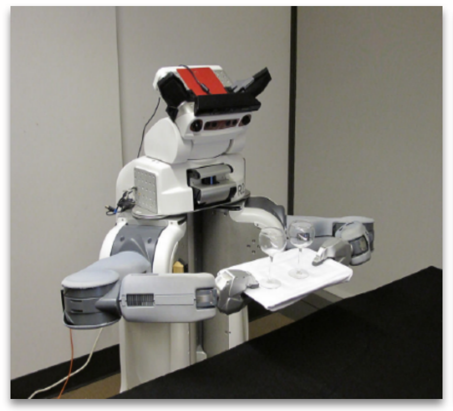
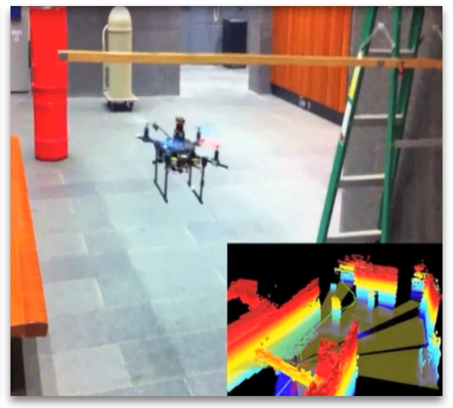
Multi-Heuristic A*

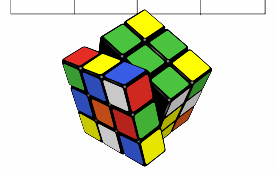
Sandip Aine, Siddharth Swaminathan, Venkatraman Narayanan, Victor Hwang, and Maxim Likhachev
IIIT Delhi   The Robotics Institute, Carnegie Mellon University

Case for Multiple Heuristics

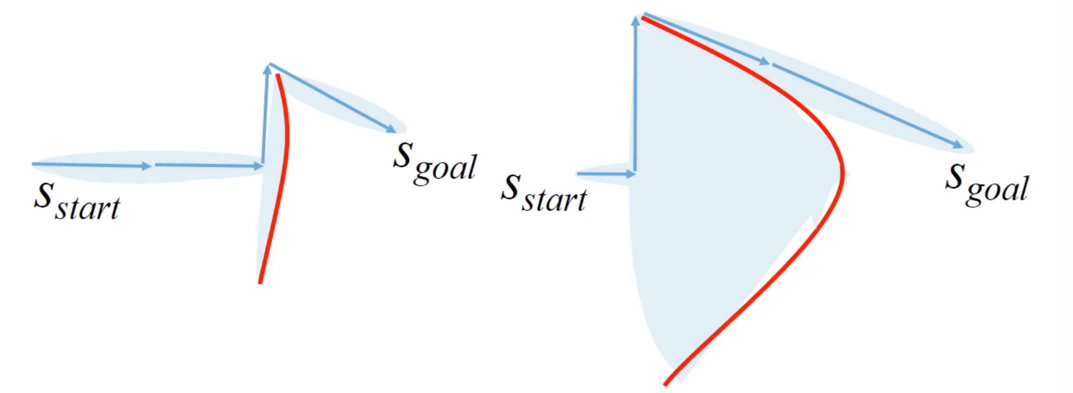




3	2	1	4
5		11	8
9	7	10	12
13	14	6	15



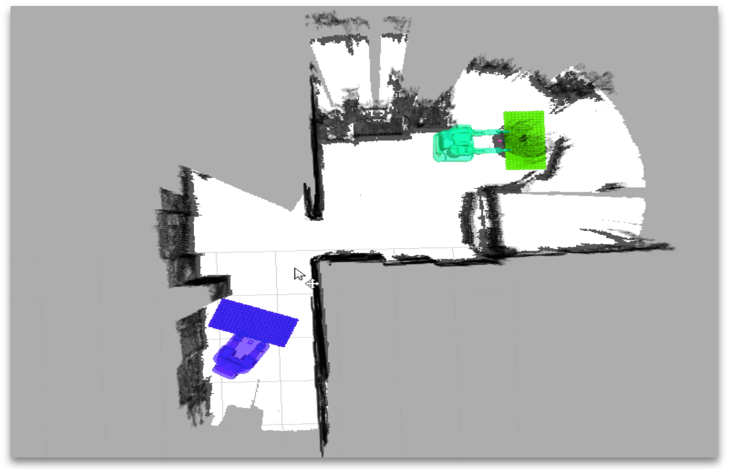
- Performance of informed search algorithms depends greatly on the quality of heuristics available
- Hard to design a single admissible heuristic that captures all the complexities of the problem, and is yet free of local-minima
- Easier to develop a set of inadmissible heuristics, each addressing a subset of the problem complexities
- Relieves the user from spending a lot of time engineering one great admissible heuristic--i.e, heuristics can now be actual ‘rules of thumb’
- Proposed algorithm uses several inadmissible heuristics in addition to one consistent heuristic for performing informed search, with provable guarantees on bounded suboptimality of the solution



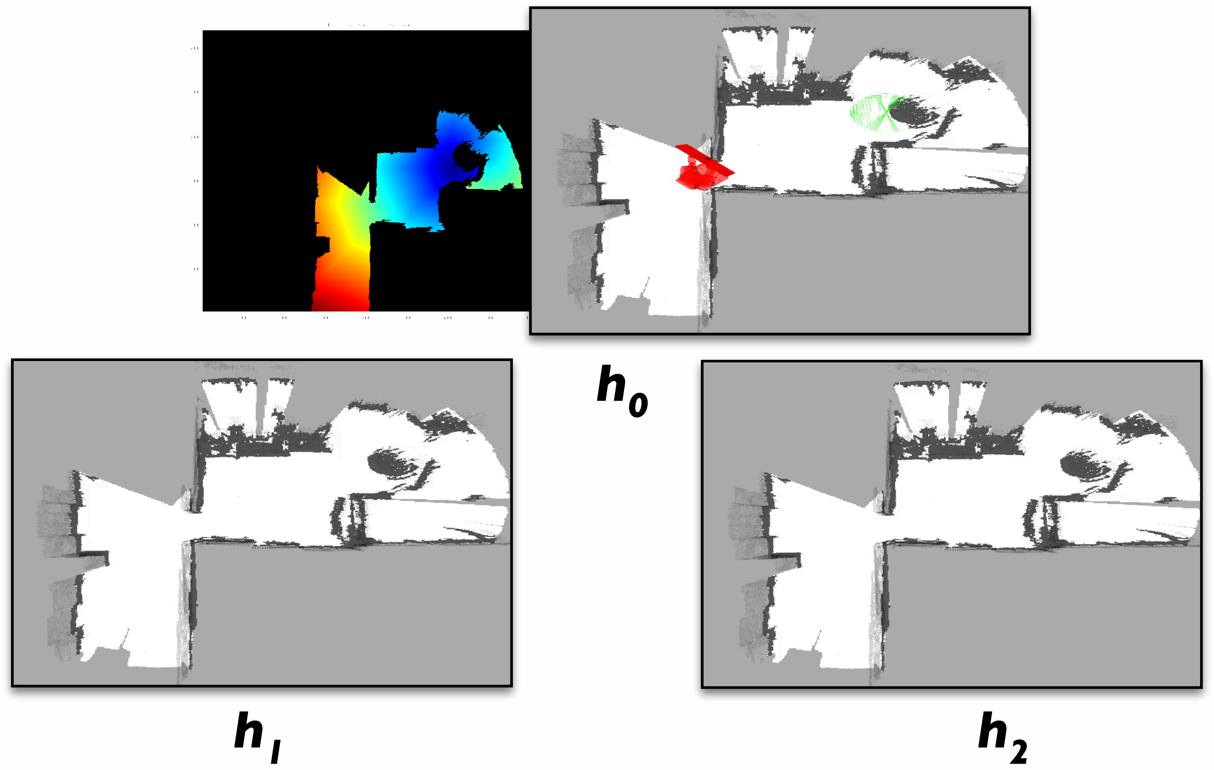
Designing “*local minima free*” heuristic is not easy for complex problems

An Example Problem

Full-body mobile manipulation: 12D planning

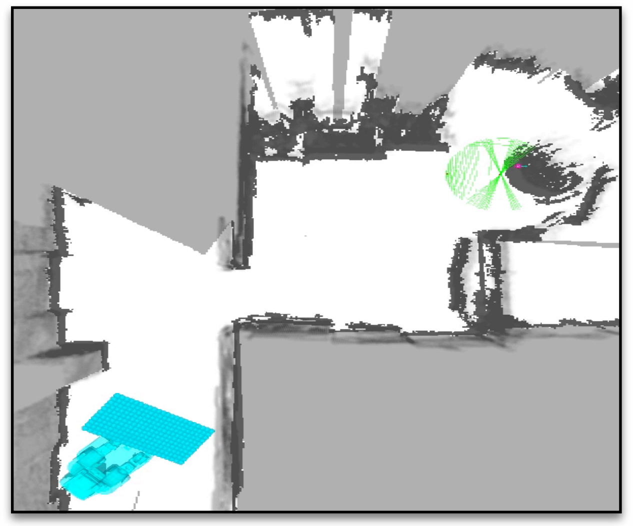


h_0: base distance (2D BFS)
h_1: base distance + orientation difference with goal
h_2: base distance + orientation difference with vertical



Multi-Heuristic A* (MHA*)

- 1 consistent heuristic (h_0) + n inadmissible heuristics (h_1, h_2 .., h_n)
- Round robin exploration

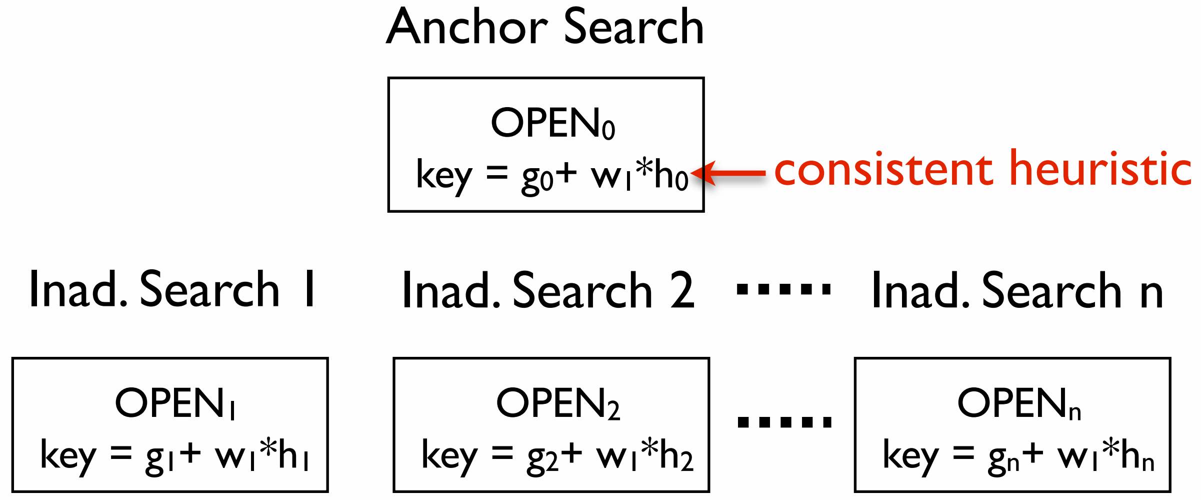


MHA*: Algorithms

- **Independent MHA* (IMHA*):** separate g, h values and priority queues for each search
- **Shared MHA* (SMHA*):** separate h values and queues for each search, but shared g

MHA*: Details

```
while goal state has not yet been expanded
    for i in 1:n
        if min. key(OPEN_i)<= w_2* min. key(OPEN_0)
            expand from OPEN_i
        else
            expand from OPEN_0
    end for
end while
```



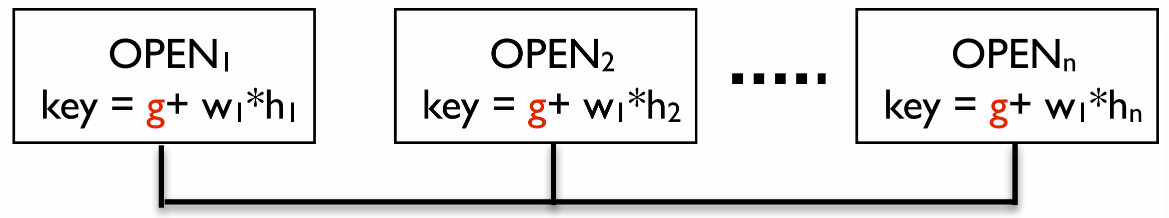
Anchor Search

OPEN$_0$
key = g_0+ w_1*h_0 ← consistent heuristic

Inad. Search 1   Inad. Search 2 ..... Inad. Search n

OPEN$_1$ key = g_1+ w_1*h_1 | OPEN$_2$ key = g_2+ w_1*h_2 ..... OPEN$_n$ key = g_n+ w_1*h_n

IMHA*: Independent Expansions

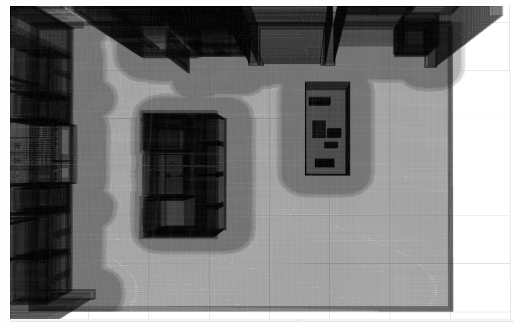


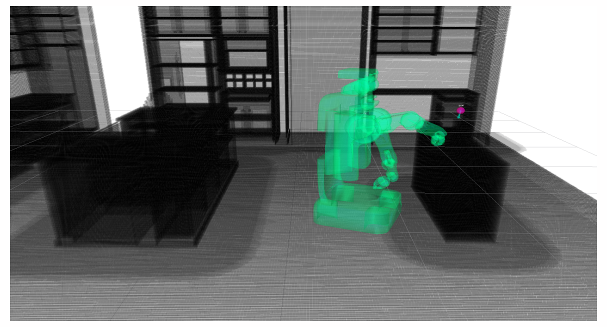
OPEN$_1$ key = g+ w_1*h_1 | OPEN$_2$ key = g+ w_1*h_2 ..... OPEN$_n$ key = g+ w_1*h_n

SMHA*: Shared Expansions
(successor states are inserted/updated in all queues)

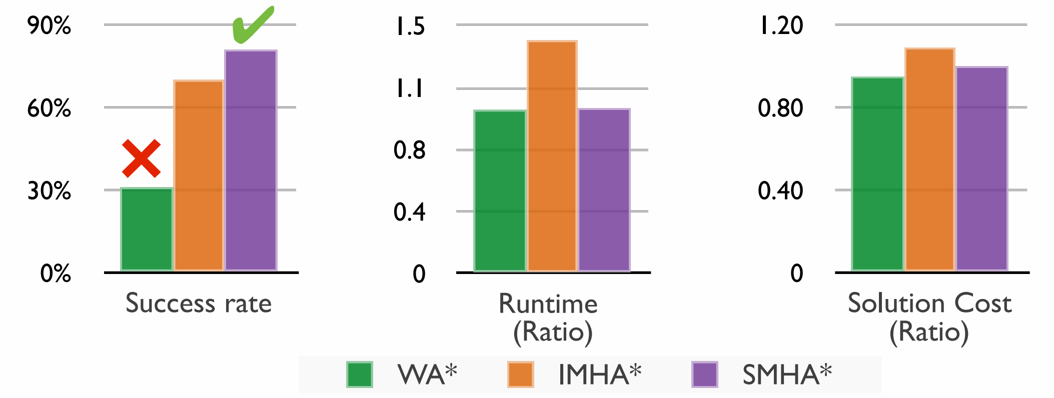
MHA* Properties	IMHA*	SMHA*
Subopt. bound	w_1*w_2	w_1*w_2
Max. re-expansions	n+1	2 ✓
Parallelization	Easy ✓	?
Path sharing	No	Yes ✓

Experiments: Mobile Manipulation

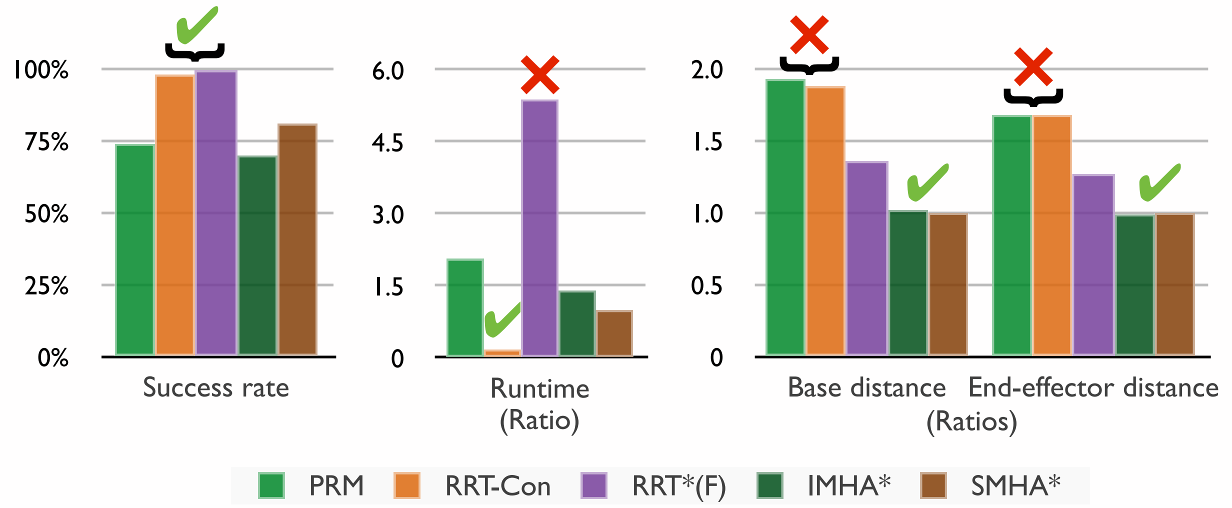




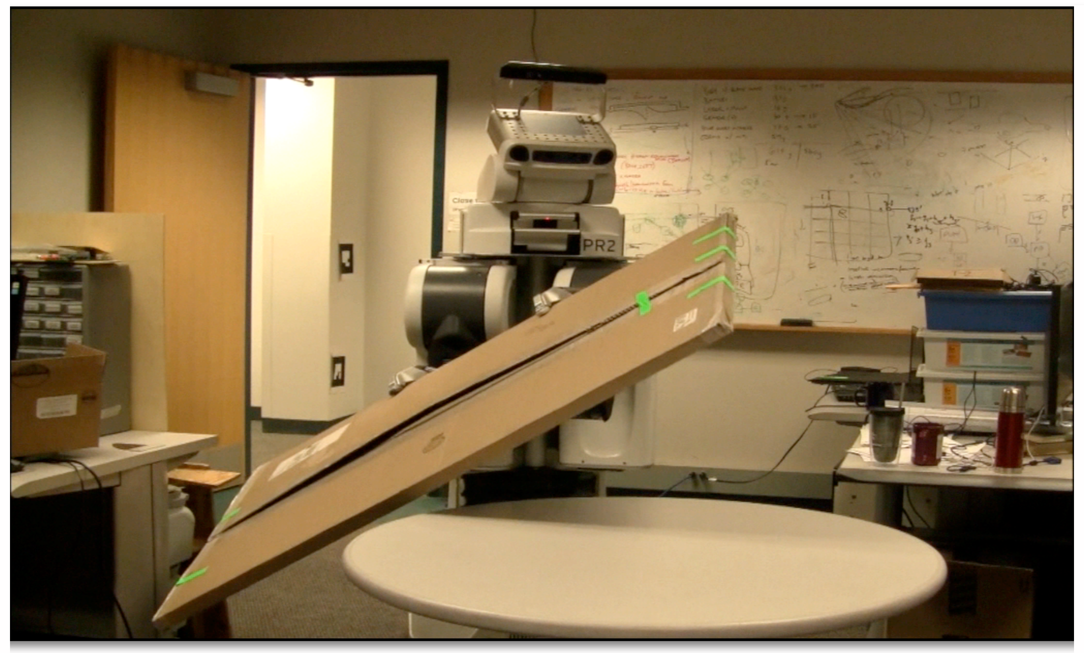
Comparison with Weighted A*



Comparison with Sampling-based Planners



Other domains: *3D navigation, Sliding tile puzzles*



[1] Fast Downward Search (Roger/Helmert ’10)
[2] Explicit Estimation Search (Thayer/Ruml ’11)
[3] MPWA* (Valenzano et al. ’10)