# Constraint Manifolds for Robotic Inference and Planning

Yetong Zhang[1], Fan Jiang[1], Gerry Chen[1], Varun Agrawal[1], Adam Rutkowski[2] and Frank Dellaert[1]

***Abstract*— We propose a manifold optimization approach for solving constrained inference and planning problems. The approach employs a framework that transforms an arbitrary nonlinear equality constrained optimization problem into an unconstrained manifold optimization problem. The core of the transformation process is the formulation of *constraint manifolds* that represent sets of variables subject to equality constraints. We propose various approaches to define the tangent spaces and retraction operations of constraint manifolds, which are crucial for manifold optimization. We evaluate our constraint manifold optimization approach on multiple constrained inference and planning problems, and show that it generates strictly feasible results with increased efficiency as compared to state-of-the-art constrained optimization methods.**

## I. Introduction

Many important problems in robotics involve constraints. In robot kinematics, each joint constrains the poses of its connected links [1] and contact surfaces are constrained to have zero distance at each contact point [2]. In state estimation and planning, the kinodynamics constraints need to be satisfied for all time steps [3], [4]. In trajectory planning, the robot needs to obey system dynamics and pass through predefined way-points or reach certain footstep locations [5]. In robot swarm applications, distances among the robots are also frequently enforced as constraints [6].

A lot of prior work has focused on tackling inference and planning problems with constraints. E.g., [7] proposes representing an inference problem with equality constraints as a constrained factor graph, [8] solves a constrained factor graph with the sequential quadratic programming (SQP) method, [9] utilizes a variable elimination strategy to solve an equality constrained linear quadratic regulator control problem, and [10] applies the Lagrangian multipliers method to solve a constrained SLAM problem.

State-of-the-art constrained optimization methods still face several issues in solving large-scale constrained optimization problems. The penalty method and the augmented Lagrangian method [11], [12] require iteratively solving unconstrained optimization problems, and can lead to problems with bad numerical properties. For SQP methods [13], finding a merit function that balances the dual objectives of reducing costs and satisfying constraints is nontrivial.

A powerful alternative to the constrained optimization methods is *manifold optimization*. It has several advantages, including lower complexity and better numerical properties [14]. The transformation from a constrained optimization problem into an unconstrained optimization problem eliminates equality constraints by defining manifold variables that represent the set of feasible assignments.

However, the transformation process from an equality constrained optimization problem into an unconstrained manifold optimization problem is not always available in cases with arbitrary constraints, since defining manifold variables requires expert knowledge, especially for large-scale, densely connected constraints. The tangent spaces and retraction operations on such manifolds need to be manually specified to perform manifold optimization, e.g., in [6], [15]–[18].

We aim to develop a general framework that transforms an arbitrary nonlinear equality constrained optimization problem into an unconstrained manifold optimization problem without prior knowledge of the constraints and manifolds. First, we define a *constraint manifold* that represents variables connected by equality constraints, enforcing that those constraints are always satisfied. The concept of a constraint manifold was first proposed in [19], and it has recently been applied to reinforcement learning and sampling [20], [21]. Then, we formulate the tangent space and retraction operations of a constraint manifold, which are derived directly from the constraints. Finally, we replace each set of constrained variables in the constrained optimization problem with a corresponding constraint manifold variable, and construct the equivalent cost on the new variables.

We evaluate our manifold optimization approach against state-of-the-art constrained optimization methods in six constrained inference and planning scenarios. We further improve the efficiency of the manifold optimization approach by exploring different ways to compute the tangent spaces and perform retractions on the constraint manifolds. We show that our manifold optimization approach outperforms the state-of-the-art constrained optimization methods in both efficiency and optimality in these scenarios.

Our contributions can be summarized as follows:

- Develop a general framework that transforms an equality constrained optimization problem into an unconstrained manifold optimization problem.
- Formulate constraint manifolds that represent variables subject to arbitrary equality constraints, and provide multiple methods to compute tangent spaces and perform retraction operations on constraint manifolds.
- Evaluate our constraint manifold optimization approach on multiple constrained inference and planning problems and show its advantages in efficiency and optimality compared to existing state-of-the-art methods.

[1]College of Computing, Georgia Institute of Technology, Atlanta, USA {yetong, fan.jiang, gerry, varunagrawal, fd27}@gatech.edu

[2]Munitions Directorate, Air Force Research Laboratory, Eglin AFB, USA adam.rutkowski@us.af.mil

## II. Preliminaries

First, we review some manifold related definitions and properties, which are crucial for defining and applying constraint manifolds. More detailed explanations are available in Boumal *et al.* [22] and Absil *et al.* [14].

**Manifold**: An $n$-dimensional manifold is a set $\mathcal{M}$ which is locally homeomorphic to $\mathbb{R}^n$.

**Tangent Space & Tangent Vector**: Let $x$ be a point on manifold $\mathcal{M}$. We consider the set, $C_x$, of smooth curves $c(t)$ passing through $x$ at $t = 0$ (1). Two curves are equivalent $c_1 \sim c_2$ if they pass $x$ with the same velocity. A tangent vector $v$ to $\mathcal{M}$ at $x \in \mathcal{M}$ is defined as a class of equivalent curves (2), and the tangent space, $\mathrm{T}_x\mathcal{M}$, is defined as the set of all equivalent classes (3) [22]. As the tangent space is a linear space, we can define its basis as $\mathcal{B}_x\mathcal{M} = (b_1, \ldots, b_n)$, a horizontal concatenation of basis tangent vectors, and any tangent vector can be expressed as a linear combination of the basis (4), with $\xi \in \mathbb{R}^n$.

$$C_x = \{c | c : I \to \mathcal{M} \text{ is smooth}, c(0) = x\} \tag{1}$$

$$v = [c] = \{\tilde{c} \in C_x : c \sim \tilde{c}\} \tag{2}$$

$$\mathrm{T}_x\mathcal{M} = \{[c] : c \in C_x\} \tag{3}$$

$$v = \sum_{i=1}^{n} \xi_i b_i = \mathcal{B}_x\mathcal{M} \cdot \xi \tag{4}$$

**Differential on Manifolds**: The differential of a smooth map $F : \mathcal{M} \to \overline{\mathcal{M}}$ from manifold $\mathcal{M}$ to manifold $\overline{\mathcal{M}}$, is a linear operator $\mathrm{D}F(x) : \mathrm{T}_x\mathcal{M} \to \mathrm{T}_{F(x)}\overline{\mathcal{M}}$ that maps a tangent vector $v = [t \to (c(t))]$ of manifold $\mathcal{M}$ to the tangent vector $\overline{v} = [t \to F(c(t))]$ of manifold $\overline{\mathcal{M}}$ (5) [22] . Equipped with the tangent space bases $\mathcal{B}_x\mathcal{M}$, $\mathcal{B}_{F(x)}\overline{\mathcal{M}}$, we can write the differential in matrix form (6), where $J_f(x)$ is the Jacobian matrix, and $\overline{\xi}, \xi$ represent the corresponding basis decompositions, i.e., $\overline{v} = \mathcal{B}_{F(x)}\overline{\mathcal{M}} \cdot \overline{\xi}$, $v = \mathcal{B}_x\mathcal{M} \cdot \xi$.

$$\mathrm{D}F(x)[v] = \overline{v} \tag{5}$$

$$J_F(x) \cdot \xi = \overline{\xi} \tag{6}$$

**Retraction**: A retraction on manifold $\mathcal{M}$ at a point $x \in \mathcal{M}$ is a smooth map $R_x : \mathrm{T}_x\mathcal{M} \to \mathcal{M}$ such that the zero tangent vector maps to $x$, and its differential at the zero tangent vector is the identity map (7) [22].

$$R_x(0) = x \tag{7a}$$

$$\mathrm{D}R_x(0)[v] = v \tag{7b}$$

**Embedded Submanifold**: A subset $\overline{\mathcal{M}}$ of a manifold $\mathcal{M}$ is an embedded submanifold of $\mathcal{M}$ if and only if for some fixed integer $k \geqslant 1$ and for each point $x \in \overline{\mathcal{M}}$, there exists a neighborhood $\mathcal{U}$ of $x$ in $\mathcal{M}$ and a smooth function $h : \mathcal{U} \to \mathbb{R}^k$ such that (8) holds. Furthermore, the dimension of $\overline{\mathcal{M}}$ is given by (9), and the tangent space of $\overline{\mathcal{M}}$ is a subspace of $\mathrm{T}_x\mathcal{M}$ given by (10) [22].

$$h^{-1}(0) = \overline{\mathcal{M}} \cap \mathcal{U} \tag{8a}$$

$$\mathrm{rank}\mathrm{D}h(x) = k \tag{8b}$$

$$\dim \overline{\mathcal{M}} = \dim \mathcal{M} - k \tag{9}$$

$$\mathrm{T}_x\overline{\mathcal{M}} = \ker \mathrm{D}h(x) \subseteq \mathrm{T}_x\mathcal{M} \tag{10}$$

## III. Problem Formulation

We consider the general nonlinear equality constrained optimization problem (11) with $p$ variables, $m$ cost terms, and $n$ constraints. Each variable $x_k \in \mathcal{M}_k$ is a manifold variable, e.g., a 3D pose variable on the SE(3) manifold. The set of all variables is represented by $X = (x_1, \cdots, x_p)$, which belongs to the product manifold of all individual manifolds $\mathcal{M} = \mathcal{M}_1 \times \ldots \times \mathcal{M}_p$. The cost function (11a) is a sum of the $m$ cost terms. Each cost term $f_i(X_i^{\mathrm{f}})$ is a function on a subset of variables $X_i^{\mathrm{f}} = \{x_k\}_{k \in \mathcal{I}_i^{\mathrm{f}}}$, and each constraint is an equality (11b) involving a subset of variables $X_j^{\mathrm{h}} = \{x_k\}_{k \in \mathcal{I}_j^{\mathrm{h}}}$, where $\mathcal{I}_i^{\mathrm{f}}$ and $\mathcal{I}_j^{\mathrm{h}}$ represent the index set of variables involved in the $i^{th}$ cost term and $j^{th}$ constraint, respectively. We assume that all cost functions $f_i(X_i^{\mathrm{f}})$ and constraint functions $h_j(X_j^{\mathrm{h}})$ are $C^1$ differentiable.

$$\underset{X \in \mathcal{M}}{\arg\min} \sum_{i=1}^{m} f_i(X_i^{\mathrm{f}}) \tag{11a}$$

$$\text{s.t.} \quad h_j(X_j^{\mathrm{h}}) = 0 \quad \text{for } j = 1, \ldots, n \tag{11b}$$

We use a factor graph [23], [24] to represent an equality constrained optimization problem (11) (Fig. 1a). Each variable $x_k$ is represented as a variable node. Each cost term $f_i(X_i^{\mathrm{f}})$ and each constraint $h_j(X_j^{\mathrm{h}}) = 0$ is represented by a factor node connected to its respective variables $X_i^{\mathrm{f}}$ or $X_j^{\mathrm{h}}$ involved in the cost or constraint.

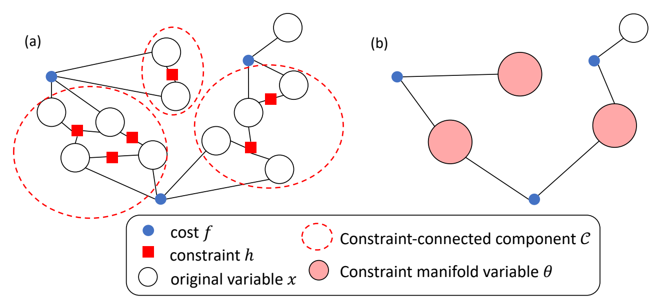


Fig. 1. (a) Factor graph representation of an equality constrained optimization problem; the constraint-connected components (CCC) are marked with dotted ellipses. (b) Factor graph representation of the transformed unconstrained manifold optimization problem. Notice that each CCC is replaced by a constraint manifold variable, and cost factors on constrained variables are replaced with ones on constraint manifold variables.

### A. Problem Transformation

Our goal is to formulate the original equality constrained optimization problem as an unconstrained manifold optimization problem. By analyzing the graph connectivity, we can identify sets of variables connected by constraint factors. We define the tuple of a set of such variables and the constraints among them as a *constraint-connected component* (CCC), represented by $\mathcal{C}_c = (X_c^{\mathcal{C}}, H_c(X_c^{\mathcal{C}}) = 0)$, where $c$ is the index for the CCC, $X_c^{\mathcal{C}} = \{x_k\}_{k \in \mathcal{I}_c^{\mathcal{C}}}$ is the set of variables in the CCC with the variable index set $\mathcal{I}_c^{\mathcal{C}}$, and $H_c(X_c^{\mathcal{C}}) = \{h_j(X_j^{\mathrm{h}})\}_{\mathcal{I}_j^{\mathrm{h}} \subseteq \mathcal{I}_c^{\mathcal{C}}} = 0$ is the vertical concatenation of constraints that only involve variables in the CCC.

Each constraint-connected component $\mathcal{C}_c$ is then replaced by a *constraint manifold* variable $\theta_c \in \overline{\mathcal{M}}_c$. The constraint manifold $\overline{\mathcal{M}}_c$ is a subset of the product manifold $\widetilde{\mathcal{M}}_c$ (12), and it represents the feasible values of $X_c^{\mathcal{C}}$ (13). A recovery function $r_k : \overline{\mathcal{M}}_c \to \mathcal{M}_k$ (14) is defined that recovers the value of any original variable $x_k$ in the CCC from the constraint manifold variable $\theta_c$. After the replacement of constrained variables, the new variables $\Theta$ include the unconstrained variables and constraint manifold variables, and we denote the domain of the new problem as $\overline{\mathcal{M}}$.

$$\widetilde{\mathcal{M}}_c = \bigtimes_{k \in \mathcal{I}_c^{\mathcal{C}}} \mathcal{M}_k \tag{12}$$

$$\overline{\mathcal{M}}_c = \{X_c^{\mathcal{C}} \in \widetilde{\mathcal{M}}_c : H_c(X_c^{\mathcal{C}}) = 0\} \tag{13}$$

$$r_k(\theta_c) = x_k \tag{14}$$

Each cost factor $f_i(X_i^{\mathrm{f}})$ of the original problem that involves constrained variables needs to be updated as an equivalent factor $\overline{f}_i(\Theta_i^{\mathrm{f}})$ (15a) on the new variables, where $\Theta_i^{\mathrm{f}}$ represents the set of new variables involved in the new cost factor. The new factor is created by substituting each constrained variable $x_k$ with its corresponding recovery function (15b), while the unconstrained variables are unchanged.

$$\overline{f}_i(\Theta_i^{\mathrm{f}}) = f_i(X_i^{\mathrm{f}}) \tag{15a}$$

$$X_i^{\mathrm{f}} = \{r_k(\theta_c)\}_{k \in \mathcal{I}_i^{\mathrm{f}}} \tag{15b}$$

The result is an unconstrained manifold optimization problem (16), with its corresponding factor graph representation shown in Fig. 1b. To run manifold optimization on the transformed problem, we still need to define the tangent spaces and retraction operations of the constraint manifolds.

$$\underset{\Theta \in \overline{\mathcal{M}}}{\arg\min} \sum_{i=1}^{m} \overline{f}_i(\Theta_i^{\mathrm{f}}) \tag{16}$$

# IV. Constraint Manifold

The constraint manifold $\overline{\mathcal{M}}_c$ defined on a constraint-connected component $\mathcal{C}_c$ is a sub-manifold of the product manifold $\widetilde{\mathcal{M}}_c$ (12), assuming the Jacobian matrix of constraints $J_{H_c}(X_c^{\mathcal{C}})$ is always full rank. Though the assumption holds true in most cases, certain rank deficient cases may exist which we leave for future study. We will then use the submanifold properties to derive the tangent spaces and retraction operations for the constraint manifolds.

## *A. Tangent Space*

We first compute the tangent space of the product manifold $\widetilde{\mathcal{M}}_c$, then derive the tangent space of the constraint manifold as its subspace. The tangent space of the product manifold $\widetilde{\mathcal{M}}_c$ at $X_c^{\mathcal{C}} \in \widetilde{\mathcal{M}}_c$ is the product space of the tangent space on each original manifold $\mathrm{T}_{x_k}\mathcal{M}_k$ (17). The basis of the tangent space $\mathcal{B}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c$ is formed by concatenating the basis of each original manifold $\mathcal{B}_{x_k}\mathcal{M}_k$ (18).

$$\mathrm{T}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c = \bigtimes_{k \in \mathcal{I}_c^{\mathcal{C}}} \mathrm{T}_{x_k}\mathcal{M}_k \tag{17}$$

$$\mathcal{B}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c = \{\mathcal{B}_{x_k}\mathcal{M}_k\}_{k \in \mathcal{I}_c^{\mathcal{C}}} \tag{18}$$

The tangent space of the constraint manifold $\overline{\mathcal{M}}_c$ can be found using the submanifold properties (10). At a point on the constraint manifold $\theta_c \in \overline{\mathcal{M}}_c$, the tangent space $\mathrm{T}_{\theta_c}\overline{\mathcal{M}}_c$ is the kernel space of the constraint function differential (19), where $X_c^{\mathcal{C}}$ represents the equivalent original variables to $\theta_c$. A basis of the tangent space can be found by computing the null space of the constraint Jacobian matrix $J_{H_c}(X_c^{\mathcal{C}})$ as in (20), with the matrix $N = \mathrm{Nul}\ J_{H_c}(X_c^{\mathcal{C}})$ denoting the null space basis of the constraint Jacobian.

$$\begin{aligned} \mathrm{T}_{\theta_c}\overline{\mathcal{M}}_c &= \ker \mathrm{D}H_c(X_c^{\mathcal{C}}) \\ &= \{v \in \mathrm{T}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c : \mathrm{D}H_c(X_c^{\mathcal{C}})[v] = 0\} \\ &= \{\mathcal{B}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c \cdot \xi : J_{H_c}(X_c^{\mathcal{C}}) \cdot \xi = 0\} \end{aligned} \tag{19}$$

$$\mathcal{B}_{\theta_c}\overline{\mathcal{M}}_c = \mathcal{B}_{X_c^{\mathcal{C}}}\widetilde{\mathcal{M}}_c \cdot N \tag{20}$$

## *B. Retraction*

The retraction on the product manifold $\widetilde{\mathcal{M}}_c$ can be formulated as performing retraction on each individual manifold $\mathcal{M}_k$ with their corresponding tangent vector component $v_k$ (21). We then provide three ways to perform retraction on the constraint manifold $\overline{\mathcal{M}}_c$.

$$\mathrm{R}_{X_c^{\mathcal{C}}}(v) = \{\mathrm{R}_{x_k}(v_k)\}_{k \in \mathcal{I}_c^{\mathcal{C}}} \tag{21}$$

*1) Metric Projection:* A common way to define retraction is the metric projection [22]. We first perform retraction on the product manifold $\mathrm{R}_{X_c^{\mathcal{C}}}(v)$, then project the point onto the constraint manifold $\overline{\mathcal{M}}_c$ using metric projection as formulated in (22), where $\mathrm{dist}(\cdot,\cdot)_{\mathcal{M}}$ is the Riemannian distance between any two points on the manifold $\mathcal{M}$. Notice that (22) is itself a constrained optimization problem, and can be almost as hard to solve as the original constrained optimization problem (11) in certain cases.

$$\begin{aligned} \mathrm{R}_{\theta_c}(v) = &\underset{Y \in \widetilde{\mathcal{M}}_c}{\arg\min}\, \mathrm{dist}(Y, \mathrm{R}_{X_c^{\mathcal{C}}}(v))^2_{\widetilde{\mathcal{M}}_c} \\ &\text{s.t.} \quad H_c(Y) = 0 \end{aligned} \tag{22}$$

*2) Approximate Metric Projection:* in practice, inspired by [25], we can solve an unconstrained optimization problem (23) instead, which minimizes the sum-of-squares of the constraint violations with the Levenberg-Marquardt method. Initial values for unconstrained optimization are chosen as the retraction on the product manifold $\mathrm{R}_{X_c^{\mathcal{C}}}(v)$.

$$\mathrm{R}_{\theta_c}(v) = \underset{Y \in \widetilde{\mathcal{M}}_c}{\arg\min} \|H_c(Y)\|^2 \tag{23a}$$

$$Y_{init} = \mathrm{R}_{X_c^{\mathcal{C}}}(v) \tag{23b}$$

*3) Retract Basis Variables:* We select a set of variables $X_c^{\mathcal{B}} = \{x_k\}_{k \in \mathcal{I}_c^{\mathcal{B}}}$ with variable indices $\mathcal{I}_c^{\mathcal{B}}$ as the basis variables, such that $\dim X_c^{\mathcal{B}} = \dim \overline{\mathcal{M}}_c$. The tangent vector is used to retract the basis variables, and the rest of the variables are matched to satisfy the constraints. The retraction solves the unconstrained optimization in (24).

$$\begin{aligned} \mathrm{R}_{\theta_c}(v) = \underset{Y \in \widetilde{\mathcal{M}}_c}{\arg\min} &\sum_{k \in \mathcal{I}_c^{\mathcal{B}}} \mathrm{dist}(y_k, \mathrm{R}_{x_k}(v_k))^2_{\mathcal{M}_k} \\ &+ \|H_c(Y)\|^2 \end{aligned} \tag{24}$$

## V. Optimization on Constraint Manifolds

### A. Gradient of the New Cost Function

To solve the manifold optimization problem (16), we need to differentiate the new cost functions $\overline{f}_i(\Theta_i^{\mathrm{f}})$. The Jacobian of the new cost function (26) can be found by applying the chain rule on (15). The Jacobian of recover function is given in (25), where $S_k$ is the selection matrix that selects the rows in the null space matrix $N$ corresponding to variable $x_k$.

$$J_{r_k}(\theta_c) = S_k \cdot N \tag{25}$$

$$J_{\overline{f}}(\theta_c) = \sum_{k \in \mathcal{I}_c^{\mathcal{C}} \cap \mathcal{I}_i^{\mathrm{f}}} J_{r_k}(\theta_c) \cdot J_f(x_k) \tag{26}$$

### B. Infeasible Methods

The efficiency of manifold optimization can be further improved by developing an "infeasible method". We notice that the retraction operation on each constraint manifold requires solving an optimization problem, which may take several iterations to converge to a solution that strictly satisfies the constraints. We draw intuition from [25] that "manifold optimization" can still converge without performing the exact retraction. In our infeasible method, the retraction optimization (23, 24) is stopped before convergence, generating some infeasible values $\widehat{\theta}_c \notin \overline{\mathcal{M}}_c$, which are on a different manifold $\widehat{\theta}_c \in \widehat{\mathcal{M}}_c$ as defined in (27). Notice that the approximate manifold $\widehat{\mathcal{M}}_c$ is also a constraint manifold, while it differs from $\overline{\mathcal{M}}_c$ in the constant terms of the constraint equations. The linear update is computed as a tangent vector on the approximate manifold $\widehat{\mathcal{M}}_c$ instead. The retraction, however, will still try to retract onto the constraint manifold $\overline{\mathcal{M}}_c$, i.e., forcing $H_c(X_c^{\mathcal{C}}) = 0$ instead of $H_c(X_c^{\mathcal{C}}) - H_c(\widehat{\theta}_c) = 0$, to ensure convergence. A visual example of the infeasible manifold optimization method on the $\mathrm{SO}(2)$ manifold is shown in Fig. 2.

$$\widehat{\mathcal{M}}_c = \{X_c^{\mathcal{C}} \in \widetilde{\mathcal{M}}_c : H_c(X_c^{\mathcal{C}}) - H_c(\widehat{\theta}_c) = 0\} \tag{27}$$

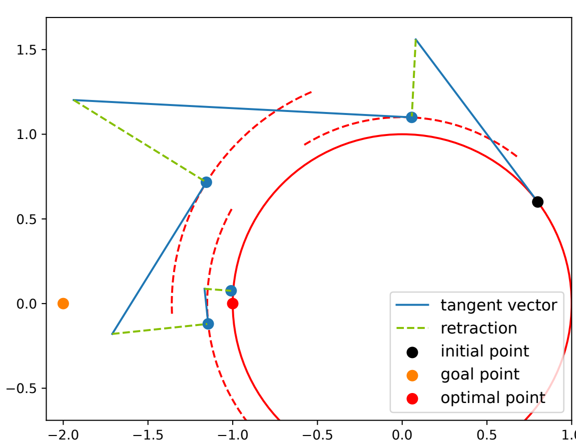


Fig. 2. Example of infeasible optimization on the $\mathrm{SO}(2)$ manifold. The constrained optimization problem minimizes the distance of a point to the goal point, while staying on the unit circle (solid red circle). The approximate manifolds at each iteration are represented by dashed red arcs. Notice that the optimization converges to the true solution.

### C. Relationship with Symbolic Variable Elimination

There is a connection between constraint manifold optimization and symbolic variable elimination. For a constraint manifold $\overline{\mathcal{M}}_c$ with a set of basis variables $X_c^{\mathcal{B}}$, we can construct its tangent space basis such that each basis vector $\overline{b} \in \mathcal{B}_{\theta_c}\overline{\mathcal{M}}_c$ corresponds to a tangent space basis vector $b \in \mathcal{B}_{x_k}\mathcal{M}_k$ of a basis variable $x_k \in X_c^{\mathcal{B}}$. Therefore, we are able to pick freely at the tangent spaces of the basis variables, while the other dimensions of the tangent vector $v \in \mathrm{T}_{\theta_c}\overline{\mathcal{M}}_c$ are chosen to satisfy the linearized constraints $\mathrm{D}H_c(X_c^{\mathcal{C}})[v] = 0$. With the basis variable retraction, the linear update is applied directly on the basis variables, while the rest of the variables are chosen to satisfy the constraints $H_c(X_c^{\mathcal{C}}) = 0$. In this way, the manifold optimization method is equivalent to applying symbolic elimination that eliminates all the non-basis variables in the CCC using the constraints.

## VI. Experiments & Results

### A. Scenarios

We conduct experiments on six robotic inference and planning problems with equality constraints.

*1) Multi-Vehicle Trajectory Estimation:* We consider a two-vehicle state estimation problem as in [26]. Two vehicles collect odometry measurements and inter-vehicle measurements while navigating through the environment. Two types of inter-vehicle measurements are considered: (1) relative pose measurements ("Connected Poses") (2) range measurements ("Range Constraint"). The inter-vehicle measurements are precise, and therefore treated as constraints. The goal is to find the maximum a posterior (MAP) estimate of the trajectories that satisfy the inter-vehicle constraints. A factor graph representation of the constrained optimization problem is shown in Fig. 3.

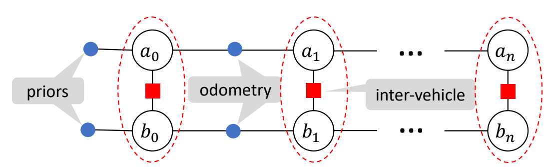


Fig. 3. Factor graph for multi-vehicle estimation problem. The poses of the two vehicles at time step $t$ are represented as variables $a_t$, $b_t$, respectively. Inter-vehicle constraints are enforced for each time step, while odometry measurements are imposed across time steps.

*2) Quadruped State Estimation:* We consider a quadruped state estimation problem on simulated trajectories as in [4]. The quadruped is equipped with IMU measurements on the torso link, joint angle measurements with uncertainty of $1^\circ$ at each joint, and contact measurements on each foot. When contact happens, we assume the contact point is static with small uncertainties that counts for slippery and rolling contacts. We explicitly model as variables the link poses, joint angles, contact points at each time step, and enforce the kinematics constraints at each joint [1], and the relative position constraints of the contact points with respect to the foot links. The constrained MAP inference problem has a factor graph representation in Fig. 4.

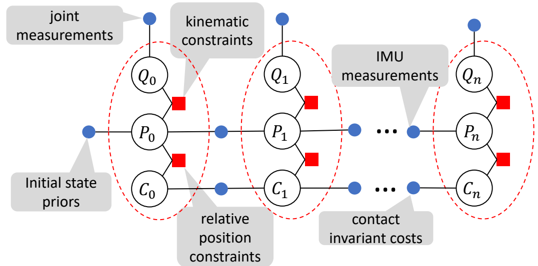


Fig. 4. Simplified factor graph for the quadruped state estimation problem. The joint angles, link poses, contact points at time step $t$ are represented by $Q_t$, $P_t$, $C_t$, respectively. Joint measurements are imposed on the joint variables at each step; integrated imu measurements are imposed on the torso poses in consecutive time steps; contact invariant objectives are imposed on contact points in consecutive time steps if contact happens.

*3) Robot Kino-dynamic Planning:* The kinodynamic trajectory planning problems of (1) a cart-pole system, (2) a cable-driven parallel robot [27], and (3) a quadruped with contact are considered. The goal for the trajectory planning problem is to find the trajectory that achieves a final target state with minimum accumulated motor torques. The target state for the cart-pole system is to rotate up the pole and balance it, while the target states for the cable robot and quadruped are the specified end-effector/torso poses as shown in Figure 5a. The continuous trajectory is represented using discrete time steps with Trapezoidal collocation [28]. For each time step, we explicitly model as variables the pose, twist, and acceleration of each link; the angle, velocity, and acceleration of each joint; and the torque and wrench on each joint. The constraints of the problems include the kino-dynamic constraints as in [29], as well as the constraints specifying the initial state. The costs include the costs for achieving the final state, minimizing motor torque actuations, and satisfying the collocation scheme. A factor graph representation of the problem is shown in Figure 5b.

## B. Performance Benchmark

We evaluate the constraint manifold optimization method against state-of-the-art constrained optimization methods. As a baseline, the soft constraint method treats constraints as part of the cost function, and minimizes the merit function (28) with a weighting coefficient $\mu = 10000$. We also implement the penalty method and the augmented Lagrangian method following [30] as baselines. We evaluated both the feasible and infeasible methods for constraint manifold optimization. In feasible methods, the retraction optimization problem (23, 24) are optimized until convergence; in infeasible methods, the retraction optimization problem is only executed with one Levenberg-Marquardt iteration. Approximate metric projection (23) is used as retraction for multi-vehicle state estimation tasks, while all other scenarios uses basis variable retraction (24). We employ the Levenberg-Marquardt method for manifold optimization. All methods are implemented using the GTSAM [31] library.

$$\underset{X\in\mathcal{M}}{\arg\min} \sum_{i=1}^{m} f_i(X_i^{\mathrm{f}}) + \mu \sum_{j=1}^{n} \|h_j(X_j^{\mathrm{h}})\|^2 \tag{28}$$

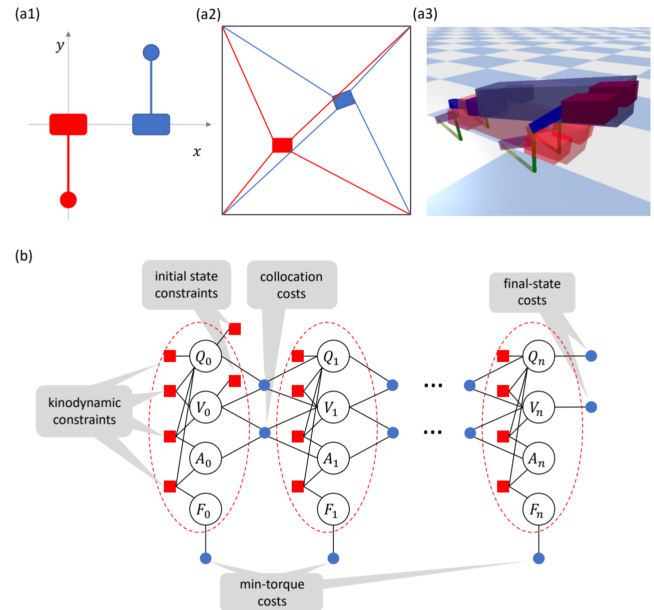


Fig. 5. (a) Initial state (red) and target final state (blue) for (1) cart-pole system, (2) cable-driven parallel robot, (3) quadruped with contact. (b) Simplified factor graph representation of the constrained optimization for trajectory planning problem. The poses and angles, velocities, accelerations, wrenches and torques at time step $t$ are represented with $Q_t$, $V_t$, $A_t$, $F_t$, respectively. Kinodynamic constraints and min-torque costs are enforced for each time step. Collocation is imposed across consecutive steps.

For all scenarios, we show the optimization problem size (as function dimension $\times$ variable dimension), optimization time, number of nonlinear iterations, total constraint-violation (as the norm of constraint violation vector in SI units) and final cost in Table I. As a gradient-based manifold optimizer iteratively linearizes cost functions, solves linear systems, and applies linear updates to variables, we also show the average timing results of the subtasks in Table II. In multi-vehicle state estimation scenarios, the trajectories generated by all methods are similar, and have the same average pose error (APE). In the quadruped state estimation scenario, the trajectories with manifold optimization achieve a smaller APE of 0.307 compared to 0.316 with the soft constraints, as evaluated on 4 different simulated trajectories.

## VII. Discussion

The manifold optimization method with constraint manifold is overall more efficient than the other methods. First, the manifold optimization method has a smaller problem size, since the enforcement of constraints within the manifolds reduces both the dimension of factors and the dimension of variables. Second, the manifold optimization method does not require iteratively solving unconstrained optimization problems compared to the penalty method and the augmented Lagrangian method. Third, the manifold optimization method converges faster than the soft constraint method due to its better numerical properties, as indicated by its smaller number of nonlinear iterations. The soft constraints, on the other hand, may suffer from scaling issues [11], since the large weighting factor $\mu$ assigned to the constraint factors can result in a poorly conditioned problem.

TABLE I

COMPARISON OF CONSTRAINT MANIFOLD OPTIMIZATION WITH CONSTRAINED OPTIMIZATION METHODS ON MULTIPLE SCENARIOS

| Scenario | Method | Problem Size | Time (s) | Iters | Constraint Vio (m, rad, etc.) | cost |
|---|---|---|---|---|---|---|
| Connected Poses | Soft Constraint | 909 × 606 | 0.0263 | 16 | 1.61e-02 | 141.43 |
| | Penalty Method | 909 × 606 | 0.1578 | 125 | 9.90e-03 | 142.43 |
| | Augmented Lagrangian | 909 × 606 | 0.1126 | 91 | 2.70e-03 | 143.72 |
| | **Constraint Manifold (F)** | 606 × 303 | 0.0237 | 16 | 1.45e-09 | 144.06 |
| | **Constraint Manifold (I)** | 606 × 303 | 0.0145 | 7 | 5.58e-13 | 144.06 |
| Range Constraint | Soft Constraint | 707 × 606 | 0.0907 | 68 | 8.13e-03 | 40.96 |
| | Penalty Method | 707 × 606 | 0.2302 | 193 | 5.04e-03 | 38.40 |
| | Augmented Lagrangian | 707 × 606 | 0.6959 | 577 | 6.57e-03 | 40.29 |
| | **Constraint Manifold (F)** | 606 × 505 | 0.0698 | 41 | 1.26e-11 | 41.63 |
| | **Constraint Manifold (I)** | 606 × 505 | 0.0650 | 41 | 1.26e-11 | 41.63 |
| Quadruped State Estimation | Soft Constraint | 34338 × 33000 | 1.8528 | 31 | 1.24e-02 | 558.3 |
| | Penalty Method | 34338 × 33000 | 3.1221 | 40 | 8.33e-03 | 610.8 |
| | Augmented Lagrangian | 34338 × 33000 | 3.9016 | 56 | 1.52e-02 | 536.7 |
| | **Constraint Manifold (F)** | 4938 × 3600 | 0.6569 | 5 | 5.95e-12 | 736.9 |
| | **Constraint Manifold (I)** | 4938 × 3600 | 0.6560 | 5 | 5.95e-12 | 736.9 |
| Cart-Pole Planning | Soft Constraint | 17293 × 17286 | 6.1404 | 203 | 1.91e-03 | 782.7 |
| | Penalty Method | 17293 × 17286 | 29.3833 | 945 | 7.74e-06 | 769.0 |
| | Augmented Lagrangian | 17293 × 17286 | 23.3063 | 756 | 4.63e-06 | 771.6 |
| | **Constraint Manifold (F)** | 1006 × 1000 | 3.9304 | 53 | 3.22e-13 | 761.1 |
| | **Constraint Manifold (I)** | 1006 × 1000 | 2.6053 | 53 | 1.77e-11 | 761.1 |
| Cable Robot Planning | Soft Constraint | 68000 × 62000 | 3.5738 | 12 | 1.80e-02 | 1.63 |
| | Penalty Method | 68000 × 62000 | 11.1374 | 34 | 1.12e-04 | 1.66 |
| | Augmented Lagrangian | 68000 × 62000 | 7.6237 | 22 | 1.63e-04 | 1.66 |
| | **Constraint Manifold (F)** | 21988 × 15988 | 2.5850 | 4 | 1.34e-12 | 1.66 |
| | **Constraint Manifold (I)** | 21988 × 15988 | 2.4474 | 4 | 3.06e-12 | 1.66 |
| Quadruped Planning | Soft Constraint | 14266 × 13950 | 6.0160 | 203 | 2.84e-05 | 105.72 |
| | Penalty Method | 14266 × 13950 | 90.5319 | 3049 | 9.35e-08 | 48.53 |
| | Augmented Lagrangian | 14266 × 13950 | 74.3907 | 2436 | 3.69e-06 | 47.16 |
| | **Constraint Manifold (F)** | 874 × 558 | 6.1469 | 25 | 7.19e-11 | 47.09 |
| | **Constraint Manifold (I)** | 874 × 558 | 5.4528 | 29 | 3.56e-08 | 37.83 |

TABLE II

TIMING RESULTS FOR OPTIMIZATION SUBTASKS IN MANIFOLD OPTIMIZATION AND SOFT CONSTRAINT METHOD

| Scenario | Method | Average Time per Nonlinear Iteration (ms) | | | | |
|---|---|---|---|---|---|---|
| | | Linearization | Solve Linear System | Apply Linear Update | Other | Total |
| Range Constraint | Soft Constraint | 0.052 | 0.929 | $\approx 0$ | 0.353 | 1.334 |
| | **Constraint Manifold (F)** | 0.083 | 0.467 | 0.910 | 0.242 | 1.702 |
| | **Constraint Manifold (I)** | 0.079 | 0.478 | 0.814 | 0.214 | 1.585 |
| Cart-Pole Planning | Soft Constraint | 1.03 | 24.14 | 0.71 | 4.36 | 30.25 |
| | **Constraint Manifold (F)** | 1.11 | 1.14 | 70.64 | 1.27 | 74.16 |
| | **Constraint Manifold (I)** | 1.08 | 1.19 | 44.84 | 2.04 | 49.16 |

The manifold optimization also provides results with better optimality than the other methods. The constraint violation is much smaller with the manifold optimization, since the constraints are enforced in each retraction operation. In the cart-pole and quadruped planning scenarios, the manifold optimization manages to generate solutions with both smaller costs and smaller constraint violations, which implies that the manifold optimization problem is better conditioned with fewer local minimums.

By inspecting the subtask timing results in Table II, we discover that the manifold optimization saves time in solving the linear system due to its smaller problem size, while it increases the time to apply the linear update. The overhead mostly results from the additional work needed to solve the retraction optimization problem (23, 24). Luckily, this overhead is reduced with the infeasible method, which consistently converges to similar solutions as the feasible method. Even though parallel computation is not implemented for the experiments, the tangent spaces and retraction operations of all constraint manifolds can be computed in parallel, which can further speed up optimization.

For future work, we aim to evaluate the constraint manifold optimization on real-world large-scale problems, study how to formulate the constraint manifold around the rank deficient conditions of the constraint Jacobian(i.e., when (8b) does not hold), find ways to further improve the efficiency in retraction operations, and incorporate inequality constraints.

## VIII. CONCLUSION

We develop a constraint manifold optimization framework to solve constrained inference and planning problems. Constraint manifolds with tangent spaces and retractions are formulated to represent sets of variables subject to equality constraints. We further improve the efficiency of constraint manifold optimization by developing the infeasible methods. In multiple scenarios, our manifold optimization generates results with improved optimality and efficiency compared to state-of-the-art constrained optimization methods.

## REFERENCES


[1] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*, 1st ed. USA: Cambridge University Press, 2017.

[2] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[3] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4507–4514, 2019.

[4] V. Agrawal, S. Bertrand, R. Griffin, and F. Dellaert, "Proprioceptive state estimation of legged robots with kinematic chain modeling," 2022. [Online]. Available: https://arxiv.org/abs/2209.05644

[5] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.

[6] Y. Zhang, G. Chen, A. Rutkowski, and F. Dellaert, "Efficient range-constraint manifold optimization with application to cooperative navigation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[7] A. Cunningham, M. Paluri, and F. Dellaert, "Ddf-sam: Fully distributed slam using constrained factor graphs," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3025–3030.

[8] D.-N. Ta, M. Kobilarov, and F. Dellaert, "A factor graph approach to estimation and model predictive control on unmanned aerial vehicles," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2014, pp. 181–188.

[9] S. Yang, G. Chen, Y. Zhang, H. Choset, and F. Dellaert, "Equality constrained linear optimal control with factor graphs," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9717–9723.

[10] B. Bazzana, T. Guadagnino, and G. Grisetti, "Handling constrained optimization in factor graphs for autonomous navigation," *arXiv preprint arXiv:2208.06325*, 2022.

[11] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[12] R. H. Byrd, F. E. Curtis, and J. Nocedal, "An inexact sqp method for equality constrained optimization," *SIAM Journal on Optimization*, vol. 19, no. 1, pp. 351–369, 2008.

[13] J. T. Betts, *Practical methods for optimal control and estimation using nonlinear programming*. SIAM, 2010.

[14] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ: Princeton University Press, 2008.

[15] S. Traversaro, S. Brossette, A. Escande, and F. Nori, "Identification of fully physical consistent inertial parameters using optimization on manifolds," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5446–5451.

[16] M. Watterson, S. Liu, K. Sun, T. Smith, and V. Kumar, "Trajectory optimization on manifolds with applications to so (3) and r3xs2." in *Robotics: Science and Systems*, 2018.

[17] M. Zhang, X. Zuo, Y. Chen, Y. Liu, and M. Li, "Pose estimation for ground robots: On manifold representation, integration, reparameterization, and optimization," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1081–1099, 2021.

[18] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[19] E. B. Dean, "Continuous optimization on constraint manifolds," in *TIMS/ORSA Joint National Meeting*, 1988.

[20] P. Liu, D. Tateo, H. B. Ammar, and J. Peters, "Robot reinforcement learning on the constraint manifold," in *Conference on Robot Learning*. PMLR, 2022, pp. 1357–1366.

[21] J. Ortiz-Haro, J.-S. Ha, D. Driess, and M. Toussaint, "Structured deep generative models for sampling on constraint manifolds in sequential manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 213–223.

[22] N. Boumal, "An introduction to optimization on smooth manifolds," Available online, Aug 2020. [Online]. Available: http://www.nicolasboumal.net/book

[23] F. R. Kschischang, B. J. Frey, and H. . Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[24] F. Dellaert and M. Kaess, *Factor Graphs for Robot Perception*. Now Publishers Inc., August 2017.

[25] P. Ablin and G. Peyré, "Fast and accurate optimization on the orthogonal manifold without retraction," *arXiv preprint arXiv:2102.07432*, 2021.

[26] A. J. Rutkowski, J. E. Barnes, and A. T. Smith, "Path planning for optimal cooperative navigation," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2016, pp. 359–365.

[27] G. Chen, S. Hutchinson, and F. Dellaert, "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, 2022.

[28] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[29] M. Xie, A. Escontrela, and F. Dellaert, "A factor-graph approach for optimization problems with dynamics constraints," *arXiv preprint arXiv:2011.06194*, 2020.

[30] L. Vandenberghe, "Constrained nonlinear least squares," http://www.seas.ucla.edu/ vandenbe/133B/lectures/nllseq.pdf, 2020.

[31] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep., 2012.