Modeling Articulated Robots



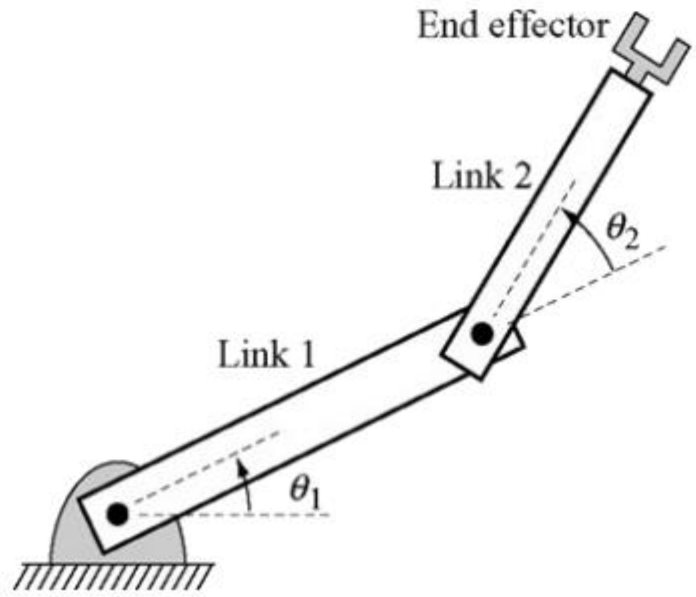
Kinematic chain:
Pose of Link n depends on the poses of Links 1...(n-1)

Transformation between frame of link (n-1) and link n, depends on a single motion parameter, say θ_n

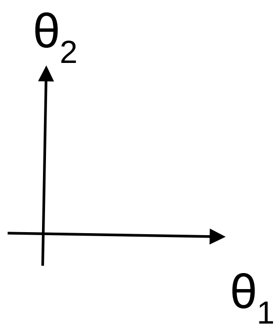
Exercise:
What are the coordinates of the orgin of the end-effector center?

Modeling Articulated Robots

workspace



configuration
space



Exercise:
Sketch the robot pose for the configuration [0, -90]

Modeling Articulated Robots

Forward kinematics

Mapping from configuration **q** to robot pose, i.e. R(**q**)

Usually, R() is the product of a sequence of transformations from frame i to frame i+1.

Note: Must be very systematic in how frames are attached to each link

Inverse kinematics

a. Given robot pose, find **q**

Or

b. Given end-effector pose, find **q**

Q. Is the answer in (b) unique?

What is a Robot?

Robots properties

- Flexibility in Motion
  - Mobile robots
  - Articulated robots
  - Digital actors

Reality: limited functionality

Mobility isnt everything

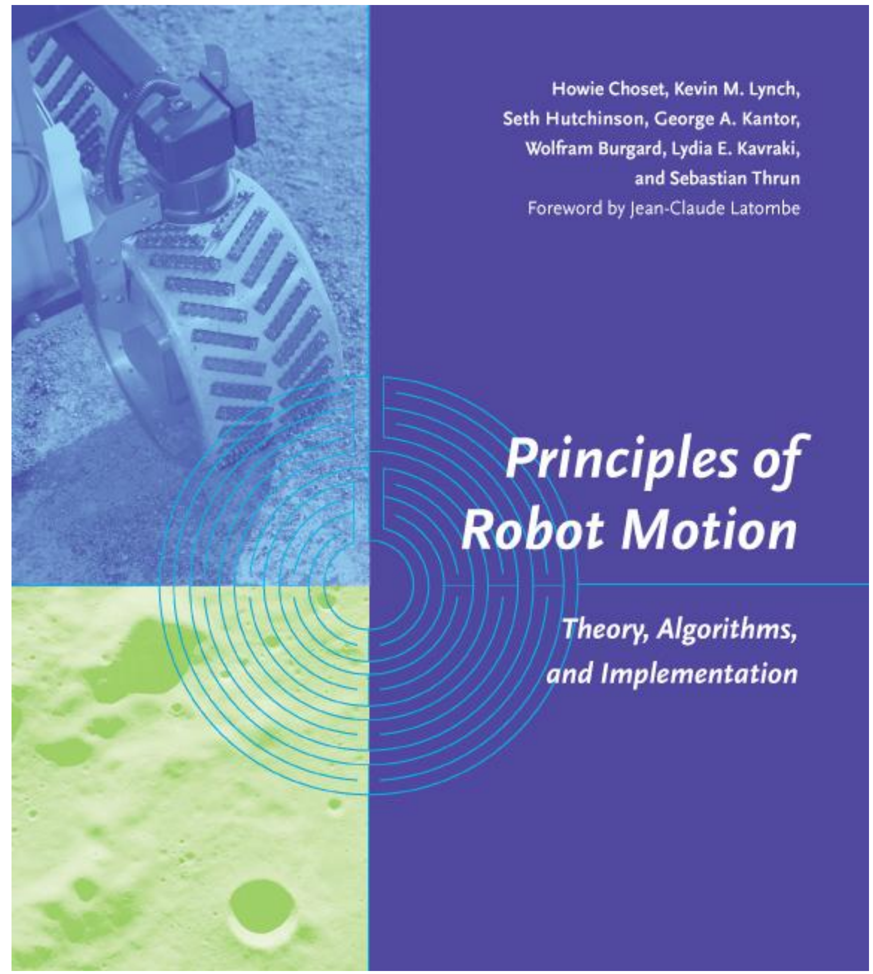
What is a Robot?

Robots involve

- Flexibility in Motion
  - Dentists cradle?
  - Washing machine?
- Intentionality
  - Measure : not default probability distribution
    - e.g. Turn-taking (contingent behaviour)
  - Goal : intrinsic or extrinsic

Robot Motion Planning

Amitabha Mukerjee
IIT Kanpur, India

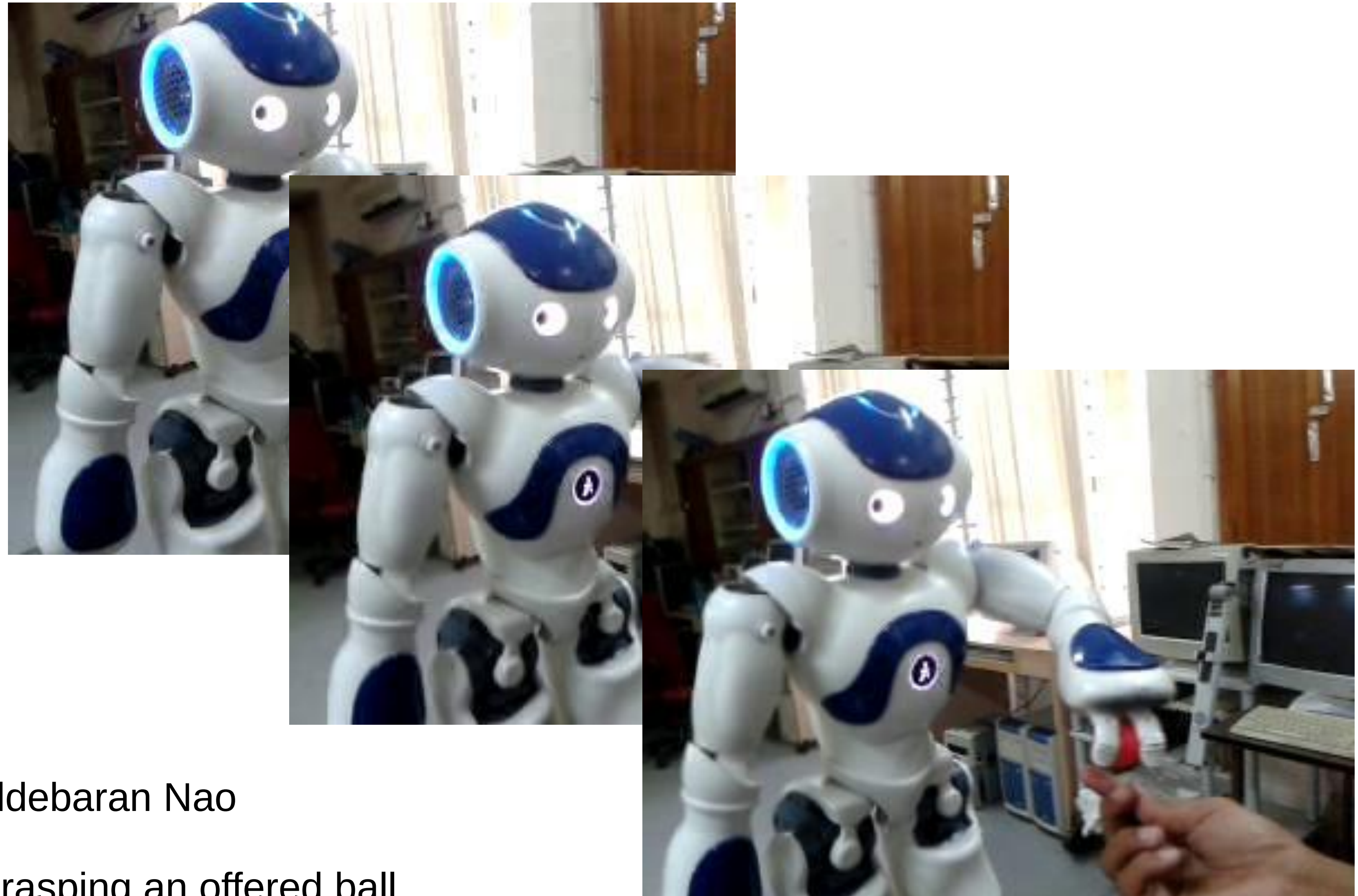


indian edition
rs 425

Sensing and Motion Planning

[bohori venkatesh singh mukerjee 05]
Bohori/Venkatesh/Singh/Mukerjee:2005

Programming a robot



Aldebaran Nao

Grasping an offered ball

Programming a robot

1. detect ball using colour:

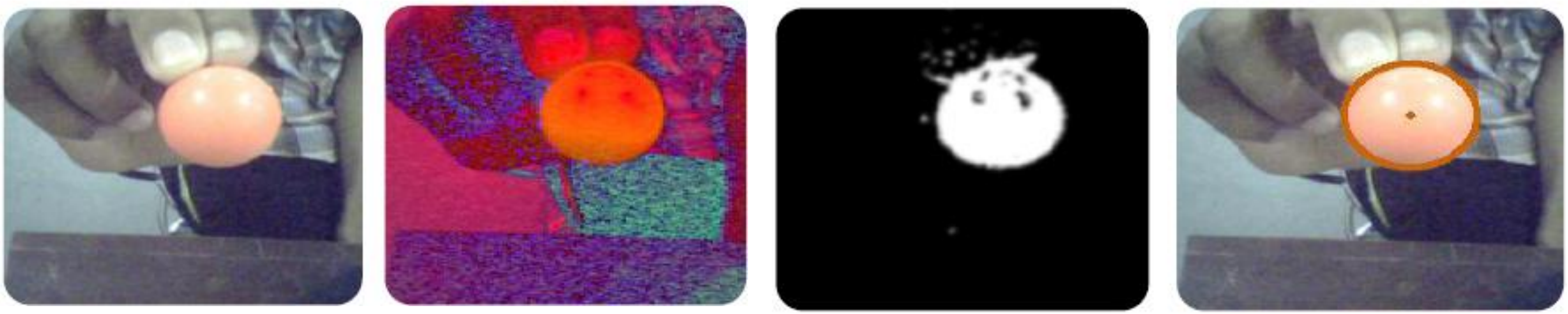


image captured by nao HSV binarized contour detected

2. estimate distance of ball (depth) from image size

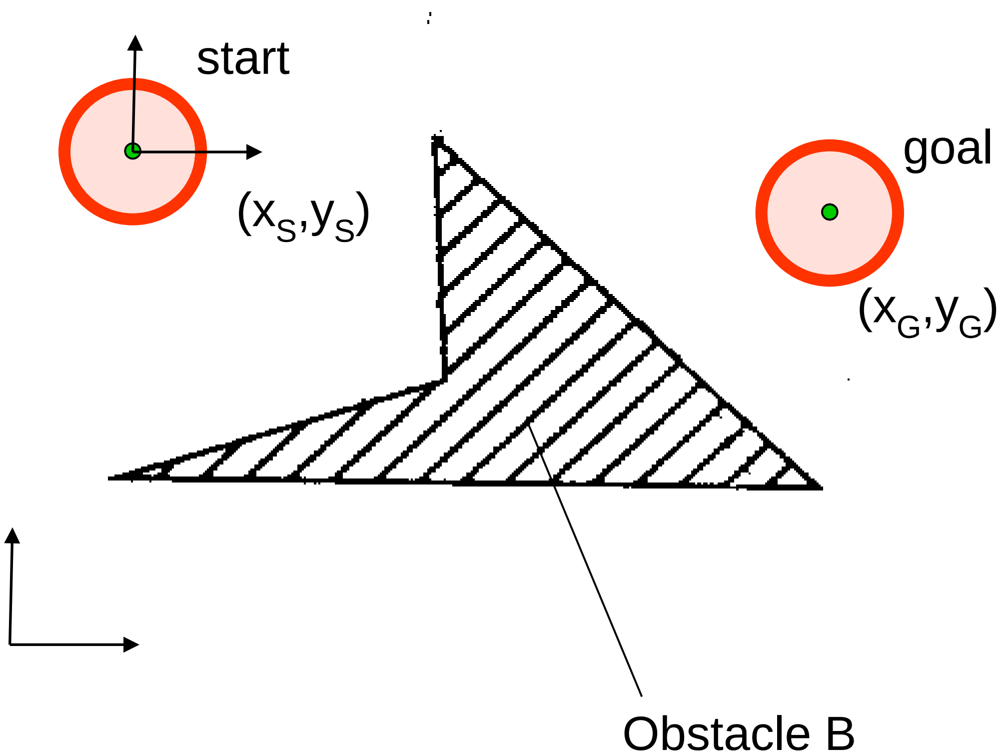
3. Inverse kinematics to grasp ball

Sensing in the workspace

Motion planning in C-space

Configuration Space

Robot Motion Planning



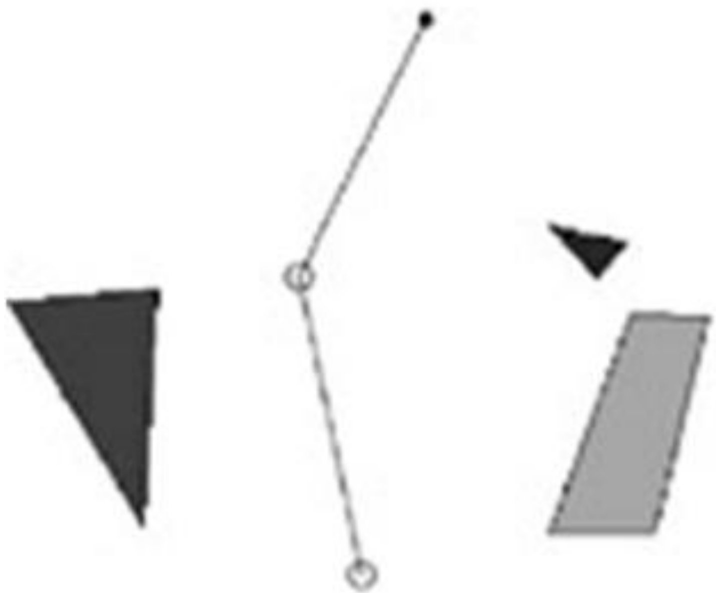
Valid paths will lie among those where the robot does not hit the obstacle

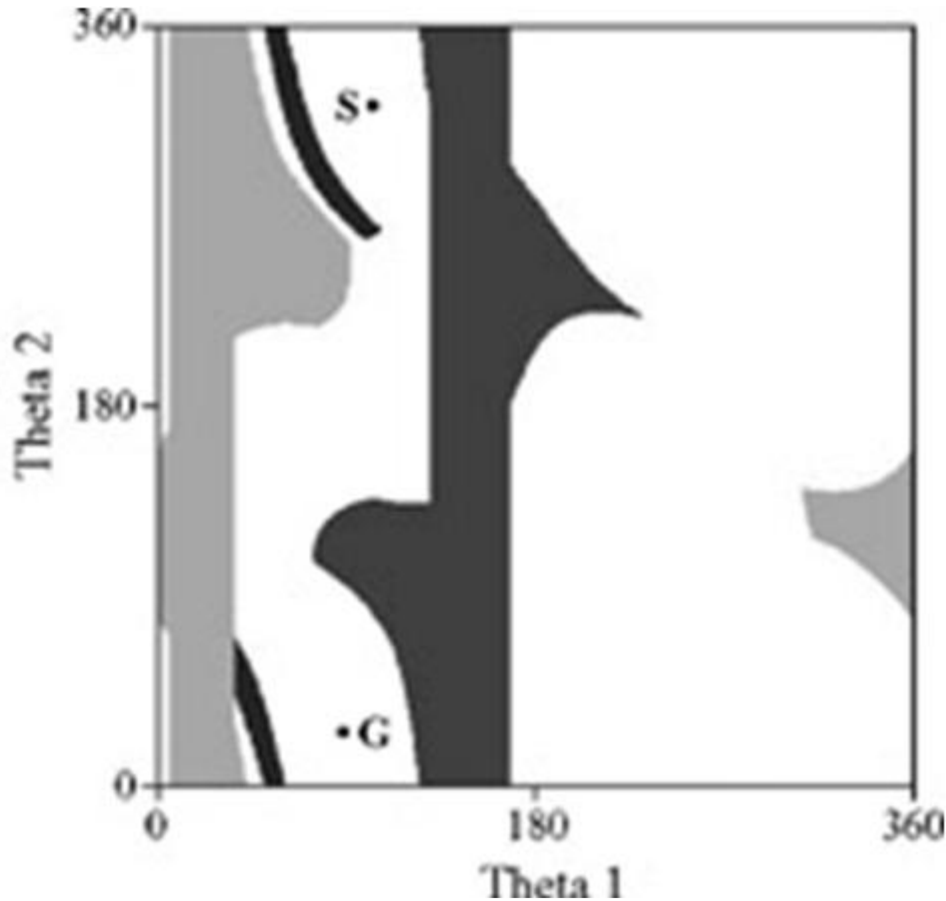
find path *P* from start to goal s.t.

for all *t*, $R(t) \cap B = \emptyset$

How to characterize the set of poses for which the robot does not hit the obstacle B?

Robot Motion Planning





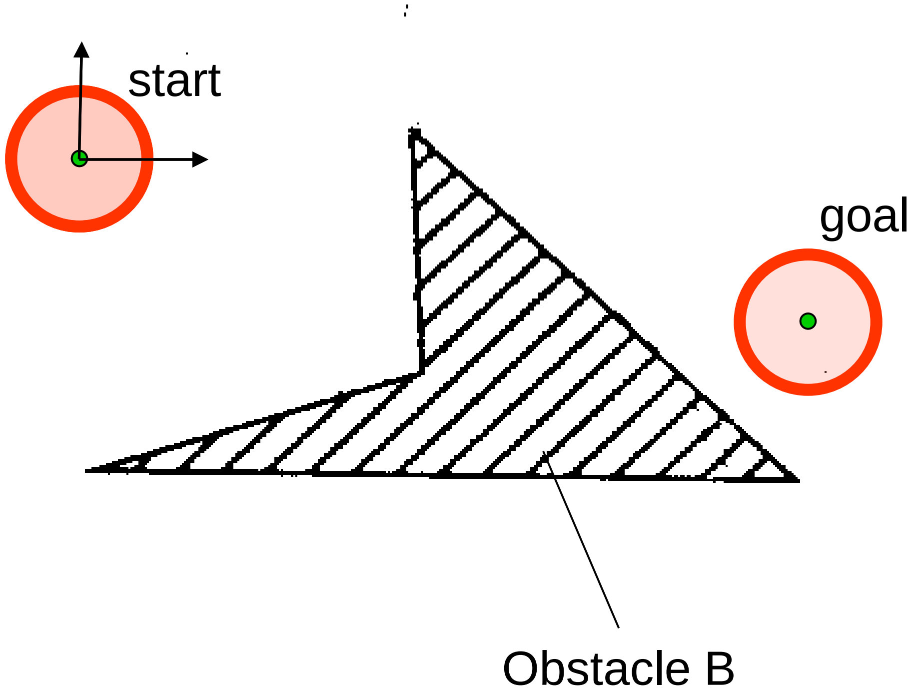
Continuum approaches vs Discretization

Two approaches to Robot motion planning:

- **continuum**:
  treat motion space as single continuum
  → optimization

- **discretization**:
  decompose motion space into regions / segments
  → graph-search

Potential fields



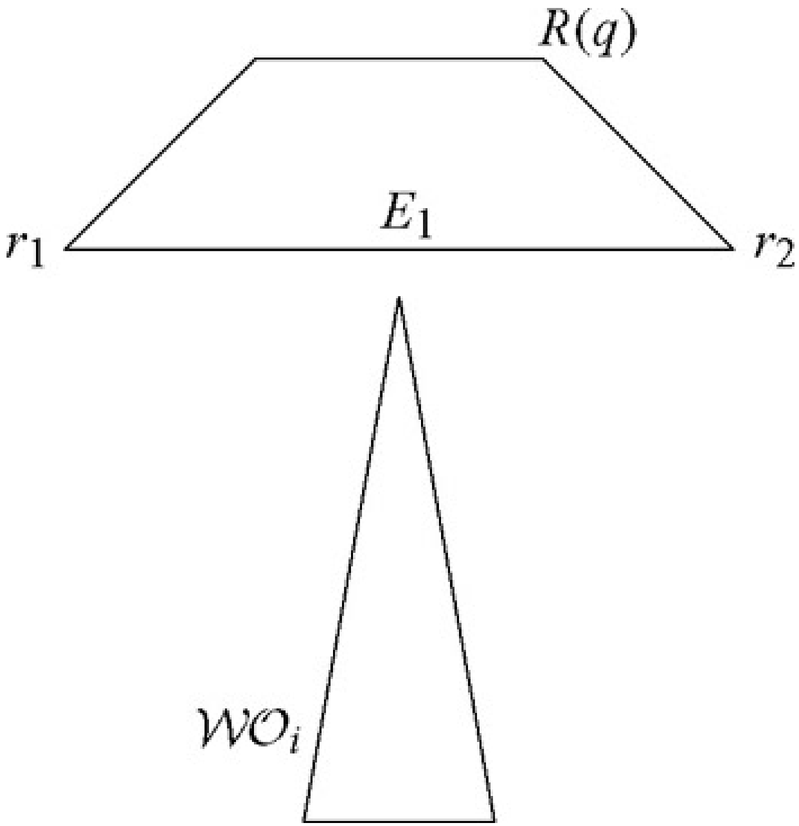
Potential fields

1. Goal: negative (attractive) potential
   Obstacles: positive (repulsive) potential
2. Robot moves along gradient
3. Problems:

   - need to integrate the potential over the area of robot

   - problem of local minima

Finite area robots



Instead of integrating over robot area, restrict to a set of *control* points

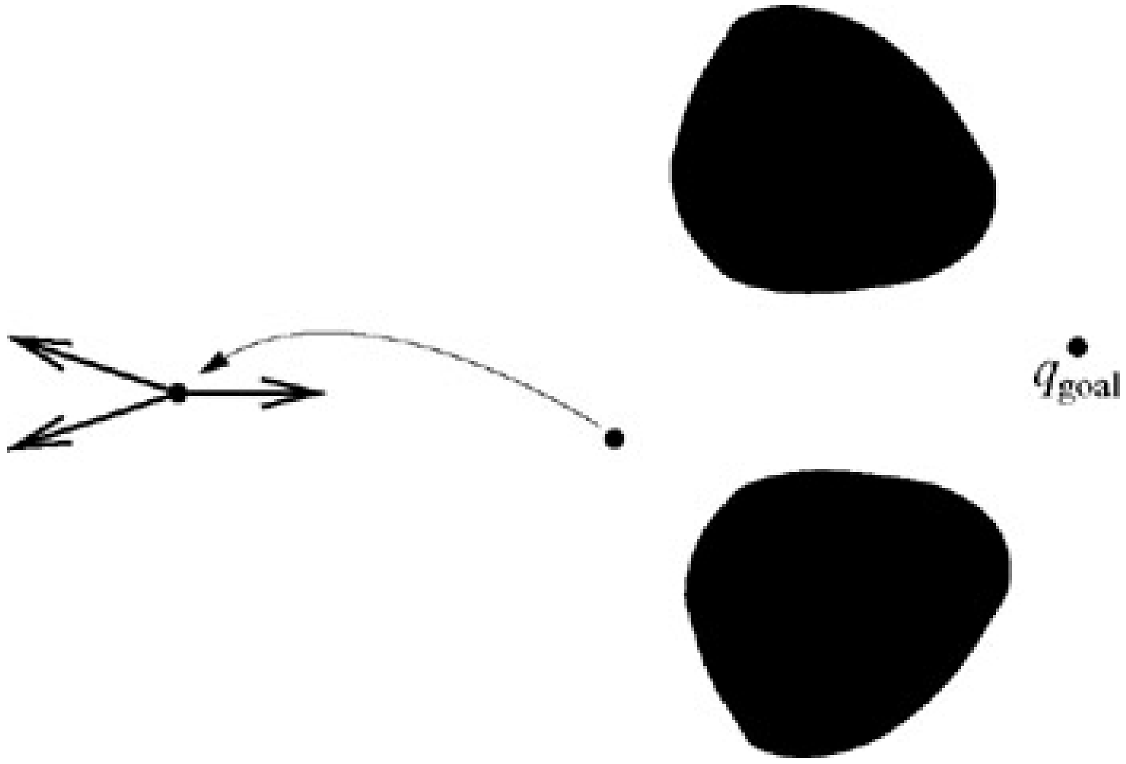
e.g. vertices

Problem:

With control points r1 and r2 on robot R(q), edge E1 may still hit Obstacle.

→ Attempt to reduce computation to points

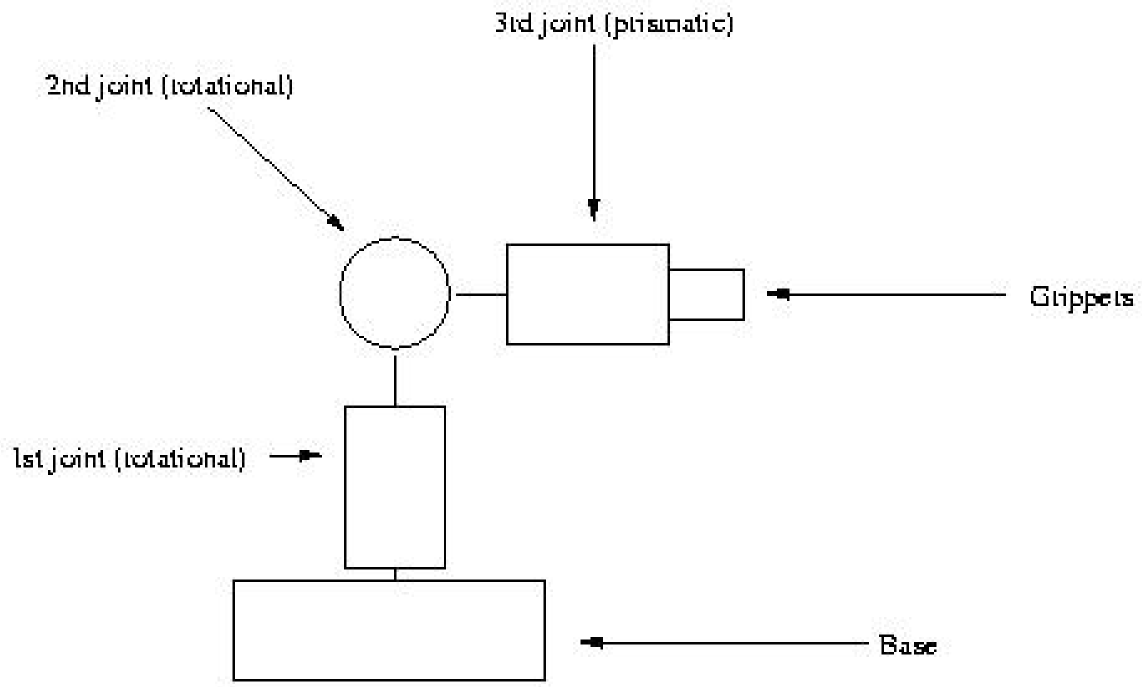
Local Minima



persists even for point robots

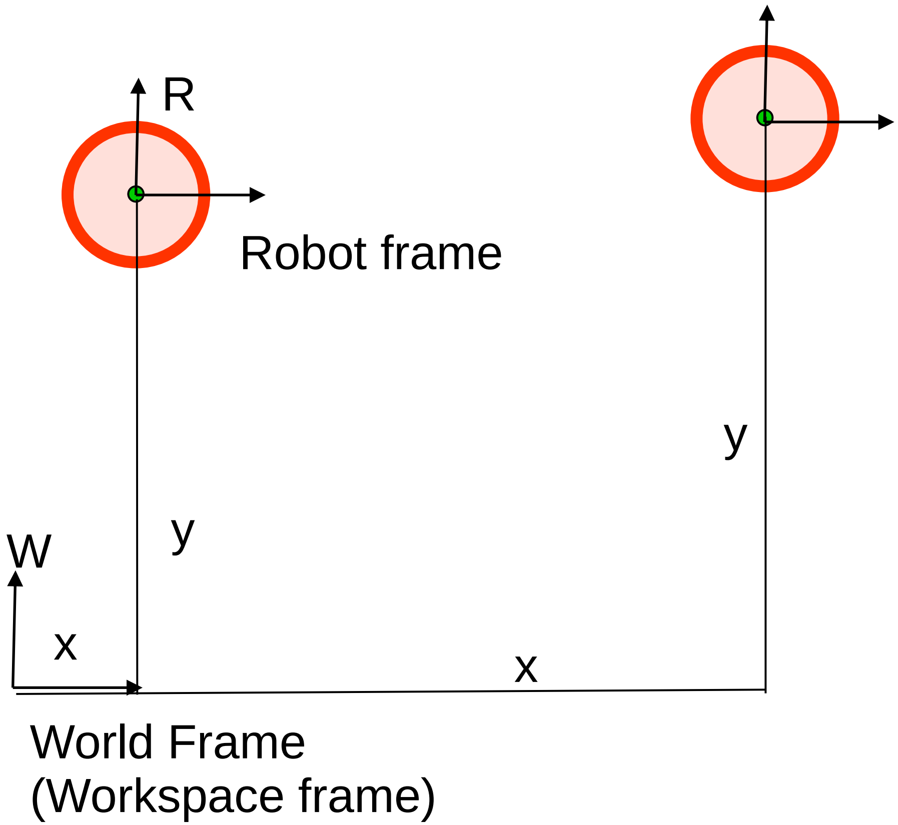
Nature of Configuration spaces

Robot Model



3rd joint (prismatic)
2nd joint (rotational)
Grippers
1st joint (rotational)
Base

Models of Robot Motion

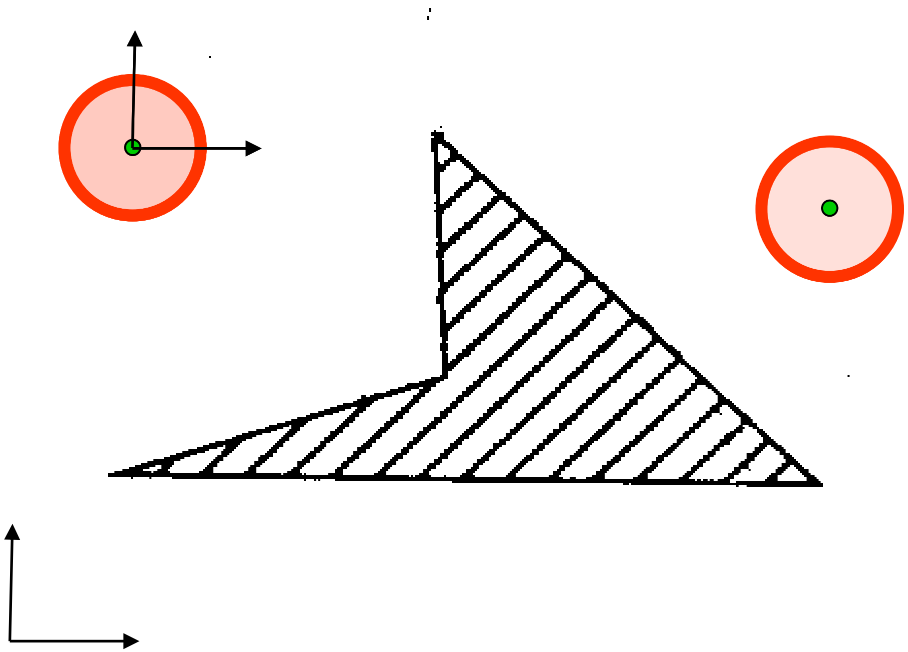


DEFINITION:
degrees of freedom:
number of parameters needed to fix the robot frame R in the world frame W

(x,y) = **configuration**
(vector **q**)

given configuration **q** for a certain pose of the robot, the set of points on the robot is a function of the configuration: say R(**q**)

Robot Motion Planning

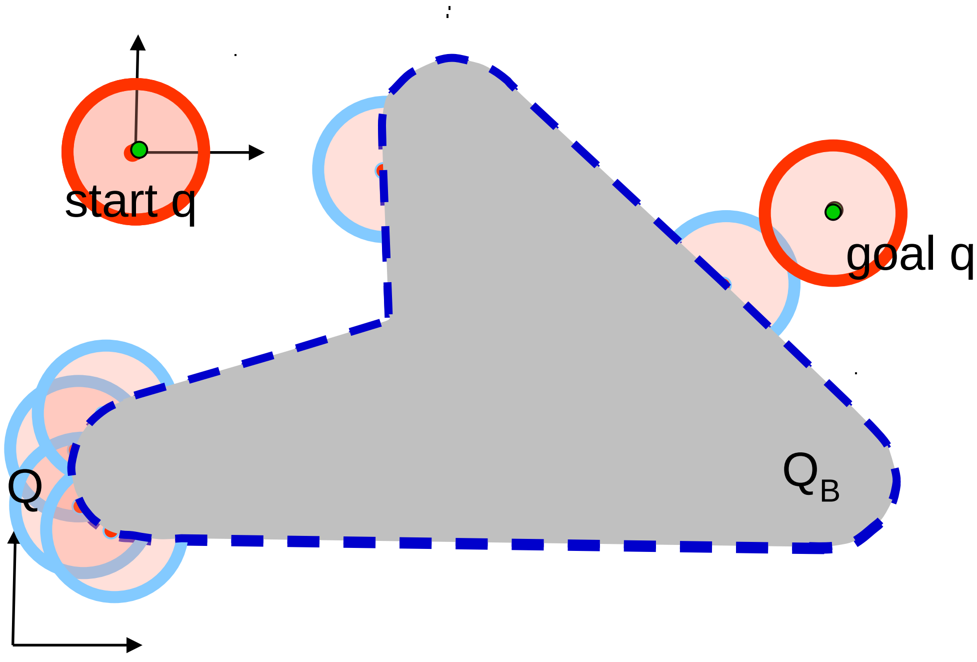


find path P from qS to qG s.t. for all q ϵ P, R(q) ∩ B = Ø

? generate paths and check each point on every path?

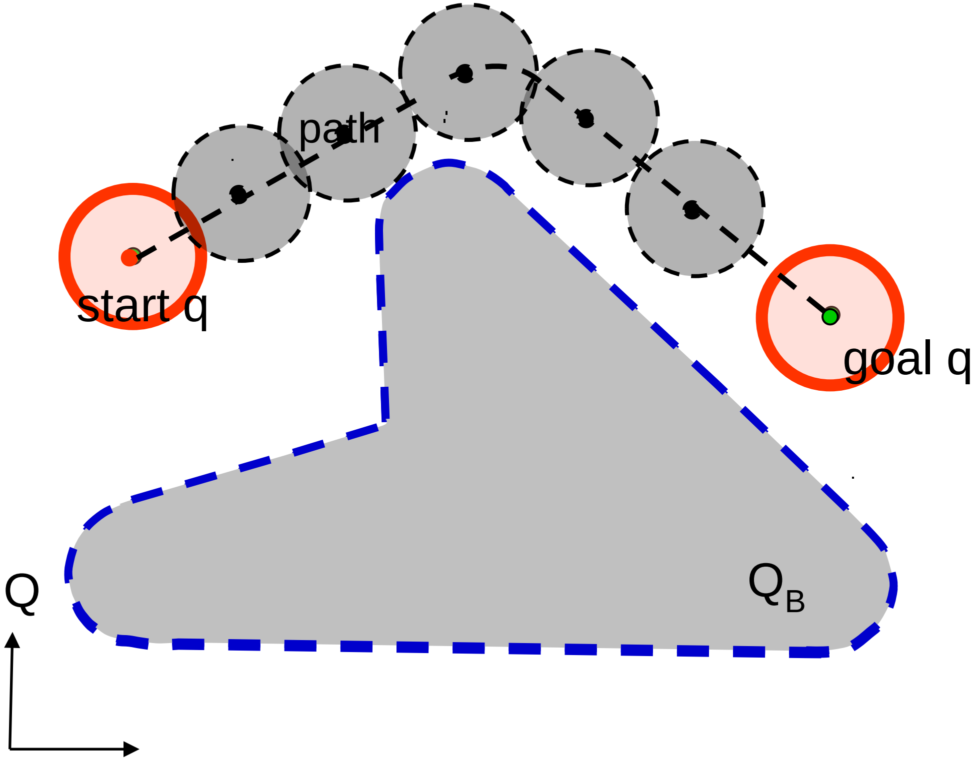
Would it be easier to identify Q_{free} first?

Robot Motion Planning



Q_B = [**q** | R(**q**) ∩ B ≠ Ø }

Motion Planning in C-space

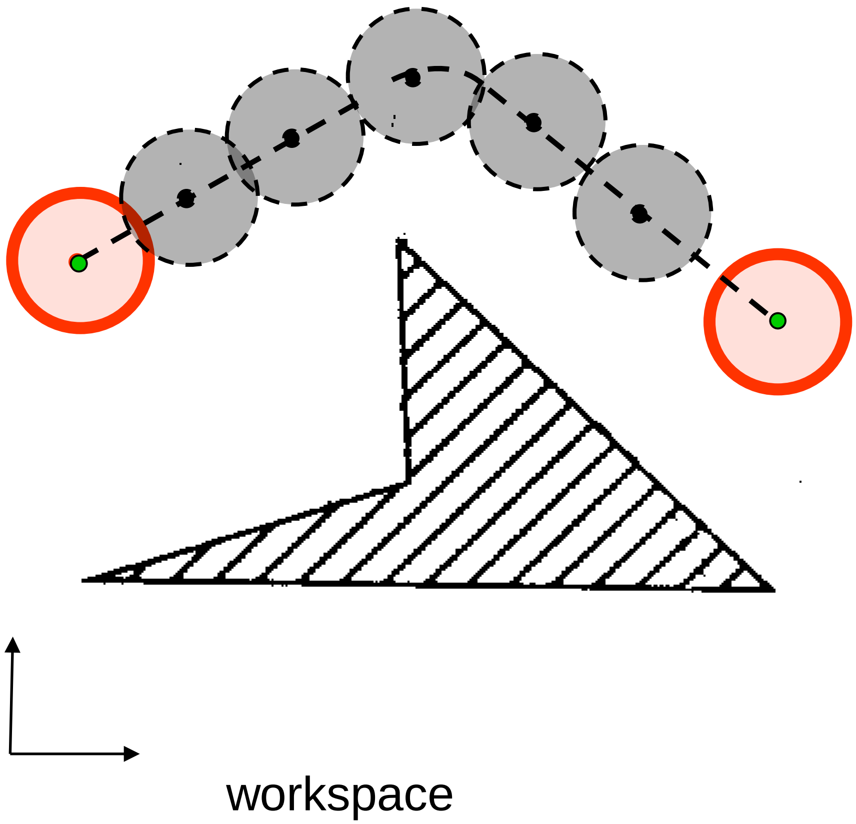


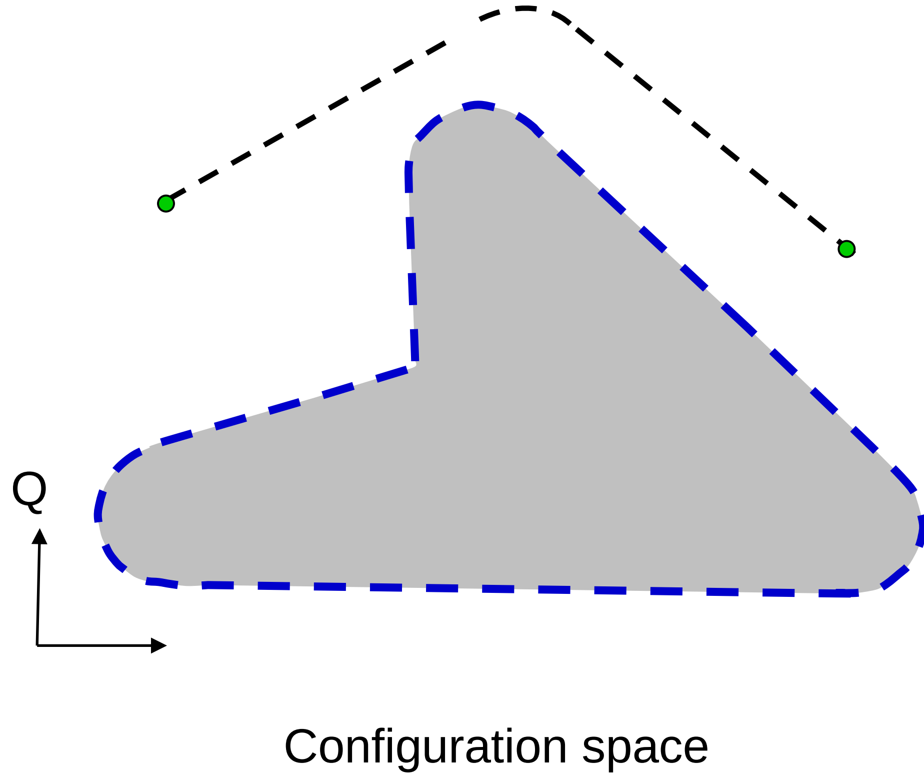
configurations are points in C-space

path P is a line

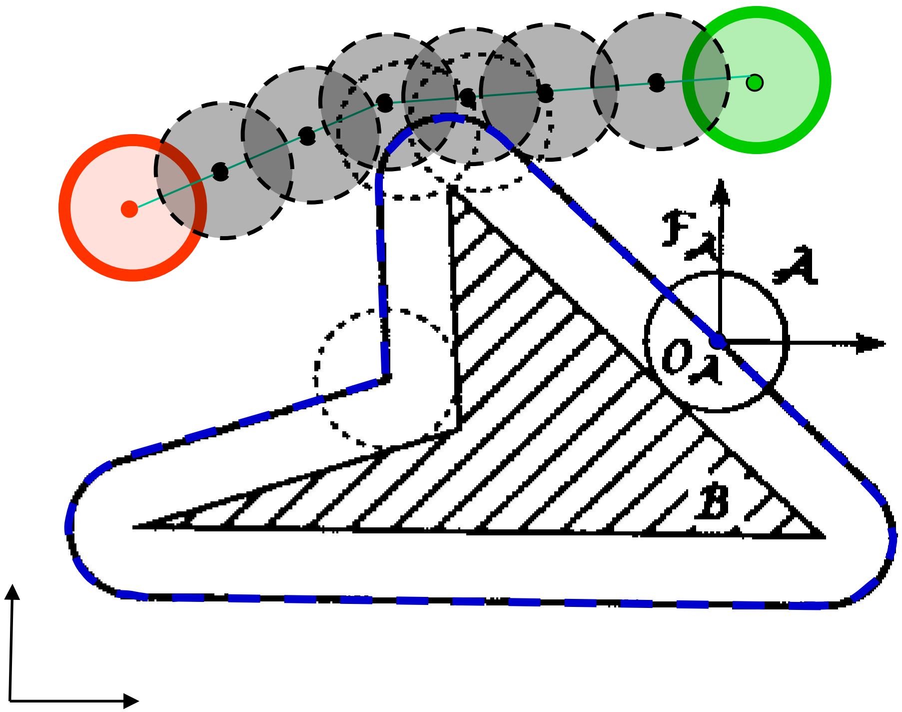
if $P \cap Q_B = \emptyset$, then path is in Q_{free}

Motion Planning in C-space

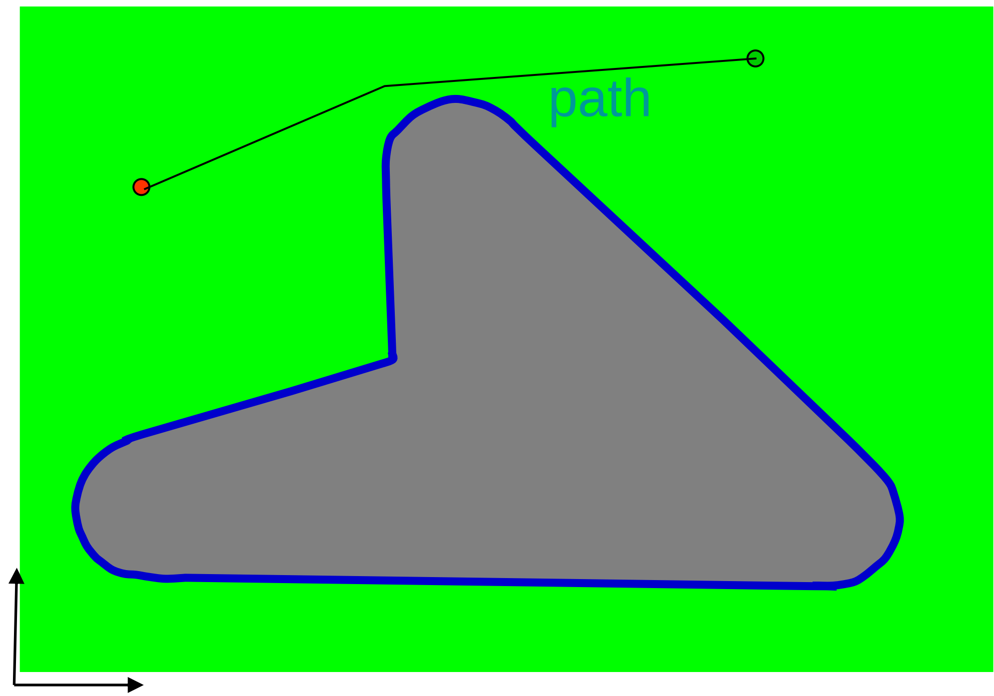




Robot Motion Planning

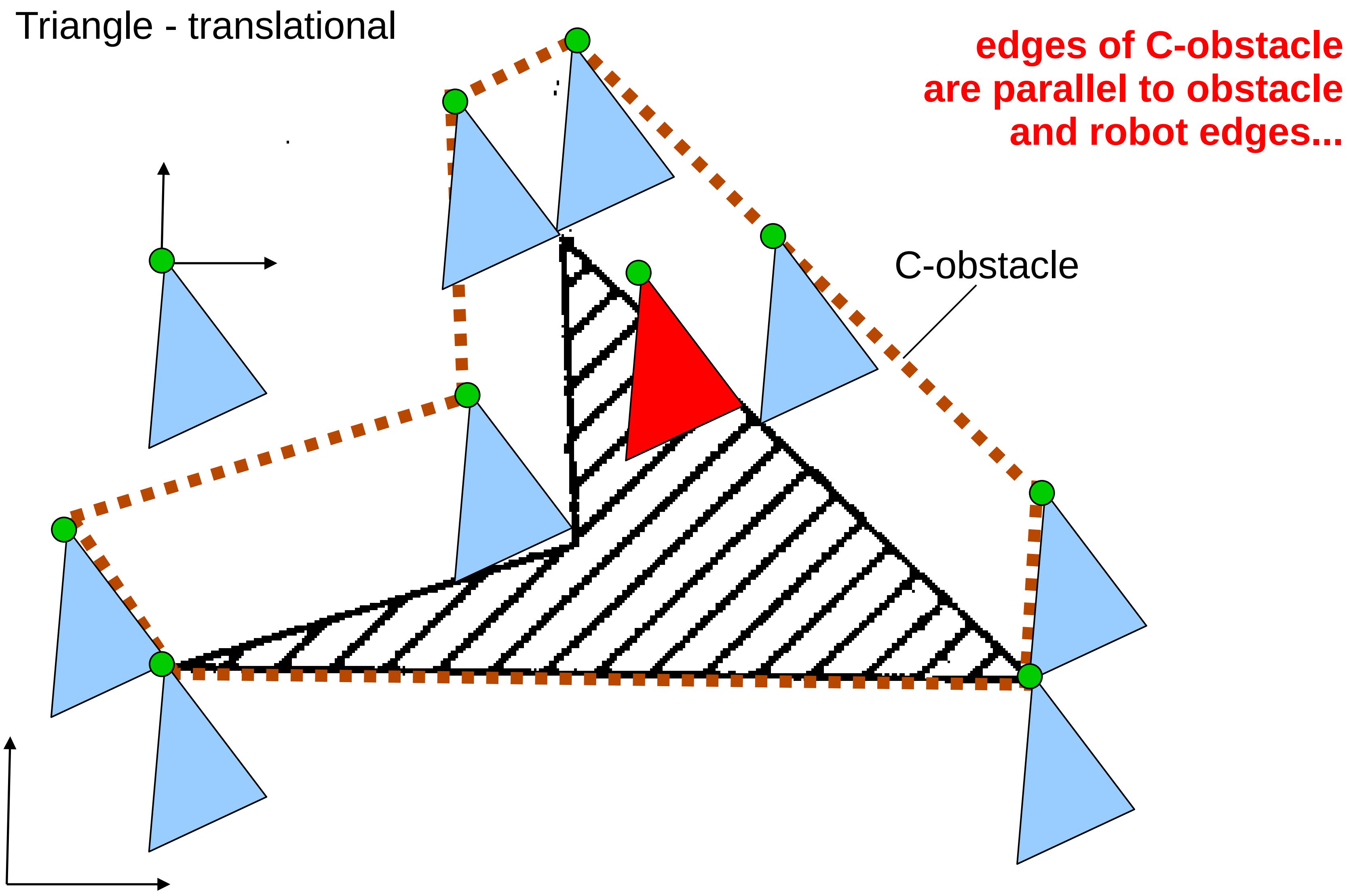


workspace
W



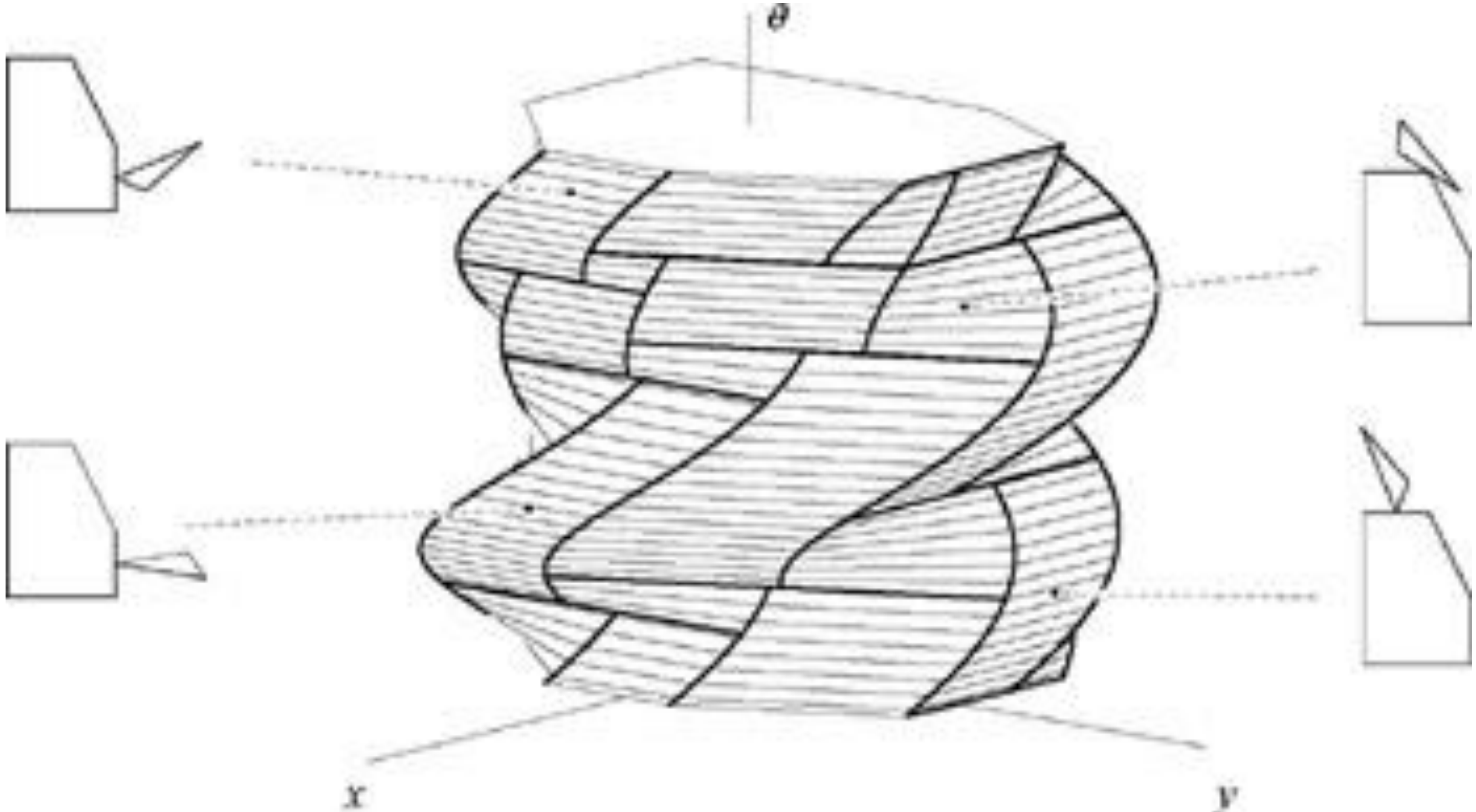
configuration space
C

Non-circular mobile robots

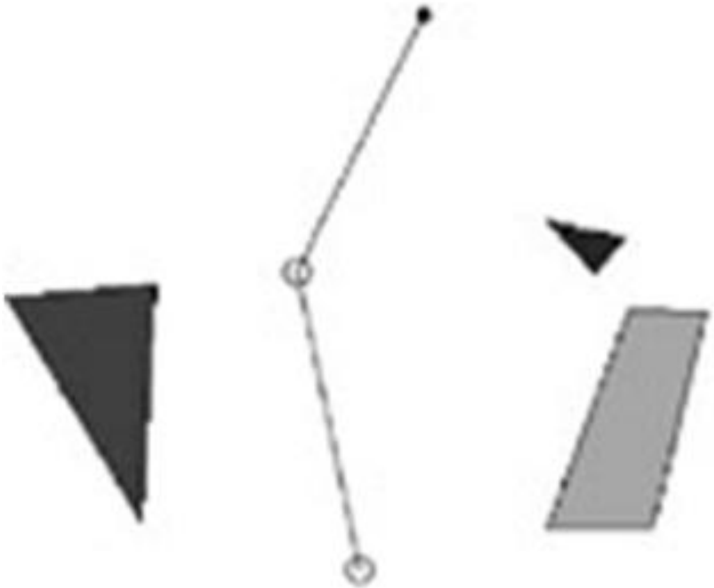


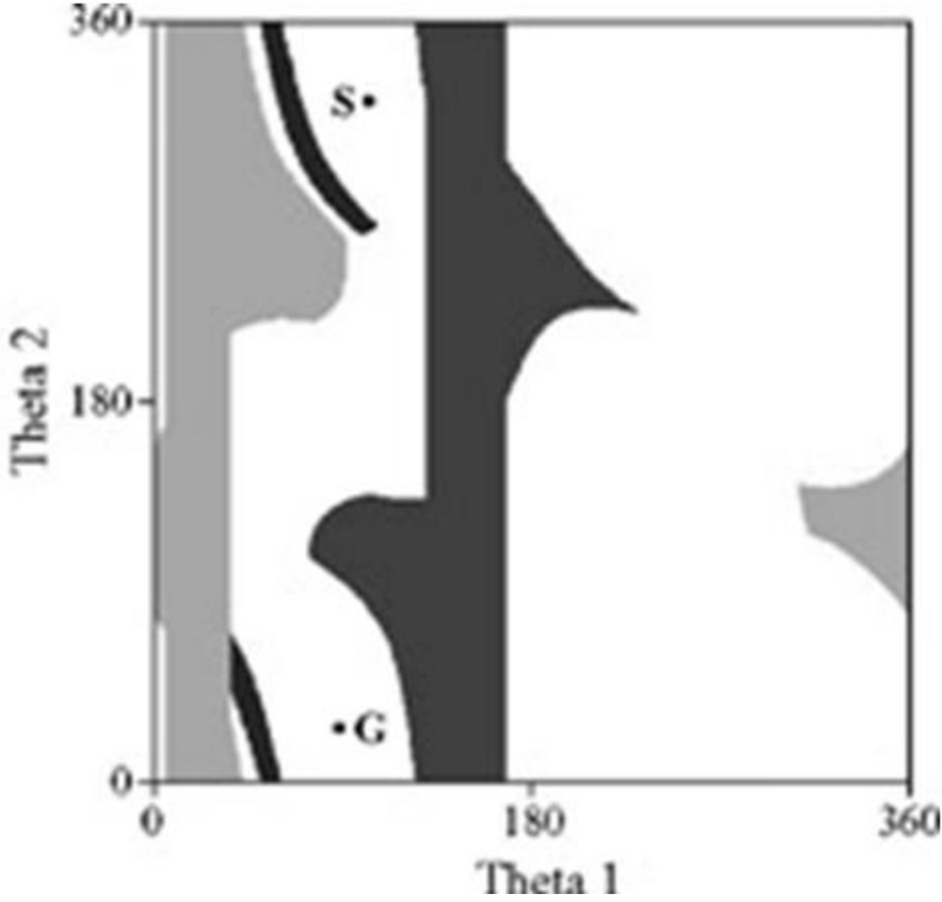
Non-circular mobile robots

C-space with rotation θ (polygonal obstacle)



Configuration Space for Articulated Robots





Configuration Space Analysis

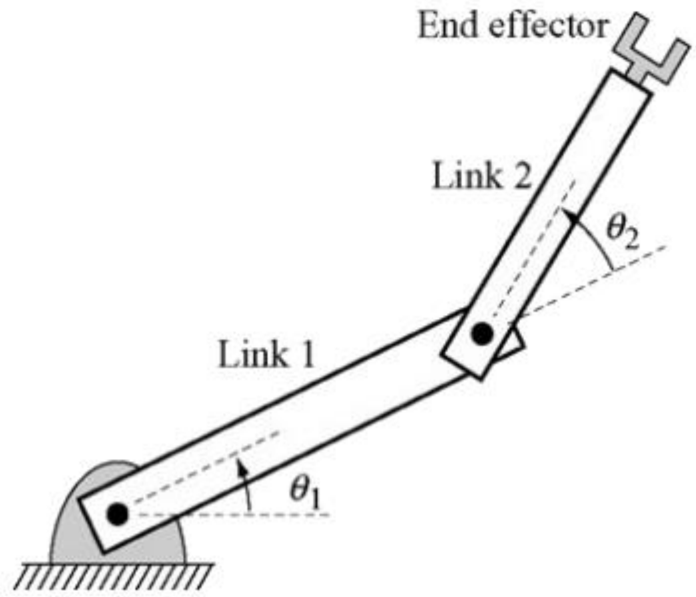
Basic steps (for ANY constrained motion system):

1. determine degrees of freedom (DOF)
2. assign a set of configuration parameters **q**

   e.g. for mobile robots, fix a frame on the robot
3. identify the mapping $R : Q \rightarrow W$, i.e. R(**q**) is the set of points occupied by the robot in configuration **q**
4. For any **q** and given obstacle B, can determine if $R(\mathbf{q}) \cap B = \emptyset$. → can identify Q_{free}

   Main benefit: The search can be done for a point
5. However, computation of C-spaces is not needed in practice; primarily a conceptual tool.

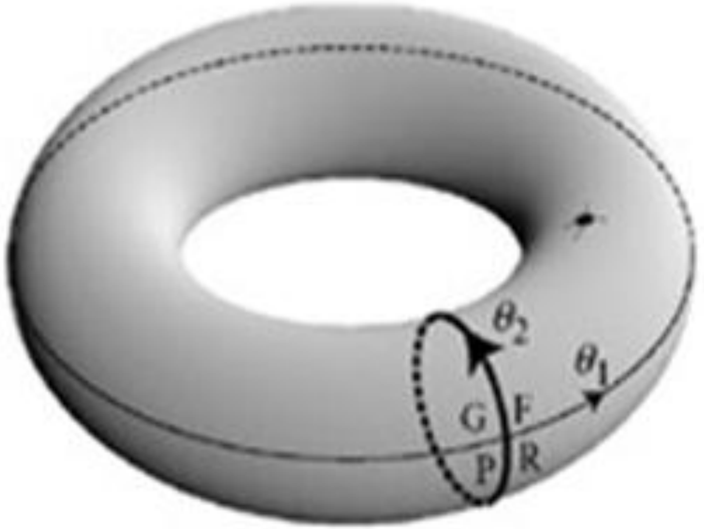
Articulated Robot C-space



How many parameters needed to fix the robot pose ?

What may be one assignment for the configuration parameters?

C-space as manifolds



Topology of C-space: Torus (S1 x S1)

Choset, H etal 2007, Principles of robot motion: Theory, algorithms, and implementations, chapter 3

C-space as manifolds

- **manifold**: generalization of curves / surfaces

  every point on manifold has a neighbourhood homeomorphic to an open set in R^n

- Mapping Φ : S→ T is bijective (covers all of T and mapping is unique)

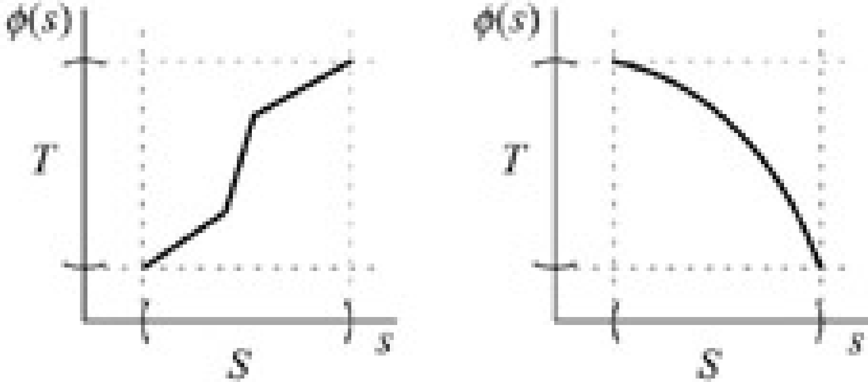
  Φ is

  homeomorphic:

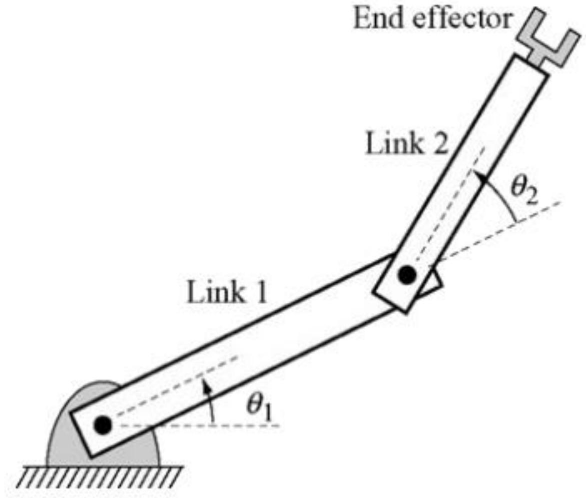
  (f / f^{-1} are continuous)

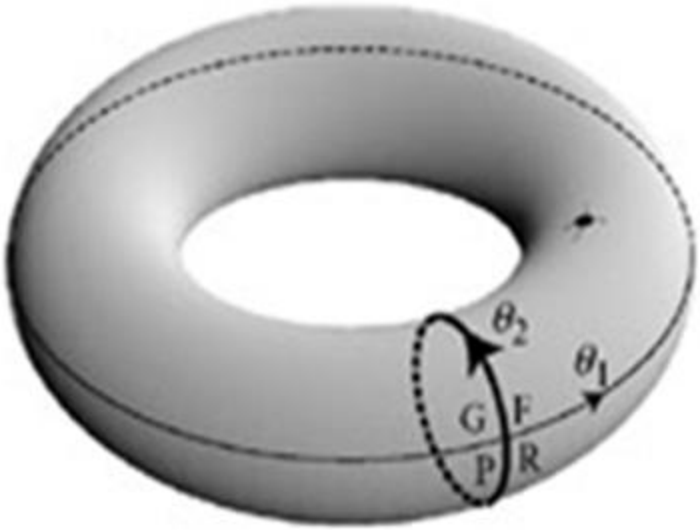
  diffeomorphic :

  (f / f^{-1} are C^∞ smooth)



C-space as manifolds





Neighbourhood of q is mappable to R2

global topology is not R2 but S1 x S1 (torus)

Map from C-space to W

Given configuration **q**, determine volume occupied by R(**q**) in workspace

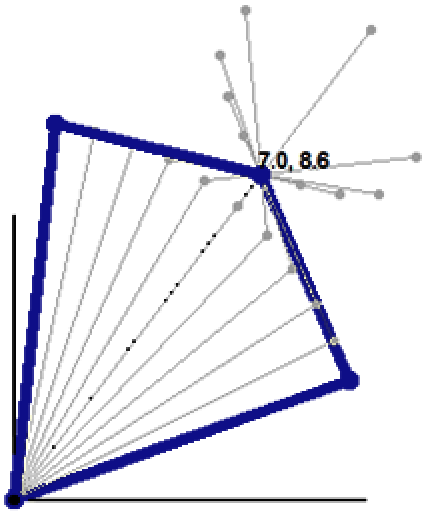
For multi-link manipulators, spatial pose of link (n+1) depends on joint configuration **q** for joints 1, 2, ..., n.

→ Forward Kinematics

Map from W to C-space: given pose in workspace, find **q**

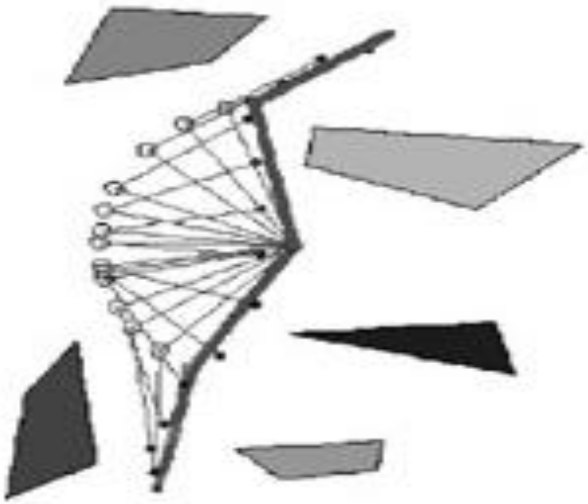
→ Inverse Kinematics

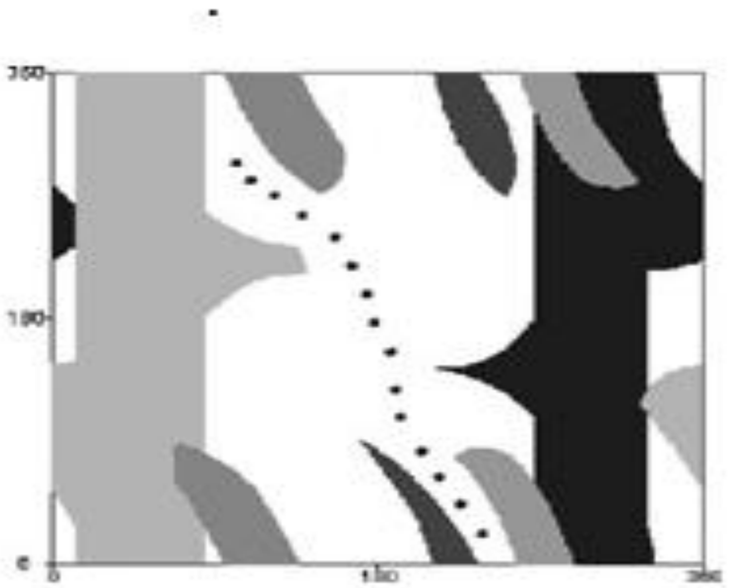
Mapping obstacles



Point obstacle in
workspace

Articulated Robot C-space



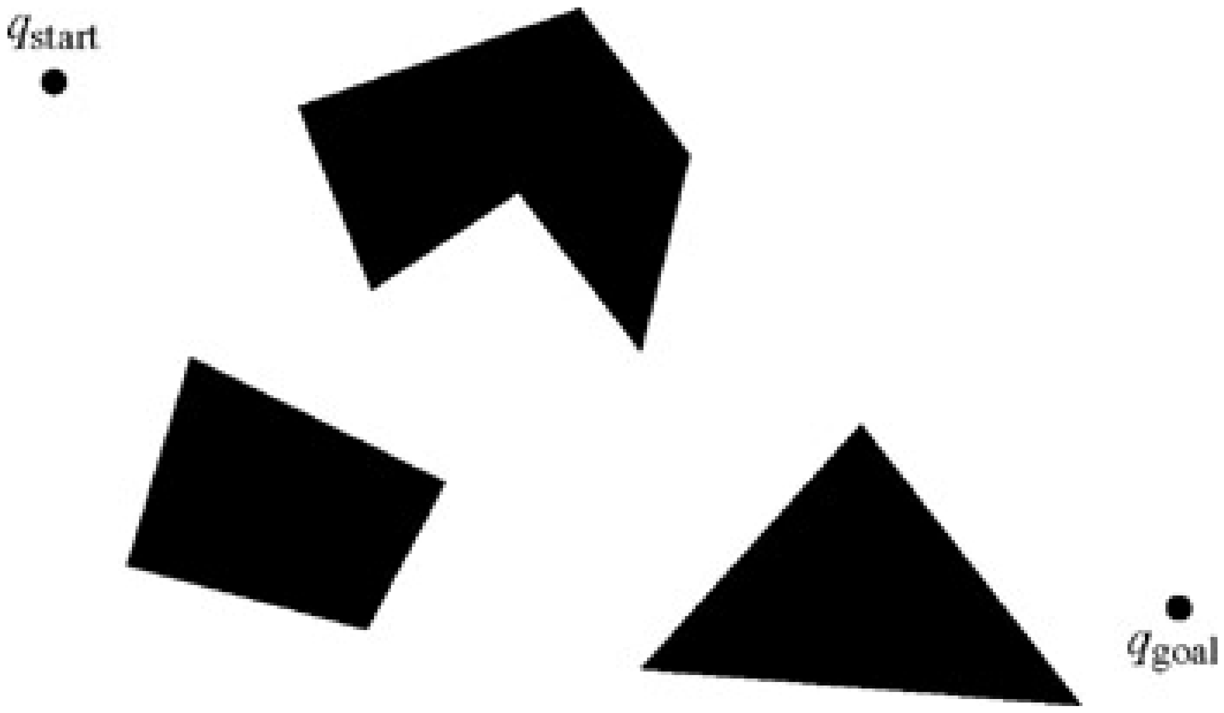


Path in workspace

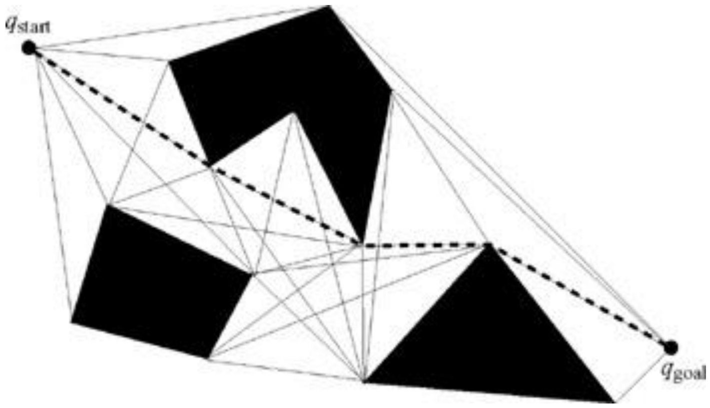
Path in Configuration Space

Graph-based approaches

Visibility Graph methods



Visibility Graph methods



Construct edges between visible vertices

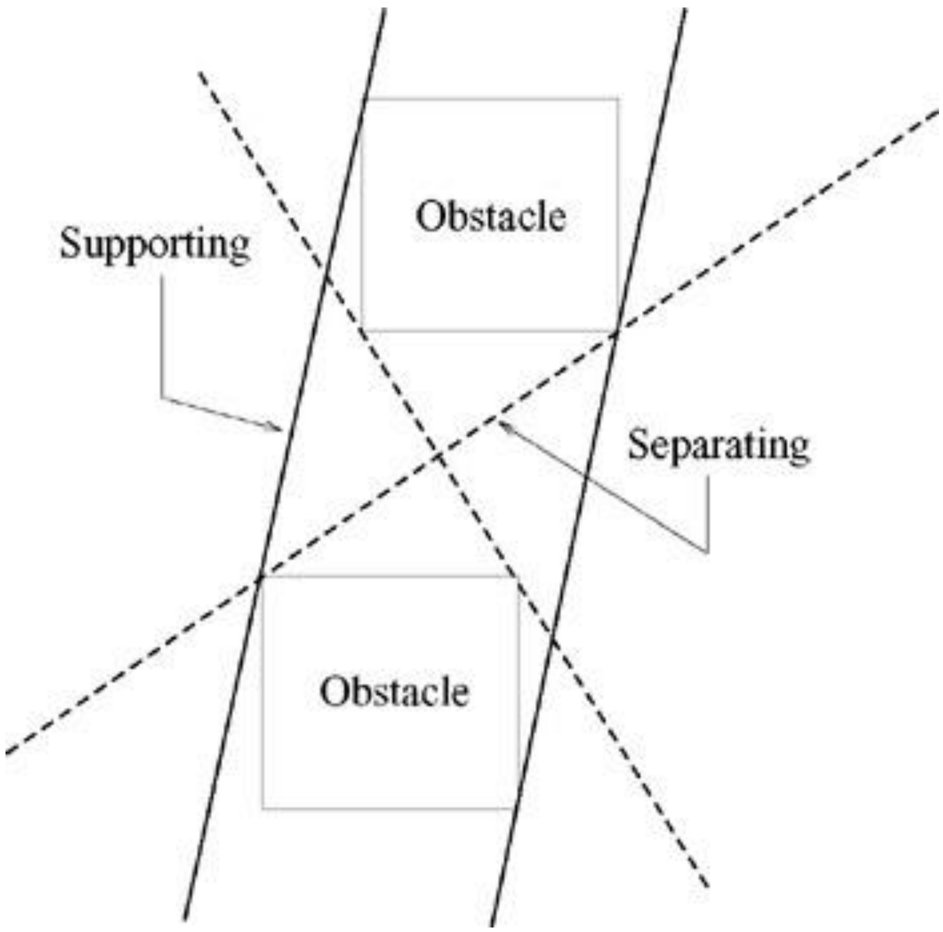
Sufficient to use only supporting and separating tangents

Complexity:

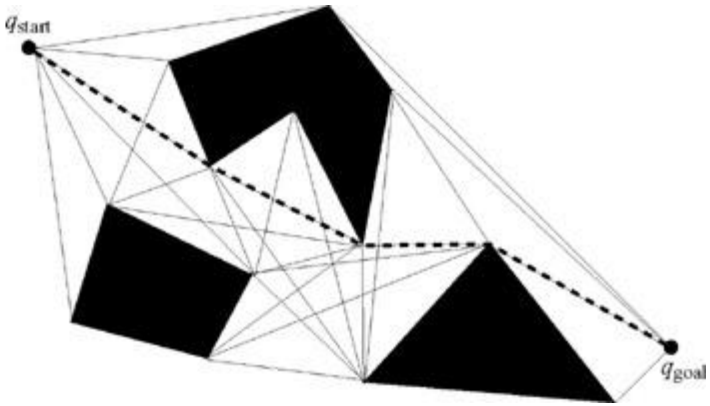
Direct visibility test: $O(n^3)$
(tests for each vtx: O(n) emanations
x O(n) obst edges)

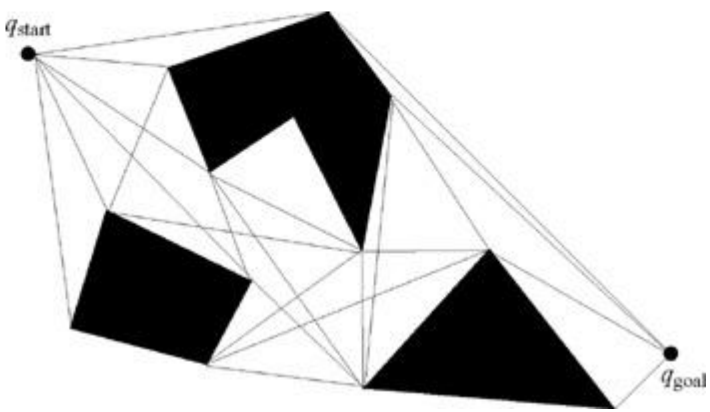
Plane sweep algorithm: $O(n^2 \log n)$

Visibility Graph methods



Visibility Graph methods





Sufficient to use only supporting and separating tangents

Finds “shortest” path – but too close to obstacles

Cell decomposition methods

Trapezoidal decomposition:
Each cell is convex.

Sweep line construction:
O(nlogn)

Graphsearch: O(nlogn)

Path: avoids obstacle
boundary but has high
curvature bends

Roadmap methods

Roadmaps

any roadmap RM must have three properties:

Connectivity:
path exists between any q'_{START} and q'_{GOAL} in RM

Accessibility:
exists a path from any $q_{START} \in Q_{free}$ to some $q'_{START} \in RM$

Departability:
exists a path from some $q'_{GOAL} \in RM$ to any $q_{GOAL} \in Q_{free}$

Staying away from Obstacles: Generalized Voronoi Graphs

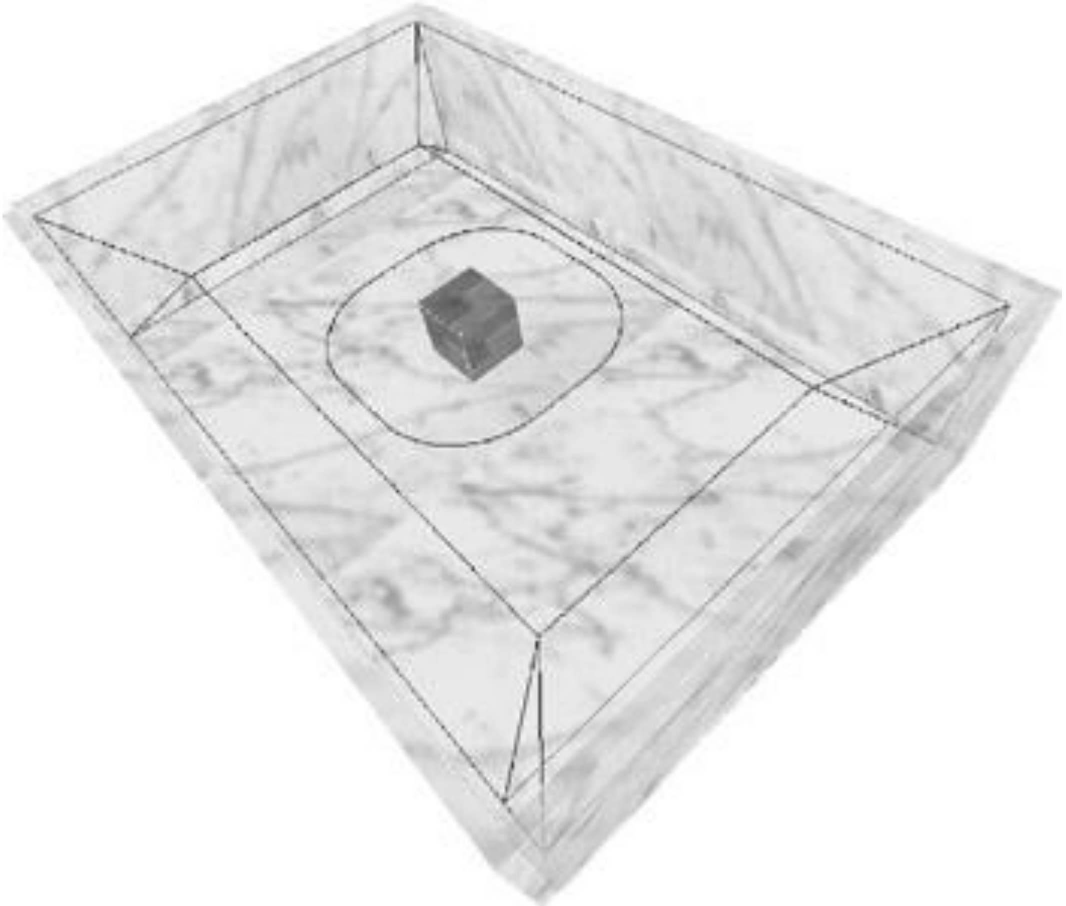
Voronoi Region of obstacle i :

$$\mathcal{F}_i = \{q \in Q_{\text{free}} \mid d_i(q) \le d_h(q) \quad \forall h \neq i\},$$

Voronoi diagram:

set of *q* equidistant from at least two obstacles

Generalized Voronoi Graphs



GVG Roadmaps

Accessibility / Deparability:

Gradient descent on distance from dominant obstacle :

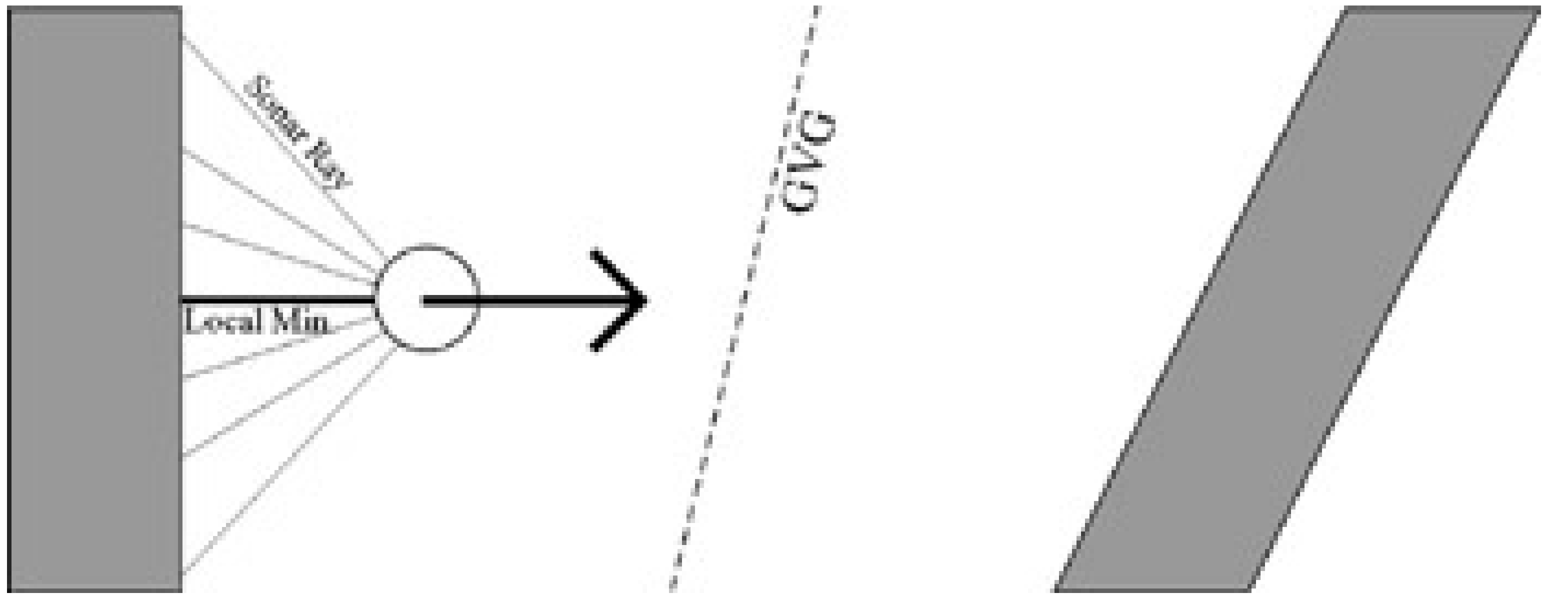
→ guaranteed to reach from any $q_{START} \in Q_{free}$ to some $q'_{START} \in RM$

→ motion is along a “retract” or brushfire trajectory

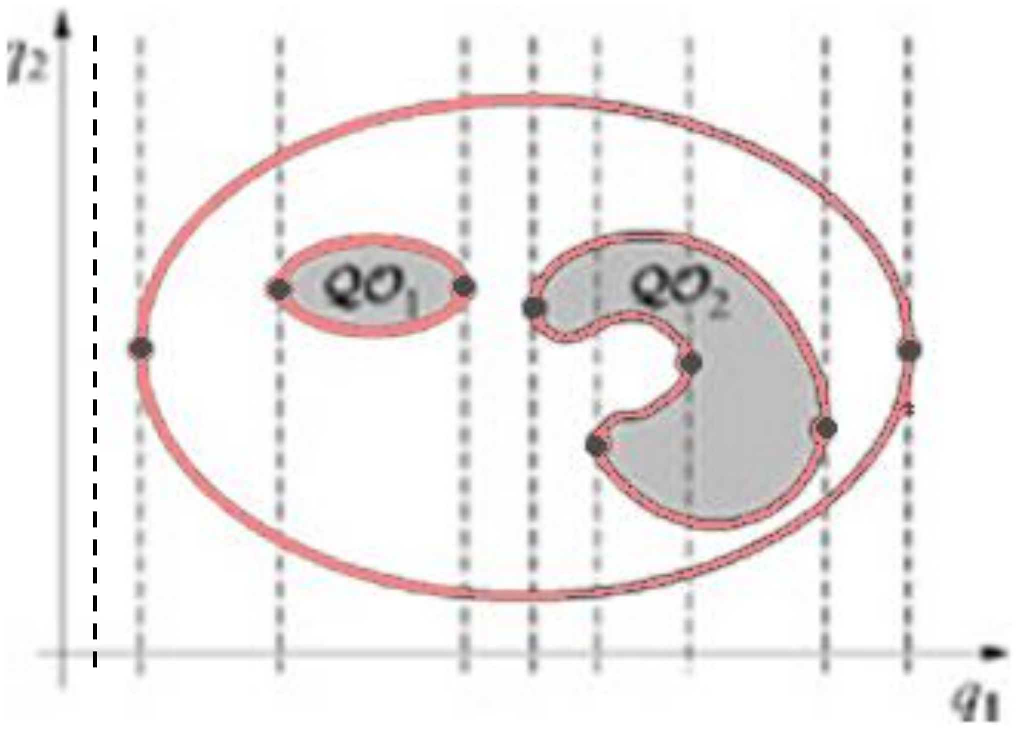
Connectivity:

GVG is Connected if path exists

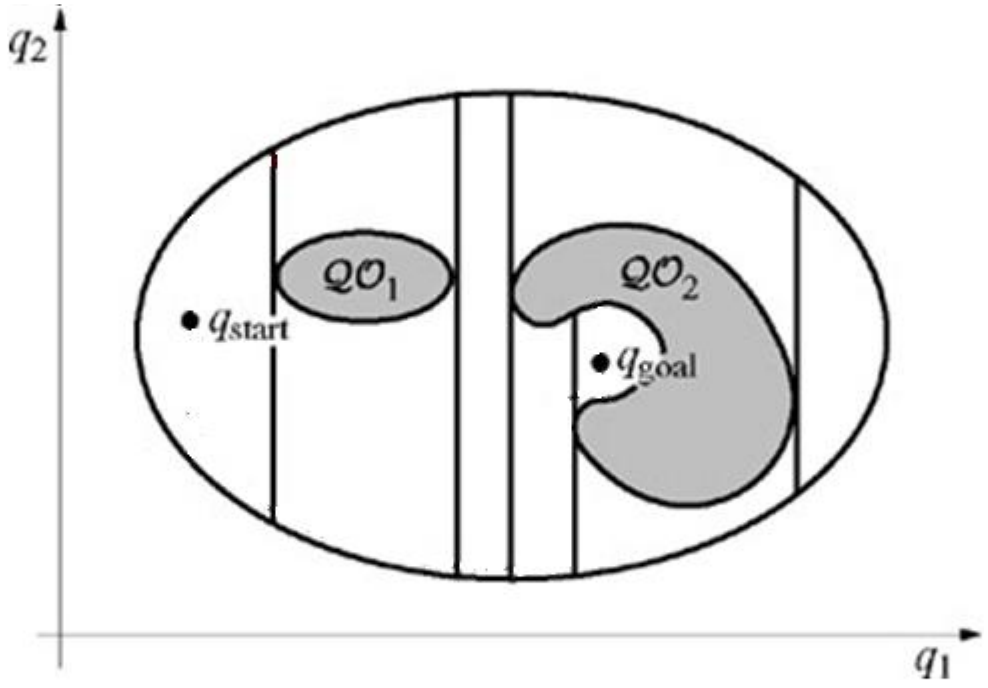
Sensor based Voronoi roadmap construction



Canny's Silhouette roadmap



Canny's Silhouette roadmap



Canny's Complexity Analysis

n: = degrees of freedom of robot (dim of C-space)

obstacles C-space boundaries represented as p polynomials of maximum degree w

Complexity:
any navigation path-planning problem can be solved in $p^n(\log p)w^{O(n^4)}$ time

Probabilistic Roadmap (PRM)

Probabilistic Roadmap

Sample n poses $q_1 \ldots q_n$ in the WORKSPACE

Free space nodes: Reject q_i that intersect with an obstacle, remaining nodes q are in Q_{free}

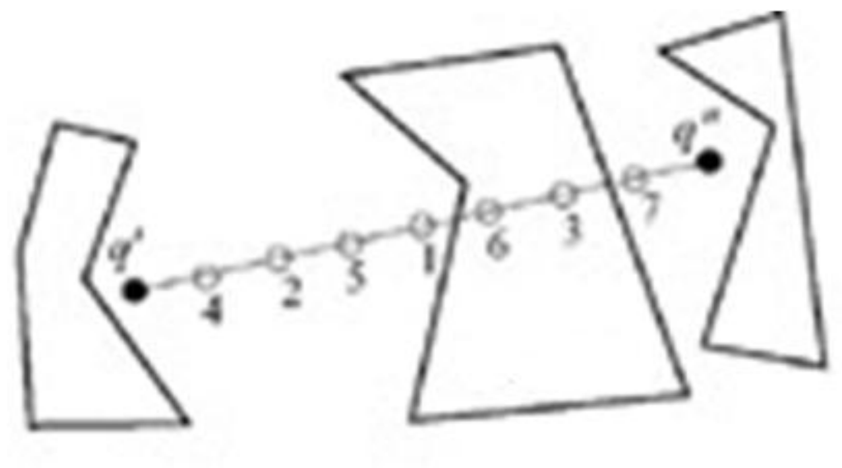
Local planning: in k-nearest neighbours, if path $<q_i,q_j>$collision-free, add edge to graph

Resulting graph = *Probabilistic Roadmap*

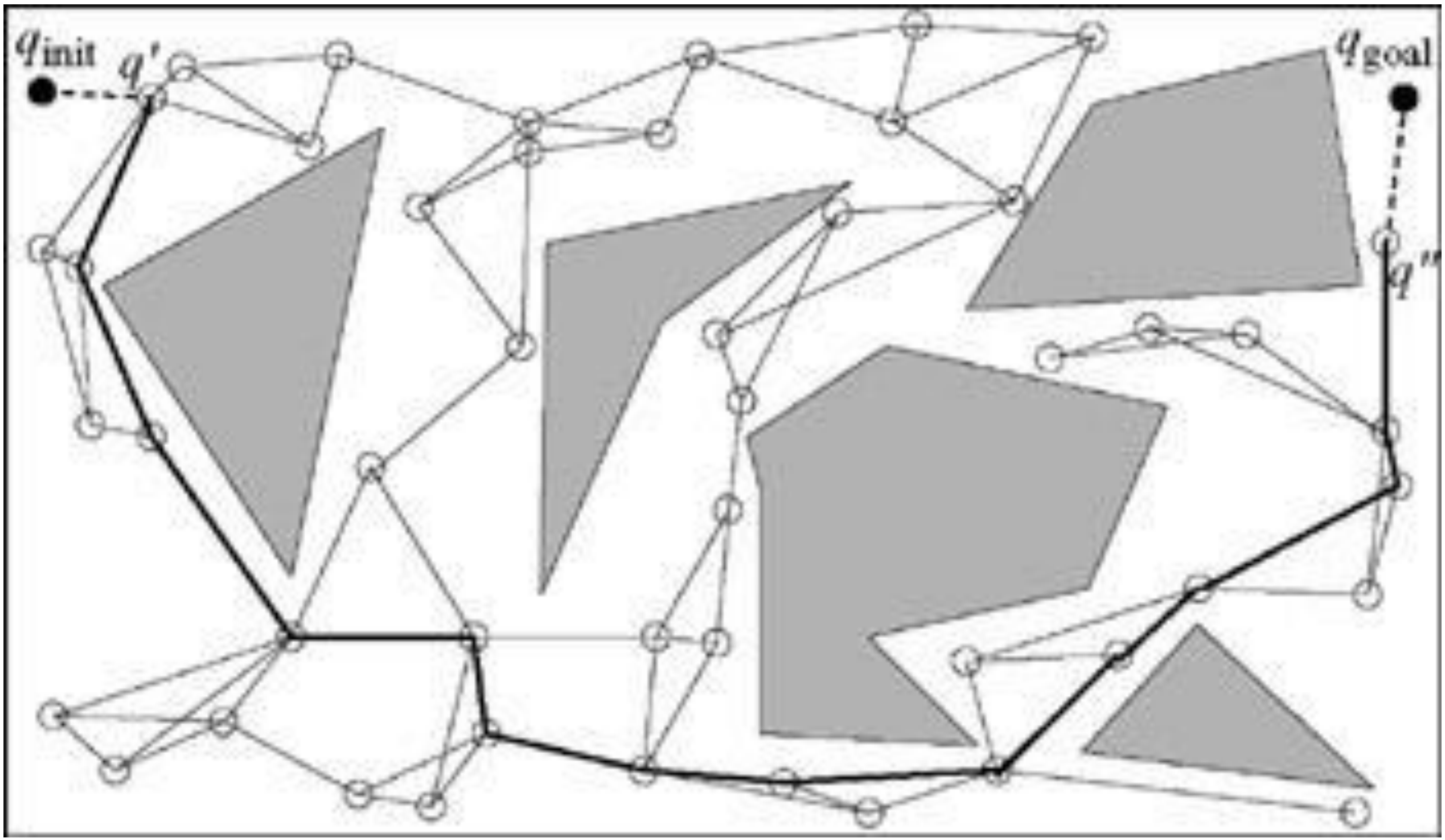
Local Planner

Objective: Test if path <q_i,q_j> is collision-free

Linear Subdivision algorithm: start at midpoint(q_i,q_j) ; subdivide recursively until desired precision



Probabilistic Roadmaps (PRM)



Sampling-based motion planning

Sample n poses $q_1 \ldots q_n$ in the workspace

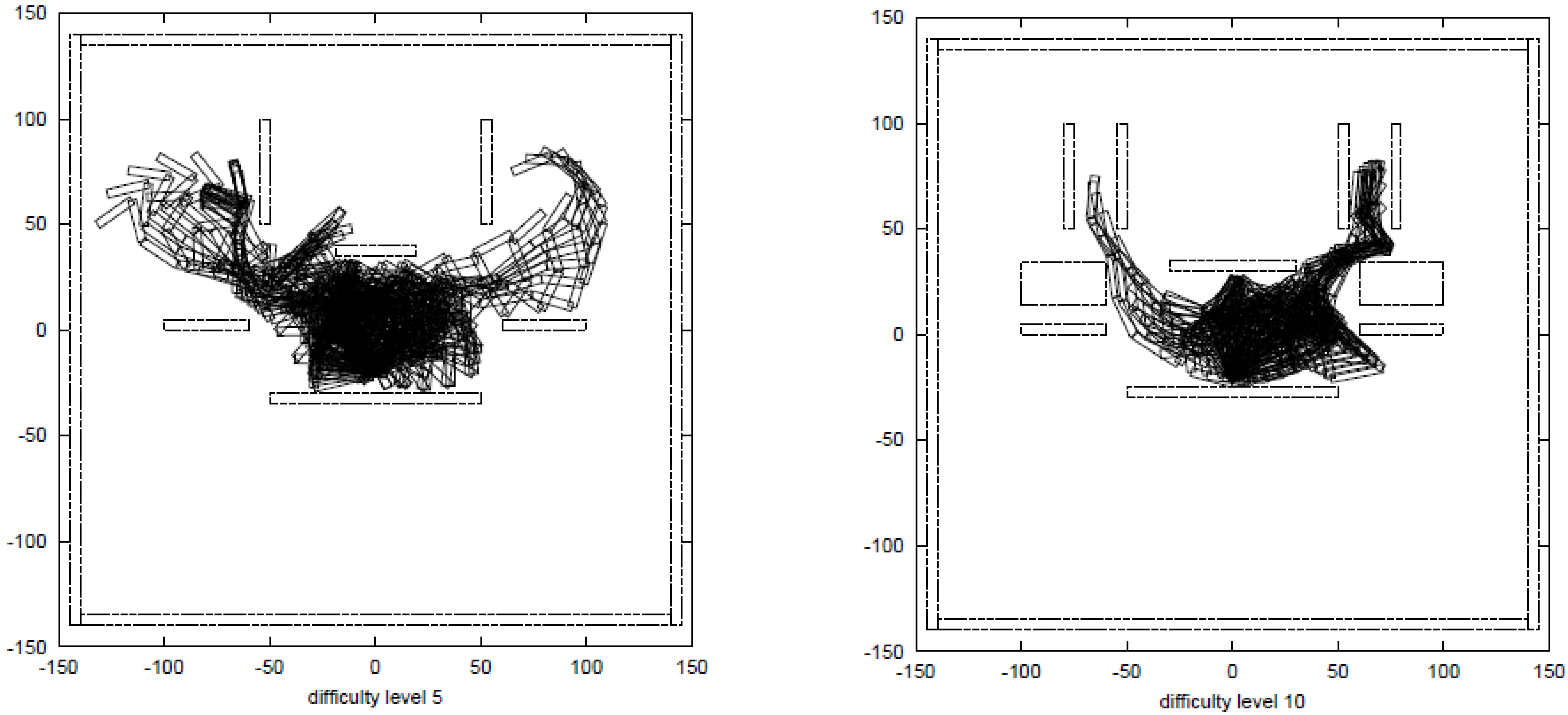
Reject q_n that overlap with an obstacle, remaining poses are in Q_{free}

Use local planning to determine if a path exists between neighbours q_i and q_j.

Resulting graph = *Probabilistic Roadmap*

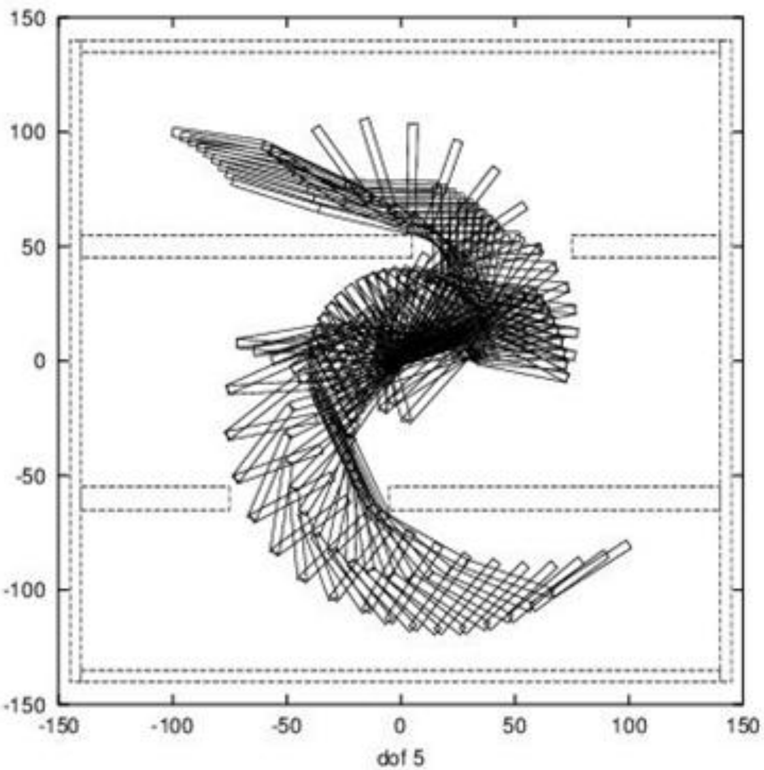
Probabilistically complete:
As #samples $n \rightarrow \infty$, Prob (success) $\rightarrow 1$

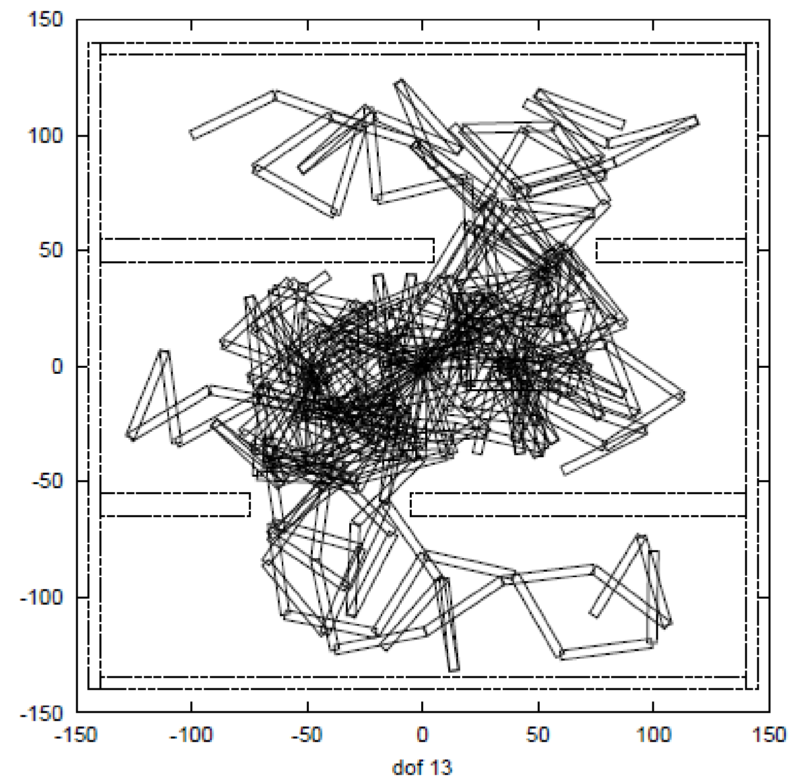
Hyper-redundant robot motion planning using PRM



[sinha mukerjee dasgupta 02]

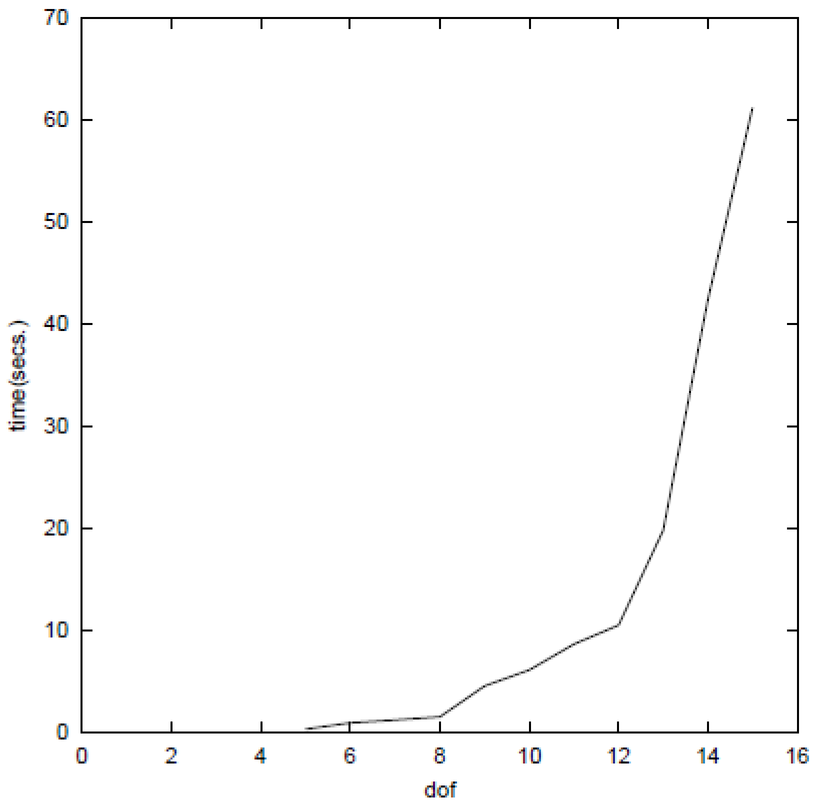
Hyper-redundant robot motion planning using PRM





[sinha mukerjee dasgupta 02]

Hyper-redundant motion planning

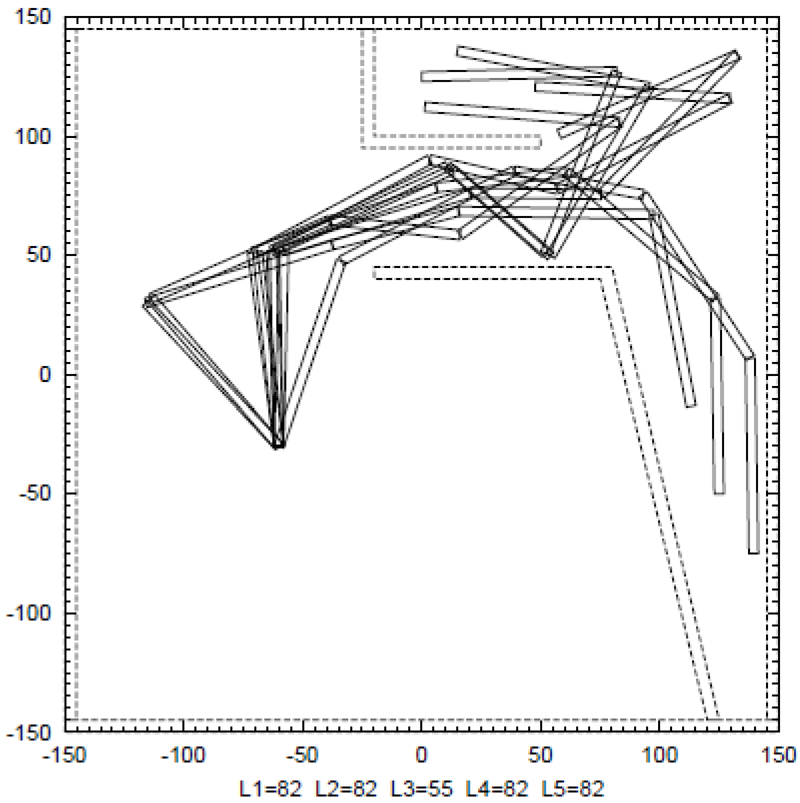


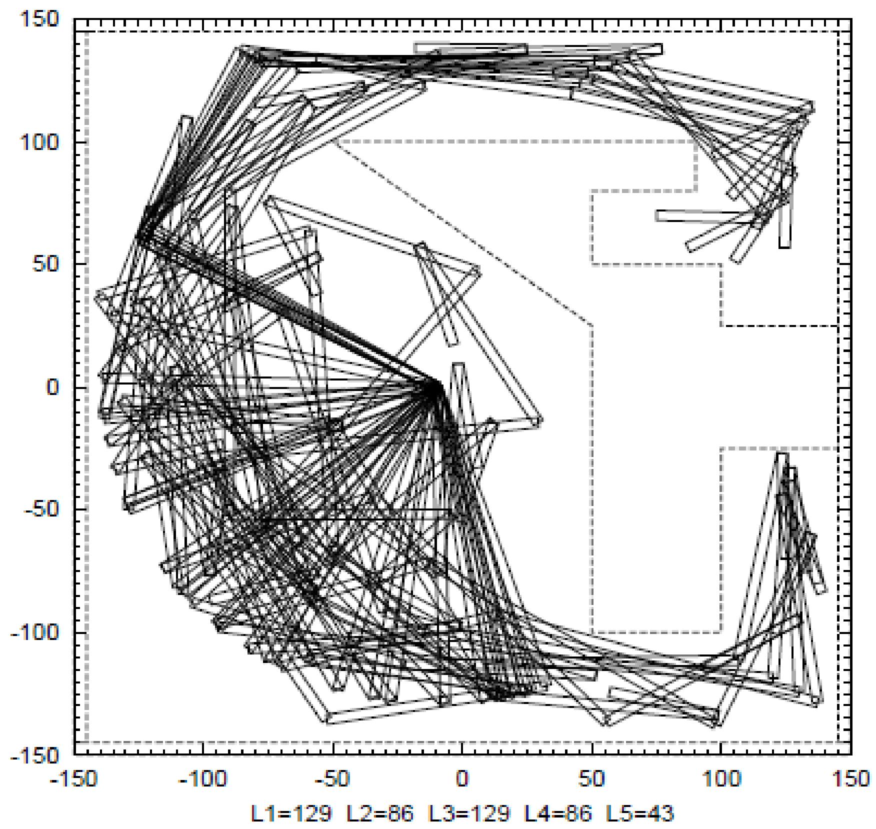
Time:

Exponential in DOFs

[sinha mukerjee dasgupta 02]

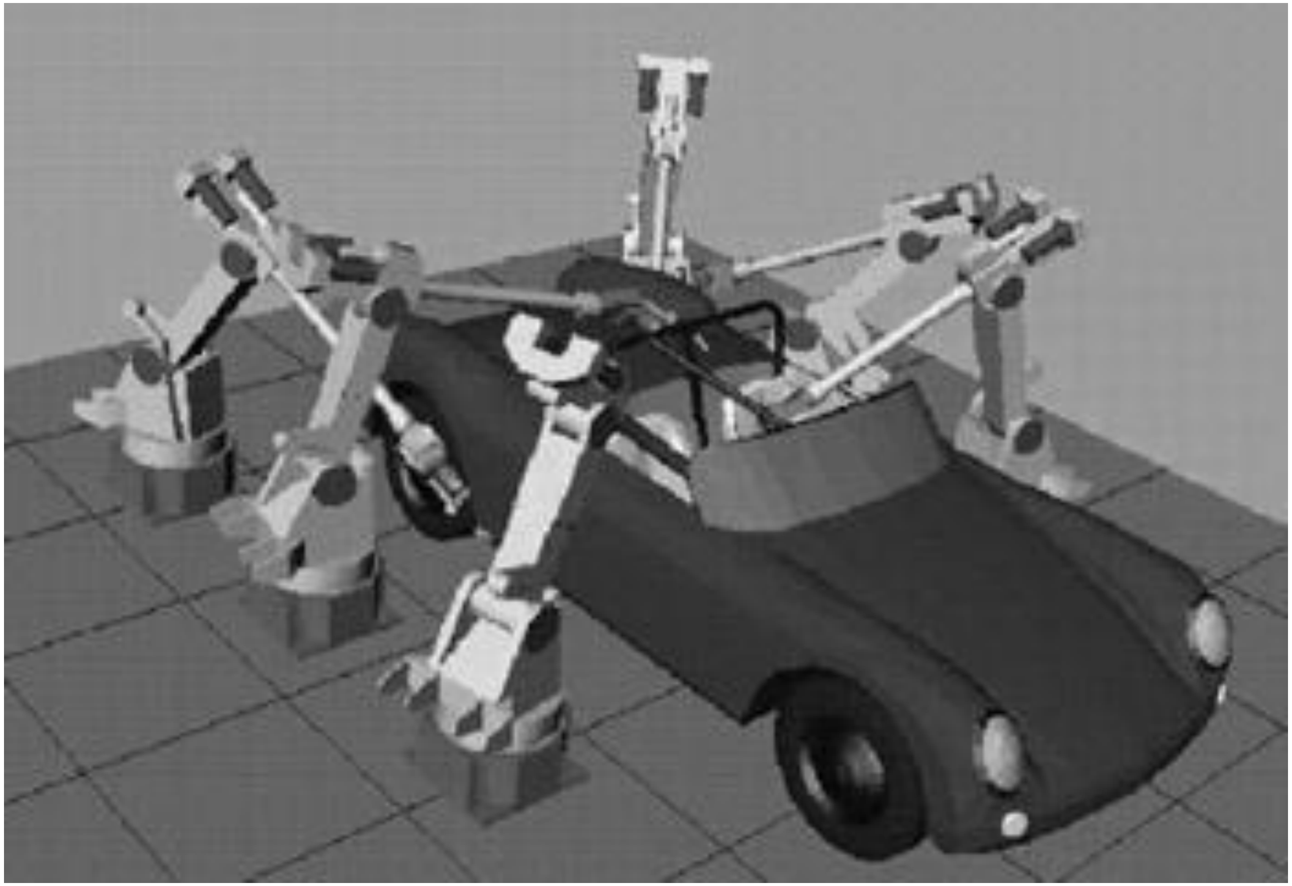
Design for manipulability





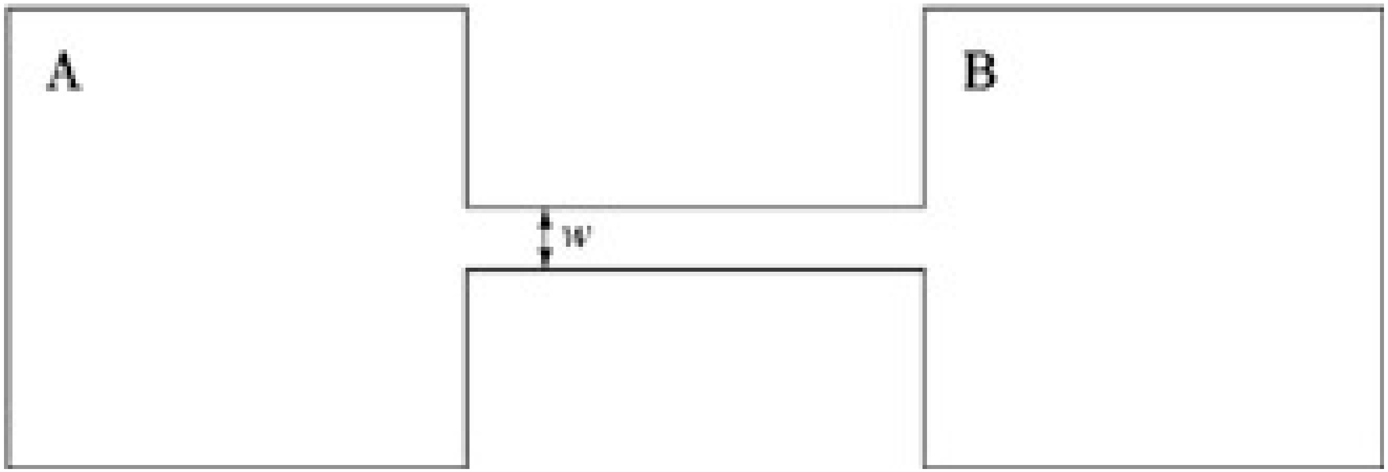
[sinha mukerjee dasgupta 02]

PRM applications



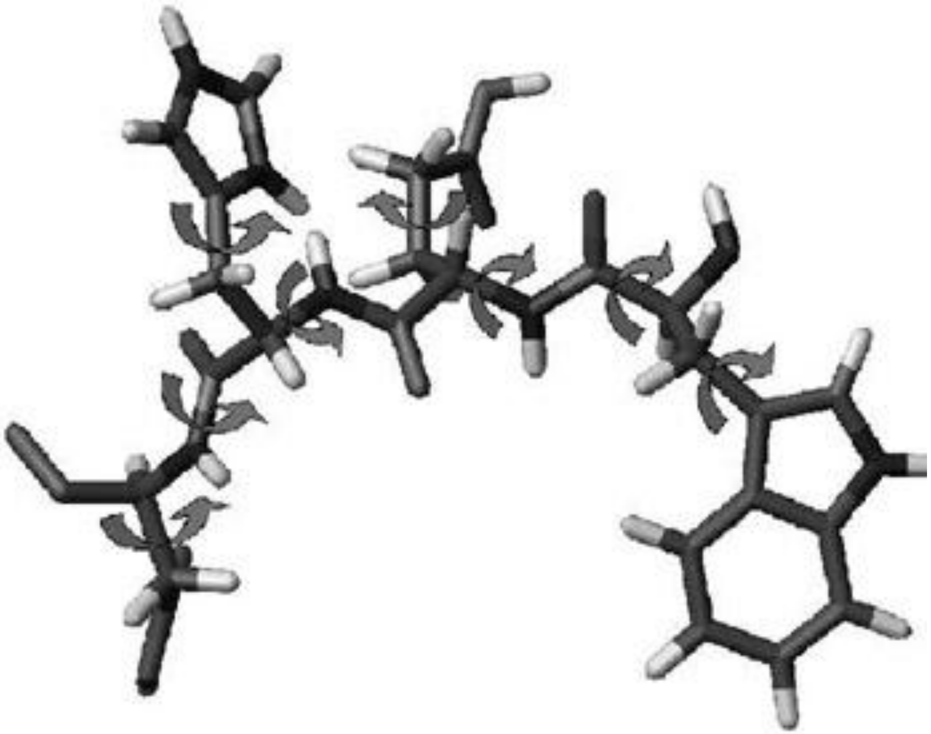
42 DOFs: [Sánchez and J. C. Latombe 02]

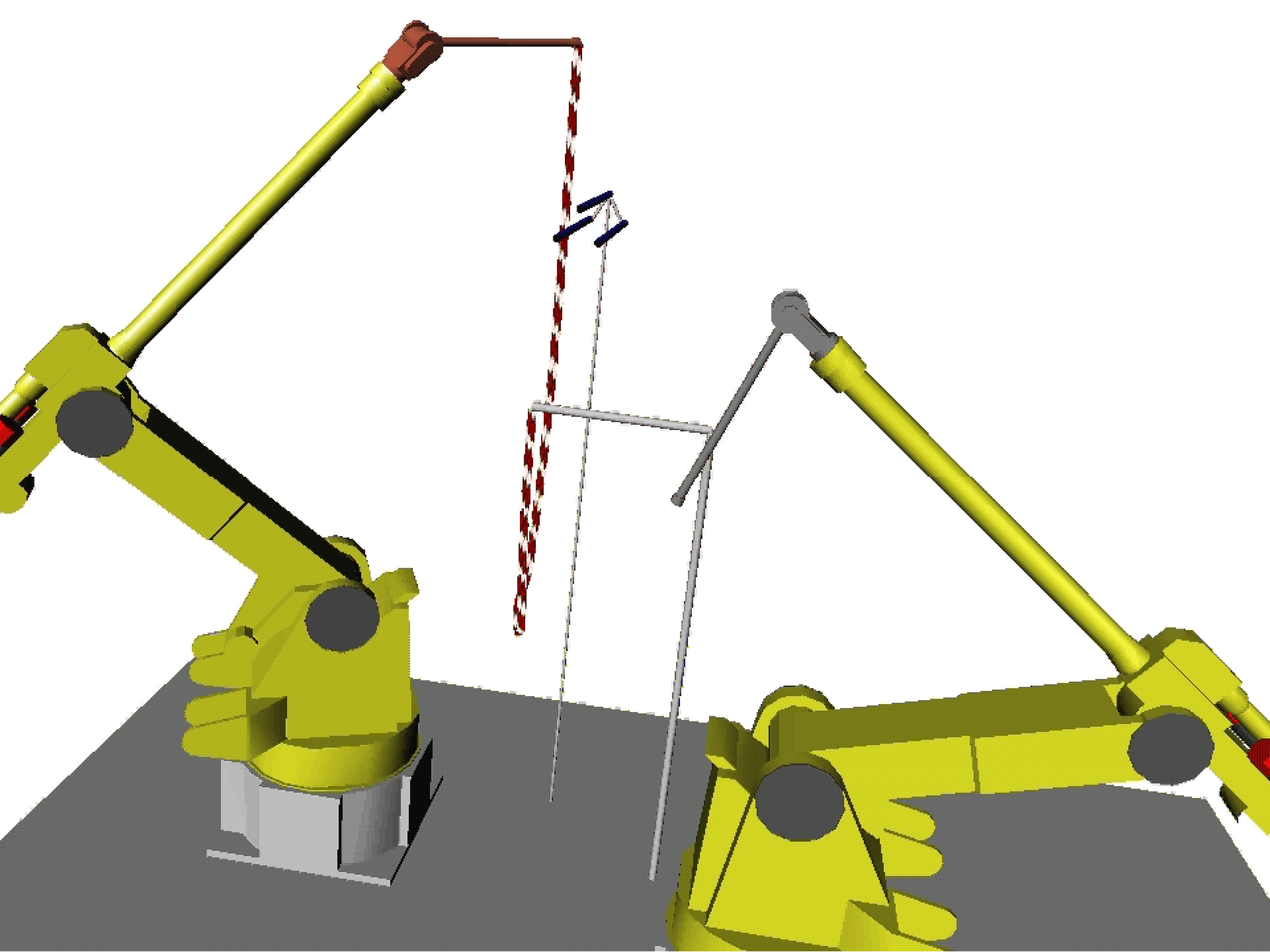
Narrow corridor problem



Solution: generate more samples near boundary
- bias the sample towards boundary region
- if midpoint between two obstacle nodes is free, add

PRM applications : Protein folding



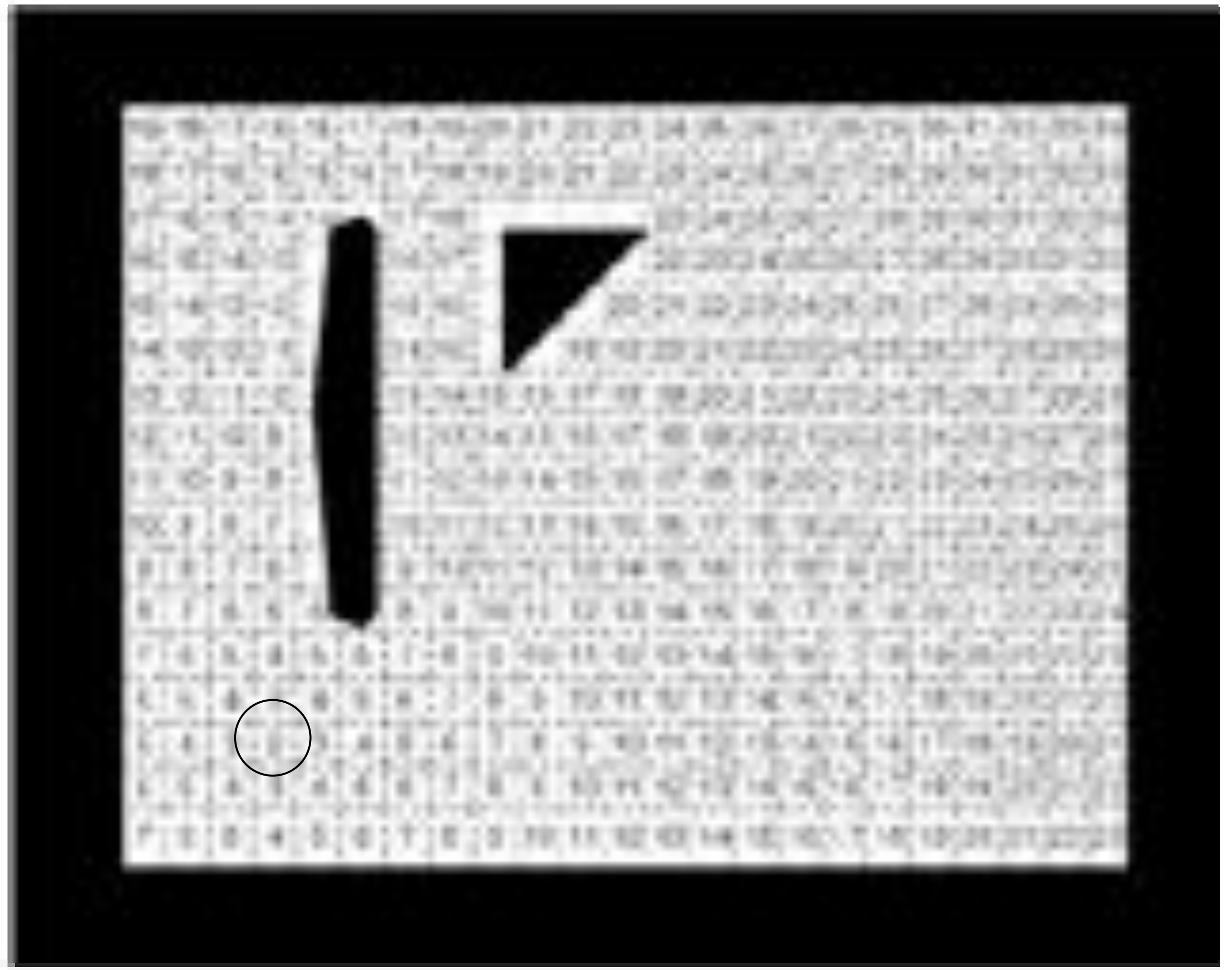


Continuum methods:
Overcoming Local minima

Grid-based: Wave-front

- Grid-based model
- given a start grid cell $\mathbf{q}_S$ assign it the value “2”
  - Every neighbour gridcell gets +1
  - Until grid is filled
- Given a goal cell $\mathbf{q}_G$ use greedy search to find path back to goal

Grid-based: Wave-front



$O(k^d)$ space / time

Navigation Function : Sphere space

- Spherical wall (r_0), with spherical obstacles inside
- Obstacle distance

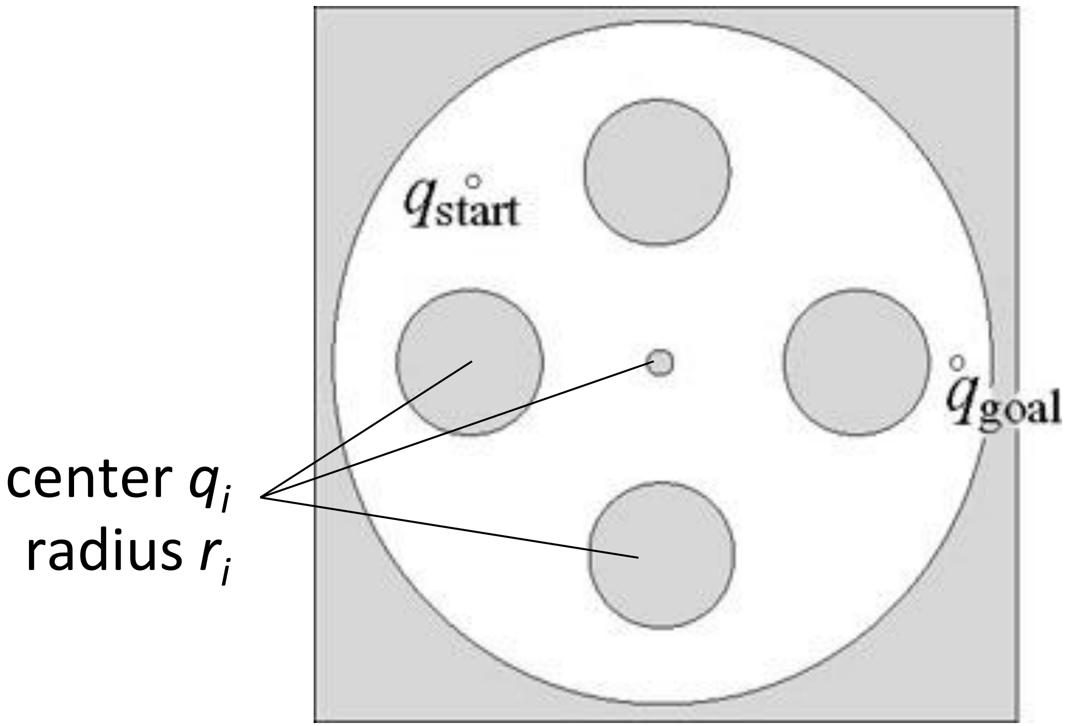
$$QO_i = \{q \mid \beta_i(q) \leq 0\}$$

$$\beta_0(q) = -d^2(q, q_0) + r_0^2, \quad \text{— wall}$$

$$\beta_i(q) = d^2(q, q_i) - r_i^2, \quad \text{— obstacles}$$

[Rimon Koditschek 92]

Sphere space



Rimon Koditschek 92

Navigation Function : Sphere space

- Spherical wall (r_0), with spherical obstacles inside
- Obstacle distance

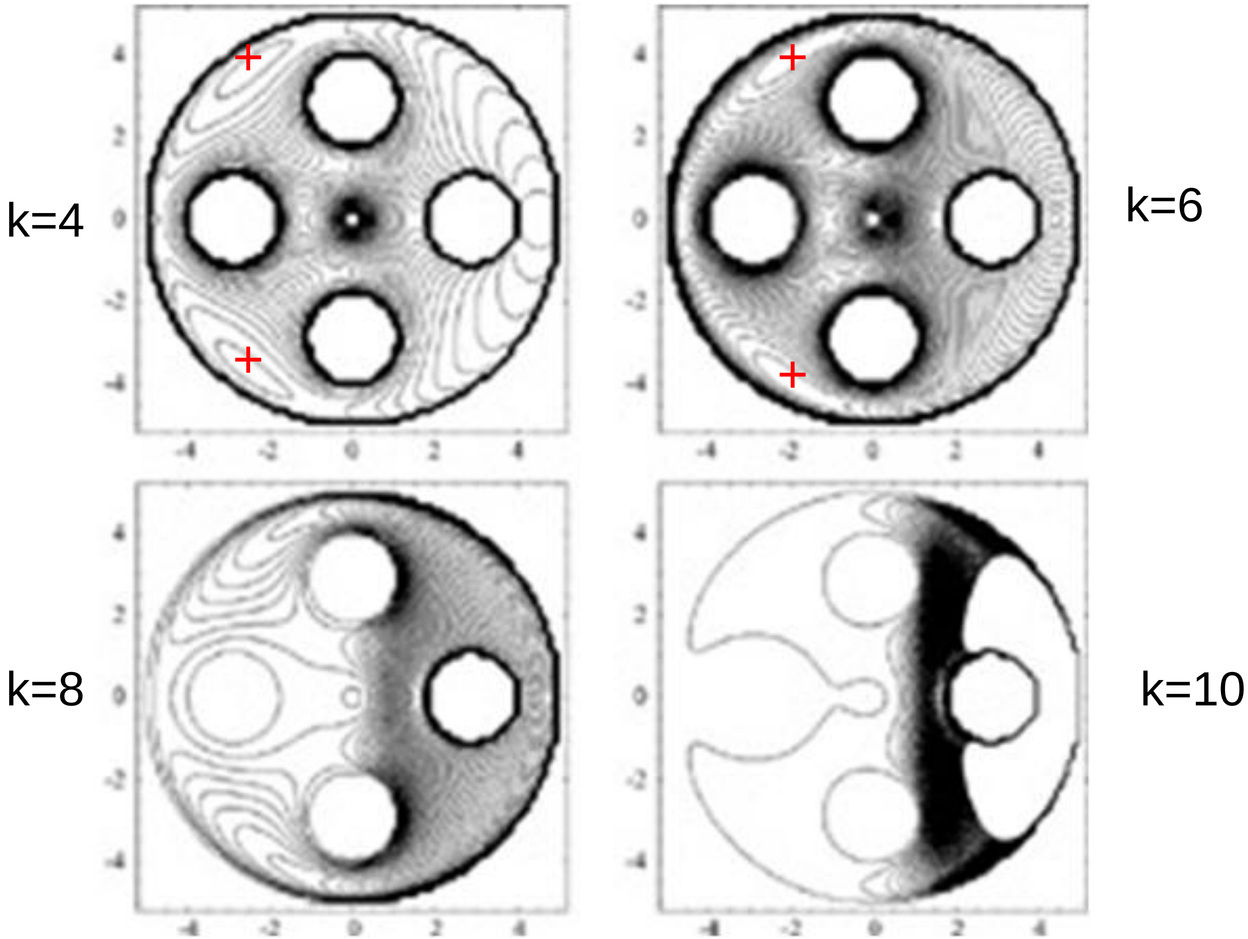
  $\mathcal{QO}_i = \{q \mid \beta_i(q) \leq 0\}$

  $\beta_0(q) = -d^2(q, q_0) + r_0^2,$ — wall

  $\beta_i(q) = d^2(q, q_i) - r_i^2,$ — obstacles
- Goal potential with high exponent $\gamma_\kappa(q) = (d(q, q_{\text{goal}}))^{2\kappa}$
- Instead of sum, use product to combine obstacle potentials $\beta(q) = \prod_{i=0}^{n} \beta_i(q).$
- For high k, $\frac{\gamma_\kappa}{\beta}(q)$ has unique minima at goal

[Rimon Koditschek 92]

Navigation Function



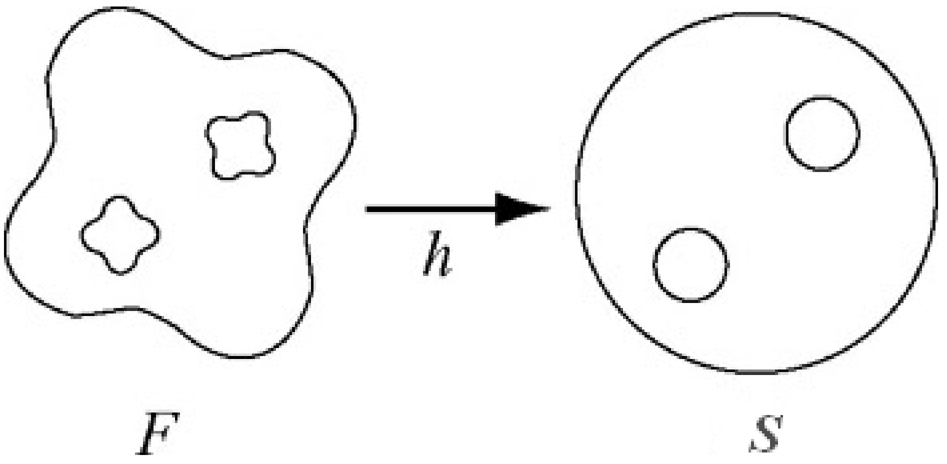
Choset etal 05

Navigation Function

$\varphi : S \rightarrow [0, 1]$:
navigation function on sphere space S.

For any space F if exists diffeomorphic mapping $h : F \rightarrow S$
(i.e. h is smooth, bijective, and has a smooth inverse),

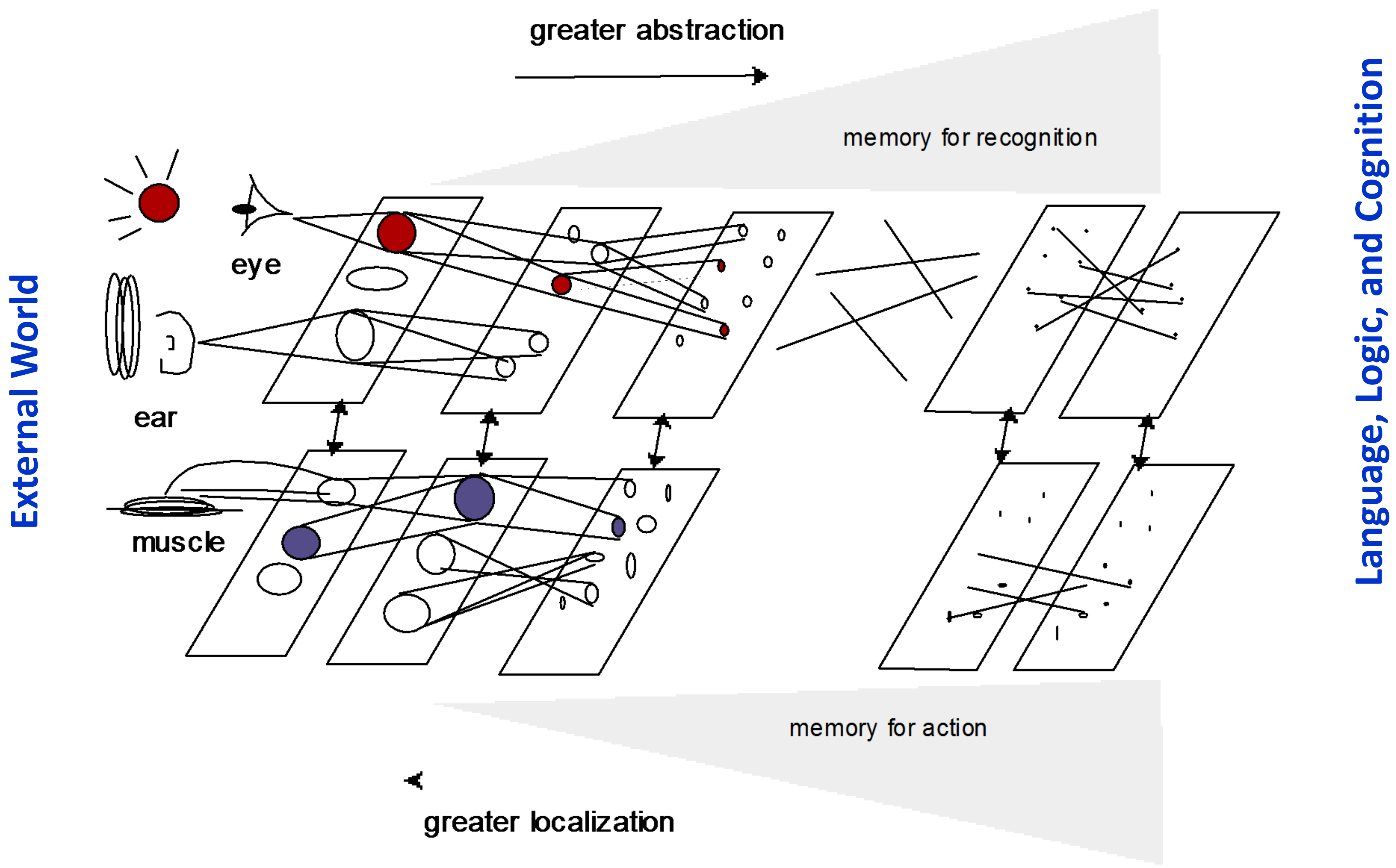
then $\varphi = \varphi \circ h$ is a navigation function on F



Choset etal 05

Sensori-motor map learning

Cognitive Architecture: Levels of Abstractions



Visuo-Motor expertise

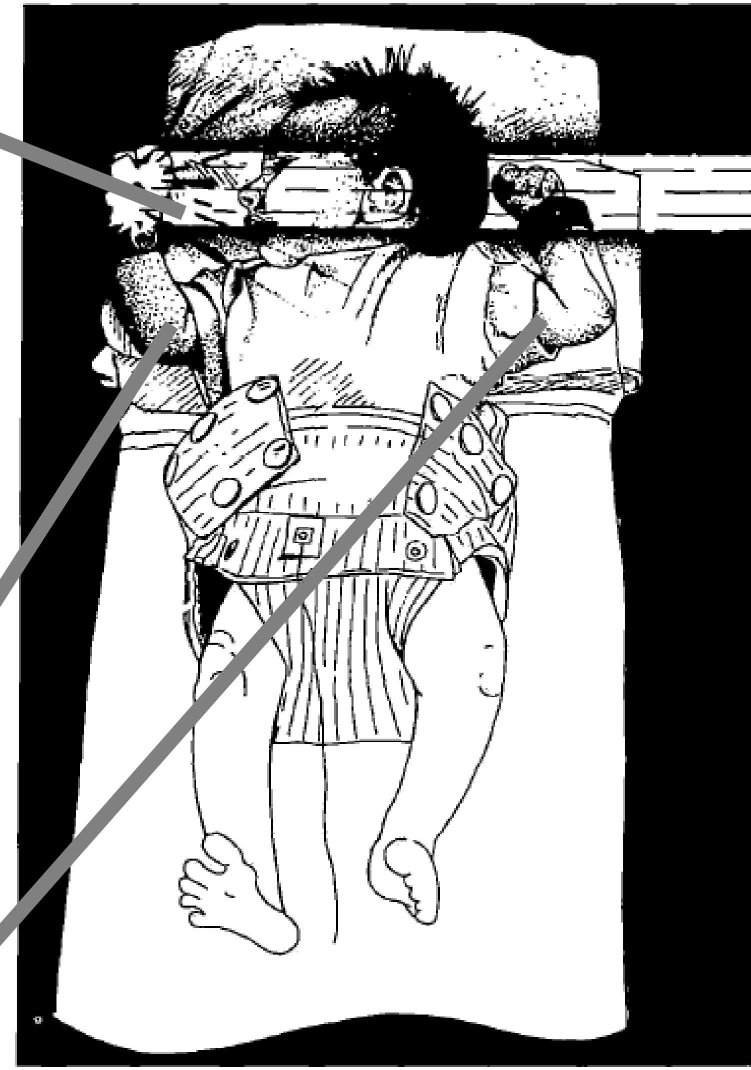
in darkened room,
works hard to position arm
in a narrow beam of light

Newborns
(10-24 days)

Small weights
tied to wrists

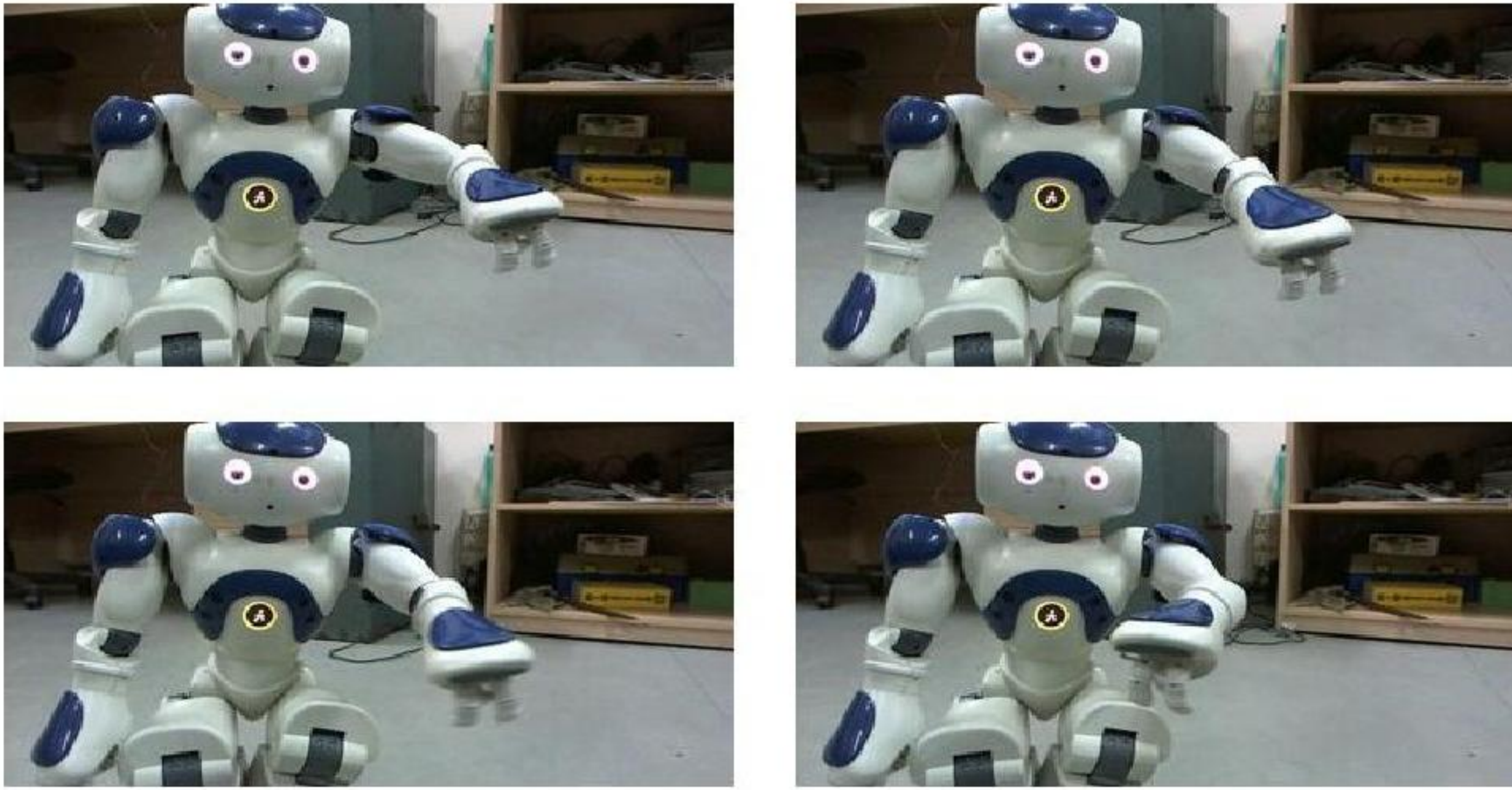
Will resist weights to move
the arm they can see

Will let it droop if
they can't see it

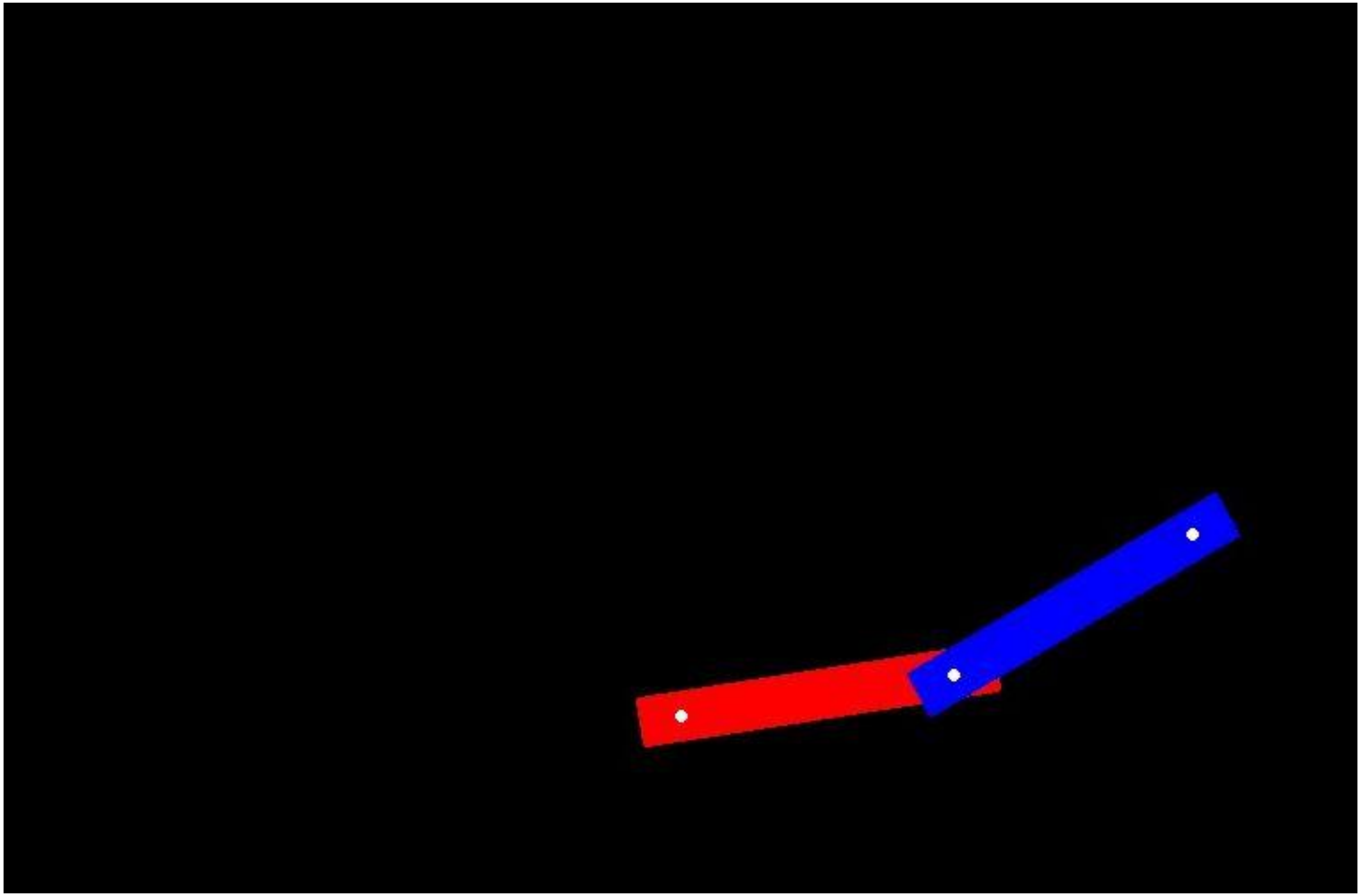


[A. van der Meer, 1997: Keeping the arm in the limelight]

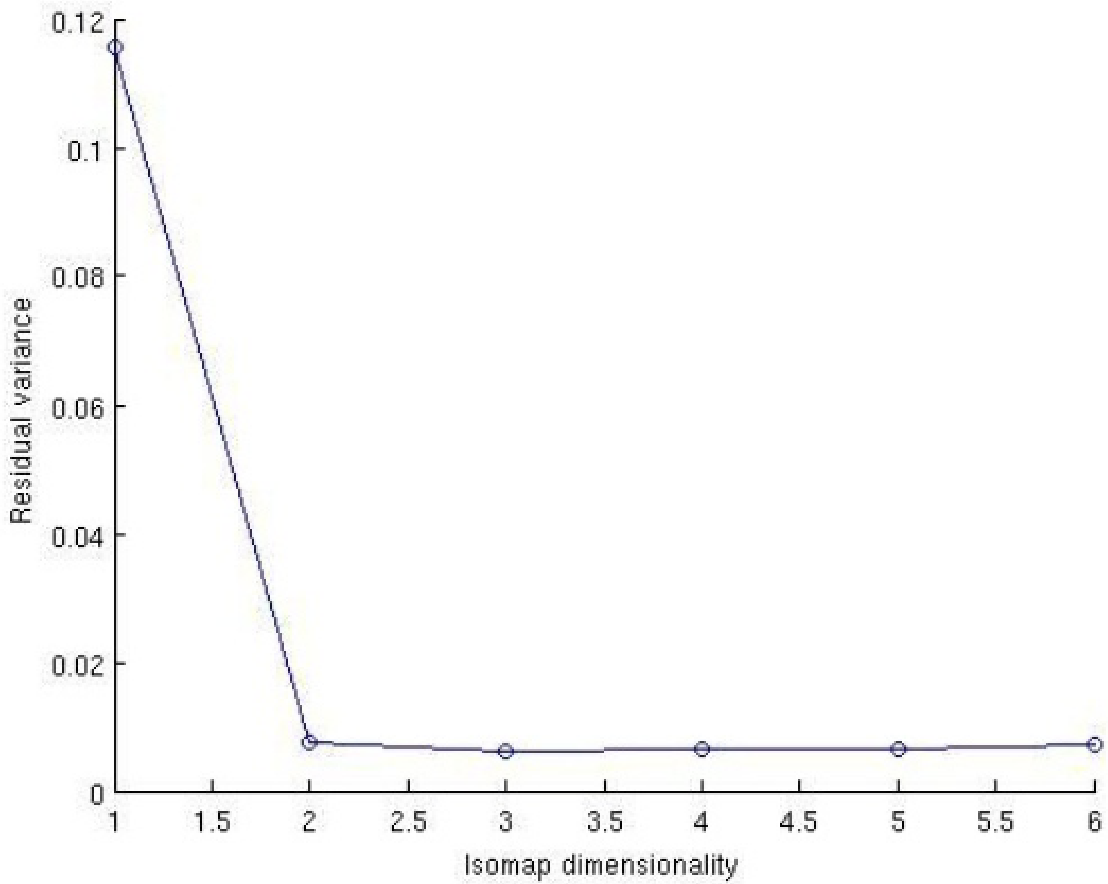
Observing self motions



Simulation



Manifold dimension



dofs : 2

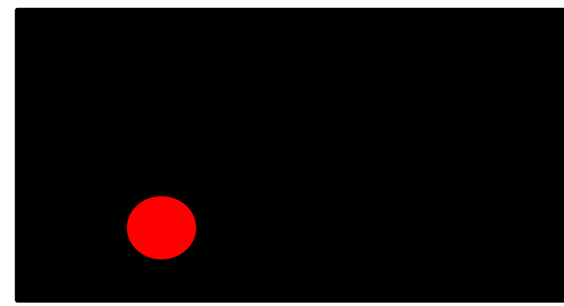
Discovering Configuration Space

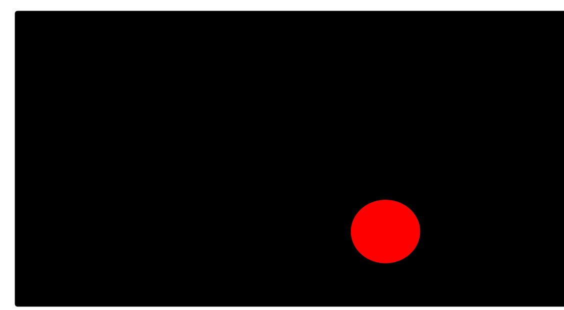
Collect a set of images of the robot at random configurations

Reduce the image set to a low dimensional representation

Latent variables: $(y_1, y_2, \ldots y_d)$

for d=2, each image represented as a pair of values





Smoothly Deformable objects

Object S = set of connected points S in $G \subset R^d$.

Deformation function $h : G \rightarrow G$ is a function of a parameter vector q = $\{q_1 \ldots q_k\}$.

Smooth Deformation : $S \rightarrow hS(\boldsymbol{q})$, is a diffeomorphism from G to G

Smoothly Deformable objects

Ex.1

Articulated chain with N rigid links, and k joints, each with 1 DOF each.

hS determined by $\{q_1 \ldots q_k\}$, where each q_i is the parameter associated with each joint.

Deformable objects

LEMMA:

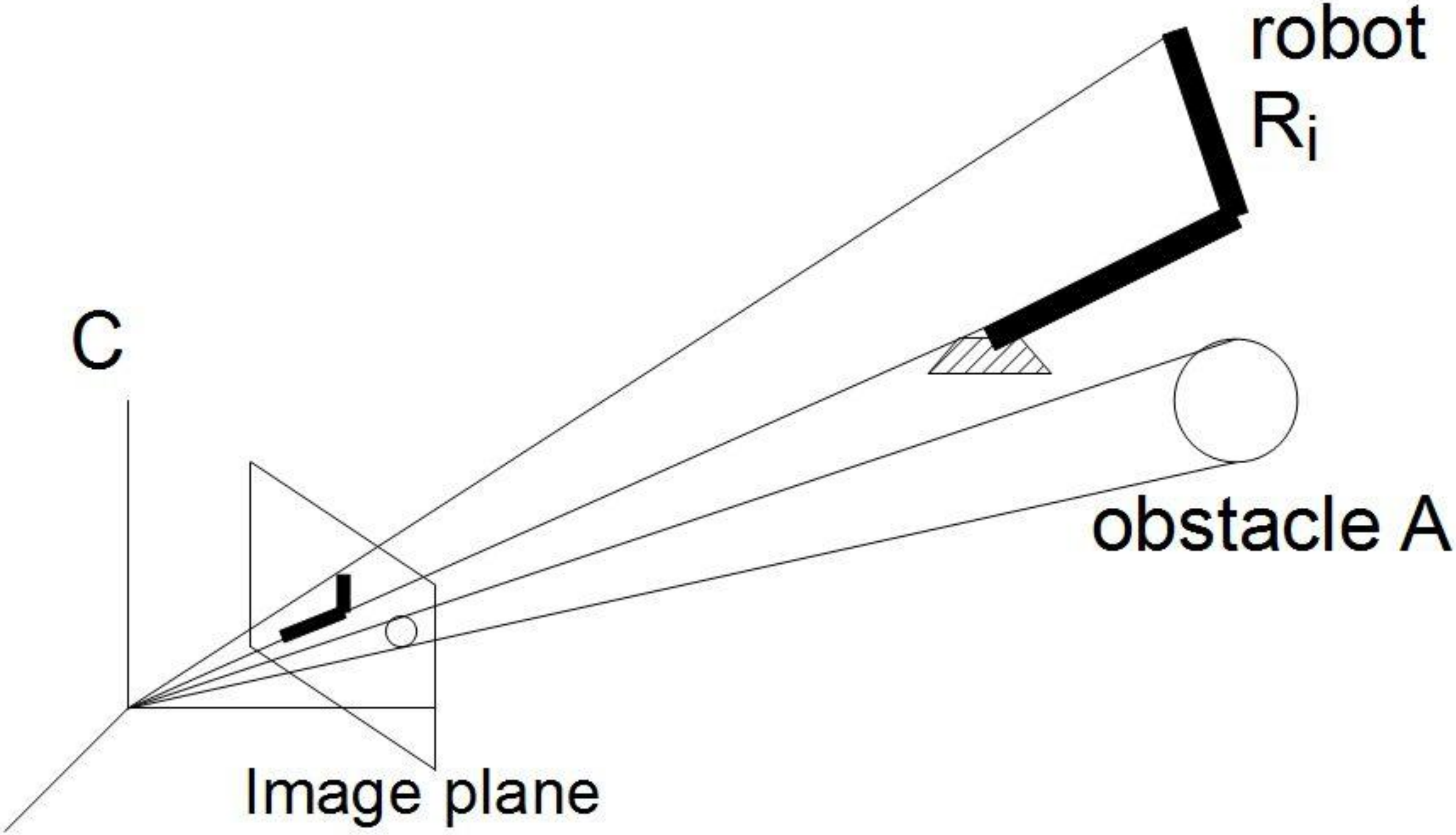
The space of shapes and poses of S is a manifold of at most k dimensions.
(k = dimensionality of **q)**.

Any x IN S(q) has a neighbourhood N, s.t. there is a homeomorphism phi from N onto the space Q of {q1...qN}. In R^N

the map phi can be composed of the motion transformation T(q1..qm) [a special euclidean group) and
the smooth deformation shape funnction hS(q1..qd)

Both of these transformations being diffeomorphic.

Imaging transformation



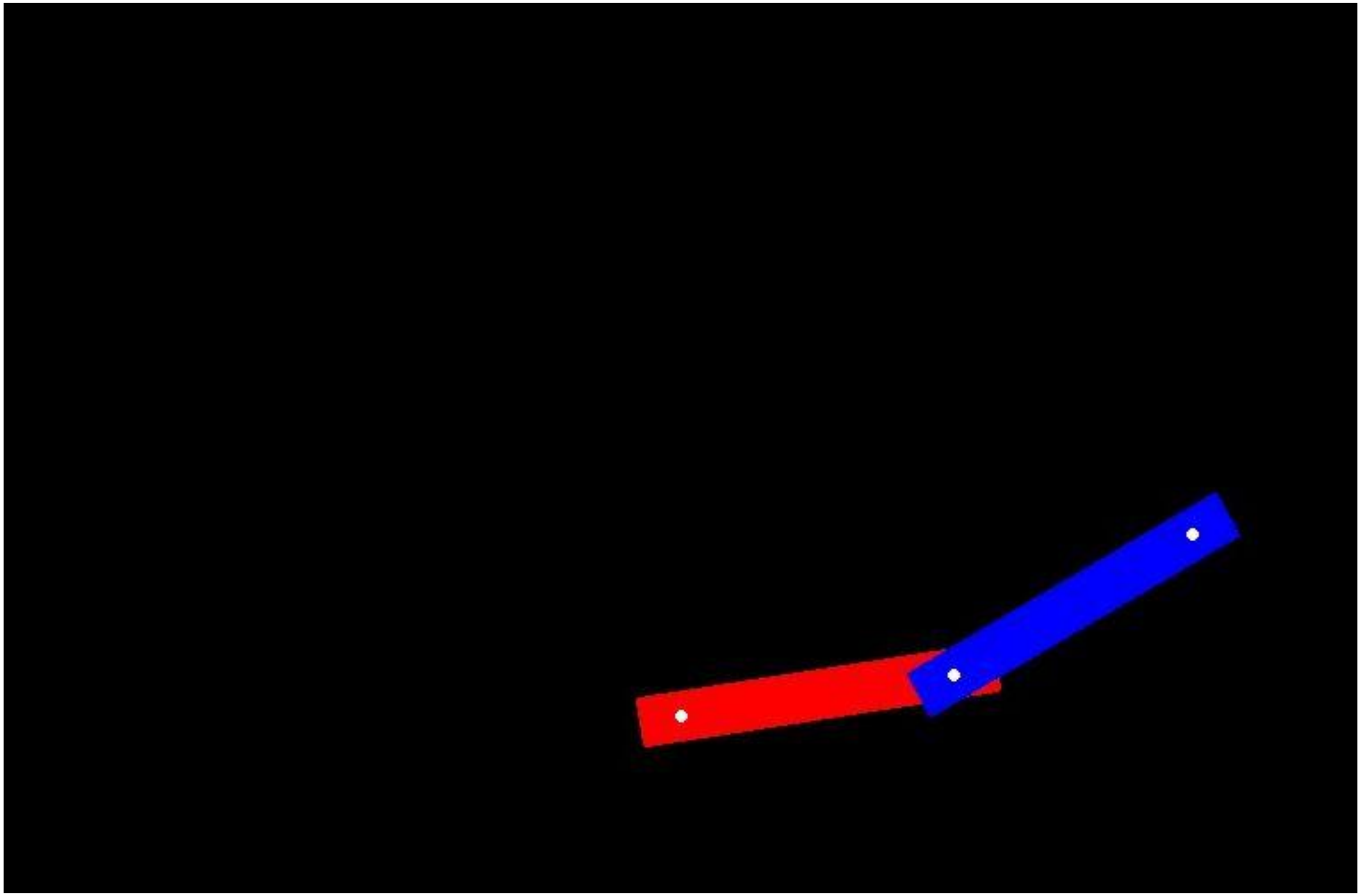
Visual Manifold theorem

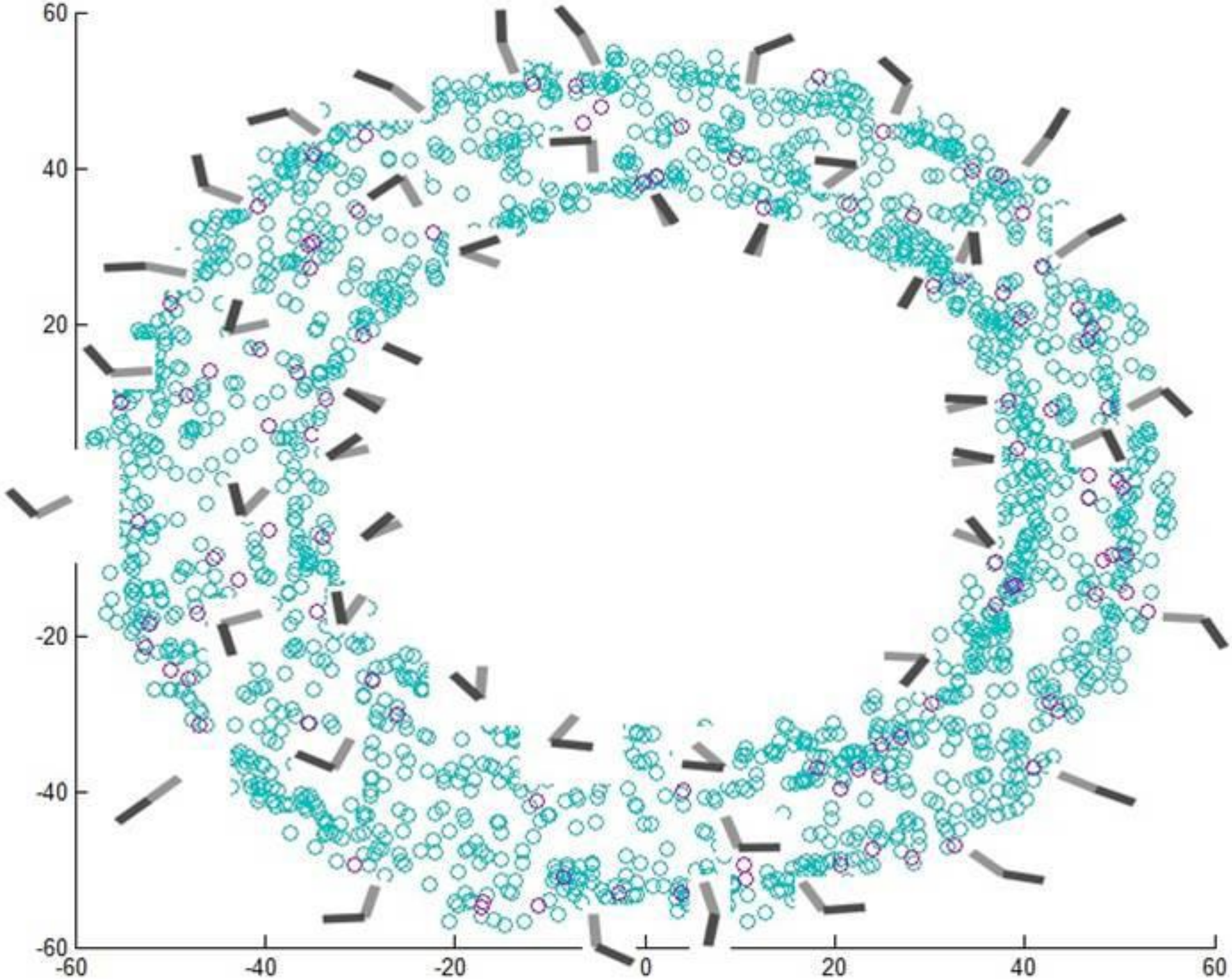
If S is a smoothly moving, visually distinguishable, deformable object, then any image of S, taken from a fixed camera, would lie on a manifold of at most N dimensions in the image space.

Visual Distinguishability

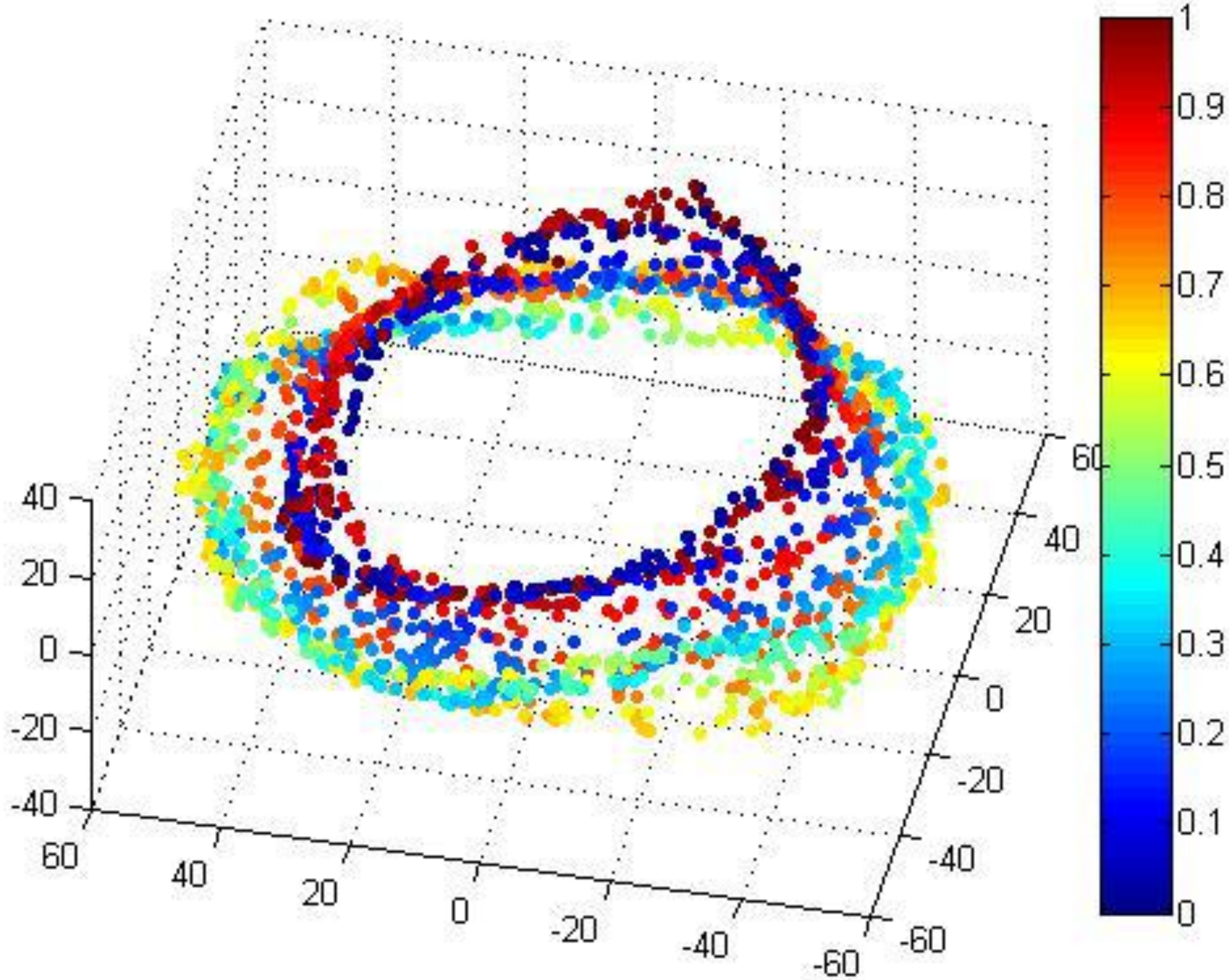
Robot Structure Discovery

Simulation

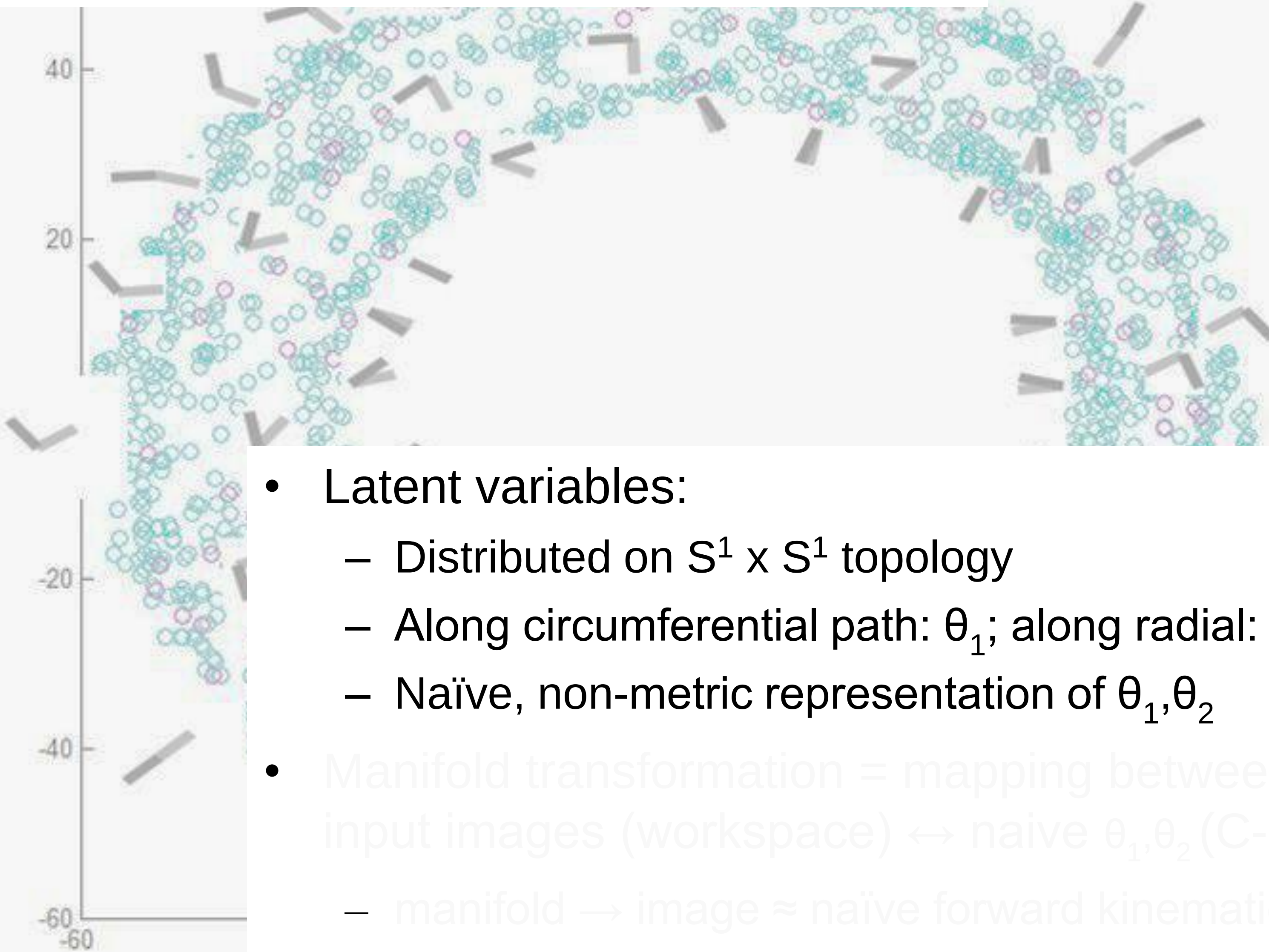



60
40
20
-20
-40
-60
-60
-40
-20
0
20
40
60

Robot Structure Learning



Robot Structure Learning



- Latent variables:
  - Distributed on S^1 x S^1 topology
  - Along circumferential path: θ_1; along radial: θ_2
  - Naïve, non-metric representation of θ_1,θ_2
- Manifold transformation = mapping between input images (workspace) ↔ naive θ_1,θ_2 (C-space)
  - manifold → image ≈ naïve forward kinematics
  - image → manifold ≈ naïve inverse kinematics

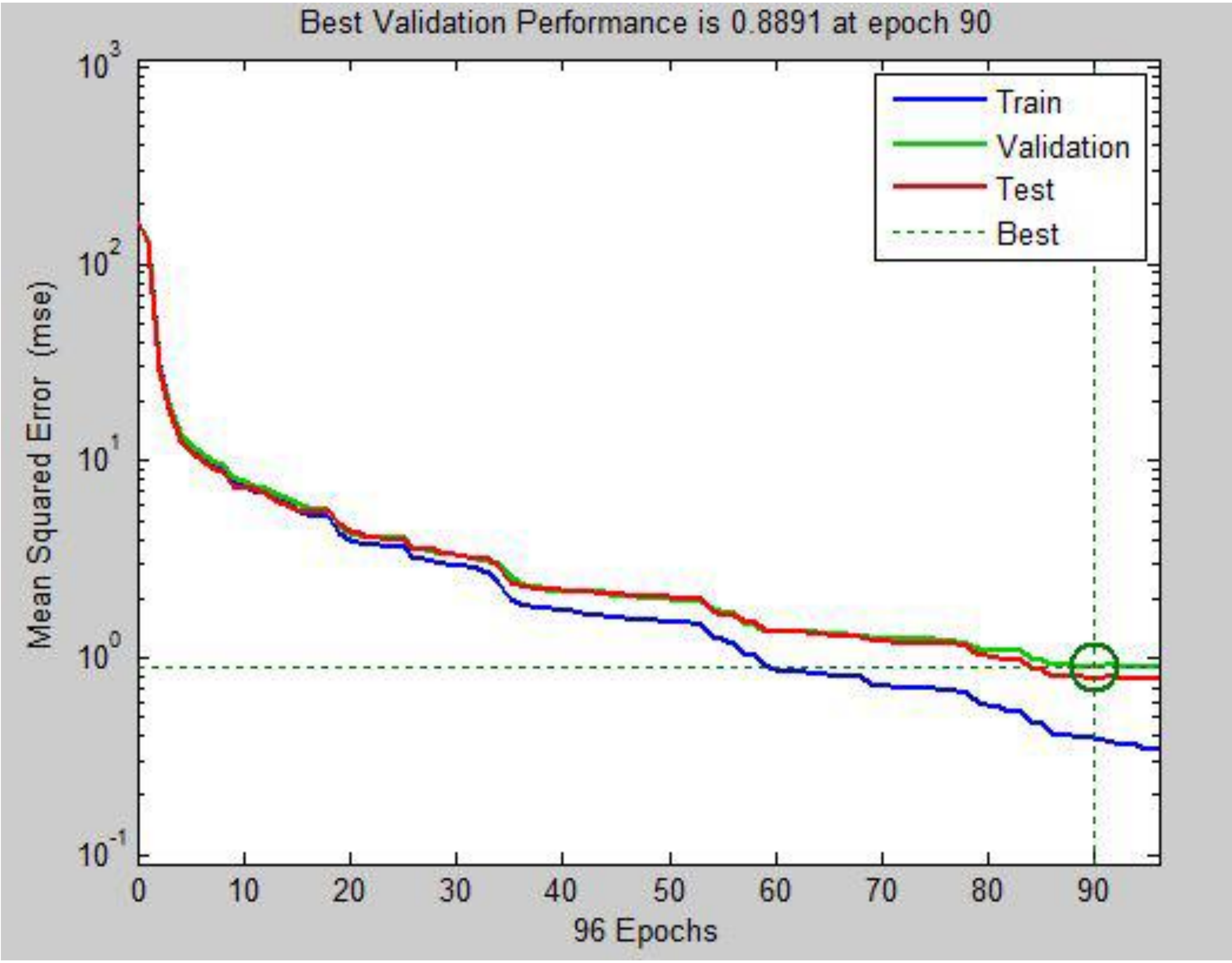
Robot Structure Learning

- Latent variables:
  - Distributed on $S^1 \times S^1$ topology
  - Along circumferential path: θ_1; along radial: θ_2
  - Naïve, non-metric representation of θ_1,θ_2
- Manifold transformation = mapping between input images (workspace) ↔ naive θ_1,θ_2 (C-space)
  - manifold → image ≈ naïve forward kinematics
  - image → manifold ≈ naïve inverse kinematics

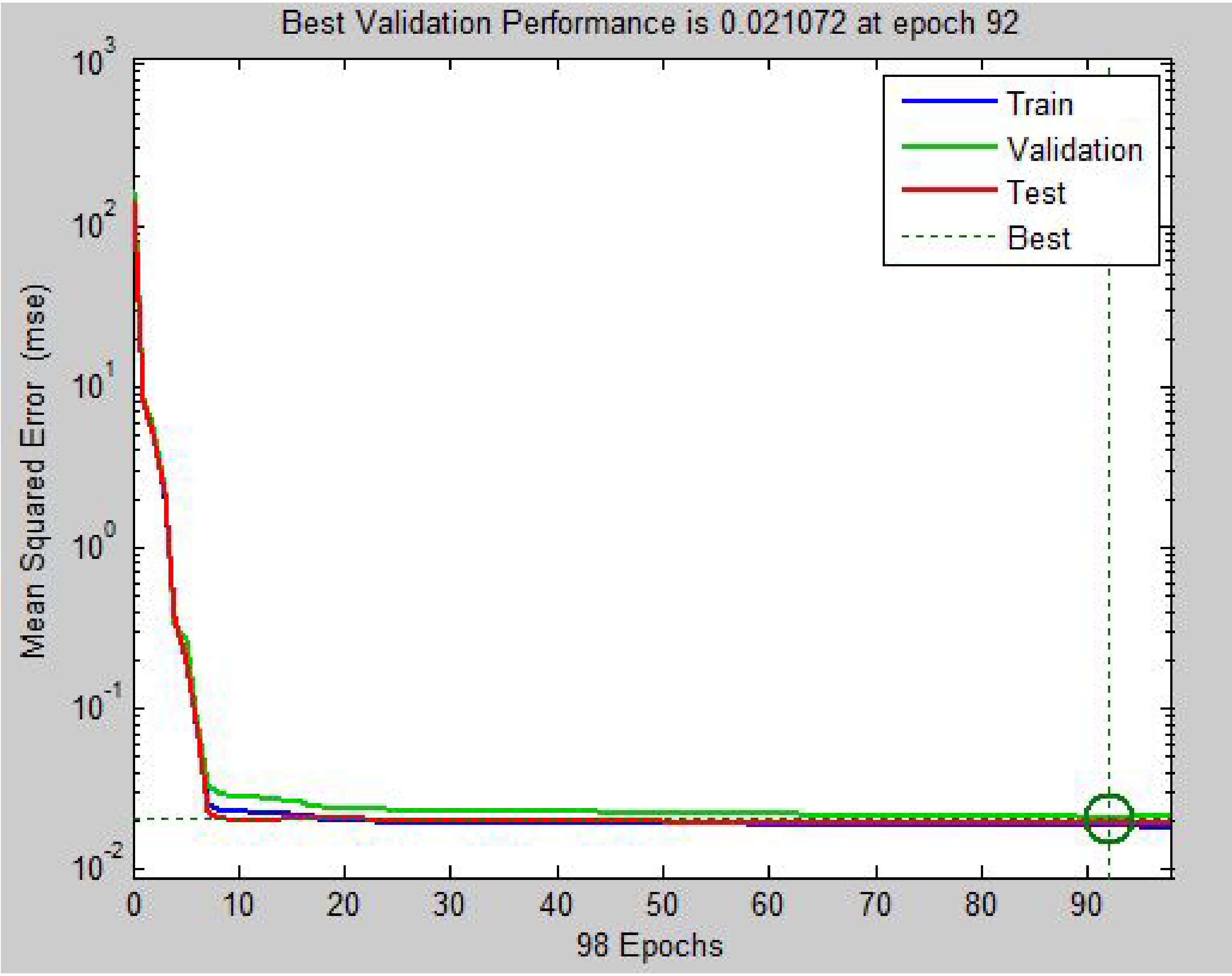
Mapping to control parameters

Supervised:

Image to (θ_1,θ_2)



Mapping to control parameters



Supervised:

Manifold (y1,y2) to (θ_1,θ_2)

Formal and Naïve Representations

Formal Representation

{

dofs - 2

parameters - **θ**: Θ_1, Θ_2 $\Theta \in Q \subseteq S^1 x S^1$

forward mapping - workspace(Θ_1, Θ_2): $Q \rightarrow I_m \subset R^D$,

I_m: *image space*

inverse mapping - Cspace(I): $I_m \rightarrow Q$

motion plan - wspace$(I_s, I_g) \rightarrow I_s, I_1, I_2 \ldots . I_n, I_g$

motion plan - Cspace$(q_s, q_g) \rightarrow q_s, q_1, q_2, \ldots, q_n, q_g$

}

Discovered (Naïve) Representation

Naive Representation

{

dimensionality : 2

discovered parameters: $\boldsymbol{\phi}:\phi_1,\phi_2 \in Y \subseteq S^1 x S^1$

forward mapping - workspace(ϕ_1,ϕ_2): $Y \rightarrow I_m \subset R^D$,

I_m: *image space*

motion plan - wspace$(I_s,I_g) \rightarrow I_s,I_1,I_2....I_n,I_g$

motion plan - Cspace$(q_s,q_g) \rightarrow q_s,q_1,q_2,....,q_n,q_g$

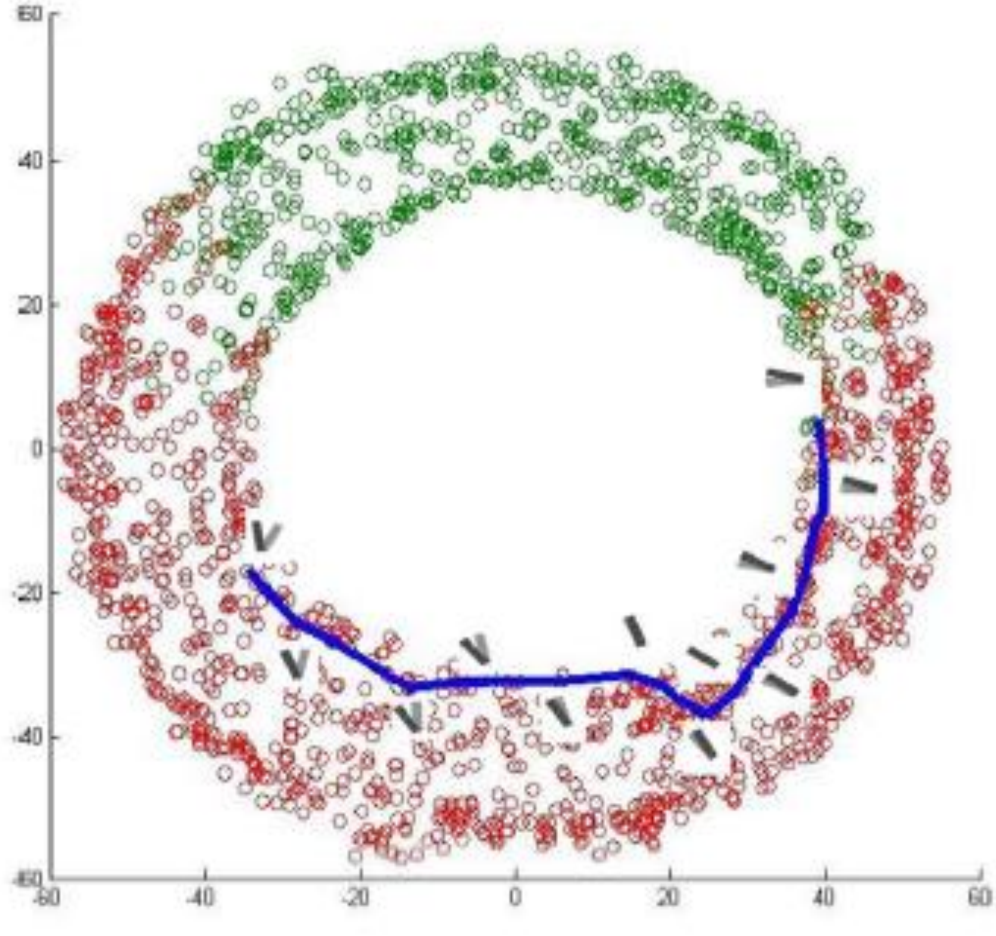
}

Motion Planning

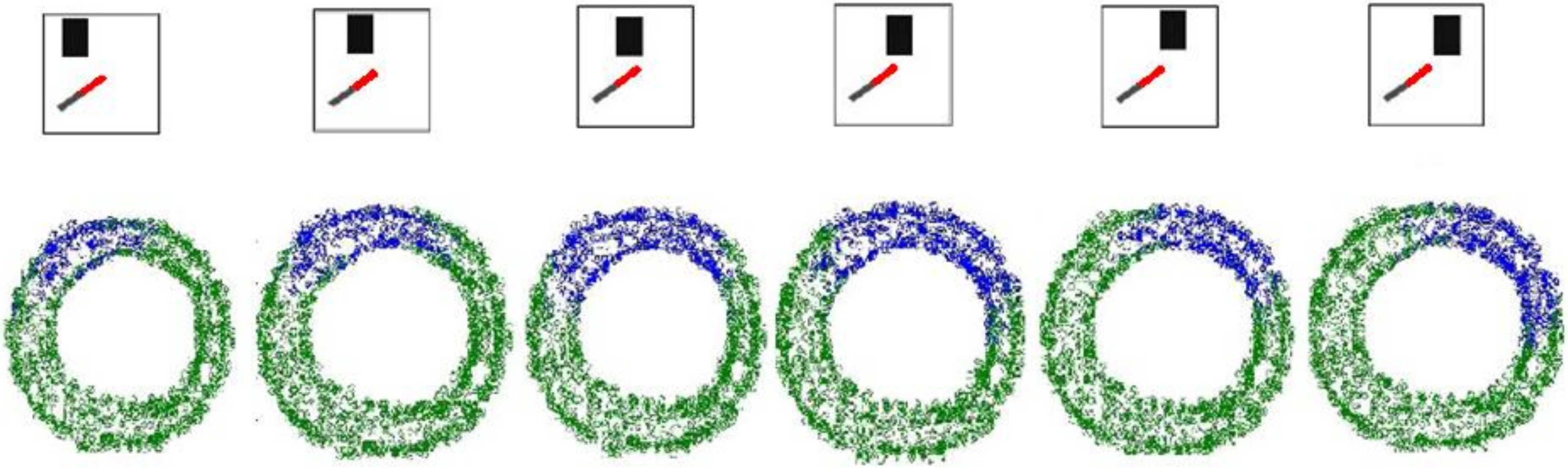
Motion planning

Given start / goal image, map to manifold using local interpolation

Use k-nn connectivity in manifold as "roadmap" for motion planning

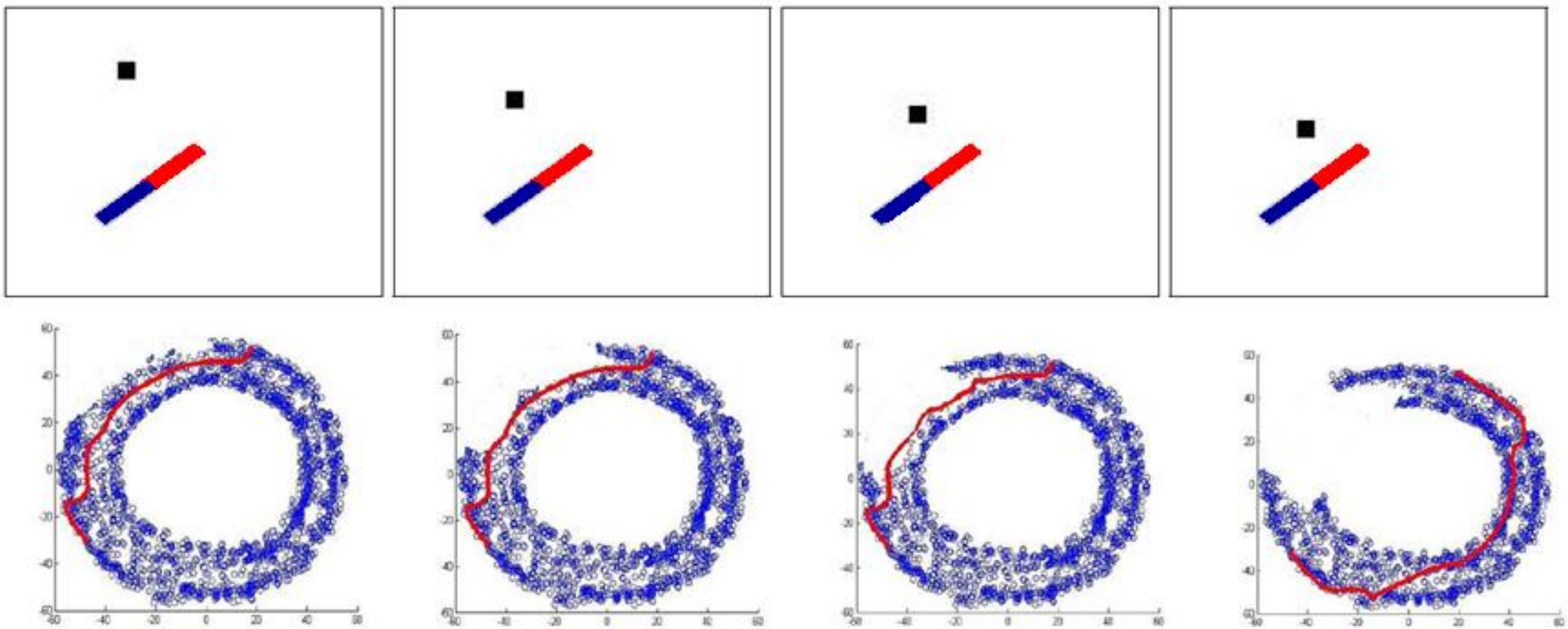


Obstacle modeling by node deletion



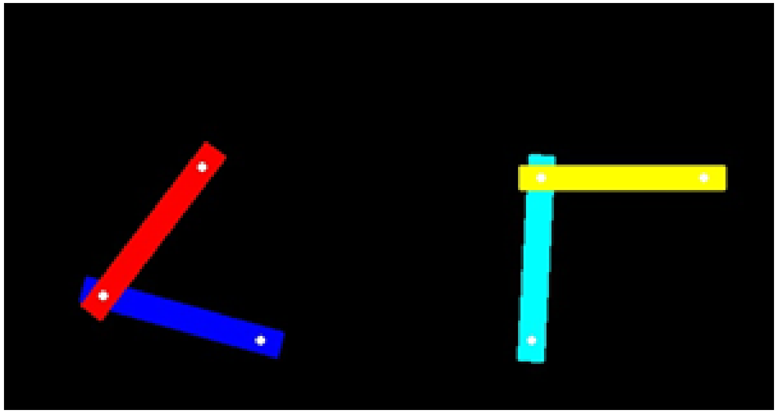
If obstacle intersects robot in image space →
delete corresponding nodes from "visual roadmap"

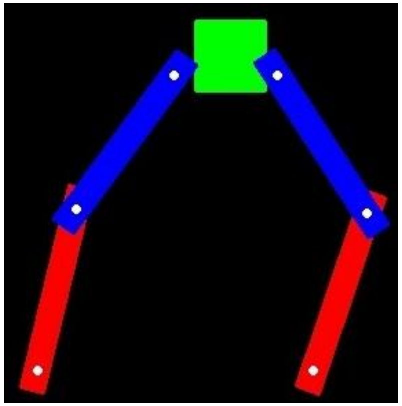
Path planning as obstacle moves

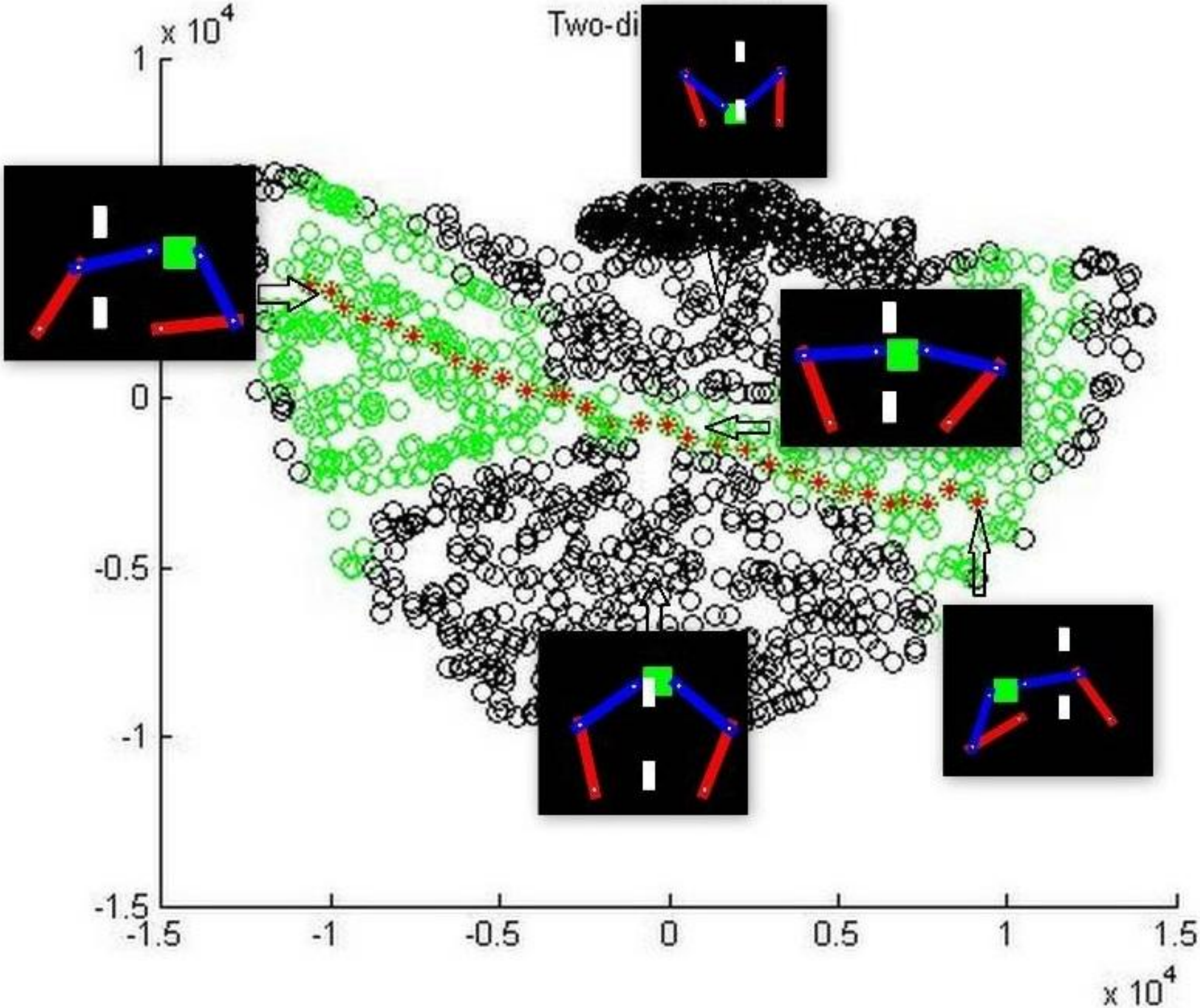


Constrained Motion

Obstacle modeling by node deletion

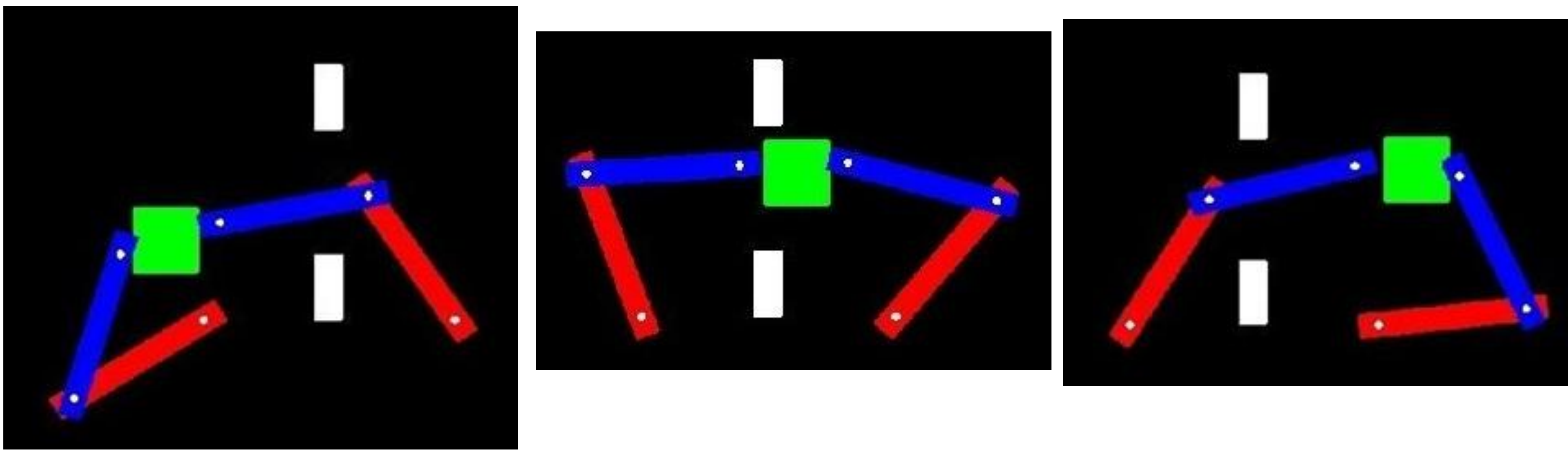




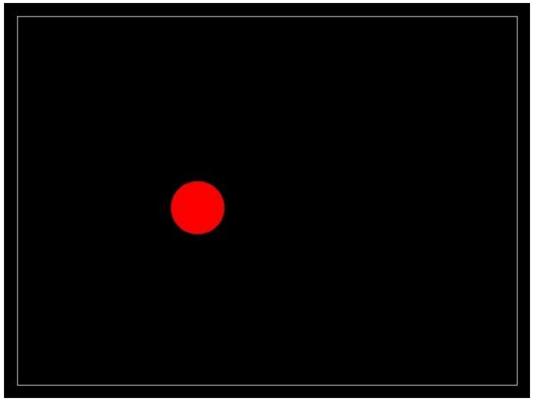
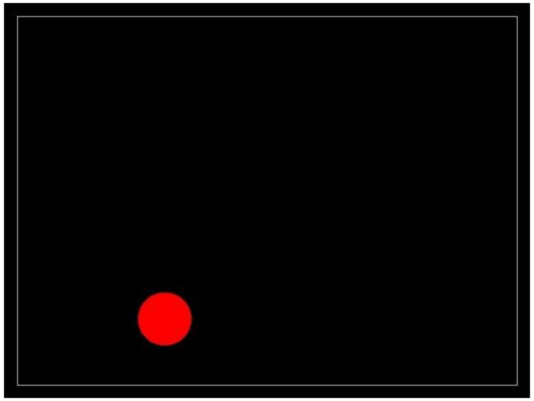


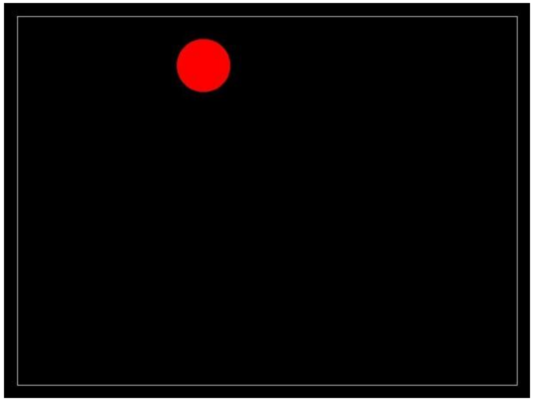
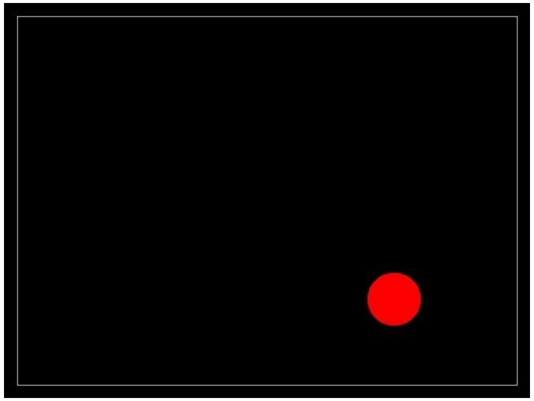
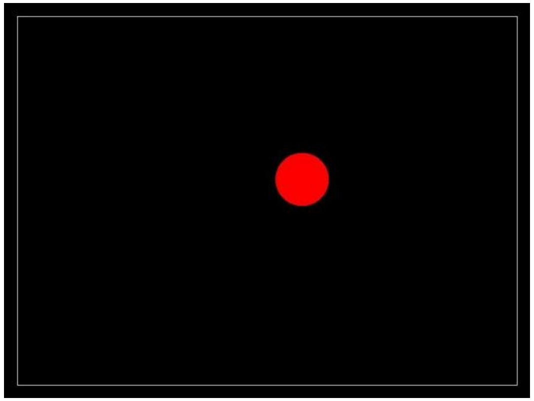
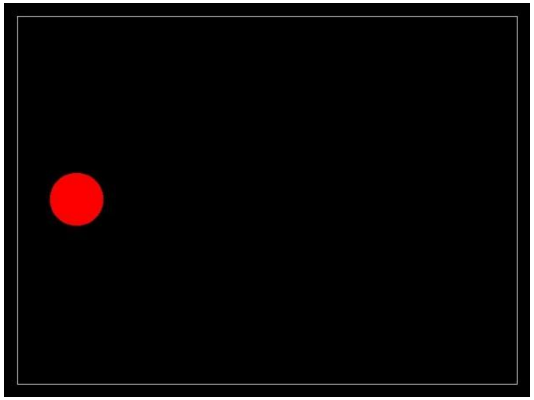
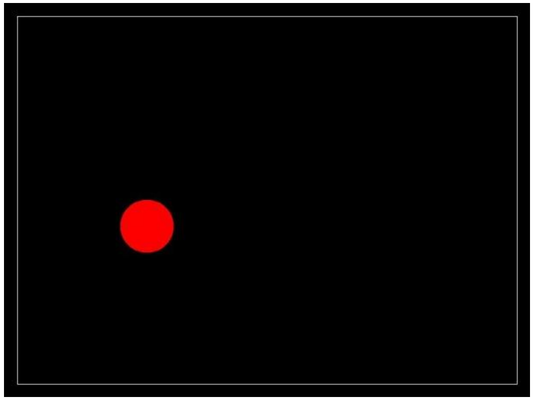
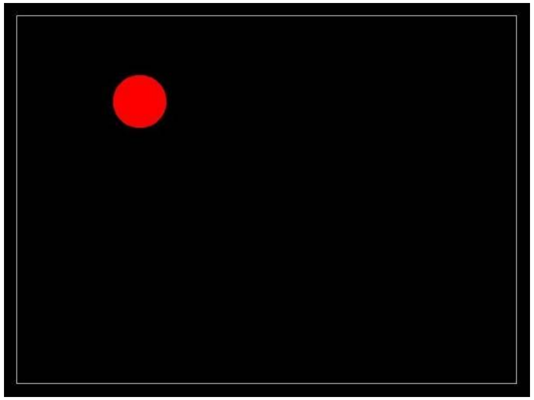
Two-di
x 10^4
1
0
-0.5
-1
-1.5
-1.5
-1
-0.5
0
0.5
1
1.5
x 10^4

Obstacle modeling by node deletion

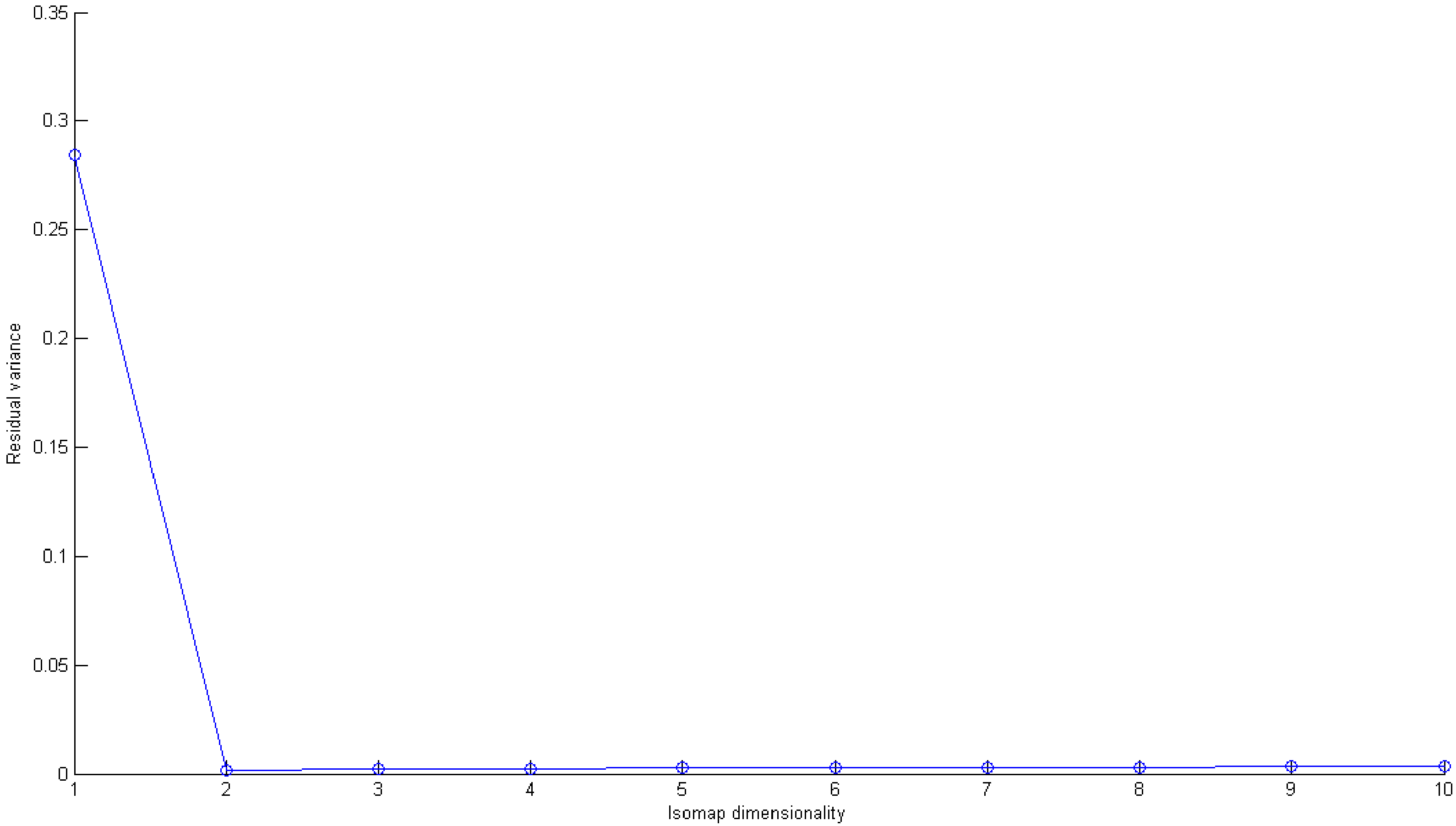


Mobile robots

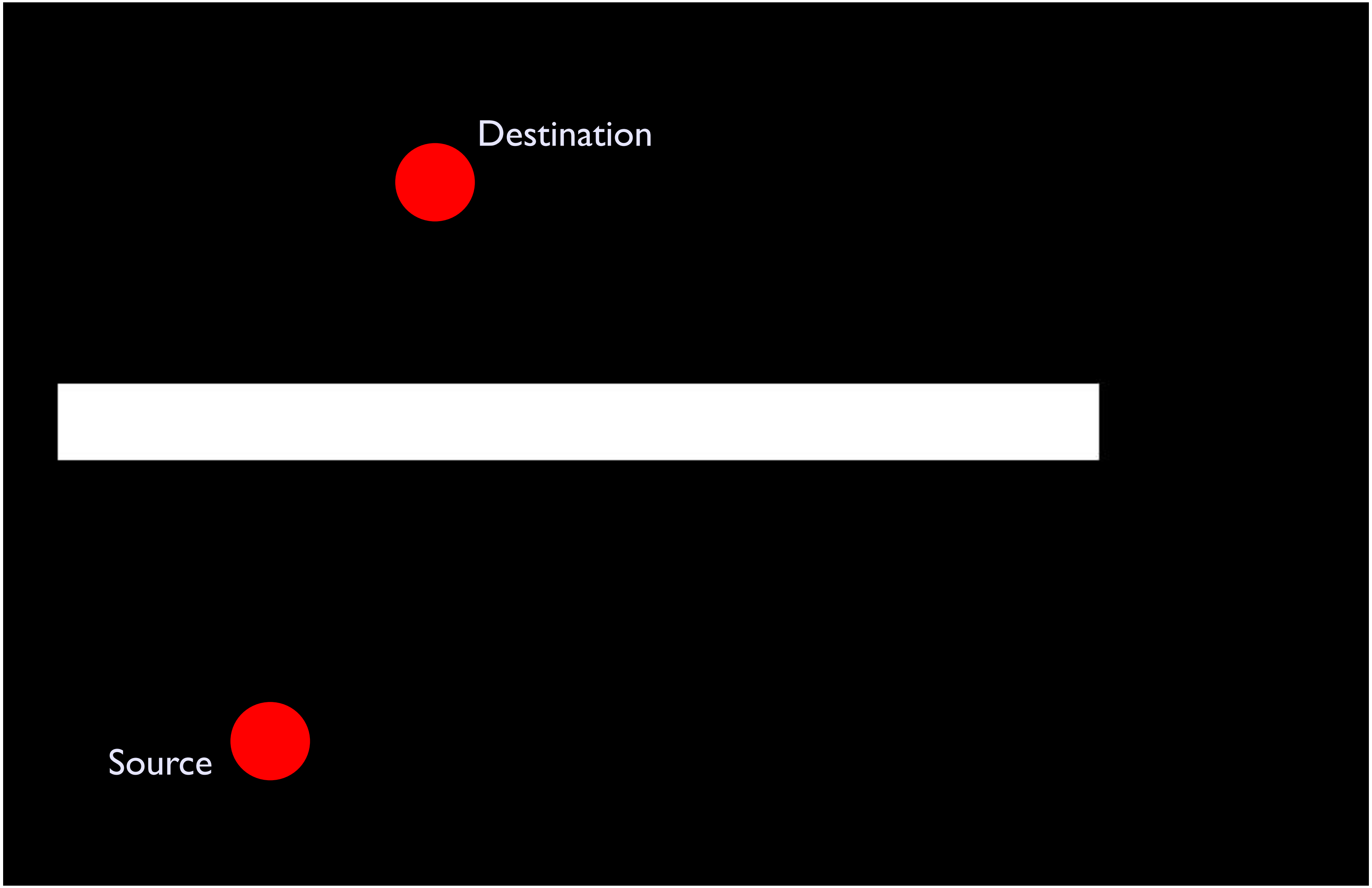



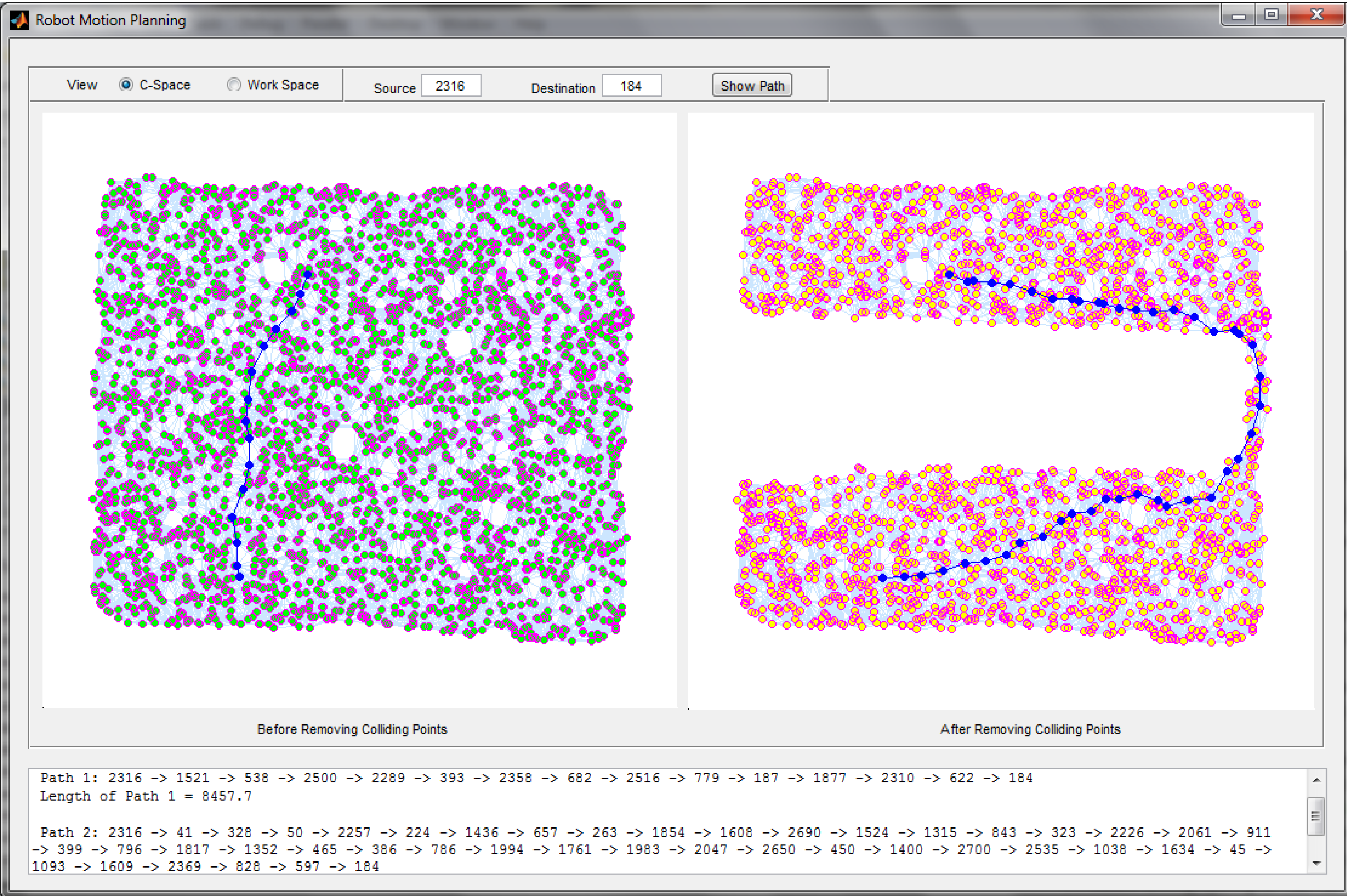
Residual error : disk robot

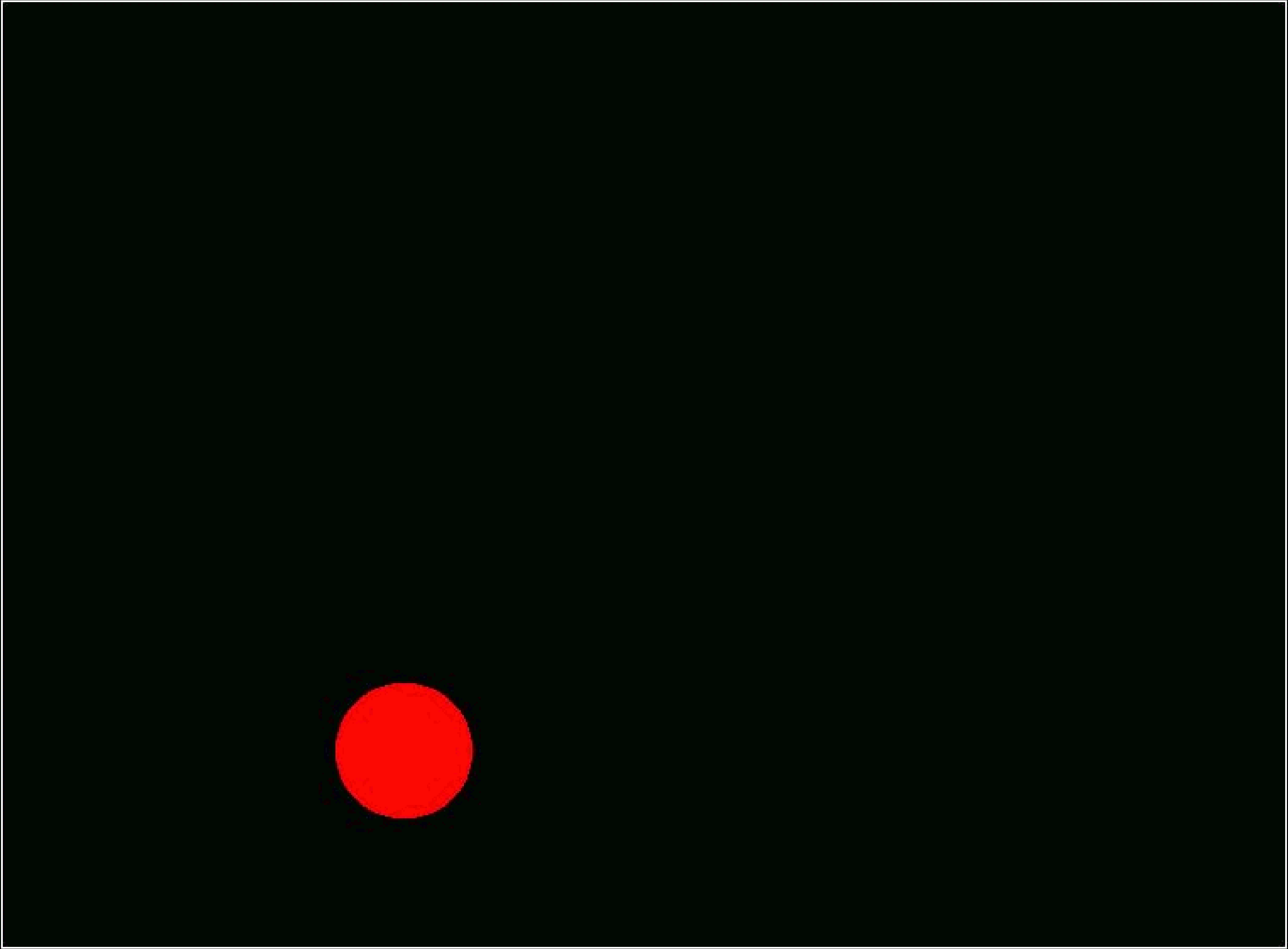


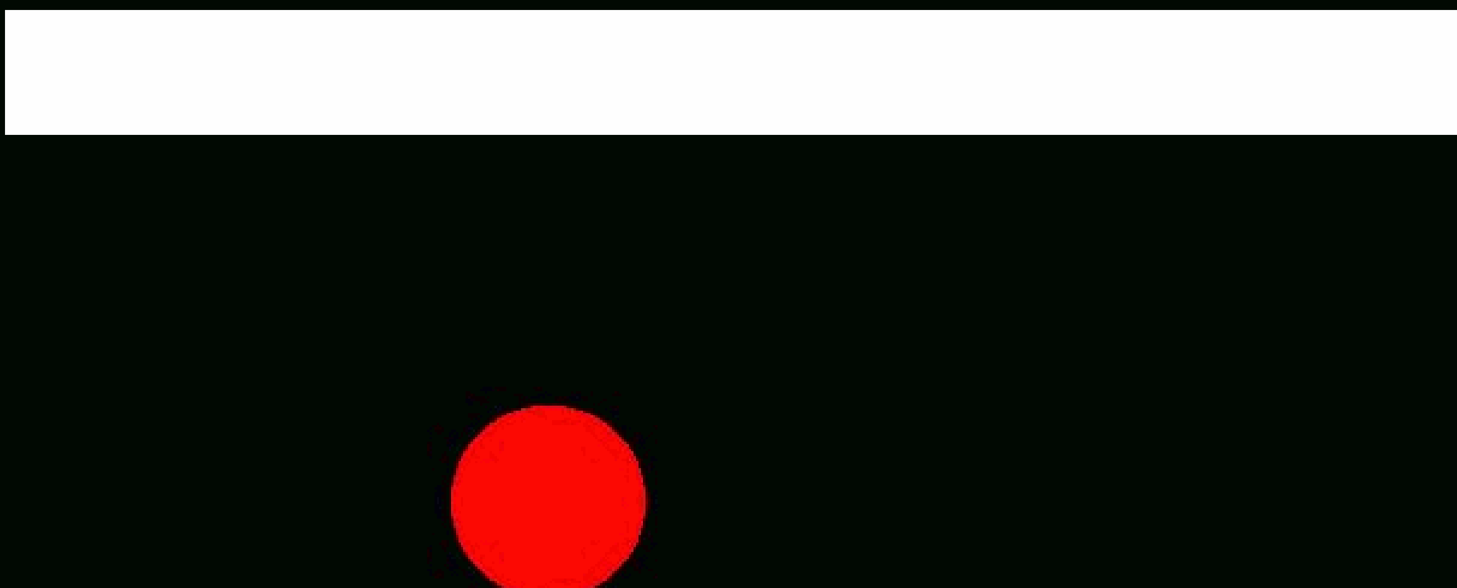
Robot Motion Planning



Path Planning Interface

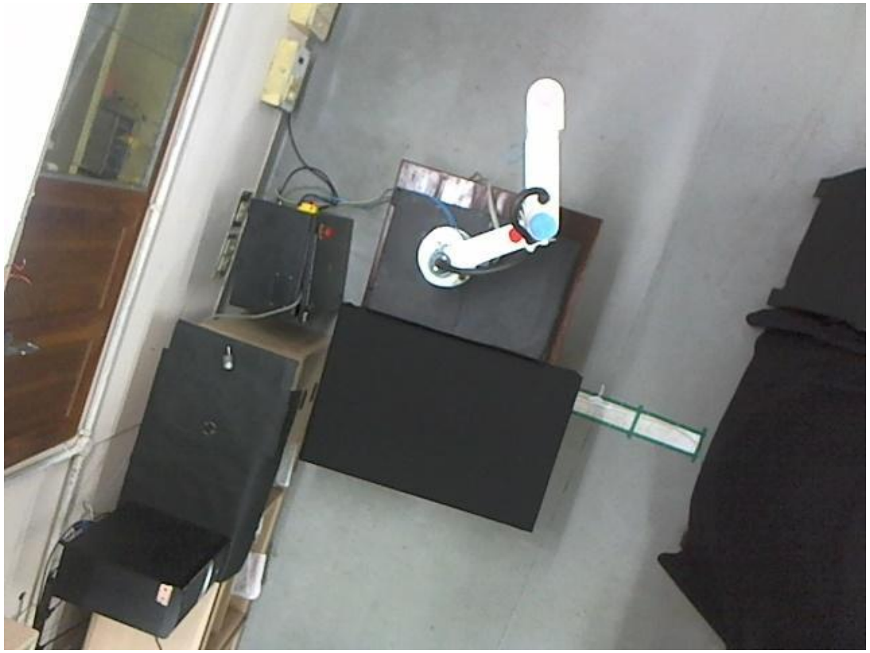




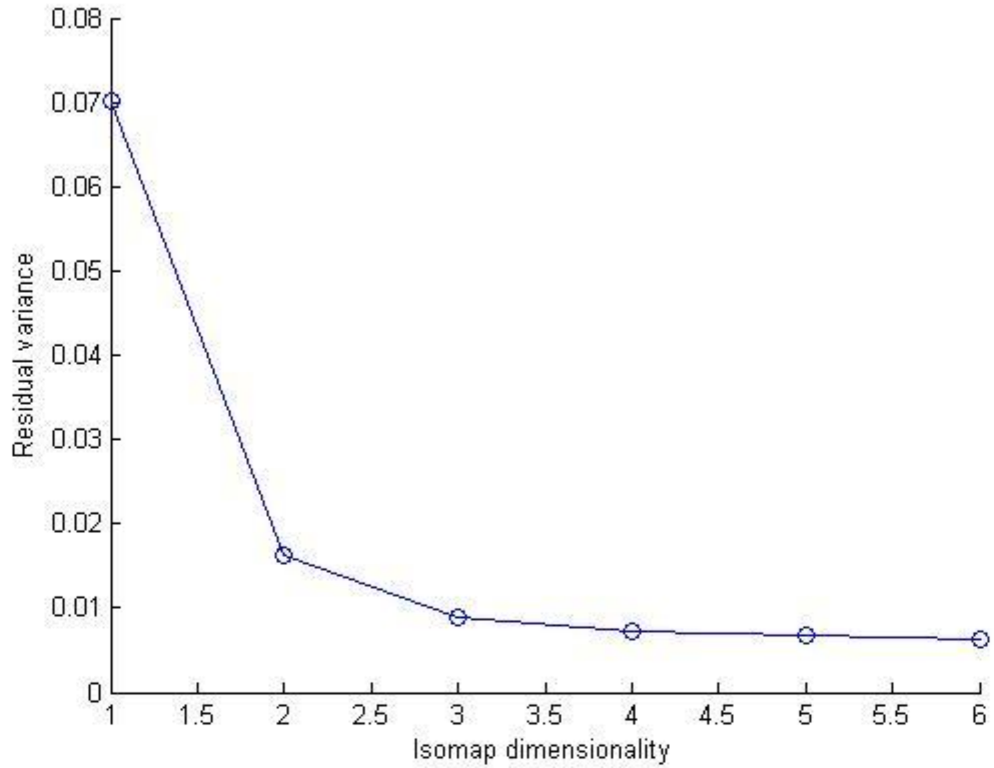


Real robots

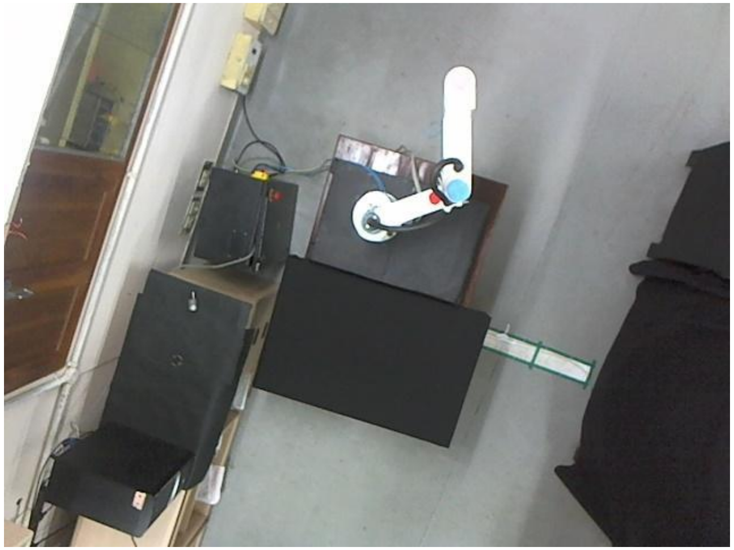
SCARA arm



SCARA arm : degrees of freedom



Background Subtraction : Robot

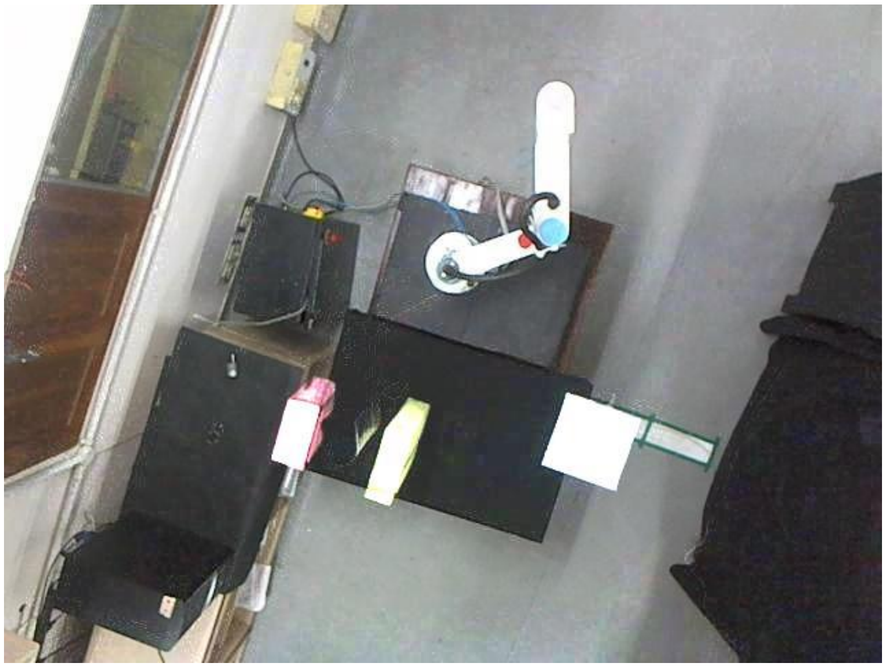


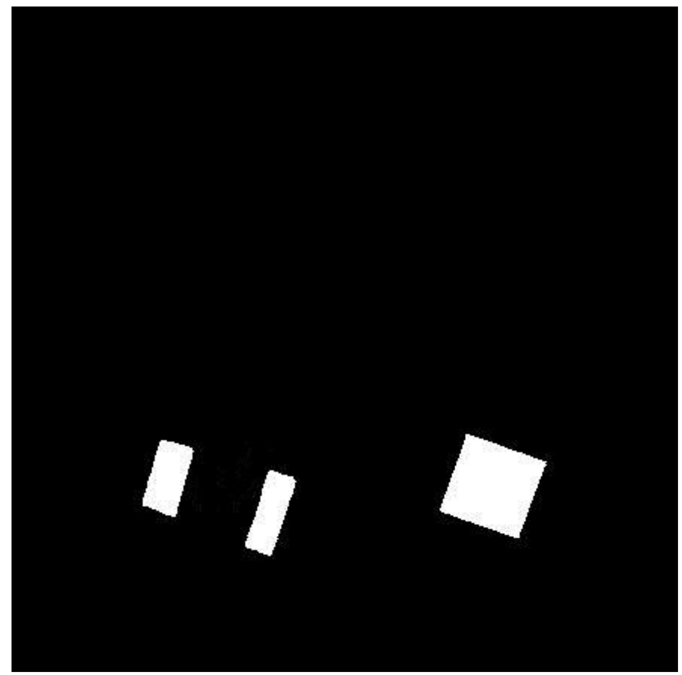
image

foreground (moving part)

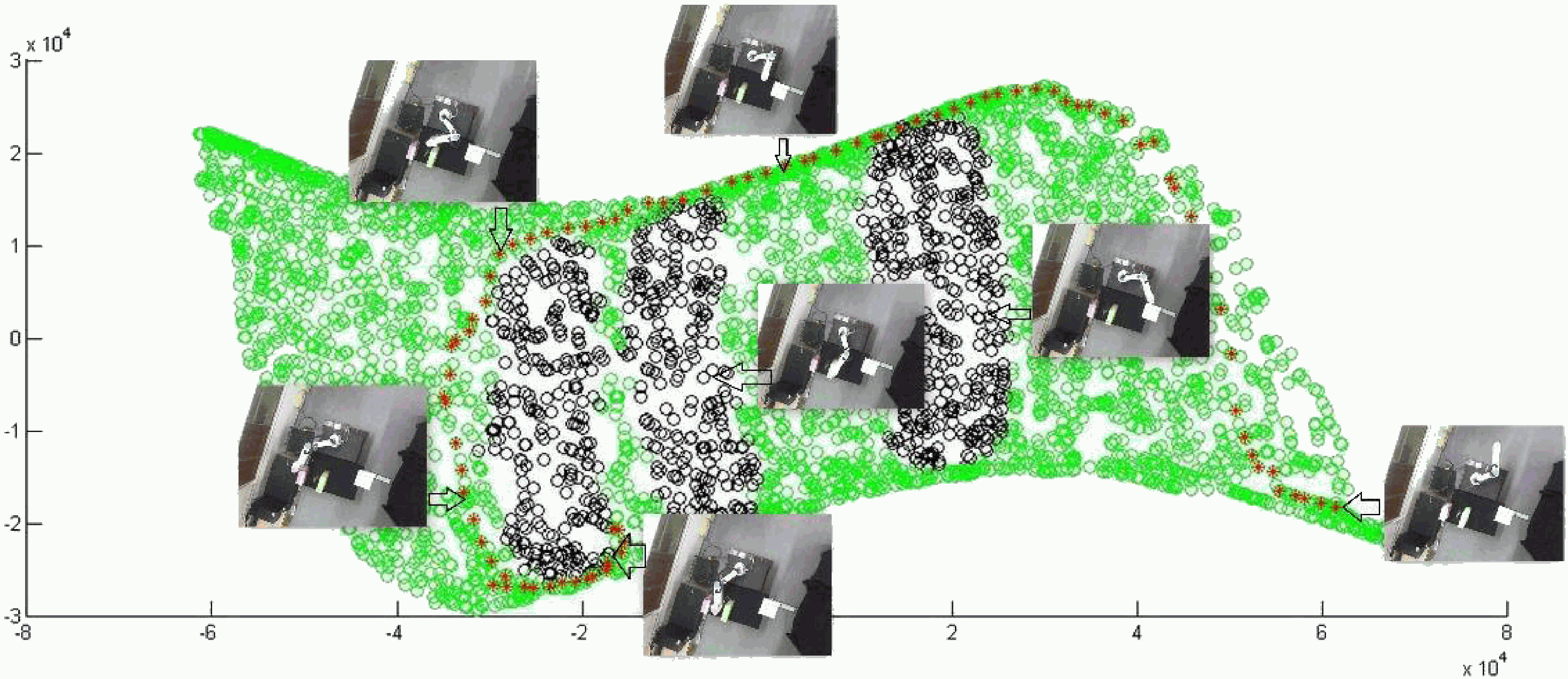
learned background

Background Subtraction : Obstacle





Visual Configuration Space



Application to Graphics

Head Motion

Conclusion

Beyond Geometry

- Geometry is not everything!!
- Real robots have limitations on acceleration owing to torque / inertia → **Dynamics**
- **Learning** to plan motions?

  - Babies learn to move arms

  - Learn low-dimensional representations of motion

- **Grasping** / Assembly : Motions along obstacle boundary

Humans and Robots

madhur ambastha cs665 2002