Unified Autonomy Stack via Gaussian Belief Propagation

Kerem Kılıç
ETH Zürich
D-ITET
Zürich, Switzerland
kkilic@ethz.ch

Oğuz Kaan Gürsoy
ETH Zürich
D-MAVT
Zürich, Switzerland
oguersoy@ethz.ch

Vikram Iyer
ETH Zürich
D-ITET
Zürich, Switzerland
viiyer@ethz.ch

***Abstract*—We present a unified probabilistic pipeline for autonomous robotic systems that integrates planning, control, estimation, and sensing through Gaussian Belief Propagation (GBP). Unlike traditional modular architectures, our framework formulates these core components as a joint-inference problem on a factor graph. The GBP-based approach supports multi-agent settings, distributed computation, and naturally incorporates measurement noise, dynamic constraints, and collision avoidance. This work highlights the potential of GBP as a unifying inference engine for general-purpose autonomy. The GitHub repository containing the implementation associated with this paper is available at: github.com/VikramIyr/UniGBP.**

***Index Terms*—Gaussian Belief Propagation, Factor graphs, Autonomous Systems, Distributed Control**

I. Introduction

Autonomous systems are increasingly central to real-world applications, such as aerial logistics, warehouse automation, multi-robot exploration, and autonomous driving. These systems must reliably operate in dynamic and uncertain environments, planning trajectories, executing control actions, fusing sensor data, and coordinating with other agents in real-time. As robotic platforms scale and become more collaborative, integrating planning, control, estimation, and sensing into a cohesive system becomes increasingly challenging.

Traditional architectures typically decompose autonomy into distinct modules; planning algorithms, controllers, estimators, and sensor fusion methods that communicate through clearly defined interfaces. Although individually effective, this modularity often leads to inefficiencies and performance degradation due to mismatches between modules, redundant computations, and latency introduced by inter-module communication. Additionally, centralized control approaches commonly employed in classical systems pose scalability challenges, especially when applied to multi-agent scenarios, where coordinated behavior is crucial but computationally demanding.

Our proposed approach fundamentally differs by formulating planning, control, estimation, and sensing as components of a unified probabilistic inference problem on a factor graph. Each robot's planned trajectory, state, control inputs, and sensor states are modeled as latent variables, while system dynamics, sensor measurements, task objectives, and inter-agent interactions are represented as interconnected local factors. Using Gaussian Belief Propagation (GBP) for inference allows distributed, scalable, and uncertainty-aware computation, naturally handling noise, constraints, and interactions between multiple robots. This integrated framework eliminates the inefficiencies associated with traditional modular designs, supports decentralized processing, and significantly simplifies system architecture by maintaining a unified and interpretable structure.

The rest of the paper is organized as follows: Section II presents a summary of related work. Section III introduces the concept of Gaussian Belief Propagation. Section IV presents the problem formulation and system model. Section V describes our unified GBP-based control and estimation architecture for multi-robot coordination. Section VI details experimental results in trajectory tracking and collision avoidance. Finally, Section VII concludes the paper, summarizing our main contributions, highlighting the key performance insights and outlining promising directions for future research.

II. Related Work

Gaussian Belief Propagation (GBP) has emerged as a powerful paradigm for distributed inference in robotics and spatial Artificial Intelligence (AI). Ortiz et al. [1] offers a visual introduction to GBP, demonstrating its convergence properties under arbitrary update schedules and highlighting why message passing rather than centralized works much better in the case of networked systems. This builds on the foundational role of factor graphs in robotics [8], as well as Davison's earlier SLAM work that emphasized real-time probabilistic inference [10].

The FutureMapping2 project [2] introduced GBP as a flexible backbone for spatial AI, proposing factor graph inference as a unifying structure for SLAM, planning, and control. Recent extensions apply GBP to deep models, enabling learning in structured probabilistic systems [3].

Several works use GBP directly for high-level decision making and coordination. Patwardhan and Davison [4] cast each robot's finite-horizon trajectory planning problem as a single Gaussian factor graph, then solve it via inter-robot

message passing, yielding collision-free plans without any centralized optimizer. Murai et al. [5], [9] extend this idea into a multi-layer framework where mapping, goal selection, and path planning co-evolve through asynchronous GBP, enabling exploration and consensus in dynamic teams. Related formulations for agent coordination using GBP have also been explored in [6], while broader unifying views are offered in [7].

These works convincingly demonstrate GBP's power for distributed trajectory planning and high-level task coordination. However, they largely generate open-loop plans or layered decision pipelines. In contrast, our approach embeds a low-level controller directly alongside planning variables in a single, distributed factor graph. By running belief propagation over this unified graph, we achieve simultaneous trajectory optimization, state estimation, closed-loop control, and sensor fusion—all in one real-time, peer-to-peer inference loop. This end-to-end integration closes the loop between high-level planning and low-level execution under uncertainty.

III. Gaussian Belief Propagation and Factor Graphs

In this section, we introduce the fundamental concepts of factor graphs and Gaussian Belief Propagation (GBP). Factor graphs provide a versatile and intuitive framework for representing complex joint probability distributions through the decomposition into local factors. For a detailed and visual explanation of the general GBP algorithm, we refer the interested reader to [1].

Factor Graphs. A *factor graph* is a bipartite graph with variable nodes x_i and factor nodes ϕ_j, where each factor encodes a local relationship, measurement, or prior over a subset of variables. The joint distribution is

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{j=1}^{|\Phi|} \phi_j\big(x_{\phi_j}\big), \tag{1}$$

where x_{ϕ_j} denotes the variables connected to factor ϕ_j, and Z is a normalization constant ensuring $p(\mathbf{X})$ sums to one. This factorization makes explicit which variables interact, yielding a sparse graph that is well suited to distributed inference.

When each factor represents a Gaussian potential—common in estimation and control—the energy associated with ϕ_j is quadratic:

$$\phi_j(x_{\phi_j}) = \exp\big(-E_j(x_{\phi_j})\big), \tag{2}$$

$$E_j(x_{\phi_j}) = \tfrac{1}{2}\,[\,z_j - h_j(x_{\phi_j})\,]^\top \Lambda_j\,[\,z_j - h_j(x_{\phi_j})\,], \tag{3}$$

where z_j is the observed measurement, $h_j(\cdot)$ a (possibly nonlinear) prediction function, and Λ_j the observation precision.

Belief Propagation. Belief Propagation (BP) is a generic, iterative message-passing algorithm on factor graphs. Each variable node x_i and factor node ϕ_j exchange messages summarizing their local information:

$$m_{x_i \to \phi_j}(x_i) \propto \prod_{k \in N(x_i)\setminus\{\phi_j\}} m_{\phi_k \to x_i}(x_i), \tag{4}$$

$$m_{\phi_j \to x_i}(x_i) \propto \int \phi_j(x_{\phi_j}) \prod_{n \in N(\phi_j)\setminus\{x_i\}} m_{x_n \to \phi_j}(x_n)\, dx \tag{5}$$

Here $N(\cdot)$ denotes the neighboring nodes in the graph. After sufficient iterations, the marginal belief at each variable is approximated by

$$b(x_i) = \frac{1}{Z_i} \prod_{j \in \mathcal{N}(x_i)} m_{\phi_j \to x_i}(x_i), \tag{6}$$

where Z_i normalizes the belief. BP leverages only local computations, making it attractive for distributed problems.

Gaussian Belief Propagation. In Gaussian factor graphs, all messages remain Gaussian and can be represented in *canonical form* by a precision matrix Λ and information vector η. The update from variable x_i to factor ϕ_j reduces to simple summations of incoming message parameters:

$$\Lambda_{x_i \to \phi_j} = \sum_{k \in \mathcal{N}(x_i)\setminus\{\phi_j\}} \Lambda_{\phi_k \to x_i}, \tag{7}$$

$$\eta_{x_i \to \phi_j} = \sum_{k \in \mathcal{N}(x_i)\setminus\{\phi_j\}} \eta_{\phi_k \to x_i}. \tag{8}$$

Factor-to-variable messages involve marginalizing out other variables, which for Gaussians can be computed in closed form by partitioned precision matrices and information vectors (see [1]).

This canonical-parameter approach avoids explicit density products and integrals, replacing them with matrix additions and Schur complements, and it naturally supports efficient sparse implementations.

Nonlinear Factor Linearization. Many real-world factors are nonlinear in their inputs. To handle these within GBP, each nonlinear factor is linearized at the current estimate $x^0_{\phi_j}$ via first-order Taylor expansion:

$$h_j(x_{\phi_j}) \approx h_j(x^0_{\phi_j}) + J_j\,(x_{\phi_j} - x^0_{\phi_j}), \quad J_j = \left.\frac{\partial h_j}{\partial x_{\phi_j}}\right|_{x^0_{\phi_j}}. \tag{9}$$

The linearized factor then yields an approximate Gaussian with

$$\Lambda^{(\phi_j)} \approx J_j^\top \Lambda_j\, J_j, \tag{10}$$

$$\eta^{(\phi_j)} \approx J_j^\top \Lambda_j \big[z_j - h_j(x^0_{\phi_j}) + J_j\, x^0_{\phi_j}\big]. \tag{11}$$

By iterating between message passing (in the linearized graph) and re-linearization at updated beliefs, GBP efficiently approximates inference in nonlinear models.

IV. Problem Formulation

We consider k mobile robots operating in a 2D environment. Let $\mathcal{S}_r = \{1, 2, \ldots, k\}$ the set of robot indices.

State and Dynamics. Without loss of generality, we define the dynamics of robot $i \in \mathcal{S}_r$

$$p_i(t) = \begin{bmatrix} p_i^x(t) \\ p_i^y(t) \end{bmatrix}, \quad v_i(t) = \begin{bmatrix} v_i^x(t) \\ v_i^y(t) \end{bmatrix}, \quad u_i(t) = \begin{bmatrix} u_i^x(t) \\ u_i^y(t) \end{bmatrix}, \tag{12}$$

which we stack into the 4-dimensional state $x_i(t)$. Its discrete-time evolution is

$$x_i(t+1) = A\, x_i(t) + B\, u_i(t).$$

Reference Trajectory vs. On-the-Fly Planning. We consider two variants of the task:

1) **Reference-Driven:** Each robot is provided with a predefined reference trajectory to follow:

   $$r = \{r(0),\, r(1),\, \ldots,\, r(N)\}, \quad r(t) \in \mathbb{R}^4.$$

   We incorporate explicit tracking factors $\| x_i(t) - r_i(t) \|$ into the factor graph for each robot i and timestep t.

2) **Full Planning:** No external reference trajectory is supplied. Instead, reference variables $r_i(t)$ are embedded directly within the factor graph and initialized with high uncertainty. These variables are optimized through GBP, guided by initial linear interpolations. Factors linking these reference variables and the robot states simultaneously enforce trajectory tracking and generate higher-level trajectories to track.

In both variants, collision avoidance is naturally integrated as factors that connect the states of different robots in the task.

Measurements. At each time step, robot i obtains

$$y_i(t) = H\, z_i(t) + n_i(t),$$

where H is a selection matrix of the noisy states, chosen to be the identity matrix I for generality. The noise $n_i(t) \sim \mathcal{N}(0, \Sigma_y)$ is Gaussian.

Objective. The objective for each robot i is to reach its assigned goal position $g_i \in \mathbb{R}^2$ by the final time horizon N. Specifically, robots must:

- Minimize deviations from their planned or reference trajectories,
- Minimize control efforts through appropriate penalties,
- Maintain safe separation distances to ensure collision avoidance,

all while operating under noisy sensor measurements.

V. Method

We formulate planning, control, estimation, and sensing as a single Gaussian inference problem on a factor graph. Each variable is represented by a node, and every modelling assumption (dynamics, objectives, measurements, inter-robot interactions) enters the graph as a quadratic factor. Solving the resulting least-squares problem with GBP yields the maximum-a-posteriori (MAP) estimate jointly over all unknowns.

Variables and Gaussian Priors. For every robot $i \in \mathcal{S}_r$ and discrete time step $k = 0, \ldots, N$ we instantiate four variable nodes

$$x_i(k) \in \mathbb{R}^4 \quad \text{(state)}, \tag{13}$$
$$u_i(k) \in \mathbb{R}^2 \quad \text{(control input)}, \tag{14}$$
$$y_i(k) \in \mathbb{R}^4 \quad \text{(sensor measurement)}, \tag{15}$$
$$r_i(k) \in \mathbb{R}^4 \quad \text{(reference / plan)}. \tag{16}$$

Each node is initialised with an independent Gaussian prior:

$$p(z) = \mathcal{N}(\mu_z, \Sigma_z), \tag{17}$$

where the diagonal covariances encode our confidence. Initial and goal states as well as their tracking factors are encoded with low covariance, i.e., high precision, while state, input and measurement uncertainties are left high for higher flexibility during optimization.

Factor Templates. All costs are written as squared-error residuals $\rho(z)$ weighted by the inverse covariance Σ^{-1}:

$$\|\rho(z)\|^2_{\Sigma^{-1}} = \rho(z)^\top \Sigma^{-1} \rho(z)$$

a) Dynamics factor: For every $k < N$ we couple successive states and the applied input

$$\rho_k^{\mathrm{d}} = x_i(k+1) - \big[x_i(k) + \Delta t\big(A x_i(k) + B u_i(k)\big)\big], \tag{18}$$
$$\Sigma_{\mathrm{d}} = \sigma_{\mathrm{d}}^2 I. \tag{19}$$

b) Reference–prior factor: To give GBP an initial guess, we pre-compute a seed path $\mathrm{ref}_i(k)$ via linear interpolation between the current state $x_i(0)$ and the goal g_i. We then include a loose quadratic prior

$$\begin{aligned} \rho_{i,k}^{\mathrm{ref}} &= r_i(k) - \mathrm{ref}_i(k), \\ \Sigma_{\mathrm{ref}} &= \sigma_{\mathrm{ref}}^2 I. \end{aligned} \tag{20}$$

with σ_{ref} chosen large except at $k = 0, N$ where the variance is tightened to anchor the endpoints.

c) Control effort factor:

$$\begin{aligned} \rho_k^{\mathrm{effort}} &= u_i(k), \\ \Sigma_{\mathrm{effort}} &= R^{-1}. \end{aligned} \tag{21}$$

d) Tracking factor:

$$\rho_k^{\text{track}} = x_i(k) - r_i(k), \quad \Sigma_{\text{track}} = Q^{-1}. \tag{22}$$

e) Measurement factor:

$$\rho_k^{\text{meas}} = y_i(k) - x_i(k), \quad \Sigma_{\text{meas}} = \sigma_{\text{meas}}^2 I. \tag{23}$$

f) Collision avoidance factor: For robots $i \neq j$ we penalise proximity below a safety radius r_* at every k:

$$\rho_{i,j,k}^{\text{col}} = \max\{0, 1 - \frac{p_i(k) - p_j(k)}{r^*}\}, \quad \Sigma_{\text{col}} = \sigma_{\text{col}}^2. \tag{24}$$

Online Execution Loop. At runtime we interleave physical roll-out with inference:

1) **Optimise controls**: Run n_{it} GBP iterations; extract the current MAP control $u_i(k)$ from the belief of $u_i(k)$.
2) **State propagation**: Apply $u_i(k)$ to the true plant: $x_i^{\text{true}}(k+1) = x_i^{\text{true}}(k) + \Delta t\big(Ax_i^{\text{true}}(k) + Bu_i(k)\big)$.
3) **Sensor update**: Sample a noisy measurement $y_i(k+1) = x_i^{\text{true}}(k+1) + w_k$, $w_k \sim \mathcal{N}(0, \Sigma_y)$, and update the prior and covariance for the measurement node $y_i(k+1)$.
4) **Belief update**: Perform another n_{it} GBP sweeps to absorb the fresh evidence and repeat.

The strictly local message updates guarantee that each robot exchanges information only with immediate neighbours, enabling fully distributed implementation with bounded communication overhead.

Formal Guarantees.

Theorem 1: [Inference $\implies$ RTS Smoother + LQR] Let $x_{k+1} = A_d x_k + B_d u_k, \quad k = 0, \dots H-1$ with $A_d = I + \Delta t A, B_d = \Delta t B$, and LQR weights $Q, R, S \succ 0$. Construct a single Gaussian factor graph over $z = \{y_k, r_k, x_k, u_k\}$ with factors $y_k - x_k \sim \mathcal{N}(0, \Sigma_y)$, $\quad r_k - x_k \sim \mathcal{N}(0, Q^{-1})$, $x_{k+1} - A_d x_k - B_d u_k \sim \mathcal{N}(0, \Sigma_d)$, $\quad u_k \sim \mathcal{N}(0, R^{-1})$, $r_k - \text{ref}_k \sim \mathcal{N}(0, Q^{-1})$, $\quad r_H - x_H \sim \mathcal{N}(0, S^{-1})$, where ref_k is the desired waypoint. Then the joint MAP on z is equivalent to (i) an RTS Kalman smoother on y_k followed by (ii) a finite-horizon LQR on smoothed $\hat{x}_k$.

Proof. The joint negative-log-posterior is the sum of all quadratic factors:

$$\begin{aligned}\mathcal{E}(z) = &\sum_{k=0}^{H} \|y_k - x_k\|^2_{\Sigma_y^{-1}} + \sum_{k=0}^{H} \|r_k - x_k\|^2_Q \\ &+ \sum_{k=0}^{H-1} \|x_{k+1} - A_d x_k - B_d u_k\|^2_{\Sigma_d^{-1}} \\ &+ \sum_{k=0}^{H-1} \|u_k\|^2_R + \sum_{k=0}^{H-1} \|r_k - \text{ref}_k\|^2_Q + \|r_H - \text{ref}_H\|^2_S.\end{aligned} \tag{25}$$

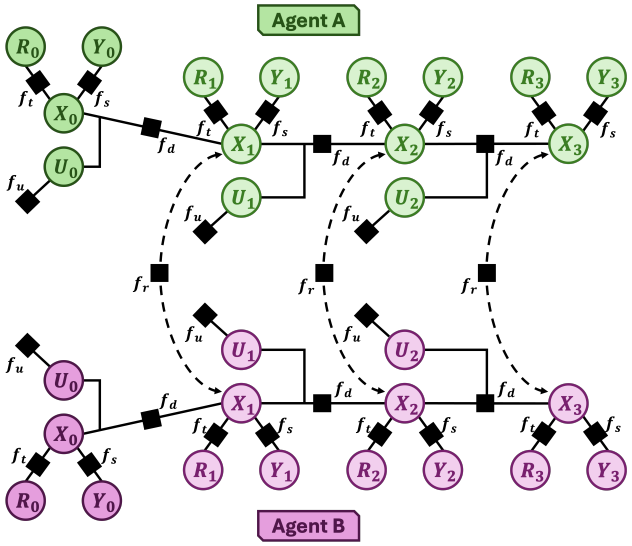

Fig. 1. **Factor graph representation for multi-agent unified autonomy stack involving two agents.** Each agent's trajectory is represented by discrete state variables across a time horizon. Factors depicted include per-agent dynamics factors f_d, tracking factors f_t and measurement factors f_s. Inter-agent collision avoidance factors f_r ensure safe distances between agents at corresponding timesteps. This unified factor graph structure enables simultaneous optimization of low-level control, planning, and state estimation via Gaussian Belief Propagation (GBP).

Where $\|y_k - x_k\|^2_{\Sigma_y^{-1}} = (y_k - x_k)^\top \Sigma_y^{-1} (y_k - x_k)$ is the precision weighted norm. GBP targets the single global minimiser:

$$\hat{z} = \arg\min_{y,r,x,u} \mathcal{E}(y, r, x, u). \tag{26}$$

For a strictly convex quadratic, that minimiser is unique and can be obtained by solving the linear system $\nabla\mathcal{E} = 0$. Algebraically, we may eliminate any subset of variables (via Schur complement or by setting their partial derivatives to zero) in any order; the solution for the remaining variables is unaffected. Hence :

$$\underbrace{\min_{y,r,x,u}}_{\text{full MAP}} = \min_{x,u}(\min_{y,r} \mathcal{E}) = \min_{y,r}(\min_{x,y} \mathcal{E}) \tag{27}$$

- **LQR Equivalence**: Marginalization of y_k, r_k gives a reduced cost that only depends on x_k, u_k. Each term $\|y_k - x_k\|^2$ and $\|r_k - x_k\|^2$ is a strictly convex quadratic with no coupling. Therefore their unique minimizers are obtained by:

$$\frac{\partial \mathcal{E}}{\partial y_k} = 2\,\Sigma_y^{-1}(y_k - x_k) = 0 \implies y_k^* = x_k,$$

$$\frac{\partial \mathcal{E}}{\partial r_k} = (r_k - x_k) + (r_k - \text{ref}_k) = 0 \implies r_k^* = \frac{1}{2}(x_k + \text{ref}_k),$$

Substituting these optimal values back into $\mathcal{E}$ yields the reduced cost:

$$\mathcal{E}(x,u) = \sum_{k=0}^{H-1} \|x_{k+1} - A_d x_k - B_d u_k\|^2_{\Sigma_d^{-1}} + \sum_{k=0}^{H-1} \|u_k\|^2_R + \sum_{k=0}^{H-1} \|x_k - \text{ref}_k\|^2_Q + \|x_H - \text{ref}_H\|^2_S. \tag{28}$$

Notice that Σ_d^{-1}, the precision matrix encoding the dynamics, is typically very large. In the limit $\Sigma_d^{-1} \to \infty$, each $\|x_{k+1} - A_d x_k - B_d u_k\|^2_{\Sigma_d^{-1}}$ enforces the equality $x_{k+1} = A_d x_k + B_d u_k$, becoming a hard constraint. The remaining cost is exactly the finite-horizon LQR objective:

$$\sum_{k=0}^{H-1} \|x_k - \text{ref}_k\|^2_Q + \sum_{k=0}^{H-1} \|u_k\|^2_R + \|x_H - \text{ref}_H\|^2_S. \tag{29}$$

- **RTS Smoother Equivalence:** Define $f_k(u_k)$ as the part of $\mathcal{E}(z)$ grouping terms depending on u_k:

$$f_k(u_k) = \underbrace{\|x_{k+1} - A_d x_k - B_d u_k\|^2_{\Sigma_d^{-1}}}_{(1)} + \underbrace{\|u_k\|^2_R}_{(2)}. \tag{30}$$

Writing (1) explicitly:

$$\begin{aligned}(1) &= (x_{k+1} - A_d x_k - B_d u_k)^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k - B_d u_k) \\ &= u_k^\top B_d^\top \Sigma_d^{-1} B_d\, u_k - 2\, u_k^\top B_d^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k) \\ &\quad + (x_{k+1} - A_d x_k)^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k).\end{aligned}$$

Then f_k can be written as:

$$\begin{aligned}f_k(u_k) &= u_k^\top (B_d^\top \Sigma_d^{-1} B_d + R) u_k \\ &\quad - 2 u_k^\top B_d^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k) + c_k\end{aligned}$$

Where $c_k = (x_{k+1} - A_d x_k)^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k)$ does not depend on u_k. Taking the partial derivative in u_k and setting equal to zero yields the unique minimizer:

$$\begin{aligned}\frac{\partial f_k}{\partial u_k} &= (B_d^\top \Sigma_d^{-1} B_d + R) u_k \\ &\quad - B_d^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k) = 0\end{aligned}$$

$$\begin{aligned}\implies u_k^* &= \underbrace{(B_d^\top \Sigma_d^{-1} B_d + R)^{-1}}_{H_u^{-1}} \underbrace{B_d^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k)}_{g_k} \\ u_k^* &= H_u^{-1} g_k\end{aligned} \tag{31}$$

Where H_u and g_k have been defined for ease of notation. We thus have:

$$\min_{u_k} f_k(u_k) = c_k - g_k^\top H_u^{-1} g_k$$

Expanding gives:

$$\begin{aligned}c_k - g_k^\top H_u^{-1} g_k &= (x_{k+1} - A_d x_k)^\top \Sigma_d^{-1} (x_{k+1} - A_d x_k) \\ &\quad - (x_{k+1} - A_d x_k)^\top \underbrace{\Sigma_d^{-1} B_d H_u^{-1} B_d^\top \Sigma_d^{-1}}_{=:X} (x_{k+1} - A_d x_k) \\ &= (x_{k+1} - A_d x_k)^\top \underbrace{\left[\Sigma_d^{-1} - X\right]}_{W} (x_{k+1} - A_d x_k).\end{aligned}$$

Where $W = \Sigma_d^{-1} - \Sigma_d^{-1} B_d (B_d^\top \Sigma_d^{-1} B_d + R)^{-1} B_d^\top \Sigma_d^{-1}$. Hence after marginalizing all $\{u_k\}$, the remaining cost is:

$$\begin{aligned}\mathcal{E}_{\text{marg}}(x,y,r) &= (x_{k+1} - A_d x_k)^\top W (x_{k+1} - A_d x_k) \\ &\quad + \sum_{k=0}^{H} \|y_k - x_k\|^2_{\Sigma_y^{-1}} + \sum_{k=0}^{H} \|r_k - x_k\|^2_Q \\ &\quad + \sum_{k=0}^{H-1} \|r_k - \text{ref}_k\|^2_Q + \|r_H - \text{ref}_H\|^2_S.\end{aligned} \tag{32}$$

Further eliminating y_k, r_k as in the first part of the proof, we obtain the the state-only reduced cost:

$$\begin{aligned}\mathcal{E}_{\text{state}}(x) &= \sum_{k=0}^{H-1} \|x_{k+1} - A_d x_k\|^2_W \\ &\quad + \sum_{k=0}^{H-1} \|x_k - \text{ref}_k\|^2_Q + \|x_H - \text{ref}_H\|^2_S.\end{aligned} \tag{33}$$

But $\mathcal{E}_{\text{state}}(x)$ is exactly the log-density, up to an additive constant, of the following linear-Gaussian system:

$$\begin{aligned}x_{k+1} &= A_d\, x_k + w_k, \quad w_k \sim \mathcal{N}(0, W^{-1}), \\ z_k &= x_k + v_k, \quad v_k \sim \mathcal{N}(0, Q^{-1}), \\ z_k &:= \text{ref}_k, \\ x_H &= \text{ref}_H + \varepsilon_H,\ \varepsilon_H \sim \mathcal{N}(0, S^{-1}).\end{aligned} \tag{34}$$

Note that the physics have not changed; w_k is the algebraic residual that remains after substituting the optimal input u_k^* back into the dynamics. Similarly, other terms translate remaining residuals after elimination of y_k, r_k. To solve for the MAP estimate of $\mathcal{E}_{\text{state}}(x)$, one can define $\Delta_k = x_{k+1} - A_d x_k$ and $d_k = x_k - \text{ref}_k$ and write the cost in matrix form:

$$\mathcal{E}_{\text{state}}(x) = \frac{1}{2} x^\top H x - \eta^\top x + \text{const} \tag{35}$$

Where:

$$\mathbf{H} = \begin{bmatrix} Q + A^\top W A & -A^\top W & & \\ -WA & Q + W + A^\top W A & \ddots & \\ & \ddots & \ddots & -A^\top W \\ & & -WA & W + S \end{bmatrix} \tag{36}$$

$$\eta = \begin{bmatrix} Q\text{ref}_0 \\ \vdots \\ Q\text{ref}_{H-1} \\ S\text{ref}_H \end{bmatrix}. \tag{37}$$

The MAP estimate $\hat{x}$ is the unique solution of the block-tridiagonal linear system $H\hat{x} = \eta$. The fastest way to solve this is:

- Forward block Gaussian elimination,
- Backward substitution.

From the first row of the block system we have

$$\big(Q + A^\top W A\big)\, x_0 \;-\; A^\top W\, x_1 \;=\; Q\,\mathrm{ref}_0.$$

Introduce the shorthand

$$K_{11} := Q + A^\top W A, \quad y_0 := K_{11}^{-1}\, Q\,\mathrm{ref}_0, \quad L_1 := W A\, K_{11}^{-1}.$$

With these definitions

$$x_0 = y_0 + L_1^\top x_1.$$

Substituting this expression into the next row eliminates x_0. Then for the general step k, the following recursion defines the forward pass, replacing the original system by a strictly lower-triangular one $L^\top x = y$:

$$\begin{aligned} L_k &:= W A\big(D_{k-1} + A^\top W A\big)^{-1}, \\ D_k &:= Q + W - L_k\, A^\top W, \\ \tilde{\eta}_k &:= Q\,\mathrm{ref}_k + L_k\, \tilde{\eta}_{k-1}, \\ y_k &:= D_k^{-1} \tilde{\eta}_k. \end{aligned} \tag{38}$$

The gain for the next pass is thus:

$$L_{k+1} := W A\big(D_k + A^\top W A\big)^{-1}. \tag{39}$$

Finally, the backward sweep is simply back-substitution:

$$x_k = y_k - L_{k+1}^\top x_{k+1} \tag{40}$$

The forward recursion is the information–form Kalman filter with $J_{k|k} = D_k,\ h_{k|k} = \tilde{\eta}_k$, because it performs the standard

$$\begin{aligned} &\text{predict: } (J, h) \mapsto (\bar{D}_k, \bar{\tilde{\eta}}_k), \\ &\text{update: } (\bar{D}_k, \bar{\tilde{\eta}}_k) \mapsto (D_k, \tilde{\eta}_k) \end{aligned}$$

using the gain $L_k = WA(D_{k-1} + A^\top WA)^{-1}$. The backward sweep $x_k = y_k - L_{k+1}^\top x_{k+1}$ is then the information-form Rauch–Tung–Striebel smoother. Thus the GBP optimisation produces exactly the KF + RTS state estimate. □

VI. Evaluation

To thoroughly evaluate our unified GBP-based autonomy stack, we designed three experiments that progressively stress tracking, planning, and coordination.

Experiment 1: GBP vs LQR. Two robots follow predefined concentric circular trajectories over a 20s horizon. We compare our MultiRobotGBPController against independently tuned LQR controllers (with gains optimized via Optuna) using identical cost matrices. Metrics include instantaneous RMSE, cumulative tracking error.

Experiment 2: GBP Circular Planning. Two robots track their own collision-aware trajectories around circular goals.

Experiment 3: GBP Linear Planning. The same two-robot setup is repurposed to follow linear goals. Trajectories are entirely generated by GBP on the fly. This isolates performance on different geometry priors (circular vs. linear) under identical inference machinery.

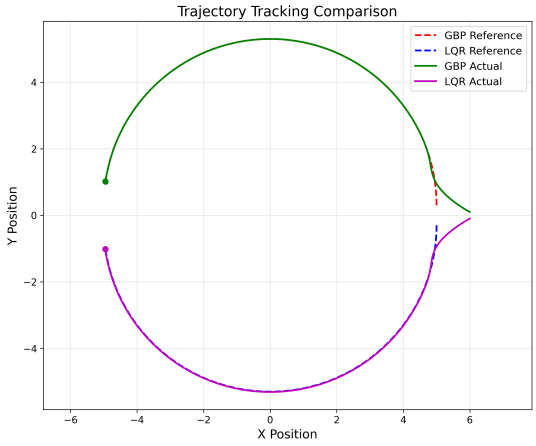

Fig. 2. Trajectories achieved by GBP (green solid) and LQR (pink solid) controllers relative to the circular reference (dotted).

TABLE I
Experiment 1: Detailed tracking performance metrics for GBP vs. LQR.

Controller	RMSE [m]	Max. Error [m]	IAE [m·s]	Total Effort	Avg. Effort
GBP	0.1370	1.0207	0.4096	33.2126	2.3438
LQR	0.1362	1.0219	0.5377	38.3474	2.7097

Results of Experiment 1

The results in Figures 2, 3, and 4, along with Table I, demonstrate that the proposed MultiRobot GBPController performs comparably to the tuned LQR baseline. As shown in Figure 2, both controllers closely follow the circular reference trajectories. Notably, GBP empirically converges to the LQR solution, reinforcing its validity as an inference-based control strategy. These results suggest GBP is a competitive alternative to classical control within a unified inference framework.

Results of Experiment 2

Figures 5 and 6 illustrate the effectiveness of GBP in generating safe, coordinated circular trajectories. As seen in Figure 5, the inter-robot distance remains consistently above the safety threshold, verifying that the collision avoidance factors are respected. Figure 6 provides a spatial snapshot of the closest approach, where both robots maintain safe separation while orbiting distinct centers. These results demonstrate that GBP can track smooth, geometry-aware motions while ensuring safety without relying on external references.

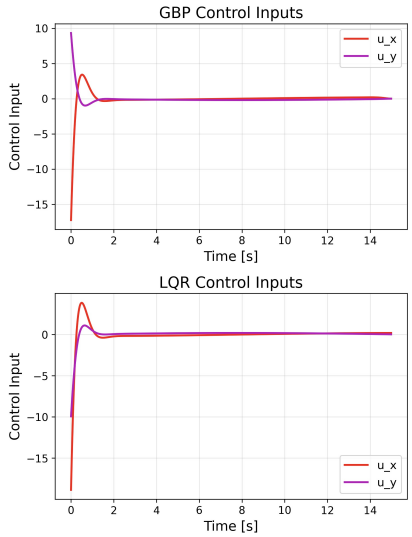

Fig. 3. Control inputs: comparison between GBP and LQR showing the transient and settling behavior in the x and y channels. GBP produces smoother long-term actuation with reduced high-frequency oscillation.

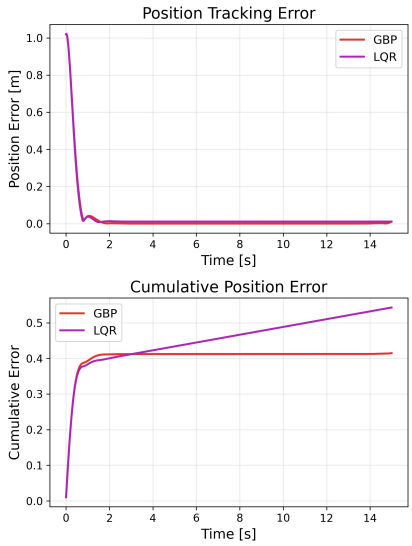

Fig. 4. Tracking performance: (top) instantaneous position error and (bottom) cumulative position error over time. GBP achieves lower drift in cumulative error compared to LQR while maintaining comparable short-term accuracy.

Results of Experiment 3

Figures 7 and 8 highlight the capability of GBP to plan and execute coordinated straight-line motions. No reference is provided, GBP's internal planning tool generates the optimal reference between the start and goal states. In Figure 7, the inter-robot distance consistently remains above the defined safety threshold, validating the effectiveness of soft collision avoidance factors. Figure 8 illustrates that both robots plan and track their own trajectories, while negotiating the closest encounter region using smooth and geometry-aware motion. These results demonstrate that GBP responds well to different geometries.

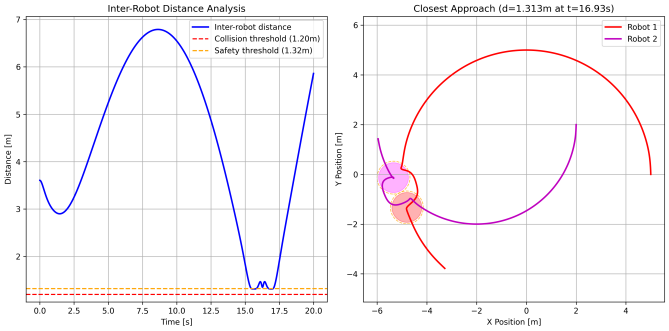

Fig. 5. Inter-robot distance over time with the collision threshold (red dashed) and safety margin (orange dashed). The plot highlights that the two robots maintain safe separation, with the closest approach annotated in the inset snapshot.

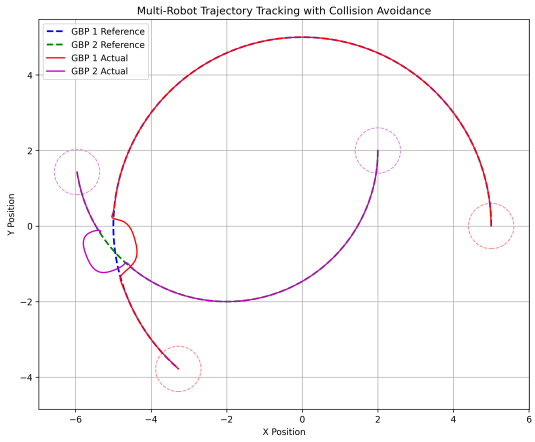

Fig. 6. Spatial view of the closest approach: actual GBP-executed trajectories (solid) and their implied goal circles, with soft safety regions visualized. The snapshot illustrates how the robots negotiate near-miss geometry while respecting the safety margin.

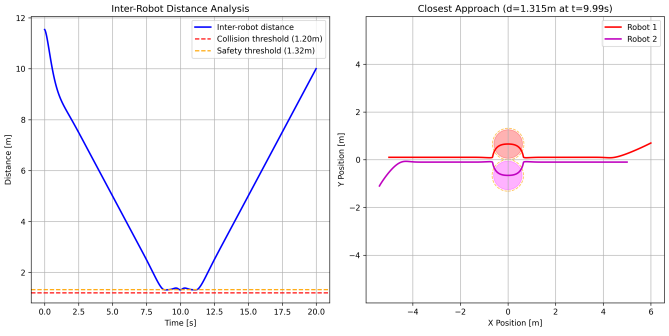

Fig. 7. Inter-robot distance over time during linear planning, with collision threshold (red dashed) and safety margin (orange dashed). The plot shows the robots maintain safe separation while negotiating straight-line objectives.

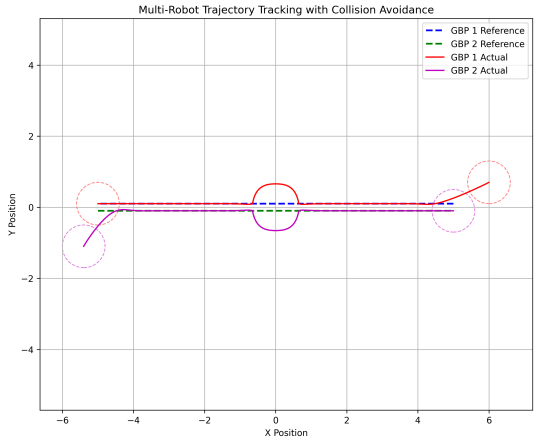

Fig. 8. Trajectory execution under GBP linear planning. Planned goals (dashed) and actual paths (solid) are shown for both robots, including soft safety regions. The plot confirms GBP maintains coordination and safety while pursuing linear objectives.

VII. Conclusion

We have presented a unified Gaussian Belief Propagation framework that casts planning, estimation, and control for multi-robot systems into a single factor-graph inference problem. By replacing separate optimisation and filtering stages with one message-passing loop, the method delivers coherent global behaviour while requiring only local communication between neighbouring robots.

In simulation, the GBP pipeline matches the tracking accuracy and collision-avoidance performance of a traditional LQR + EKF stack while generating its own high level plans, yet does so with a markedly simpler architecture— a single data structure, and one solver. This architectural economy yields code that is easier to maintain, reason about, and extend, while naturally exposing uncertainty information that can be reused elsewhere in the autonomy stack.

Looking forward, we plan to embed mapping and high-level perception factors (e.g., SLAM landmarks, semantic detections) into the same graph. Doing so would create a fully end-to-end autonomous pipeline—mapping, localisation, planning, and control all expressed as Gaussian inference—opening the door to highly adaptable, distributed robotic teams that re-optimise their entire belief state on the fly.

References

[1] J. Ortiz, T. Evans, and A. J. Davison, "A visual introduction to Gaussian Belief Propagation," *arXiv preprint arXiv:2107.02308*, Jul. 2021. [Online]. Available: https://arxiv.org/abs/2107.02308

[2] A. J. Davison and J. Ortiz, "FutureMapping2: Gaussian Belief Propagation for Spatial AI," *arXiv preprint arXiv:1910.14139*, 2022. [Online]. Available: https://arxiv.org/abs/1910.14139

[3] S. Nabarro, M. van der Wilk, and A. J. Davison, "Learning in deep factor graphs with Gaussian belief propagation," *arXiv preprint arXiv:2311.14649*, 2024. [Online]. Available: https://arxiv.org/abs/2311.14649

[4] A. Patwardhan and A. J. Davison, "A Distributed Multi-Robot Framework for Exploration, Information Acquisition and Consensus," *arXiv preprint arXiv:2310.01930*, 2023. [Online]. Available: https://arxiv.org/abs/2310.01930

[5] A. Patwardhan, R. Murai, and A. J. Davison, "Distributing Collaborative Multi-Robot Planning With Gaussian Belief Propagation," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 552–559, Feb. 2023. doi:10.1109/LRA.2022.3227858. [Online]. Available: http://dx.doi.org/10.1109/LRA.2022.3227858

[6] J. Ortiz, M. van der Wilk, and A. J. Davison, "Factor Graph Message Passing for Multi-Agent Coordination," *arXiv preprint arXiv:2209.11680*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2209.11680

[7] J. Ortiz and A. J. Davison, "Gaussian Belief Propagation for Spatial AI: A Unifying Perspective," *arXiv preprint arXiv:2306.09339*, Jun. 2023. [Online]. Available: https://arxiv.org/abs/2306.09339

[8] F. Dellaert and M. Kaess, "Factor graphs for robot perception," *Foundations and Trends in Robotics*, vol. 6, no. 1–2, pp. 1–139, 2017. doi:10.1561/2300000043.

[9] R. Murai, A. Patwardhan, and A. J. Davison, "Distributed Inference in Multirobot Systems via Gaussian Belief Propagation," *arXiv preprint arXiv:2401.10203*, Jan. 2024. [Online]. Available: https://arxiv.org/abs/2401.10203

[10] A. J. Davison, "Real-Time Simultaneous Localisation and Mapping with a Single Camera," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2003, pp. 1403–1410. doi:10.1109/ICCV.2003.1238654.