frontiers | Frontiers in Robotics and AI

TYPE Original Research
PUBLISHED 20 January 2026
DOI 10.3389/frobt.2025.1698333

OPEN ACCESS

EDITED BY
Shaoming He,
Beijing Institute of Technology, China

REVIEWED BY
Nauman Qadeer,
Federal Urdu University of Arts, Sciences and Technology Islamabad, Pakistan
Liang Fang,
Imperial College, United Kingdom

*CORRESPONDENCE
Anas Abdelkarim,
anas.abdelkarim@uni.lu,
abdelkarim@eit.uni-kl.de

RECEIVED 03 September 2025
REVISED 14 November 2025
ACCEPTED 19 November 2025
PUBLISHED 20 January 2026

CITATION
Abdelkarim A, Görges D and Voos H (2026) ecg2o: a seamless extension of g2o for equality-constrained factor graph optimization.
Front. Robot. AI 12:1698333.
doi: 10.3389/frobt.2025.1698333

ecg2o: a seamless extension of g2o for equality-constrained factor graph optimization

Anas Abdelkarim[1,2]*, Daniel Görges[2] and Holger Voos[1]

[1]Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg, Luxembourg, [2]Department of Electrical and Computer Engineering (EIT), RPTU University of Kaiserslautern-Landau, Kaiserslautern, Germany

Factor graph optimization serves as a fundamental framework for robotic perception, enabling applications such as pose estimation, simultaneous localization and mapping (SLAM), structure-from-motion (SfM), and situational modeling. Traditionally, these methods solve unconstrained least squares problems using algorithms such as Gauss-Newton and Levenberg-Marquardt. However, extending factor graphs with native support for hard equality constraints can yield more accurate state estimates and broaden their applicability, particularly in planning and control. Prior work has addressed equality handling either by soft penalties (large weights) or by nested-loop Augmented Lagrangian (AL) schemes. In this paper, we propose a novel extension of factor graphs that seamlessly incorporates hard equality constraints without requiring additional optimization techniques. Our approach maintains the efficiency and flexibility of existing second-order optimization techniques while ensuring constraint satisfaction. To validate the proposed method, an autonomous-vehicle velocity-tracking optimal control problem is solved and benchmarked against an AL baseline, both implemented in g2o. Additional comparisons are conducted in GTSAM, where the penalty method and AL are evaluated against our g2o implementations. Moreover, we introduce ecg2o, a header-only C++ library that extends the widely used g2o library with full support for hard equality-constrained optimization. This library, along with demonstrative examples and the optimal control problem, is available as open source at https://github.com/snt-arg/ecg2o.

KEYWORDS

constrained factor graphs, optimal control, SLAM, equality-constrained optimization, SQP

1 Introduction

Factor graphs are extensively used in robotic perception tasks for efficiently modeling large-scale probabilistic inference problems (Dellaert and Kaess, 2017). These methods use graph-based optimization techniques, such as weighted least squares, to address key problems like SLAM and situational awareness (Tourani et al., 2022; Bavle et al., 2022). Typically, these optimization problems are addressed using second-order unconstrained algorithms, including Gauss-Newton (Lai et al., 2017) and Levenberg–Marquardt (Moré, 2006). Several well-established libraries, such as GTSAM (Dellaert et al., 2022), g2o (Kümmerle et al., 2011), and SRRG2 (Grisetti et al., 2020), have been developed to provide robust back-end implementations of these optimization algorithms, along with user-friendly front-end interfaces tailored for robotic applications.

Recent advancements in the literature have extended factor graph–based optimization beyond perception tasks to also encompass optimal control applications. The primary challenge in this extension lies in handling constraints, where optimal control problems inherently involve hard constraints, unlike perception problems which are typically soft-constrained. However, factor graph optimization can provide a unified framework for perception and control tasks in robotics. Furthermore, this approach facilitates the reuse of established and computationally efficient algorithms available in factor graph libraries (Abdelkarim et al., 2025). In contrast, traditional optimization frameworks widely used in optimal control, such as CasADi (Andersson et al., 2019), AMPL (Abdelkarim and Zhang, 2020; Abdelkarim et al., 2023), and IPOPT (Wächter and Biegler, 2006), although highly effective, do not naturally integrate with factor graph–based optimization frameworks. This disconnect can introduce computational inefficiencies and added complexity, particularly in robotic applications that demand a seamless integration of perception and control.

It is important to highlight that in optimal control, hard equality constraints define strict and deterministic relationships between variables. A typical example is the system dynamics, which must be satisfied exactly to ensure that state transitions remain physically consistent. In contrast, the motion and sensor models in SLAM establish probabilistic or soft constraints between variables. These constraints do not enforce exact relationships but instead incorporate uncertainty directly into the factor graph formulation. As a result, hard equality constraints enforce exact dependencies to guarantee consistency, e.g., with the underlying physics, while soft probabilistic constraints model uncertainty and enable more flexible estimation.

In this work, we address the integration of hard equality constraints within factor graphs. Our contributions can be summarized as follows: (i) Method: we introduce an Sequential Quadratic Programming (SQP)-inspired approach that leverages the Karush-Kuhn-Tucker (KKT) conditions to natively integrate hard equality constraints into the factor graph framework. (ii) Implementation: we provide a lightweight, header-only C++ library, `ecg2o`, that extends the g2o optimization framework with support for both the Augmented Lagrangian (AL) method and the proposed KKT-based approach. The library is publicly available at https://github.com/snt-arg/ecg2o to foster reproducibility and further research. In addition, we introduce a stopping criterion based on the norm of the update step in g2o, ensuring termination once the step size becomes sufficiently small. (iii) Validation: we conduct a case study on trajectory-tracking optimal control, demonstrating the effectiveness of the proposed method. Furthermore, we benchmark our implementations against the penalty and AL methods in GTSAM, highlighting the efficiency and robustness of the implemented approaches.

The remainder of this paper is organized as follows. Section 2 reviews the state of the art in factor-graph-based optimization and motivates the need for more efficient methods for equality constraint handling. Section 3 introduces the necessary preliminaries, including weighted least squares, Gauss–Newton for unconstrained optimization, and the soft-constraint formulation. Section 4 presents the AL and the proposed KKT-based Gauss–Newton approaches. Section 5 reports the experimental evaluation comprising two case studies: trajectory-tracking optimal control and a nonlinear least-squares problem with equality constraints, where the proposed implementation is also compared against the penalty and AL methods in GTSAM. Finally, Section 6 concludes the paper and outlines directions for future work.

2 State of the art and motivation

2.1 State of the art

While factor graphs have initially been introduced to handle robotic perception tasks, they are increasingly used also for robotic optimal control problems, as summarized in the comprehensive review (Abdelkarim et al., 2025) along with a description of various methods for contsraint handling. In the following, we provide an overview of the key literature in this area.

Chen et al., 2019 introduced a general framework for applying Linear Quadratic Regulators (LQR) using factor graphs. In this framework, equality constraints are used to enforce system dynamics and are incorporated into the cost function as a weighted least squares term. By assigning a sufficiently large weighting matrix, these equality constraints effectively act as soft constraints, which are approximately satisfied as the weighting approaches infinity.

Yang et al. (2021) extended this approach by introducing additional equality constraints between variables within the LQR framework. Factor graph–based LQR has been applied to a variety of domains, including wireless mesh network control (Darnley, 2021), tactile estimation and extrinsic contact control (Kim et al., 2023), and trajectory generation for graffiti robots (Chen et al., 2022a; Chen et al., 2022b). Notably, these implementations have primarily relied on the GTSAM library to formulate and solve the optimization problems.

Beyond LQR-based approaches, more advanced constrained optimization techniques have been explored for handling equality constraints, most notably the AL method (Bertsekas, 2014). The fundamental idea of AL is to introduce Lagrange multiplier terms and a quadratic penalty term for constraint violations into the cost function, thereby forming the AL function. The optimization process involves two nested loops: the inner loop minimizes the AL function while keeping the Lagrange multipliers fixed, while the outer loop updates the Lagrange multipliers and penalty parameters until a specified stopping criterion is met.

The AL method has been implemented in GTSAM for improved state estimation (Sodhi et al., 2020) and has been applied in navigation and 3D manipulation planning (Qadri et al., 2022). Furthermore, the AL has been utilized in SRRG2 for localization and control of unicycle robots (Bazzana et al., 2022) and for pseudo-omnidirectional platform control (Bazzana et al., 2024).

While these methods have been successfully applied to various robotics and control problems, existing approaches still suffer from key limitations as described below.

2.2 Motivation

Incorporating equality constraints into factor graphs has been primarily achieved through two approaches: soft constraints

and the AL method. In the soft constraints approach, enforcing equality constraints requires assigning an infinitely large weighting matrix, which is not feasible in practical implementations. Instead, a large but finite weighting matrix is typically used. However, choosing an appropriate weight requires careful tuning to achieve balance between effectively enforcing the constraint and avoiding ill-conditioned optimization problems. If the original optimization problem is already ill-conditioned, determining an appropriate weighting matrix becomes even more challenging Abdelkarim et al. (2025).

In contrast, the AL method systematically enforces equality constraints without requiring direct weight tuning. However, the AL has notable limitations. First, as previously mentioned, it employs a nested loop structure, which may lead to unnecessary iterations in the inner loop. Second, the performance of the AL method is highly dependent on hyperparameter tuning Nocedal and Wright (2006), including the initial penalty term, penalty update factor, maximum penalty value, inner-loop iteration limit, and stopping criteria. Suboptimal parameter selection can significantly degrade performance.

To address these limitations, we propose an alternative approach that extends the Gauss-Newton and Levenberg–Marquardt methods, which are the primary unconstrained optimization algorithms used in factor graphs, by incorporating KKT conditions to explicitly enforce equality constraints. This approach offers faster convergence and eliminates the need for extensive hyperparameter tuning. Furthermore, despite g2o′s efficiency in robotic perception tasks, it does not provide built-in mechanisms for explicitly handling equality constraints. This limitation necessitates either manual constraint encoding using soft constraints (which leads to tuning challenges) or extending the solver framework itself.

3 Preliminaries

3.1 Weighted least squares in factor graphs

Factor graphs consist of variable nodes and factor nodes and represent a factorization of probability density functions (PDFs), which are typically assumed to follow a Gaussian distribution. In maximum *a posteriori* (MAP) inference, the objective is to maximize the factorized probability function, mathematically formulated in Equation 1.

$$X^{\mathrm{MAP}} = \underset{X}{\operatorname{argmax}} \prod_{j=1}^{r} \exp\left(-\frac{1}{2}||e_j(X_j)||^2_{\Omega_j}\right). \tag{1}$$

Here, $\exp()$ represents the exponential function, $||e||^2_{\Omega} = e^{\mathrm{T}}\Omega e$ denotes the Mahalanobis norm, and $.^{\mathrm{T}}$ is the transpose operator for vectors or matrices. The term e_j is an error function associated with the factor j, which depends on a subset of the variable nodes, and Ω_j is the information matrix of factor j, computed as the inverse of the covariance matrix. Furthermore, X is the vector containing all variable nodes X_j.

Since maximizing the MAP objective is equivalent to minimizing the negative log-likelihood, this optimization problem can be reformulated as a weighted least squares problem (Abdelkarim et al., 2025, § III.B)

$$X^{\mathrm{MAP}} = \arg\min_{X} \sum_{j=1}^{r} ||e_j(X_j)||^2_{\Omega_j}. \tag{2}$$

In the following sections, we present solutions to the optimization problem defined in Equation 2, first without hard constraints and then with incorporated hard equality constraints.

3.2 Gauss-newton method for unconstrained least-squares

In factor graphs, the factor nodes typically define a nonlinear error function. Thus, optimization methods operate on a linearized version of this function. This is achieved through first-order Taylor expansion described in Equation 3.

$$e_j\left(\underbrace{\widetilde{X_j} + \Delta X_j}_{X_j}\right) \approx \hat{e}_j(\Delta X_j) = e_j\left(\widetilde{X_j}\right) + \mathcal{J}_{e_j}(\widetilde{X_j})\Delta X_j, \tag{3}$$

where $\widetilde{X_j}$ is a linearization point and $\mathcal{J}e_j(\widetilde{X_j})$ is the Jacobian matrix of the error function of the factor j evaluated at the linearization point $\widetilde{X_j}$.

Applying the stationarity condition (the gradient of the cost function vanishes at an optimal point), the Gauss-Newton update step at iteration i is computed as

$$\underbrace{\sum_{j=1}^{r} \mathrm{map}\left(\mathrm{H}_j^i\right)}_{\mathrm{H}^i} \Delta X^i_{\mathrm{gn}} = \underbrace{\sum_{j=1}^{r} \mathrm{map}\left(\mathfrak{b}_j^i\right)}_{\mathfrak{b}^i}, \tag{4}$$

where the contribution of the factor j at iteration i in building the linear system is

$$\mathrm{H}_j^i = ||\mathcal{J}_{e_j}(X_j^i)||^2_{\Omega_j} \tag{5a}$$

$$\mathfrak{b}_j^i = -\mathcal{J}_{e_j}(X_j^i)^{\mathrm{T}}\Omega_j e_j\left(X_j^i\right), \tag{5b}$$

and map(·), in Equation 4, is an operator that maps the local H_j^i and $\mathfrak{b}_j^i$ to the global H^i and $\mathfrak{b}^i$. This mapping accounts for the fact that error functions generally depend only on subsets of the variable nodes, and map(·) ensures proper zero-padding where necessary.

Constructing and solving this linear system, presented in Equation 5, is a key computational step in factor graph optimization. Once the linear system is solved, the variable nodes are updated in each iteration as described in Equation 6.

$$X^{i+1} = X^i + \Delta X^i_{gn}. \tag{6}$$

Although the linear system formulation remains the same across unconstrained optimization methods, different techniques introduce modifications. For example, in the Levenberg-Marquardt method, numerical stability is improved by adding a positive damping term to the diagonal elements of the matrix H, mitigating issues related to ill-conditioned matrices.

3.3 Equality constraints as soft constraints

For vector-valued equality constraints $h_j(X_j) = 0$, a common approach is to add a quadratic penalty $||h_j(X_j)||^2_{W_{h_j}}$ with a large positive-definite (typically diagonal) weighting matrix W_{h_j} to the factor-graph objective in (2). This yields the unconstrained problem as presented in Equation 7.

$$\min_X \quad \sum_{j=1}^{r} ||e_j(X_j)||^2_{\Omega_j} + \sum_{j=1}^{l} ||h_j(X_j)||^2_{W_{h_j}}. \tag{7}$$

In a factor graph, this is implemented by adding one cost factor per constraint with error $e_{h_j} = h_j(X_j)$ and information matrix $\Omega_{h_j} = W_{h_j}$. In the limit $W_{h_j} \to \infty$, the optimizer drives $h_j(X_j) \to 0$. In practice, however, very large weights can cause ill-conditioning and sensitivity to variable/constraint scaling, so careful (problem-dependent) weighting or normalization is required.

4 Methodology

We consider the optimization problem formulated in Equation 2, now extended with linearly independent equality constraints:

$$\min_X \quad \sum_{j=1}^{r} ||e_j(X_j)||^2_{\Omega_j} \tag{8a}$$

$$\text{s.t.} \quad h_j(X_j) = 0 \qquad j = 1, \ldots, l \tag{8b}$$

The associated Lagrangian (Abdelkarim, 2020, §2.4.2) for the optimization problem in Equation 8 is given in Equation 9.

$$L(X, \gamma) = \sum_{j=1}^{r} ||e_j(X_j)||^2_{\Omega_j} + \sum_{j=1}^{l} \gamma_{h_j}{}^{\mathrm{T}} h_j(X_j), \tag{9}$$

where the first term is the standard factor-graph objective (regular factors) and the second term introduces the Lagrange multipliers γ_{h_n} for the equality constraints.

This section provides a brief overview of the two methods implemented in `ecg2o`: the AL method and the KKT-based approach. Throughout, we use $||v||^2_W = v^{\mathrm{T}} W v$.

4.1 Augmented lagrangian method

Inspired by the approach presented in (Bazzana et al., 2024), we implemented the AL method in `ecg2o` to enforce equality constraints. Particularly, the AL method adds a penalty term $||h_j(X_j)||^2_{\mathrm{P}_{h_n}}$ for each equality constraint, to the Lagrangian function. This results in the following augmented Lagrangian function:

$$L_{\mathrm{aug}}(X, \gamma) = L(X, \gamma) + \sum_{n=1}^{l} ||h_n(X_n)||^2_{\mathrm{P}_{h_n}}, \tag{10}$$

where the penalty term in Equation 10 is weighted by the matrix described in Equation 11.

$$\mathrm{P}_{h_n} = \mathrm{diag}\left(\rho_{h_n,1}, \ldots, \rho_{h_n,d}\right) \tag{11}$$

and diag(·) represents a diagonal matrix of dimension d. The penalty parameter ρ. Can either be uniform across all constraints or vary based on the algorithm's settings. Here, we assume a uniform ρ. For all constraints.

Notably, in the AL, the penalty terms do not need to be excessively large, as discussed in (Nocedal and Wright, 2006, Example 17.4). The algorithm iteratively minimizes the AL function with respect to X, using the stationary condition, which yields an update step similar to Equation 4.

While the contribution of regular factors remains unchanged (as described in Equation 5), the contribution of equality factor n at iteration i can be described as shown in Equation 12.

$$\mathrm{H}^i_n = ||\mathcal{J}_{h_n}(X^i_n)||^2_{\mathrm{P}_{h_n}} \tag{12a}$$

$$\mathfrak{b}^i_n = -\mathcal{J}_{h_n}(X^i_n)^{\mathrm{T}} \left(\mathrm{P}_{h_n} h_n(X^i_n) - \gamma_{h_n}\right). \tag{12b}$$

The AL algorithm implemented in the `ecg2o` library iteratively solves the inner-loop unconstrained optimization while incorporating both cost and equality factor contributions when constructing the linear system. Upon satisfying the termination criteria or reaching the maximum number of inner-loop iterations, the algorithm exits the inner loop and updates the Lagrange multipliers using Equation 13.

$$\gamma^{i+1}_{h_n} = \gamma^i_{h_n} + \mathrm{P}_{h_n} h_n(X^i_n). \tag{13}$$

Optionally, we increase the penalties according to Equation 14.

$$\rho_{\cdot} = \max\left(\rho_{max}, \alpha\rho_{\cdot}\right), \tag{14}$$

where ρ_{max} is the upper bound on the penalty parameter, and $\alpha \geq 1$ is the penalty update factor.

4.2 KKT-based gauss-newton method

In this section, we present an SQP-inspired method that enables factor graphs to natively solve equality-constrained optimization problems by leveraging KKT conditions. Unlike traditional approaches that require specialized algorithms like Penalty or Augmented Lagrangian methods, our approach demonstrates that factor graphs inherently possess the mathematical structure to handle equality constraints through appropriate factor design, eliminating the need for algorithmic modifications or nested loops.

Our core contribution is the formulation of equality constraints as regular factor nodes that, when combined with standard factor graph optimization, naturally enforce the KKT conditions of the original constrained problem. This transforms the equality constrained problem into an unconstrained optimization over an extended variable space that includes both original variables and Lagrange multipliers. Crucially, this formulation maintains the computational efficiency and sparsity exploitation of standard factor graph optimization while ensuring exact constraint satisfaction—unlike penalty methods that only achieve approximate satisfaction.

The mathematical foundation establishes that standard Gauss-Newton updates applied to our specially designed equality constraint factors yield iterations equivalent to SQP methods. This equivalence represents a significant theoretical advancement, demonstrating that factor graphs can directly implement

sophisticated equality-constrained optimization without external algorithmic frameworks.

Consistent with the SQP framework, each subproblem is constructed by linearizing the nonlinear system. We begin by linearizing both the error functions e_j and the equality constraints h_j around the current estimate $\overline{X}_j$. Their linearized versions, denoted by $\hat{e}_j$ and $\hat{h}_j$ respectively, are given by:

$$e_j\left(\underbrace{\overline{X}_j + \Delta X_j}_{X_j}\right) \approx \hat{e}_j(\Delta X_j) = e_j\left(\overline{X}_j\right) + \mathcal{J}_{e_j}(\overline{X}_j)\Delta X_j, \tag{15a}$$

$$h_j\left(\underbrace{\overline{X}_j + \Delta X_j}_{X_j}\right) \approx \hat{h}_j(\Delta X_j) = h_j\left(\overline{X}_j\right) + \mathcal{J}_{h_j}(\overline{X}_j)\Delta X_j. \tag{15b}$$

The Lagrangian function for the linearized system in Equation 15 is given in Equation 16.

$$\hat{L}(\Delta X, \gamma) = \sum_{j=1}^{r} ||\hat{e}_j(\Delta X_j)||^2_{\Omega_j} + \sum_{j=1}^{l} \gamma_{h_n}{}^{\mathrm{T}} \hat{h}_j\left(\Delta X_j\right), \tag{16}$$

Let $\Delta X^*, \gamma^*$ be the local optimal points. These must satisfy the KKT conditions (Boyd and Vandenberghe, 2004, §5.3.3) of the linearized optimization problem:

$$\nabla_{\Delta X}\hat{L}(\Delta X^*, \gamma^*) = 0, \quad \hat{h}(\Delta X^*) = 0, \tag{17}$$

where ∇, in Equation 17, denotes the gradient, and $\hat{h}$ is the vector of all equality constraints.

Assuming $\gamma^* = \gamma^i + \Delta\gamma^i$, the KKT system is formulated as:

$$\underbrace{\begin{bmatrix} \mathrm{H}^i & \mathcal{J}_h(X^i)^{\mathrm{T}} \\ \mathcal{J}_h(X^i) & 0 \end{bmatrix}}_{\text{KKT matrix}} \underbrace{\begin{bmatrix} \Delta X^i_{\text{gn}} \\ \Delta\gamma^i \end{bmatrix}}_{\text{update step}} = \underbrace{\begin{bmatrix} \mathfrak{b}^i - \mathcal{J}_h^T(X^i)\gamma^i \\ -h(X^i) \end{bmatrix}}_{\text{KKT vector}} \tag{18}$$

The introduction of equality constraints modifies the linear system compared to unconstrained optimization, as highlighted in red. The key challenge lies in representing these constraints within the factor graph using appropriate variables and factors. Specifically, how can we define the error function and the weighting matrix associated with the equality constraints to enforce them effectively?

4.2.1 Design of equality constraint factors

We propose defining a regular factor for each equality constraint such that it produces the same update step as in (18). Specifically, for an equality constraint h_j, we define a regular factors with the following error function and weighting matrix:

$$e_{h_j} = \begin{bmatrix} h_j\left(X_j\right) \\ \gamma_{h_j} \end{bmatrix}, \quad \Omega_{h_j} = \begin{bmatrix} 0_{d\times d} & \mathrm{I}_{d\times d} \\ \mathrm{I}_{d\times d} & 0_{d\times d} \end{bmatrix}, \tag{19}$$

where h_j and γ_{h_j} have the dimension d, and I denotes the identity matrix. This formulation results in a linear system equivalent to the KKT system of the equality-constrained optimization problem. Consequently, it allows us to enforce equality constraints while maintaining an unconstrained factor graph structure. Furthermore, in this approach, the Lagrange multipliers are treated as variable nodes, naturally integrating them into the optimization process.

For ease of implementation, the `ecg2o` library provides a dedicated class for equality factors, which automatically incorporates Lagrange multipliers and defines the associated weighting matrix described in Equation 19. Additionally, the class simplifies the Jacobian implementation for equality factors, making the equality implementation as straightforward as for regular factors.

This design ensures an efficient and flexible extension of factor graphs, maintaining the computational advantages of existing optimization techniques while enabling constrained optimization in a natural manner.

5 Results and evaluation

In this section, we present two case studies: (i) an optimal control problem, and (ii) a comparison between the AL and penalty methods in GTSAM, and the AL and KKT-based methods in `ecg2o`.

5.1 Case study 1

We solve here an optimization problem that generates a sequence of control force inputs for an autonomous car to track a reference velocity trajectory. Our objective is to compare the performance of the AL method with our proposed approach, which models equality constraints as regular edges by incorporating Lagrange multipliers.

5.1.1 Problem formulation

Inspired by Jia et al. (2023), we formulate the optimal control problem as

$$\min \quad (||x_N - r_N||^2_{\mathrm{P}} + \sum_{k=1}^{N-1} \left(||x_k - r_k||^2_{\mathrm{Q}} + ||u_k||^2_{\mathrm{R}}\right) \tag{20a}$$

$$\text{s.t.} \quad x_{k+1} - \left[x_k + \frac{\delta t}{m}\left(u_k - F_{\text{resis}}\right)\right] = 0, \quad k = 0, \ldots, N-1, \tag{20b}$$

where x_k represents the vehicle velocity, u_k is the input force applied to the car (measured in Newtons), and r_k is the reference velocity. In addition, the subscript k denotes the time instance, while δt represents the sampling time of the controller, which defines the horizon length.

The resistance force F_{resis} depends on the gravitational force F_{grav}, the rolling resistance F_{roll}, and the aerodynamic resistance force F_{air}. The total resistance force is given by

$$F_{resist} = \underbrace{m_v g \sin\theta}_{Fgrav} + \underbrace{\frac{1}{2}\rho_a A_f c_a x^2}_{Fair} + \underbrace{m_v g c_r \cos\theta}_{Froll}. \tag{21}$$

The model is nonlinear due to the aerodynamic resistance force. The model parameters are as follows: m represents the effective mass of the mass under the effect of the rotational mass of the powertrain, while m_v denotes the actual mass of the vehicle. The gravitational constant is given by g, and θ represents the road slope. The air density is denoted by ρ_a, and A_f is the frontal area of the vehicle. The coefficients c_a and c_r correspond to the air drag and rolling resistance coefficient, respectively.

The factor graph representing the optimal control problem in Equation 20 is illustrated in Figure 1, where the equality constraints are incorporated using our proposed approach.

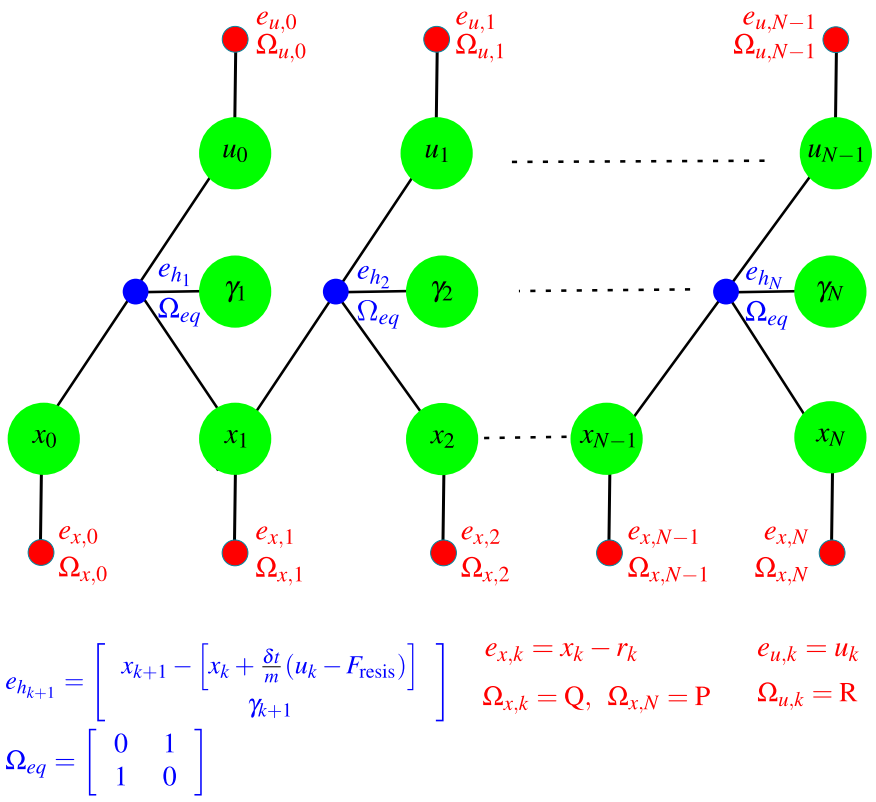

FIGURE 1
The factor graph for the optimal control problem. The red circles represent the cost terms as regular factors, while the blue circles illustrate the representation of equality constraints as regular factors. $k \in \{0, \ldots, \mathrm{N}\text{-}1\}$.

5.1.2 Evaluation

The reference trajectory for the velocity profile consists of 385 points with a sampling rate of 1 s. Using the factor graph-based optimization method, we solve the optimal control problem with weighting parameters $Q = P = 1000$ and $R = 0.0007$ to determine the optimal input sequence. Applying this control sequence results in the optimal velocity profile. The velocity profile corresponding to the optimal control sequence obtained from solving the optimization problem is illustrated in Figure 2. While different controller performances can be achieved with varying parameter settings, our primary focus is to evaluate the optimizer itself. The solution of the optimization problem using the AL method results in a velocity trajectory that is quite similar to our proposed approach. This is because both methods share the same stopping criteria. We computed the root mean squared error (RMSE) between both trajectories to quantify their similarity. The RMSE between the velocity trajectories obtained from the AL method and our proposed approach is 0.2426, indicating a high degree of similarity.

5.1.2.1 Iteration numbers

In our case, the presence of nonlinear terms in the equality constraints might influence the number of iterations required for convergence. To address this effect, we considered an approximation for the nonlinear aerodynamic resistance term using a linear model:

$$x^2 = p_1 + p_2 x, \tag{22}$$

where p_1 and p_2, in Equation 22, are constants. Therefore, we considered five distinct scenarios for comparison. First, we solve the unconstrained optimization problem, where the constraints are ignored, serving as a baseline to understand the effect of enforcing constraints. Next, we apply our proposed method to both a linearized dynamic model and the nonlinear system, allowing us to analyze how constraint approximation influences performance. Additionally, we compare these results with the AL approach, solving the problem in both linearized and nonlinear forms. In addition, we consider three different reference trajectory lengths—5, 100, and 385—to evaluate the optimization performance across varying problem sizes. The results of the iteration number across all scenarios for the three trajectory lengths are summarized in Figure 3.

Figure 3 shows that the iteration number in the unconstrained optimization matches that of our proposed method with linearized dynamics, converging in just two iterations across all trajectory lengths. For nonlinear dynamics, our method requires slightly more iterations (3–4), reflecting the added complexity while maintaining efficiency.

In contrast, the AL method requires 11–13 iterations, approximately three times more than our proposed method, highlighting its higher computational cost. These results were obtained after careful parameter tuning, where we set the maximum number of inner iterations to 1, with $\rho_{\text{init}} = 10$, $\rho_{\max} = 50000$, and $\alpha = 10$. However, further tuning is still required for different optimization problems.

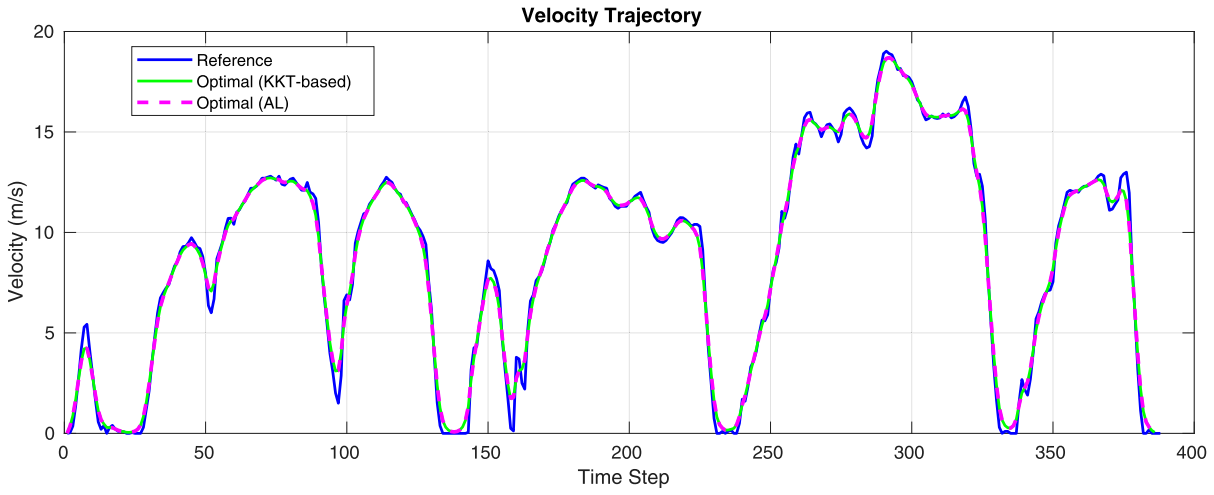

FIGURE 2
Optimal velocity trajectory obtained from the factor graph-based optimal control problem.

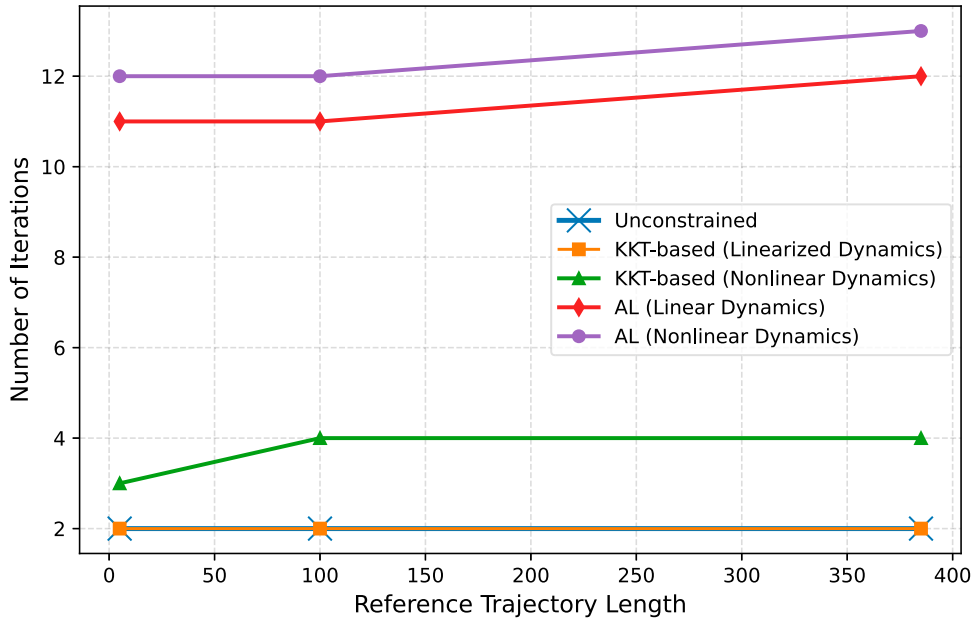

FIGURE 3
Iteration number across different optimization scenarios and trajectory lengths.

Regarding sensitivity to the initial value of the Lagrange multiplier, we found that both algorithms maintained a stable iteration number across different Lagrange multiplier initial values.

5.1.2.2 Computation time

We compare the computation time for the nonlinear dynamics scenario across three different trajectory lengths. The optimization problem is solved 1,000 times, and the reported computation time represents the average over these runs to ensure consistency. All experiments were conducted on a Linux system with an Intel Core i9 12th Gen processor and 32 GB of RAM. The results are shown in Figure 4.

The computation time for the AL method is slightly lower than that of our proposed method, which was expected since our solver handles a larger linear system in each iteration. However, advancements in linear solvers have reduced the impact of solving larger systems on overall computation time, making the difference less apparent. The increase in computation time for our method ranges between 20% and 35%, but the significant reduction in the number of iterations makes it a worthwhile trade-off, especially for applications requiring fast convergence.

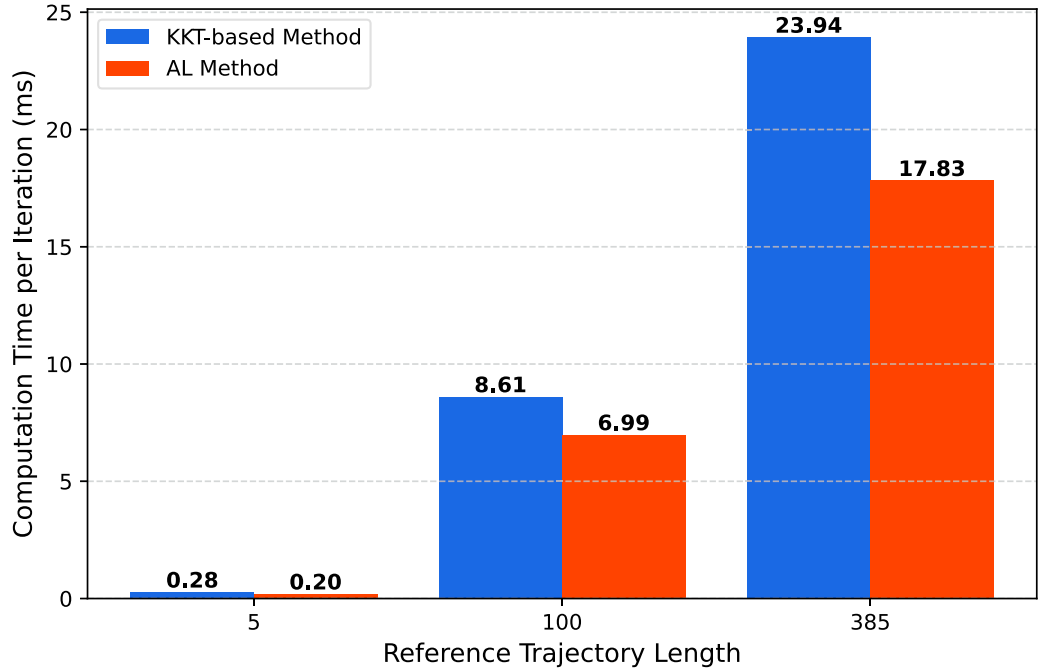

FIGURE 4
Computation time per iteration for different trajectory lengths using the Proposed Method and Augmented Lagrangian (AL) Method.

5.2 Case study 2: nonlinear least–squares with an equality constraint

For further validation of the implemented solvers, we adopted a nonlinear equality-constrained problem from the GTSAM library. In this case study, four solvers are considered: Penalty (GTSAM), AL (GTSAM), AL (`ecg2o`), and the KKT-based method (`ecg2o`). The constrained optimization problem is formulated in Equation 23.

$$\min_{x_1,x_2} \frac{1}{2}(x_1 - e^{-x_2})^2 + \frac{1}{2}(x_1^2 + 2x_2 + 1)^2, \tag{23a}$$

$$\text{s.t. } x_1 + x_1^3 + x_2 + x_2^2 = 0. \tag{23b}$$

A key distinction between our AL implementation in `ecg2o` and GTSAM's approach lies in the inner iteration control strategy. In the AL method, each outer iteration solves an equivalent unconstrained optimization problem with fixed penalty parameters and Lagrange multipliers. While one could solve this inner problem to high accuracy, this is computationally inefficient since the penalty and multiplier estimates are updated in subsequent outer iterations. Therefore, we explicitly limit the maximum number of inner iterations to balance computational efficiency with sufficient progress toward constraint satisfaction. In `ecg2o`, we set this limit to five inner iterations per outer loop, whereas GTSAM employs a different convergence strategy without such explicit limits.

This difference is illustrated in Figure 5, which shows the number of inner iterations per outer iteration for each method. The choice of inner iteration limit represents a trade-off: too few iterations may hinder progress, while too many may waste computational effort on prematurely accurate inner solutions. Notably, when we increased the maximum inner iteration limit from 5 to 50 in our AL implementation for the initial guess $(-0.2, -0.2)$, the total iteration count increased from 20 to 45 without additional improvement in solution quality.

The iteration numbers reported in Table 1 are measured consistently across implementations. In `ecg2o`, we directly count the number of inner iterations (i.e., iterations where the main calculations and linear system updates occur), while for GTSAM we sum the reported *uopt_iters* across all outer iterations to obtain a comparable metric. All solvers successfully converged to the optimal solution $x^* = (0, 0)$ with accuracy better than 10^{-4} across all initial guesses

$$x^{(0)} \in \{(-1,-1),\ (-0.2,-0.2),\ (0,0),\ (2,2)\}.$$

The KKT-based Gauss-Newton method in `ecg2o` achieved the fastest convergence (average of five iterations), followed by AL in `ecg2o` (average of 20 iterations). Both GTSAM baselines required substantially more iterations (averages of 73 for penalty and 72 for AL). The performance advantage of our KKT-based approach is consistent across initial conditions, demonstrating its efficiency for this class of constrained problems.

We further investigated the effect of the penalty parameter α on AL performance. Increasing α from 1.5 to 2 reduced the iteration count from 20 to 18 for the $(-0.2, -0.2)$ case, as accelerated penalty growth drives faster constraint satisfaction. However, overly aggressive penalty increases risk numerical ill-conditioning, particularly in large-scale systems. This parameter sensitivity highlights a key advantage of our KKT-based method, which requires no such tuning while maintaining robust performance. The AL parameters used in this study were $\rho_{\text{init}} = 1$, $\rho_{\max} = 50000$, and $\alpha = 1.5$, with a maximum of five inner iterations per outer loop.

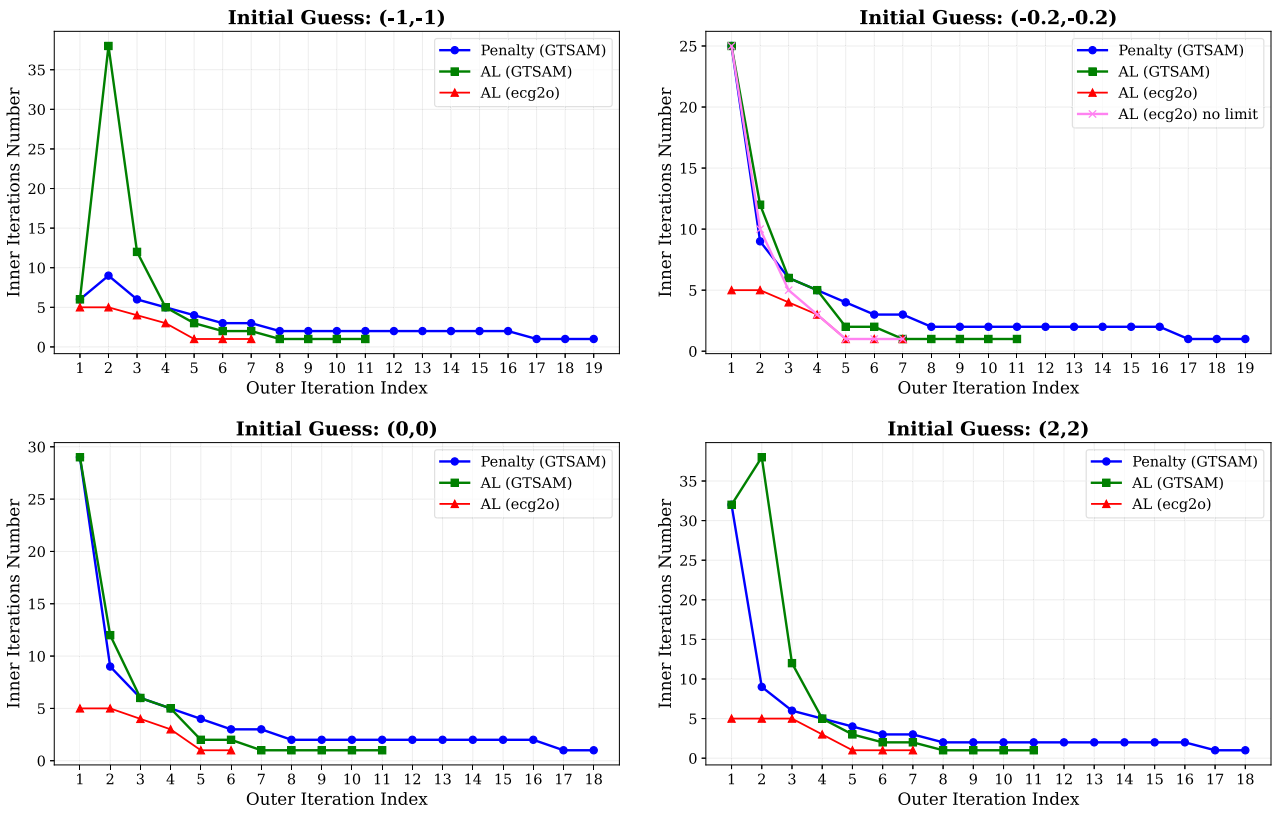

FIGURE 5
Inner iterations number per outer iteration index for different methods and initial guesses.

TABLE 1 Case study 2: iterations number to convergence for each method and initial guess.

Method	$x^{(0)} = (-1,-1)$	$x^{(0)} = (-0.2,-0.2)$	$x^{(0)} = (0,0)$	$x^{(0)} = (2,2)$	Avg.
Penalty (GTSAM)	56	75	79	82	73.0
AL (GTSAM)	72	57	61	98	72.0
AL (ecg2o)	20	20	19	21	20.0
KKT-based (ecg2o)	6	4	1	9	5.0

6 Conclusion and future work

In this paper, we addressed the challenge of incorporating equality constraints into factor graphs, an essential extension for applications beyond robotic perception, such as optimal control. While the AL method is the state-of-the-art approach for handling equality constraints, we proposed an SQP-inspired, KKT-based method that natively integrates constraints within factor graph optimization without requiring additional iterative adjustments.

The proposed approach outperforms the AL method by achieving faster convergence and eliminating the need for parameter tuning, thereby enhancing its practicality. Its efficiency and robustness were validated through a case study on velocity tracking in autonomous vehicles. In addition, the implementation was compared against the Penalty method and AL in GTSAM, demonstrating superior performance.

As future work, we aim to explore more advanced constrained optimization techniques beyond AL, specifically targeting the handling of inequality constraints within factor graph frameworks. This would further expand the applicability of factor graphs to control and planning problems in robotics.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

AA: Validation, Conceptualization, Investigation, Writing – review and editing, Software, Writing – original draft, Formal Analysis, Visualization, Data curation, Methodology. DG: Validation, Conceptualization, Supervision, Writing – review and editing, Formal Analysis, Methodology, Visualization. HV: Conceptualization, Supervision, Writing – review and editing, Methodology, Formal Analysis, Funding acquisition, Visualization, Project administration, Validation, Resources.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. This research was funded in whole, or in part, by the Luxembourg National Research Fund (FNR), MOCCA Project, ref. 17041397. For the purpose of open access, and in fulfilment of the obligations arising from the grant agreement, the author has applied a Creative Commons Attribution 4.0 International (CC BY 4.0) license to any Author Accepted Manuscript version arising from this submission.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The authors declare that no Generative AI was used in the creation of this manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

References

Abdelkarim, A. (2020). Development of numerical solvers for online optimization with application to MPC-Based energy-optimal adaptive cruise control. *Master's Thesis, Tech. Univ. Kaiserslaut*. doi:10.13140/RG.2.2.11897.28000

Abdelkarim, A., and Zhang, P. (2020). "Optimal scheduling of preventive maintenance for safety instrumented systems based on mixed-integer programming," in *Model-based safety and assessment: 7Th international symposium, IMBSA 2020, Lisbon, Portugal, September 14–16, 2020, proceedings 7* (Springer), 83–96.

Abdelkarim, A., Jia, Y., and Gorges, D. (2023). "Optimization of vehicle-to-grid profiles for peak shaving in microgrids considering battery health," in *IECON 2023-49th annual conference of the IEEE industrial electronics society (IEEE)*, 1–6.

Abdelkarim, A., Voos, H., and Görges, D. (2025). Factor graphs in optimization-based robotic control-a tutorial and review. *IEEE Access* 13, 28315–28334. doi:10.1109/access.2025.3534993

Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). Casadi: a software framework for nonlinear optimization and optimal control. *Math. Program. Comput.* 11, 1–36. doi:10.1007/s12532-018-0139-4

Bavle, H., Sanchez-Lopez, J. L., Shaheer, M., Civera, J., and Voos, H. (2022). S-Graphs+: real-Time localization and mapping leveraging hierarchical representations. *IEEE Robotics Automation Lett.* 8, 4927–4934. doi:10.1109/lra.2023.3290512

Bazzana, B., Guadagnino, T., and Grisetti, G. (2022). Handling constrained optimization in factor graphs for autonomous navigation. *IEEE Robotics Automation Lett.* 8, 432–439. doi:10.1109/lra.2022.3228175

Bazzana, B., Andreasson, H., and Grisetti, G. (2024). How-to augmented lagrangian on factor graphs. *IEEE Robotics Automation Lett.* 9, 2806–2813. doi:10.1109/lra.2024.3361282

Bertsekas, D. P. (2014). *Constrained optimization and lagrange multiplier methods*. Academic Press.

Boyd, S. P., and Vandenberghe, L. (2004). *Convex optimization*. Cambridge, United Kingdom: Cambridge University Press.

Chen, G., Zhang, F., and Dellaert, Y. (2019). LQR control using factor graphs. Available online at: https://gtsam.org/2019/11/07/lqr-control.html (Accessed June 15, 2024).

Chen, G., Baek, S., Florez, J.-D., Qian, W., Leigh, S.-w., Hutchinson, S., et al. (2022a). "Gtgraffiti: spray painting graffiti art from human painting motions with a cable driven parallel robot," in *2022 international conference on robotics and automation (ICRA)* (IEEE), 4065–4072.

Chen, G., Hutchinson, S., and Dellaert, F. (2022b). "Locally optimal estimation and control of cable driven parallel robots using time varying linear quadratic Gaussian control," in *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (IEEE), 7367–7374.

Darnley, R. (2021). *Flow control of wireless mesh networks using LQR and factor graphs*. Carnegie Mellon University. Master's thesis.

Dellaert, F., and Kaess, M. (2017). Factor graphs for robot perception. *Found. Trends® Robotics* 6, 1–139. doi:10.1561/2300000043

Dellaert, F., Agrawal, V., Roberts, R., Cunningham, A., Beall, C., Ta, D.-N., et al. (2022). Borglab/GTSAM. doi:10.5281/zenodo.5794541

Grisetti, G., Guadagnino, T., Aloise, I., Colosi, M., Della Corte, B., and Schlegel, D. (2020). Least squares optimization: from theory to practice. *Robotics* 9, 51. doi:10.3390/robotics9030051

Jia, Y., Abdelkarim, A., Klingbeil, X., Savelsberg, R., and Görges, D. (2023). Performance evaluation of energy-optimal adaptive cruise control in simulation and on a test track. *IFAC-PapersOnLine* 56, 4994–5000. doi:10.1016/j.ifacol.2023.10.1276

Kim, S., Jha, D. K., Romeres, D., Patre, P., and Rodriguez, A. (2023). "Simultaneous tactile estimation and control of extrinsic contact," in *2023 IEEE international conference on robotics and automation (ICRA)* (IEEE), 12563–12569.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). "g2o: a general framework for graph optimization," in *2011 IEEE international conference on robotics and automation (IEEE)*, 3607–3613.

Lai, W. H., Kek, S. L., and Tay, K. G. (2017). Solving nonlinear least squares problem using gauss-newton method. *Int. J. Innovative Sci. Eng. and Technol.* 4 (1), 258–262. Available online at: https://ijiset.com/vol4/v4s1/IJISET_V4_I01_35.pdf.

Moré, J. J. (2006). "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis: proceedings of the biennial conference held at Dundee, June 28–July 1, 1977* (Springer), 105–116.

Nocedal, J., and Wright, S. J. (2006). *Numerical optimization*. Springer.

Qadri, M., Sodhi, P., Mangelson, J. G., Dellaert, F., and Kaess, M. (2022). "Incopt: incremental constrained optimization using the bayes tree," in *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (IEEE), 6381–6388.

Sodhi, P., Choudhury, S., Mangelson, J. G., and Kaess, M. (2020). "ICS: incremental constrained smoothing for state estimation," in *2020 IEEE international conference on robotics and automation (ICRA)* (IEEE), 279–285.

Tourani, A., Bavle, H., Sanchez-Lopez, J. L., and Voos, H. (2022). Visual SLAM: what are the current trends and what to expect? *Sensors* 22, 9297. doi:10.3390/s22239297

Wächter, A., and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Programming* 106, 25–57. doi:10.1007/s10107-004-0559-y

Yang, S., Chen, G., Zhang, Y., Choset, H., and Dellaert, F. (2021). "Equality constrained linear optimal control with factor graphs," in *2021 IEEE international conference on robotics and automation (ICRA)* (IEEE), 9717–9723.