Computation and Use of Topology-based Heuristic Functions for Motion Planning

Maxim Likhachev
Robotics Institute & NREC
Search-based Planning Lab (SBPL)
Carnegie Mellon University

Joint work with V. Ranganeni, O. Salzman, S. Chintalapudi, A. Dornbush, K. Vijayakumar, S. Bardapurkar, F. Islam
Motivation for Topology-based Heuristics

• Single-planner approach to humanoid planning [Dornbush et al., ICRA’18]
Motivation for Topology-based Heuristics

- Single-planner approach to humanoid planning [Dornbush et al., ICRA’18]

What path to take? What gait to use and when? Which limbs to exercise? What support surfaces to use? What limb motion to utilize?

Decomposition into a whole bunch of planners/decision trees is brittle.
Solving it as a single search is intractable.
Motivation for Topology-based Heuristics

- Single-planner approach to humanoid planning [Dornbush et al., ICRA’18]

What path to take? What gait to use and when? Which limbs to exercise? What support surfaces to use? What limb motion to utilize?

Heuristics allow us to “softly” decompose the problem without loosing guarantees on global completeness/bounded sub-optimality

Solving it as a single search is intractable.
What are Heuristic Functions?

- Heuristic values in A*-like planning = estimates of the cost-to-goal
- A*-like planning (e.g., weighted A*, etc.) biases its search efforts along the gradient given by the heuristic function while maintaining guarantees on completeness and bounded sub-optimality

$h(S)$: Euclidean distance

Robot

goal

$S$
What are Heuristic Functions?

- Heuristic values in A*-like planning = estimates of the cost-to-goal
- A*-like planning (e.g., weighted A*, etc.) biases its search efforts along the gradient given by the heuristic function while maintaining guarantees on completeness and bounded sub-optimality

- Heuristics based on 2D solution costs often provide “good” gradients
What are Heuristic Functions?

- Heuristic values in A*-like planning = estimates of the cost-to-goal.
- A*-like planning (e.g., weighted A*, etc.) biases its search efforts along the gradient given by the heuristic function while maintaining guarantees on completeness and bounded sub-optimality.

Heuristics based on 2D solution costs often provide “good” gradients.

But what if this class is infeasible?

such heuristic function guides planning along a topological class corresponding to an optimal 2D solution

Robot

S

distance

goal

S

h(S):

2D pathcost

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

goal

Robot

S

going
What are Heuristic Functions?

- Heuristic values in A*-like planning = estimates of the cost-to-goal
- A*-like planning (e.g., weighted A*, etc.) biases its search efforts along the gradient given by the heuristic function while maintaining guarantees on completeness and bounded sub-optimality
- Such heuristic function guides planning along a topological class corresponding to an optimal 2D solution

But what if this class is infeasible?

The planner needs to be capable of using multiple heuristic functions simultaneously!
Use of Multiple Topology-based Heuristics

- Computation of Topology-based Heuristics
  - Specification of topology classes to consider during planning (optional)
  - Multiple heuristic functions, one per topology class

- Multi-Heuristic A* (MHA*)
  [Aine et al., IJRR’15]

- Computed plan
Use of Multiple Topology-based Heuristics

Use Beams [Tovar et al., ‘09] to define topology signature in 2D

Computation of Topology-based Heuristics

- Multiple heuristic functions, one per topology class
- Specification of topology classes to consider during planning (optional)

Multi-Heuristic A* (MHA*) [Aine et al., IJRR’15]

Computed plan
Use of Multiple Topology-based Heuristics

Use Beams and Gates [Ranganeni et al., in submission] to define topology signature in 2.5D

Computation of Topology-based Heuristics

- Multiple heuristic functions, one per topology class
- Computed plan
Use of Multiple Topology-based Heuristics

Runs a single Dijkstra’s search backwards from the goal on graph $G$, where each vertex $v$ is defined by $(x,y,h(\gamma))$.

Computation of Topology-based Heuristics

- Specification of topology classes to consider during planning (optional)
- Multiple heuristic functions, one per topology class

Multi-Heuristic $A^*$ (MHA*)

[Aine et al., IJRR’15]

Computed plan
Use of Multiple Topology-based Heuristics

Runs MHA* search forward (from start) guided by computed heuristic functions on:

- original graph where each vertex $v$ is defined by $q_{full}$
  OR
- augmented graph where each vertex $v$ is defined by $(q_{full}, h(\gamma))$
Benefits of Topology-based Heuristics

- Footstep planning for humanoid [Ranganeni et al., ICAPS ‘18]

\[ \text{planning with one topology-based heuristic function (sec)} \]

\[ \text{planning without topology-based heuristic function (sec)} \]

- \textit{complex queries}
- \textit{simple queries}
Potential Future Directions

- Automatically figuring out what topology classes to consider for computing heuristics
- Dynamically instantiating new topology-based heuristics
- Understanding when planning hits a local minimum and a new topological class needs to be explored
Thanks

- Students/postdocs who contributed:
  - V. Ranganeni
  - O. Salzman
  - S. Chintalapudi
  - A. Dornbush
  - K. Vijayakumar
  - S. Bardapurkar
  - F. Islam

- Funding:
  - Mitsubishi
  - ONR
  - ARL
  - NSF