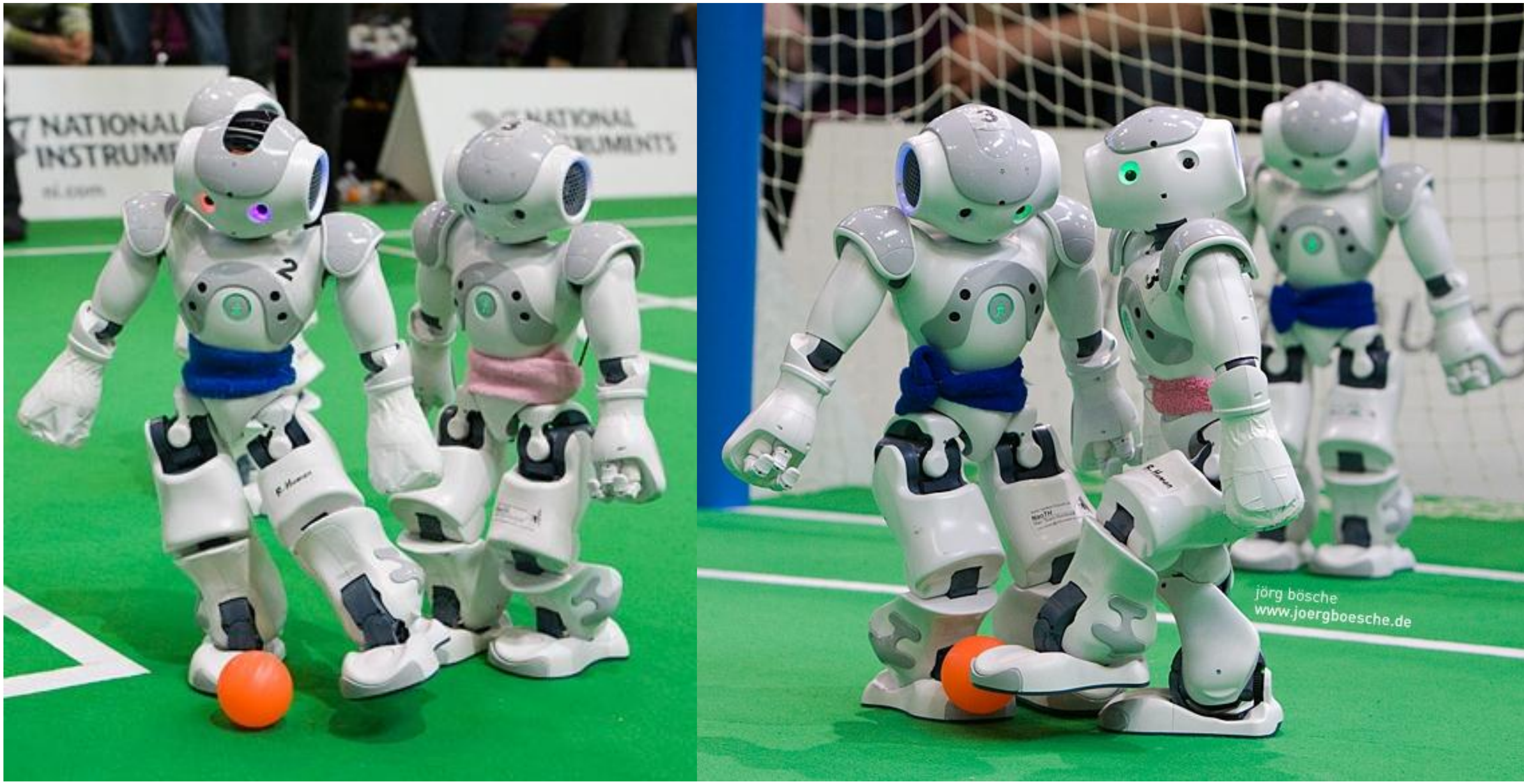
Search-Based Footstep Planning

Armin Hornung and Maren Bennewitz

University of Freiburg, Germany

Joint work with J. Garimort, A. Dornbush, M. Likhachev

Motivation

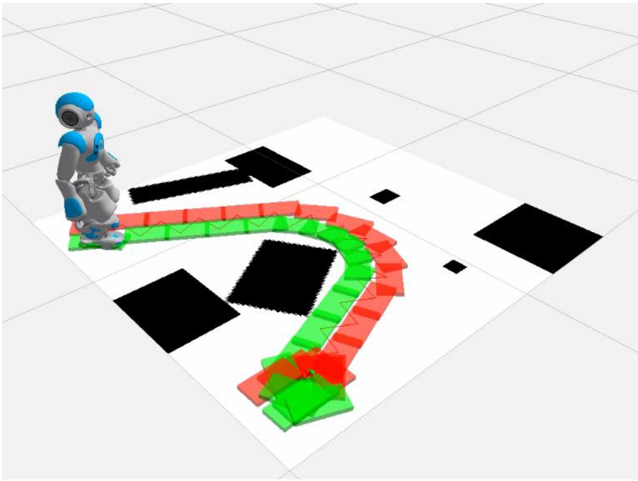


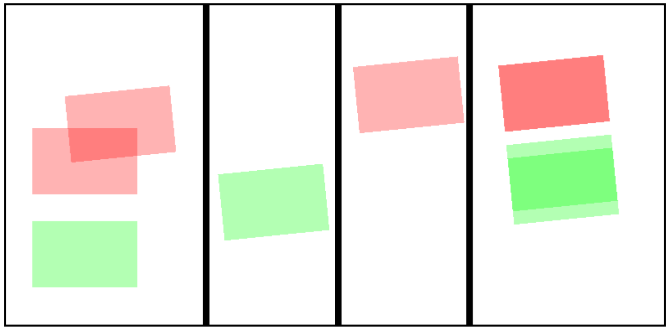
BHuman vs. Nimbro, RoboCup German Open 2010

Photo by J. Bösche, www.joergboesche.de

Path Planning for Humanoids

- Humanoids can avoid obstacles by stepping over or close to them
- However, planning whole-body motions has a high computational complexity
  [Hauser et al. ‘07, Kanoun ’10, …]
- Footstep planning given possible foot locations reduces the planning problem



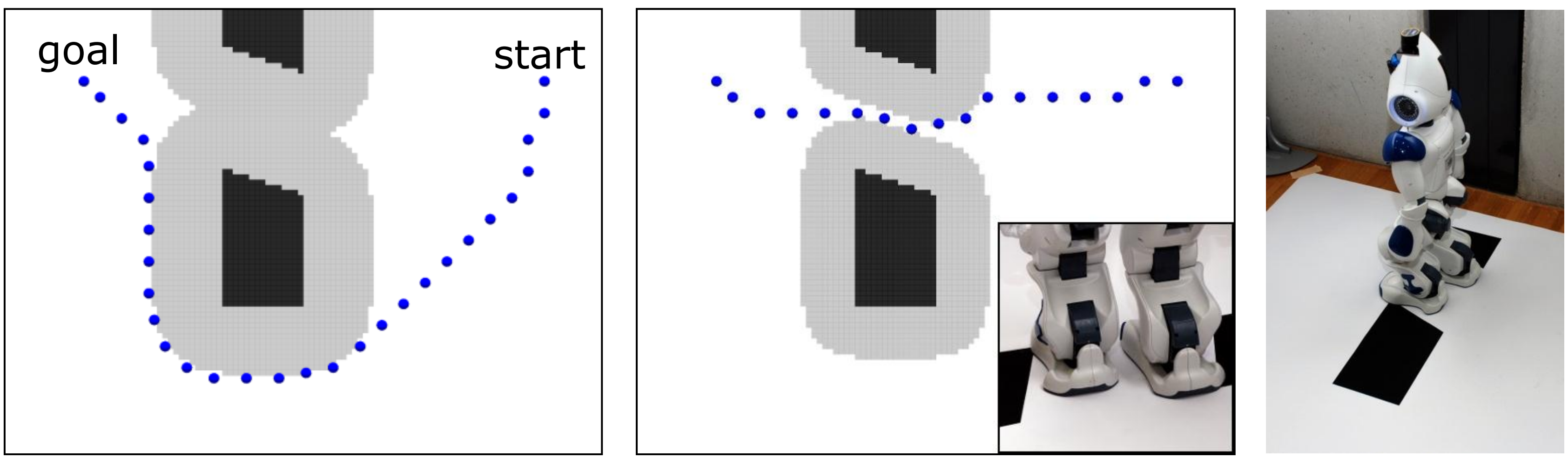


Previous Approaches

- Compute collision-free 2D path first, then footsteps in a local area

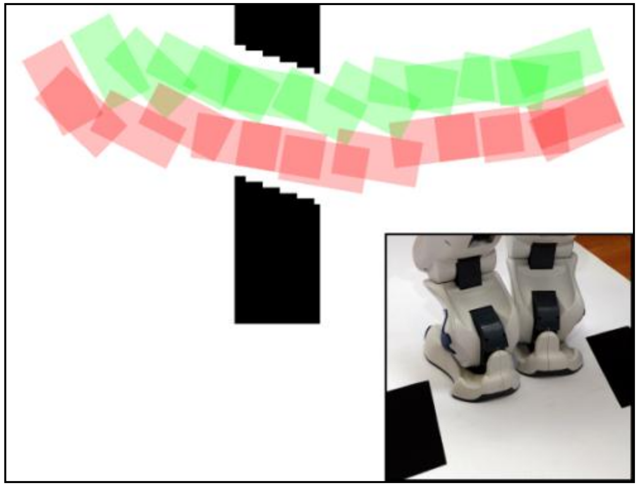
  [Li et al. ‘03, Chestnutt & Kuffner ‘04]

- Problem: 2D planner cannot consider all capabilities of the robot

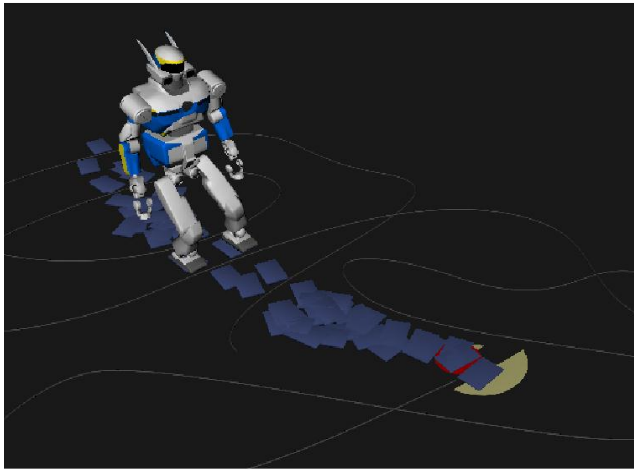


Previous Approaches

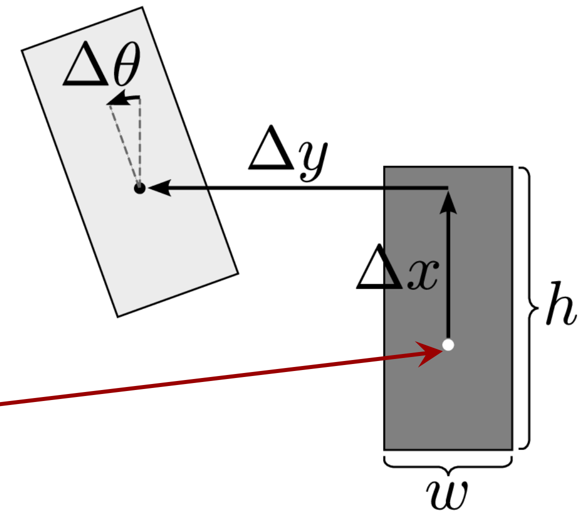
- Footstep planning with A*
  [Kuffner ‘01, Chestnutt et al. ‘05, ‘07]
  - Search space: (x,y,θ)
  - Discrete footstep set
  - Optimal solution with A*



- Probabilistic Footstep Planning
  [e.g. Perrin et al. ‘11]
  - Search space of footstep actions with RRT / PRM
  - Fast planning results
  - No guarantees on optimality or completeness



Footstep Planning



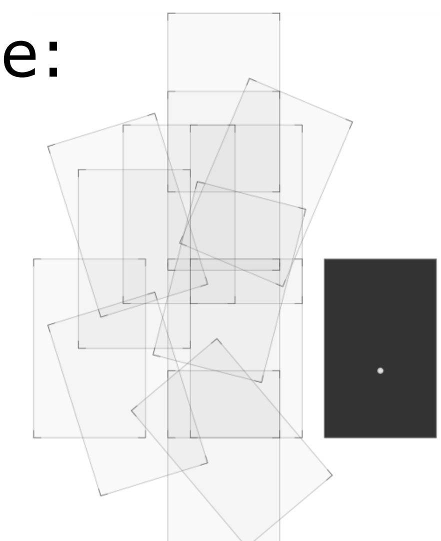
- State $s = (x, y, \theta)$
- Footstep action $a = (\Delta x, \Delta y, \Delta\theta)$
- Fixed set of footstep actions $F = \{a_1, \ldots, a_n\}$
- Successor state $s' = t(s, a)$
- Transition costs reflect execution time:

$$c(s, s') = \|(\Delta x, \Delta y)\| + k + d(s')$$

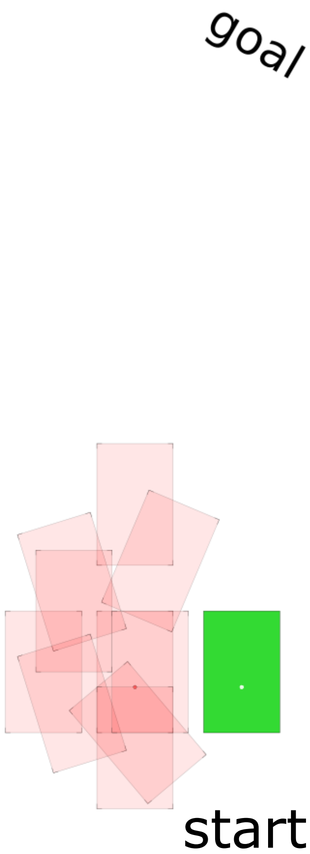
Euclidean distance

constant step cost

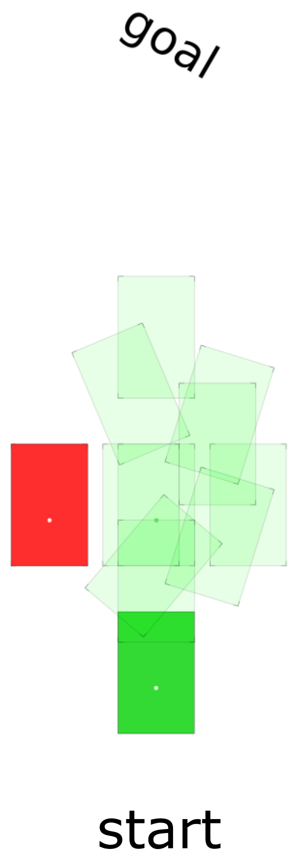
costs based on the distance to obstacles



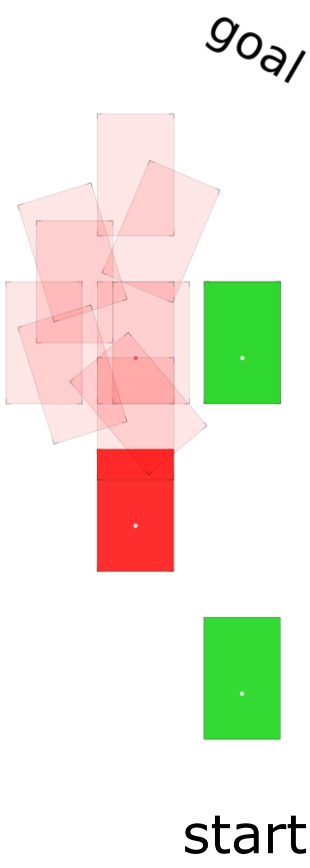
Footstep Planning



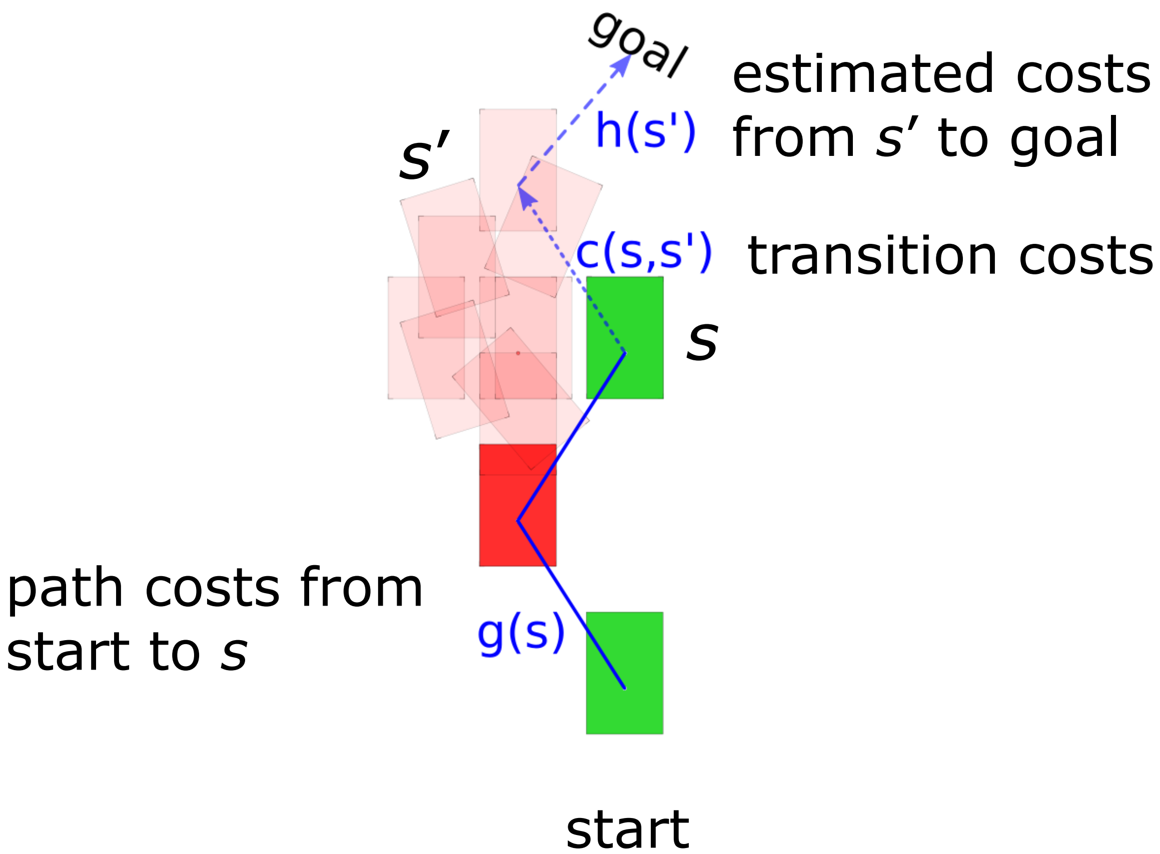
Footstep Planning



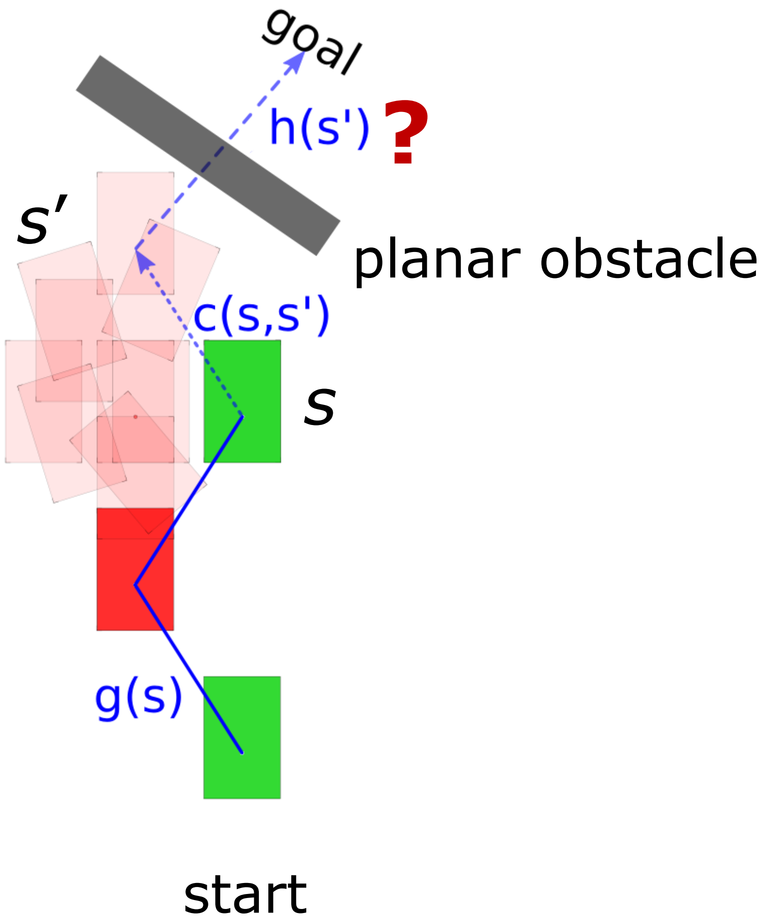
Footstep Planning



Footstep Planning

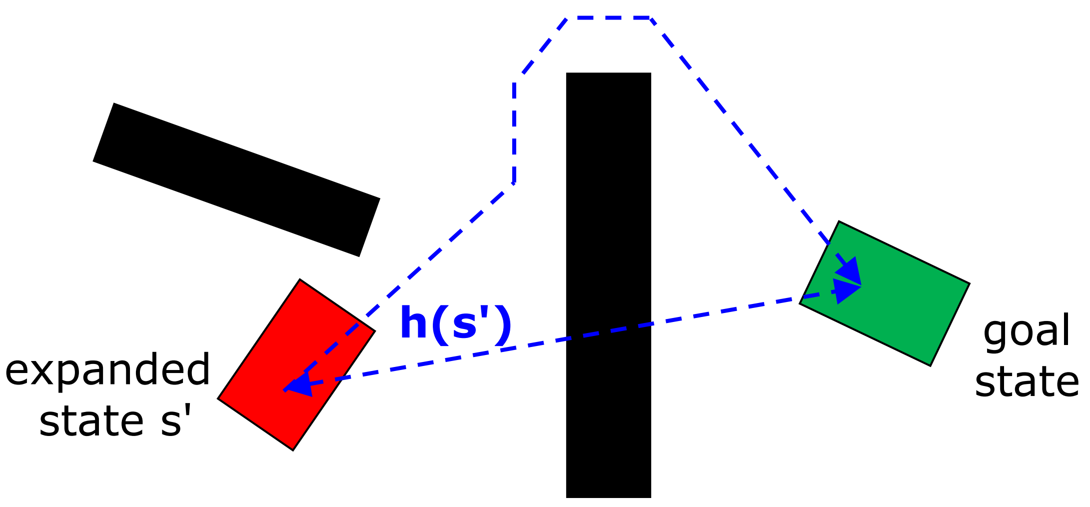


Footstep Planning



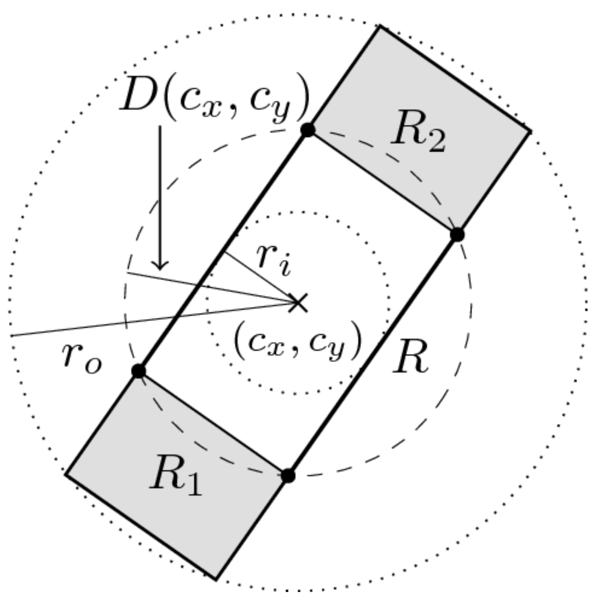
Heuristic

- Estimates the costs to the goal
- Critical for planner performance
- Usual choices:
  - Euclidean distance
  - 2D Dijkstra path



Efficient Collision Checking

- Footprint is rectangular with arbitrary orientation
- Evaluating the distance between foot center and the closest obstacle may not yield correct or optimal results
- Recursively subdivide footstep shape

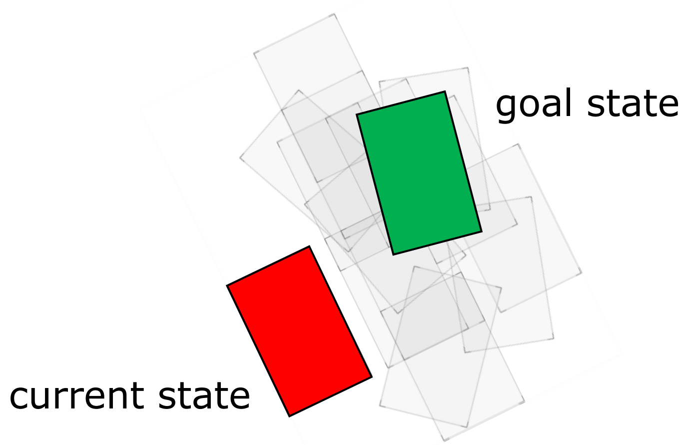


$D(c_x, c_y)$ = distance to the closest obstacle (precomputed map)

[Sprunk et al. (ICRA '11)]

Search-Based Footstep Planning

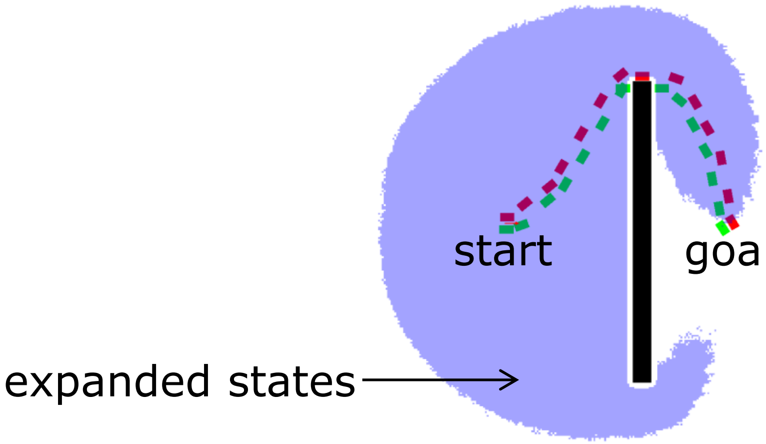
- Concatenation of footstep actions builds a lattice in the global search space
- Only valid states after a collision check are added
- Goal state may not be exactly reached, but it is sufficient to reach a state close by (within the motion range)



Search-Based Footstep Planning

- We can now apply heuristic search methods on the state lattice
- Search-based planning library: www.ros.org/wiki/sbpl
- Footstep planning implementation based on SBPL: www.ros.org/wiki/footstep_planner

Local Minima in the Search Space



- A* will search for the optimal result
- Initially sub-optimal results are often sufficient for navigation
- Provable sub-optimality instead of randomness yields more efficient paths

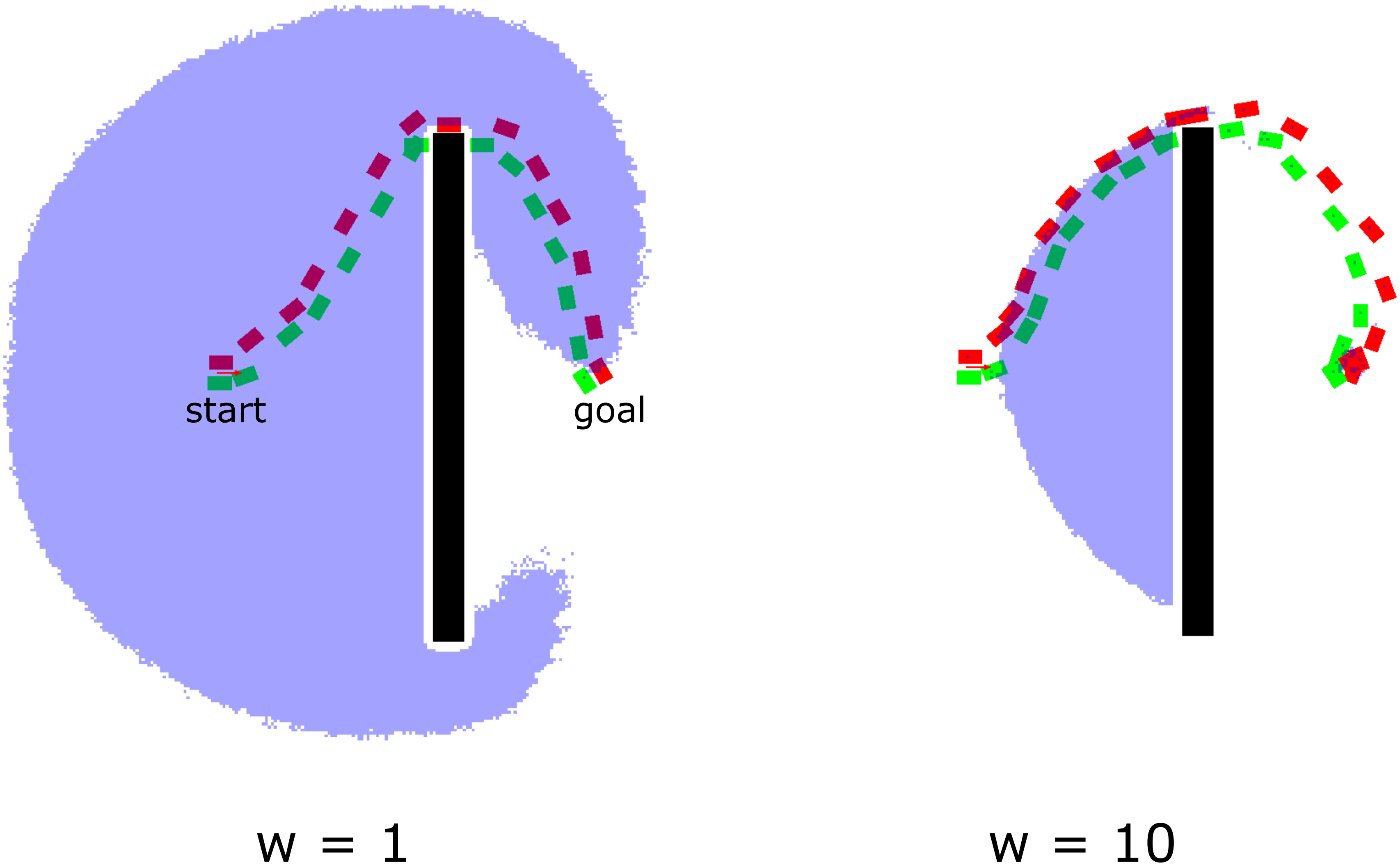
Anytime Repairing A* (ARA*)

- Heuristic inflation by a factor w allows to efficiently deal with local minima: weighted A* (wA*)
- ARA* runs a series of wA* searches, iteratively lowering w as time allows
- Re-uses information from previous iterations

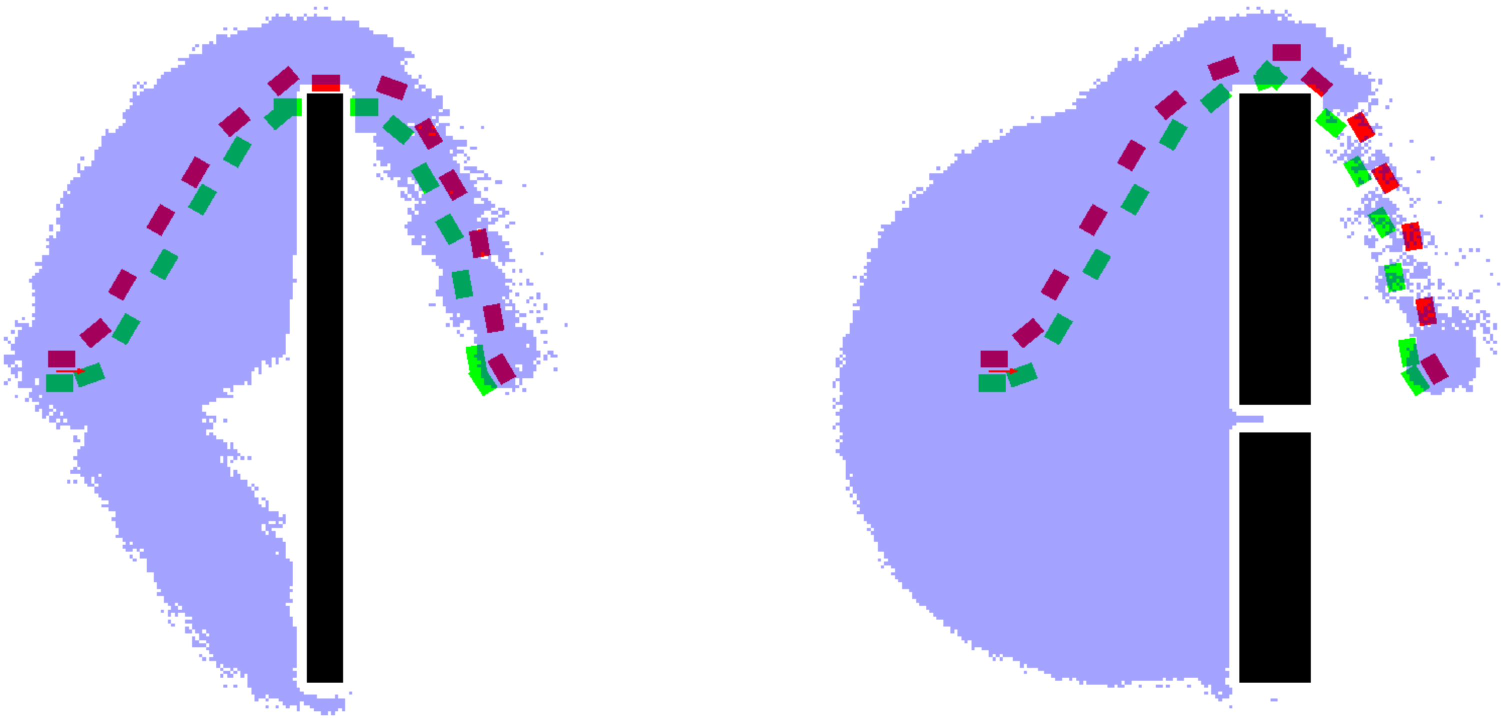
[Likhachev et al. (NIPS 2004), Hornung et al. (Humanoids 2012)]

Interactive Session III (Sa., 15:00)

ARA* with Euclidean Heuristic



ARA* with Dijkstra Heuristic

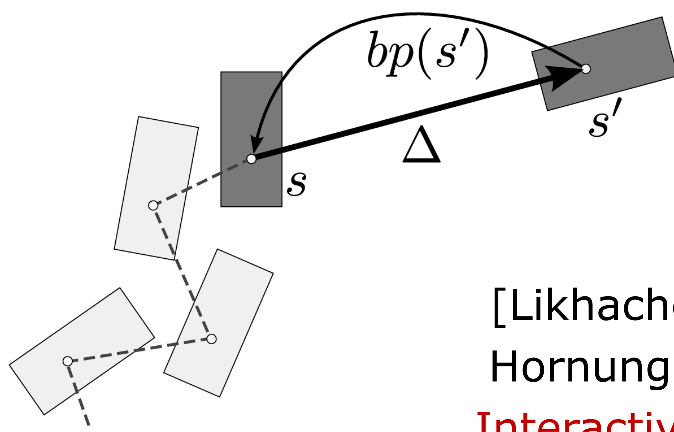


w = 1

Performance depends on well-designed heuristic

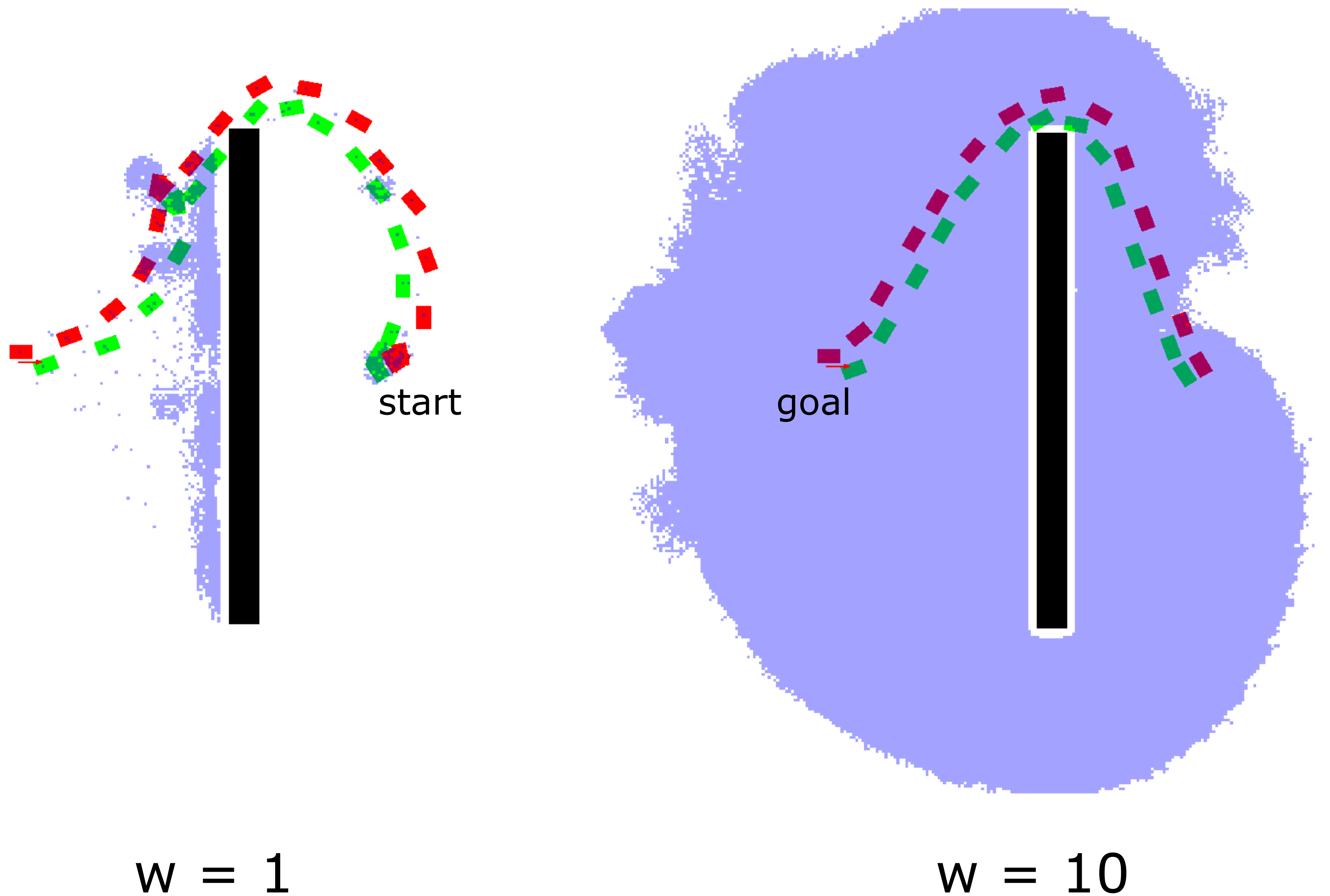
Randomized A* (R*)

- Iteratively constructs a graph of sparsely placed randomized sub-goals (exploration)
- Plans between sub-goals with wA*, preferring easy-to-plan sequences
- Iteratively lowers w as time allows



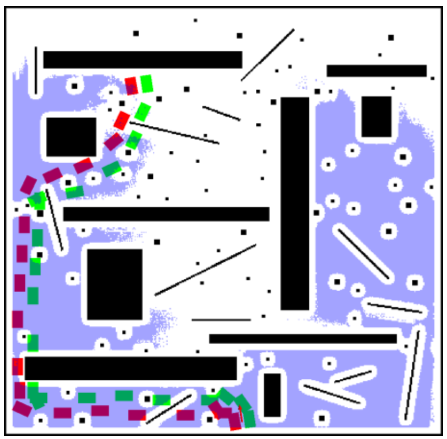
[Likhachev & Stentz (AAAI 2008), Hornung et al. (Humanoids 2012)]
Interactive Session III (Sa., 15:00)

R* with Euclidean Heuristic



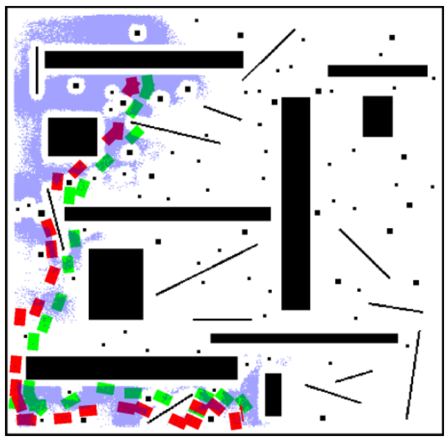
Planning in Dense Clutter Until First Solution

A*
Euclidean heur.



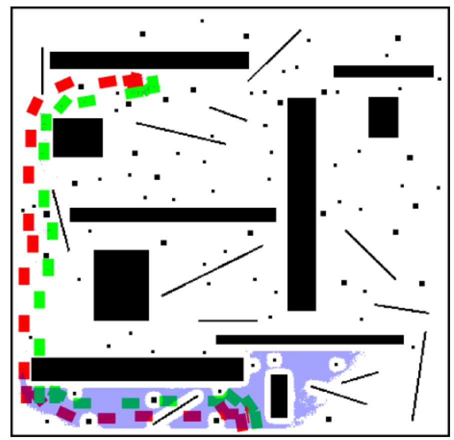
11.9 sec.

R*
Euclidean heur.



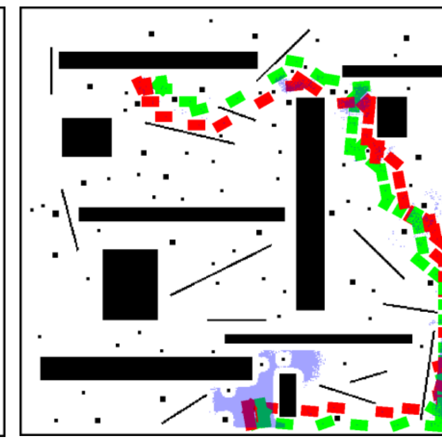
0.4 sec.

ARA*
Euclidean heur.



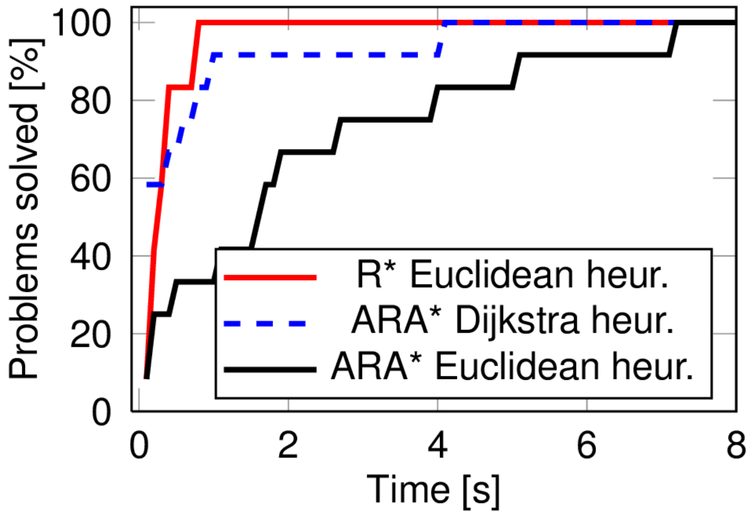
2.7 sec.

ARA*
Dijkstra heur.



0.7 sec.

Planning in Dense Clutter Until First Solution

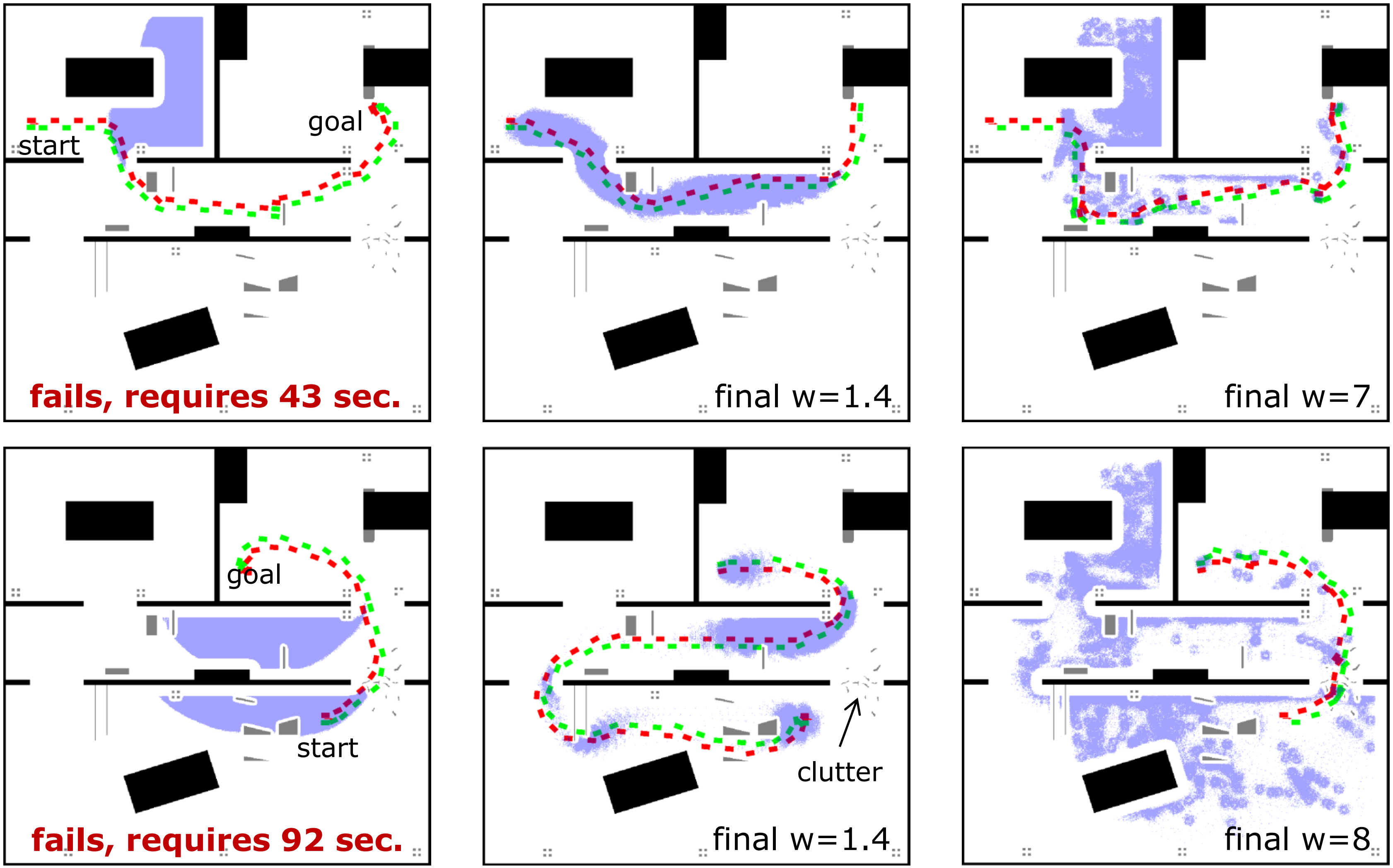


Planner	Heuristic	Planning time [s]	Path costs
R* (w=5)	Euclidean	0.32 ± 0.23	16.45 ± 3.16
ARA* (w=5)	Euclidean	2.15 ± 2.21	13.57 ± 1.15
ARA* (w=5)	2D Dijkstra	0.56 ± 1.13	20.41 ± 5.08
A* (w=1)	Euclidean	33.31 ± 15.00	11.06 ± 1.20

- 12 random start and goal locations
- ARA* finds fast results only with the 2D Dijkstra heuristic, leading to longer paths due to its inadmissibility
- **R* finds fast results even with the Euclidean heuristic**

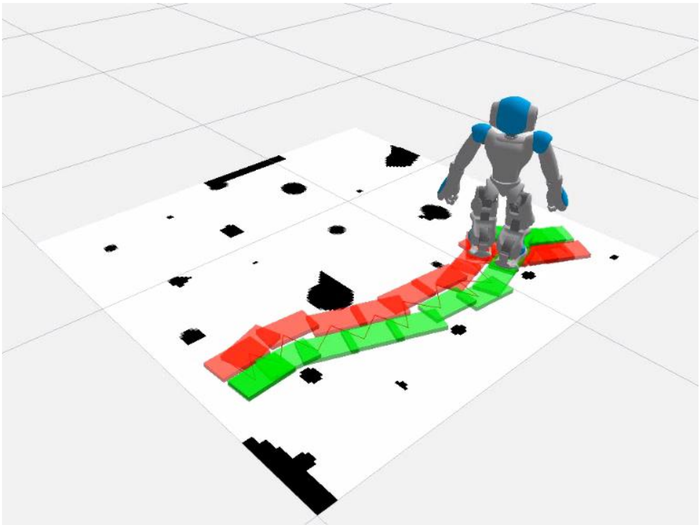
Planning with a Time Limit (5s)

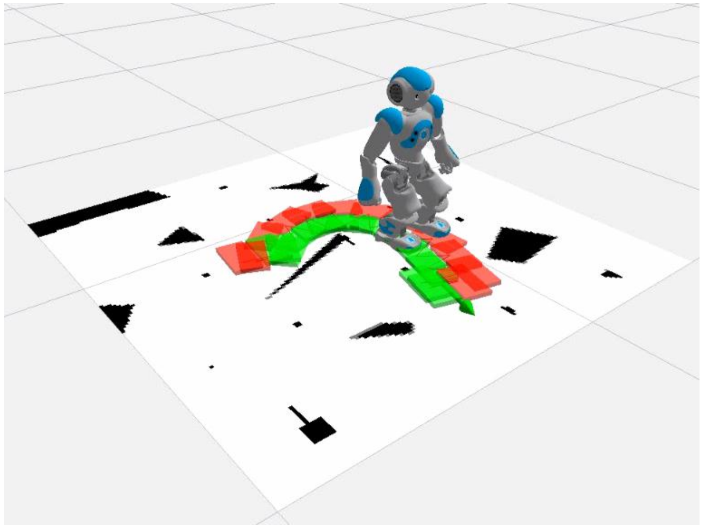
ARA* Euclidean heuristic | **ARA*** Dijkstra heuristic | **R*** Euclidean heuristic



Anytime Planning Results

- Performance of ARA* depends on well-designed heuristic
- Dijkstra heuristic may be inadmissible and can lead to wrong results
- R* with the Euclidean heuristic finds efficient plans in short time



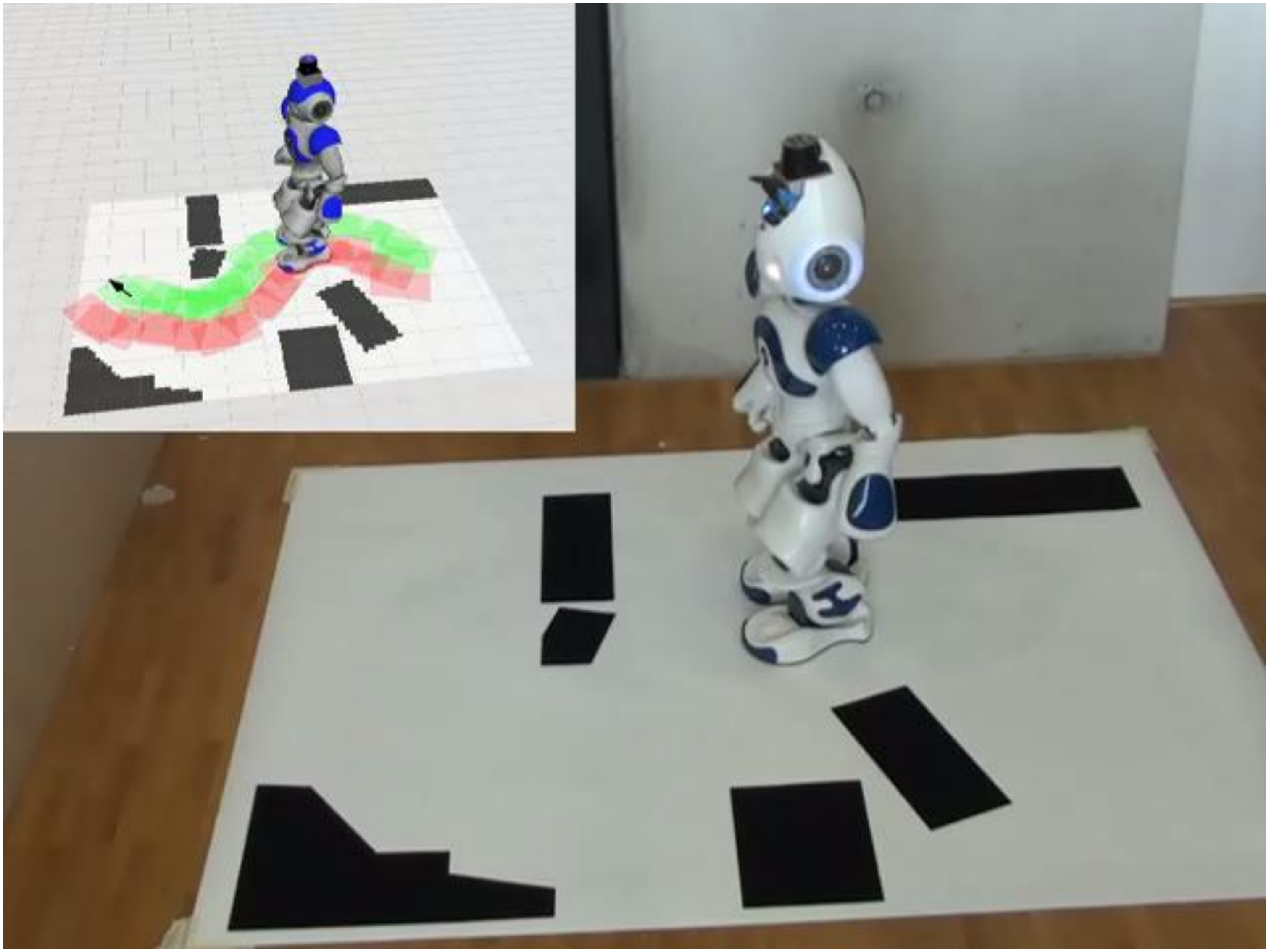


Dynamic A* (D*)

- Allows for efficient re-planning in case of
  - Changes in the environment
  - Deviations from the initial path
- Re-uses state information from previous searches
- Planning backwards increases the efficiency in case of updated localization estimates
- Anytime version: AD*

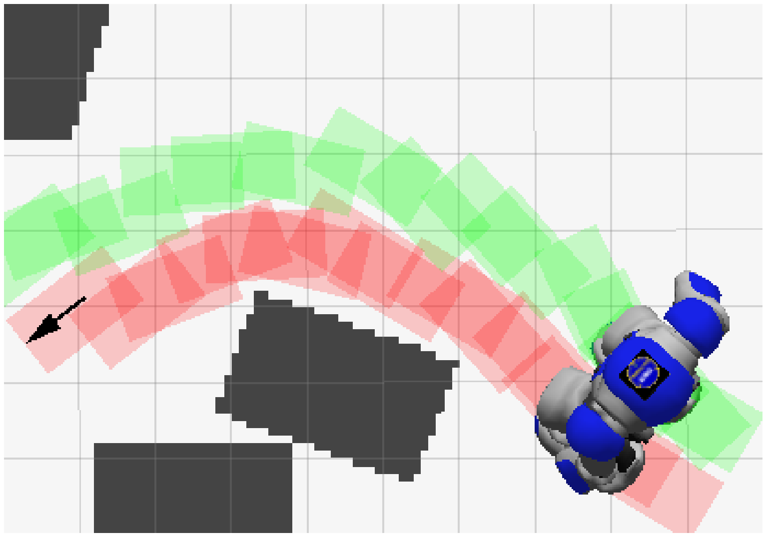
[Koenig & Likhachev (AAAI ‘00), Garimort (ICRA ’11)]

D* Plan Execution with a Nao

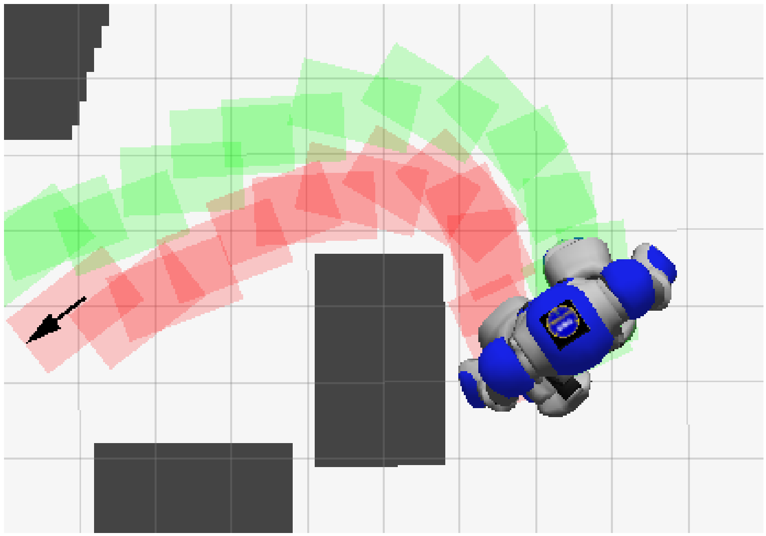


Efficient Replanning

- Plans may become invalid due to changes in the environment
- D* allows for efficient plan re-usage

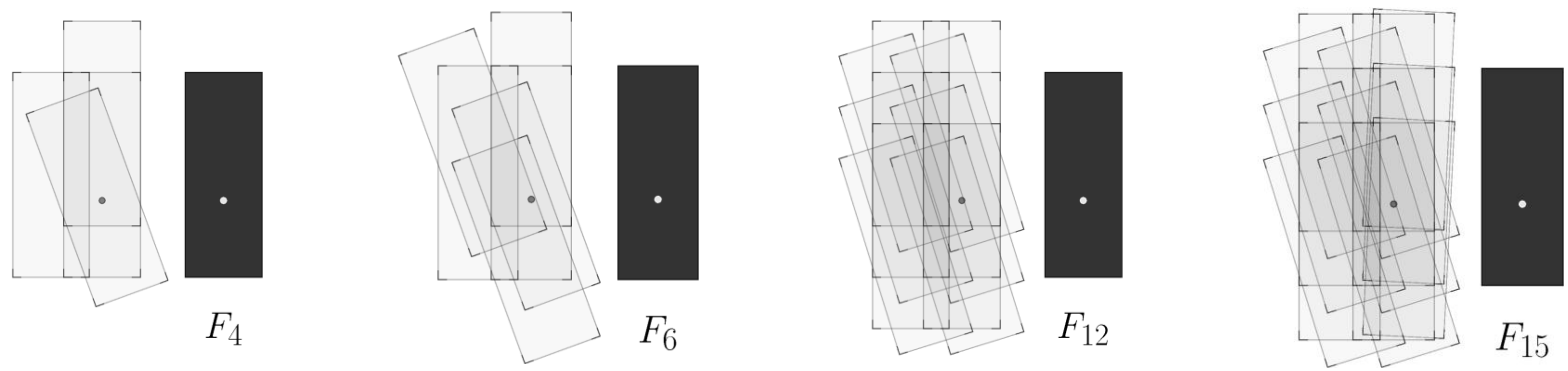


2966 states, 1.05s

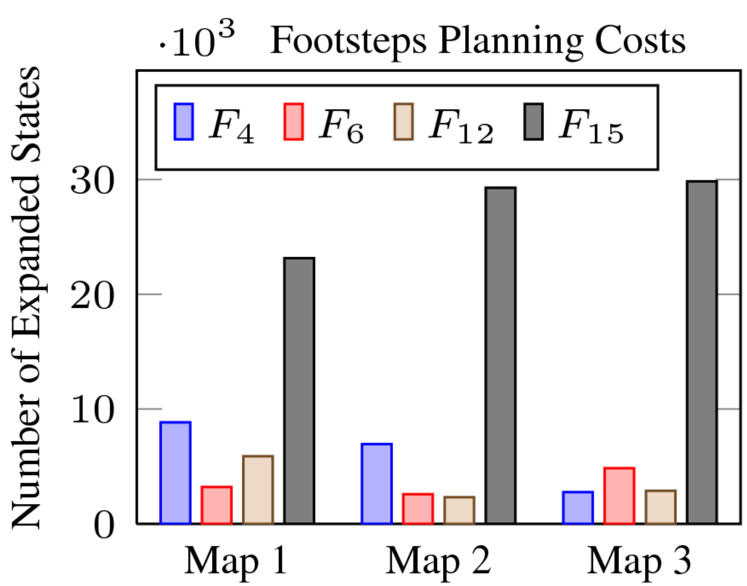


956 states, 0.53s

Different Footstep Sets for Nao

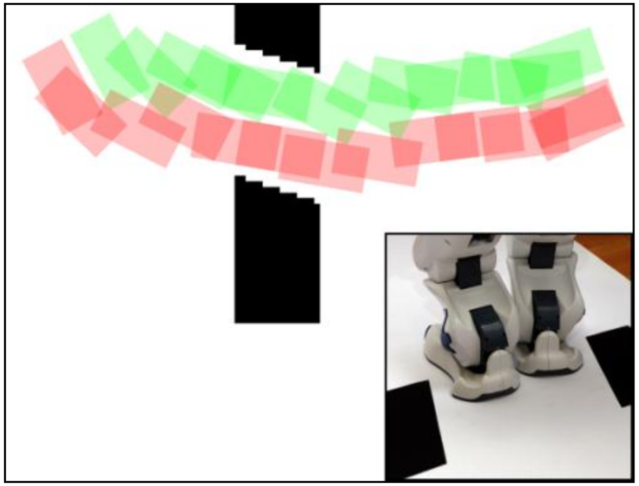


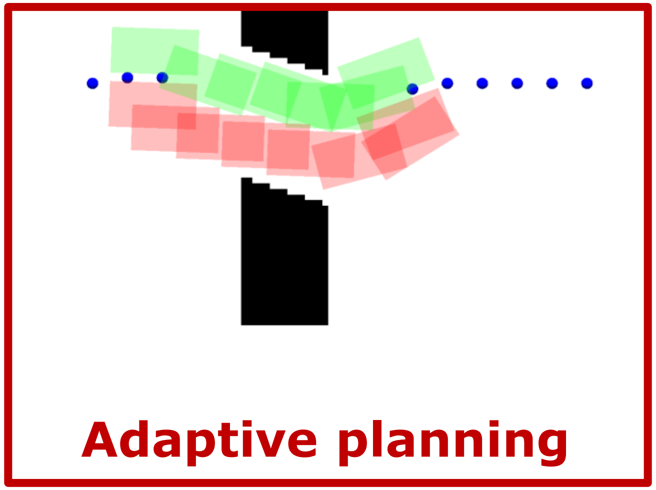
- F_{12} and F_{15} lead to significantly shorter paths
- F_{15} has a significantly higher planning time
- Result: F_{12} yields shortest paths with efficient planning times



Adaptive Level-of-Detail Planning

- Planning the whole path with footsteps may not always be desired in large open spaces
- Adaptive level-of-detail planning: Combine fast grid-based 2D planning in open spaces with footstep planning near obstacles

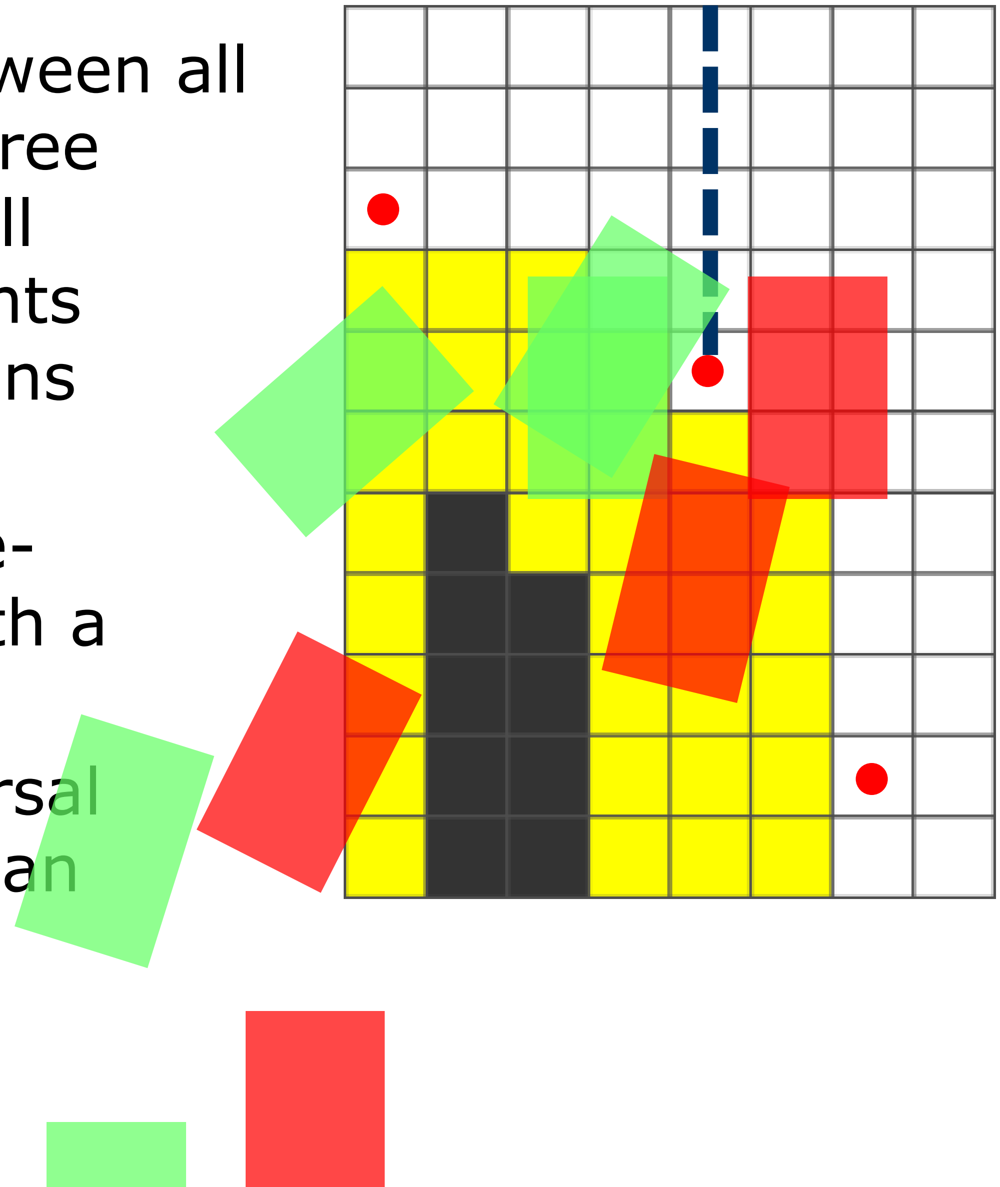




[Hornung & Bennewitz (ICRA ‘11)]

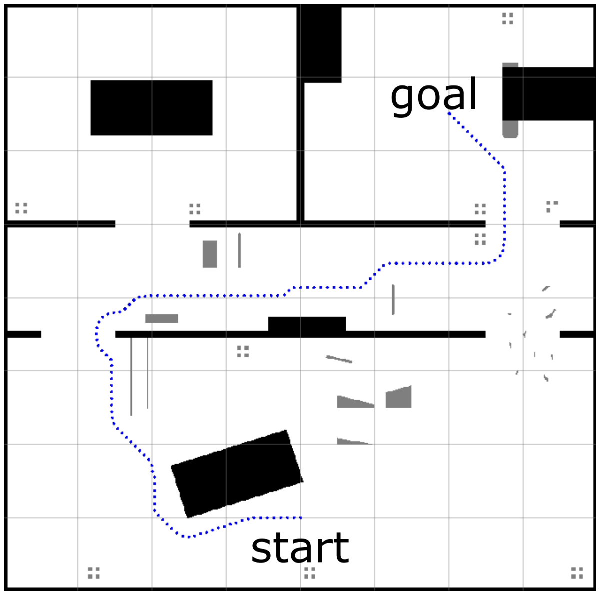
Adaptive Level-of-Detail Planning

- Allow transitions between all neighboring cells in free areas and between all sampled contour points across obstacle regions
- Traversal costs are estimated from a pre-planning stage or with a learned heuristic
- Every obstacle traversal triggers a footstep plan



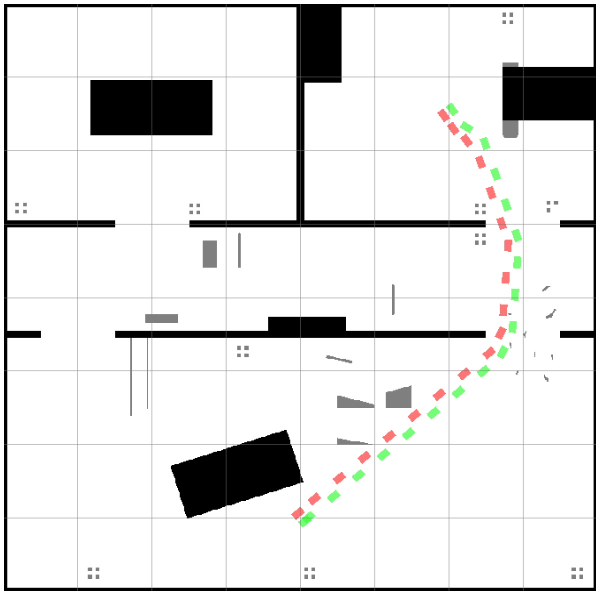
Adaptive Planning Results

2D Planning



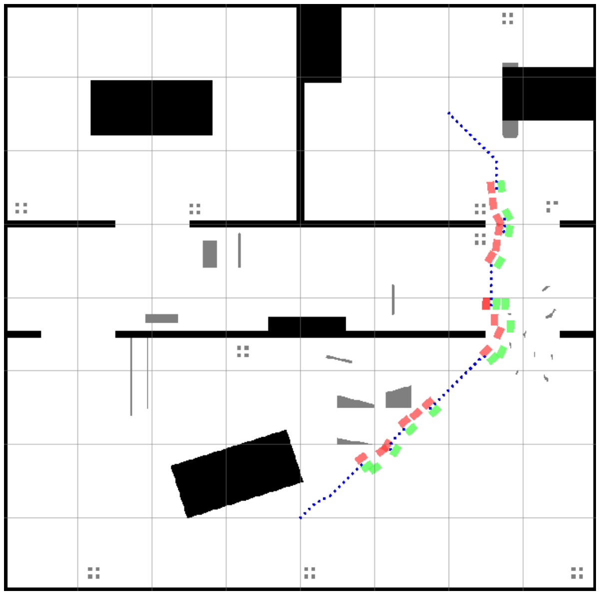
<1 s planning time
High path costs

Footstep Planning



29 s planning time

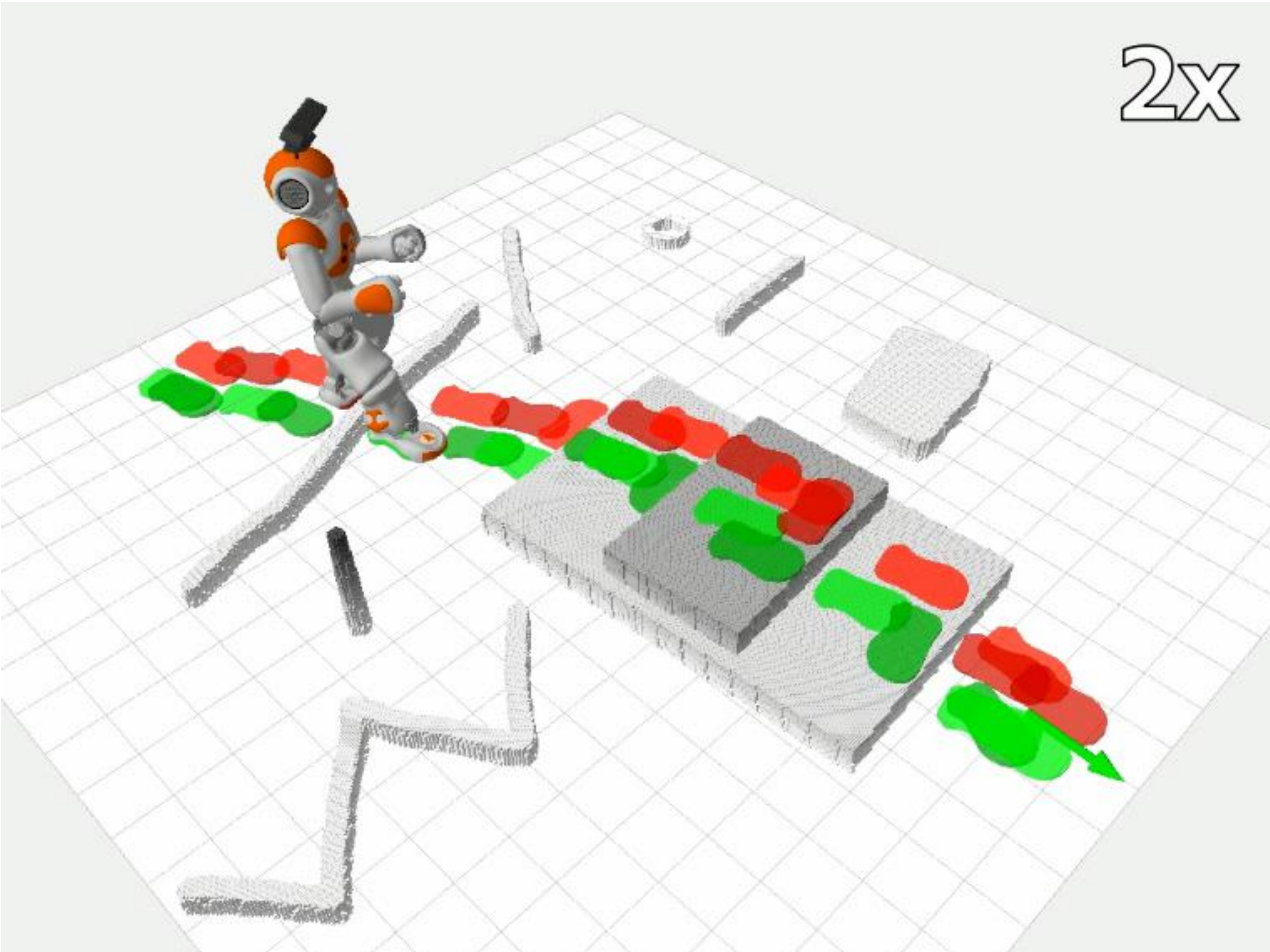
Adaptive Planning



<1s planning time, costs only 2% higher

Fast planning times and efficient solutions with adaptive level-of-detail planning

Current Work: Planning in 3D



Summary

- Anytime search-based footstep planning with suboptimality bounds: ARA* and R*
- Replanning during navigation with AD*
- Heuristic influences planner behavior
- Adaptive level-of-detail planning to combine 2D with footstep planning
- Available open source in ROS: www.ros.org/wiki/footstep_planner

➡ Interactive Session III (Saturday, 15:00)

Thank you!