# Towards Narrowing the Search in Bounded-Suboptimal Safe Interval Path Planning

**Tomas Rybecky,**[1,2] **Miroslav Kulich,**[2] **Anton Andreychuk,**[3,4] **Konstantin Yakovlev**[3,5]
[1]Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University
[2]Czech Institute for Informatics, Robotics and Cybernetics, Czech Technical University
[3]Federal Research Center for Computer Science and Control of Russian Academy of Sciences
[4]Peoples Friendship University of Russia (RUDN University)
[5]National Research University Higher School of Economics
tomas.rybecky@cvut.cz, kulich@cvut.cz, andreychuk@mail.com, yakovlev@isa.ru

## Abstract

Path planning in the presence of dynamic obstacles is challenging as the time dimension has to be considered. A prominent approach to tackle this problem known to be complete and optimal is the A*-based Safe-interval Path Planning (SIPP). Bounded-suboptimal variants of SIPP employing the ideas of Weighted A* (WSIPP) and Focal Search (FocalSIPP) have been introduced recently, trading-off optimality for decreased planning time. In this paper, we revisit FocalSIPP and design several secondary heuristics for Focal Search with the intention to narrow the search in the direction of a preplanned optimal single-agent path not considering dynamic obstacles. The experimental results on various maps show that the designed heuristics generally outperform the hops-to-the-goal heuristic used in the original FocalSIPP and successfully compete with WSIPP as well.

## Introduction

Path planning for an agent operating in a static environment is a well-studied problem often reduced in AI/Robotics communities to graph search. While the traditional search algorithms, such as A* (Hart, Nilsson, and Raphael 1968) or Dijkstra (Dijkstra 1959), guarantee finding optimal paths, the problem gets challenging when other moving entities are present in the environment as well. These can be either other agents that are under the control of the planner, in which case the general problem is referred to as Multi-Agent Path-Finding (MAPF), or uncontrolled dynamic obstacles that the agent has to avoid (e.g., pedestrians).

When planning with dynamic obstacles, the planner's required output is not only a path in space but also the time moments at which the transitions should be made. In conventional setups, waiting in place is also considered, so the plan may include both move and wait actions.

In many real-world applications, e.g., in mobile robotics, the speed of finding a feasible plan might be of a higher priority than its quality, i.e., the cost, which is typically defined as the time it takes to reach the destination. On the other hand, getting plans of extremely high costs may be undesirable as well. Meeting both of the requirements are the bounded-suboptimal planners, which provide theoretical guarantees that the cost of the resultant solution will not exceed the cost of the optimal solution by some margin (which can be either additive or multiplicative). Meanwhile, they typically find the solutions much faster.

In this work, we introduce several novel secondary heuristics for the bounded-suboptimal FocalSIPP. These are based on the idea of sticking to the optimal plans that do not take the dynamic obstacles into account. We empirically compare them with state-of-the-art methods, showing that in certain setups, the usage of the suggested methods reduces the cost and search complexity by up to 80%.



## State of the Art

A prominent algorithm to find a feasible path in the presence of dynamic obstacles is the Safe Interval Path Planning (SIPP), which is provably optimal and complete (Phillips and Likhachev 2011).

An anytime version of SIPP (Anytime SIPP) finds the initial bounded-suboptimal solution and then constantly improves it while the allocated computational budget permits (Narayanan, Phillips, and Likhachev 2012). One of the core components of Anytime SIPP is the bounded-suboptimal (BS) version of SIPP that takes as a parameter the suboptimality factor, $w$, and returns the plan, $\pi$, s.t. $cost(\pi) \leq w \cdot cost(\pi^*)$, where $\pi^*$ is the optimal solution. The BS SIPP used in Anytime SIPP is the adaptation of the prominent Weighted A* (Pohl 1970) algorithm that explicitly prohibits re-expansions by introducing two copies of the certain states.

In (Yakovlev, Andreychuk, and Stern 2020) two other versions of BS SIPP were considered. Weighted SIPP (WSIPP) again relies on a Weighted A*, but allows re-expansions instead of duplicates (therefore WSIPPr). The second one, FocalSIPP, adapts the ideas of Focal Search (FS) (Pearl and Kim 1982). FS needs two heuristic functions: the primary one, which has to be admissible to guarantee meeting the suboptimality bound, and the secondary one, which is intended to aid the search progress faster towards the goal and does not have to be admissible. Designing the latter heuristic is not trivial. In the considered domain, hops-to-the-goal (HTG) heuristic (Wilt and Ruml 2014) is widely used, navi-

gating the search by the number of expected necessary steps needed to reach the goal. In this work, we introduce several novel secondary heuristics for FocalSIPP and study them empirically.

## Problem Statement

Consider a graph representing a grid-like workspace composed of the adjacent tiles. The vertices of the graph match the centers of the tiles, and edges fit the transitions between them (we assume the grid is either 4- or 8-connected). Each edge has a weight corresponding to the duration of the transition. We assume that the agent moves with constant speed and the duration is 1 for transition between orthogonally-adjacent cells and $\sqrt{2}$ for the diagonally-adjacent ones.

Certain vertices/edges are blocked for certain time intervals due to the moving obstacles that populate the environment. Consequently, *safe intervals* are defined as the inversions of the blocked ones. A plan required from the planner is a set of the move and wait actions that reaches the predefined goal vertex from the predefined start vertex while avoiding collisions with the dynamic obstacles by visiting all vertices and edges only in their respective safe intervals.

The cost of a plan is the time moment by which the agent reaches the goal (we assume that the timeline starts at 0). In this work, we are interested in getting bounded-suboptimal solutions, i.e., the plans which cost is not greater than the cost of the optimal plan multiplied by a user-defined factor $w$ (the suboptimality bound).

## Algorithm

In this section, we shortly summarize the function of Safe Interval Path Planning (SIPP). Then we build on its bounded-suboptimal modifications, the WSIPP and FocalSIPP, and propose several new suboptimal heuristics to be used with them to improve some aspects of the search.

The search space for SIPP is comprised of the elements defined as $n = (v, [t, t'])$, where $v$ is the graph vertex and $[t, t']$ is the safe time interval (thus different search nodes that correspond to the same graph vertex might be encountered). For each search node SIPP stores its predecessor, $p(n)$, and the $g$-value, which is the earliest time by which $n$ can be reached via $p(n)$. Indeed, $g(n) \in [t, t']$. The search frontier, OPEN, is maintained and at each iteration the node with minimum $f$-value is extracted from OPEN for the expansion, where $f(n) = g(n) + h(n)$, where $h(n)$ is the consistent heuristic estimating time to reach the goal from $n$. When expanding the node and generating successors, SIPP takes care that their safe intervals are properly aligned, and the transition is valid w.r.t. dynamic obstacles (this involves augmenting the transition with the preceding wait action if needed). Overall, SIPP can be seen as the A*-search in the complex search space composed of the vertex-time interval pairs augmented with an involved procedure for successor generation. Indeed, SIPP is provably complete and optimal.

There are two existing bounded-suboptimal versions of SIPP: WSIPP and FocalSIPP. WSIPP inflates the heuristic for the underlying A* and must either allow re-expansions (WSIPPr) or introduce duplicate states (WSIPPd) to avoid them. FocalSIPP maintains the so-called FOCAL list (besides OPEN) and chooses the nodes for expansion from it utilizing the secondary heuristic, which does not have to be consistent or even admissible. In (Yakovlev, Andreychuk, and Stern 2020) HTG was used as the secondary heuristic for FocalSIPP. While both WSIPP and FocalSIPP provide the bounded-suboptimal solutions faster than SIPP by skipping several search tree branches, we want to motivate the suboptimal search more purposefully.

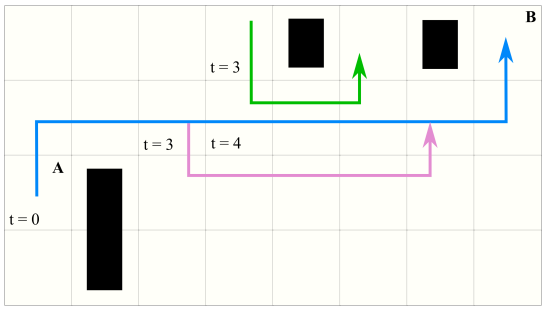



Figure 1: An example of an optimal single-agent path (*blue*) that only needs a local update (*purple*) when being applied to an area with a dynamic obstacle (*green*).

For that, we take motivation from the D*Lite (Koenig and Likhachev 2005) algorithm, which is an effective tool for updating a single-agent path based on new knowledge about the search space. To achieve similar behavior in a dynamic environment, we want to narrow the search to a preplanned optimal single-agent path and only modify its parts that could lead to collisions. An example is shown in Figure 1, where we consider the 4-neighborhood and a speed of one tile per discrete timestep. At $t = 4$, the optimal single-agent path from **A** to **B** (*blue*) reaches a collision interval with a dynamic obstacle (*green*). Our target is to quickly find a local detour (*purple*) with minimum new expansions.

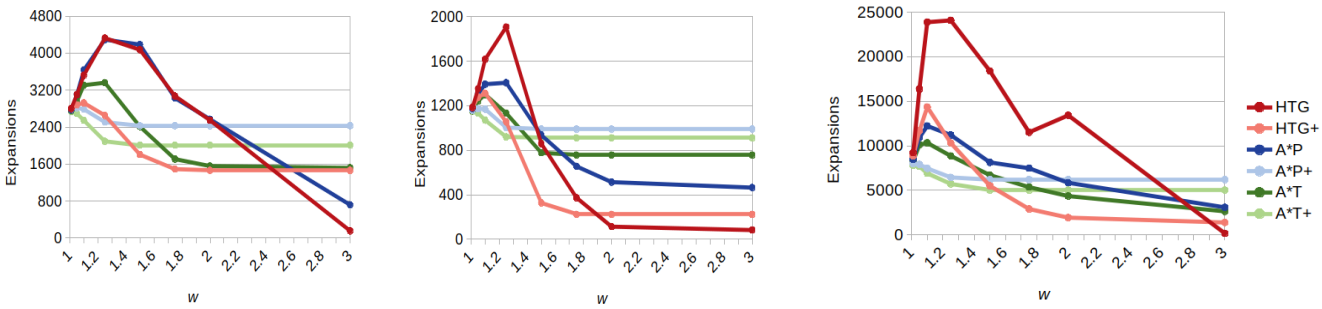
Using the single-agent path as a lead, we want to speed up the search by skipping the potentially uninteresting branches of the search tree. Since we try to stick to a path that is optimal in a less crowded area, we expect the approach to not increase the plan cost extensively. Moreover, when the search space would be too crowded and the original optimal path would require extensive waiting, the architecture of FocalSIPP allows finding a cheaper solution elsewhere.

## Heuristics

FocalSIPP uses two heuristics to compute the cost of the states and sort the OPEN and FOCAL lists. The OPEN list uses the standard f-value based on the cost of the path $g(n)$ to the node $n$ and an admissible heuristic estimate to goal $h(n)$:

$$f(n) = g(n) + h(n) \quad (1)$$

Since SIPP works with time, the cost-so-far element $g(n)$ represents the time of reaching the vertex $n$. The estimate $h(n)$ is computed as the Manhattan distance (for connectivity set to 4) or an octile distance (for connectivity set to 8) to goal divided by the constant agent speed, which is assumed




(a) Room (b) Warehouse (c) den520d

Figure 2: The number of expanded states by different approaches for sorting the FOCAL list in FocalSIPP.

unitary. This notion of time cost is proved to maintain the heuristic's consistency and admissibility (Yuan et al. 2017).

For the FOCAL list, we propose two new heuristics which aim to narrow the search to the optimal single-agent path $\pi^*$ not considering the dynamic obstacles and two different ways of sorting the list itself. The first heuristic works by evaluating the expanded nodes by their distance $d$ (Manhattan or octile, according to the connectivity) from $\pi^*$, i.e., the distance $\delta$ from the closest node in the path.

$$d_P(n, \pi^*) = \min_{v \in \pi^*} \delta(n, v) \tag{2}$$

The second heuristic computes the distance from the node $\pi^*(t)$ at which the agent would be at the time $t$ at which it reaches $n$.

$$d_T(n, \pi^*, t) = \delta(n, \pi^*(t)) \tag{3}$$

For comparison, we take the HTG heuristic used in FocalSIPP. That estimates the number of nodes $d_{HTG}(n)$ the plan would pass on the shortest path from $n$ to the goal if no dynamic obstacles are encountered.

Analogically to HTG in the original FocalSIPP, the values from Eq. (2) and Eq. (3) can be directly used to sort the nodes in FOCAL list.

$$h_{FOCAL}(n) = d(n, \pi^*, t), \tag{4}$$

where $d$ equals $d_P$, $d_T$ or $d_{HTG}$ and takes only the arguments required by the corresponding version.

Additionally, we further reduce the suboptimal relaxation by adding the admissible f-value of $n$ to the three heuristics:

$$h^+_{FOCAL}(n) = d(n, \pi^*, t) + f(n), \tag{5}$$

where $d$ equals $d_P$, $d_T$ or $d_{HTG}$ (with the corresponding arguments) and it is transformed to time under the same assumption of unit speed as the admissible $h$ value.

## Experiments

In the experimental part, we mainly evaluate the impact of the allowed suboptimality bound $w$ on the planning results combined with different heuristics. We compare the proposed sorting methods for FocalSIPP to the optimal SIPP planner, the suboptimal WSIPPr, and the original FocalSIPP with the hops-to-goal (HTG) heuristic. The particular version of WSIPP was chosen based on its better performance with lower suboptimality, which is also the goal of our approach. Finally, we try to apply the HTG estimate to goal for WSIPPr as well, using the inflated f-value $f(n) = g(n) + w \cdot d_{HTG}(n)$.

For the experiments, we use the Moving AI benchmark maps (Stern et al. 2019) Room ($64 \times 64$), Warehouse ($161 \times 63$) and den520d ($256 \times 257$). Due to the limited space, we present results on maps using an 8-connected grid, as the 4-connected variants showed similar trends. For Room and Warehouse, we work with 250 dynamic obstacles; den520d was filled with 1000 of these. We evaluated 100 unique instances on each of the maps and display the average values.

The different versions of FocalSIPP are labeled as follows, based on the $h_{FOCAL}$ used to sort the FOCAL list:

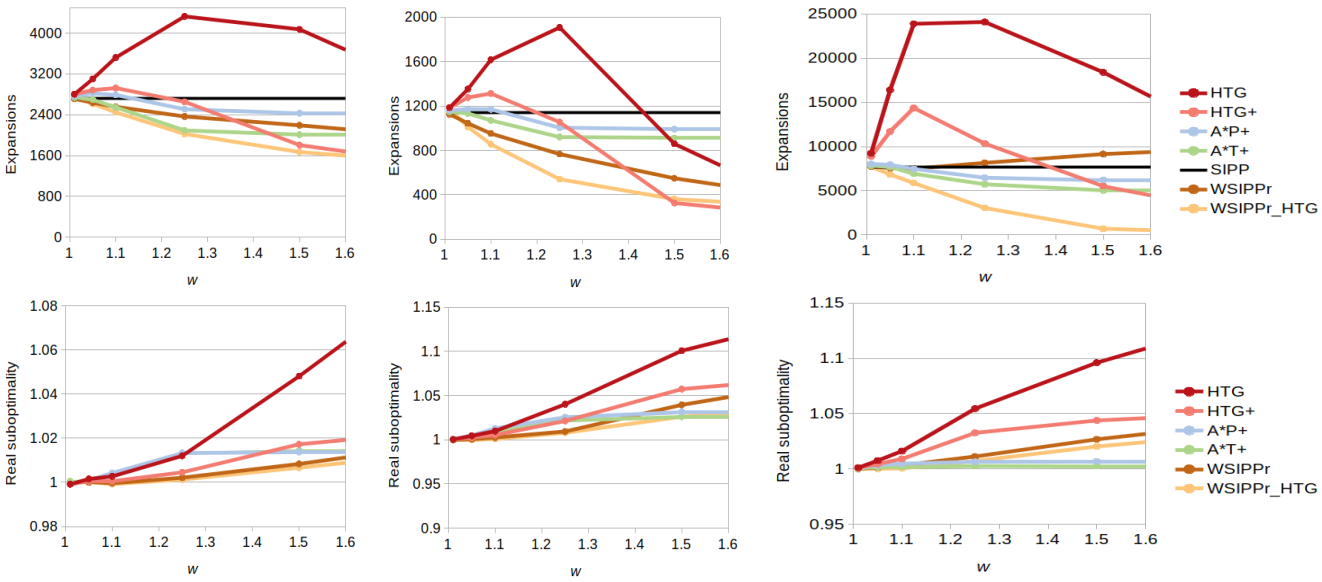
- A*P - the distance from the optimal single-agent path according to Eq. (2)
- A*T - the distance from the node in the optimal single-agent path visited at the same time according to Eq. (3)
- HTG - the estimated number of steps to reach the goal

These are additionally labelled A*P+, A*T+ and HTG+ when the $h^+_{FOCAL}$ is used according to Eq. (5). The optimal single-agent paths (for A*P and A*T), as well as the shortest path node count from each vertex (for HTG), are precomputed prior to the planning.

In Figure 2 we show the number of states expanded by FocalSIPP using each of the secondary heuristics. The experiments were performed with the suboptimality bound $w$ ranging from 1.01 to 5, but we focus on the lower values for better readability, as the ongoing trends do not significantly change later.

The first difference can be seen in the way the heuristic estimate is applied, i.e., if we sort the FOCAL list only by the value itself or if we add the admissible f-value as well. While in the first approach, the number of expansions quickly rises for $w < 1.3$, for the latter, the maximal increase is about 50% lower and stops with a lower $w$. This is most probably caused by the f-value immediately navigating the search towards the goal, while the estimate alone leads to a broader search for low suboptimality bounds until the greedy aspect takes over with higher $w$.

The proposed approaches penalizing the distance from an optimal single-agent path lead to a faster decrease of the performed expansions, especially for lower $w$. They work the




(a) Room (b) Warehouse (c) den520d

Figure 3: The number of expanded states (first row) and the ratio between achieved and optimal cost (second row) by the selected bounded-suboptimal versions of SIPP.

fastest in combination with the f-value until they get outperformed by HTG+ around $w = 1.4$. In most cases, standard HTG requires the most expansions, while it benefits the most from adding the admissible f-value in HTG+.

In Figure 3 we compare the selected best performing FocalSIPP secondary heuristics to the optimal solution produced by SIPP and two versions of the bounded-suboptimal WSIPPr planner using either octile distance or HTG to estimate the cost to the goal. The cost of the resulting plans is compared based on their cost ratio to the cost of the optimal solution. We focus on the most dynamic range of $w$ values for the plots.

The WSIPPr versions greedily aim for the goal, and with $w < 2$, they usually need half of the expansions to find it compared to SIPP, while the cost of their results rises proportionally. For WSIPPr, the HTG cost estimate returns overall better results in both the number of expanded states and plan cost, which is natural in less connected grids.

Among the FocalSIPP versions, the proposed heuristics aiming to follow the single-agent optimum generally compete better with WSIPPr in the number of expansions when the bound $w$ is lower, with better results achieved by A*T. For higher $w$, A*P and A*T maintain a constant near-optimal cost while reducing the number of expansions of SIPP by 20-30%. The speed-up of the other solvers grows higher, but the cost of their solutions keeps rising by units to tens of percent. The main reason for the near-optimal cost achieved by the new heuristics is probably in the application of the admissible f-value, and since HTG does not utilize that nearly as well, there is also a strong impact of the navigation towards the single-agent optimum.

In all presented methods, the runtime corresponds to the number of performed expansions and is therefore not displayed. Also, all secondary heuristics used in FocalSIPP require an initialization phase prior to planning. For HTG, the initialization stands for precomputing the expected number of steps to a goal from all nodes of the graph using Dijkstra, while for the proposed heuristics, it is finding the optimal single-agent path with A*. These are generally lower by one order than the overall planning time, and for HTG, the value is about 4-5 times larger than for the A*P and A*T versions.

## Conclusions

In this paper, we present a comparison of several methods for finding a path in an area with dynamic obstacles. Bounded-suboptimal Safe Interval Path Planning algorithm proves to be a powerful tool for quickly retrieving a solution in the given situation, while there are still ways of improving its performance.

Motivated by the single-agent path updates in D* Lite, we propose new heuristics for the FOCAL list in FocalSIPP. A*P and A*T show overall better results than the original method used in FocalSIPP, the hops-to-goal estimate. Even with a low suboptimality bound, the new methods quickly reduce the number of expanded states, while with higher $w$, they reasonably compete with Weighted SIPP in the cost of the results. Although the choice of the algorithm depends on the application, WSIPP maintains its overall top position, especially with the newly applied HTG estimate, which proves its quality in less-connected grids.

Additionally, we show that further limiting the suboptimal relaxation by adding the optimal f-value improves the performance of bounded-suboptimal solvers, especially for lower suboptimality bounds.

## Acknowledgments

The work has been supported by the European Regional Development Fund under the project Robotics for Industry 4.0 (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000470) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS21/185/OHK3/3T/37. Anton Andreychuk is supported by the RUDN University Strategic Academic Leadership Program.

## References

Andreychuk, A.; Yakovlev, K.; Atzmon, D.; and Stern, R. 2019. Multi-Agent Pathfinding with Continuous Time. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 39–45. International Joint Conferences on Artificial Intelligence Organization.

Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1: 269–271.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics* 4(2): 100–107.

Koenig, S.; and Likhachev, M. 2005. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics* 21(3): 354–363.

Likhachev, M.; Gordon, G.; and Thrun, S. 2003. ARA*: Anytime A* with Provable Bounds on Sub-Optimality. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, 767–774.

Mors, A.; Witteveen, C.; Zutt, J.; and Kuipers, F. 2010. Context-Aware Route Planning. In *8th German Conference on Multiagent system technologies, MATES 2010*, 138–149.

Narayanan, V.; Phillips, M.; and Likhachev, M. 2012. Anytime Safe Interval Path Planning for dynamic environments. In *Proceedings 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4708–4715.

Pearl, J.; and Kim, J. H. 1982. Studies in Semi-Admissible Heuristics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-4(4): 392–399.

Phillips, M.; and Likhachev, M. 2011. SIPP: Safe interval path planning for dynamic environments. In *Proceedings 2011 IEEE International Conference on Robotics and Automation*, 5628–5635.

Pohl, I. 1970. Heuristic search viewed as path finding in a graph. *Artificial Intelligence* 1(3): 193–204.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Boyarski, E.; and Bartak, R. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. *Symposium on Combinatorial Search (SoCS)* 151–158.

Wilt, C. M.; and Ruml, W. 2014. Speedy versus Greedy Search. In *Symposium on Combinatorial Search (SoCS)*.

Yakovlev, K.; Andreychuk, A.; and Stern, R. 2020. Revisiting Bounded-Suboptimal Safe Interval Path Planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 300–304.

Yuan, W.; Ganganath, N.; Cheng, C.; Qing, G.; and Lau, F. C. M. 2017. A consistent heuristic for efficient path planning on mobility maps. In *Proceedings 2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 1–5.