A*-Connect: Bounded Suboptimal Bidirectional Heuristic Search

Fahad Islam[†] Venkatraman Narayanan[*] Maxim Likhachev[*]

***Abstract*— The benefits of bidirectional planning over the unidirectional version are well established for motion planning in high-dimensional configuration spaces. While bidirectional approaches have been employed with great success in the context of sampling-based planners such as in RRT-Connect, they have not enjoyed popularity amongst search-based methods such as A*. The systematic nature of search-based algorithms, which often leads to consistent and high-quality paths, also enforces strict conditions for the connection of forward and backward searches. Admissible heuristics for the connection of forward and backward searches have been developed, but their computational complexity is a deterrent. In this work, we leverage recent advances in search with inadmissible heuristics to develop an algorithm called A*-Connect, much in the spirit of RRT-Connect. A*-Connect uses a fast approximation of the classic front-to-front heuristic from literature to lead the forward and backward searches towards each other, while retaining theoretical guarantees on completeness and bounded suboptimality. We validate A*-Connect on manipulation as well as navigation domains, comparing with popular sampling-based methods as well as state-of-the-art bidirectional search algorithms. Our results indicate that A*-Connect can provide several times speedup over unidirectional search while maintaining high solution quality.**

I. Introduction

Typically in motion planning problems, the planning complexity arises near the start and goal regions. Consider a standard robot manipulation scenario as shown in Figure 1 where a robot has to pick and place objects while operating in a cluttered space. Such tasks are common in industrial settings as well as in domestic robot applications, where the robot needs to manipulate through clutter and circumvent obstacles in order to accomplish the task. The problem is hard not only because of the high dimensionality of the state space but also because of the tight spaces in the robot's work space that translate to narrow corridors in the configuration space (C-Space). The problem is especially aggravated when clutter is present around the start or goal regions, causing unidirectional planning approaches to get stuck around the entrance to the induced narrow passage in C-space.

This led to the motivation of bidirectional planning methods. A popular sampling-based method, RRT-Connect [1], attempts to solve this problem by incrementally building two trees rooted at the start and the goal with a greedy heuristic that tries to connect each other. Bidirectional sampling- based methods like RRT-Connect have shown tremendous success in solving complex motion planning problems which involve tight spaces also referred to as "bug traps". Although such sampling-based approaches are extremely efficient in terms of finding a path very quickly, their solutions are often low quality and inconsistent for similar start-goal pairs, because of the inherent randomization. Search-based methods on the other hand generate good quality solutions but take longer due to their systematic exploration of the configuration space and the aforementioned limitations of unidirectional planning.

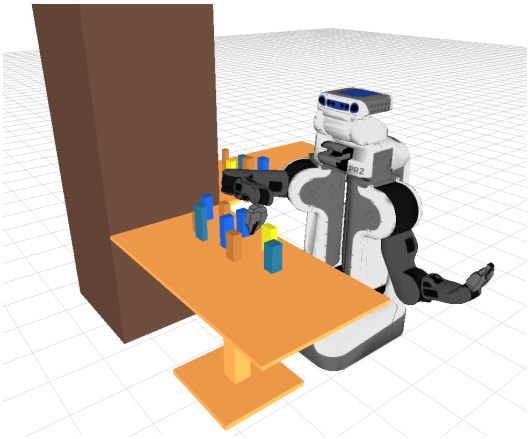

Fig. 1. A typical pick and place manipulation scenario in unstructured environments. Clutter around the start and goal introduce narrow passages in the configuration space that are problematic for unidirectional planning algorithms.

Although the idea of bidirectional planning has existed for long in the classical AI search literature, none of the existing approaches have proven outstandingly successful in solving complex motion planning problems. There has always been a trade off between time efficiency and the guarantee of optimality, the key trait of search-based planning methods. To come up with a bidirectional search strategy that is efficient in planning time and still provides good solution quality has always been a challenge.

The primary contribution of our work is a fast bounded-suboptimal bidirectional heuristic search algorithm called A*-Connect. It works by running bi-frontal searches both from the start and the goal configurations, as opposed to having single frontier searches. It runs a front-to-back and a fast-to-compute front-to-front heuristic search in parallel from each side, leveraging the framework of Multi-Heuristic A* (MHA*) [2] to do so. MHA* is a recently developed multi-heuristic search algorithm that allows for the use of multiple arbitrarily inadmissible heuristics while preserving guarantees on completeness and bounded suboptimality. By employing a bidirectional multi-heuristic search in a prin-

[†]School of Mechanical & Manufacturing Engineering (SMME), National Univeristy of Sciences & Technology (NUST), Islamabad, Pakistan fahad.islam@fulbrightmail.org

[*]The Robotics Institute, Carnegie Mellon University, Pittsburgh, USA {venkatraman,maxim} at cs.cmu.edu

cipled manner, A*-Connect is able to find solutions several times faster than a unidirectional search, while still guaranteeing bounded suboptimality. A*-Connect also achieves comparable planning times to popular sampling-based methods, but provides significantly better quality solutions.

II. Related Work

Sampling-based motion planners have been popular for high-dimensional planning problems in robotics, with the bidirectional variants being particularly useful in complex scenarios involving narrow passages and tight spaces. RRT-Connect, a bidirectional variant of the RRT, dynamically builds two trees from the initial and goal configuration and uses a greedy "Connect" heuristic to make the two trees meet. It is seen to show fast convergence as compared to the unidirectional RRT planner. While the original RRT-Connect does not provide any guarantees on the solution quality, recent variants of bidirectional sampling-based planners [3], [4] have been developed that provide *asymptotic* optimality guarantees.

In the heuristic search community, it has been counterintuitively observed that bidirectional search methods are more efficient for *uninformed* searches (i.e., searches where the heuristic knowledge is unavailable such as in breadth-first search) rather than *informed* searches. In the case of informed methods where heuristic guidance is available, bidirectional methods fall short of expectations. The search confronts a problem where the two frontiers pass by without intersection. Pohl metaphorically discusses this scenario as missiles passing each other [5]. In such a scenario at worse the search can expand twice as many nodes as an unidirectional search would. Kaindl and Kainz [6] argue against the missile metaphor, claiming that bidirectional search is inefficient not because the two frontiers pass by without intersection, but rather due to a major effort spent after the search frontiers meet to ensure optimality of the solution. In other words, even if the two frontiers overlap, the termination criteria for optimality requires a lot more expansions in practice.

The conception of the missile metaphor led to the development of different wave shaping algorithms such as BHFFA (Bidirectional Heuristic Front-to-Front Algorithm) [7]. These algorithms attempt to shape the wavefronts such that they intersect each other, by computing a 'front-to-front' heuristic that estimates the minimum distance from one frontier to the other. While it gives significant reductions in terms of number of expansions, the front-to-front heuristic computation is extremely expensive. Politowski and Pohl derived another method from BHFFA called d-node targeting [8] that guides the search to one central promising node in the opposite frontier, but the method compromised optimality assurance as the heuristic thus computed is inadmissible. Kwa [9] suggested optimization techniques to prune nodes by omitting a node that cannot contribute to the optimal solution or by avoiding re-expansion of a node that has already been expanded in the opposite search.

Weighted A* (WA*) [10], originally developed for speeding up unidirectional search methods by settling for bounded suboptimality, was extended to bidirectional methods to reduce the effort spent after the intersection of the two frontiers [11]. While this relaxes the termination condition for the bidirectional search, it aggravates the problem of the two frontiers passing each other (due to the inflated heuristics), unless other wave shaping methods are employed.

Single-Frontier Bidirectional Search (SF-BDS) [12] uses some variation of front-to-front evaluation to improve heuristic accuracy. It deals with a single frontier as opposed to two frontiers and every node in the search tree consists of a pair of states (one from the forward and one from the backward tree). Each node in this single frontier tree can be seen as an independent task of finding a path from a start to a goal. A major caveat for this method is that there are $O(V^2)$ possible tasks that can be created out of all possible pairs of states. SF-BDS allows multiple re-expansions of states which further aggravates the problem in spatial domains such as motion planning. An enhancement of SF-BDS called eSBS [13] attempts to reduce the number of re-expansions by pruning states that cannot contribute to the optimal path; however, it fails to provide any bounds on the number of re-expansions.

In comparison to these existing approaches, our proposed algorithm is fundamentally different. While it introduces a novel wave-shaping method which is computationally efficient, it also provides bounds on suboptimality and number of re-expansions. It uses an approximation of the front-to-front heuristic estimate instead of computing the exact estimate of cost to the opposite frontier. Although the approximation renders the heuristic inadmissible, the MHA* framework allows the algorithm to provide guarantees on suboptimality bounds.

III. Background: Multi-Heuristic A*

Since we use the Multi-Heuristic A* [2], [14] framework in our method, we first provide a brief summary of its working and properties.

Notations : We first assume that the planning problem can be represented as a path finding problem on an implicit graph. Throughout the paper, S denotes the finite set of states of the domain, $c(s, s')$ denotes the cost of the edge between s and s'; if there is no such edge, then $c(s, s') = \infty$. $\textsc{Children}(s) := \{s' \in S | c(s, s') \neq \infty\}$, denotes the set of all children of s. We use $c^*(s, s')$ to denote the cost of the optimal path from state s to s', $g(s)$ to denote the current best path cost from s_{start} to s, and $h(s)$ to denote the heuristic for state s, which is an estimate of the best path cost from s to s_{goal}. A heuristic is *admissible* if it never overestimates the best path cost to s_{goal} and *consistent* if it satisfies, $h(s_{goal}) = 0$ and $h(s) \leq h(s') + c(s, s')$, $\forall s, s'$ such that $s' \in \textsc{Children}(s)$ and $s \neq s_{goal}$. OPEN denotes a priority queue, and is typically implemented as a min-heap.

Optimal search algorithms such as A* [15] suffer from the curse of dimensionality when dealing with high-dimensional configuration spaces, due to the time and memory required for systematic exploration of the search space. WA* [10], a variant of A* that forsakes optimality for speed is often used in such high-dimensional problems. It uses a priority function

$f'(s) = g(s) + w * h(s)$ $(w > 1)$ to provide a greedy flavor to the search, which often results in faster termination [16], [17], [18]. WA* guarantees that the suboptimality of the solution is bounded by w times the optimal cost [19] if $h(s)$ is admissible, and does not require re-expansions to guarantee the bound if $h(s)$ is consistent [18].

While WA* has been shown to provide a speedup in many domains, it relies heavily on the accuracy of the heuristic function. If the heuristic is subject to local minima however, then WA*'s performance can degrade severely [20], [21] owing to its greedy nature. Multi-Heuristic A* (MHA*) [2] is a recently developed search algorithm that builds on the observation that while designing a single heuristic that is admissible, consistent and has shallow local minima is challenging for complex domains, it is often possible to design a number of inadmissible heuristics. MHA* uses multiple such (possibly inadmissible) heuristics to guide the search around local minima, by exploiting the synergy provided by these heuristics, each of which may be useful in different parts of the search space, while ensuring that bounds on suboptimality are still maintained.

Improved MHA* : In our work, we use an enhanced version of MHA* called Improved Multi-Heuristic A* [14], that can incorporate inadmissible heuristics that might be completely unrelated to edge costs on the graph. It has been shown to provide better performance than MHA* whenever tighter suboptimality bounds on solution quality are desired. Improved MHA* works very similar to WA*: it maintains a priority queue OPEN containing the frontier states, sorted by priority $g(s) + w * h(s)$ $(w > 1)$, with h being a consistent heuristic. Now, instead of simply expanding the state with the best priority like WA* does, Improved MHA* repeats the following until a solution is found or OPEN is empty: a) it first obtains a subset of OPEN called the P-SET, containing a list of 'promising' frontier states determined according to some criterion, b) given n inadmissible heuristics, it picks the n best states from P-SET greedily based on the inadmisisble heuristics, c) expands those n states and marks them as expanded *inadmissibly*, d) expands the state with the best priority in OPEN and marks it as expanded *admissibly*. At a high level, Improved MHA* interleaves an admissible or *anchor* search with an inadmissible search guided by multiple inadmissible heuristics. By choosing an appropriate termination criterion, Improved MHA* guarantees that the solution found is bounded suboptimal. Finally, it ensures that no state will be expanded more than twice—once admissibly and once inadmissibly.

IV. Our Contribution

We present a bidirectional search method that (a) uses an efficient approximation of the front-to-front heuristic, (b) provides solutions that are guaranteed to be bounded suboptimal with respect to the underlying search graph and (c) provides guarantees on the number of times a state in the graph is re-expanded.

Our algorithm builds upon the framework of Improved MHA*. Rather than using WA* searches for the backward and forward search, we use Improved MHA* searches that use two heuristics each: one guiding towards the opposite root, and one towards the opposite frontier. This results in a bidirectional dual frontier search where the search from each end has two frontiers—one progressing towards the opposite end and the other advancing towards the opposite frontier. Intuitively, the algorithm simultaneously runs a forward and a backward search, each of which is again a simultaneous front-to-front and front-to-back heuristic search. Next, we describe in detail how the admissible and the inadmissible searches of Improved MHA* are utilized in our proposed strategy.

A. Front-to-Back Heuristic Search

As discussed in the previous section, the Improved MHA* algorithm uses one consistent (and hence admissible) heuristic for the *anchor* search. We use an admissible front-to-back heuristic as the anchor heuristic for each of the two Improved MHA* searches. Front-to-back heuristics calculate the h value of a node s by using the heuristic estimate between s and the root of the opposite search tree (start/ goal). Front-to-back heuristics are typically inexpensive to compute because they are static—i.e., they do not depend on the frontier of the opposite search, but instead only on the root of the opposite search.

B. Front-to-Front Heuristic Search

The core of our algorithm lies in our front-to-front search methodology. The algorithm exploits the fact that Improved MHA* permits an inadmissible heuristic to compute a fast approximation of the usual front-to-front heuristic. To differentiate our front-to-front heuristic from the classical approach of estimating the distance between the two frontiers [7] we call it the *connect* heuristic. The h value of a node s for our connect heuristic is computed by finding the minimum heuristic estimate between s and the "most promising" last expanded state of the opposite search, rather than scanning through the complete frontier of the opposite search.

Connect Heuristic Explained: At a high level the connect heuristic is an approximation of the conventional front-to-front heuristic. Instead of getting an exact estimate of the minimum distance of a state s to the opposite frontier we compute an approximation of this distance. For the regular front-to-front search the cost of the path through every node in the opposite frontier needs to be computed. Instead, we compute the cost of the paths only through the last expanded states from the opposite search. This process is depicted in Figure 2 for a search iteration in forward direction. Assume we need to find a path from the start state S to the goal G (shown in blue). The dark grey circles show the frontier of the forward search whereas the light grey circles constitute the backward search frontier. The states P_0 and P_1 denote the last expanded states of the backward search from the anchor heuristic and the connect heuristic respectively. We call these states *pivots*. Now, whenever a heuristic update needs to be made for the connect heuristic, the algorithm

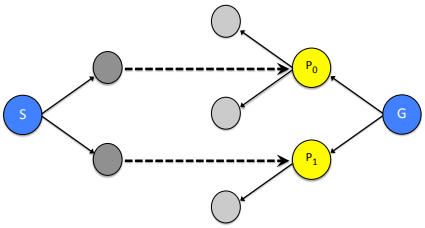

Fig. 2. Connect Heuristic Illustration.

only estimates the cost to these pivots and not the entire frontier (light grey states). This way it saves the computation time that is required by the classical front-to-front search (BHFFA) whose complexity is quadratic in the number of states in the two frontiers. For our approach the complexity is linear in the number of heuristics for each Improved MHA* search, which is 2 in this case.

It is trivial to explain how the algorithm can be generalized to use any number of inadmissible heuristic searches on top of the two existing ones, whether the heuristics be front-to-front, front-to-back or other domain dependent arbitrarily inadmissible searches [22].

V. Algorithm

A*-Connect (Alg. 1) builds upon Improved MHA*, which uses a single OPEN list for the frontier states and two CLOSED lists for keeping track of states expanded admissibly and inadmissibly. As A*-Connect runs two Improved MHA* searches, the OPEN and CLOSED lists are instantiated for both, the forward and the backward searches (lines 20 to 22). The symbols f and b represent the forward and backward directions respectively. The forward search queue is initialized with the start state whereas the backward search queue is initialized with the goal state (lines 26 and 27). In other words the goal state is taken as the start state for the backward search. The algorithm then repeatedly runs the Improved MHA* loop (lines 30 to 39) for the forward and backward directions, alternating between the two. Both the forward and the backward searches maintain independent data structures (OPEN/CLOSED lists and the P-SETs) and respective functions (CHILDREN(s), h(s) and g(s)). The superscript dir shows which direction a particular data structure or a function is associated to. $\overline{dir}$ denotes the direction complementary to dir. Then the algorithm repeatedly performs the following until a termination condition is satisfied: A subset of OPEN list called the potential set (P-SET) is constructed (line 32). P-SET contains the frontier states that are likely to lead to a bounded suboptimal solution and are determined according to P-CRITERION (line 5). Then the connect heuristic selects a state from the P-SET (line 33) based on the estimated domain-dependent distance ($\triangle$) to the *pivots* of the opposite search (line 8).

This state is marked as expanded by the connect heuristic (line 35). Finally the state at the top of OPEN is expanded admissibly (line 37). The entire process toggles for the search in the complementary direction, and the algorithm runs until a termination criteria is reached (line 29). After every expansion epoch, we check if any state has a valid path to it found from both the start and goal, and keep track of the state with the best estimated path cost ($u = g^f(s) + g^b(s)$) during every expansion (line 18). When this quantity satisfies the TERMINATIONCRITERION (line 4), the search terminates and returns the path computed by tracing the backpointers of the intersection state s in the forward and backward directions. The criterion states that the search should terminate when the best estimated path cost is not larger than the maximum PRIORITY observed for the admissibly expanded states in either of the two searches (forward and backward).

Algorithm 1 A*-Connect:

1: **procedure** PRIORITY(s, dir)
2: **return** $g^{dir}(s) + w.h^{dir}(s)$
3: **procedure** TERMINATIONCRITERION(u)
4: **return** $u \leq \max\,(\max_{s \in \text{CLOSED}_a^f} \text{PRIORITY}(s, f), \max_{s \in \text{CLOSED}_a^b} \text{PRIORITY}(s, b))$
5: **procedure** P-CRITERION(s, dir)
6: **return** $g^{dir}(s) + h^{dir}(s) \leq \max_{s \in \text{CLOSED}_a^{dir}} \text{PRIORITY}(s, dir)$
7: **procedure** CONNECTHEURISTIC(s, dir)
8: **return** $\min_{p \in (\text{P}_0{}^{dir}, \text{P}_1{}^{dir})} \triangle(s, p)$
9: **procedure** EXPANDSTATE(s, dir)
10: remove s from OPEN^{dir}
11: **for all** $s' \in$ CHILDREN$^{dir}(s)$ **do**
12: **if** s' was not seen before **then**
13: $g^{dir}(s') = \infty$
14: **if** $g^{dir}(s') > g^{dir}(s) + c(s, s')$ **then**
15: $g^{dir}(s') = g^{dir}(s) + c(s, s')$
16: **if** $s \notin$ CLOSED$_a{}^{dir}$ **then**
17: Insert/Update s' in OPEN^{dir} with PRIORITY(s', dir)
18: $u = min(u, g^f(s') + g^b(s'))$
19: **procedure** MAIN()
20: OPEN$^f \leftarrow \emptyset$, OPEN$^b \leftarrow \emptyset$
21: CLOSED$_a^f \leftarrow \emptyset$, CLOSED$_c^f \leftarrow \emptyset$
22: CLOSED$_a^b \leftarrow \emptyset$, CLOSED$_c^b \leftarrow \emptyset$
23: $g^f(s_{start}) \leftarrow 0$, $g^f(s_{goal}) \leftarrow \infty$
24: $g^b(s_{start}) \leftarrow \infty$, $g^b(s_{goal}) \leftarrow 0$
25: $u \leftarrow \infty$
26: Insert s_{start} in OPEN^f with PRIORITY(s_{start}, f)
27: Insert s_{goal} in OPEN^b with PRIORITY(s_{goal}, b)
28: $dir = f$
29: **while not** TERMINATIONCRITERION(u) **do**
30: **if** OPEN^f.EMPTY() **or** OPEN^b.EMPTY() **then**
31: **return** null
32: P-SET$^{dir} \leftarrow$ $\{s : s \in$ OPEN^{dir} $\wedge\ s \notin$ CLOSED$_c^{dir}$ $\wedge$ P-CRITERION$(s, dir)\}$
33: $P_1{}^{dir} \leftarrow \operatorname{argmin}_{s \in \text{P-SET}^{dir}} g^{dir}(s) + w \cdot$ CONNECTHEURISTIC(s, dir)
34: EXPANDSTATE($P_1{}^{dir}, dir$)
35: CLOSED$_c^{dir} \leftarrow$ CLOSED$_c^{dir} \cup \{P_1{}^{dir}\}$
36: $P_0{}^{dir} \leftarrow$ OPEN^{dir}.TOP()
37: EXPANDSTATE($P_0{}^{dir}, dir$)
38: CLOSED$_a^{dir} \leftarrow$ CLOSED$_a^{dir} \cup \{P_0{}^{dir}\}$
39: $dir = \overline{dir}$
40: **return** solution path

A. Theoretical Analysis

All the theoretical guarantees for Improved MHA* hold for A*-Connect:

Theorem 1: The cost of the solution returned by A*-Connect is no greater than w times the cost of the optimal solution.

Proof: (Sketch) The algorithm terminates when the TERMINATIONCRITERION (Alg. 1 line 4) is reached. Since the forward and backward searches are both Improved MHA* searches, they maintain the invariant that every expanded state (i.e, all states in the closed list) have a priority value that is no more than w of the optimal solution cost. Thus, the right hand side of the condition is also upper bounded by w times the optimal solution cost. The proof follows from the construction of u, which is the lowest cost path discovered so far by A*-Connect. ■

Theorem 2: Any state in the graph is expanded no more than 4 times by A*-Connect—at most twice by each, the forward and backward searches.

Proof: (Sketch) This statement makes direct use of Improved MHA*'s property that no state is expanded more than twice. Since we are essentially running two Improved MHA* searches on the same graph, a state could be expanded at most 4 times. ■

B. Implementation Details

We have already shown how our approach for front-to-front heuristic evaluation is significantly more efficient than the conventional front-to-front search. However we propose further optimization techniques to reduce the heuristic computation time. Instead of updating the heuristic values of the entire $OPEN$ list, we pick the top n candidates from the P-SET (the states with minimum key values) and only update the heuristic of these states. Although similar optimization can also be done for the conventional front-to-front search, the heuristic no longer retains admissibility, and thus optimality guarantees can not be provided. However with our approach we no longer require admissiblity of the connect heuristic and we can leverage this fact in our optimization. Another optimization we do is that instead of switching the direction of search every iteration we do it every k iterations (For all of our experiments we use k = 10). This reduces the number of times the heuristic updates are made. Other optimization approaches might also be used for A*-Connect such as making use of the meta-methods [22] for switching the search directions based on which one is making progress, or dynamic heuristic generation proposed in [23]. The key idea in the latter is to only use the connect heuristic when required instead of keeping the inadmissible search queue at all times.

VI. EXPERIMENTAL RESULTS

We tested our algorithm on two motion planning domains, 7 DOF Single-Arm planning and 3 DOF Navigation Planning. We compared our results against popular sampling-based and search-based planning algorithms. All experiments were performed on an Intel i7 - 4770 (3.4GHz) PC with 8GB RAM.

A. Single-Arm Planning

We evaluated the performance of A*-Connect on a manipulation planning problem for the 7 DOF PR2 robot arm. The start state and the goal state are both represented by fully defined 7 DOF configurations. This is different from the usual search-based arm planning approach [24] where the goal is under-specified as a 6 DOF object pose. We use it for A*-Connect as the backward search requires to be initialized by a fully defined goal state. This is identical to how bidirectional sampling-based planners operate in that they also require a fully defined goal pose. The environment setup we use for our testing is a common industrial manipulation setting where the robot needs to pick and place objects in clutter (Figure 3). These problems are challenging as they require the robot to manipulate through tight spaces while circumventing other objects. The search graph for the problem is constructed on the fly, by applying a set of motion primitives [25] to the state being expanded. These motion primitives are small kinodynamically feasible motions that the robot can execute.

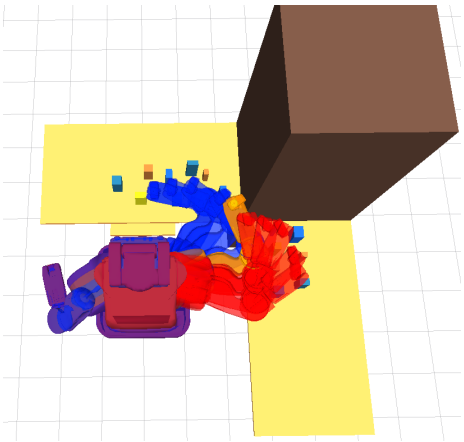

Fig. 3. Illustration of A*-Connect algorithm for 7 DOF Arm Planning domain. The arm visualizations in red, blue and orange show the forward segment of the path, the backward segment of the path and the intersection state respectively.

1) Heuristics: For arm planning we use two common and easy to compute heuristics. The h_{endeff} (end-effector heuristic) is computed by running a 3D Dijkstra's search from the goal position to compute the cost of the least-cost path from the end effector position at a given state to the end effector position at the goal state. The second heuristic $h_{euclidean}$ is computed as the Euclidean distance in 7 DOF space between two states. h_{endeff} is used as the anchor (admissible) heuristic and $h_{euclidean}$ works as the connect heuristic for our algorithm. As a connect heuristic we need a heuristic which is fast to compute and can be calculated on demand because the heuristic values need to be dynamically computed based on the opposite frontier. On the other hand, h_{endeff}, the admissible heuristic takes longer to compute, but is done only once before the search starts.

2) Adaptive Motion Primitives: We augment the set of predefined static motion primitives with adaptive motion primitives that are generated on the fly during the search [24]. The way we compute them differs from their original version because we use a fully defined goal state as opposed to an under-specified 6 DOF pose. For the forward search,

when a state s is expanded whose end-effector position is within a threshold distance from the goal state's end-effector position, we simply generate the goal state as a child if the interpolated path between s and the goal is collision-free and kinematically feasible. The same step is executed for the backward search by taking the start state end-effector position into account.

3) Adaptive Frontier Connection Primitive: In addition to the above adaptive motions, we introduce a third motion which we call adaptive frontier connection primitive. When a state s is expanded whose end-effector position is within a specified threshold from the end-effector position of a pivot from the opposite search, that pivot state is generated as a child if the interpolated segment between the two states is kinematically feasible and collision-free. This step is introduced to accelerate intersection of the frontiers when operating in high-dimensional state spaces.

4) Simulation Results and Comparisons: For comprehensive evaluation we compared A*-Connect against the unidirectional WA* search [10], the bidirectional best-first search (BHPA$_w$) [11] which basically is the bounded suboptimal version of the classical front-to-back search (BHPA) [5] and popular sampling-based algorithms: RRT Connect and RRT* from the Open Motion Planning Library (OMPL) [26]. The cost function for RRT* is the distance traveled in joint space. RRT* was configured to return the first solution that was found. For a fair comparison we provide the complete 7 DOF goal state and use the similar method of generating adaptive motions for the other two search-based methods. 100 experiments were run with randomly generated start and goal configurations of the arms such that the end-effector positions lie within the clutter shown over the tabletops. The results reported are averaged over these 100 tests. For all the runs, the search-based planners were initialized with w = 100 and a timeout of 30 seconds. We used the heuristic h_{endeff} for WA* and BHPA$_w$.

Table I shows the performance comparison of A*-Connect with the other planners. The metrics measured for each plan are the success rate, planning time, mean number of expansions (for the search-based methods) and the average path length for the joints measured in radians. Here, the success rate measures the number of trials in which a solution was found within the given time limit. The results are averaged over common successful trials. In terms of planning times although RRT-Connect outperforms our method, our method is certainly competitive in comparison to other search based and sampling-based planners. As far as path quality is concerned A*-Connect happens to provide shortest paths among all the other planners. Particularly, it provides significantly better quality solutions compared to RRT-Connect

TABLE I

PERFORMANCE COMPARISON OF A*-CONNECT WITH OTHER SEARCH-BASED AND SAMPLING-BASED ALGORITHMS FOR SINGLE-ARM PLANNING

	A*-Connect*	WA*	BHPA$_w$	RRTC	RRT*
Success Rate(percent)	100	51	83	100	90
Mean planning time(ms)	34.33	789.23	640.79	5.59	712.44
Mean state expansions	115	2093	1763	–	–
Mean path length(rad)	5.00	5.72	6.15	11.05	6.27

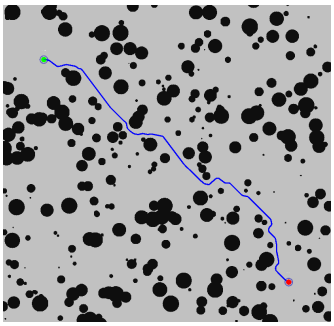

Fig. 4. Outdoor environment for navigation planning results with a size of 267 x 267 meters

B. Navigation Planning

We also evaluated our algorithm in the 3D navigation domain (x, y, θ) for non-holonomic robots (i.e, no turn-in-place or sideways motions). The tests were run on a randomly generated outdoor environment (Figure 4). The bidirectional planning search approach has strong application in such domains because non-holonomic constraints introduce local minima at the start and/or the goal. A*-Connect is effective in escaping these regions and can potentially intersect at other regions of the configuration space given an effective connect heuristic is employed.

1) Heuristics: For the navigation domain we use two heuristics. Similar to the h_{endeff} used in the arm-planning domain, we use the heuristic h_{base} (base-heuristic) that is computed by running a 2D Dijkstra's search from the goal position. The second heuristic is computed as the Euclidean distance in 3D (x, y, θ), $h_{euclidean}$, from a given state to the pivot state from the opposite search. The admissible h_{base} is used as the heuristic for the anchor search whereas the inadmissible $h_{euclidean}$ is used as the connect heuristic for A*-Connect.

2) Simulation Results and Comparisons: For the navigation domain we compare our results against the unidirectional WA*, BHPA$_w$ and additionally with the bidirectional front-to-front search approaches including the Bidirectional Heuristic Front-to-Front Algorithm (BHFFA) [7] and the enhanced Single Frontier Bidirectional Search (eSBS) [13]. We do not report comparisons with sampling-based methods for the navigation domain as they are meant to be useful mainly in higher dimensional planning problems. For WA* and BHPA$_w$ we used the heuristic h_{base} whereas for BHFFA and eSBS we used the heuristic $h_{euclidean}$.

We compare the performance of A*-Connect with the other methods against the metrics of planning time, number of expansions and the solution cost (Table II) for 100 tests with randomly generated 3 DOF start and goal configurations. For all the experiments the search-based planners were initialized with w = 3 and a timeout of 30 seconds.

TABLE II

PERFORMANCE COMPARISON OF A*-CONNECT WITH OTHER SEARCH-BASED ALGORITHMS FOR NAVIGATION PLANNING

	A*-Connect	WA*	BHPA$_w$	BHFFA	eSBS
Success Rate (percent)	100	100	100	48	70
Mean planning time (ms)	2.06	5.03	1.88	1254.37	179.31
Mean state expansions	256	1135	259	202	39659
Solution Cost (ratio)	1.0462	1.00	1.0013	1.0178	1.3059

TABLE III

PERFORMANCE COMPARISON OF A*-CONNECT ONLY WITH WA* AND BHPA$_w$ FOR NAVIGATION PLANNING

	A*-Connect	WA*	BHPA$_w$
Success Rate(percent)	100	100	100
Mean planning time(ms)	2.3	10.88	4.97
Mean state expansions	282	2509	907
Solution Cost	1.0767	1.00	1.0054

We show the solution cost as a ratio to the mean solution cost of the unidirectional WA* method, which tends to have the lowest solution cost. These results are computed for the common successful trials for all the planners. The BHPA$_w$ search performs significantly better in the navigation domain as the chances of intersection are high for lower dimensional problems. Also, these trials are relatively easier ones as these are taken for the cases where all the algorithms show success. However if we analyze the common successful trials only for A*-Connect, WA* and BHPA$_w$ (results averaged over all 100 experiments, Table III), A*-Connect provides better planning times on average, compared to the other methods. In general other methods either suffer because of the number of expansions or the computation cost per expansion. In terms of solution quality our methods provides almost similar solution costs as the WA* search.

VII. CONCLUSIONS

We presented a bidirectional heuristic search algorithm, A*-Connect, for computing fast bounded-suboptimal solutions. A*-Connect uses the Improved Multi-Heuristic A* framework to run bi-frontal searches in both forward and backward directions, with one frontier solely devoted to accelerating the connection of the two searches. We demonstrated how A*-Connect is much faster than unidirectional and state-of-the-art bidirectional heuristic searches, while still providing guarantees on completeness and bounded suboptimality. In our experiments on manipulation planning, A*-Connect provides superior quality solutions compared to popular sampling-based planners for comparable planning times. Our future work involves generalizing A*-Connect to multi-root multi-frontal searches for addressing common bottlenecks in motion planning.

ACKNOWLEDGMENTS

This work was supported by NSF grant IIS-1409549 and ONR grant N00014-15-1-2129.

REFERENCES

[1] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *ICRA*, pp. 995–1001, IEEE, 2000.

[2] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic A*," in *Proceedings of Robotics: Science and Systems*, (Berkeley, USA), July 2014.

[3] M. Jordan and A. Perez, "Optimal bidirectional rapidly-exploring random trees," 2013.

[4] J. Starek, E. Schmerling, L. Janson, and M. Pavone, "Bidirectional fast marching trees: An optimal sampling-based algorithm for bidirectional motion planning," in *Workshop on Algorithmic Foundations of Robotics*, 2014.

[5] I. Pohl, *Bi-directional and heuristic search in path problems*. PhD thesis, Dept. of Computer Science, Stanford University., 1969.

[6] H. Kaindl and G. Kainz, "Bidirectional heuristic search reconsidered," *Journal of Artificial Intelligence Research*, pp. 283–317, 1997.

[7] D. de Champeaux, "Bidirectional heuristic search again," *Journal of the ACM (JACM)*, vol. 30, no. 1, pp. 22–32, 1983.

[8] G. Politowski and I. Pohl, "D-node retargeting in bidirectional heuristic search.," in *AAAI*, pp. 274–277, 1984.

[9] J. B. Kwa, "Bs: An admissible bidirectional staged heuristic search algorithm," *Artificial Intelligence*, vol. 38, no. 1, pp. 95–109, 1989.

[10] I. Pohl, "First results on the effect of error in heuristic search," *Machine Intelligence*, vol. 5, pp. 219–236, 1970.

[11] A. L. Koll and H. Kaindl, "Bidirectional best-first search with bounded error: Summary of results," in *Proceedings of the 13th international joint conference on Artifical intelligence-Volume 1*, pp. 217–223, Morgan Kaufmann Publishers Inc., 1993.

[12] C. Moldenhauer, A. Felner, N. Sturtevant, and J. Schaeffer, "Single-frontier bidirectional search," in *Third Annual Symposium on Combinatorial Search*, 2010.

[13] M. Lippi, M. Ernandes, and A. Felner, "Efficient single frontier bidirectional search.," in *SOCS*, 2012.

[14] V. Narayanan, S. Aine, and M. Likhachev, "Improved multi-heuristic a* for searching with uncalibrated heuristics," in *Eighth Annual Symposium on Combinatorial Search*, 2015.

[15] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, pp. 100–107, July 1968.

[16] B. Bonet and H. Geffner, "Planning as heuristic search," *Artificial Intelligence*, vol. 129, no. 1-2, pp. 5–33, 2001.

[17] R. Zhou and E. A. Hansen, "Multiple sequence alignment using anytime a*," in *Proceedings of 18th National Conference on Artificial Intelligence AAAI'2002*, pp. 975–976, 2002.

[18] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems 16*, Cambridge, MA: MIT Press, 2004.

[19] J. Pearl, *Heuristics: intelligent search strategies for computer problem solving*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1984.

[20] C. Hernández and J. A. Baier, "Avoiding and escaping depressions in real-time heuristic search," *J. Artif. Intell. Res. (JAIR)*, vol. 43, pp. 523–570, 2012.

[21] C. M. Wilt and W. Ruml, "When does weighted A* fail?," in *SOCS*, AAAI Press, 2012.

[22] M. Phillips, V. Narayanan, S. Aine, and M. Likhachev, "Efficient search with an ensemble of heuristics," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence. AAAI Press (to appear)*, 2015.

[23] F. Islam, V. Narayanan, and M. Likhachev, "Dynamic Multi-Heuristic A*," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, IEEE, 2015.

[24] B. J. Cohen, G. Subramanian, S. Chitta, and M. Likhachev, "Planning for manipulation with adaptive motion primitives," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 5478–5485, IEEE, 2011.

[25] B. Cohen, S. Chitta, and M. Likhachev, "Single-and dual-arm motion planning with heuristic search," *The International Journal of Robotics Research*, p. 0278364913507983, 2013.

[26] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, pp. 72–82, December 2012.