Submitted to the International Conference on Intelligent Robots and Systems (IROS), October 2025.

Extended Version: Multi-Robot Motion Planning with Cooperative Localization

Anne Theurkauf, Nisar Ahmed, Morteza Lahijanian

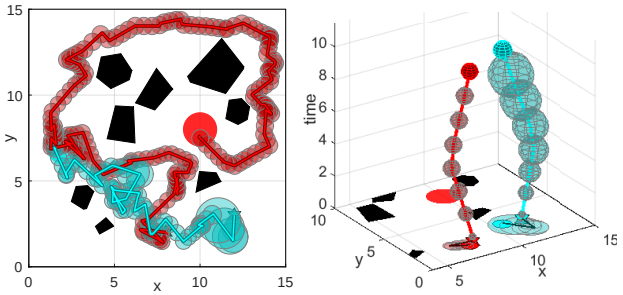
***Abstract*—We consider the uncertain multi-robot motion planning (MRMP) problem with cooperative localization (CL-MRMP), under both motion and measurement noise, where each robot can act as a sensor for its nearby teammates. We formalize CL-MRMP as a chance-constrained motion planning problem, and propose a safety-guaranteed algorithm that explicitly accounts for robot-robot correlations. Our approach extends a sampling-based planner to solve CL-MRMP while preserving probabilistic completeness. To improve efficiency, we introduce novel biasing techniques. We evaluate our method across diverse benchmarks, demonstrating its effectiveness in generating motion plans, with significant performance gains from biasing strategies.**

I. Introduction

Multi-robot teams are powerful assets, offering diverse capabilities and enabling parallel operation to improve efficiency and coverage in applications ranging from warehouse automation to space exploration [10], [13]. Multi-robot teams are particularly advantageous in adversarial environments such as GPS-denied settings, as each robot can serve as a sensor for others, reducing overall uncertainty through cooperation localization (CL). For example, robots can obtain relative measurements from nearby teammates [3] to correct drift from inertial sensors [11], [16] (e.g., Fig. 1). This however introduces a significant challenge in motion planning, as the planner must not only determine collision-free trajectories but also coordinate CL opportunities. In this work, we focus on the multi-robot motion planning (MRMP) problem for teams operating with CL, which we refer to as *CL-MRMP*.

A general approach to MRMP is centralized, coupled planning, which models all robots as a single meta-agent and plans for them simultaneously in the joint state space [20]. This is computationally challenging since the search space grows exponentially with the number of robots. Nevertheless, it allows the planner to maintain information about all the robots. More scalable decoupled methods [8], [18] plan for individual robots and selectively resolve conflicts. Yet, they neglect or only partly account for robot-robot interactions. This is problematic for scenarios requiring safe navigation via CL, which introduces correlations between robots' state estimates [15] that must be considered for feasible planning.

Online methods can mitigate this challenge by planning for the robot team over a short horizon. A common approach for CL is online distributed planning, where each robot plans locally over a short horizon while accounting for nearby robots and obstacles [12], [19]. Another widely used technique is formation control, in which robot formations are designed to minimize uncertainty for the team [5], [14], and online execution consists of maintaining the prescribed formation [1], [4].



(a) Trajectories (b) Trajectory with time axis

Fig. 1: CL-MRMP solution plan for 2 robots with motion and sensing uncertainties (initial states are near the bottom of the figure, and their goal regions are indicated by red and cyan circles). Cyan robot lacks onboard sensors, but the solution plan enables it to use the red robot as a sensor, reducing its uncertainty and allowing it to successfully navigate to its goal region. Afterward, the plan guides the red robot to its goal. (trajectory circles: 2σ bounds).

While these methods are effective in unknown environments, they lack formal guarantees for safety or completeness.

This work formally defines the CL-MRMP problem and proposes a safety-guaranteed algorithm that explicitly accounts for both motion and measurement uncertainty. We first establish the necessity of a centralized estimator to track the coupling of robot states induced by CL. Then, we formulate CL-MRMP as a chance-constrained planning problem, allowing us to adapt existing algorithms. We extend the sampling-based algorithm in [6] to solve CL-MRMP and demonstrate that it inherits the probabilistic completeness properties of the underlying algorithm. Additionally, we introduce biasing techniques to improve performance. Our algorithm is evaluated on a diverse set of benchmarking problems, and the results show that it effectively addresses the CL-MRMP problem, with our biasing techniques significantly enhancing performance.

Our main contributions are: (i) a formalization of the CL-MRMP problem as a chance-constrained motion planning problem, (ii) a sampling-based planning algorithm that accurately accounts for robot-robot correlations in state estimates, (iii) novel biasing techniques for more efficient planning with CL, and (iv) extensive benchmarks and illustrative examples demonstrating the efficacy of our formulation and approach.

II. Problem Formulation

In this work, we consider $N_A \in \mathbb{N}_{\geq 2}$ robots that evolve in a shared workspace $\mathcal{W} \subset \mathbb{R}^w$, $w \in \{2, 3\}$ under both motion and sensor uncertainties. These robots are capable of sensing and communicating with nearby robots.

A. Robot Dynamics

The evolution of robot $i \in \{1, \ldots, N_A\}$ is governed by stochastic linear dynamics

$$x^i_{k+1} = A^i x^i_k + B^i u^i_k + w^i_k, \tag{1}$$

where $x^i_k \in \mathcal{X}^i \subseteq \mathbb{R}^{n_i}$ and $u^i_k \in \mathcal{U}^i \subseteq \mathbb{R}^{m_i}$ are the state and control at time step $k \in \mathbb{N}_{\geq 0}$, respectively with associated matrices $A^i \in \mathbb{R}^{n_i \times n_i}$ and $B^i \in \mathbb{R}^{n_i \times m_i}$, and $w^i_k \in \mathbb{R}^{n_i}$ is Gaussian distributed noise with zero mean and $Q^i \in \mathbb{R}^{n_i \times n_i}$ covariance, i.e., $w^i_k \sim \mathcal{N}(0, Q^i)$.

Each robot body $\mathcal{B}^i$ is defined as a set of points. We use $\mathcal{B}^i(x^i_k) \subset \mathcal{W}$ to denote the set of points in the workspace that robot i occupies when placed at state x^i_k.

B. Robot Measurements

We assume the robots are equipped with *proprioceptive* and *exteroceptive* sensors, allowing them to measure (and communicate) not only their own states but also those of nearby robots. Proprioceptive measurements pertain to an individual robot's state and are independent of all others (e.g., velocity encoder). Such measurements of robot i are given by:

$$y^{i,prop}_k = C^{i,prop} x^i_k + v^{i,prop}_k, \tag{2}$$

where $y^{i,prop}_k \in \mathbb{R}^{q^{prop}_i}$, with associated matrix $C^{i,prop} \in \mathbb{R}^{q^{prop}_i \times n_i}$, and zero-mean Gaussian distributed noise $v^{i,prop}_k \sim \mathcal{N}(0, R^{i,prop})$, with covariance $R^{i,prop} \in \mathbb{R}^{q^{prop}_i \times q^{prop}_i}$.

In contrast, exteroceptive measurements are taken relative to other robots, introducing dependencies between their states (e.g., range). Such measurement between robots i and j, where $i \neq j \in \{1, \ldots, N_r\}$, are modeled as

$$y^{ij,ext}_k = C^{ij,ext}_k x^{ij}_k + v^{ij,ext}_k, \tag{3}$$

where $y^{ij,ext}_k \in \mathbb{R}^{m^{ext}_{ij}}$ is defined with respect to the composed state $x^{ij}_k = [x^i_k, x^j_k]^T \in \mathbb{R}^{n_i+n_j}$, with associated mapping $C^{ij,ext}_k \in \mathbb{R}^{m^{ext}_{ij} \times (n_i+n_j)}$ and noise $v^{ij,ext}_k \sim \mathcal{N}(0, R^{ij,ext}_k)$ and covariance $R^{ij,ext}_k \in \mathbb{R}^{m^{ext}_{ij} \times m^{ext}_{ij}}$. Measurement $y^{ij,ext}_k$ is *only enabled* when the robots are within workspace radius $r_{ext} \in \mathbb{R}_{\geq 0}$, i.e., for $\text{PROJ}_{\mathcal{W}} : \cup_{i=1}^{N_A} \mathcal{X}^i \to \mathcal{W}$ denoting the projection operator of the state into the workspace,

$$\|\text{PROJ}_{\mathcal{W}}(x^i_k) - \text{PROJ}_{\mathcal{W}}(x^j_k)\| \leq r_{ext} \implies y^{ij,ext}_k \text{ available to robots } i \text{ and } j, \tag{4}$$

else $y^{ij,ext}_k$ does not exist. In contrast, note $y^{i,prop}_k$ is always available to robot i. Also note that $C^{ij,ext}_k$ is time-varying as condition (4) depends on the time-varying robot states.

C. Centralized Estimation and Cooperative Localization

Exteroceptive measurements induce correlations between robot states. To fully account for all correlations, we assume a centralized estimator over the composed states of all robots that has access to all measurements. We denote concatenation of a set of N (column) vectors $\{\text{v}_i\}_{i=1}^N$ as

$$\text{CONCAT}(\{\text{v}_i\}_{i=1}^N) = [\text{v}_1^T, \text{v}_2^T, \ldots, \text{v}_N^T]^T, \tag{5}$$

and the block diagonal matrix constructed from a set of N matrices $\{M_i\}_{i=1}^N$ as $\text{BLOCKDIAG}(\{M_i\}_{i=1}^N)$.

The composed dynamics of the team of robots are given by:

$$X_{k+1} = AX_k + BU_k + W_k. \tag{6}$$

where $X_k = \text{CONCAT}(\{x^i_k\}_{i=1}^{N_A}) \in \mathcal{X} \subseteq \mathbb{R}^{n_N}$ with $n_N = \sum_{i=1}^{N_A} n_i$ and $U_k = \text{CONCAT}(\{u^i_k\}_{i=1}^{N_A}) \in \mathcal{U} \subseteq \mathbb{R}^{m_N}$ with $m_N = \sum_i^{N_A} m_i$ are the composed state and control, respectively, and matrices $A = \text{BLOCKDIAG}(\{A^i\}_{i=1}^{N_A})$ and $B = \text{BLOCKDIAG}(\{B^i\}_{i=1}^{N_A})$. The composed noise is distributed as $W_k \sim \mathcal{N}(0, Q)$, with $Q = \text{BLOCKDIAG}(\{Q^i\}_{i=1}^{N_A})$.

The composed robot proprioceptive measurement model is:

$$Y^{prop}_k = C^{prop} X_k + V^{prop}_k \tag{7}$$

where $Y^{prop}_k = \text{CONCAT}(\{y^{i,prop}_k\}_{i=1}^{N_A})$ and $V^{prop}_k \sim \mathcal{N}(0, R^{prop})$. Because the individual proprioceptive measurements are independent, $C^{prop} = \text{BLOCKDIAG}(\{C^{i,prop}\}_{i=1}^{N_A})$ and $R^{prop} = \text{BLOCKDIAG}(\{R^{i,prop}\}_{i=1}^{N_A})$.

The composed exteroceptive measurement model is:

$$Y^{ext}_k = C^{ext}_k X_k + V^{ext}_k \tag{8}$$

where $Y^{ext}_k = \text{CONCAT}(\{y^{ij,ext}_k\}_{\models})$, with $\{y^{ij,ext}_k\}_{\models}$ being the set of measurements obtained according to (4), and noise $V^{ext}_k \sim \mathcal{N}(0, R^{ext}_k)$ with $R^{ext}_k = \text{BLOCKDIAG}(\{R^{ij,ext}_k\}_{\models})$. Note that the constituent $C^{ij,ext}_k$ matrices are defined over the states of the two robots i and j, and therefore C^{ext}_k is not a simple BLOCKDIAG. It instead must be constructed to preserve the mapping of the individual $C^{ij,ext}_k$ to the full composed state. Additionally, while C^{prop} is time-invariant, C^{ext}_k is time-varying as the distances between robots change (as noted earlier). The full measurement equation is thus:

$$Y_k = C_k X_k + V_k, \tag{9}$$

where $Y_k = \text{CONCAT}(\{Y^{prop}_k, Y^{ext}_k\})$, $C_k = \text{CONCAT}(\{C^{prop}_k, C^{ext}_k\})$, and noise $V_k \sim \mathcal{N}(0, R_k)$, with covariance $R_k = \text{BLOCKDIAG}(\{R^{prop}, R^{ext}_k\})$. If no robot pairs satisfy (4) at time step k, then (9) reduces to $Y_k = Y^{prop}_k$ with $C_k = C^{prop}$.

The composed system therefore reduces the multi-robot system to a single linear system with Gaussian noise governed by (6) and (9). Hence, we can use a centralized Kalman Filter (KF) to maintain an online estimate of X_k as Gaussian belief $b(X_k)$ with mean $\hat{X}_k \in \mathbb{R}^{n_N}$ and covariance $\Sigma_k \in \mathbb{R}^{n_N \times n_N}$,

$$X_k \sim b(X_k) = \mathcal{N}(\hat{X}_k, \Sigma_k).$$

Note that Σ_k fully captures robot-robot correlations. From this, we can extract the marginal belief for each robot. We denote the belief of robot i state x^i_k by $b(x^i_k)$, i.e.,

$$x^i_k \sim b(x^i_k) = \mathcal{N}(\hat{x}^i_k, \Sigma^i_k).$$

D. Motion Plan and Control

We define a *motion plan* for robot i to be a tuple $(\check{u}^i, \check{x}^i, \check{C})$, where $\check{u}^i = (\check{u}^i_0, \check{u}^i_1, ..., \check{u}^i_{T-1}) \in (\mathcal{U}^i)^*$ is a nominal control trajectory; $\check{x}^i = (\check{x}^i_0, \check{x}^i_1, ..., \check{x}^i_T) \in (\mathcal{X}^i)^*$ is the nominal state trajectory obtained by propagation of the nominal dynamics

$\check{x}^i_{k+1} = A^i\check{x}^i_k + B^i\check{u}^i_k$ on $\check{u}^i$; and $\check{C} = (\check{C}_0, \check{C}_1, \ldots, \check{C}_{T-1})$ is a sequence of (measurement) matrices used for KF.

Robot i executes the motion plan $(\check{u}^i, \check{x}^i, \check{C})$ with the feedback control law $u^i_k = \check{u}^i_k - K^i(\hat{x}^i_k - \check{x}^i_k)$, with gain matrix $K^i \in \mathbb{R}^{m_i \times n_i}$, where $\hat{x}^i_k$ is the centralized KF state estimate obtained using the measurement matrix $\check{C}_k$. Note that (i) this controller stabilizes the robot about the nominal trajectory $\check{x}^i$, and (ii) the KF relies on exteroceptive measurements at time step k if $\check{C}_k$ requires them (the following section elaborates on the feasibility of this). Finally, note that these definitions of the motion plan and controller extend those in [18] to appropriately account for CL.

E. Probabilistic Objectives

Each robot i is assigned a goal region $\mathcal{X}^i_G \subset \mathcal{X}^i$ in its state space, which contains N_O (disjoint) static obstacles $\mathcal{X}^i_{O_j} \subset \mathcal{X}^i$, where $j \in \{1, \ldots, N_O\}$. Each of the N_A robots acts as a dynamic obstacle and sensor. The motion planning task is to compute a plan for each robot to reach its goal while avoiding collisions with both static and moving obstacles. Additionally, if the motion plan requires two robots to use exteroceptive measurements for CL at time step k, those robots must be within a distance r_{ext} of each other at that time step during the execution. Since the robots operate under uncertainty, all three requirements (goal satisfaction, collision avoidance, and CL) must be analyzed probabilistically.

The probability of robot i being in $\mathcal{X}^i_G$ at time k is $P^{i_k}_G = P(x^i_k \in \mathcal{X}^i_G) = \int_{\mathcal{X}^i_G} b(x^i_k)(s)\, ds$, where $b(x^i_k)(s)$ is the probability density function evaluated at s. Similarly, the probability of colliding with a static obstacle is $P^{i_k}_O = P(x^i_k \in \mathcal{X}^i_O) = \int_{\mathcal{X}^i_O} b(x^i_k)(s)\, ds$, where $\mathcal{X}^i_O = \bigcup_{j=1}^{N_O} \mathcal{X}^i_{O_j}$ is the union of all obstacle regions. Finally, the probability of colliding with another robot j is $P^{ij_k}_{\text{coll}} = P((x^i_k, x^j_k) \in \mathcal{X}^{ij}_{\text{coll}})$, where $\mathcal{X}^{ij}_{\text{coll}} = \{(x^i_k, x^j_k) \in \mathcal{X}^i \times \mathcal{X}^j \mid \mathcal{B}^i(x^i_k) \cap \mathcal{B}^j(x^j_k) \neq \emptyset\}$ is the set of states where the two robots' projected positions in the workspace overlap (collide).

For probabilistic analysis of CL, let $r^{ij}_k = \|\text{PROJ}_{\mathcal{W}}(x^i_k) - \text{PROJ}_{\mathcal{W}}(x^j_k)\|$ be the workspace distance between robots i and j at time k. Since x^i_k and x^j_k are random variables, r^{ij}_k is also a random variable distributed as $r^{ij}_k \sim b(r^{ij}_k)$. The probability that robots i and j fail to use their exteroceptive measurements at time step k is

$$P(r^{ij}_k > r_{ext}) = 1 - \int_0^{r_{ext}} b(r^{ij}_k)(s)\, ds. \tag{10}$$

If the motion plan requires CL, and it is unavailable during execution, this results in a failure. The probability of this failure must be captured in planning, as formalized below.

For two matrices D and E, let $D \sqsubset E$ denote that D is a submatrix of E. Then, the probability of failure of CL for robots i and j at time step k under a given motion plan with measurement matrix $\check{C}_k$ is defined as

$$P^{ij_k}_{\neg CL} = \begin{cases} P(r^{ij}_k > r_{ext}), & \text{if } C^{ij,ext}_k \sqsubset \check{C}_k, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Below, we state the safe CL-MRMP problem.

F. CL-MRMP Problem

Consider N_A robots with noisy dynamics in (1) and noisy measurements in (2) and (3), equipped with the (centralized) KF and feedback control law described in Secs. II-C and II-D. Given a set of initial distributions $\{x^i_0 = \mathcal{N}(\hat{x}^i_0, \Sigma^i_0)\}_{i=1}^{N_A}$, goal regions $\{\mathcal{X}^i_G\}_{i=1}^{N_A}$, and obstacles $\mathcal{X}_O$, and safety threshold p_{safe}, compute motion plan $(\check{u}^i, \check{x}^i, \check{C})$ for each robot $i \in \{1, \ldots, N_A\}$ from its initial state to goal region such that

$$P^{i_k}_O + \sum_{j=1, j\neq i}^{N_A} (P^{ij_k}_{coll} + P^{ij_k}_{\neg CL}) \leq 1 - p_{\text{safe}} \;\; \forall k \in \{1, .., T\}, \tag{12a}$$

$$P^{i_T}_G \geq p_{\text{safe}} \tag{12b}$$

where $P^{i_k}_O$, $P^{ij_k}_{coll}$, and $P^{ij_k}_{\neg CL}$ are defined in Sec. II-E. The safety requirements in (12) are known as *chance constraints*.

Approach Overview: This problem is challenging as it requires both motion planning and scheduling cooperative measurements for a set of uncertain robots. As discussed earlier, this requires assuming a centralized estimator to accurately track state correlations without losing information. Given this assumption, where the state of every robot is accessible, we adopt a centralized planning framework. Since the system is already represented as a single composed robot, a coupled planning approach is a natural first step toward effectively solving the CL-MRMP problem. Based on the lessons learned from this study, we can investigate a decoupled approach in future work.

Here, we propose a coupled planning framework that explicitly accounts for robot-robot correlations during planning. Our approach extends existing single-robot Gaussian belief planners by incorporating constraints on robot-robot collisions and exteroceptive measurement availability. We also introduce biasing techniques to enhance exploration of CL behaviors.

III. Sampling-Based Planner Framework

In this section, we detail a safety-guaranteed algorithm for CL-MRMP. Safety constraints are formulated with respect to the online belief $b(X^i_k)$ and thus conditioned on realized measurements. Without prior knowledge of the particular realization of $b(X^i_k)$, a motion plan cannot be guaranteed to satisfy the safety constraints. Our approach instead reasons over all possible online distributions by planning over the *expected* belief,

$$\mathbf{b}(X_k) = \mathbb{E}_Y[b(X_k|X_0, Y_{0:k})] = \int_{Y_{0:k}} b(X_k|X_0, Y_{0:k}) pr(Y_{0:k}) dY.$$

This guarantees that any execution of the returned plan satisfies the chance constraints. With known linear dynamics and measurement models with Gaussian noise, and the feedback control law of Sec. II-D, the expected belief is Gaussian distributed: $\mathbf{b}(X_k) = \mathcal{N}(\hat{X}_k, \Gamma_k)$. As shown in [2], the covariance $\Gamma_k = \Sigma_k + \Lambda_k$ can be calculated and evolved as sum of the online state uncertainty Σ_k inflated by the uncertainty due to a priori unknown measurements Λ_k.

The Belief-$\mathcal{A}$ planner [6] provides a framework for uncertain single agent motion planning using the expected belief

formulation with propagation according to [2]. In the following sections, we propose a sampling-based algorithm that adapts Belief-$\mathcal{A}$ for CL-MRMP. We detail specific adaptations for Belief-RRT and Belief-EST in Sec. III-A, propose efficient methods for checking the safety constraints in Sec. IV, and provide three different CL biasing methods in Sec. V.

A. Belief-$\mathcal{A}$ for CL-MRMP

We adapt the Belief-$\mathcal{A}$ framework for $\mathcal{A}$ = RRT [9] and $\mathcal{A}$ = EST [7] to solve the CL-MRMP problem. Both Belief-RRT and Belief-RRT build a search tree G consisting of nodes $\mathbb{V}$ and edges $\mathbb{E}$. Nodes are the beliefs $\mathbf{b}(X_k)$ (denoted $\mathbf{b}_k$), and edges ($\overline{\mathbf{b}_{k_1} \to \mathbf{b}_{k_2}}$) consist of the nominal control, state, and measurement: ($\check{U}$, $\check{X}$, $\check{C}$). The algorithm follows the same steps as the original RRT and EST algorithms (selection, extension, and validation), as presented in Algorithm 1.

Algorithm 1: Belief-$\mathcal{A}$ for CL-MRMP

Input: $\mathcal{X}$, $\{\mathcal{X}_G^i\}$, $\mathcal{W}_O$, N
Output: G

$G \leftarrow (\mathbb{V} \leftarrow \{b_0\}, \mathbb{E} \leftarrow \emptyset)$;
for N *iterations* **do**
  $\mathbf{b}_{select} \leftarrow$ SELECTBELIEF();
  $(\mathbf{b}_{new}, \overline{\mathbf{b}_{select} \to \mathbf{b}_{new}}) \leftarrow$ EXTENDBELIEF();
  if VALIDBELIEF($\boldsymbol{b}_{new}$) **then**
    $\mathbb{V} \leftarrow \mathbb{V} \cup \{\mathbf{b}_{new}\}$;
    $\mathbb{E} \leftarrow \mathbb{E} \cup \{\overline{\mathbf{b}_{select} \to \mathbf{b}_{new}}\}$;
return G

RRT selects a state by uniformly sampling the state space (UNIFORMSAMPLEBELIEF), then selecting the closest node to the sampled state (NEAREST). EST maintains a sparsity pdf over all the nodes in the tree, the selected node is sampled from this PDF (SPARSITYPDFSAMPLE). Both the generic RRT and EST algorithms can be made more efficient by biasing, e.g., classic goal biasing. We propose various methods to bias toward CL with rate ϵ in Sec. V: RRT is biased by modifying the sampled state (BIASEDSAMPLEBELIEF), whereas EST is biased by sampling from a pdf (BIASEDPDFSAMPLE).

Algorithm 2: SELECTBELIEF-RRT()

Input: $\mathcal{X}$, ϵ
Output: $\mathbf{b}_{select}$

$p \leftarrow$ STANDARDUNIFORMSAMPLE();
if $p < \epsilon$ **then**
  $\mathbf{b}_{sample} \leftarrow$ BIASEDSAMPLEBELIEF();
else
  $\mathbf{b}_{sample} \leftarrow$ UNIFORMSAMPLEBELIEF();
$\mathbf{b}_{select} \leftarrow$ NEAREST($G, \mathbf{b}_{sample}$);
return $\mathbf{b}_{select}$

The belief validation function VALIDBELIEF checks for collisions with each obstacle and each other robots. Note that exactly checking the safety constraint requires integration over the belief, which is generally intractable, suitably efficient approximations are described in Sec IV.

The EXTENDBELIEF function uses the belief propagation equations of [2], but with the measurement matrix constructed to respect the CL chance constraint, i.e. C_k is constructed only from the $C_k^{ij,ext}$ for robots i,j that satisfy the constraint on (11). As with checking the probabilistic safety constraint, exact evaluation of the CL constraint is intractable, we provide a detailed description of an efficient implementation in Section IV with the EXTENABLED function.

Algorithm 3: SELECTBELIEF-EST()

Input: $\mathcal{X}$, pdf_G, ϵ
Output: $\mathbf{b}_{select}$

$p \leftarrow$ STANDARDUNIFORMSAMPLE();
if $p < \epsilon$ **then**
  $\mathbf{b}_{select} \leftarrow$ BIASEDPDFSAMPLE();
else
  $\mathbf{b}_{select} \leftarrow$ SPARSITYPDFSAMPLE();
return $\mathbf{b}_{select}$

Under the theoretical results in [6], Belief-$\mathcal{A}$ inherits the completeness properties of the underlying algorithm (RRT and EST in our case), therefore our algorithm is probabilistically complete. Correctness is preserved by correct evaluation of the chance constraints in the VALIDBELIEF and EXTENABLED functions, which is discussed in Section IV. If VALIDBELIEF and EXTENABLED use conservative approximations, then our algorithm is complete only with respect to the approximation.

IV. EFFICIENT VALIDATION OF CHANCE CONSTRAINTS

Validation ensures correctness by requiring that any node added to the tree satisfies the chance constraints. Because this operation is called in every planning iteration it must be efficient. As discussed earlier, exact evaluation of the chance constraints is intractable. In this section, we provide suitably efficient and conservative approximations.

For all of the methods described in the subsequent sections we rely on the expedient of probability contours rather than exact integration of probability mass. For a random variable $z \in \mathbb{R}^n$ that is Gaussian distributed as $z \sim \mathcal{N}(\mu, \Sigma)$, the contour containing p probability mass is an ellipse defined by the spectral decomposition of the covariance Σ and a scaling factor α calculated from the inverse χ^2 distribution.

A. Probability allocation

We allocate the safety probability from (12a) among robot-obstacle collision, robot-robot collision, and violation of the exteroceptive measurement condition as $1 - p_{\text{safe}} = p_{obs} + p_{rob} + p_{\neg CL}$, and correspondingly separate the safety constraint:

$$P_O^{i_k} \leq p_{obs}, \quad \sum_{j=1, j\neq i}^{N_A} P_{coll}^{ij_k} \leq p_{rob}, \quad \sum_{j=1, j\neq i}^{N_A} P_{\neg CL}^{ij_k} \leq p_{\neg CL}.$$

This formulation vastly simplifies checking for constraint violation by fixing a threshold for each type of violation.

B. Robot-Obstacle Collision Checking

Our validity checking uses the simplest and most computationally efficient collision checking method from [18]. The robot body $\mathcal{B}^i$ is bounded by a sphere $\mathcal{B}^i \subset \mathcal{S}_{rob}^i$; a safety contour is then defined containing p_{safe} probability mass for

each robot. That elliptical contour is then bounded by a sphere which is inflated by the radius of $\mathcal{S}^i_{rob}$. If the inflated sphere does not intersect with any obstacles, the obstacle chance constraint is satisfied. We directly apply this method with the marginal $\mathbf{b}(x^i_k)$. The simplicity of this method is well-suited to the complex CL-MRMP problem, especially with the more expensive biasing methods in Sec. V.

C. Robot-Robot Collision Checking

For the next two sections, we consider the joint distribution $\mathbf{b}(X_k)$ to leverage information from correlation, and use the difference $\mathrm{x}^{ij}_k = \text{PROJ}_{\mathcal{W}}(x^i_k) - \text{PROJ}_{\mathcal{W}}(x^j_k)$, which is distributed as $\mathbf{b}(\mathrm{x}^{ij}_k) = \mathcal{N}(\hat{\mathrm{x}}^{ij}_k, \sigma^{ij}_k)$, with mean $\hat{\mathrm{x}}^{ij}_k = \hat{\mathrm{x}}^i_k - \hat{\mathrm{x}}^j_k$ and covariance $\sigma^{ij}_k = \Sigma^i_k + \Sigma^j_k - 2\Sigma^{ij}_k$. The term Σ^{ij}_k captures the off-diagonal terms of Σ_k correlating robot i's and j's Euclidean state estimates. Note x^{ij}_k is not the *distance* r^{ij} from Sec. II-E, which is nonlinear in x^i_k and therefore non-Gaussian.

Define the bounding spheres for each robot $\mathcal{B}^i \subset \mathcal{S}^i_{rob}$ and $\mathcal{B}^j \subset \mathcal{S}^j_{rob}$, with radii r_i and r_j respectively. Collision is then determined as $(\mathrm{x}^i_k, \mathrm{x}^j_k) \in \mathcal{X}^{ij}_{coll} \iff \|\mathrm{x}^i_k - \mathrm{x}^j_k\| \leq r_i + r_j$, and thus the set of collision states is a sphere $\mathcal{R}_r$ of radius $r_i + r_j$ centered on the origin in x^{ij}_k space. The collision probability is the intractable integral $P((\mathrm{x}^i_k, \mathrm{x}^j_k) \in \mathcal{X}^{ij}_{coll}) = \int_{\mathcal{R}_r} \mathbf{b}(\mathrm{x}^{ij}_k)(s)ds$. Next define a probability ellipsoid on $\mathbf{b}(\mathrm{x}^{ij}_k)$ containing $1-p_{rob}$ probability mass, and bound it with sphere $\mathcal{S}_{p_{rob}}$ such that $P(\mathrm{x}^{ij}_k \in \mathcal{S}_{p_{rob}}) \geq 1 - p_{rob}$. If this sphere does not intersect with the ball $\mathcal{R}_r$, we can conclude that it is entirely subsumed by the excluded set $\mathcal{R}_r \subset \tilde{\mathcal{S}}_{p_{rob}}$, $\tilde{\mathcal{S}}_{p_{rob}} = \{\mathrm{x}^{ij}_k \notin \mathcal{S}_{p_{rob}}\}$, and therefore $P(\mathrm{x}^{ij}_k \in \mathcal{R}_r) \leq p_{rob}$. This validation procedure is presented in Alg. 4.

Algorithm 4: RobotRobotCollision

Input: $\mathbf{b}^{ij} = \mathcal{N}(\hat{\mathrm{x}}^{ij}, \sigma^{ij})$, r_i, r_j
$\lambda_{max} \leftarrow \text{MaxEigenvalue}(\sigma^{ij})$;
$\alpha \leftarrow \text{inv}\chi^2(p_{rob}, 2)$;
$r_{rob} \leftarrow \sqrt{\alpha\lambda_{max}}$;
if $\|\hat{x}^{ij}\| - r_{rob} > r_i + r_j$ **then**
  return True;
return False

Theorem 1. *The validity checking Alg. 4 guarantees the satisfaction of the robot-robot collision constraint $P^{ij_k}_{coll} \leq p_{rob}$ if it returns True.*

Proof: Under the definition of the contour $\mathcal{S}_{p_{rob}}$ it holds that $P(\mathrm{x}^{ij}_k \in \mathcal{S}_{p_{rob}}) \geq 1 - p_{rob}$, and conversely $P(\mathrm{x}^{ij}_k \notin \mathcal{S}_{p_{rob}}) < p_{rob}$. It follows that if the sphere $\mathcal{R}_r$ containing all possible robot-robot collision states does not intersect with $\mathcal{S}_{p_{rob}}$, then it is subsumed by the excluded set, and therfore $P^{ij_k}_{coll} = P(\mathrm{x}^{ij}_k \in \mathcal{R}_r) < P(\mathrm{x}^{ij}_k \notin \mathcal{S}_{p_{rob}}) < p_{rob}$. ∎

D. CL Condition

Using the same variable x^{ij}_k as the prior section, we define a sphere centered on the origin containing all states with exteroceptive measurements enabled such that $\mathcal{R}_{ext} = \{\mathrm{x}^{ij}_k \mid \|\mathrm{x}^i_k - \mathrm{x}^j_k\| \leq r_{ext}\}$. As with the prior section, we find the probability ellipsoid on $\mathbf{b}(\mathrm{x}^{ij}_k)$ containing $1-p_{\neg CL}$ probability mass and bound it with sphere $\mathcal{S}_{p_{\neg CL}}$ such that $P(\mathrm{x}^{ij}_k \in \mathcal{S}_{p_{\neg CL}}) \geq 1 - p_{\neg CL}$. If the sphere $\mathcal{R}_{ext}$ subsumes $\mathcal{S}_{p_{\neg CL}}$, then we can conclude that $P(\|\mathrm{x}_i - \mathrm{x}_j\| > r_{ext}) < p_{\neg CL}$. This validation procedure is presented in Alg. 5.

Algorithm 5: ExtEnabled

Input: $\mathbf{b}^{ij} = \mathcal{N}(\hat{\mathrm{x}}^{ij}, \sigma^{ij})$, r_{ext}
$\lambda_{max} \leftarrow \text{MaxEigenvalue}(\sigma^{ij})$;
$\alpha \leftarrow \text{inv}\chi^2(p_{rob}, 2)$;
$r_{prob} \leftarrow \sqrt{\alpha\lambda_{max}}$;
if $\|\hat{x}^{ij}\| + r_{prob} < r_{ext}$ **then**
  return True;
return False

Theorem 2. *The validity checking Alg. 5 guarantees satisfaction of the CL constraint $P^{ij_k}_{\neg CL} \leq p_{\neg CL}$ if it returns True.*

Proof: Under the definition of the contour $\mathcal{S}_{p_{\neg CL}}$ it follows $P(\mathrm{x}^{ij}_k \in \mathcal{S}_{p_{\neg CL}}) \geq 1 - p_{\neg CL}$, and conversely $P(\mathrm{x}^{ij}_k \notin \mathcal{S}_{p_{\neg CL}}) < p_{\neg CL}$. It follows that if the sphere $\mathcal{R}_{ext}$ containing all possible CL states subsumes $\mathcal{S}_{p_{\neg CL}}$, then the probability of not being in $\mathcal{R}_{ext}$ (no CL) is: $P^{ij_k}_{\neg CL} = P(\mathrm{x}^{ij}_k \notin \mathcal{R}_{ext}) < P(\mathrm{x}^{ij}_k \notin \mathcal{S}_{p_{\neg CL}}) < p_{\neg CL}$. ∎

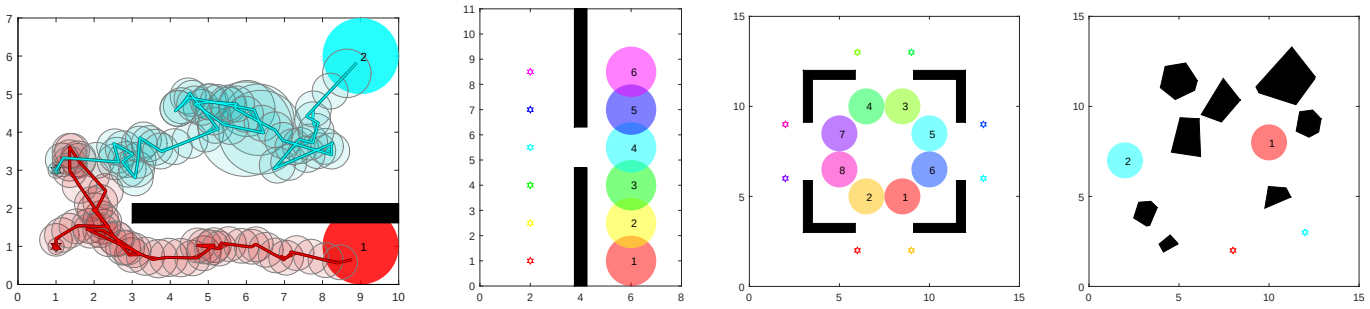
V. Biasing For CL

To encourage cooperative behaviors in our system, we preferentially select nodes with robot states where CL is enabled, i.e. nodes where the robots are close together. Three proposed biasing methods are detailed in this section. Note that biasing necessarily encourages unsafe behavior by increasing the likelihood of robot-robot collisions, thus raisingan interesting tension: more effective CL biasing will limit tree growth due to safety violation. This makes biasing effects difficult to predict, with inconsistencies dependent on the system and environment (as seen in Sec. VI).

A. State Cloning

The first method is implemented in RRT node selection (Alg. 2) with the BiasedSampleBelief function. The native selection scheme of RRT (sampling then finding the nearest neighbor) is not well suited to bias toward nodes that contain close robots because it relies on the distance *between* nodes, not the distance *within* nodes. Our biased sampler forms sampled beliefs where all robots occupy the same state. We call this 'Cloning'. Begin by sampling the composed mean, $\hat{X}_{samp}$, and covariance, Σ_{samp}, as usual. Then choose a robot to clone $i_{clone} \in \{1, ..., N_A\}$, with projected state $\hat{\mathrm{x}}^{i_{clone}}_k = \text{PROJ}_{\mathcal{W}}(\hat{x}^{i_{clone}})$. Our implementation alternates among all robots, but other methods, e.g. at random, are possible. Then, iterate over all robots replacing their projected state means with the cloned robot's. This gives a composed mean vector with all robots at the same projected state.

This is the most computationally efficient and scalable of the proposed biasing methods, being independent of tree size. This allows the planner to achieve roughly the same number



(a) Corridor (b) Pincer (c) Hive (d) Random

Fig. 2: Test Environments

of iterations within a fixed time regardless of ϵ. In contrast, the other two methods have a distinct trade-off as increased ϵ limits tree growth due to computational complexity.

B. Distance Weighting

The second biasing method is implemented in EST node selection (Alg. 3) with the BIASEDPDFSAMPLE function. Unlike indirect sampling and nearest neighbor selection in RRT, the likelihood of sampling a given node in EST is determined only by the node's weight. We can therefore directly sample according to the distance between robots within each node by maintaining a biasing pdf. For a node $\mathbf{b}^n = \mathcal{N}(\hat{X}_n, \Sigma_n)$ with projected Euclidean state $\hat{\mathrm{x}}^i = \text{PROJ}_{\mathcal{W}}(\hat{x}_n^i)$ for each robot, we define node weight $\mathcal{W}(\mathbf{b}^n)$ as:

$$\mathcal{W}(\mathbf{b}^n) = \frac{1}{\mathcal{D}(\mathbf{b}^n)}, \quad \mathcal{D}(\mathbf{b}^n) = \sum_{i=1}^{N_A} \sum_{j=1}^{N_A} \|\hat{\mathrm{x}}_i - \hat{\mathrm{x}}_j\|. \tag{13}$$

A proper PDF over all tree nodes is obtained by normalizing over the node weights. Note the double sum in $\mathcal{D}(\mathbf{b}^n)$ scales poorly with N_A, which limits tree growth.

C. Re-Branching

Finally, we propose a more complex biasing technique for both RRT and EST that modifies the selected node. This method reshuffles individual robot pairings to form new branches from existing branches; we call this 'Re-branching'. This method slightly modifies the sampling-based planner framework, as described in Alg. 6.

Algorithm 6: Sampling-Based Planner, Re-branch

Input: $\mathcal{X}$, $\{\mathcal{X}_G^i\}$, $\mathcal{W}_O$, N, ϵ
Output: G

$G \leftarrow (\mathbb{V} \leftarrow \{\mathbf{b}_0\}, \mathbb{E} \leftarrow \emptyset)$;
for N *iterations* **do**
  $\mathbf{b}_{select} \leftarrow$ SELECTBELIEF();
  $p \leftarrow$ STANDARDUNIFORMSAMPLE();
  if $p < \epsilon$ **then**
    $\mathbf{b}_{select} \leftarrow$ REBRANCH(b_{select});
  $(\mathbf{b}_{new}, \overline{\mathbf{b}_{select} \rightarrow \mathbf{b}_{new}}) \leftarrow$ EXTENDBELIEF();
  if VALIDBELIEF($\boldsymbol{b}_{new}$) **then**
    $\mathbb{V} \leftarrow \mathbb{V} \cup \{\mathbf{b}_{new}\}$;
    $\mathbb{E} \leftarrow \mathbb{E} \cup \{\overline{\mathbf{b}_{select} \rightarrow \mathbf{b}_{new}}\}$;
return G

We start with the selected node $\mathbf{b}_{k^\dagger}^{sel}$, defining a trajectory in belief space terminating at $\mathbf{b}_{k^\dagger}^{sel}$ at time $k^\dagger$. A single target robot $i_{target} \in \{1, ..., N_A\}$ is chosen, with projection of the mean from $\mathbf{b}_{k^\dagger}^{sel}$ into Euclidean space $\hat{\mathrm{x}}_{k^\dagger}^{i_{target}}$ (similar to Cloning). We then search the entire motion tree for the robot whose projected Euclidean state is closest to $\hat{\mathrm{x}}_{k^\dagger}^{i_{target}}$ at time $k^\dagger$; this is the new paired robot i_{pair} which corresponds to belief node $\mathbf{b}_{k^\dagger}^{pair}$. We then form a new branch by replacing the nominal control edges for robot i_{pair} in the original coupled trajectory for $\mathbf{b}_{k^\dagger}^{sel}$ and re-propagating the trajectory. This results in the new coupled belief node $\mathbf{b}_{k^\dagger}^{re}$ where robots i_{target} and i_{pair} have the closest possible Euclidean distances at time $k^\dagger$ of the existing tree states.

Re-branching is conceptually straightforward and offers promising results, particularly for smaller problems. However, its implementation presents challenges that impact scalability due to three key factors. First, it requires re-propagating and validating new branch edges. Second, it demands precise time synchronization of nodes to form new states, introducing additional states whenever the intermediate nodes of the selected and paired nodes differ in time. Third, we must search the tree over individual robot states rather than the *coupled* states of the motion tree. We address this by maintaining nearest neighbor data structures over the projected states for each robot. While these factors introduce computational overhead for large search trees, our results demonstrate the potential of Re-branching to efficiently handle smaller-scale problems.

VI. EVALUATIONS

We evaluate our algorithm across four environments shown in Fig. 2 to assess performance across different planners and biasing techniques. Robot start locations (stars) and goal regions (shaded circles) are predefined. We implemented the planners (Belief-RRT and -EST) in OMPL [17] and ran all experiments on an Intel Core i7-12700K CPU with 32GB RAM.

In all cases, each robot has 2D dynamics given by: $A_k^i = B_k^i = I_{2\times2}$, $Q_k^i = 0.01 I_{2\times2}$. A subset of robots (corresponding to odd indices i) has access to proprioceptive measurements, making the system unobservable in the absence of exteroceptive measurements. Primary results are reported with two robots, scaling results are reported up to six robots, with additional results in the Appendix. The measurement model is given by $C_k^{i,prop} = I_{2\times2}$ and $C_k^{ij,ext} = [I_{2\times2}, -I_{2\times2}]$. The

chance constraints are set to $p_{obs} = p_{rob} = p_{\neg CL} = 0.05$.

A. Illustrative Examples

Figs. 1a and 2a show two example trajectories for the Random and Corridor environments, respectively. The cyan robot states are unobservable without CL, as shown in Fig. 1b. In Fig. 1a the red robot must divert to enable CL so that the cyan robot reaches its goal. In Fig. 2a, the cyan robot diverts to remain close to the red robot throughout its trajectory and keep its uncertainty small. An online method that only drives toward the goal would fail to find these cooperative behaviors, and a decoupled approach that does not account for CL would fail to plan for the An online method that only drives toward the goal would fail to find these cooperative behaviors, and a decoupled approach that does not account for CL would fail to plan for the cyan robot. Our algorithm finds both plans within 2 minutes.

B. Benchmarks

We use benchmarks to compare Belief-RRT and Belief-EST planners with biasing rates (ϵ) from 0.01 to 0.5, alongside a no-biasing baseline. Each instance is run 50 times, with a planning time of 1 minute for 'Hive' and 2 minutes for others. We report success rates, computation time, and iteration counts, summarized in Figs. 3 and 4, where all plots follow the same legend.

Overall, the results show that our proposed algorithm reliably finds solutions in each environment with both RRT and EST variants, and biasing techniques better suited to some over others. In particular, we observe that Re-branching can improve planning time and iterations in simple environments, and RRT with Cloning scales well with N_A. Detailed discussion is provided below.

A Note on the Extended Results: Our main results are summarized in Figs. 3 and 4, however we include all our additional results in the Appendix. These include additional plots of the robot-robot collision rate, robot-obstacle collision rate, and tree size for the Random environment in Fig. 5, as well as the Hive environment for 2, 3, and 4 robots in Figs. 6, 7, and 8. We include abbreviated results for 5 and 6 robots in the Hive environement, Figs. 9 and 10. We additionally tested the Hive environment with 7 and 8 robots, but the success rate was too low to draw any useful conclusions. We also include plots of the success rate, solution time, and number of iterations for the Pincer environment for 2, 3, 4, and 5 robots in Figs. 11, 12, 13, and 14. We additionally tested the Pincer environment with 6 and 7 robots, but the success rate was too low to draw any useful conclusions.

CL vs. Robot-robot collision: To illustrate the tension introduced between CL and robot-robot collisions, consider the boxplots of the proportion of nodes rejected for robot-obstacle collision (Fig. 4a) and robot-robot collision (Fig. 4b) for the Random environment. As ϵ increases, robots are drawn closer together, causing more robot-robot collisions. This is particularly true of Re-branching, indicating that (despite the high computation cost) it is the most effective CL biasing method. We can see further evidence of this in the simpler 2, 3, and 4-robot Hive environments, Figs. 6, 7, 8, however the collision rates for the Hive are more robust to CL biasing compared to the Random environment, resulting in higher success rates for Re-branching.

Re-branching: Re-branching is an effective biasing technique in that it can efficiently find a solution for simple cases. In particular, Re-branching in the Hive environment shows a substantial decrease in solution times (Fig. 3e) and number of iterations (Fig. 3f) from the baseline compared to all other techniques for the same success rate (Fig. 3d). However, because Re-branching is computationally intensive its performance can drop rapidly as more robots are added or for complex environments. We can see this in the 2-robot Pincer environment Fig. 11, where the computation time for EST with Re-branching drastically increases and the success rate plummets. We can also see that Re-branching results in larger trees for RRT in the Random and 3-robot Hive environments, see Figs. 5, 7, which likely compounds the computation inefficiency and results in especially low success rates in those environments. This indicates that a more efficient re-branching method could be a promising future direction.

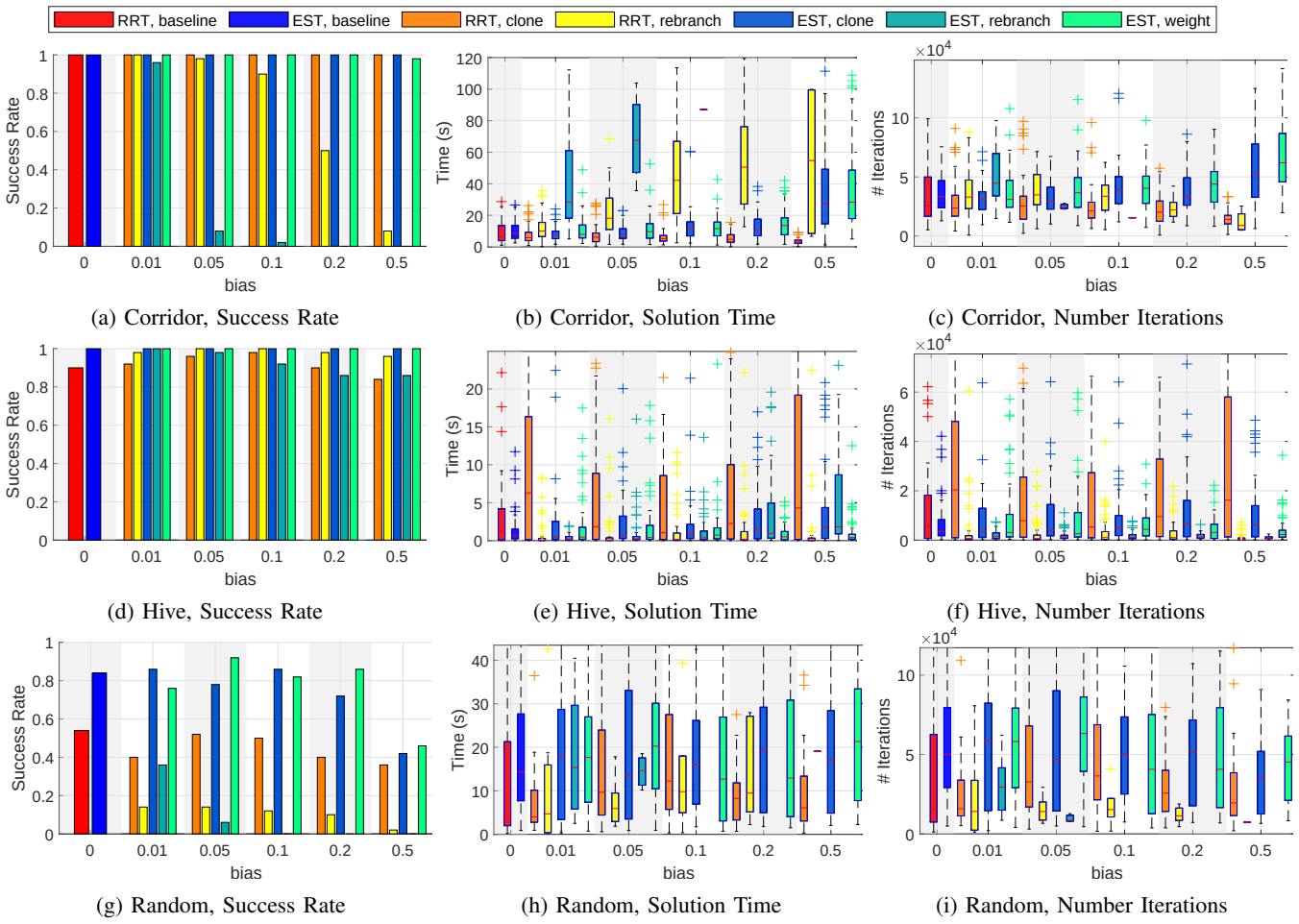
Cloning: RRT with Cloning performs well in the Pincer environment for larger N_A, increasing the success rate from 0.86 to 0.68 for four robots, and 0.04 to 0.5 for five robots (Fig. 4c). This indicates that the simplicity of the Cloning method can be quite effective for larger robot teams. Also note that while EST generally does worse than RRT in the Pincer environment, for the three robot case EST with Cloning increases the success from from 0.76 to 0.98 while significantly decreasing the solution time and number of robots, as seen in Fig. 12 in the Appendix. However, all EST methods failed to find any plans for greater than three robots in the Pincer environment.

Distance Weighting: The Distance Weighting technique has the most ambiguous effect on performance, with no obvious advantages anywhere in Fig. 3. The increased computation of Weighting likely outweighs any benefits of the biasing technique, especially with larger N_A.

However, in the Random environment it seems to provide modest improvement in success rate, and does at least as well as Cloning, see Figs. 3g, 3h, 3i. It has a similar effect for three and four robots in the Hive environment, see Figs. 7 and 8 respectively in the Appendix.

This underperformance is unintuitive: Distance Weighting directly reasons over the distances between robots within a node, rather than indirectly biasing like the Cloning technique. It is possible that the increased computation of Weighting outweighs any benefits of the biasing technique, especially with larger numbers of robots (this can be seen in the high computation times of Weighting vs Cloning especially in Fig. 12).

Planner Type: Finally, note that EST generally performs better in the Random and Hive environments, while RRT performs better in the Pincer environment. We were able to find plans for 5 and 6 robots in the Hive environemnt with EST,

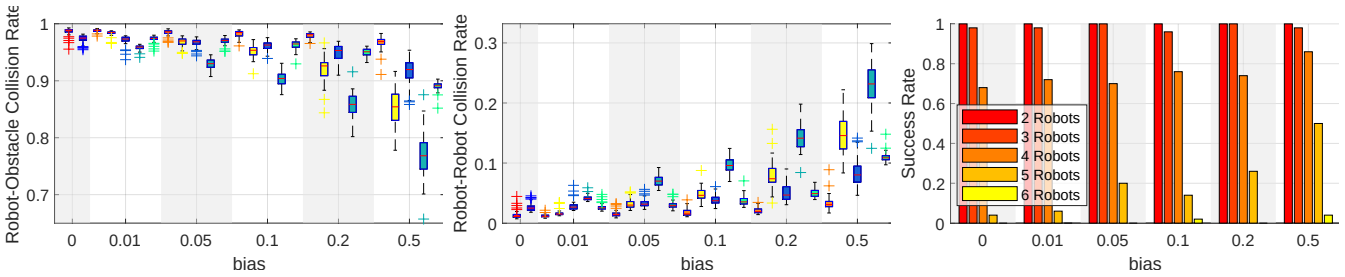


(a) Corridor, Success Rate (b) Corridor, Solution Time (c) Corridor, Number Iterations

(d) Hive, Success Rate (e) Hive, Solution Time (f) Hive, Number Iterations

(g) Random, Success Rate (h) Random, Solution Time (i) Random, Number Iterations

Fig. 3: Two Robot Environments



(a) Random, Robot-Obstacle Collision Rate (b) Random, Robot-Robot Collision Rate (c) Pincer, RRT Cloning Method

Fig. 4: Benchmarking results for (a)-(b) collision rates of 2 robots in Random Env., and (c) success rates for 2-6 robots in Pincer Env.

but not RRT, Figs. 9, 10, albeit with low success rates. We were able to find plans for 4 and 5 robots in the Pincer environment with RRT, but not EST, Figs. 13, 14. Both perform similarly well in the Corridor environment. This possibly due to the topology of the environments. RRT tends to pull tree growth to unexplored parts of the state space (via Voronoi biasing), while EST more exhaustively explores the space outward from the initial (via sparsity). In the Pincer environemnt RRT more successfully pulls the search tree through the passage and explores the goal side, whereas EST is slow to explore the entire space. On the other hand, the Hive and Random environments do not require extensive searches (especially the Hive), and therefore the exhaustive EST search more quickly finds satisfying plans. The Corridor environment is not particularly suited to either method of tree expansion.

VII. Conclusion

We consider the uncertain CL-MRMP problem and propose an algorithm that returns guaranteed safe plans to goal locations. Our proposed biasing techniques improve performance by encouraging exploration of cooperative behaviors, and we study their effectiveness in different scenarios. While our

algorithm does not scale well beyond five robots, it reliably finds plans in scenarios that require CL where online methods would fail. Future work will investigate ways to decouple planning to improve scalability.

REFERENCES

[1] Javier Alonso-Mora, Eduardo Montijano, Tobias Nägeli, Otmar Hilliges, Mac Schwager, and Daniela Rus. Distributed multi-robot formation control in dynamic environments. *Autonomous Robots*, 43(5):1079–1100, 2019.

[2] Adam Bry and Nicholas Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *IEEE Int'l Conf. on Robotics and Automation*, pages 723–730, 2011.

[3] Chao Gao, Guorong Zhao, and Hassen Fourati, editors. *Cooperative Localization and Navigation: Theory, Research, and Practice*. Taylor & Francis Group, LLC, Boca Rton, FL, 2020.

[4] Audrey Guillet, Roland Lenain, Benoit Thuilot, and Philippe Martinet. Adaptable robot formation control: Adaptive and predictive formation control of autonomous vehicles. *IEEE Robotics & Automation Magazine*, 21(1):28–39, 2014.

[5] Y.S. Hidaka, A.I. Mourikis, and S.I. Roumeliotis. Optimal formations for cooperative localization of mobile robots. In *in IEEE Int. Conf. on Robotics and Automation*, pages 4126–4131, 2005.

[6] Qi Heng Ho, Zachary N. Sunberg, and Morteza Lahijanian. Gaussian belief trees for chance constrained asymptotically optimal motion planning. In *Int. Conf. on Robotics and Automation*, 2022.

[7] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *Proceedings of International Conference on Robotics and Automation*, volume 3, pages 2719–2726 vol.3, 1997.

[8] Justin Kottinger, Shaull Almagor, and Morteza Lahijanian. Conflict-based search for multi-robot motion planning with kinodynamic constraints. In *IROS*, pages 13494–13499, 2022.

[9] S. Lavalle. Rapidly-exploring random trees : a new tool for path planning. *Research Report 9811*, 1998.

[10] Jürgen Leitner. Multi-robot cooperation in space: A survey. In *Advanced Technologies for Enhanced Quality of Life*, pages 144–151, 2009.

[11] Hamid Mokhtarzadeh and Demoz Gebre-Egziabher. Cooperative inertial navigation. *NAVIGATION*, 61(2):77–94, 2014.

[12] Aalok Patwardhan, Riku Murai, and Andrew J. Davison. Distributing collaborative multi-robot planning with gaussian belief propagation. *IEEE Robot. and Auto. Letters*, 8(2):552–559, 2023.

[13] Jorge Pomares, Leonard Felicetti, and Damiano Varagnolo. Editorial: Multi-robot systems for space applications. *Frontiers in Robotics and AI*, 10, 2023.

[14] Junqi Qu, Xinguang Li, and Gongwu Sun. Optimal formation configuration analysis for cooperative localization system of multi-auv. *IEEE Access*, 9:90702–90714, 2021.

[15] Stergios I. Roumeliotis. *Robust mobile robot localization: From single-robot uncertainties to multi-robot interdependencies*. PhD thesis, University of Southern California, 2000.

[16] James S. Russell, Mengbin Ye, Brian D. O. Anderson, Hatem Hmam, and Peter Sarunic. Cooperative localization of a gps-denied uav using direction-of-arrival measurements. *IEEE Transactions on Aerospace and Electronic Systems*, 56(3):1966–1978, 2020.

[17] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. https://ompl.kavrakilab.org.

[18] Anne Theurkauf, Justin Kottinger, Nisar Ahmed, and Morteza Lahijanian. Chance-constrained multi-robot motion planning under gaussian uncertainties. *IEEE Robotics and Auto. Letters*, 9(1):835–842, 2024.

[19] Ruben Van Parys and Goele Pipeleers. Online distributed motion planning for multi-vehicle systems. In *European Control Conf. (ECC)*, pages 1580–1585, 2016.

[20] Glenn Wagner and Howie Choset. Subdimensional expansion for multirobot path planning. *Artificial Intelligence*, 219:1–24, 2015.

APPENDIX

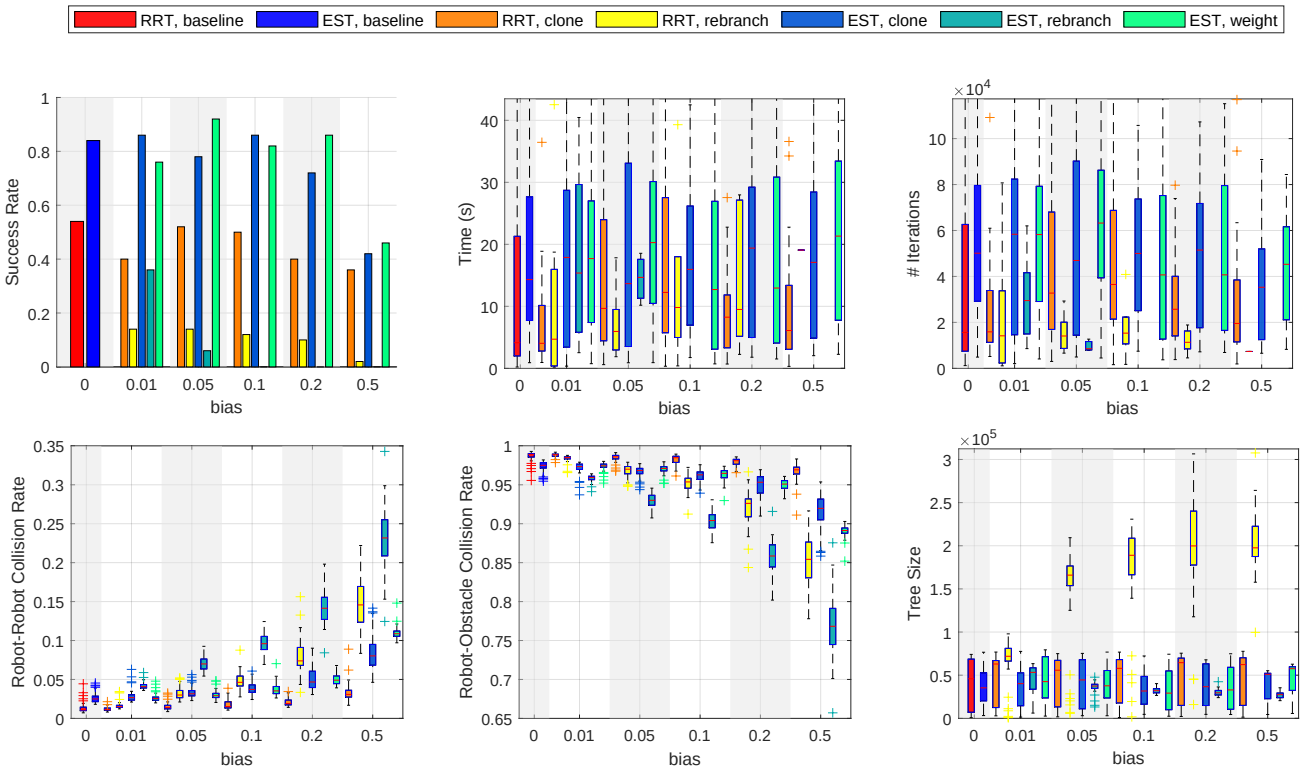


Fig. 5: Random Results

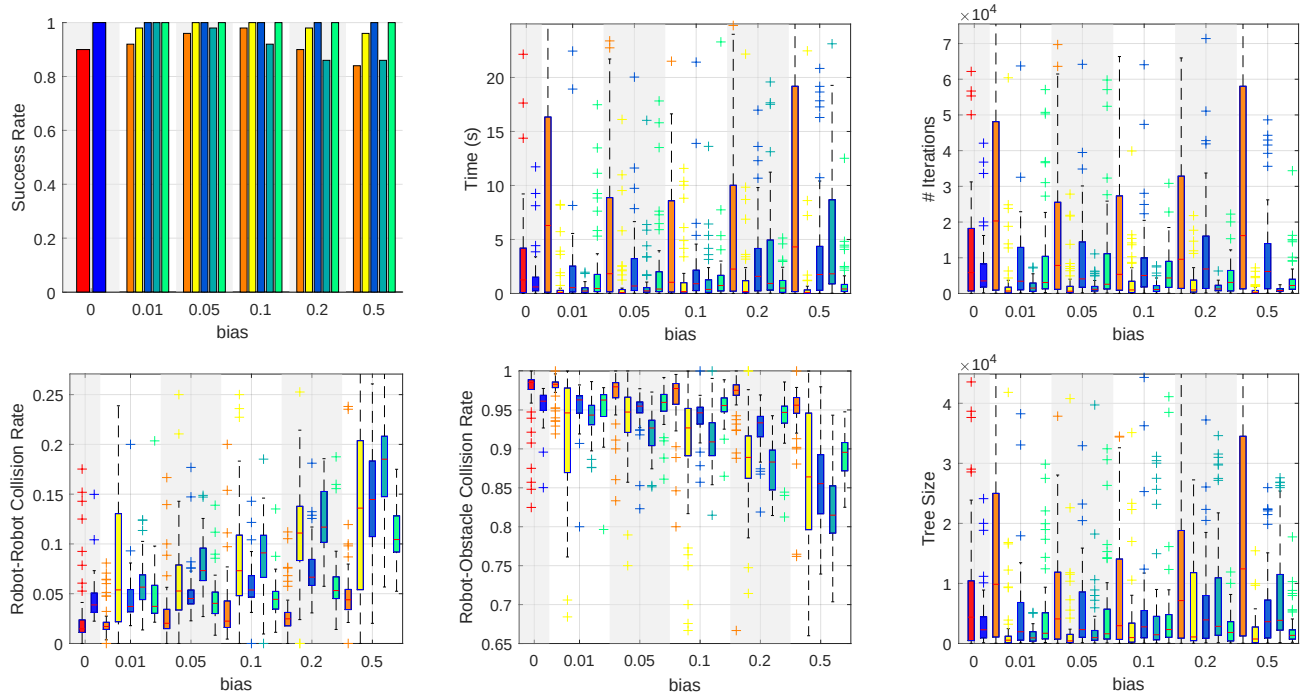


Fig. 6: Hive 2 Results

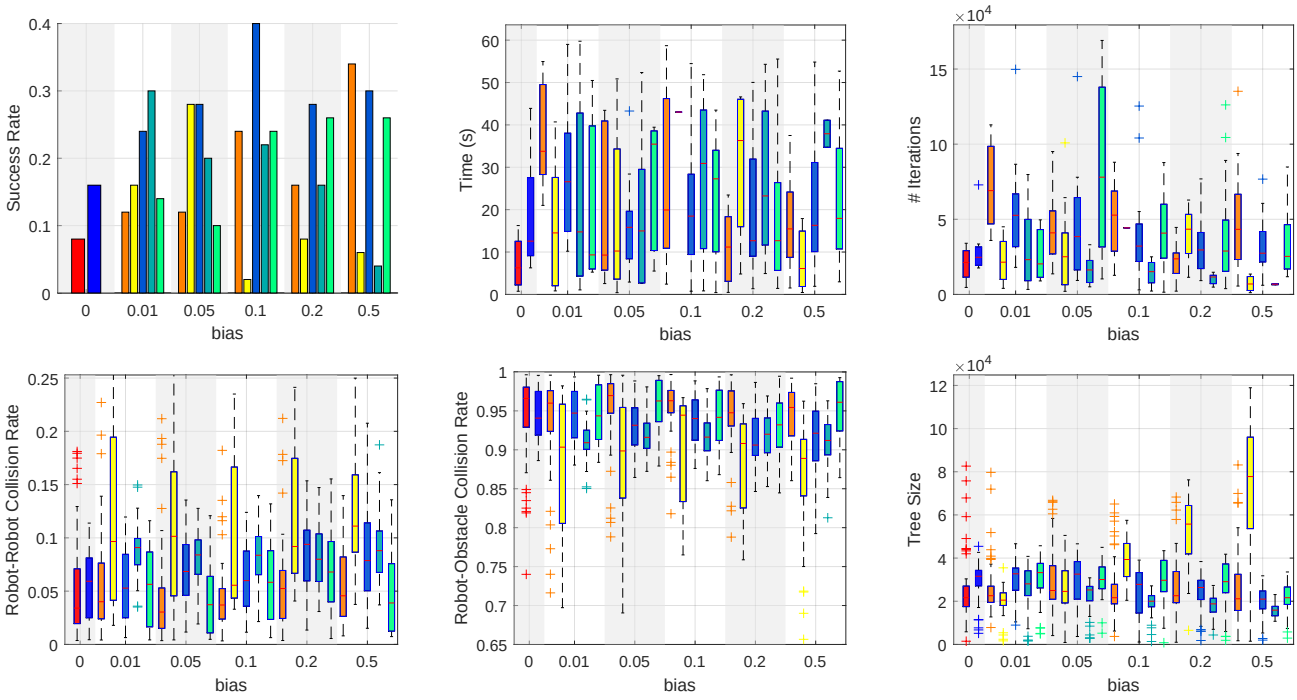


Fig. 7: Hive 3 Results

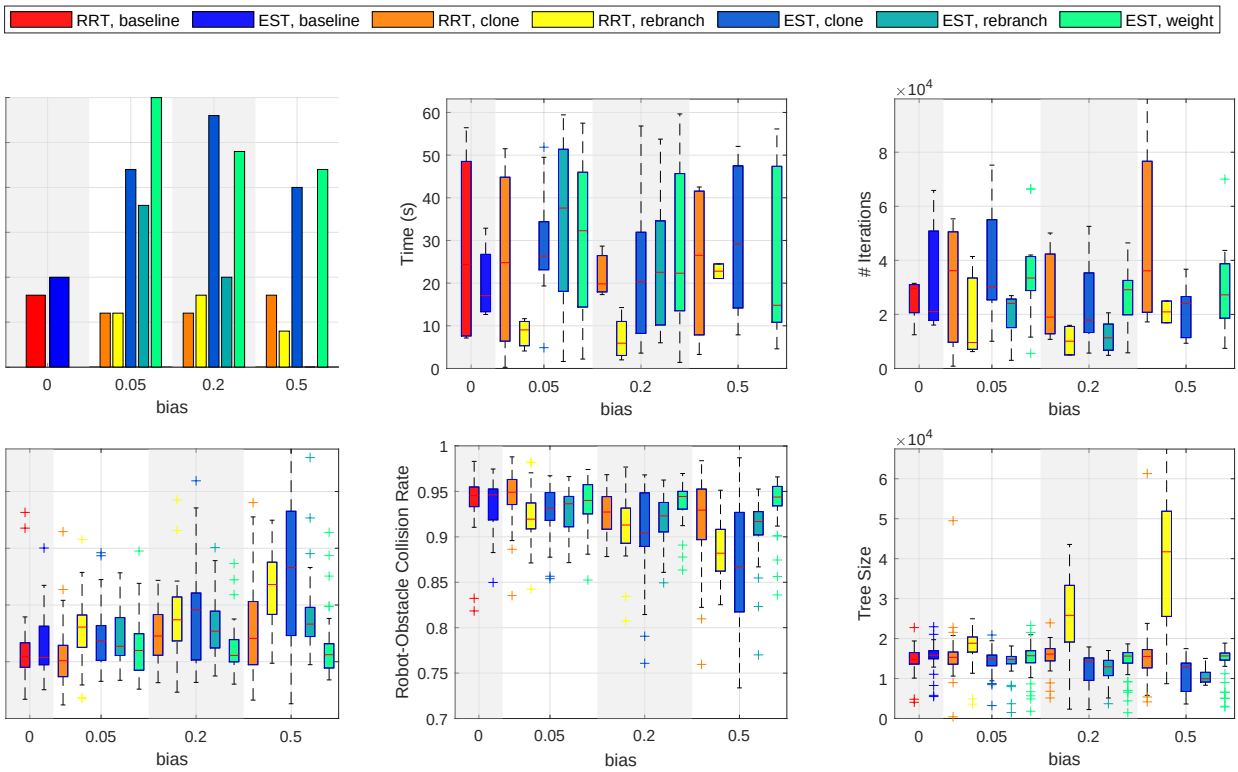


Fig. 8: Hive 4 Results

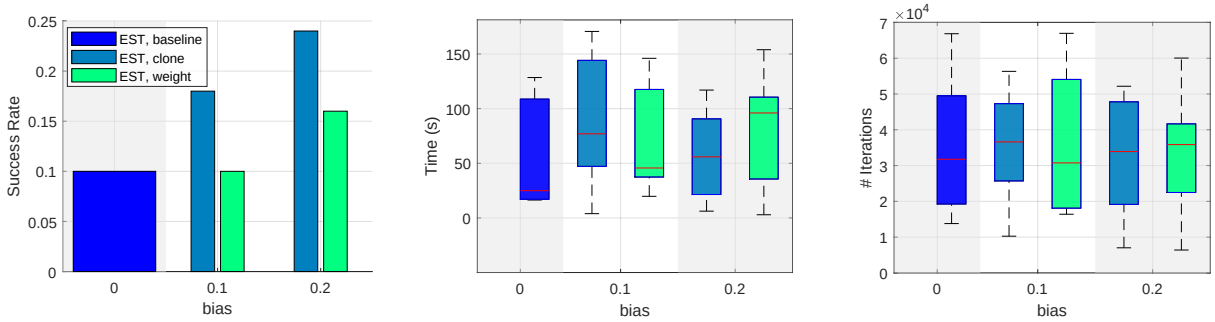


Fig. 9: Hive 5 Results

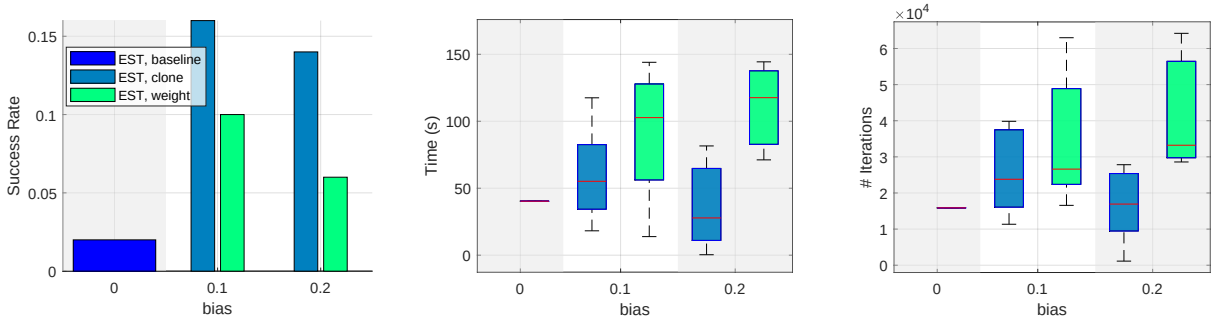


Fig. 10: Hive 6 Results

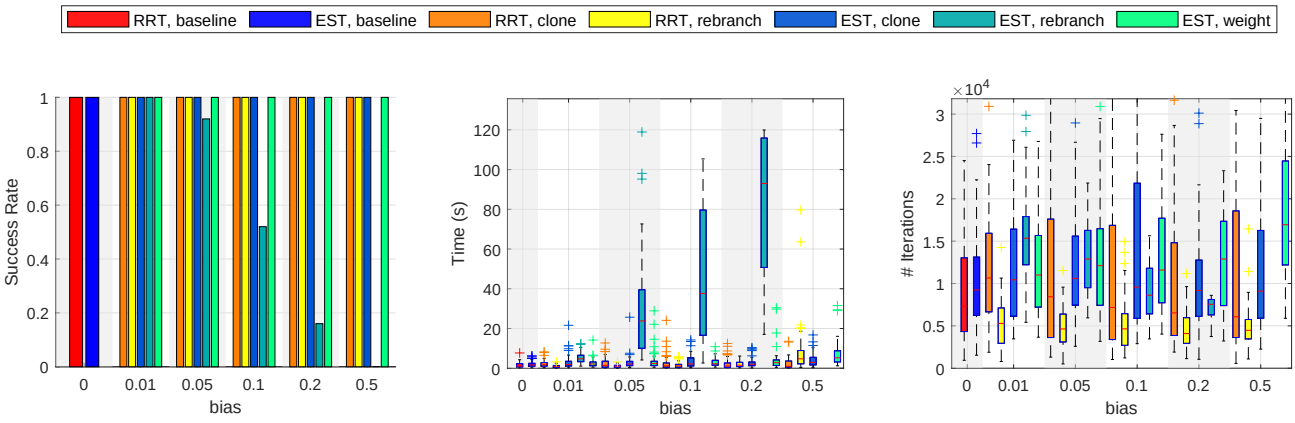


Fig. 11: Pincer 2 Results

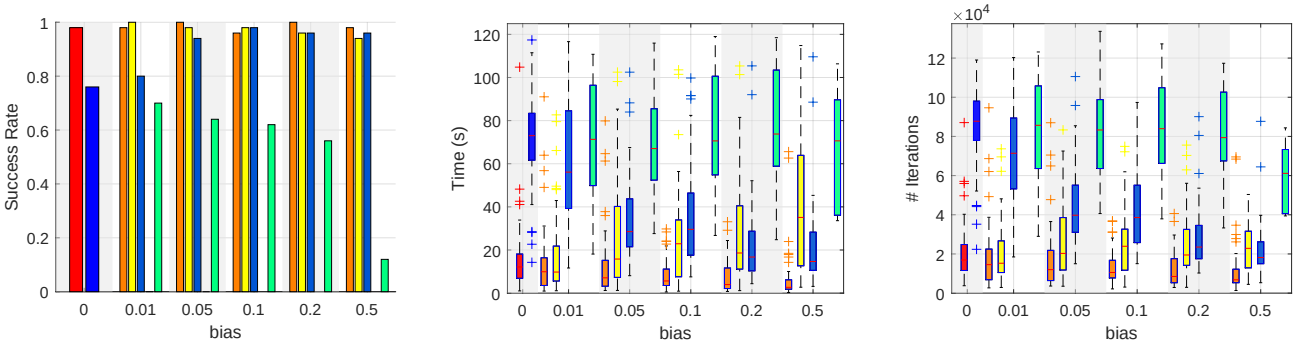


Fig. 12: Pincer 3 Results

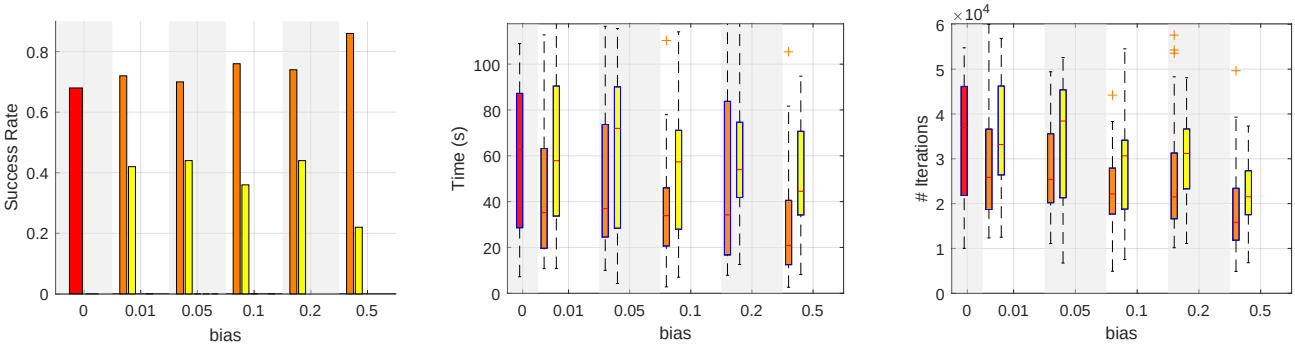


Fig. 13: Pincer 4 Results

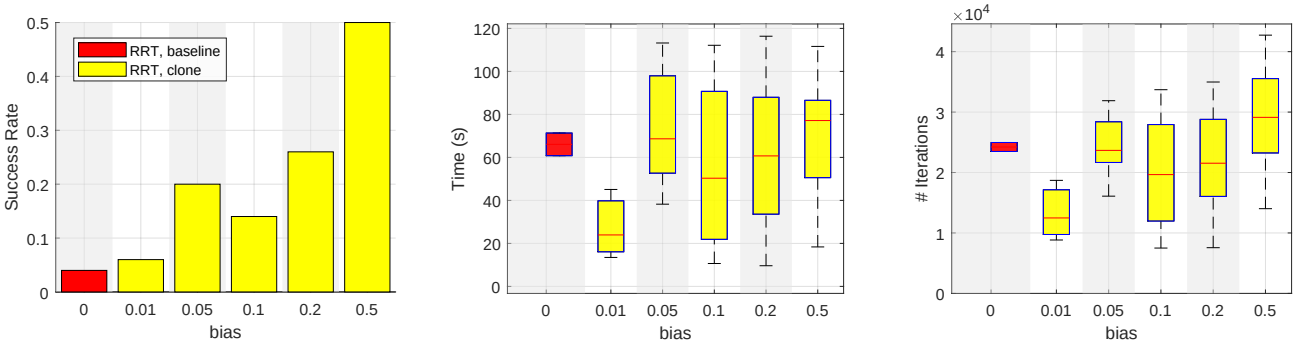


Fig. 14: Pincer 5 Results