Planning Under Topological Constraints Using Beam-Graphs

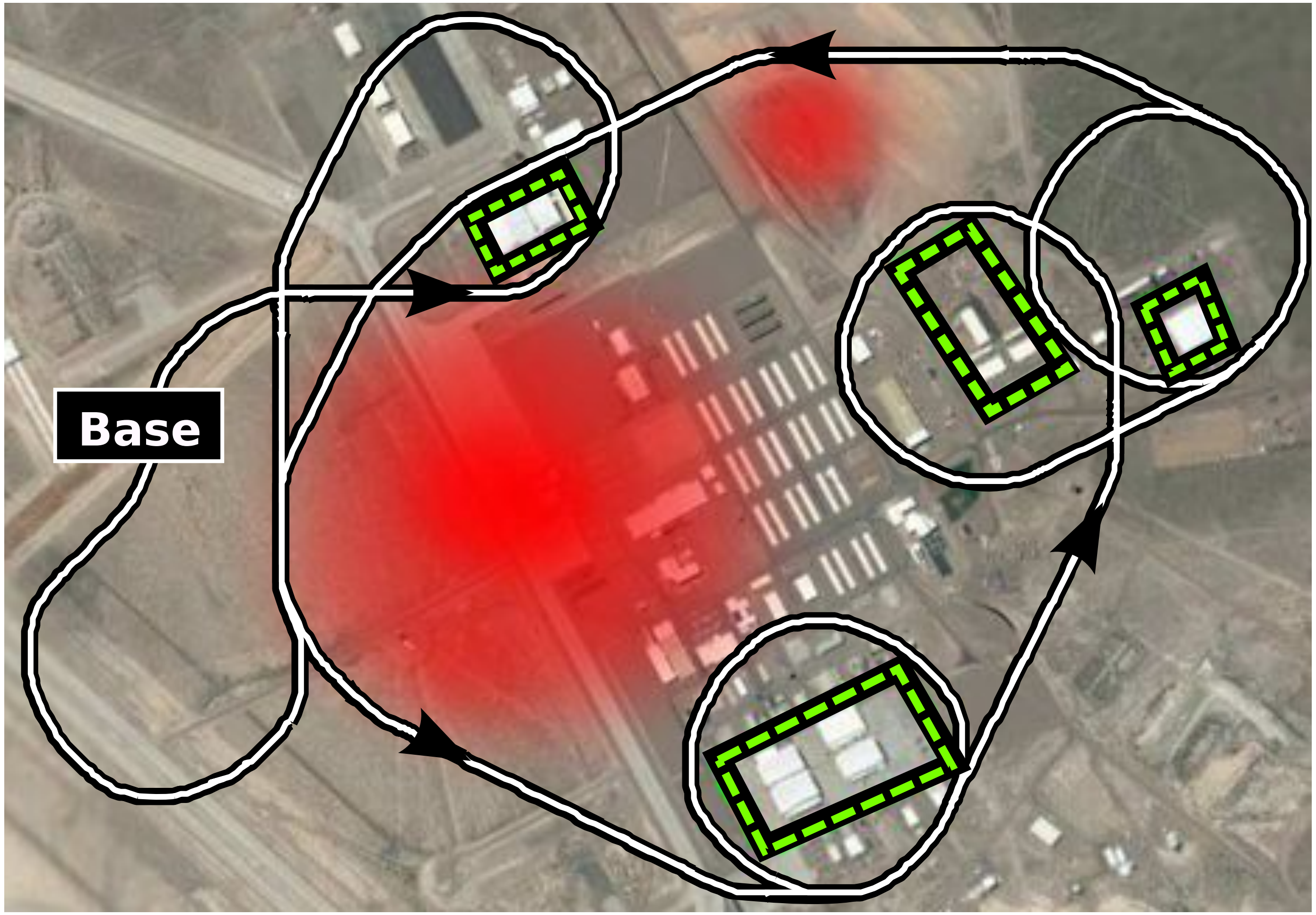
Venkatraman Narayanan, Paul Vernaza, Maxim Likhachev
and Steven M. LaValle

Robotics Institute, Carnegie Mellon University
Department of Computer Science, University of Illinois, Urbana

Carnegie Mellon
THE ROBOTICS INSTITUTE

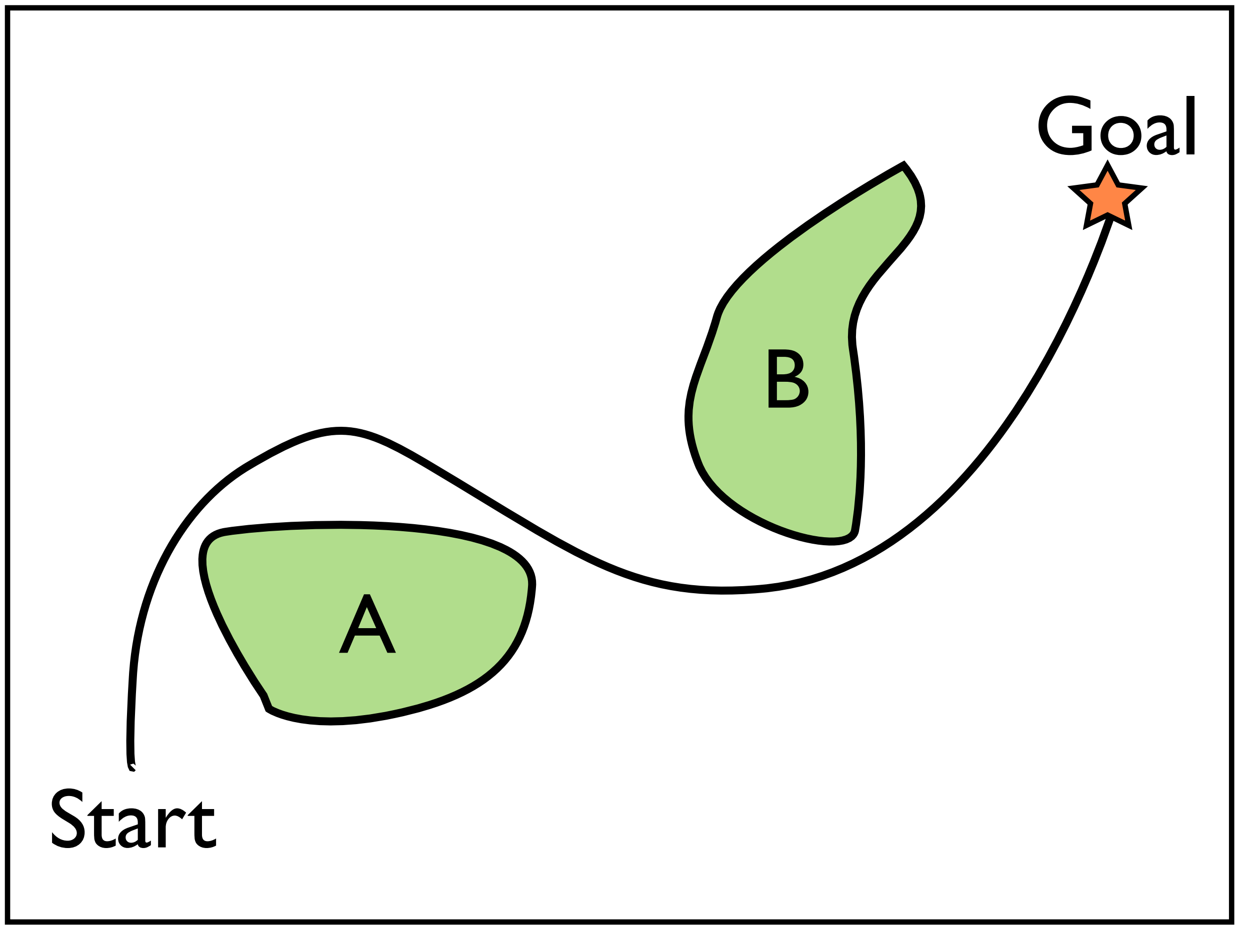
ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Topological Constraints



"Find a circuit path for an UAV such that it obtains 360° views of certain regions of interest, in a particular order"

Topological Constraints

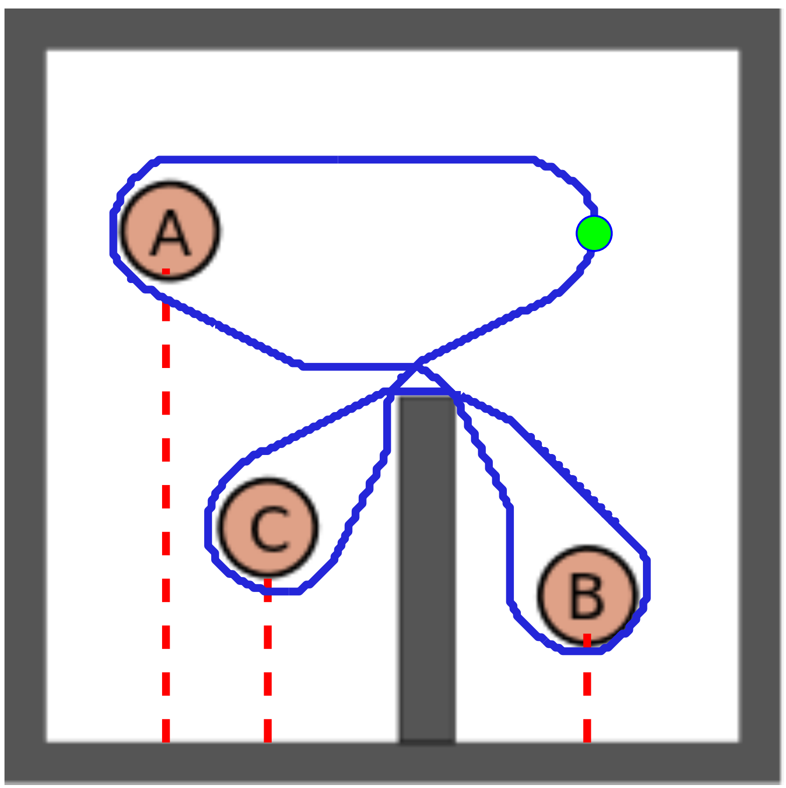


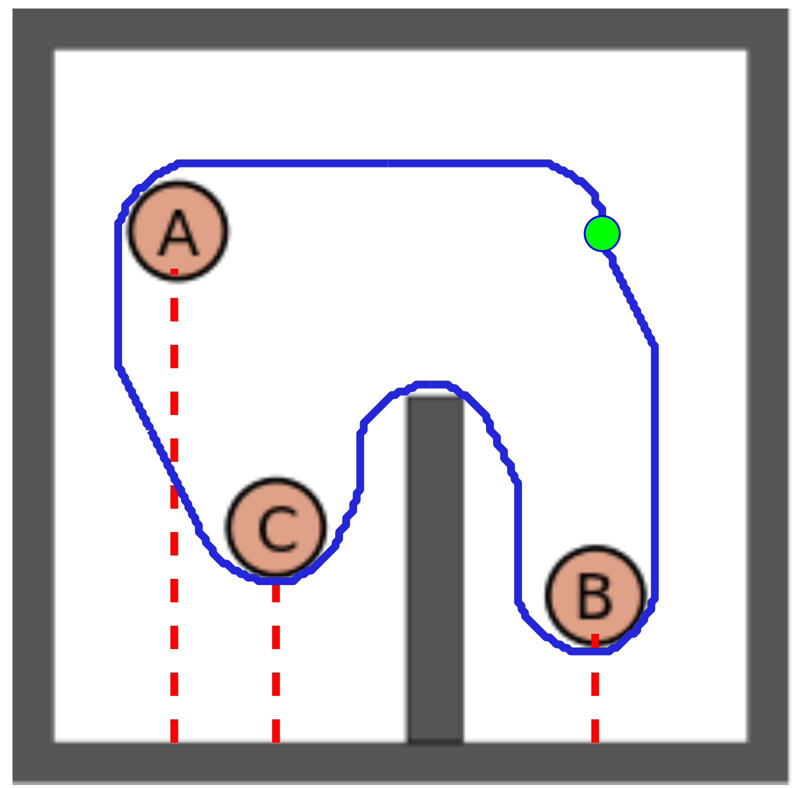
"Find the optimal path from start to goal such that it goes *above* region A and *below* region B"

Topological Constraints

Find the optimal loop that encloses the regions of interest in some *specified order*

Find the optimal loop that encloses the regions of interest in *any order*

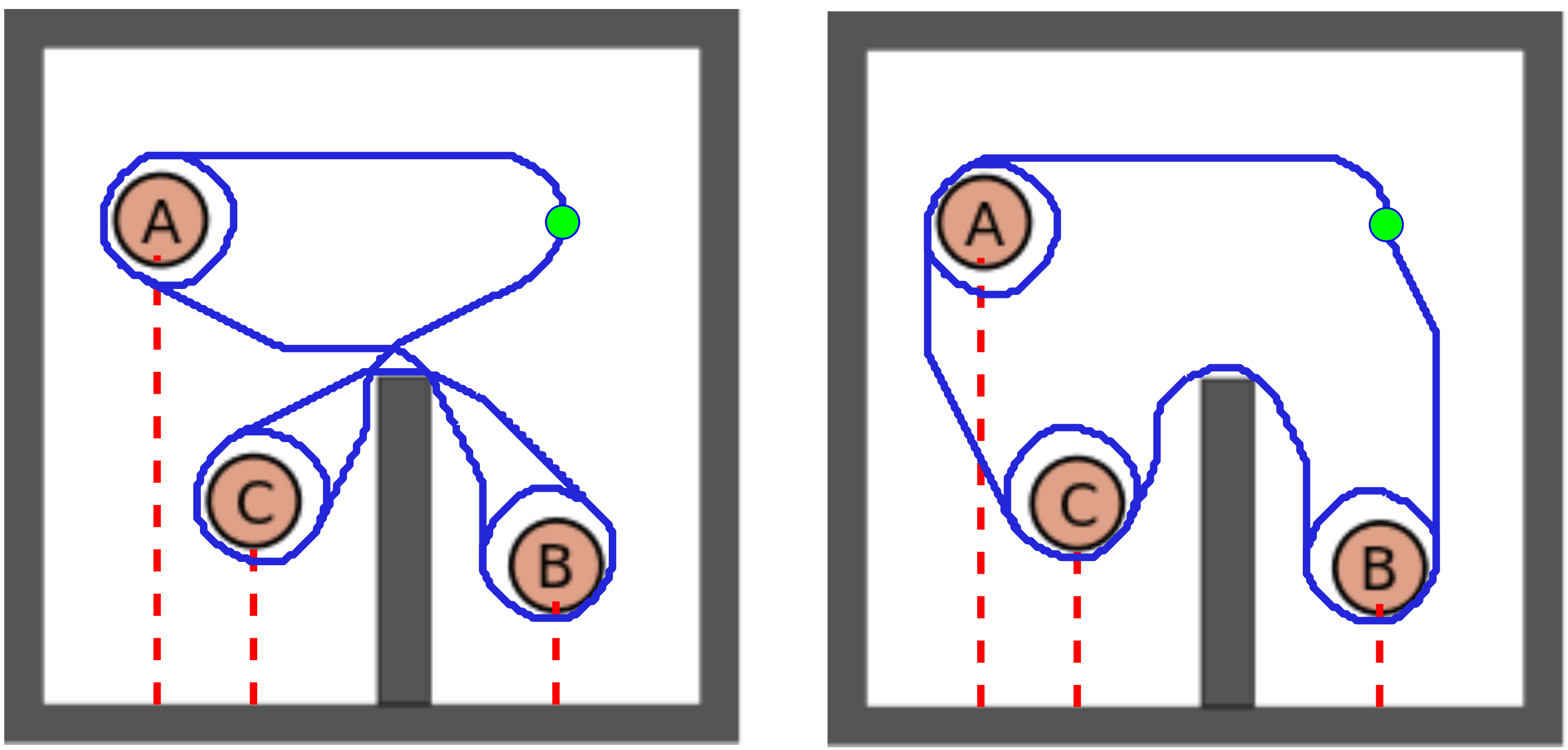




Topological Constraints

Find the optimal loop that obtains 360° views of the regions of interest in some *specified order*

Find the optimal loop that obtains 360° views of the regions of interest in *any order*



Related Work

Methods inspired by complex analysis, electromagnetic theory and flow theory:

• S. Bhattacharya, M. Likhachev, and V. Kumar. *Topological constraints in search-based robot path planning*. Autonomous Robots, 2012
• P. Vernaza, V. Narayanan, and M. Likhachev. *Efficiently finding optimal winding-constrained loops in the plane*. In Proceedings of Robotics: Science and Systems, 2012
• H. Gong, J. Sim, M. Likhachev, and J. Shi. *Multihypothesis motion planning for visual object tracking*. In Thirteenth International Conference on Computer Vision, 2011

Related Work

Methods inspired by complex analysis, electromagnetic theory and flow theory:

- S. Bhattacharya, M. Likhachev, and V. Kumar. *Topological constraints in search-based robot path planning*. Autonomous Robots, 2012
- P. Vernaza, V. Narayanan, and M. Likhachev. *Efficiently finding optimal winding-constrained loops in the plane*. In Proceedings of Robotics: Science and Systems, 2012
- H. Gong, J. Sim, M. Likhachev, and J. Shi. *Multihypothesis motion planning for visual object tracking*. In Thirteenth International Conference on Computer Vision, 2011

Designed for "homology" constraints. Cannot handle general topology constraints

Mathematically and numerically intensive

Related Work

Computational geometry approaches:

- S. Cabello, Y. Liu, A. Mantler, and J. Snoeyink. Testing homotopy for paths in the plane. In Proceedings of the eighteenth annual symposium on Computational Geometry, 2002
- A. Efrat, S. Kobourov, and A. Lubiw. Computing homotopic shortest paths efficiently. Computational Geometry

Related Work

Computational geometry approaches:

- S. Cabello, Y. Liu, A. Mantler, and J. Snoeyink. Testing homotopy for paths in the plane. In Proceedings of the eighteenth annual symposium on Computational Geometry, 2002
- A. Efrat, S. Kobourov, and A. Lubiw. Computing homotopic shortest paths efficiently. Computational Geometry

Not easily applicable to robotics--difficult to incorporate kinodynamic constraints

Cannot handle arbitrary cost functions

Related Work

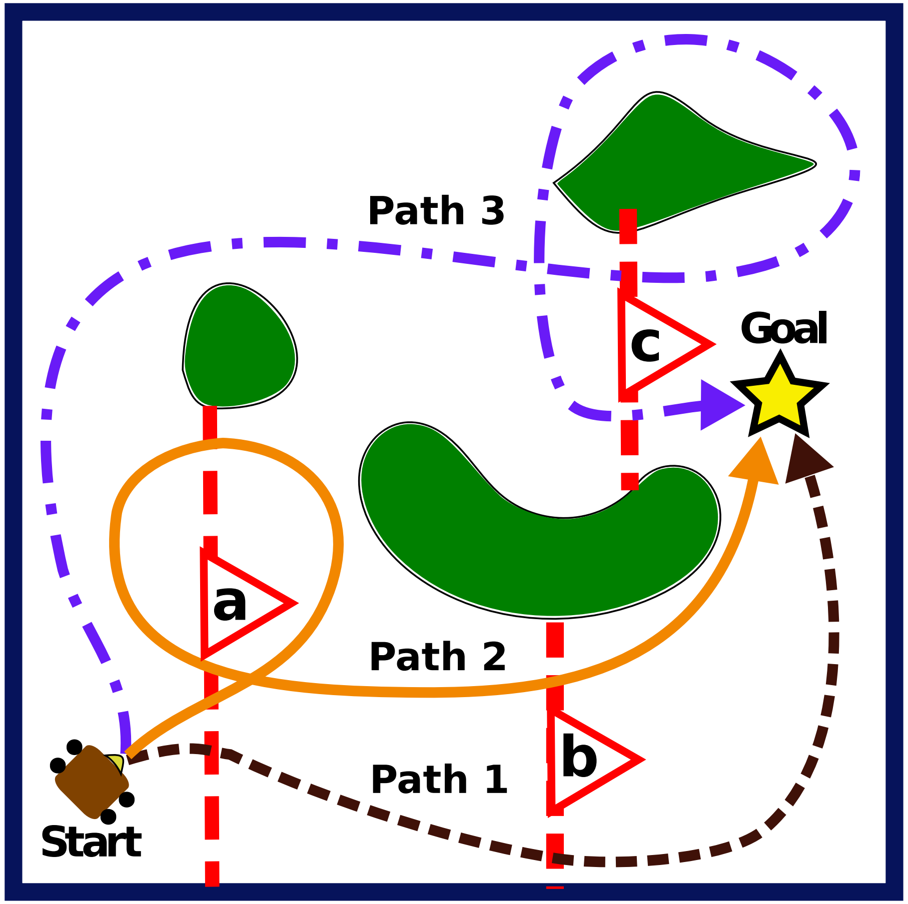
Idea of 'virtual beams':

B. Tovar, F. Cohen, and S. M. LaValle. *Sensor beams, obstacles, and possible paths*. In G. Chirikjian, H. Choset, M. Morales, and T. Murphey, editors, Algorithmic Foundations of Robotics, VIII

"Virtual beams" construction - captures topological information, intuitive

Our work shows how virtual beams can be integrated into existing search-based motion planning frameworks for robotics

Beam-Graph Construction



Path 1: ab

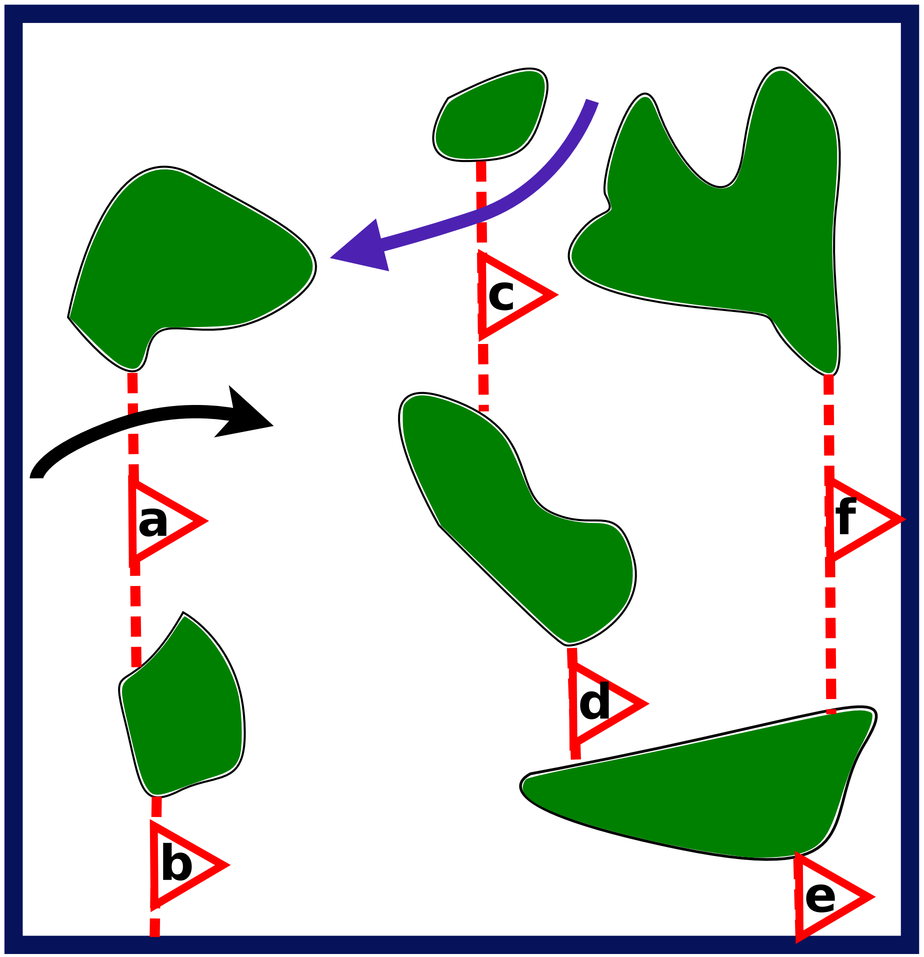
Path 2: aa'ab

Path 3: cc

- Sequences of beam-crossings capture 'topological' structure
- Hallucinate 'virtual' beams to differentiate topologically different paths

Beam-Graph Construction

- For every region of interest, construct a beam starting at the *lowest* point on the ROI, and terminating at another ROI, or the boundary
- Beams are directed



Beam crossing: **a**

Beam crossing: **c'**

Beam-Graph Construction

A state in the graph: s - [**x**(s) b(s)]

Spatial configuration of the robot

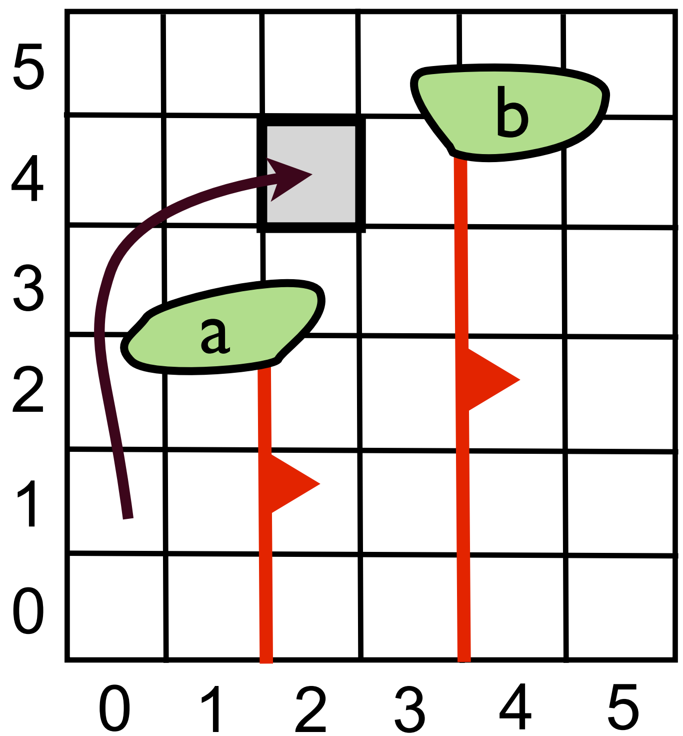
(e.g: **x**(s) = (x y θ))

History of beam crossings

(e.g: b(s) = aabc'de')

Beam-Graph Construction

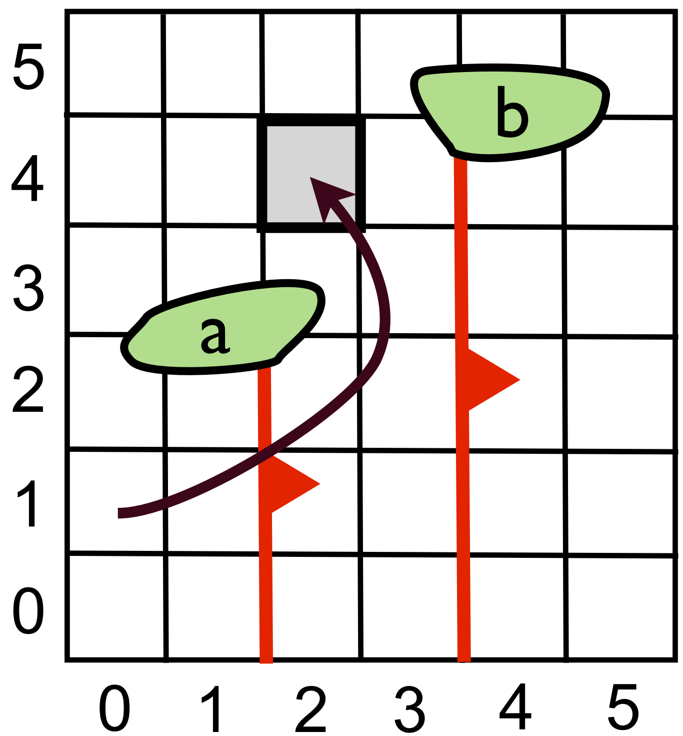
A state in the graph: s - [**x**(s) b(s)]



(2, 4, ε)

Beam-Graph Construction

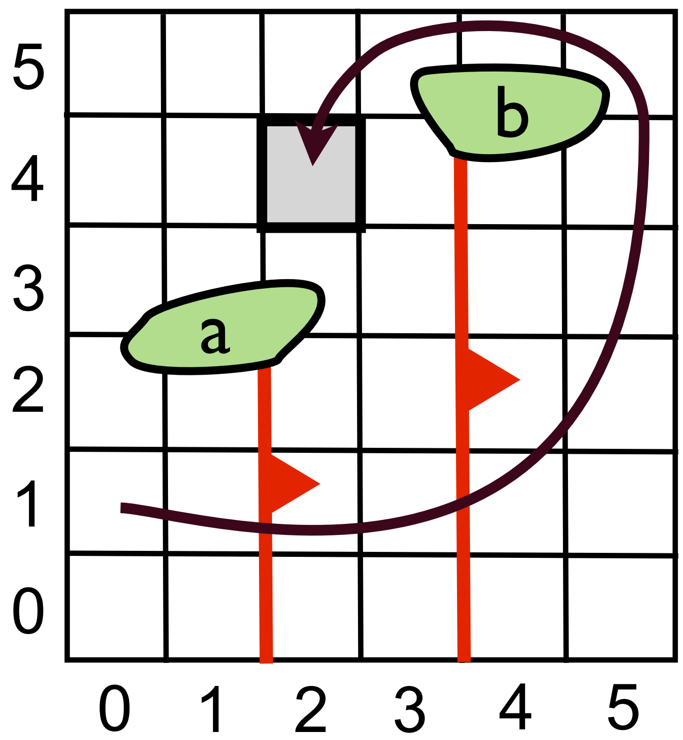
A state in the graph: s - [**x**(s) b(s)]



(2, 4, a)

Beam-Graph Construction

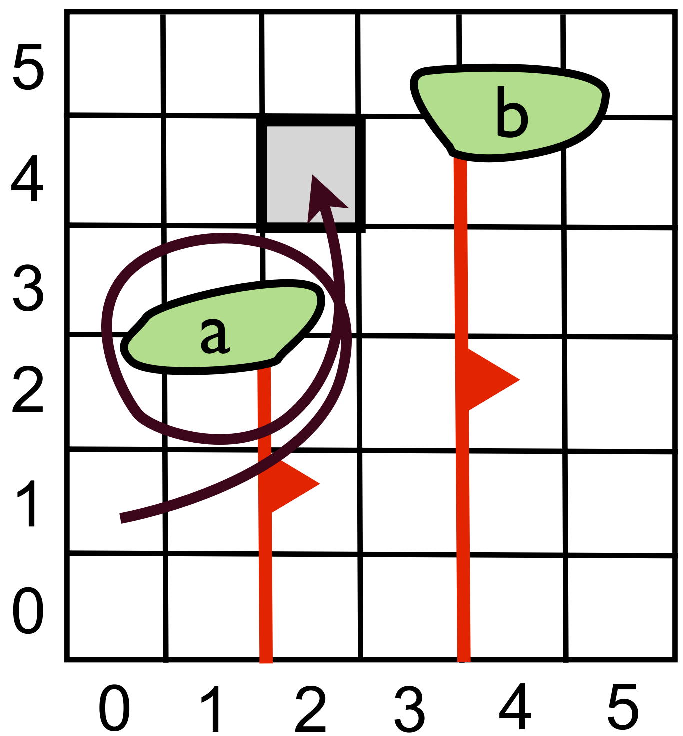
A state in the graph: s - [**x**(s) b(s)]



(2, 4, ab)

Beam-Graph Construction

A state in the graph: s - [**x**(s) b(s)]

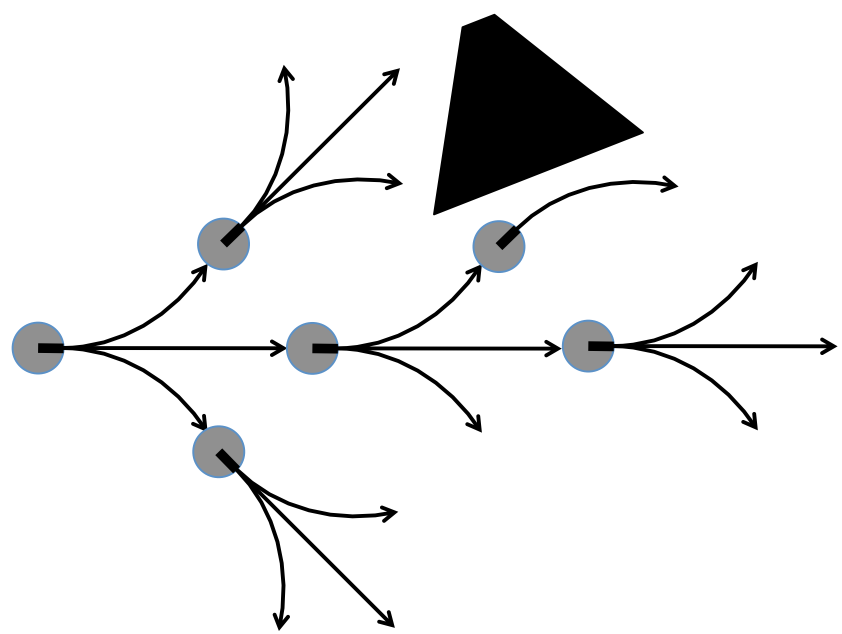


(2, 4, aa)

Beam-Graph Construction

A state s' is a successor of s iff. :

- There exists a kinodynamically feasible motion primitive between s and s'
- b(s') is a concatenation of b(s) and the string of beam crossings formed by the motion primitive, i.e, b(s') = b(s) + w(s,m)

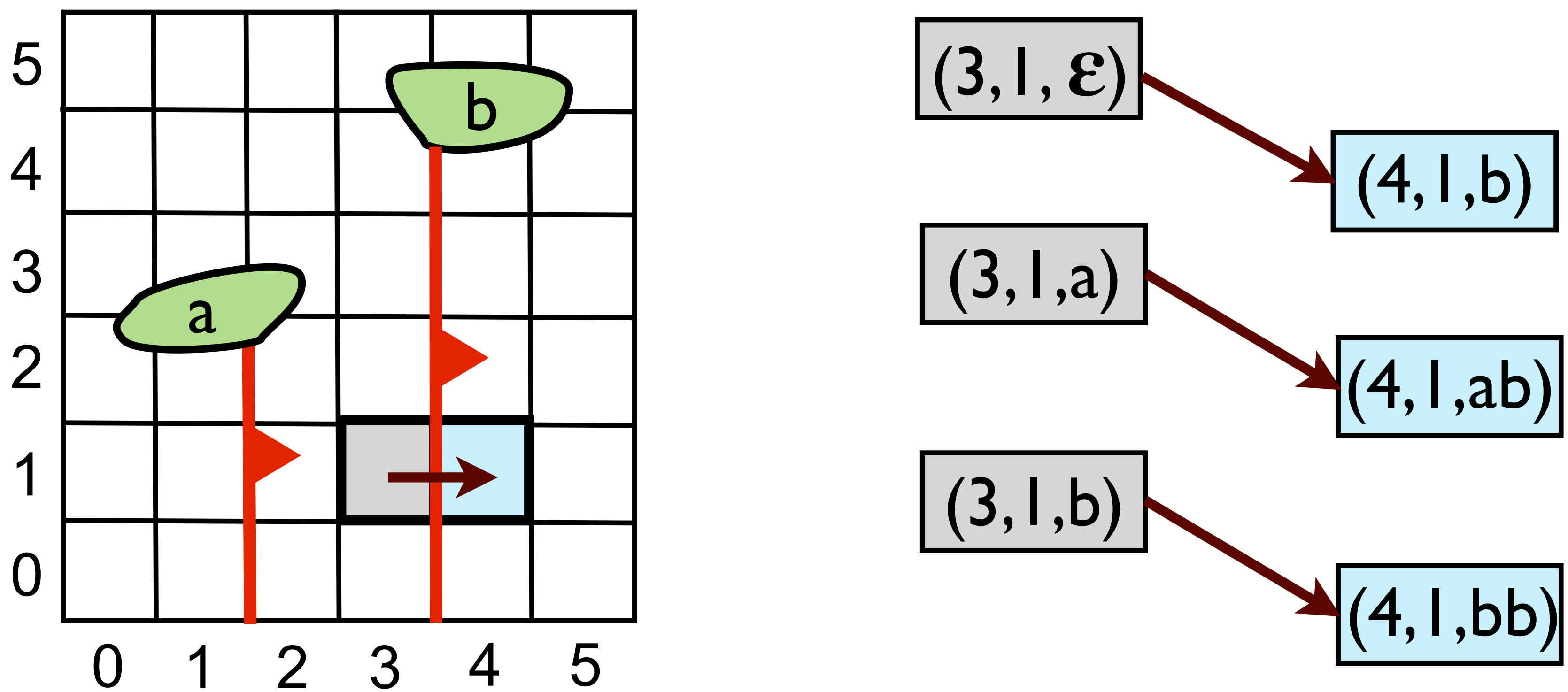


Pivtoraiko, M., Kelly, A., "Generating Near Minimal Spanning Control Sets for Constrained Motion Planning in Discrete State Spaces", Proceedings IROS 2005, Edmonton Alberta, Canada, August 2005

Beam-Graph Construction

A state s’ is a successor of s iff. :

- There exists a kinodynamically feasible motion primitive between s and s’
- b(s’) is a concatenation of b(s) and the string of beam crossings formed by the motion primitive, i.e, b(s’) = b(s) + w(s,m)

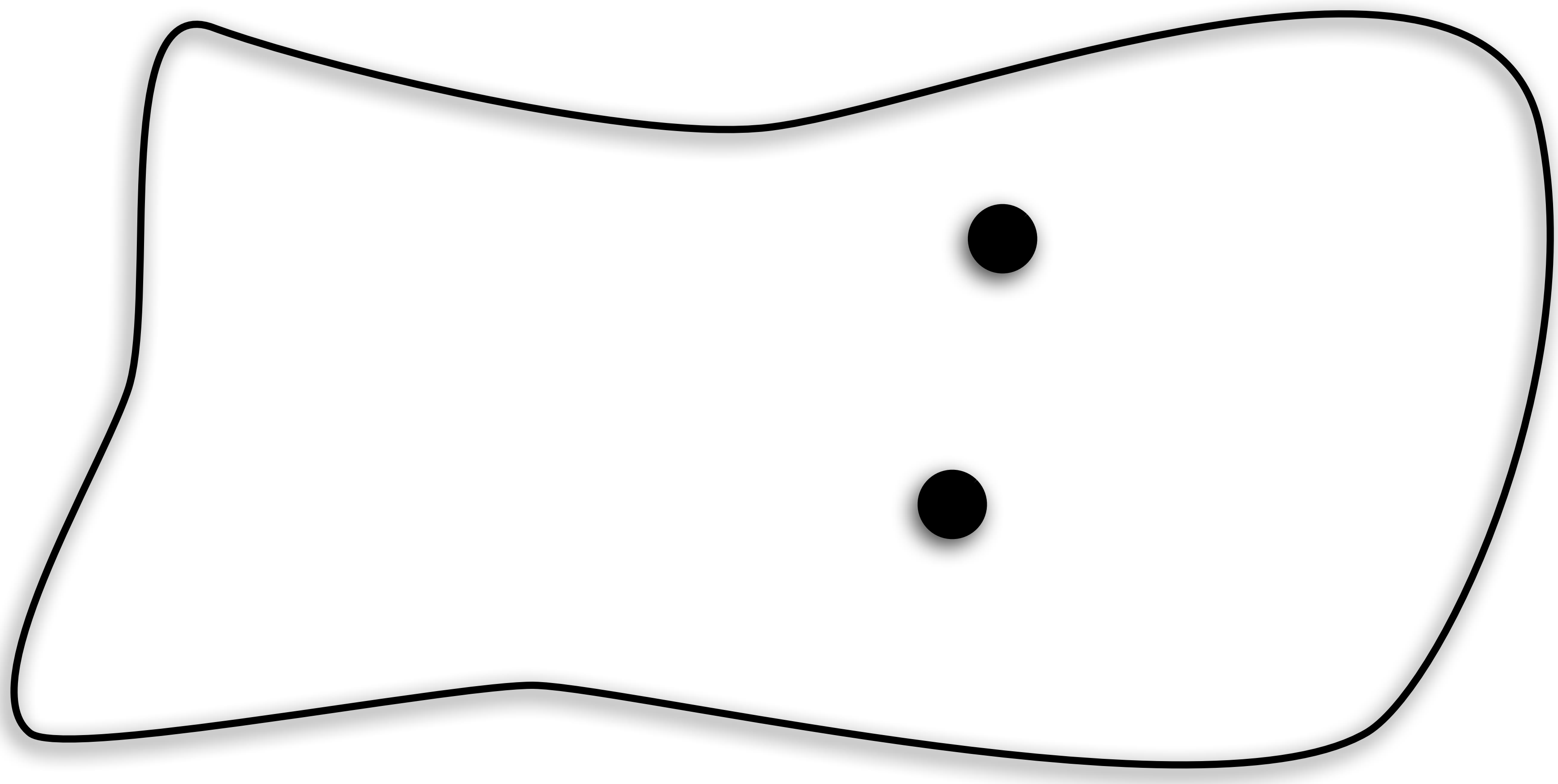


Formalizing Topological Constraints

For a topological space X, the set of all loops based at some $x_0 \in X$, is the first homotopy group or the fundamental group

Formalizing Topological Constraints

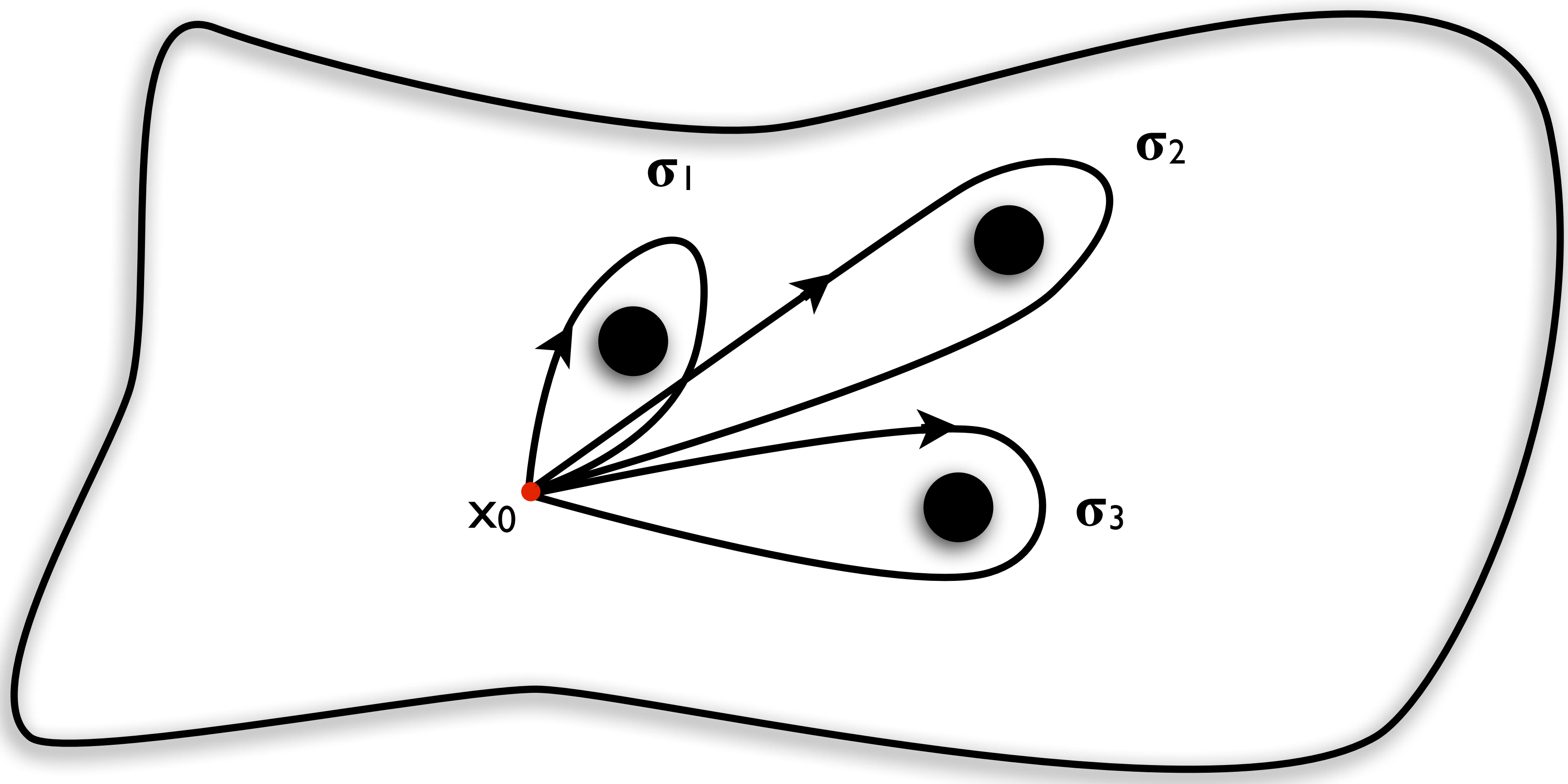
The multi-punctured plane



Formalizing Topological Constraints

The multi-punctured plane

For the multi-punctured plane, the fundamental group is the *free group on n letters*, F_n

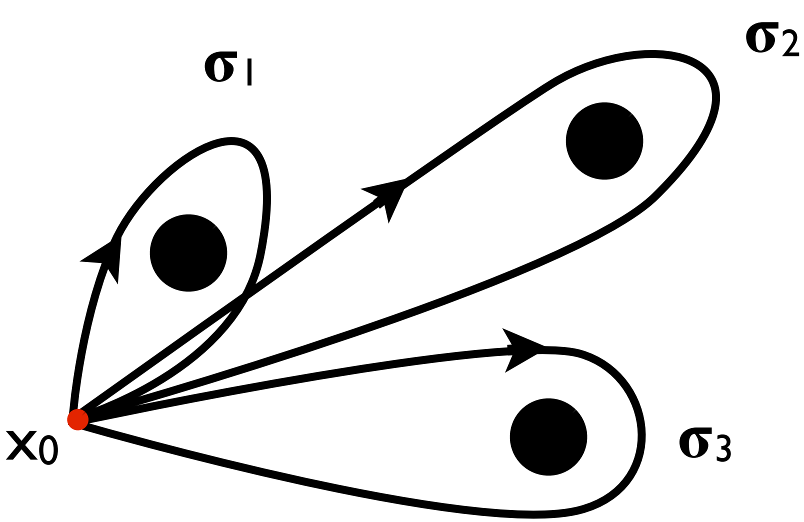


Formalizing Topological Constraints

The multi-punctured plane

For the multi-punctured plane, the fundamental group is the *free group on n letters*, F_n

"Free group" because there are no relations, other than the group axioms



$$\mathcal{L} = \{\sigma_1, \sigma_2, \sigma_3, \sigma_1^{-1}, \sigma_2^{-1}, \sigma_3^{-1}, \varepsilon\}$$

F_3 = {All words formed from letters in $\mathcal{L}$}

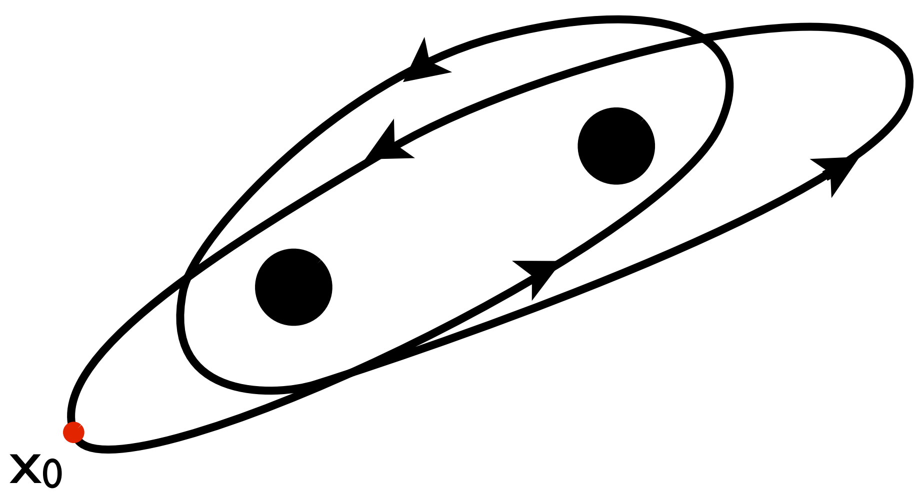
Formalizing Topological Constraints

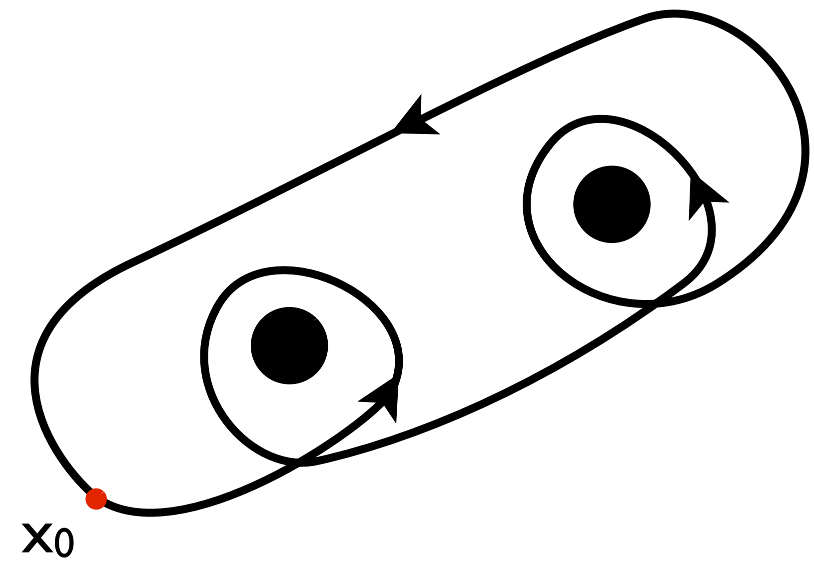
Homology

Same as the homotopy group, but also satisfies commutative property

Eg: $\sigma_1\sigma_2$ and $\sigma_2\sigma_1$ belong to the same homology group, whereas they belong to different homotopy groups

Homology constraints are winding constraints





Formalizing Topological Constraints

	Homotopy	Homology
Identity	Yes	Yes
Inverse	Yes	Yes
Commutativity	No	Yes

Identity

$\sigma_i \varepsilon = \varepsilon \sigma_i = \sigma_i$

Inverse

$\sigma_i \sigma_i^{-1} = \sigma_i^{-1} \sigma_i = \varepsilon$

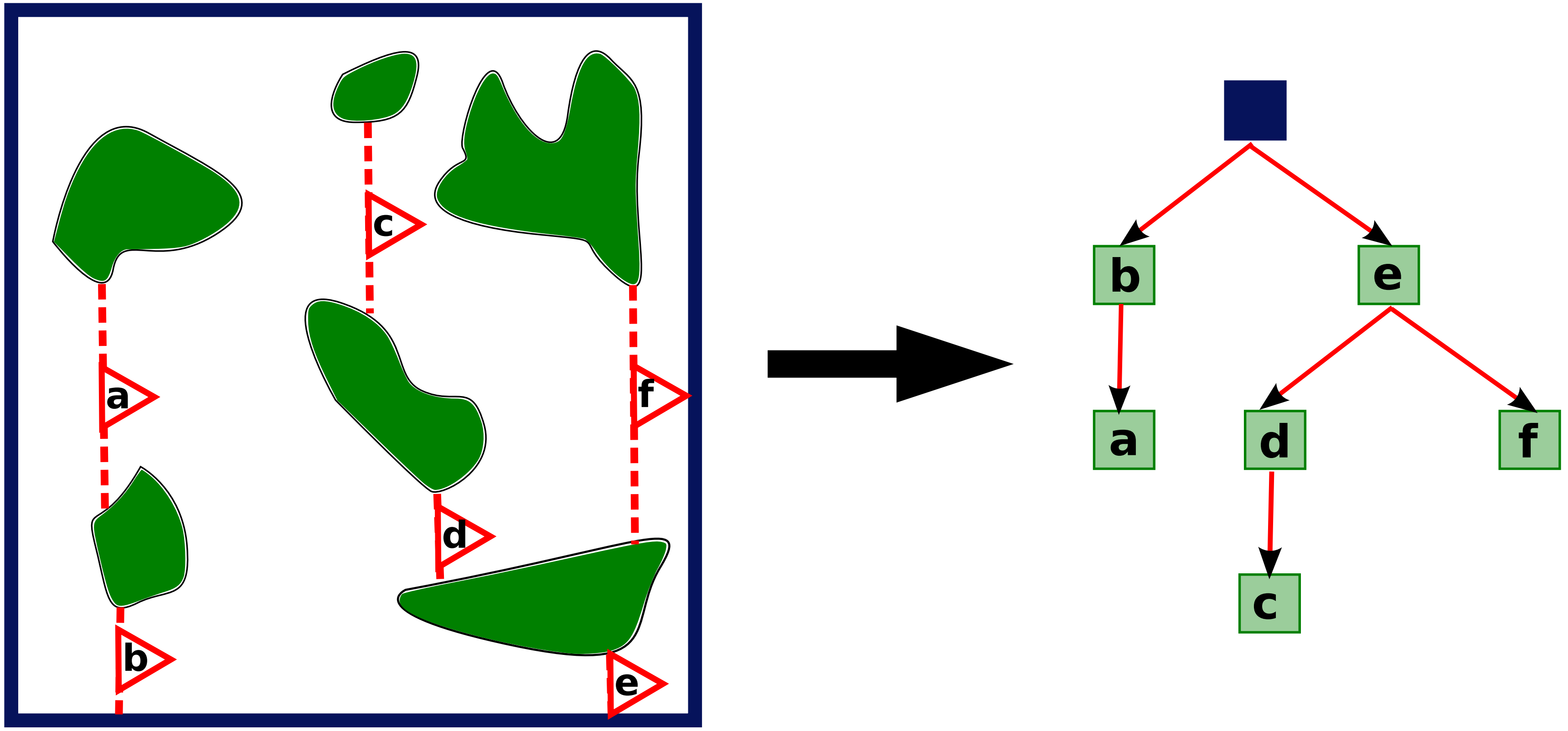
Commutativity

$\sigma_i \sigma_j = \sigma_j \sigma_i$

Relating Beam Crossings and Topology

How do beam crossings relate to homotopy and homology?

Recollect construction of virtual beams



Relating Beam Crossings and Topology

Spanning tree ensures "minimally sufficient collection of sensor beams"

Mapping from beam crossings to elements of F_n

Beam Crossings → **Letters of F_n**

a, a', b', b, ..., x, x' → σ_1, σ_1^{-1}, σ_2, σ_2^{-1}, ..., σ_n, σ_n^{-1}

Relating Beam Crossings and Topology

Applying equivalence relations on the string of beam crossings yields corresponding topological constraints

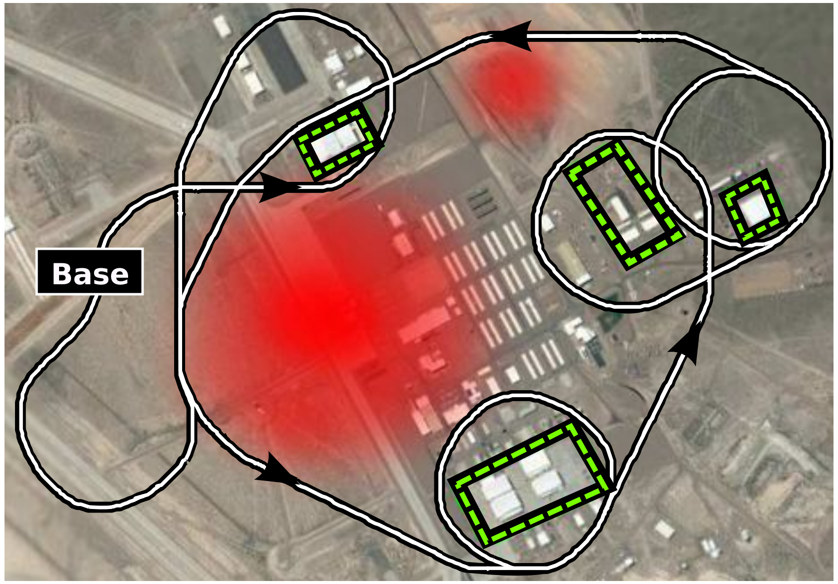
Inverse	Commutativity
aa’ ~ a’a ~ ε	ab ~ ba

Example reduction: b(s) = abcc’b’ca’

Inverse (Homotopy): abcc’b’ca’ ~ abb’ca’ ~ **aca’**

Inverse and Commutativity (Homology): abcc’b’ca’ ~ abb’ca’ ~ aca’ ~ aa’c ~ **c**

Relating Beam Crossings and Topology



"Find a circuit path for an UAV such that it obtains 360° views of *n* regions of interest, in a particular order"

↓

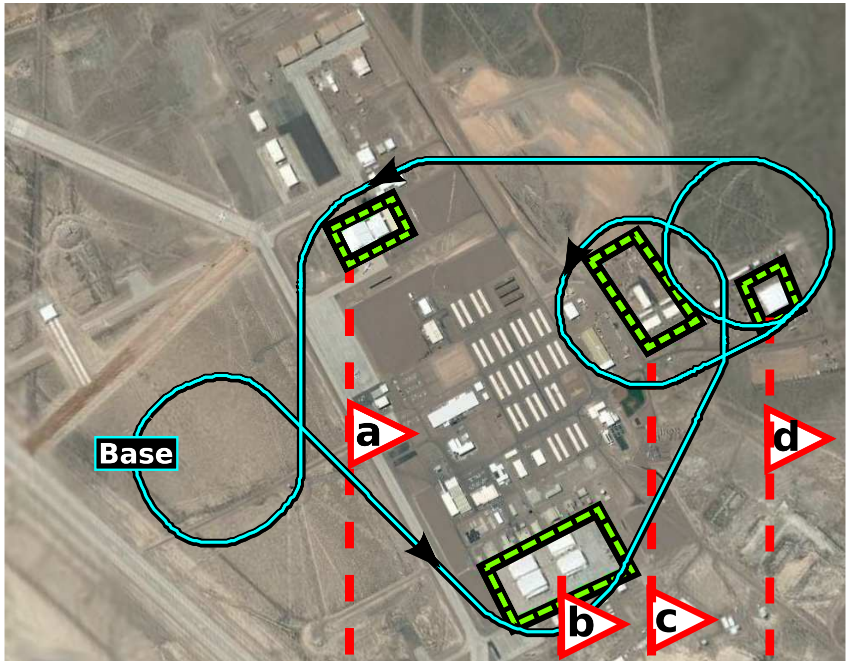
"Find a loop in the n-punctured plane, that belongs to a particular element of the fundamental group F_n"

↓

"Find a loop that has a particular sequence of beam crossings"

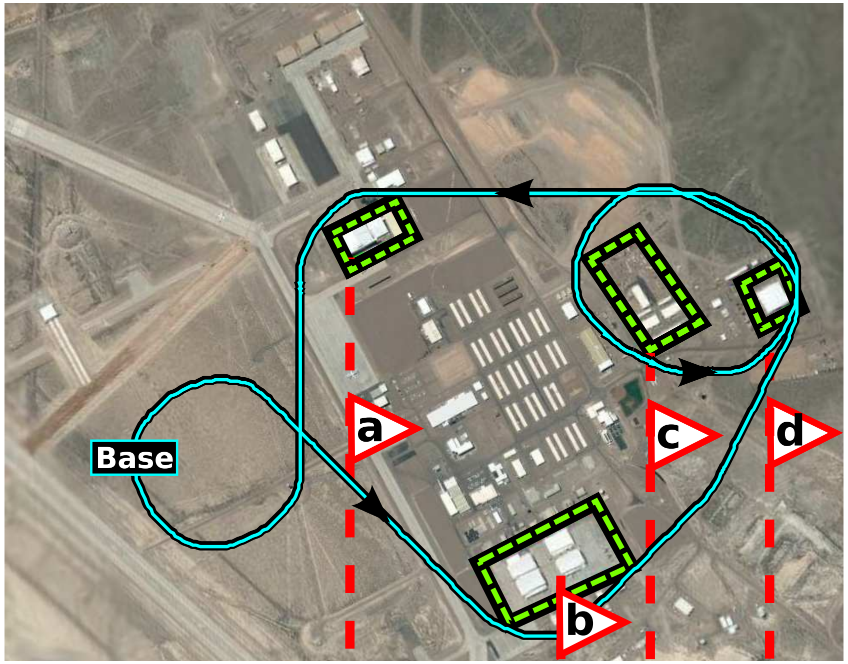
Homotopy vs Homology

Goal string: **abccdd**



Homotopy

Order matters



Homology

Order does not matter

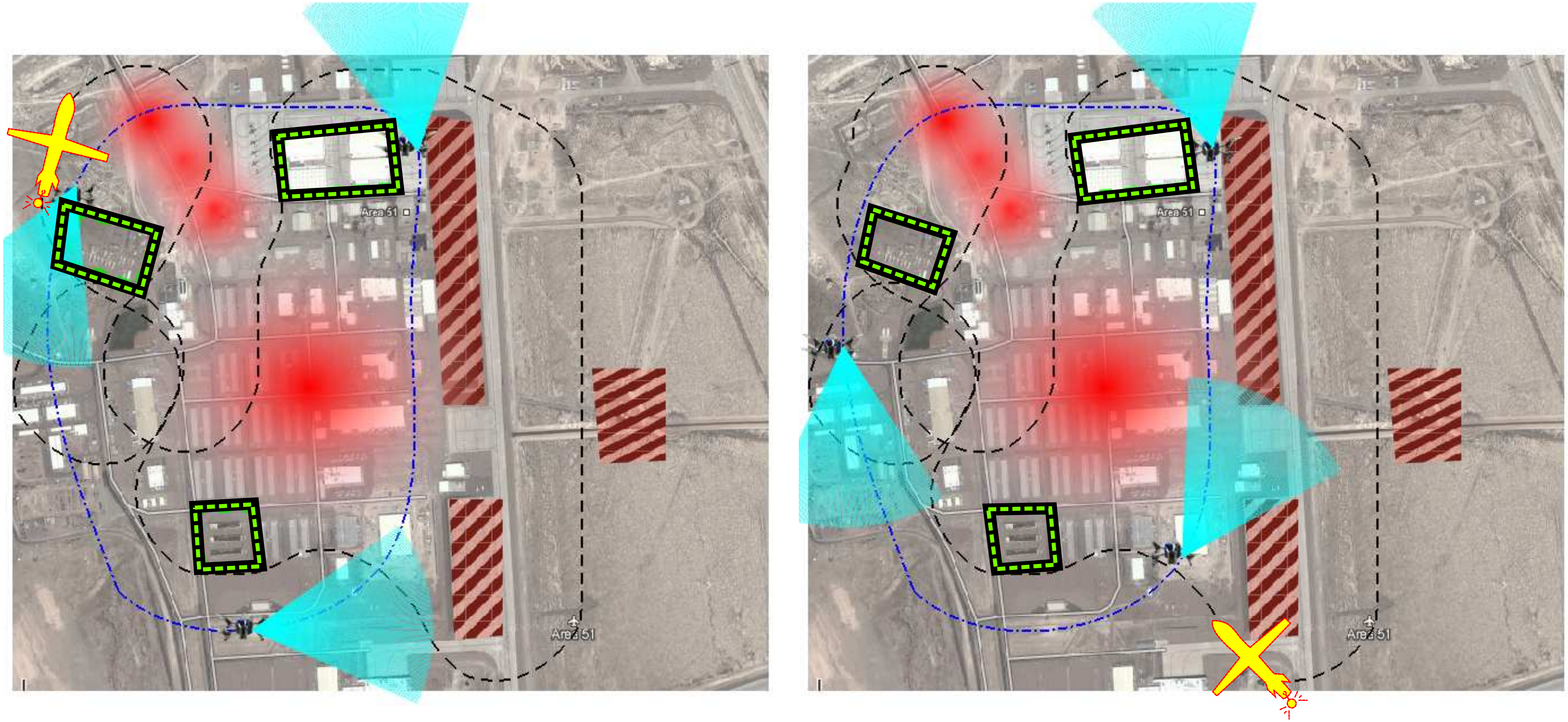
Experiments

UAV surveillance in dynamic environments

- Gather information about ROI in the environment
- Avoid hostile patrollers
- 5D (x, y, θ, t, b) planning problem
- A* search
- 2D (x,y) Dijkstra search to precompute admissible heuristics
- Cost function: Length of the path + Risk associated with getting close to a radar installation

Experiments

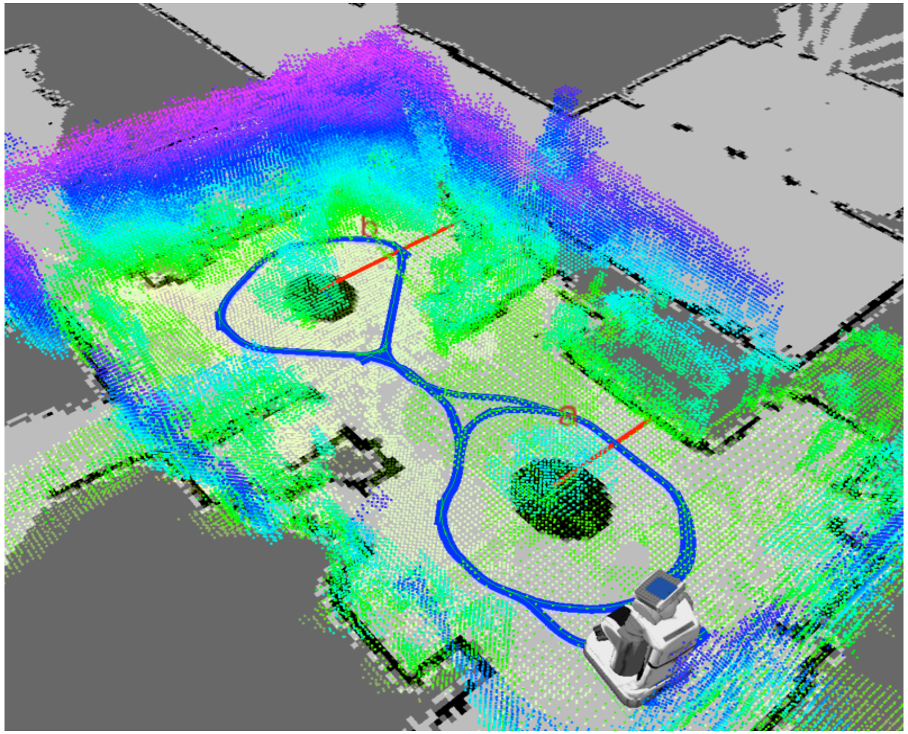
UAV surveillance in dynamic environments



Experiments

Loop planning for object modeling

- Obtain 3D models of objects in partially structured environments
- Mobile robot equipped with RGB-D sensor circumnavigates areas of interest
- 4D (x, y, θ, b) planning problem



Experiments

Scaling Experiments