Risk-Bounded Multi-Agent Visual Navigation via Dynamic Budget Allocation

Viraj Parimi*, **Brian Williams**
Massachusetts Institute of Technology

Abstract: Safe navigation is essential for autonomous systems operating in hazardous environments, especially when multiple agents must coordinate using just visual inputs over extended time horizons. Traditional planning methods excel at solving long-horizon tasks but rely on predefined distance metrics, while safe Reinforcement Learning (RL) can learn complex behaviors using high-dimensional inputs yet struggles with multi-agent, goal-conditioned scenarios. Recent work combined these paradigms by leveraging goal-conditioned RL (GCRL) to build an intermediate graph from replay buffer states, pruning unsafe edges, and using Conflict-Based Search (CBS) for multi-agent path planning. Although effective, this graph-pruning approach can be overly conservative, limiting mission efficiency by precluding missions that must traverse high-risk regions. To address this limitation, we propose RB-CBS, a novel extension to CBS that dynamically allocates and adjusts user-specified risk bound (Δ) across agents to flexibly trade off safety and speed. Our improved planner ensures that each agent receives a local risk budget (δ) enabling more efficient navigation while still respecting overall safety constraints. Experimental results demonstrate that this iterative risk-allocation framework yields superior performance in complex environments, allowing multiple agents to find collision-free paths within the user-specified Δ.

Keywords: Multi-Agent Planning, Risk-bounded Planning, Visual Navigation

1 Introduction

Safe and efficient navigation of multiple autonomous agents remains a critical requirement in domains such as disaster relief [1], large-scale inspections [2], and environmental monitoring [3]. In these settings, failures are costly not only in terms of resources but also in potential harm to human operators or the environment. Existing Multi-Agent Path Finding (MAPF) methods range from centralized approaches that ensure optimality but face scalability challenges to decentralized ones prioritizing scalability [4, 5, 6, 7, 8, 9]. A leading two-level method that balances these properties is Conflict-Based Search (CBS) [10]. Its modular design has spurred a host of enhancements [11, 12] and extensions to various problems [13, 14, 15]. However, they typically rely on graph representations that explicitly encode valid transitions and associated metrics for each agent. Such an assumption does not hold when agents operate in visually rich or otherwise unstructured environments with only high-level inputs like images.

Meanwhile, goal-conditioned reinforcement learning (GCRL) [16, 17, 18] can learn navigation policies directly from images, making it well suited for unstructured scenarios. However, GCRL alone struggles when tasks span extended horizons [19, 20], especially if agents must balance risk mitigation with the desire to travel efficiently over long distances. Constrained RL [21] approaches offer a way to incorporate safety, but ensuring that multiple agents coordinate amongst them to reach their goals while remaining within some global risk bound is still an open challenge.

*Correspondence to vparimi@mit.edu.

Workshop on Safe and Robust Robot Learning for Operation in the Real World at CoRL 2025.

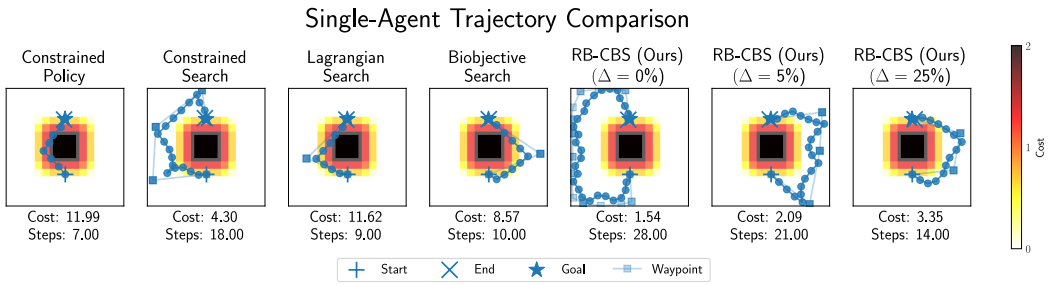

Figure 1: Comparison of different approaches on the 2D navigation problem for a single agent setup. Total cumulative cost and average steps count are annotated below each method. While standard baselines often cut directly through high-cost regions leading to riskier paths, our RB-CBS planner flexibly trades off efficiency for safety by adapting its route to the user-specified Δ.

Previous work [22, 23] bridged these gaps by integrating GCRL with graph-based planners. They constructed an intermediate graph from replay buffer states and learned distance and risk functions. Further, they pruned edges deemed unsafe, yielding safer waypoint-based plans that respect learned safety metrics. Finally, they employed MAPF approaches, particularly CBS to coordinate multiple agents over these waypoint graphs. Despite their effectiveness, this approach leads to conservatively pruned graphs, which may overly restrict agents' paths.

This issue becomes even more apparent in scenarios such as search-and-rescue, where the goal like a victim or critical resource may lie in a high-risk region of the environment. Simply pruning out these hazardous areas or assigning an overly strict risk allowance can render the mission impossible, since the agent has no choice but to enter the risky zone. On the contrary, by allowing a small yet controlled portion of risk, the agent can still complete its objective while maintaining reasonable safety guarantees, and often significantly reduce travel time or overall mission costs.

In this paper, we build on top of this prior framework to address these shortcomings by introducing a global user-specified risk bound (Δ) and an iterative risk-allocation mechanism. Instead of statically pruning edges in a conservative fashion, our approach dynamically adjusts local risk budgets of each agent, enabling a more balanced trade-off between safety and efficiency. Through iterative updates, agents collectively adjust their paths so that the total risk remains within the global risk bound (Δ) while still avoiding excessive conservatism. Our iterative risk-allocation approach ensures that this trade-off is reflected at the planning level, balancing the need for safety with the necessity of fulfilling the mission in a timely manner, even in high-dimensional visual domains.

The rest of the paper is as follows. First, we begin with preliminaries in Section 2. We then detail our proposed iterative risk-allocation algorithm in Section 3. Section 4 presents extensive experiments demonstrating that our method improves mission time and scalability against multiple baselines while adhering to user-specified safety constraints. Finally, we conclude in Section 5.

2 Preliminaries

We consider the problem of multi-agent navigation in visually rich environments, where each agent must reach a goal while ensuring that the *total accumulated risk* across all agents does not exceed a global risk bound Δ.

2.1 Learned Distance and Cost Graph

We train a GCRL agent to estimate both distance and risk between observations using critics Q and Q_C, following Feng et al.. From the agent's replay buffer, we sample states $\mathcal{B}$ to construct a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}_d, \mathcal{W}_c)$, where vertices are $\mathcal{V} = \mathcal{B}$ and a fully-connected set of edges $\mathcal{E}$ is assumed. The weights for an edge $e_{s_i \to s_j}$ are derived from the learned critics. The cost weight is the direct output, $\mathcal{W}_c(e_{s_i \to s_j}) = c_\pi(s_i, s_j)$, while the distance weight is thresholded by a maximum

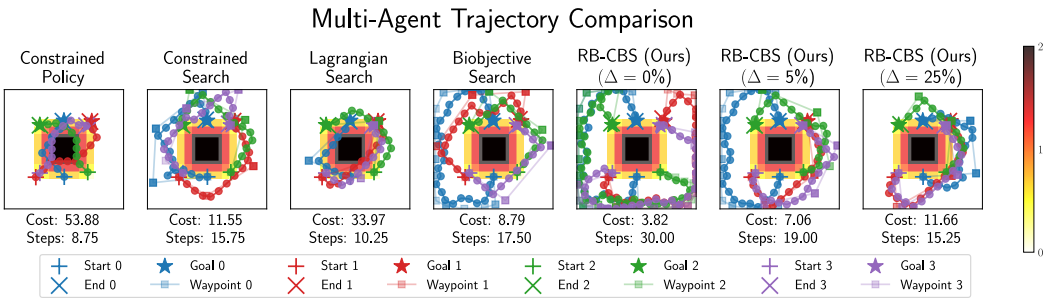

Figure 2: Comparison of different approaches on the 2D navigation problem for a multi-agent setup. Total cumulative cost and average steps count are annotated below each method. Unlike the baselines which cut through high-cost regions leading to riskier paths, our RB-CBS planner coordinates all agents and flexibly adjusts each route to the user-specified Δ

value $\mathrm{d}_{\max}$:

$$\mathcal{W}_d(e_{s_i \to s_j}) = \begin{cases} d_\pi(s_i, s_j), & \text{if } d_\pi(s_i, s_j) < \mathrm{d}_{\max}, \\ \infty, & \text{otherwise}, \end{cases}$$

where the predicted distance $d_\pi(s_i, s_j)$ and cost $c_\pi(s_i, s_j)$ are obtained from the critics Q and Q_C, respectively.

Unlike Feng et al., we do not prune edges based on predicted costs. All edges with valid distances are retained, including those that may lead through high-risk regions. The cost critic $\mathcal{W}_c$ instead provides additional risk information to the planner, which is used to reason about trade-offs under a global risk bound.

2.2 Global Risk-Bounded MAPF

Let $\mathcal{A} = \{a_1, \ldots, a_N\}$ denote a set of N agents, each with a start-goal pair $(s_i, g_i) \in \mathcal{V}$. The objective is to find a set of collision-free paths $\{\pi_1, \ldots, \pi_N\}$, where each path π_i connects s_i to g_i, such that the total accumulated risk across all agents is bounded by a user-specified global risk bound Δ. We define the risk of a single path π_i as, $\rho(\pi_i) = \sum_{(u,v)\in\pi_i} \mathcal{W}_c(u, v)$, where $\mathcal{W}_c(u, v)$ is the learned risk cost associated with edge (u, v). The global constraint is given by, $\sum_{i=1}^{N} \rho(\pi_i) \leq \Delta$. This formulation allows the planner to flexibly allocate risk budgets to agents, enabling traversal through risky regions when necessary, while ensuring that the total risk remains within Δ.

3 Risk-Bounded Conflict-Based Search

To address the Multi-Agent Path Finding (MAPF) problem under a global risk bound Δ, we propose Risk-Bounded Conflict-Based Search (RB-CBS), an extension of the standard CBS framework. RB-CBS is a two-level algorithm designed to find a set of conflict-free paths for N agents, $\Pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$, such that the total cumulative risk of all paths does not exceed Δ. The algorithm operates by managing not only spatio-temporal constraints to resolve collisions but also a dynamic *risk budget* (δ) for each agent.

3.1 High-Level Search

The high-level search of RB-CBS explores a Constraint Tree (CT), similar to standard CBS. However, each node $\mathcal{P}$ in our tree is augmented with information essential for dynamic risk allocation. A node $\mathcal{P}$ contains the standard set of spatio-temporal constraints, $\boldsymbol{\mathcal{C}}$, and a set of paths, $\boldsymbol{\Pi}$, along with their total cost, $\boldsymbol{J}(\boldsymbol{\Pi})$. To manage risk, each node also includes a vector of local risk budgets, $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_N]$, which must sum to less than the global budget Δ. Finally, a boolean vector, $\boldsymbol{\phi}$, tracks whether a valid path satisfying both the constraints and the local budget δ_i has been successfully found for each agent.

The high-level search maintains a frontier set $\mathcal{F}$, which is a priority queue of CT nodes, prioritized first by their cost ($\boldsymbol{J}(\boldsymbol{\Pi})$), then by the number of collisions in their paths ($\boldsymbol{\Pi}$), and finally by the number of agents whose risk allocations were recently changed. This encourages the search to first expand nodes that are both low cost and stable in their risk distribution. The main loop starts by finding initial paths for all agents (ignoring risk) and calculates an initial risk allocation using a chosen strategy described later forming the root of the constraint tree. It then repeatedly pops the most promising node from the frontier set where it first checks if all agents have paths within their assigned risk budgets. If not, it implies that the node was created after a risk reallocation, and it now tries to find paths for those affected agents with their new budgets.

If all paths are found then the search proceeds like standard CBS where it detects collisions and if there are no collisions and the total path risk is under Δ, a solution is returned. If there are collisions, it splits the node by creating new constraints using disjoint split [24] for the colliding agents and generates new successor nodes to explore. If some paths are not found within their respective risk-budgets the search reallocates risk budgets among the agents. This complete high-level search procedure is formally outlined in Algorithm 1

Algorithm 1: RB-CBS High-Level Search

Create root node $\mathcal{P}_0$ with initial (unconstrained) paths and risk allocations.
$\mathcal{F}$.Insert($\mathcal{P}_0$)
while $\mathcal{F}$ not empty and time not exceeded **do**
  $\mathcal{P} \leftarrow \mathcal{F}$.ExtractMin() ▷ *Pop node from frontier set with the highest priority*
  if $\exists\ i$ s.t. $\mathcal{P}.\phi_i = 0$ **then**
    Re-plan paths for agents with $\phi_i = 0$ within their new budget δ_i
    if any replan fails **then**
      $\mathcal{A}_{\text{fail}} \leftarrow$ set of failing agents
      $\delta' \leftarrow$ REALLOCATERISK($\mathcal{P}, \mathcal{A}_{\text{fail}}$) ▷ *See Algorithm 2*
      if $\delta' \neq$ FAIL **then**
        Create successor node $\mathcal{P}'$ from $\mathcal{P}$ with δ' as new allocations
        $\mathcal{F}$.Insert($\mathcal{P}'$)
      end if
      continue
    end if
  end if
  $\mathcal{P}.\mathcal{C} \leftarrow$ DETECTCOLLISIONS($\mathcal{P}.\Pi$) ▷ *All paths satisfy their assigned budgets; proceed to conflict resolution*
  if $\mathcal{P}.\mathcal{C}$ is empty and $\sum_{i=1}^{N} \rho(\pi_i) \leq \Delta$ **then**
    return $\mathcal{P}.\Pi$
  end if
  $c \leftarrow$ Choose a collision from $\mathcal{P}.\mathcal{C}$
  Generate constraints to satisfy the collision c using disjoint split
  for each new constraint on agent a_k **do**
    Create successor node $\mathcal{P}'$ by adding the constraint
    $\mathcal{P}'.\pi_k \leftarrow$ Re-plan for a_k within its budget δ_k
    if path found **then**
      $\mathcal{P}'.J(\Pi) \leftarrow$ SUMOFCOSTS($\mathcal{P}'.\Pi$)
      $\mathcal{F}$.Insert($\mathcal{P}'$)
    else
      $\delta' \leftarrow$ REALLOCATERISK($\mathcal{P}', \{a_k\}$) ▷ *Low-level search failed due to new constraint so we reallocate new risk*
      if $\delta' \neq$ FAIL **then**
        Update successor node $\mathcal{P}'$ with δ' as the new allocations
        $\mathcal{F}$.Insert($\mathcal{P}'$)
      end if
    end if
  end for
end while
return No solution found.

3.2 Low-Level Search

The low-level planner for RB-CBS is a variant of A* (RBA*) algorithm. For a given agent a_i, its constraints, and an assigned risk budget δ_i, the RBA* planner finds a path π_i that minimizes path length while ensuring that $\rho(\pi_i) \leq \delta_i$. If no such path exists, the planner returns a failure. To speed up the performance of this low-level planner, we make use of the experience re-use as outlined by Shaoul et al..

3.3 Risk Budget Management

Initial Budget Allocation. At the root node of the CT, an initial risk allocation is computed. We implement three strategies for this initial allocation, based on a *utility* metric u_i for each agent's initial (unconstrained) path:

- **Uniform**: The budget is divided equally: $\delta_i = \Delta/N$.
- **Utility-based**: The budget is allocated proportionally to the utility (e.g., path risk): $\delta_i = \Delta \cdot \frac{u_i}{\sum_{j=1}^{N} u_j}$. This gives more budget to agents with inherently riskier paths.
- **Inverse Utility-based**: The budget is allocated inversely proportional to the utility (e.g., path length): $\delta_i = \Delta \cdot \frac{1/u_i}{\sum_{j=1}^{N}(1/u_j)}$. This strategy prioritizes shorter, potentially riskier paths by allocating them a larger share of the risk budget, based on the intuition that longer paths may already be safer by nature of avoiding direct routes.

Risk Reallocation Mechanism. During the search, a low-level RBA* call may fail for an agent if no path exists within its currently allocated budget δ_i. This triggers the risk reallocation procedure, detailed in Algorithm 2, which attempts to borrow budget from other agents. The procedure first identifies the failing agents, $\mathcal{A}_{\text{fail}}$, and computes their total required budget increase, δ_{req}. This is achieved by running a risk-minimizing low-level search for each failing agent to find its minimum feasible risk, $\delta_i^{\min}$, and summing the deficits (lines 3–6). Concurrently, it calculates the total available surplus budget, δ_{avail}, by determining the minimum feasible risk for each passing agent and summing the difference from their current allocation (lines 8–11).

A feasibility check is then performed where if the required budget exceeds the available surplus ($\delta_{\text{req}} > \delta_{\text{avail}}$), the re-allocation is impossible, and the current CT node is pruned. Otherwise, a new risk allocation δ' is created by satisfying the needs of the failing agents and reducing the budgets of the passing agents to cover the deficit (lines 15–27). This new node, with its updated risk distribution, is then pushed back onto the frontier set $\mathcal{F}$ for future exploration.

Algorithm 2: Risk Reallocation Procedure

1: **function** REALLOCATERISK($\mathcal{P}$, $\mathcal{A}_{\text{fail}}$)
2: $\quad\delta_{\text{req}} \leftarrow 0$
3: $\quad$**for** $a_i \in \mathcal{A}_{\text{fail}}$ **do**
4: $\quad\quad\delta_i^{\min} \leftarrow$ Minimum feasible risk for a_i via low-level search with constraints imposed by $\mathcal{P}$
5: $\quad\quad\delta_{\text{req}} \leftarrow \delta_{\text{req}} + (\delta_i^{\min} - \mathcal{P}.\delta_i)$
6: $\quad$**end for**
7: $\quad\delta_{\text{avail}} \leftarrow 0$
8: $\quad$**for** $a_j \notin \mathcal{A}_{\text{fail}}$ **do**
9: $\quad\quad\delta_j^{\min} \leftarrow$ Minimum feasible risk for a_j via low-level search with constraints imposed by $\mathcal{P}$
10: $\quad\quad\delta_{\text{avail}} \leftarrow \delta_{\text{avail}} + (\mathcal{P}.\delta_j - \delta_j^{\min})$
11: $\quad$**end for**
12: $\quad$**if** $\delta_{\text{req}} > \delta_{\text{avail}}$ **then**
13: $\quad\quad$**return** FAIL ▷ *Not enough surplus budget*
14: $\quad$**end if**
15: $\quad\delta' \leftarrow \mathcal{P}.\delta$
16: $\quad$**for** $a_i \in \mathcal{A}_{\text{fail}}$ **do**
17: $\quad\quad\delta'_i \leftarrow \delta_i^{\min}$ ▷ *Give failing agents what they need*
18: $\quad$**end for**
19: $\quad\delta_{\text{rem}} \leftarrow \delta_{\text{req}}$ ▷ *Take surplus from passing agents*
20: $\quad$**for** $a_j \notin \mathcal{A}_{\text{fail}}$ **do**
21: $\quad\quad\Delta\delta_j \leftarrow \mathcal{P}.\delta_j - \delta_j^{\min}$
22: $\quad\quad\epsilon_j \leftarrow \min(\Delta\delta_j, \delta_{\text{rem}})$
23: $\quad\quad\delta'_j \leftarrow \delta'_j - \epsilon_j$
24: $\quad\quad\delta_{\text{rem}} \leftarrow \delta_{\text{rem}} - \epsilon_j$
25: $\quad\quad$**if** $\delta_{\text{rem}} \leq 0$ **then break**
26: $\quad\quad$**end if**
27: $\quad$**end for**
28: $\quad$**return** δ'
29: **end function**

3.4 Collision Checking on Irregular Graphs

Unlike standard MAPF formulations that operate on regular grids or lattice graphs with implicit spatial structure, our learned waypoint graph $\mathcal{G}$ is irregular and lacks any global coordinate system. In such non-grid graphs, conventional notions of conflict such as vertex collisions or edge collisions are insufficient. Due to the lack of regularity in node connectivity, agents can follow distinct but spatially intersecting edges, resulting in physical collisions that are not captured by these standard definitions. Addressing this requires reasoning about potential edge-edge intersections, which cannot be resolved using only the graph topology.

To detect these collisions, we model each agent as a disc of fixed radius r. When two agents traverse distinct graph edges (p_o, p_1) and (q_o, q_1), their positions over normalized time $\tau \in [0, 1]$ can be represented as, $P(\tau) = p_0 + \tau(p_1 - p_0)$ and $Q(\tau) = q_0 + \tau(q_1 - q_0)$. The squared distance between

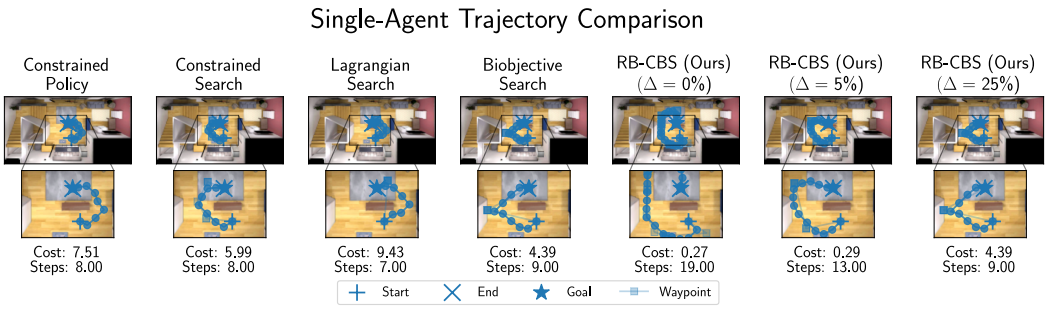

Figure 3: Single-agent trajectory comparison on the visual navigation task. Total cumulative cost and average steps count are annotated below each method. Our RB-CBS planner flexibly trades off efficiency and safety via the user-specified Δ. (See Appendix 15 for the corresponding multi-agent trajectory comparison.)

them, $f(\tau) = \|P(\tau) - Q(\tau)\|^2$, is a quadratic in time. By defining the relative offset $\mathbf{D} = p_0 - q_0$ and relative velocity $\mathbf{V} = (p_1 - p_0) - (q_1 - q_0)$, we can express this quadratic in its standard form:

$$f(\tau) = \|\mathbf{D} + \tau\mathbf{V}\|^2 = a\tau^2 + b\tau + c, \tag{1}$$

where the coefficients are given by the dot products $a = \mathbf{V} \cdot \mathbf{V}$, $b = 2\mathbf{D} \cdot \mathbf{V}$, and $c = \mathbf{D} \cdot \mathbf{D}$.

The minimum of this quadratic occurs at the unconstrained minimizer $\tau^* = -b/(2a)$. We clamp this value to the valid time interval, $\tau_{\text{clamped}} = \min(1, \max(0, \tau^*))$, to find the time of closest approach. A collision is then reported if this minimum distance is less than or equal to the sum of the radii.

This test is applied at every timestep for each pair of concurrently moving agents enabling the detection of geometric edge-edge intersections that are not captured by conventional vertex or edge conflict definitions. As a result, RB-CBS can resolve spatial interactions in irregular graphs induced from visual environments. The coefficients of this squared distance quadratic can be efficiently computed using the pairwise distance matrix. More details can be found in the Appendix.

4 Experiments

Our experiments evaluate RB-CBS against baseline methods by addressing three key questions regarding its adaptability, goal-achieving capability, and scalability.

Q1: Can RB-CBS showcase agent behavior that respects the user-specified risk-bounds compared to baselines?
Q2: Does RB-CBS sacrifice its ability to reach distant goals when balancing the accumulated cost compared to other methods?
Q3: Does RB-CBS effectively demonstrate superior performance when scaled to a larger number of agents compared to baselines?

To ensure consistent benchmarking, we adopt the environments and learned models from Feng et al., including both 2D point environment and visually complex Habitat scenarios [26, 27, 28]. In the visual navigation tasks, agents operate using only first-person RGB observations from four cardinal directions, without access to a global map or ground-truth cost information.

Defining Risk Bounds. Since our approach relies on a user-specified risk budget, Δ, we first establish a meaningful range for this parameter on a per-problem basis. For each instance, we solve it twice using CBS. First we use it to minimize total path length (yielding an upper risk bound, $\overline{\Delta}$), and then again to minimize total cumulative risk (yielding a lower risk bound, $\underline{\Delta}$). This defines the feasible risk interval $[\underline{\Delta}, \overline{\Delta}]$. We then test all methods at five distinct risk levels, corresponding to 0%, 25%, 50%, 75%, and 100% of this interval, to simulate a range of user preferences from aggressive risk-avoidance to prioritizing shorter paths.

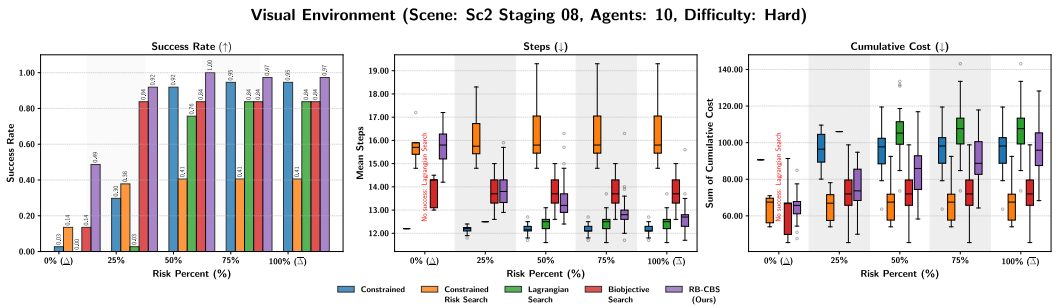

Figure 4: Quantitative results on the visual environment (Scene: SC2 Staging 08) with 10 agents and hard difficulty, as a function of Δ. **Left:** Success rate. **Center:** Average steps showing RB-CBS's path length decrease as Δ grows. **Right:** Total cumulative cost illustrating that RB-CBS respects strict Δ values. Overall, RB-CBS sustains high success and smoothly trades off cost vs. efficiency across all budgets. See Appendix A.2 for more results on different settings.

Evaluation Protocol: We conduct experiments with both 5 and 10 agents on 50 unique problem instances with varying difficulty ranging from easy, medium to hard. Each trial is subject to a time-out of $60 \times N$ seconds, where N is the number of agents. All experiments were conducted using a fixed set of random seeds for problem generation and policy evaluation to ensure reproducibility. The results presented in our main analysis utilize a uniform initial risk allocation. We include a ablation study in the Appendix that compares the performance of this approach against other alternative strategies. Finally, to demonstrate practical applicability, we integrated RB-CBS into the ROS2 framework [29] and successfully deployed it to command multiple Crazyflie drones in a Gazebo simulation, qualitatively validating its ability to generate feasible, collision-free trajectories for real-world robotic systems. Example demonstrations can be found in the Appendix.

Baselines: We evaluate RB-CBS against four alternative strategies that also incorporate learned cost estimates. The first, **Constrained Policy**, is a pure safe GCRL policy trained via Lagrangian actor–critic. The other three are search-based methods. **Constrained Risk Search**, follows the work of Feng et al. by planning over a pre-filtered *safe* graph. In contrast, **Lagrangian Search** performs a single-objective search on the full graph using a cost-distance scalarization with the trained Lagrange multiplier, while **Bi-Objective Search** approximates the full (distance, cost) Pareto front before selecting the shortest path that meets the risk bound Δ [30, 31].

Q1: Adaptability to User-Specified Risk: The qualitative trajectory comparisons in Figures 1, 2, and 3 clearly demonstrate that RB-CBS successfully adapts its behavior to respect user-specified risk bounds. In all three scenarios, the behavior of RB-CBS changes predictably with the risk budget. When the budget is at its tightest (0%), the generated paths are deliberately conservative, maintaining a large clearance from high cost regions at the expense of longer travel times. Conversely, when the budget is relaxed to 25%, the planner produces significantly more direct, efficient routes that cut closer to the hazard, reducing the step count. This visually confirms that RB-CBS is not just finding a single *safe* path, but is actively navigating the trade-off between safety and efficiency as directed by the user. This adaptability stands in stark contrast to baselines like Constrained Policy, which often takes a risky, direct path, or Constrained Search and Bi-Objective Search, which find one safe path but lack the tunable flexibility of our approach.

This qualitative behavior provides the intuition for the aggregate statistical trends seen in our quantitative results. As shown in Figure 4, the visual patterns are confirmed by the data. For example, we observe that RB-CBS (purple) shows a clear statistical trend where at low Δ's, average steps are high while total cumulative cost is low. As Δ increases, the trend inverts. This numerical evidence perfectly matches the behavior seen in the trajectory plots, fulfilling the core requirement of **Q1**.

Q2: Goal Success vs. Safety: RB-CBS maintains high success rates while balancing safety and cost, directly addressing **Q2**. The success rate is most clearly quantified in Table 1. We observe

that RB-CBS achieves success rates that are highly competitive, matching or exceeding the best-performing baselines in most cases. However, a notable drop in success rate to 16% and 49% on the point and visual navigation experiment respectively occurs at the tightest risk bound ($\Delta = \underline{\Delta}$). This is an expected outcome as enforcing strict, low-risk paths while optimizing for path-lengths in a crowded environment causes traversable regions to become narrow making it more challenging to find collision-free paths increasing the likelihood of timeouts.

Q3: Scalability to More Agents: The performance and adaptive behavior of RB-CBS remain robust when scaling from 5 to 10 agents, affirming **Q3**. This is best observed by comparing the top and bottom halves of Table 1. For both 5 and 10 agents, RB-CBS exhibits the same characteristic trend where as Δ increases, the average steps decrease while the total cumulative cost is allowed to rise. For instance, with 5 agents, average steps decrease from 15.32 to 12.68 as Δ goes from 0% to 100%. This same pattern holds for 10 agents, where steps decrease from 15.79 to 12.70. While total accumulated costs naturally increase with more agents due to a more complex multi-agent problem, the consistent and predictable adaptive behavior demonstrates scalability of the approach.

Problem Configurations		**Methods (Cumulative Cost (Success Rate) / Average Steps)**				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	45.32 ± 4.65 (4%) 12.20 ± 0.00	29.99 ± 4.40 (48%) 15.92 ± 0.85	55.20 ± 0.00 (2%) 12.60 ± 0.00	32.30 ± 6.84 (21%) 13.86 ± 0.56	32.06 ± 6.49 (71%) 15.32 ± 0.87
	25%	46.19 ± 6.18 (29%) 12.26 ± 0.19	31.21 ± 5.19 (77%) 15.89 ± 0.98	46.40 ± 5.57 (10%) 12.52 ± 0.10	36.44 ± 7.90 (90%) 13.49 ± 0.78	37.21 ± 6.86 (98%) 13.84 ± 0.81
	50%	48.08 ± 6.48 (90%) 12.18 ± 0.30	31.21 ± 5.19 (77%) 15.89 ± 0.98	52.80 ± 9.27 (77%) 12.48 ± 0.36	37.46 ± 8.24 (100%) 13.45 ± 0.75	41.13 ± 8.32 (100%) 13.22 ± 0.64
	75%	48.62 ± 6.63 (96%) 12.19 ± 0.30	31.73 ± 6.01 (79%) 16.04 ± 1.33	53.57 ± 9.71 (94%) 12.43 ± 0.41	37.46 ± 8.24 (100%) 13.45 ± 0.75	44.59 ± 9.58 (100%) 12.86 ± 0.55
	100%	48.58 ± 6.51 (100%) 12.19 ± 0.29	31.73 ± 6.01 (79%) 16.04 ± 1.33	53.57 ± 9.71 (94%) 12.43 ± 0.41	37.46 ± 8.24 (100%) 13.45 ± 0.75	48.47 ± 9.89 (100%) 12.68 ± 0.58
10	0%	90.65 ± 0.00 (3%) 12.20 ± 0.00	63.68 ± 7.12 (14%) 15.80 ± 0.79	N/A	64.06 ± 16.18 (14%) 13.84 ± 0.65	64.78 ± 8.71 (49%) 15.79 ± 0.77
	25%	96.22 ± 9.49 (30%) 12.15 ± 0.17	65.46 ± 8.09 (38%) 16.09 ± 1.02	106.05 ± 0.00 (3%) 12.50 ± 0.00	72.44 ± 12.29 (84%) 13.72 ± 0.61	75.57 ± 11.89 (92%) 13.93 ± 0.73
	50%	95.94 ± 11.69 (92%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	105.49 ± 13.56 (76%) 12.36 ± 0.33	72.44 ± 12.29 (84%) 13.72 ± 0.61	85.44 ± 13.65 (100%) 13.42 ± 0.87
	75%	96.13 ± 11.58 (95%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	107.69 ± 15.01 (84%) 12.41 ± 0.39	72.44 ± 12.29 (84%) 13.72 ± 0.61	90.52 ± 13.15 (97%) 12.91 ± 0.74
	100%	96.13 ± 11.58 (95%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	107.69 ± 15.01 (84%) 12.41 ± 0.39	72.44 ± 12.29 (84%) 13.72 ± 0.61	96.08 ± 15.02 (97%) 12.70 ± 0.66

Table 1: **Safe Visual Navigation**: Comparison of total cost, average steps and success rate across different approaches and agent count on the hard difficulty inside SC2 Staging 08 Scene. Lower cumulative costs, lower steps and higher success rates are better. See Appendix A.2 for more results.

5 Conclusion

We address the conservative nature of prior methods which rely on static graph pruning with, RB-CBS, a novel framework for dynamic risk allocation within CBS. Our approach moves beyond simple edge pruning approach by accepting a global, user-specified risk bound (Δ) and intelligently distributing it as local budgets (δ) to individual agents. This allows RB-CBS to flexibly trade safety for efficiency, enabling navigation through potentially hazardous areas when necessary while ensuring the overall mission risk respects user-specified Δ.

Our comprehensive experiments in different navigation tasks demonstrate that RB-CBS provides a flexible and effective solution for risk-bounded multi-agent navigation. By dynamically adjusting risk allocated to each agent, RB-CBS smoothly adapts its behavior from prioritizing safety to focusing on path efficiency directly in response to the user's input. Further, it maintains high success rates and scales effectively where static pruning approaches fail. This demonstrates its practicality for real-world deployment under full observability.

References

[1] Z. Wang and S. Zlatanova. Multi-agent based path planning for first responders among moving obstacles. *Computers, Environment and Urban Systems*, 56:48–58, 2016. ISSN 0198-9715. doi:https://doi.org/10.1016/j.compenvurbsys.2015.11.001. URL https://www.sciencedirect.com/science/article/pii/S0198971515300314.

[2] J. Im and B.-Y. Lee. Multi-agent inspection path planning with large-scale vehicle routing problem. *Journal of Aerospace Information Systems*, 20:1–9, 03 2023. doi:10.2514/1.I011202.

[3] F. Peralta, M. Pearce, M. Poloczek, D. G. Reina, S. Toral, and J. Branke. Multi-objective path planning for environmental monitoring using an autonomous surface vehicle. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '22, page 747–750, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392686. doi:10.1145/3520304.3528978. URL https://doi.org/10.1145/3520304.3528978.

[4] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar, et al. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, volume 10, pages 151–158, 2019.

[5] R. Luna and K. E. Bekris. Push and swap: fast cooperative path-finding with completeness guarantees. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One*, IJCAI'11, page 294–300. AAAI Press, 2011. ISBN 9781577355137.

[6] D. Silver. Cooperative pathfinding. In *Artificial Intelligence and Interactive Digital Entertainment Conference*, 2005. URL https://api.semanticscholar.org/CorpusID:17714238.

[7] P. Surynek, A. Felner, R. Stern, and E. Boyarski. Modifying optimal sat-based approach to multi-agent path-finding problem to suboptimal variants, 2017. URL https://arxiv.org/abs/1707.00228.

[8] K.-H. Wang and A. Botea. Fast and memory-efficient multi-agent pathfinding. pages 380–387, 01 2008.

[9] J. Yu and S. M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, page 1443–1449. AAAI Press, 2013.

[10] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015.

[11] M. Barer, G. Sharon, R. Stern, and A. Felner. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. *Proceedings of the International Symposium on Combinatorial Search*, 5:19–27, 09 2021. doi:10.1609/socs.v5i1.18315.

[12] E. Boyarski, A. Felner, R. Stern, G. Sharon, D. Tolpin, O. Betzalel, and E. Shimony. Icbs: improved conflict-based search algorithm for multi-agent pathfinding. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 740–746. AAAI Press, 2015. ISBN 9781577357384.

[13] L. Cohen, T. Uras, T. Kumar, and S. Koenig. Optimal and bounded-suboptimal multi-agent motion planning. *Proceedings of the International Symposium on Combinatorial Search*, 10: 44–51, 09 2021. doi:10.1609/socs.v10i1.18501.

[14] H. Ma, J. Li, T. K. S. Kumar, and S. Koenig. Lifelong multi-agent path finding for online pickup and delivery tasks, 2017. URL https://arxiv.org/abs/1705.10868.

[15] J. Olkin, V. Parimi, and B. Williams. Multi-agent vulcan: An information-driven multi-agent path finding approach. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10253–10259, 2024. doi:10.1109/IROS58592.2024.10801571.

[16] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell. Learning to navigate in complex environments, 2017.

[17] T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1312–1320, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

[18] V. Pong, S. Gu, M. Dalal, and S. Levine. Temporal difference models: Model-free deep rl for model-based control, 2020. URL https://arxiv.org/abs/1802.09081.

[19] A. Levy, R. Platt, and K. Saenko. Hierarchical reinforcement learning with hindsight, 2019.

[20] O. Nachum, S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning, 2018.

[21] E. Altman. *Constrained Markov decision processes*. Routledge, 2021.

[22] B. Eysenbach, R. R. Salakhutdinov, and S. Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[23] M. Feng, V. Parimi, and B. Williams. Safe multi-agent navigation guided by goal-conditioned safe reinforcement learning, 2025. URL https://arxiv.org/abs/2502.17813.

[24] J. Li, D. Harabor, P. J. Stuckey, H. Ma, and S. Koenig. Disjoint splitting for multi-agent path finding with conflict-based search. In *ICAPS*, pages 279–283. AAAI Press, 2019.

[25] Y. Shaoul, I. Mishani, M. Likhachev, and J. Li. Accelerating search-based planning for multi-robot manipulation by leveraging online-generated experiences. *Proceedings of the International Conference on Automated Planning and Scheduling*, 34(1):523–531, May 2024. doi:10.1609/icaps.v34i1.31513. URL https://ojs.aaai.org/index.php/ICAPS/article/view/31513.

[26] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[27] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[28] X. Puig, E. Undersander, A. Szot, M. D. Cote, R. Partsey, J. Yang, R. Desai, A. W. Clegg, M. Hlavac, T. Min, T. Gervet, V. Vondrus, V.-P. Berges, J. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023.

[29] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66):eabm6074, 2022. doi:10.1126/scirobotics.abm6074. URL https://www.science.org/doi/abs/10.1126/scirobotics.abm6074.

[30] Z. Ren, S. Rathinam, and H. Choset. Multi-objective conflict-based search for multi-agent path finding. *CoRR*, abs/2101.03805, 2021. URL https://arxiv.org/abs/2101.03805.

[31] C. Hernández Ulloa, W. Yeoh, J. A. Baier, H. Zhang, L. Suazo, and S. Koenig. A simple and fast bi-objective search algorithm. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30(1):143–151, Jun. 2020. doi:10.1609/icaps.v30i1.6655. URL https://ojs.aaai.org/index.php/ICAPS/article/view/6655.

A Appendix

A.1 Collision Checking with Learned Critics

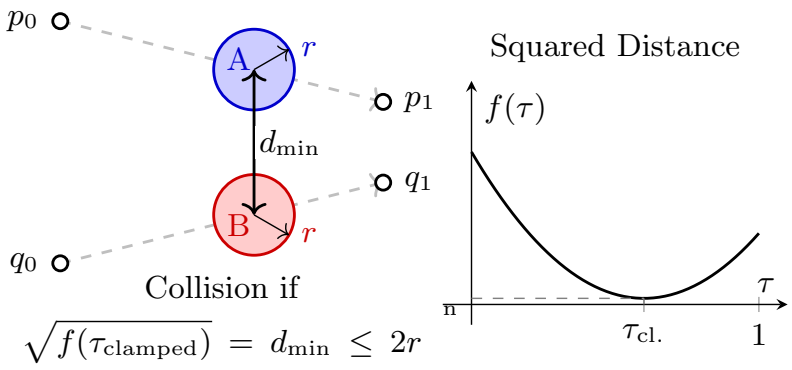

Figure 5: Continuous-time collision checking between two agents on spatially intersecting edges. We geometrically determine the minimum separation distance, $d_{\min}$, between the agents, each with radius r using the learning distance critic. This corresponds to minimizing the squared distance function, $f(\tau)$, over the time interval $\tau \in [0, 1]$. A collision is detected if $\sqrt{f(\tau_{\text{clamped}})} = d_{\min} \leq 2r$.

To detect the collisions illustrated in Figure 5, we model each agent as a disc of fixed radius r. When two agents traverse distinct graph edges, (p_0, p_1) and (q_0, q_1), we define their positions over normalized time $\tau \in [0, 1]$ as linear interpolations:

$$P(\tau) = p_0 + \tau(p_1 - p_0), \qquad Q(\tau) = q_0 + \tau(q_1 - q_0).$$

The squared distance between the agents' centers is a quadratic function of time, $f(\tau) = \|P(\tau) - Q(\tau)\|^2$. By defining the initial relative offset $\mathbf{D} = p_0 - q_0$ and the relative velocity $\mathbf{V} = (p_1 - p_0) - (q_1 - q_0)$, we can write this function in its standard form:

$$f(\tau) = \|\mathbf{D} + \tau\mathbf{V}\|^2 = a\tau^2 + b\tau + c, \tag{2}$$

where the coefficients are $a = \mathbf{V} \cdot \mathbf{V}$, $b = 2\mathbf{D} \cdot \mathbf{V}$, and $c = \mathbf{D} \cdot \mathbf{D}$.

Coefficients from Learned Distances. Our objective is to compute these coefficients using only distance queries, which can be provided by our learned distance critic. The constant term, c, is trivially the initial squared distance: $c = \|p_0 - q_0\|^2 = \text{dist}(p_0, q_0)^2$.

To express a and b, we use a vector identity derived from the law of cosines to replace dot products with squared distances. The key identity for two vectors $\mathbf{u}$ and $\mathbf{v}$ is:

$$\begin{aligned} 2(\mathbf{u} \cdot \mathbf{v}) = \text{dist}(p_i, q_i)^2 + \text{dist}(p_j, q_j)^2 \\ - \text{dist}(p_i, q_j)^2 - \text{dist}(p_j, q_i)^2 \end{aligned}$$

where $\mathbf{u} = p_i - p_j$ and $\mathbf{v} = q_i - q_j$. By systematically applying this identity to the dot products within the expressions for $a = \mathbf{V} \cdot \mathbf{V}$ and $b = 2\mathbf{D} \cdot \mathbf{V}$, we arrive at their final forms:

$$\begin{aligned} a = \text{dist}(p_0, p_1)^2 + \text{dist}(q_0, q_1)^2 + \text{dist}(p_0, q_0)^2 \\ + \text{dist}(p_1, q_1)^2 - \text{dist}(p_0, q_1)^2 - \text{dist}(p_1, q_0)^2 \end{aligned} \tag{3}$$

$$\begin{aligned} b = \text{dist}(p_1, q_0)^2 - \text{dist}(p_0, p_1)^2 - 2\,\text{dist}(p_0, q_0)^2 \\ - \text{dist}(q_0, q_1)^2 + \text{dist}(p_0, q_1)^2 \end{aligned} \tag{4}$$

Minimizer and Collision Test. With the coefficients computed, we find the time of closest approach. For $a > 0$, the unconstrained minimizer of the quadratic is $\tau^* = -b/(2a)$. We clamp this value to the valid time interval to get $\tau_{\text{clamped}} = \min(1, \max(0, \tau^*))$. A collision is detected if the minimum distance is less than or equal to the sum of the radii:

$$\sqrt{f(\tau_{\text{clamped}})} \leq 2r. \tag{5}$$

Crucially, this entire continuous-time collision check is performed using only the outputs of the learned distance critic, seamlessly integrating it into our planning framework with unstructured graphs.

A.2 More Results

A.2.1 Ablation Study

Figure 6 presents an ablation study on the initial risk allocation strategies for RB-CBS. The results are shown for the 2D Point environment and challenging, high-fidelity visual navigation environments on three distinct scenes. The findings are remarkably consistent across all environments. The Constant allocation strategy (blue) involves an initial allocation of risk budgets that is not dynamically adjusted by the planner. This approach consistently proves to be brittle. It achieves very low or zero success rates at tight risk budgets ($\Delta \leq 25\%$) because its rigid, pre-set allocation cannot adapt when one or more agents require a larger share of the risk budget to find a valid path.

In contrast, the Uniform (orange), Utility-based (green) and Inverse Utility-based (red) strategies demonstrate the power of dynamic risk allocation. They maintain high success rates across nearly all risk levels and environments. Crucially, they exhibit the desired adaptive behavior where at $\Delta = \underline{\Delta}$, they accept longer paths (higher steps) to ensure safety (lower cost), and at $\Delta = \overline{\Delta}$, they prioritize shorter paths. The similar performance between these three strategies, even in complex visual scenes, suggests that the simplicity of uniform allocation is highly effective. This comprehensive result validates our choice to use the Uniform strategy in the main paper's analysis, as it provides the benefits of dynamic allocation without the overhead of a the utility computation.

A.2.2 Point Environment

We now present the complete experimental results for the simpler 2D Point Navigation environment, detailing performance across Easy, Medium, and Hard difficulties with both 5 and 10 agents. Figures 7 and 8 provides a visual summary of these results, while Tables 2 and 3 offer the precise numerical data.

Across all difficulties and agent counts in the 2D Point Environment, our method, RB-CBS, consistently demonstrates superior flexibility and performance. As shown in Figures 7 and 8 , RB-CBS (purple) exhibits the most desirable adaptive behavior. At tight risk budgets ($\Delta = \underline{\Delta}$), it prioritizes safety, achieving low cumulative costs, often at the expense of taking more steps than balanced methods like Bi-Objective Search. As the risk budget is relaxed to $\Delta = \overline{\Delta}$, RB-CBS effectively shifts its priority, reducing the number of steps taken to be highly competitive with the most path-efficient methods while maintaining a very high success rate.

The numerical data in Tables 2 and 3 corroborates these findings. For instance, in the Hard 10 agent scenario (Table 3), our method improves its step count from 17.39 to 9.75 as Δ increases, a clear demonstration of its ability to leverage the available risk budget. Furthermore, its success rate remains robust (e.g., 84-96% for $\Delta \geq 25\%$) where other methods like Lagrangian Search can be brittle. This confirms that RB-CBS provides a reliable and adaptable solution for multi-agent navigation, effectively balancing user-specified risk with path efficiency.

A.2.3 Habitat Environments

The complete results for the visual navigation tasks, presented across three distinct scenes (SC2 Staging 08, SC0 Staging 20, and SC3 Staging 11) and three difficulty levels, reinforce the core

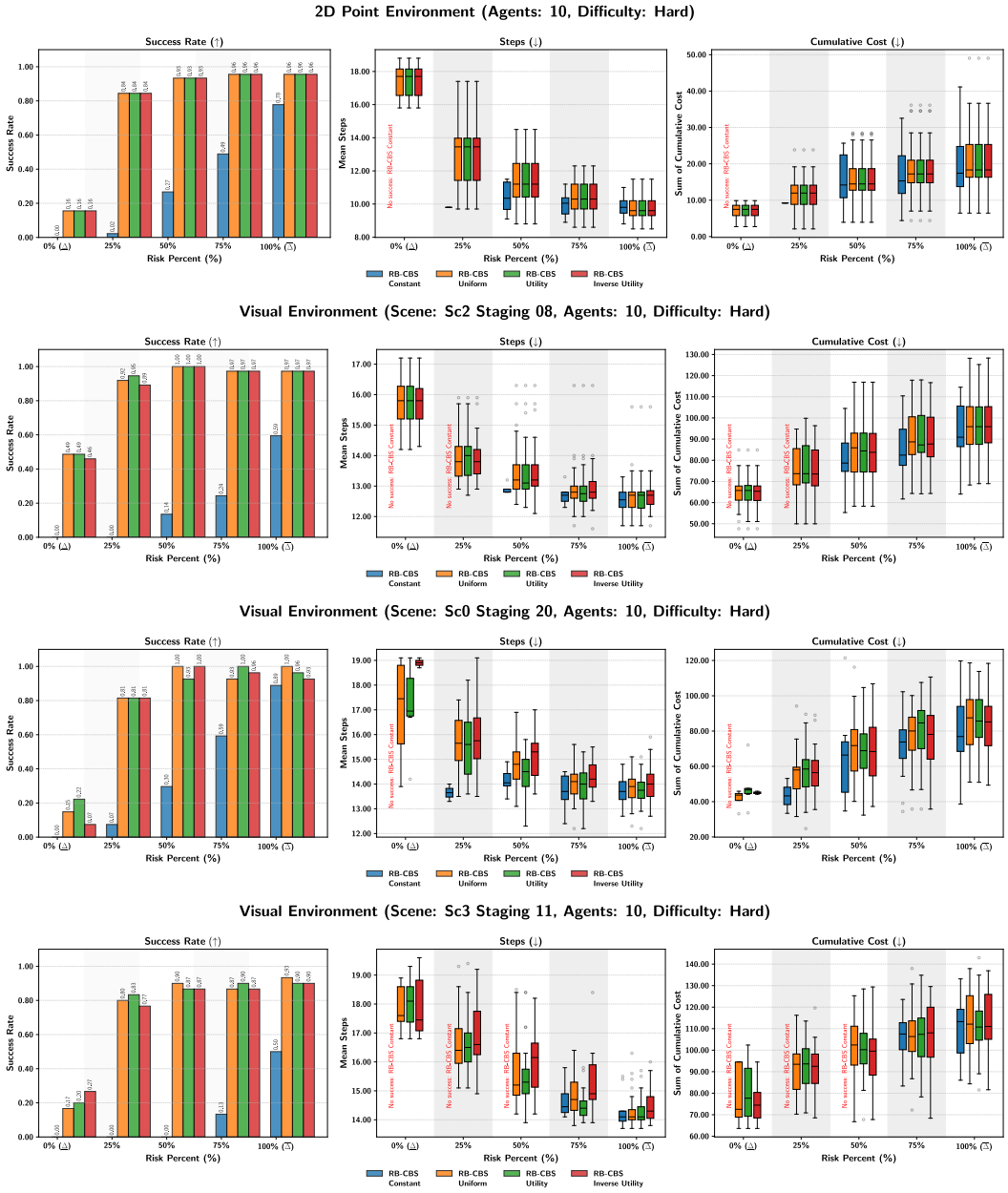

Figure 6: **Ablation Study on Initial Risk Allocation Strategies.** Comparison of *Constant*, *Uniform*, *Utility-based*, and *Inverse Utility-based* risk allocation for RB-CBS across four distinct environments (10 Agents, Hard Difficulty). The results consistently demonstrate that dynamic allocation strategies (Uniform, Utility and Inverse Utility) far outperform the rigid Constant approach, especially at tight risk budgets ($\Delta = \underline{\Delta}$). Higher success rates, lower steps and lower costs are better.

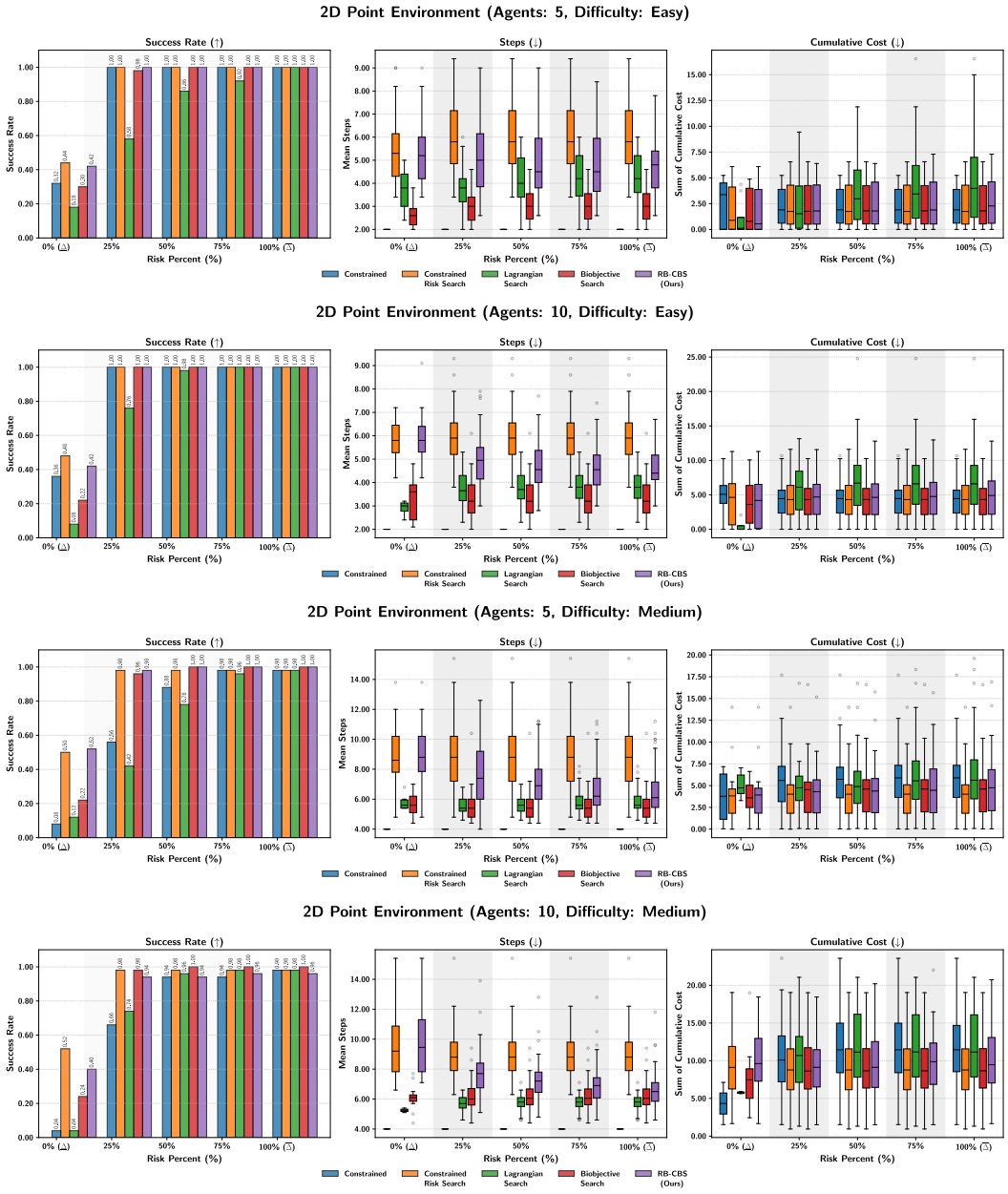

Figure 7: Quantitative results for the 2D Point environment comparing all methods across Easy and Medium difficulties for 5 and 10 agents. These plots visually demonstrate the trade-offs between success rate, steps, and cumulative cost as the risk budget (Δ) is varied. Our method, RB-CBS (purple), consistently shows a strong adaptive profile.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	2.55 ± 2.11 (32%) 2.00 ± 0.00	2.01 ± 2.22 (44%) 5.59 ± 1.62	1.14 ± 1.62 (18%) 3.73 ± 0.84	1.65 ± 1.97 (30%) 2.63 ± 0.52	1.90 ± 2.22 (42%) 5.42 ± 1.48
	25%	2.28 ± 1.71 (100%) 2.00 ± 0.00	2.37 ± 2.06 (100%) 5.97 ± 1.57	2.40 ± 2.52 (58%) 3.81 ± 0.91	2.34 ± 2.04 (98%) 3.00 ± 0.65	2.40 ± 2.08 (100%) 5.14 ± 1.53
	50%	2.28 ± 1.71 (100%) 2.00 ± 0.00	2.37 ± 2.06 (100%) 5.97 ± 1.57	3.68 ± 3.19 (86%) 4.16 ± 1.04	2.37 ± 2.03 (100%) 3.02 ± 0.66	2.46 ± 2.10 (100%) 4.85 ± 1.54
	75%	2.28 ± 1.71 (100%) 2.00 ± 0.00	2.37 ± 2.06 (100%) 5.97 ± 1.57	4.08 ± 3.65 (92%) 4.23 ± 1.04	2.37 ± 2.03 (100%) 3.02 ± 0.66	2.55 ± 2.16 (100%) 4.87 ± 1.43
	100%	2.28 ± 1.71 (100%) 2.00 ± 0.00	2.37 ± 2.06 (100%) 5.97 ± 1.57	4.62 ± 4.16 (100%) 4.29 ± 1.05	2.37 ± 2.03 (100%) 3.02 ± 0.66	2.64 ± 2.13 (100%) 4.72 ± 1.19
10	0%	4.99 ± 2.98 (36%) 2.00 ± 0.00	4.19 ± 3.28 (48%) 5.78 ± 0.85	0.52 ± 0.90 (8%) 2.90 ± 0.31	4.04 ± 3.58 (22%) 3.28 ± 0.88	4.05 ± 3.52 (42%) 5.84 ± 1.07
	25%	4.27 ± 2.82 (100%) 2.00 ± 0.00	4.34 ± 3.06 (100%) 5.88 ± 1.14	5.80 ± 3.55 (76%) 3.73 ± 0.68	4.42 ± 3.01 (100%) 3.31 ± 0.82	4.51 ± 3.16 (100%) 5.00 ± 1.13
	50%	4.27 ± 2.82 (100%) 2.00 ± 0.00	4.34 ± 3.06 (100%) 5.88 ± 1.14	7.10 ± 4.87 (98%) 3.80 ± 0.67	4.42 ± 3.01 (100%) 3.31 ± 0.82	4.61 ± 3.29 (100%) 4.71 ± 1.03
	75%	4.27 ± 2.82 (100%) 2.00 ± 0.00	4.34 ± 3.06 (100%) 5.88 ± 1.14	7.05 ± 4.83 (100%) 3.82 ± 0.68	4.42 ± 3.01 (100%) 3.31 ± 0.82	4.67 ± 3.28 (100%) 4.58 ± 0.91
	100%	4.27 ± 2.82 (100%) 2.00 ± 0.00	4.34 ± 3.06 (100%) 5.88 ± 1.14	7.05 ± 4.83 (100%) 3.82 ± 0.68	4.42 ± 3.01 (100%) 3.31 ± 0.82	4.86 ± 3.41 (100%) 4.64 ± 0.88

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	3.69 ± 3.00 (8%) 4.00 ± 0.00	3.75 ± 2.94 (50%) 8.90 ± 2.09	5.06 ± 1.37 (12%) 5.80 ± 0.50	3.55 ± 2.12 (22%) 5.64 ± 0.77	3.80 ± 2.90 (52%) 8.92 ± 2.06
	25%	5.56 ± 3.97 (56%) 4.00 ± 0.00	4.14 ± 2.92 (98%) 9.02 ± 2.09	4.89 ± 3.35 (42%) 5.51 ± 0.64	4.18 ± 2.99 (96%) 5.53 ± 0.96	4.19 ± 2.87 (98%) 7.68 ± 1.97
	50%	5.55 ± 3.42 (88%) 4.00 ± 0.00	4.14 ± 2.92 (98%) 9.02 ± 2.09	5.24 ± 3.35 (78%) 5.64 ± 0.66	4.35 ± 3.07 (100%) 5.57 ± 0.97	4.50 ± 3.15 (100%) 7.19 ± 1.67
	75%	5.72 ± 3.41 (98%) 4.00 ± 0.00	4.14 ± 2.92 (98%) 9.02 ± 2.09	6.09 ± 3.80 (96%) 5.81 ± 0.83	4.35 ± 3.07 (100%) 5.57 ± 0.97	4.68 ± 3.24 (100%) 6.83 ± 1.71
	100%	5.72 ± 3.41 (98%) 4.00 ± 0.00	4.14 ± 2.92 (98%) 9.02 ± 2.09	6.36 ± 4.22 (98%) 5.81 ± 0.83	4.35 ± 3.07 (100%) 5.57 ± 0.97	4.91 ± 3.37 (100%) 6.61 ± 1.63
10	0%	4.32 ± 2.80 (4%) 4.00 ± 0.00	9.33 ± 4.20 (52%) 9.48 ± 2.01	5.77 ± 0.14 (4%) 5.25 ± 0.15	7.63 ± 4.24 (24%) 6.09 ± 0.85	9.87 ± 4.52 (40%) 9.80 ± 2.28
	25%	10.48 ± 4.86 (66%) 4.00 ± 0.00	8.76 ± 3.99 (98%) 9.02 ± 1.71	10.43 ± 4.89 (74%) 5.68 ± 0.52	9.01 ± 4.27 (98%) 6.21 ± 0.98	9.00 ± 4.00 (94%) 7.84 ± 1.62
	50%	11.82 ± 5.12 (94%) 4.00 ± 0.00	8.76 ± 3.99 (98%) 9.02 ± 1.71	11.54 ± 5.10 (96%) 5.77 ± 0.53	9.04 ± 4.23 (100%) 6.22 ± 0.98	9.36 ± 4.24 (94%) 7.30 ± 1.41
	75%	11.82 ± 5.12 (94%) 4.00 ± 0.00	8.76 ± 3.99 (98%) 9.02 ± 1.71	11.56 ± 5.05 (98%) 5.79 ± 0.54	9.04 ± 4.23 (100%) 6.22 ± 0.98	9.84 ± 4.36 (96%) 6.99 ± 1.44
	100%	11.83 ± 5.03 (98%) 4.00 ± 0.00	8.76 ± 3.99 (98%) 9.02 ± 1.71	11.56 ± 5.05 (98%) 5.79 ± 0.54	9.04 ± 4.23 (100%) 6.22 ± 0.98	10.09 ± 4.54 (96%) 6.68 ± 1.27

Table 2: **2D Navigation - Easy(above) and Medium(below) Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

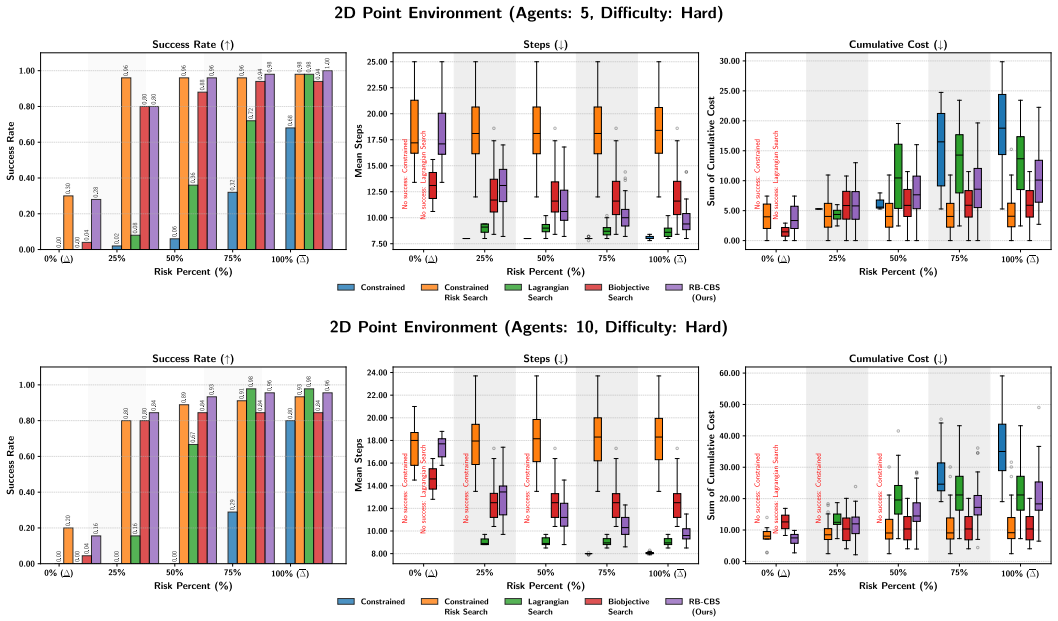

Figure 8: Quantitative results for the 2D Point environment comparing all methods across Hard difficulty for 5 and 10 agents. These plots again visually demonstrate the trade-offs between success rate, steps, and cumulative cost as the risk budget (Δ) is varied. Our method, RB-CBS (purple), consistently shows a strong adaptive profile.

Problem Configurations		**Methods (Cumulative Cost (Success Rate) / Average Steps)**				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	N/A	4.01 ± 2.32 (30%) 18.83 ± 3.70	N/A	1.46 ± 1.46 (4%) 13.10 ± 2.50	3.80 ± 2.28 (28%) 18.47 ± 3.56
	25%	5.28 ± 0.00 (2%) 8.00 ± 0.00	4.44 ± 2.64 (96%) 18.47 ± 3.42	4.29 ± 1.29 (8%) 8.90 ± 0.57	5.74 ± 2.89 (80%) 12.38 ± 2.48	6.00 ± 3.14 (80%) 13.05 ± 2.14
	50%	6.28 ± 1.19 (6%) 8.00 ± 0.00	4.44 ± 2.64 (96%) 18.47 ± 3.42	10.82 ± 5.75 (36%) 8.98 ± 0.62	6.01 ± 2.97 (88%) 12.24 ± 2.42	7.68 ± 3.67 (96%) 11.25 ± 1.96
	75%	15.22 ± 6.60 (32%) 8.03 ± 0.10	4.44 ± 2.64 (96%) 18.47 ± 3.42	13.07 ± 5.95 (72%) 8.79 ± 0.56	6.04 ± 2.93 (94%) 12.12 ± 2.48	9.11 ± 4.46 (98%) 10.25 ± 1.44
	100%	18.35 ± 7.31 (68%) 8.05 ± 0.14	4.66 ± 3.03 (98%) 18.50 ± 3.39	13.19 ± 5.76 (98%) 8.73 ± 0.55	6.04 ± 2.93 (94%) 12.12 ± 2.48	10.40 ± 4.79 (100%) 9.72 ± 1.31
10	0%	N/A	7.85 ± 3.35 (20%) 17.61 ± 1.96	N/A	12.58 ± 4.32 (4%) 14.60 ± 1.80	6.97 ± 2.35 (16%) 17.39 ± 1.05
	25%	N/A	9.03 ± 3.93 (80%) 17.88 ± 2.42	13.17 ± 3.33 (16%) 9.09 ± 0.33	10.88 ± 4.64 (80%) 12.54 ± 1.53	11.87 ± 4.37 (84%) 13.07 ± 1.88
	50%	N/A	10.30 ± 5.58 (89%) 18.19 ± 2.54	20.15 ± 7.46 (67%) 9.07 ± 0.37	11.00 ± 4.74 (84%) 12.50 ± 1.52	15.70 ± 6.06 (93%) 11.35 ± 1.45
	75%	28.64 ± 8.04 (29%) 7.98 ± 0.05	10.71 ± 6.09 (91%) 18.26 ± 2.56	22.74 ± 8.97 (98%) 9.05 ± 0.35	11.00 ± 4.74 (84%) 12.50 ± 1.52	18.07 ± 7.19 (96%) 10.40 ± 0.99
	100%	36.40 ± 10.67 (80%) 8.03 ± 0.09	11.21 ± 6.81 (93%) 18.29 ± 2.53	22.74 ± 8.97 (98%) 9.05 ± 0.35	11.00 ± 4.74 (84%) 12.50 ± 1.52	20.70 ± 8.77 (96%) 9.75 ± 0.72

Table 3: **2D Navigation - Hard Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

findings from our main analysis and highlight the robustness of our approach in complex, high-dimensional environments.

Across all scenes and configurations, RB-CBS (purple) consistently demonstrates a superior adaptive profile compared to the baseline methods. As visually summarized in the accompanying figures, our method effectively navigates the trade-off between path efficiency and safety. At the tightest risk budgets ($\Delta = \underline{\Delta}$)), RB-CBS prioritizes finding safe, low-cost paths, often accepting a higher step

count. A notable caveat is that as the problem difficulty and the agent count increases, the success rate of all approaches starts to decline pointing to the fact that tighter spaces causes the planners to work harder to find low-risk collision free paths leading to timeouts. As the risk budget is relaxed towards $\Delta = \overline{\Delta}$, our approach successfully leverages this flexibility to reduce the number of steps, achieving path lengths that are highly competitive with the most efficient baselines while consistently maintaining high success rates.

The numerical data provided in the tables further solidifies these observations. For example, in the challenging 'SC0 Staging 20 - Hard' scene with 10 agents, RB-CBS improves its average step count from 16.98 to 13.81 as the budget increases, while its success rate climbs to 100%. This contrasts sharply with methods like Lagrangian Search, which can be brittle and fail to find solutions in many scenarios (marked as N/A), or Constrained Policy and Bi-Objective Search, which often achieve lower success rates for a given risk level. The consistent performance across varied visual environments each with unique layouts and risk distributions underscores the generalization capability of our framework. It confirms that RB-CBS provides a reliable and adaptable solution for risk-bounded multi-agent visual navigation, effectively balancing user-specified risk preferences with mission success.

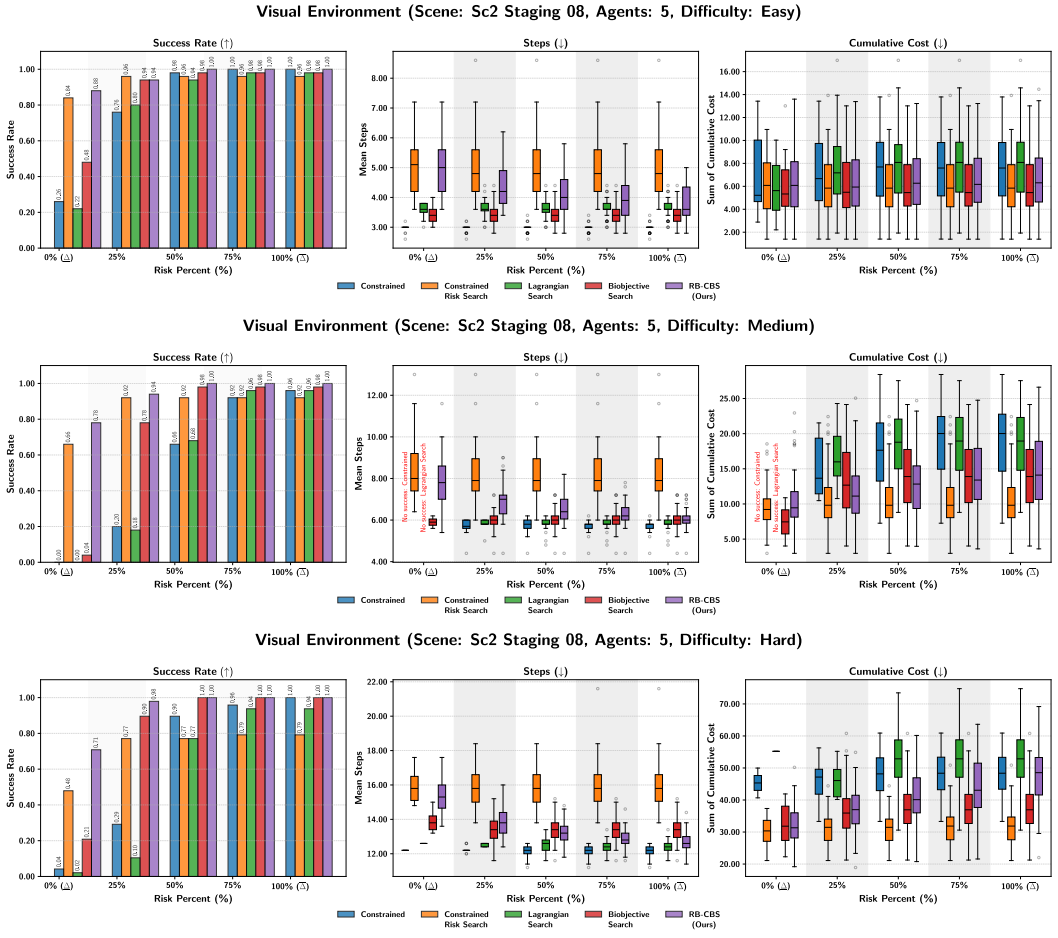

Figure 9: Quantitative results for the visual navigation environment on SC2 Staging 08 scene comparing all methods across Easy, Medium and Hard difficulties for 5 agents.

A.3 Hyperparameters

Table 9 shows the hyperparameters used for training our low-level RL agent. All models were trained, and all planning experiments conducted, on a system equipped with a 32-core Intel i9-

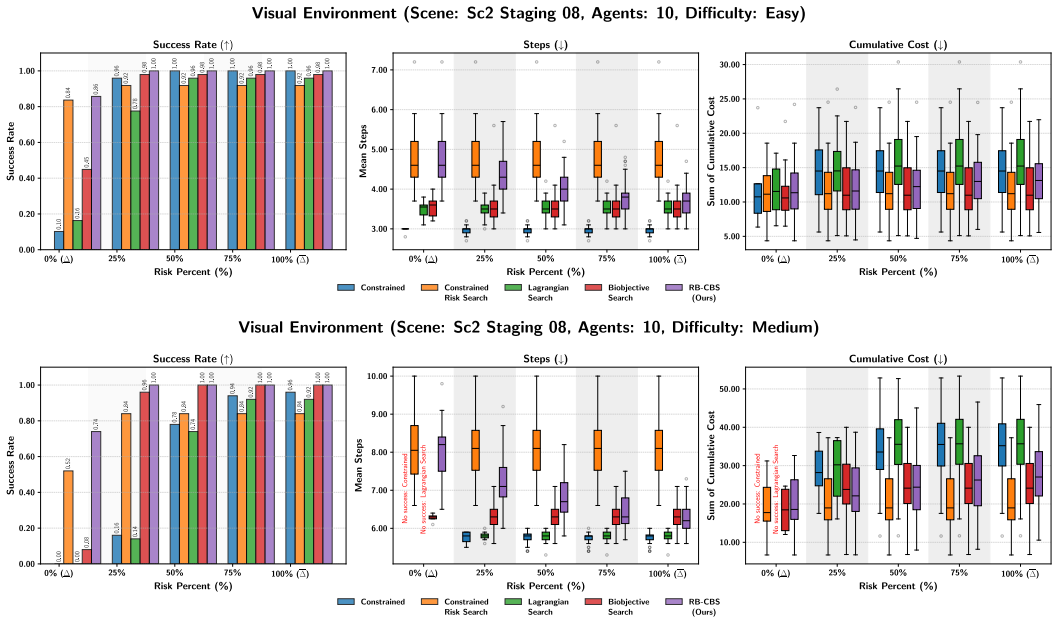

Figure 10: Quantitative results for the visual navigation environment on SC2 Staging 08 scene comparing all methods across Easy and Medium difficulties for 10 agents.

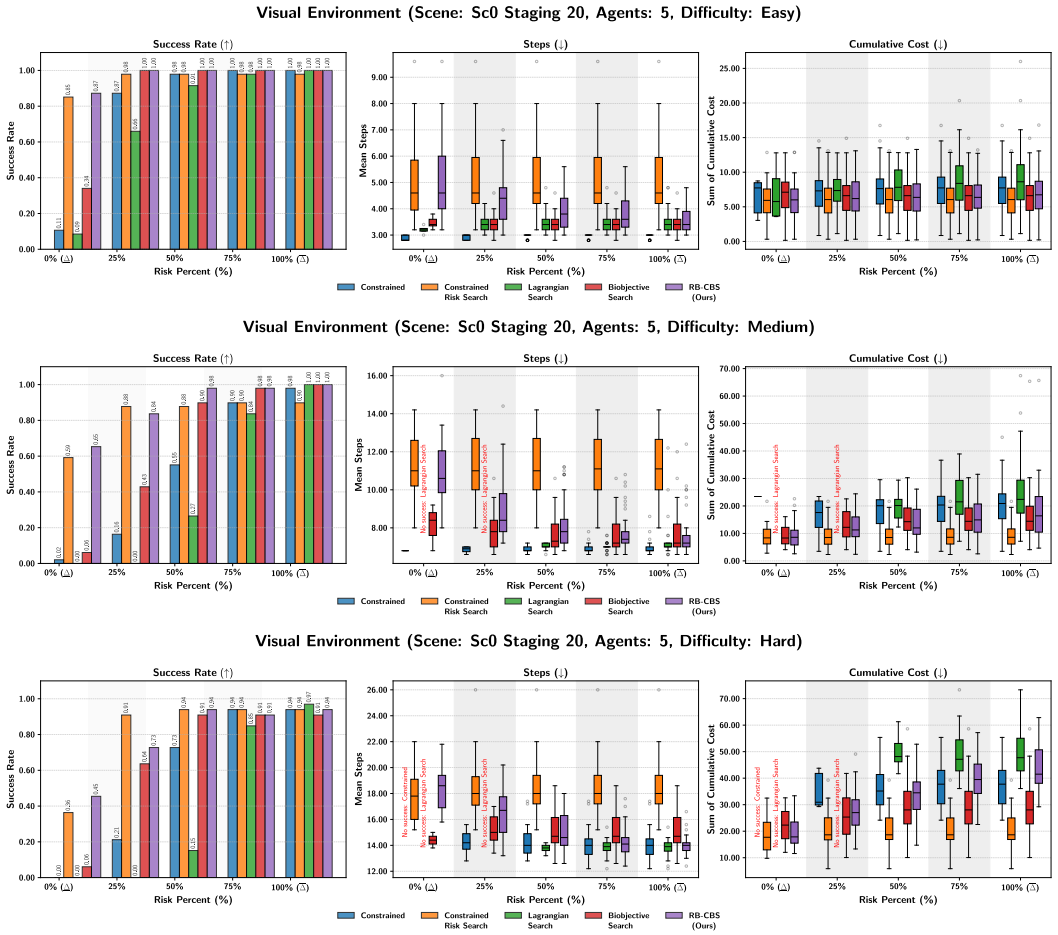

Figure 11: Quantitative results for the visual navigation environment on SC0 Staging 20 scene comparing all methods across Easy, Medium and Hard difficulties for 5 agents.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	6.83 ± 3.18 (26%) 2.95 ± 0.14	6.08 ± 2.51 (84%) 5.03 ± 0.89	5.91 ± 2.49 (22%) 3.56 ± 0.25	5.77 ± 2.55 (48%) 3.41 ± 0.27	6.28 ± 2.76 (88%) 5.01 ± 0.88
	25%	7.16 ± 3.03 (76%) 2.97 ± 0.11	6.12 ± 2.66 (96%) 5.03 ± 1.02	7.61 ± 3.27 (80%) 3.63 ± 0.32	6.19 ± 2.74 (94%) 3.46 ± 0.35	6.40 ± 2.92 (94%) 4.42 ± 0.72
	50%	7.58 ± 3.07 (98%) 2.98 ± 0.13	6.12 ± 2.66 (96%) 5.03 ± 1.02	7.86 ± 3.32 (94%) 3.64 ± 0.33	6.14 ± 2.69 (98%) 3.45 ± 0.35	6.57 ± 2.87 (100%) 4.09 ± 0.69
	75%	7.56 ± 3.04 (100%) 2.98 ± 0.12	6.12 ± 2.66 (96%) 5.03 ± 1.02	8.04 ± 3.38 (98%) 3.64 ± 0.32	6.14 ± 2.69 (98%) 3.45 ± 0.35	6.60 ± 2.87 (100%) 3.96 ± 0.66
	100%	7.56 ± 3.04 (100%) 2.98 ± 0.12	6.12 ± 2.66 (96%) 5.03 ± 1.02	8.04 ± 3.38 (98%) 3.64 ± 0.32	6.14 ± 2.69 (98%) 3.45 ± 0.35	6.83 ± 2.98 (100%) 3.84 ± 0.64
10	0%	12.37 ± 6.06 (10%) 2.96 ± 0.08	11.37 ± 3.40 (84%) 4.80 ± 0.70	11.74 ± 3.55 (16%) 3.47 ± 0.22	10.99 ± 3.61 (45%) 3.52 ± 0.22	11.90 ± 3.95 (86%) 4.77 ± 0.69
	25%	14.39 ± 4.16 (96%) 2.95 ± 0.10	11.81 ± 3.88 (92%) 4.81 ± 0.68	14.70 ± 4.70 (78%) 3.50 ± 0.23	11.81 ± 3.86 (98%) 3.57 ± 0.39	12.09 ± 3.63 (100%) 4.39 ± 0.53
	50%	14.40 ± 4.09 (100%) 2.95 ± 0.10	11.81 ± 3.88 (92%) 4.81 ± 0.68	15.80 ± 5.35 (96%) 3.53 ± 0.26	11.81 ± 3.86 (98%) 3.57 ± 0.39	12.51 ± 3.96 (100%) 4.01 ± 0.46
	75%	14.40 ± 4.09 (100%) 2.95 ± 0.10	11.81 ± 3.88 (92%) 4.81 ± 0.68	15.80 ± 5.35 (96%) 3.53 ± 0.26	11.81 ± 3.86 (98%) 3.57 ± 0.39	13.06 ± 3.82 (100%) 3.80 ± 0.42
	100%	14.40 ± 4.09 (100%) 2.95 ± 0.10	11.81 ± 3.88 (92%) 4.81 ± 0.68	15.80 ± 5.35 (96%) 3.53 ± 0.26	11.81 ± 3.86 (98%) 3.57 ± 0.39	13.21 ± 3.65 (100%) 3.67 ± 0.38

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	N/A	9.75 ± 3.52 (66%) 8.40 ± 1.42	N/A	7.45 ± 3.43 (4%) 5.90 ± 0.30	10.53 ± 4.59 (78%) 7.89 ± 1.26
	25%	15.04 ± 4.22 (20%) 5.64 ± 0.46	10.84 ± 4.52 (92%) 8.23 ± 1.33	17.16 ± 4.15 (18%) 5.80 ± 0.30	13.42 ± 5.15 (78%) 6.00 ± 0.45	12.16 ± 4.92 (94%) 6.94 ± 0.88
	50%	17.99 ± 5.55 (66%) 5.71 ± 0.31	10.84 ± 4.52 (92%) 8.23 ± 1.33	18.44 ± 4.88 (68%) 5.80 ± 0.28	13.97 ± 4.80 (98%) 6.02 ± 0.46	12.94 ± 4.69 (100%) 6.54 ± 0.70
	75%	18.66 ± 5.35 (92%) 5.71 ± 0.28	10.84 ± 4.52 (92%) 8.23 ± 1.33	18.52 ± 4.93 (96%) 5.81 ± 0.25	13.97 ± 4.80 (98%) 6.02 ± 0.46	14.07 ± 4.87 (100%) 6.29 ± 0.59
	100%	18.64 ± 5.39 (96%) 5.71 ± 0.27	10.84 ± 4.52 (92%) 8.23 ± 1.33	18.52 ± 4.93 (96%) 5.81 ± 0.25	13.97 ± 4.80 (98%) 6.02 ± 0.46	14.74 ± 5.36 (100%) 6.09 ± 0.46
10	0%	N/A	18.81 ± 6.66 (52%) 8.04 ± 0.82	N/A	18.41 ± 5.73 (8%) 6.28 ± 0.11	20.18 ± 6.56 (74%) 7.97 ± 0.71
	25%	28.79 ± 6.51 (16%) 5.75 ± 0.16	20.57 ± 6.97 (84%) 8.07 ± 0.72	28.67 ± 8.12 (14%) 5.80 ± 0.12	24.43 ± 7.54 (96%) 6.32 ± 0.35	22.69 ± 7.66 (100%) 7.21 ± 0.60
	50%	33.80 ± 8.81 (78%) 5.74 ± 0.15	20.57 ± 6.97 (84%) 8.07 ± 0.72	34.81 ± 8.85 (74%) 5.81 ± 0.14	24.71 ± 7.52 (100%) 6.33 ± 0.34	24.78 ± 7.94 (100%) 6.79 ± 0.54
	75%	34.96 ± 9.35 (94%) 5.74 ± 0.14	20.57 ± 6.97 (84%) 8.07 ± 0.72	35.48 ± 9.09 (92%) 5.82 ± 0.14	24.71 ± 7.52 (100%) 6.33 ± 0.34	26.02 ± 8.11 (100%) 6.46 ± 0.46
	100%	34.88 ± 9.26 (96%) 5.74 ± 0.14	20.57 ± 6.97 (84%) 8.07 ± 0.72	35.48 ± 9.09 (92%) 5.82 ± 0.14	24.71 ± 7.52 (100%) 6.33 ± 0.34	27.80 ± 8.36 (100%) 6.24 ± 0.38

Table 4: **SC2 Staging 08 - Easy(above) and Medium(below) Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	45.32 ± 4.65 (4%) 12.20 ± 0.00	29.99 ± 4.40 (48%) 15.92 ± 0.85	55.20 ± 0.00 (2%) 12.60 ± 0.00	32.30 ± 6.84 (21%) 13.86 ± 0.56	32.06 ± 6.49 (71%) 15.32 ± 0.87
	25%	46.19 ± 6.18 (29%) 12.26 ± 0.19	31.21 ± 5.19 (77%) 15.89 ± 0.98	46.40 ± 5.57 (10%) 12.52 ± 0.10	36.44 ± 7.90 (90%) 13.49 ± 0.78	37.21 ± 6.86 (98%) 13.84 ± 0.81
	50%	48.08 ± 6.48 (90%) 12.18 ± 0.30	31.21 ± 5.19 (77%) 15.89 ± 0.98	52.80 ± 9.27 (77%) 12.48 ± 0.36	37.46 ± 8.24 (100%) 13.45 ± 0.75	41.13 ± 8.32 (100%) 13.22 ± 0.64
	75%	48.62 ± 6.63 (96%) 12.19 ± 0.30	31.73 ± 6.01 (79%) 16.04 ± 1.33	53.57 ± 9.71 (94%) 12.43 ± 0.41	37.46 ± 8.24 (100%) 13.45 ± 0.75	44.59 ± 9.58 (100%) 12.86 ± 0.55
	100%	48.58 ± 6.51 (100%) 12.19 ± 0.29	31.73 ± 6.01 (79%) 16.04 ± 1.33	53.57 ± 9.71 (94%) 12.43 ± 0.41	37.46 ± 8.24 (100%) 13.45 ± 0.75	48.47 ± 9.89 (100%) 12.68 ± 0.58
10	0%	90.65 ± 0.00 (3%) 12.20 ± 0.00	63.68 ± 7.12 (14%) 15.80 ± 0.79	N/A	64.06 ± 16.18 (14%) 13.84 ± 0.65	64.78 ± 8.71 (49%) 15.79 ± 0.77
	25%	96.22 ± 9.49 (30%) 12.15 ± 0.17	65.46 ± 8.09 (38%) 16.09 ± 1.02	106.05 ± 0.00 (3%) 12.50 ± 0.00	72.44 ± 12.29 (84%) 13.72 ± 0.61	75.57 ± 11.89 (92%) 13.93 ± 0.73
	50%	95.94 ± 11.69 (92%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	105.49 ± 13.56 (76%) 12.36 ± 0.33	72.44 ± 12.29 (84%) 13.72 ± 0.61	85.44 ± 13.65 (100%) 13.42 ± 0.87
	75%	96.13 ± 11.58 (95%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	107.69 ± 15.01 (84%) 12.41 ± 0.39	72.44 ± 12.29 (84%) 13.72 ± 0.61	90.52 ± 13.15 (97%) 12.91 ± 0.74
	100%	96.13 ± 11.58 (95%) 12.15 ± 0.21	67.26 ± 10.32 (41%) 16.31 ± 1.27	107.69 ± 15.01 (84%) 12.41 ± 0.39	72.44 ± 12.29 (84%) 13.72 ± 0.61	96.08 ± 15.02 (97%) 12.70 ± 0.66

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	6.42 ± 2.37 (11%) 2.88 ± 0.10	5.91 ± 2.50 (85%) 5.05 ± 1.41	6.98 ± 3.77 (9%) 3.20 ± 0.14	6.66 ± 3.49 (34%) 3.45 ± 0.18	6.07 ± 2.69 (87%) 5.08 ± 1.40
	25%	7.23 ± 3.06 (87%) 2.95 ± 0.09	6.14 ± 2.84 (98%) 5.12 ± 1.40	7.19 ± 2.76 (66%) 3.41 ± 0.34	6.65 ± 2.99 (100%) 3.40 ± 0.30	6.45 ± 2.97 (100%) 4.37 ± 0.93
	50%	7.59 ± 3.31 (98%) 2.95 ± 0.09	6.14 ± 2.84 (98%) 5.12 ± 1.40	8.09 ± 3.00 (91%) 3.47 ± 0.38	6.65 ± 2.99 (100%) 3.40 ± 0.30	6.54 ± 3.01 (100%) 3.91 ± 0.74
	75%	7.71 ± 3.37 (100%) 2.95 ± 0.08	6.14 ± 2.84 (98%) 5.12 ± 1.40	8.67 ± 3.69 (98%) 3.48 ± 0.37	6.65 ± 2.99 (100%) 3.40 ± 0.30	6.65 ± 2.93 (100%) 3.76 ± 0.61
	100%	7.71 ± 3.37 (100%) 2.95 ± 0.08	6.14 ± 2.84 (98%) 5.12 ± 1.40	9.03 ± 4.42 (100%) 3.51 ± 0.41	6.65 ± 2.99 (100%) 3.40 ± 0.30	6.94 ± 3.27 (100%) 3.60 ± 0.48
10	0%	8.64 ± 1.19 (4%) 2.90 ± 0.10	11.10 ± 3.63 (82%) 4.96 ± 1.00	N/A	12.03 ± 4.28 (31%) 3.36 ± 0.25	10.97 ± 3.57 (87%) 4.91 ± 0.98
	25%	14.18 ± 4.28 (91%) 2.94 ± 0.07	11.53 ± 4.29 (93%) 4.99 ± 1.00	15.65 ± 4.29 (64%) 3.52 ± 0.26	12.00 ± 3.85 (98%) 3.51 ± 0.43	12.12 ± 4.19 (100%) 4.24 ± 0.67
	50%	14.28 ± 4.29 (96%) 2.95 ± 0.07	11.53 ± 4.29 (93%) 4.99 ± 1.00	17.24 ± 4.96 (98%) 3.54 ± 0.25	12.31 ± 4.33 (100%) 3.56 ± 0.54	12.39 ± 4.16 (100%) 3.87 ± 0.53
	75%	14.66 ± 4.92 (98%) 2.97 ± 0.15	11.53 ± 4.29 (93%) 4.99 ± 1.00	17.24 ± 4.96 (98%) 3.54 ± 0.25	12.31 ± 4.33 (100%) 3.56 ± 0.54	12.67 ± 4.12 (100%) 3.62 ± 0.40
	100%	14.66 ± 4.92 (98%) 2.97 ± 0.15	11.53 ± 4.29 (93%) 4.99 ± 1.00	17.24 ± 4.96 (98%) 3.54 ± 0.25	12.31 ± 4.33 (100%) 3.56 ± 0.54	13.38 ± 4.53 (100%) 3.49 ± 0.34

Table 5: **SC2 Staging 08 - Hard(above), SC0 Staging 20 - Easy(below) Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	23.44 ± 0.00 (2%) 6.80 ± 0.00	8.97 ± 3.91 (59%) 11.08 ± 1.81	N/A	9.50 ± 4.97 (6%) 8.13 ± 1.00	9.63 ± 4.78 (65%) 10.89 ± 1.81
	25%	16.04 ± 6.79 (16%) 6.85 ± 0.17	9.20 ± 4.06 (88%) 11.16 ± 1.76	N/A	12.74 ± 5.13 (43%) 7.97 ± 1.07	12.24 ± 5.13 (84%) 8.99 ± 1.58
	50%	17.81 ± 6.21 (55%) 6.86 ± 0.15	9.20 ± 4.06 (88%) 11.16 ± 1.76	20.02 ± 5.43 (27%) 7.02 ± 0.17	15.28 ± 6.27 (90%) 7.75 ± 0.97	14.12 ± 6.09 (98%) 8.21 ± 1.29
	75%	20.27 ± 7.42 (90%) 6.90 ± 0.21	9.42 ± 4.28 (90%) 11.16 ± 1.74	22.83 ± 7.95 (84%) 7.01 ± 0.20	15.37 ± 6.11 (98%) 7.73 ± 0.97	15.41 ± 6.94 (98%) 7.73 ± 0.96
	100%	21.16 ± 8.10 (98%) 6.92 ± 0.32	9.42 ± 4.28 (90%) 11.16 ± 1.74	24.83 ± 11.14 (100%) 7.21 ± 0.88	16.40 ± 9.30 (100%) 7.82 ± 1.14	18.19 ± 10.01 (100%) 7.57 ± 1.10
10	0%	44.20 ± 0.00 (2%) 6.80 ± 0.00	18.26 ± 5.14 (52%) 11.09 ± 1.34	N/A	31.13 ± 0.00 (2%) 7.50 ± 0.00	20.75 ± 7.75 (43%) 11.38 ± 1.45
	25%	49.12 ± 4.92 (4%) 6.75 ± 0.05	19.45 ± 6.42 (83%) 11.21 ± 1.20	N/A	27.33 ± 6.75 (61%) 8.22 ± 0.69	25.06 ± 7.29 (85%) 9.53 ± 0.98
	50%	40.88 ± 7.98 (46%) 6.87 ± 0.12	19.65 ± 6.46 (85%) 11.23 ± 1.20	43.27 ± 8.55 (33%) 7.00 ± 0.13	29.01 ± 8.68 (91%) 8.21 ± 0.70	29.40 ± 8.62 (91%) 8.39 ± 0.91
	75%	44.70 ± 10.70 (89%) 6.93 ± 0.18	19.65 ± 6.46 (85%) 11.23 ± 1.20	48.50 ± 12.26 (89%) 7.08 ± 0.18	30.15 ± 11.31 (93%) 8.26 ± 0.76	33.91 ± 12.45 (98%) 7.90 ± 0.86
	100%	44.79 ± 10.58 (91%) 6.94 ± 0.19	19.65 ± 6.46 (85%) 11.23 ± 1.20	50.89 ± 15.57 (98%) 7.19 ± 0.48	30.15 ± 11.31 (93%) 8.26 ± 0.76	37.34 ± 12.40 (96%) 7.56 ± 0.67

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	N/A	19.08 ± 7.14 (36%) 17.92 ± 1.94	N/A	22.30 ± 10.28 (6%) 14.40 ± 0.60	19.67 ± 6.70 (45%) 18.29 ± 1.75
	25%	35.33 ± 6.22 (21%) 14.26 ± 0.92	20.33 ± 7.42 (91%) 18.37 ± 2.07	N/A	25.80 ± 9.08 (64%) 15.16 ± 1.14	27.89 ± 8.52 (73%) 16.57 ± 1.82
	50%	35.80 ± 7.25 (73%) 14.28 ± 1.20	20.45 ± 7.33 (94%) 18.43 ± 2.07	50.08 ± 6.69 (15%) 13.76 ± 0.34	29.01 ± 10.67 (91%) 15.09 ± 1.32	34.45 ± 8.67 (94%) 15.03 ± 1.40
	75%	37.15 ± 7.96 (94%) 14.10 ± 1.16	20.45 ± 7.33 (94%) 18.43 ± 2.07	49.33 ± 8.51 (85%) 13.84 ± 0.57	29.01 ± 10.67 (91%) 15.09 ± 1.32	39.18 ± 8.32 (91%) 14.27 ± 1.15
	100%	37.15 ± 7.96 (94%) 14.10 ± 1.16	20.45 ± 7.33 (94%) 18.43 ± 2.07	49.81 ± 8.87 (97%) 13.80 ± 0.60	29.01 ± 10.67 (91%) 15.09 ± 1.32	43.78 ± 8.36 (94%) 13.99 ± 0.79
10	0%	N/A	34.92 ± 9.42 (22%) 17.63 ± 1.11	N/A	N/A	41.60 ± 4.99 (15%) 16.98 ± 2.09
	25%	83.32 ± 8.80 (15%) 13.83 ± 0.75	41.30 ± 12.05 (74%) 18.41 ± 1.26	N/A	50.84 ± 12.88 (67%) 15.14 ± 1.00	55.65 ± 13.76 (81%) 15.67 ± 0.98
	50%	75.00 ± 13.51 (81%) 13.84 ± 0.78	42.74 ± 13.42 (78%) 18.63 ± 1.57	92.49 ± 0.00 (4%) 13.80 ± 0.00	52.11 ± 13.30 (78%) 15.00 ± 1.00	70.64 ± 17.69 (100%) 14.83 ± 0.85
	75%	75.07 ± 12.82 (93%) 13.81 ± 0.75	42.74 ± 13.42 (78%) 18.63 ± 1.57	98.47 ± 14.32 (93%) 13.65 ± 0.53	52.11 ± 13.30 (78%) 15.00 ± 1.00	77.74 ± 15.75 (93%) 14.06 ± 0.76
	100%	75.07 ± 12.82 (93%) 13.81 ± 0.75	42.74 ± 13.42 (78%) 18.63 ± 1.57	98.47 ± 14.32 (93%) 13.65 ± 0.53	52.11 ± 13.30 (78%) 15.00 ± 1.00	85.53 ± 15.05 (100%) 13.81 ± 0.61

Table 6: **SC0 Staging 20 - Medium(above) and Hard(below) Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	1.05 ± 0.74 (10%) 3.00 ± 0.00	2.51 ± 1.60 (78%) 4.89 ± 0.94	2.47 ± 1.23 (16%) 3.65 ± 0.30	2.02 ± 1.55 (36%) 3.36 ± 0.30	2.49 ± 1.56 (78%) 4.90 ± 0.94
	25%	2.88 ± 1.95 (64%) 3.00 ± 0.00	2.69 ± 1.59 (96%) 4.90 ± 0.96	3.06 ± 1.55 (60%) 3.62 ± 0.33	2.78 ± 1.68 (88%) 3.50 ± 0.34	2.73 ± 1.54 (92%) 4.46 ± 0.89
	50%	3.12 ± 1.93 (82%) 3.00 ± 0.00	2.68 ± 1.58 (100%) 4.88 ± 0.94	3.80 ± 2.19 (90%) 3.64 ± 0.31	2.84 ± 1.72 (98%) 3.51 ± 0.35	2.79 ± 1.64 (100%) 4.32 ± 1.06
	75%	3.39 ± 2.13 (88%) 3.00 ± 0.00	2.68 ± 1.58 (100%) 4.88 ± 0.94	3.77 ± 2.16 (94%) 3.63 ± 0.32	2.83 ± 1.71 (100%) 3.50 ± 0.35	2.87 ± 1.68 (100%) 4.20 ± 1.04
	100%	3.50 ± 2.26 (94%) 3.00 ± 0.00	2.68 ± 1.58 (100%) 4.88 ± 0.94	3.83 ± 2.12 (100%) 3.62 ± 0.32	2.83 ± 1.71 (100%) 3.50 ± 0.35	2.90 ± 1.62 (100%) 4.07 ± 1.03
10	0%	4.79 ± 3.07 (4%) 3.00 ± 0.00	4.89 ± 2.34 (84%) 4.73 ± 0.72	8.32 ± 0.00 (2%) 3.80 ± 0.00	3.38 ± 1.75 (28%) 3.50 ± 0.28	4.66 ± 2.21 (76%) 4.64 ± 0.60
	25%	6.38 ± 2.97 (72%) 3.01 ± 0.02	5.08 ± 2.27 (98%) 4.80 ± 0.81	6.82 ± 2.72 (68%) 3.61 ± 0.25	5.54 ± 2.39 (100%) 3.59 ± 0.30	5.09 ± 2.15 (94%) 4.27 ± 0.59
	50%	6.96 ± 2.86 (98%) 3.01 ± 0.02	5.12 ± 2.27 (100%) 4.80 ± 0.80	7.88 ± 3.30 (96%) 3.61 ± 0.25	5.54 ± 2.39 (100%) 3.59 ± 0.30	5.51 ± 2.90 (98%) 4.11 ± 0.69
	75%	7.14 ± 3.10 (100%) 3.01 ± 0.02	5.12 ± 2.27 (100%) 4.80 ± 0.80	8.06 ± 3.41 (100%) 3.61 ± 0.24	5.54 ± 2.39 (100%) 3.59 ± 0.30	5.56 ± 2.34 (100%) 3.90 ± 0.61
	100%	7.14 ± 3.10 (100%) 3.01 ± 0.02	5.12 ± 2.27 (100%) 4.80 ± 0.80	8.06 ± 3.41 (100%) 3.61 ± 0.24	5.54 ± 2.39 (100%) 3.59 ± 0.30	5.67 ± 2.46 (100%) 3.82 ± 0.55

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	11.85 ± 0.00 (2%) 7.00 ± 0.00	6.98 ± 4.44 (62%) 11.01 ± 1.82	N/A	N/A	6.95 ± 4.73 (60%) 10.83 ± 2.10
	25%	16.65 ± 5.90 (6%) 7.07 ± 0.09	8.06 ± 4.75 (90%) 11.15 ± 1.81	13.42 ± 0.00 (2%) 6.80 ± 0.00	10.44 ± 5.54 (28%) 7.93 ± 1.15	9.18 ± 5.05 (86%) 9.33 ± 1.60
	50%	16.67 ± 5.28 (26%) 7.05 ± 0.08	8.27 ± 4.80 (94%) 11.26 ± 1.85	16.84 ± 5.19 (18%) 7.00 ± 0.09	12.62 ± 5.52 (78%) 7.63 ± 0.85	10.62 ± 5.32 (98%) 8.61 ± 1.38
	75%	17.16 ± 4.94 (68%) 7.01 ± 0.08	8.38 ± 4.73 (98%) 11.31 ± 1.83	18.62 ± 6.09 (74%) 7.04 ± 0.14	13.16 ± 5.44 (98%) 7.55 ± 0.79	11.95 ± 5.74 (98%) 8.00 ± 1.01
	100%	18.65 ± 5.97 (88%) 7.00 ± 0.07	8.54 ± 4.82 (100%) 11.33 ± 1.81	18.49 ± 5.75 (98%) 7.03 ± 0.13	13.25 ± 5.42 (100%) 7.54 ± 0.79	13.03 ± 5.88 (100%) 7.75 ± 0.87
10	0%	N/A	13.88 ± 4.20 (43%) 10.61 ± 1.21	N/A	22.34 ± 4.63 (6%) 8.50 ± 0.98	16.16 ± 5.93 (35%) 10.61 ± 1.07
	25%	17.83 ± 5.59 (4%) 7.00 ± 0.00	18.24 ± 7.58 (82%) 10.95 ± 1.29	N/A	21.74 ± 7.02 (53%) 8.18 ± 0.88	20.55 ± 7.42 (86%) 9.50 ± 1.02
	50%	33.31 ± 9.51 (33%) 7.00 ± 0.05	19.91 ± 8.42 (94%) 11.04 ± 1.30	33.73 ± 3.68 (10%) 7.02 ± 0.04	24.12 ± 7.74 (98%) 8.22 ± 0.87	24.55 ± 7.61 (92%) 8.60 ± 0.97
	75%	36.90 ± 9.65 (82%) 7.01 ± 0.08	20.02 ± 8.36 (96%) 11.01 ± 1.29	38.59 ± 9.69 (88%) 7.04 ± 0.09	24.12 ± 7.74 (98%) 8.22 ± 0.87	26.99 ± 8.02 (98%) 8.23 ± 0.81
	100%	37.70 ± 9.39 (94%) 7.01 ± 0.07	20.02 ± 8.36 (96%) 11.01 ± 1.29	38.73 ± 9.62 (98%) 7.04 ± 0.09	24.12 ± 7.74 (98%) 8.22 ± 0.87	29.04 ± 8.86 (98%) 7.92 ± 0.67

Table 7: **SC3 Staging 11 - Easy(above) and Medium(below) Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

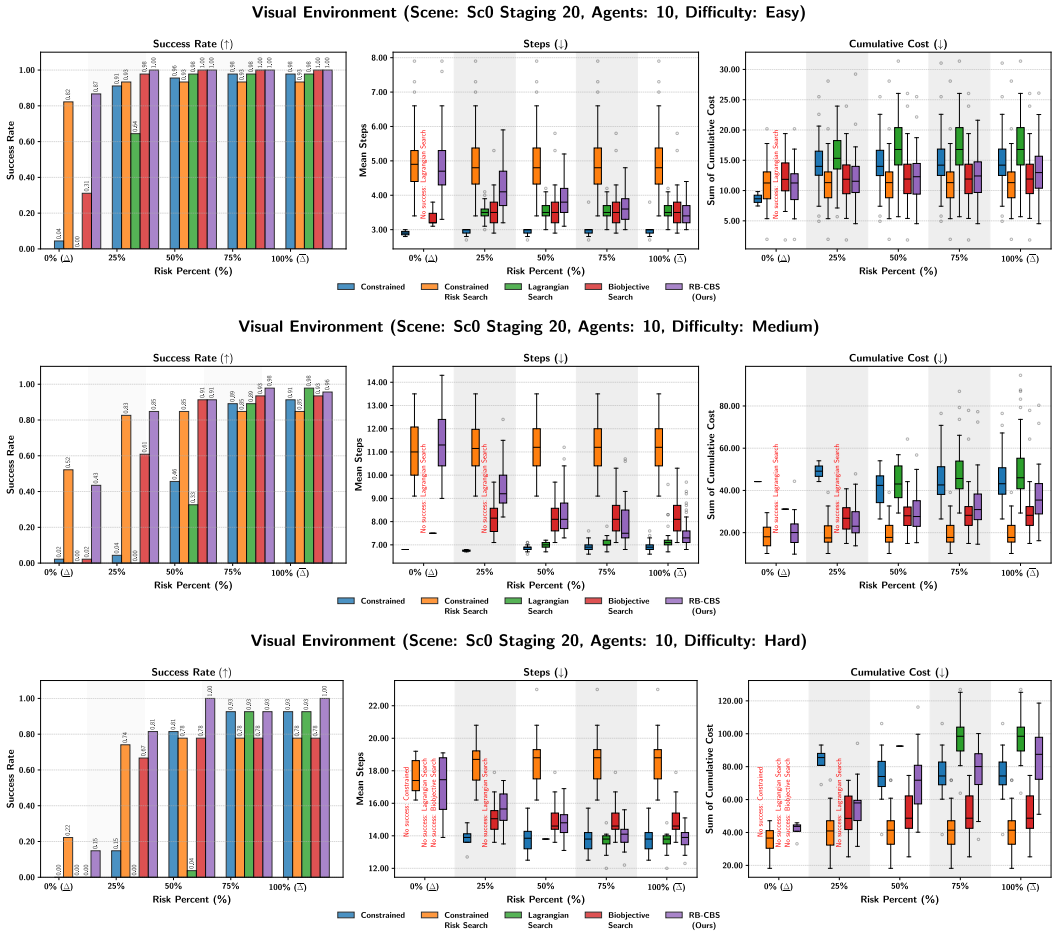

Figure 12: Quantitative results for the visual navigation environment on SC0 Staging 20 scene comparing all methods across Easy, Medium and Hard difficulties for 10 agents.

Problem Configurations		Methods (Cumulative Cost (Success Rate) / Average Steps)				
Agents	**Δ (%)**	**Constrained Policy**	**Constrained Risk Search**	**Lagrangian Search**	**Bi-Objective Search**	**RB-CBS (Ours)**
5	0%	43.89 ± 0.41 (4%) 13.60 ± 0.00	37.54 ± 6.07 (54%) 19.13 ± 1.89	N/A	41.11 ± 3.75 (12%) 15.90 ± 1.78	40.15 ± 7.65 (42%) 19.04 ± 1.89
	25%	49.99 ± 6.08 (30%) 14.59 ± 1.17	39.90 ± 7.28 (80%) 18.85 ± 1.94	50.72 ± 1.09 (4%) 13.80 ± 0.00	43.98 ± 6.22 (56%) 15.70 ± 1.46	45.56 ± 6.82 (90%) 16.53 ± 1.49
	50%	52.16 ± 6.52 (76%) 14.37 ± 0.85	40.77 ± 8.13 (84%) 19.17 ± 2.41	59.03 ± 8.25 (40%) 13.88 ± 0.22	46.60 ± 7.89 (82%) 15.39 ± 1.37	50.06 ± 7.55 (96%) 15.30 ± 1.13
	75%	52.95 ± 6.81 (88%) 14.33 ± 0.80	41.03 ± 8.20 (86%) 19.27 ± 2.47	61.69 ± 8.92 (94%) 13.91 ± 0.22	46.60 ± 7.89 (82%) 15.39 ± 1.37	53.62 ± 8.32 (96%) 14.66 ± 0.84
	100%	52.95 ± 6.81 (88%) 14.33 ± 0.80	41.73 ± 9.30 (88%) 19.36 ± 2.51	62.75 ± 10.98 (100%) 14.03 ± 0.85	46.60 ± 7.89 (82%) 15.39 ± 1.37	56.75 ± 8.62 (100%) 14.32 ± 0.82
10	0%	N/A	73.40 ± 4.82 (17%) 18.06 ± 0.67	N/A	84.95 ± 6.87 (17%) 15.72 ± 0.53	78.82 ± 13.14 (17%) 17.86 ± 0.78
	25%	103.22 ± 10.08 (30%) 14.67 ± 0.94	81.93 ± 11.46 (47%) 18.22 ± 0.89	N/A	86.89 ± 9.59 (70%) 15.85 ± 0.79	91.90 ± 11.41 (80%) 16.67 ± 1.04
	50%	104.64 ± 11.21 (73%) 14.48 ± 0.68	83.90 ± 13.30 (50%) 18.27 ± 0.87	117.68 ± 12.84 (30%) 13.89 ± 0.13	88.64 ± 10.96 (77%) 15.80 ± 0.77	101.42 ± 13.13 (90%) 15.76 ± 1.17
	75%	105.04 ± 11.12 (77%) 14.46 ± 0.67	83.90 ± 13.30 (50%) 18.27 ± 0.87	122.69 ± 13.02 (87%) 13.92 ± 0.11	88.64 ± 10.96 (77%) 15.80 ± 0.77	106.89 ± 14.36 (87%) 14.87 ± 0.71
	100%	105.04 ± 11.12 (77%) 14.46 ± 0.67	83.90 ± 13.30 (50%) 18.27 ± 0.87	123.59 ± 13.57 (90%) 13.99 ± 0.35	88.64 ± 10.96 (77%) 15.80 ± 0.77	112.88 ± 14.73 (93%) 14.32 ± 0.60

Table 8: **SC3 Staging 11 - Hard Difficulty.** Numerical results for all methods across 5 and 10 agents. Lower cumulative costs, lower steps, and higher success rates are better.

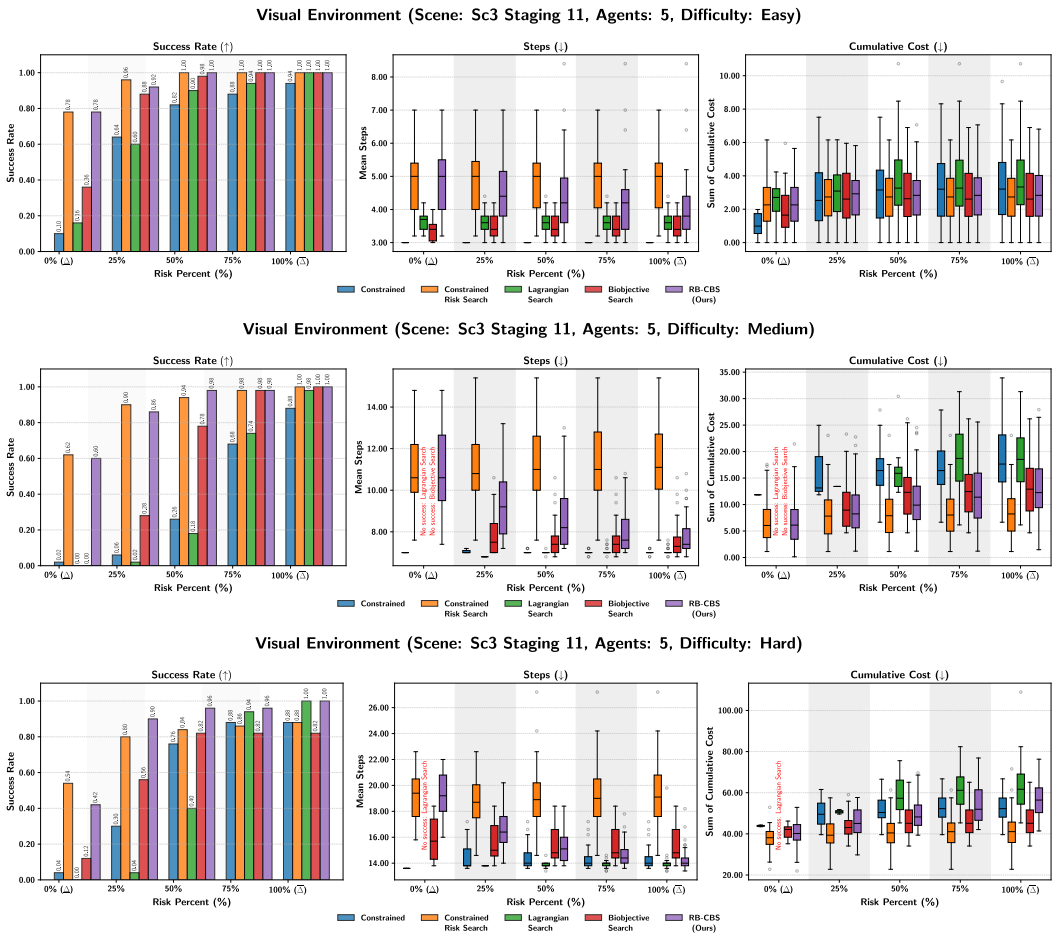

Figure 13: Quantitative results for the visual navigation environment on SC3 Staging 11 scene comparing all methods across Easy, Medium and Hard difficulties for 5 agents.

14900K CPU and an NVIDIA GeForce RTX 4090 GPU. The algorithm is implemented in Python, building upon the official codebase released by Feng et al..

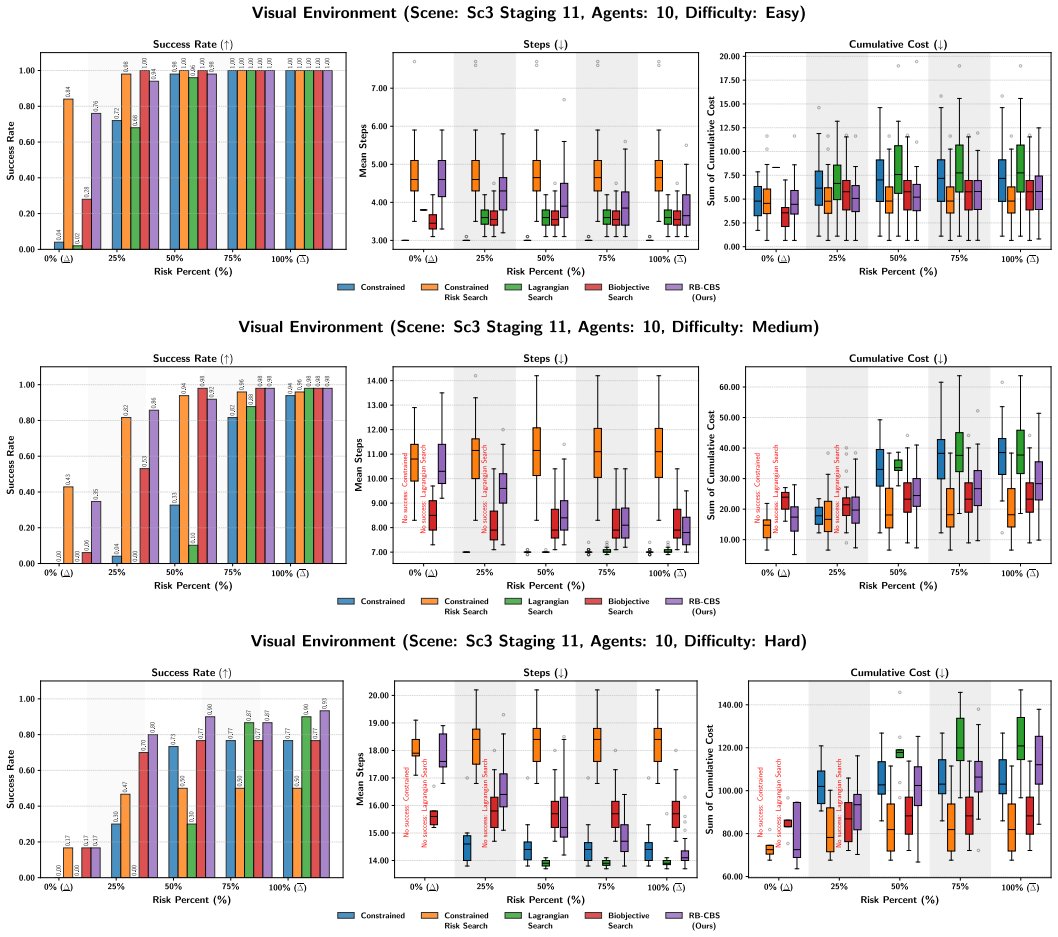

Figure 14: Quantitative results for the visual navigation environment on SC3 Staging 11 scene comparing all methods across Easy, Medium and Hard difficulties for 10 agents.

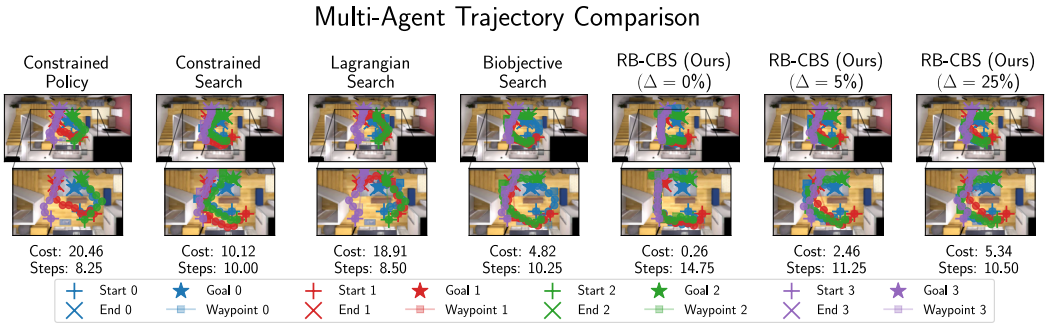

Figure 15: Comparison of different approaches on the visual navigation task for a multi-agent setup. Total cumulative cost and average steps count are annotated below each method. Our RB-CBS planner flexibly trades off efficiency and safety via the user-specified Δ while coordinating multiple agents to reach their goals.

Parameter	Value
Actor Learning Rate	1e-5
Actor Update Interval	1
Critic Learning Rate	1e-4
Cost Critic Learning Rate	1e-4
Distance Critic Bins	20
Cost Critic Bins	40
Targets Update Interval	5
Polyak Update Coefficient	0.05
Initial Lagrange Multiplier	0
Lagrange Learning Rate	0.035
Optimizer	Adam
Visual Inputs Dimensions	(4, 32, 32, 4)
Replay Buffer Size	100000
Batch Size	64
Initial Collect Steps	1000
Training Iterations	600000
Neural Network Architecture	Conv(16, 8, 4) + Conv(32, 4, 4) + FC(256)
Maximum Episode Steps	20

Table 9: Hyperparameters and Training Settings for Visual Navigation