

# Change of Optimal Values: A Pre-calculated Metric

Fang Bai

**Abstract**—A variety of optimization problems takes the form of a minimum norm optimization. In this paper, we study the change of optimal values between two incrementally constructed least norm optimization problems, with new measurements included in the second one. We prove an exact equation to calculate the change of optimal values in the linear least norm optimization problem. With the result in this paper, the change of the optimal values can be pre-calculated as a metric to guide online decision makings, without solving the second optimization problem as long the solution and covariance of the first optimization problem are available. The result can be extended to linear least distance optimization problems, and nonlinear least distance optimization with (nonlinear) equality constraints through linearizations. This derivation in this paper provides a theoretically sound explanation to the empirical observations shown in [1]. As an additional contribution, we propose another optimization problem, i.e. aligning two trajectories at given poses, to further demonstrate how to use the metric. The accuracy of the metric is validated with numerical examples, which is quite satisfactory in general (see the experiments in [1] as well), unless in some extremely adverse scenarios. Last but not least, calculating the optimal value by the proposed metric is at least one magnitude faster than solving the corresponding optimization problems directly.

## I. INTRODUCTION

Many robotic applications rely on metrics to make decisions. A metric is usually formed as a pre-calculated scalar which captures certain aspects of the underlying problem, for example the entropy in the information theory, or the cost in an optimization problem.

The most widely used metrics in robotics, at least in estimation tasks, are attributed to the information theoretic metrics [2]. This is a broad class, and we mention several of them for completeness: (1) Fisher Information Matrix (FIM). If the processing noise is Gaussian, the metric always boils down to the evaluation of a proper covariance matrix with respect to its eigen value, trace or determinant [3][4]. (2) Mutual Information. This metric is quite popular which has been extensively used in robotics, for example, the active control/planning strategy [5][6][7][8][9], graph pruning [10][11], online sparse pose-graph construction [12][13], path planning on a pose graph [14] and so forth. (3) Kullback-Leibler Divergence (KL Divergence). While the first two metrics deal merely with uncertainty, KL divergence can incorporate the mean and covariance of two Gaussian distributions together. This is quite similar to the optimal value in the optimization context that we will discuss later, but they are obviously in different tracks. See [15] for an example of graph pruning using KL divergence.

Fang Bai is with Center for Autonomous Systems, University of Technology Sydney (UTS-CAS), Sydney, Australia. Email: fang.bai@yahoo.com, fang.bai@student.uts.edu.au

Besides the information theoretic metrics, in the context of optimization, the optimal value is essentially a metric as well. However, the optimal value was considered dilatory in the past, because the optimal value can only be obtained after solving the underlying problem. This is meaningless because in most applications, we often encounter a bundle of candidate problems and a metric is required to tell us which problem is worth solving. Obviously, if solving the problem is compulsory, the optimal value is unsuitable for tasks of this kind. As a result, the optimal value is basically used for evaluating the optimality of the solution [16][17]. Some authors managed to use the optimal value [18][19] to detect outliers by using an “optimize-revoke” strategy. However, this strategy is quite expensive as many undesirable subproblems are formed and solved in the process.

Recently, for an incremental optimization problem, Bai et al. [1] empirically show that the optimal value can be approximated prior to solving the problem. In [1], we provided an equation for the optimal value, but were unable to prove it exactly. Later in [20], the authors studied incremental least squares optimization, and proved that the optimal value can be pre-calculated exactly for the linear case. The theory extends to nonlinear cases through linearization. However, there is still no clear explanation for the result on the equality constrained optimization in [1].

In this paper, we study the least norm optimization, which is the counterpart of least squares. Analogous to [20], we first derive the result for the linear case which is exact, and then extend it to more general problems like least distance optimization and nonlinear scenarios. The analysis in this paper eventually provides a theoretical sound explanation for the empirical observation in [1]. Moreover, the theoretical result in this paper is actually quite general, which can be applied to other similar problems easily.

Overall, this paper includes the following contributions:

- 1) The change of optimal values in minimum norm optimization (Section II), and its extension to linear least distance optimization (Section III).
- 2) An extension to equality constrained nonlinear least distance optimization on manifold (Section IV), which explains why the change of optimal values can be pre-calculated as empirically shown in the paper [1].
- 3) An example on the cost of aligning trajectories to illustrate the usage of the metric (Section V-B), with numerical experiments in Section VI.

## II. RESULTS ON LEAST NORM OPTIMIZATION

### A. Problem Statement and Standard Form

Let us consider a minimum norm optimization problem

$$\min \mathbf{x}^T \mathbf{x} \quad \text{s.t.} \quad \mathbf{A} \mathbf{x} = \mathbf{b} \quad (1)$$

where the constraints are classified into two parts:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}.$$

This is particularly the case for temporally incremental optimization problems. The constraint  $\mathbf{A}_1 \mathbf{x} = \mathbf{b}_1$  represents those that are already available at a specific time point, while the constraint  $\mathbf{A}_2 \mathbf{x} = \mathbf{b}_2$  can only be obtained after that time point. Let us assume both  $\mathbf{A}_1$  and  $\mathbf{A}_2$  have full row rank.

The optimization problem established using the previously available measurement explicitly writes

$$\min \mathbf{x}^T \mathbf{x} \quad \text{s.t.} \quad \mathbf{A}_1 \mathbf{x} = \mathbf{b}_1. \quad (2)$$

Now at some point, we have solved the problem (2), but not yet problem (1). The question is that: Can we tell something useful about problem (1) from the results of problem (2)?

In this section, we establish the equation on how to calculate the optimal value (of the objective function) of problem (1), without solving it directly.

### B. Classical Results on Minimum Norm Optimization

To proceed, let us recall some classical results of minimum norm optimization, taking problem (1) as an example.

The optimal solution  $\mathbf{x}_{opt}$  is

$$\mathbf{x}_{opt} = \mathbf{A}^\dagger \mathbf{b} = \mathbf{A}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b}.$$

The covariance (with Gaussian assumption) of the optimal solution,  $\text{Cov}(\mathbf{x}_{opt})$ , is the projection matrix to the Null space of  $\mathbf{A}$ , which writes

$$\text{Cov}(\mathbf{x}_{opt}) = \mathbf{P}_{(N(\mathbf{A}))} = \mathbb{I} - \mathbf{A}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{A}.$$

The cost at the optimal solution, termed the optimal value (of the objective function), i.e.  $\mathbf{x}_{opt}^T \mathbf{x}_{opt}$ , writes

$$f_{opt} = \mathbf{b}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b}.$$

We will use the Schur complement [21] to decompose the below block matrix

$$\begin{bmatrix} \mathbf{U} & \mathbf{P} \\ \mathbf{Q} & \mathbf{V} \end{bmatrix} = \begin{bmatrix} \mathbb{I} & \mathbb{O} \\ \mathbf{Q} \mathbf{U}^{-1} & \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbf{U} & \mathbb{O} \\ \mathbb{O} & \mathbf{V} - \mathbf{Q} \mathbf{U}^{-1} \mathbf{P} \end{bmatrix} \begin{bmatrix} \mathbb{I} & \mathbf{U}^{-1} \mathbf{P} \\ \mathbb{O} & \mathbb{I} \end{bmatrix}. \quad (3)$$

Eq. (3) stands as long as  $\mathbf{A}$  is invertible.

### C. Derivation of the Main Result

The derivation consists of a manipulation of matrices via linear algebra laws. Basically, we will expand the optimal value of problem (1), i.e.,  $\mathbf{b}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b}$ , and examine its components.

The two matrices below are useful to express the Schur complement of  $\mathbf{A} \mathbf{A}^T$ . Let us define two invertible matrices:

$$\mathbf{W} = \mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T = \mathbf{A}_2 [\mathbb{I} - \mathbf{A}_1^T (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} \mathbf{A}_1] \mathbf{A}_2^T \quad (4)$$

$$\mathbf{H} = \begin{bmatrix} \mathbb{I} & \mathbb{O} \\ \mathbf{A}_2 \mathbf{A}_1^T (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} & \mathbb{I} \end{bmatrix} = \begin{bmatrix} \mathbb{I} & \mathbb{O} \\ \mathbf{A}_2 \mathbf{A}_1^\dagger & \mathbb{I} \end{bmatrix}. \quad (5)$$

Let us apply Schur complement (3) to the matrix  $\mathbf{A} \mathbf{A}^T$ . The result writes

$$\begin{aligned} \mathbf{A} \mathbf{A}^T &= \begin{bmatrix} \mathbf{A}_1 \mathbf{A}_1^T & \mathbf{A}_1 \mathbf{A}_2^T \\ \mathbf{A}_2 \mathbf{A}_1^T & \mathbf{A}_2 \mathbf{A}_2^T \end{bmatrix} \\ &= \mathbf{H} \begin{bmatrix} \mathbf{A}_1 \mathbf{A}_1^T & \mathbb{O} \\ \mathbb{O} & \mathbf{A}_2 \mathbf{A}_2^T - \mathbf{A}_2 \mathbf{A}_1^T (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} \mathbf{A}_1 \mathbf{A}_2^T \end{bmatrix} \mathbf{H}^T \\ &= \mathbf{H} \begin{bmatrix} \mathbf{A}_1 \mathbf{A}_1^T & \mathbb{O} \\ \mathbb{O} & \mathbf{W} \end{bmatrix} \mathbf{H}^T. \end{aligned}$$

Noting that  $\mathbf{H}^{-1} \mathbf{b}$  can be written as

$$\mathbf{H}^{-1} \mathbf{b} = \begin{bmatrix} \mathbb{I} & \mathbb{O} \\ -\mathbf{A}_2 \mathbf{A}_1^\dagger & \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ -\mathbf{A}_2 \mathbf{A}_1^\dagger \mathbf{b}_1 + \mathbf{b}_2 \end{bmatrix},$$

then we can finally expand  $\mathbf{b}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b}$  by a straightforward matrix calculation as follow

$$\begin{aligned} f^{**} &= \mathbf{b}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b} \\ &= \mathbf{b}^T \mathbf{H}^{-T} \begin{bmatrix} \mathbf{A}_1 \mathbf{A}_1^T & \mathbb{O} \\ \mathbb{O} & \mathbf{W} \end{bmatrix}^{-1} \mathbf{H}^{-1} \mathbf{b} \\ &= (\mathbf{H}^{-1} \mathbf{b})^T \begin{bmatrix} (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} & \mathbb{O} \\ \mathbb{O} & \mathbf{W}^{-1} \end{bmatrix} \mathbf{H}^{-1} \mathbf{b} \\ &= \begin{bmatrix} \mathbf{b}_1 \\ -\mathbf{A}_2 \mathbf{A}_1^\dagger \mathbf{b}_1 + \mathbf{b}_2 \end{bmatrix}^T \begin{bmatrix} (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} & \mathbb{O} \\ \mathbb{O} & \mathbf{W}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 \\ -\mathbf{A}_2 \mathbf{A}_1^\dagger \mathbf{b}_1 + \mathbf{b}_2 \end{bmatrix} \\ &= \mathbf{b}_1^T (\mathbf{A}_1 \mathbf{A}_1^T)^{-1} \mathbf{b}_1 + (\mathbf{A}_2 \mathbf{A}_1^\dagger \mathbf{b}_1 - \mathbf{b}_2)^T \mathbf{W}^{-1} (\mathbf{A}_2 \mathbf{A}_1^\dagger \mathbf{b}_1 - \mathbf{b}_2) \\ &= f^* + \underbrace{(\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2)^T [\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T]^{-1} (\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2)}_{\Delta f}. \end{aligned}$$

**Conclusion:** Therefore the **change of optimal values**,  $\Delta f = f^{**} - f^*$ , from problem (1) to problem (2) can be compactly written as

$$\Delta f = (\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2)^T [\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T]^{-1} (\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2). \quad (6)$$

In (6), the unknown piece of information,  $\mathbf{x}_1^*$  and  $\text{Cov}(\mathbf{x}_1^*)$ , can be obtained by solving the problem (2). In other words, given  $\mathbf{x}_1^*$  and its covariance  $\text{Cov}(\mathbf{x}_1^*)$ , we can calculate  $\Delta f$  through Eq. (6), then we calculate the the optimal value of problem (1) simply by  $f^{**} = f^* + \Delta f$ , without solving it!

## III. EXTENSION TO LEAST DISTANCE OPTIMIZATION

In this section, we consider the problem of a least distance optimization, whose standard form writes

$$\min (\mathbf{H} \mathbf{x} - \mathbf{h})^T \Sigma^{-1} (\mathbf{H} \mathbf{x} - \mathbf{h}) \quad \text{s.t.} \quad \mathbf{A} \mathbf{x} = \mathbf{b}. \quad (7)$$

In (7), we assume  $\mathbf{H}$ ,  $\Sigma$  to be invertible. In this case, by letting  $\mathbf{y} = \Sigma^{-\frac{1}{2}} (\mathbf{H} \mathbf{x} - \mathbf{h})$ , i.e.  $\mathbf{x} = \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \mathbf{y} + \mathbf{H}^{-1} \mathbf{h}$ , (7) can be converted to a least norm optimization

$$\min \mathbf{y}^T \mathbf{y} \quad \text{s.t.} \quad \mathbf{A} \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \mathbf{y} = \mathbf{b} - \mathbf{A} \mathbf{H}^{-1} \mathbf{h}. \quad (8)$$

Therefore we can extend the conclusion on the least norm optimization to (7).

As before, we assume  $\mathbf{A}$  and  $\mathbf{b}$  are obtained in two different phases comprising

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

with  $\mathbf{A}_1, \mathbf{A}_2$  of full row rank. Let us write the optimization problem using the measurement from the first phase only as

$$\min (\mathbf{H}\mathbf{x} - \mathbf{h})^T \Sigma^{-1} (\mathbf{H}\mathbf{x} - \mathbf{h}) \quad \text{s.t.} \quad \mathbf{A}_1 \mathbf{x} = \mathbf{b}_1. \quad (9)$$

Formally, we would like to calculate the change of optimal values from the problem (9) to the problem (7).

Let us write (9) in the form of a least norm optimization

$$\min \mathbf{y}^T \mathbf{y} \quad \text{s.t.} \quad \mathbf{A}_1 \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \mathbf{y} = \mathbf{b}_1 - \mathbf{A} \mathbf{H}^{-1} \mathbf{h} \quad (10)$$

and denote its optimal solution as  $\mathbf{y}_1^*$ . The covariance of  $\mathbf{y}_1^*$  is related with that of  $\mathbf{x}_1^* = \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \mathbf{y}_1^* + \mathbf{H}^{-1} \mathbf{h}$  by the linear transformation of Gaussian distribution

$$\mathbf{A}_2 \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \text{Cov}(\mathbf{y}_1^*) \Sigma^{\frac{1}{2}} \mathbf{H}^{-T} \mathbf{A}_2^T = \mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T, \quad (11)$$

and considering  $\mathbf{y}_1^* = \Sigma^{-\frac{1}{2}} (\mathbf{H}\mathbf{x}_1^* - \mathbf{h}_1)$  we have

$$\mathbf{A}_2 \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \mathbf{y}_1^* - (\mathbf{b}_2 - \mathbf{A}_2 \mathbf{H}^{-1} \mathbf{h}_2) = \mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2. \quad (12)$$

Inserting (11) and (12) into the result of minimum norm optimization, we immediately conclude that the change of optimal values writes

$$\Delta f = (\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2)^T [\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T]^{-1} (\mathbf{A}_2 \mathbf{x}_1^* - \mathbf{b}_2). \quad (13)$$

At last, we can write down a formula of  $\text{Cov}(\mathbf{x}_1^*)$  by considering together

$$\begin{aligned} \text{Cov}(\mathbf{y}_1^*) &= \mathbb{I} - \Sigma^{\frac{1}{2}} \mathbf{H}^{-T} \mathbf{A}_1^T (\mathbf{A}_1 \mathbf{H}^{-1} \Sigma \mathbf{H}^{-T} \mathbf{A}_1^T)^{-1} \mathbf{A}_1 \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \\ \text{Cov}(\mathbf{x}_1^*) &= \mathbf{H}^{-1} \Sigma^{\frac{1}{2}} \text{Cov}(\mathbf{y}_1^*) \Sigma^{\frac{1}{2}} \mathbf{H}^{-T} \end{aligned}$$

to reach its explicit form

$$\text{Cov}(\mathbf{x}_1^*) = \mathbf{Q} - \mathbf{Q} \mathbf{A}_1^T (\mathbf{A}_1 \mathbf{Q} \mathbf{A}_1^T)^{-1} \mathbf{A}_1 \mathbf{Q} \quad (14)$$

where  $\mathbf{Q} = \mathbf{H}^{-1} \Sigma \mathbf{H}^{-T}$ .

To conclude, if we have known  $\mathbf{x}_1^*$  and  $\text{Cov}(\mathbf{x}_1^*)$  by solving the problem (9), then we can predict the change of optimal values between (9) and (7) by (13) easily. In the linear case, the result is exact. For nonlinear cases, a linear approximation may apply around its current working point.

#### IV. EXTENSION TO NONLINEAR CASES WITH AN EXAMPLE ON MANIFOLD

A nonlinear least distance optimization with equality constraints can be generalized as

$$\min \mathbb{D}\text{ist}(\mathbf{x}, \tilde{\mathbf{x}}) \quad \text{s.t.} \quad \mathbf{C}(\mathbf{x}) = \mathbf{0}$$

with  $\tilde{\mathbf{x}}$  being the measurement data having the same dimension as  $\mathbf{x}$ , and  $\mathbb{D}\text{ist}(\mathbf{x}, \tilde{\mathbf{x}})$  being a scalar metric.

Usually, there are many ways to define a distance function  $\mathbb{D}\text{ist}(\mathbf{x}, \tilde{\mathbf{x}})$ . Here we specifically focus on the Mahalanobis distance, which extends well to the Frobenius norm and  $\ell_2$

norm based distance. Explicitly, the problem we will discuss takes the form of

$$\min \|\mathbb{D}\text{iff}(\mathbf{x}, \tilde{\mathbf{x}})\|_{\Sigma}^2 \quad \text{s.t.} \quad \mathbf{C}(\mathbf{x}) = \mathbf{0} \quad (15)$$

where  $\mathbb{D}\text{iff}(\mathbf{x}, \tilde{\mathbf{x}})$  is a vectorized difference between  $\mathbf{x}$  and  $\tilde{\mathbf{x}}$ .  $\Sigma$  models our confidence on  $\mathbb{D}\text{iff}(\mathbf{x}, \tilde{\mathbf{x}})$  which is usually interpreted as the covariance of some Gaussian distribution.

A standard extension is to linearize (15) to the form of (7), and then apply the result in linear cases, as the paradigm found in Extended Kalman Filter (EKF) techniques. In essence, the objective function in the linearized problem is an *approximation* to that of its original nonlinear scenario, which means the result is not exact any more. We mention EKF as an example because (13) works incrementally, which shows some similarities to the Kalman Filter (KF). *For an incremental optimization problem, the linearization point will not change dramatically.* Therefore it is reasonable to assume that the linearized Jacobian provides a valid approximation to the original nonlinear problem.

In what follows, we illustrate the idea of using linearization with an example on manifold.

Let  $\mathbf{x}$  and  $\tilde{\mathbf{x}}$  be two elements on the same manifold. We use  $\boxminus$  to describe their difference in the tangent space, i.e.  $\mathbb{D}\text{iff}(\mathbf{x}, \tilde{\mathbf{x}}) = \mathbf{x} \boxminus \tilde{\mathbf{x}}$ . At a certain time point, we have an optimization problem, say **Phase I**:

$$\begin{aligned} \min \|\mathbf{x} \boxminus \tilde{\mathbf{x}}\|_{\Sigma}^2 \\ \text{s.t.} \quad \mathbf{C}_1(\mathbf{x}) = \mathbf{0} \end{aligned} \quad (\text{Phase I})$$

Let  $\mathbf{x}_1^*$  and  $\text{Cov}(\mathbf{x}_1^*)$  be the solution and covariance of the optimization problem in **Phase I**. Now if we obtain another constraint  $\mathbf{C}_2(\mathbf{x}) = \mathbf{0}$ , which constitutes another optimization problem named **Phase II**:

$$\begin{aligned} \min \|\mathbf{x} \boxminus \tilde{\mathbf{x}}\|_{\Sigma}^2 \\ \text{s.t.} \quad \mathbf{C}_1(\mathbf{x}) = \mathbf{0} \\ \mathbf{C}_2(\mathbf{x}) = \mathbf{0} \end{aligned} \quad (\text{Phase II})$$

Then the change of optimal values from **Phase I** to **Phase II** can be approximated by

$$\Delta f = \mathbf{C}_2(\mathbf{x}_1^*)^T [\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T]^{-1} \mathbf{C}_2(\mathbf{x}_1^*) \quad (16)$$

with  $\mathbf{A}_2$  given by

$$\mathbf{A}_2 = - \left. \frac{\partial \mathbf{C}_2(\mathbf{x}_1^* \boxplus \xi)}{\partial \xi} \right|_{\xi=\mathbf{0}}$$

The notation  $\boxplus$  is used to apply a perturbation  $\xi$  in the tangent space of  $\mathbf{x}_1^*$ , and return an element on manifold. The covariance  $\text{Cov}(\mathbf{x}_1^*)$  is given in the tangent space of the solution  $\mathbf{x}_1^*$ . It is calculated by by Eq. (14), by letting

$$\begin{aligned} \mathbf{A}_1 &= - \left. \frac{\partial \mathbf{C}_1(\mathbf{x}_1^* \boxplus \xi)}{\partial \xi} \right|_{\xi=\mathbf{0}} \\ \mathbf{H} &= - \left. \frac{\partial ((\mathbf{x}_1^* \boxplus \xi) \boxminus \tilde{\mathbf{x}})}{\partial \xi} \right|_{\xi=\mathbf{0}} \end{aligned}$$

The derivation of Eq. (16) is quite straightforward via linearization. Let

$$\mathbf{h} = \mathbf{x}_1^* \boxminus \tilde{\mathbf{x}}, \quad \mathbf{b}_1 = \mathbf{C}_1(\mathbf{x}_1^*), \quad \mathbf{b}_2 = \mathbf{C}_2(\mathbf{x}_1^*).$$

At  $\mathbf{x}_1^*$ , let us linearize the problem **Phase I** by a perturbation  $\xi$ . The linearized problem writes

$$\min (\mathbf{H}\xi - \mathbf{h})^T \Sigma^{-1} (\mathbf{H}\xi - \mathbf{h}) \quad \text{s.t.} \quad \mathbf{A}_1 \xi = \mathbf{b}_1. \quad (17)$$

Since  $\mathbf{x}_1^*$  is optimal for **Phase I**, the solution of the above linearized problem is  $\xi_1^* = \mathbf{0}$ . Now if we linearize the constraint  $\mathbf{C}_2(\mathbf{x}) = \mathbf{0}$  to  $\mathbf{A}_2 \xi = \mathbf{b}_2$  at  $\mathbf{x}_1^*$  and add it in (17), then by the result in the linear case, i.e. (13),

$$\begin{aligned} \Delta f &= (\mathbf{A}_2 \xi_1^* - \mathbf{b}_2)^T [\mathbf{A}_2 \text{Cov}(\xi_1^*) \mathbf{A}_2^T]^{-1} (\mathbf{A}_2 \xi_1^* - \mathbf{b}_2) \\ &= \mathbf{b}_2^T [\mathbf{A}_2 \text{Cov}(\xi_1^*) \mathbf{A}_2^T]^{-1} \mathbf{b}_2 \\ &= \mathbf{C}_2(\mathbf{x}_1^*)^T [\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T]^{-1} \mathbf{C}_2(\mathbf{x}_1^*)_2 \end{aligned}$$

which completes the proof.

**Remark 1:** Eq. (16) firstly appeared in the paper [1] (Eq. (15) in that paper). Unfortunately at the time, the mechanism behind the equation was not completely clear. So the analysis here also fills the gap in [1].

## V. APPLICATIONS

### A. Outlier Detection in Constrained Pose Graph Formulation

The constrained pose graph formulation was firstly proposed in [1], while the original idea on the feature based SLAM was published in [22]. In [1], the change of optimal values was used as a metric [1] to detect outliers in an incremental scheme. Although the mechanism was not clear in that paper, the experiments provided in [1] demonstrated the effectiveness of the change of optimal values as a metric to improve the robustness of the framework. As an additional material, for the conventional least squares based pose graph formulation, the outlier detection based on the change of optimal values was published in [20].

### B. Cost of Aligning Two Trajectories

A trajectory is, in its explicit form, a collection of poses. Let  $\mathcal{T}$  be a trajectory, then it can be described by  $\mathcal{T} = \cup \mathbf{T}_i$ ,  $\mathbf{T}_i \in \text{SE}(d)$ ,  $d = 2, 3$ . However, in many applications, if one pose is moved, we want the impact to be passed to other poses as well. Therefore, it is more desirable to exploit an alternative parameterization based on relative transformations (termed relative poses in robotics). Without loss of generality, let us define a trajectory as a head pose, plus a chain of relative poses following the head pose. Let  $\mathcal{T}_A$  and  $\mathcal{T}_B$  be two trajectories given by

$$\begin{aligned} \mathcal{T}_A &\stackrel{\text{def}}{=} \{^A \mathbf{T}_1\} \cup \{^A \mathbf{T}_{i,i+1}\}_{i=1,2,\dots,N_A} \\ \mathcal{T}_B &\stackrel{\text{def}}{=} \{^B \mathbf{T}_1\} \cup \{^B \mathbf{T}_{j,j+1}\}_{j=1,2,\dots,N_B} \end{aligned}$$

where  $^A \mathbf{T}_1$  is the head pose of trajectory  $\mathcal{T}_A$  and  $\{^A \mathbf{T}_{i,i+1}\}_{i=1,2,\dots,N_A}$  is a set of relative poses following the head pose  $^A \mathbf{T}_1$ . The same rule applies to trajectory  $\mathcal{T}_B$ . In this parameterization, we obtain any arbitrary pose by chaining out the relative poses. For example, in the trajectory  $\mathcal{T}_A$ , an arbitrary pose  $^A \mathbf{T}_k$  can be written as

$$^A \mathbf{T}_k = ^A \mathbf{T}_1 ^A \mathbf{T}_{1,2} \cdots ^A \mathbf{T}_{k-1,k}.$$

Now at some point, we know that the  $l$ -th pose  $^A \mathbf{T}_l$  in the trajectory  $\mathcal{T}_A$  and the  $r$ -th pose  $^B \mathbf{T}_r$  in the trajectory  $\mathcal{T}_B$  are the same pose, which results to a hard constraint  $^A \mathbf{T}_l = ^B \mathbf{T}_r$ , and an optimization problem in the form of

$$\begin{aligned} \min & \underbrace{\| ^A \mathbf{T}_1 \boxminus ^A \tilde{\mathbf{T}}_1 \|_{\mathcal{A}\Sigma_1}^2 + \sum_{i=1}^{N_A} \| ^A \mathbf{T}_{i,i+1} \boxminus ^A \tilde{\mathbf{T}}_{i,i+1} \|_{\mathcal{A}\Sigma_{i,i+1}}^2}_{\text{Cost of trajectory } \mathcal{T}_A} \\ & + \underbrace{\| ^B \mathbf{T}_1 \boxminus ^B \tilde{\mathbf{T}}_1 \|_{\mathcal{B}\Sigma_1}^2 + \sum_{j=1}^{N_B} \| ^B \mathbf{T}_{j,j+1} \boxminus ^B \tilde{\mathbf{T}}_{j,j+1} \|_{\mathcal{B}\Sigma_{j,j+1}}^2}_{\text{Cost of trajectory } \mathcal{T}_B} \\ \text{s.t.} & \quad ^A \mathbf{T}_l = ^B \mathbf{T}_r \\ & \quad \text{with } ^A \mathbf{T}_l = ^A \mathbf{T}_1 ^A \mathbf{T}_{1,2} \cdots ^A \mathbf{T}_{l-1,l} \\ & \quad \quad ^B \mathbf{T}_r = ^B \mathbf{T}_1 ^B \mathbf{T}_{1,2} \cdots ^B \mathbf{T}_{r-1,r} \end{aligned} \quad (18)$$

where we use the notations  $^A \tilde{\mathbf{T}}_1$ ,  $^B \tilde{\mathbf{T}}_1$ ,  $^A \tilde{\mathbf{T}}_{i,i+1}$ ,  $^B \tilde{\mathbf{T}}_{i,i+1}$  to describe the initial configuration of  $\mathcal{T}_A$  and  $\mathcal{T}_B$  without the hard constraint  $^A \mathbf{T}_l = ^B \mathbf{T}_r$ , while  $^A \mathbf{T}_1$ ,  $^B \mathbf{T}_1$ ,  $^A \mathbf{T}_{i,i+1}$ ,  $^B \mathbf{T}_{i,i+1}$  are state variables to be estimated by imposing the hard constraint  $^A \mathbf{T}_l = ^B \mathbf{T}_r$ .

**Remark 2:** The problem described in (18) is essentially an extension to the problem ‘‘trajectory bending’’ studied by Dubbelman et al. [23][24]. Please refer to [23][24] for a reference of its application backgrounds. Here in (18), we define the ‘‘bending’’ as a more general constraint by aligning any arbitrary two poses together; while the original paper was to ‘‘bend’’ a pose in a trajectory to a specific given value. However mathematically, I think this difference is marginal and one should extend to another easily. Nevertheless, it is worth noticing how seemingly different problems are essentially a description of kind of the same thing.

**Problem:** Due to process noise, there might exist many potential candidate pairs for  $^A \mathbf{T}_l$  and  $^B \mathbf{T}_r$ . The question is: What is the cost for each specific pair? A brutal force answer is to solve the optimization problem in (18) for each pair. However, alternatively we can use the theory described in this paper to predict the change of optimal values directly.

In (18), let us choose  $\boxminus$  as the logarithm mapping of  $\text{SE}(3)$  in the vector space. To apply the result in Section IV, we further define  $\boxplus$  by the exponential mapping of  $\text{SE}(3)$ . Formally these two binary operators are defined as

$$\boxminus: \quad \mathbf{T}_1 \boxminus \mathbf{T}_2 = \mathbf{Log}(\mathbf{T}_1^{-1} \mathbf{T}_2), \quad \mathbf{T}_1, \mathbf{T}_2 \in \text{SE}(3)$$

$$\boxplus: \quad \mathbf{T} \boxplus \xi = \mathbf{T} \cdot \mathbf{Exp}(\xi), \quad \mathbf{T} \in \text{SE}(3), \xi \in \mathbb{R}^6.$$

The constraint  $^A \mathbf{T}_l = ^B \mathbf{T}_r$  can be written in the standard form

$$^A \mathbf{T}_l = ^B \mathbf{T}_r \iff \mathbf{Log}(^A \mathbf{T}_l ^B \mathbf{T}_r^{-1}) = \mathbf{0}.$$

Let us denote  $\mathbf{C}_2(\mathbf{x}) = \mathbf{Log}(^A \mathbf{T}_l ^B \mathbf{T}_r^{-1})$ . Now we are at the stage to apply the result in Section IV directly.

**Previous solution and its covariance:** Without the constraint  $^A \mathbf{T}_l = ^B \mathbf{T}_r$ , the optimal trajectory for  $\mathcal{T}_A$  and  $\mathcal{T}_B$  are simply their original forms

$$\mathcal{T}_A: \quad ^A \mathbf{T}_1^* = ^A \tilde{\mathbf{T}}_1, \quad ^A \mathbf{T}_{i,i+1}^* = ^A \tilde{\mathbf{T}}_{i,i+1} \quad (i = 1, 2, \dots, N_A)$$

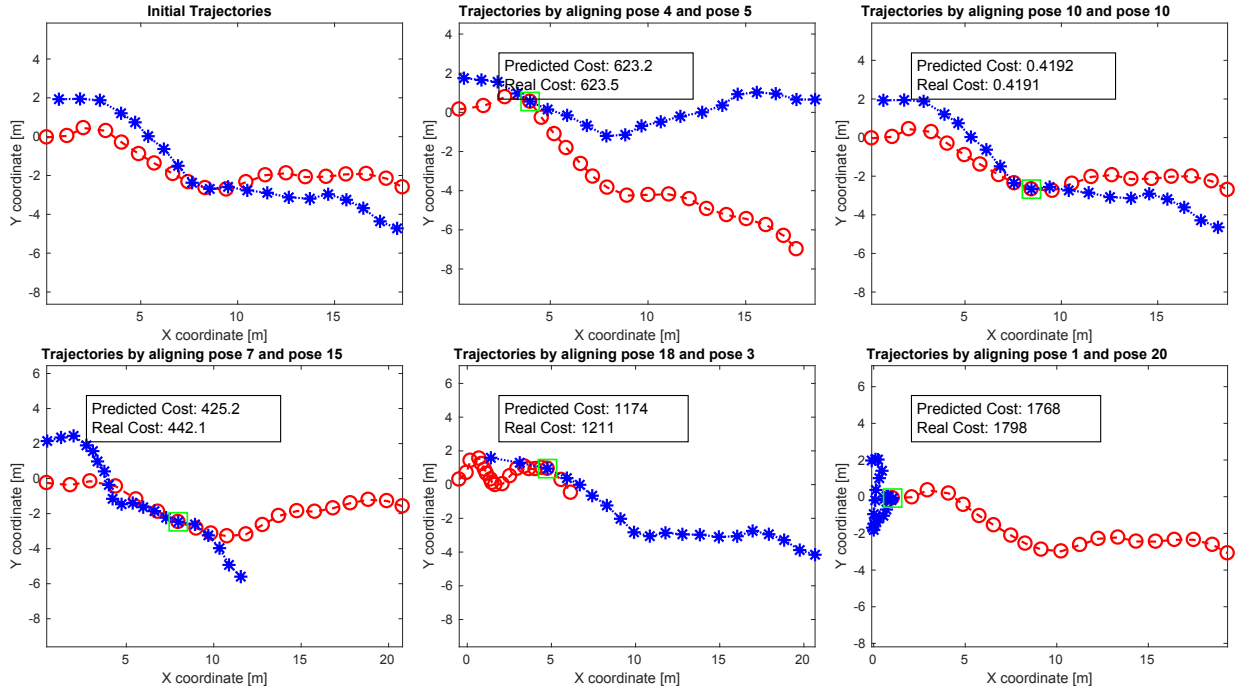


Fig. 1. Examples of aligning two trajectories, plotted in red and blue respectively. For each aligned case, we show the resulting trajectory and the cost it takes. Both the predicted cost (by the OFC metric) and the real cost (by fully optimizing the problem) are reported for comparison.

$$\mathcal{T}_B: {}^B\mathbf{T}_1^* = {}^B\tilde{\mathbf{T}}_1, \quad {}^B\mathbf{T}_{j,j+1}^* = {}^B\tilde{\mathbf{T}}_{j,j+1} \quad (j = 1, 2, \dots, N_B).$$

Let us collect the above result in the variable  $\mathbf{x}_1^*$ . To calculate its covariance  $\text{Cov}(\mathbf{x}_1^*)$ , we linearize the cost function at  $\mathbf{x}_1^*$ . By some trivial linear algebra on Lie group, we get

$$\text{Cov}(\mathbf{x}_1^*) = \mathbf{H}^{-1}\Sigma\mathbf{H}^{-T} \quad (\text{by (14)})$$

with  $\mathbf{H} = \mathbb{I}$ , and  $\Sigma$  constructed diagonally with corresponding  ${}^A\Sigma_i, {}^A\Sigma_{i,i+1}, {}^B\Sigma_i, {}^B\Sigma_{j,j+1}$ . The optimal value  $f^* = 0$ .

**New Constraint:** For the constraint, let us index the  $i$ -th component of a trajectory, for example the trajectory  $\mathcal{T}_A$  as  $\mathcal{T}_A[i]$ , which basically means  $\mathcal{T}_A[1] = {}^A\mathbf{T}_1$ ,  $\mathcal{T}_A[2] = {}^A\mathbf{T}_{1,2}$ ,  $\mathcal{T}_A[3] = {}^A\mathbf{T}_{2,3}$  and so forth. Let us further denote

$$\mathbf{C}_2(\mathbf{x}_1^*) = \text{Log}({}^A\tilde{\mathbf{T}}_l {}^B\tilde{\mathbf{T}}_r^{-1}) = \eta.$$

The linearization of  $\mathbf{C}_2(\mathbf{x}_1)$  at  $\mathbf{x}_1^*$  consists of trivial linear algebras based on the BCH formula and the adjoint operation of SE(3) [25][26]. We provide the analytical Jacobian here and ignore the calculation details for simplicity:

$$\mathbf{A}_2(\mathcal{T}_A[i]) = -\mathbf{J}_l(\eta)^{-1}\text{Ad}({}^A\mathbf{T}_i^*), \quad (i = 1, 2, \dots, l)$$

$$\mathbf{A}_2(\mathcal{T}_B[j]) = \mathbf{J}_r(\eta)^{-1}\text{Ad}({}^A\tilde{\mathbf{T}}_l {}^B\tilde{\mathbf{T}}_r^{-1} {}^B\mathbf{T}_{r-j+1}^*), \quad (j = 1, 2, \dots, r)$$

with  $\mathbf{J}_l(\cdot)$ , and  $\text{Ad}(\cdot)$  being the left-hand Jacobian and the adjoint operation of SE(3) respectively.

**Apply Eq. (16):** Finally, the change of optimal values  $\Delta f$  can be calculated by Eq. (16), and the optimal value of the problem (18) can be approximated by  $f^{**} \approx f^* + \Delta f = \Delta f$ . Note that for this specific example, we do not need to solve any optimization problem. The only requirement is to linearize the constraints, and then apply (16)!

## VI. NUMERICAL EXPERIMENTS

In this section, we provide numerical experiments for the trajectory alignment problem defined in (18). We use Matlab to simulate two random robot trajectories, by firstly rotating the robot by a random angle, and then moving the robot  $1m$  forward. These two trajectories are intersected in the middle in case they are completely apart from one other. Afterwards, we add Gaussian additive noise with standard deviation  $0.1m$  on the translational part, and  $0.01\text{rads}$  on the rotational part. We use an Intel i5-5300U CPU @ 2.30GHz  $\times$  4, with Ubuntu 16.04 LTS distribution to run all the experiments.

### A. An Intuitive Example: Trajectory Change, Predicted Cost and Real Cost

First, let us consider a simple example where each trajectory contains 20 poses. We choose several pose combinations to demonstrate the idea as shown in Fig. 1. We visualize the change of the trajectory with respect to the cost predicted by the metric, as well as the real cost obtained by completely solving the problem (18). From Fig. 1, we would say that the metric is quite robust in general; even for the cases of misalignments, like pose 1 and 20, the resulting trajectories are completely twisted, while the metric is still pretty accurate. Note that the initial noise free trajectories are perfectly aligned at the 10-th poses, where the problem attains the lowest cost (with noise) and the metric is particularly precise.

Then we calculate the cost for all possible pose combinations, which is  $20 \times 20$ , and report the corresponding relative error in Fig 2. It can be envisioned that there exists many unrealistic alignments within these  $20 \times 20$  combinations, however the relative error is essentially very low, with a

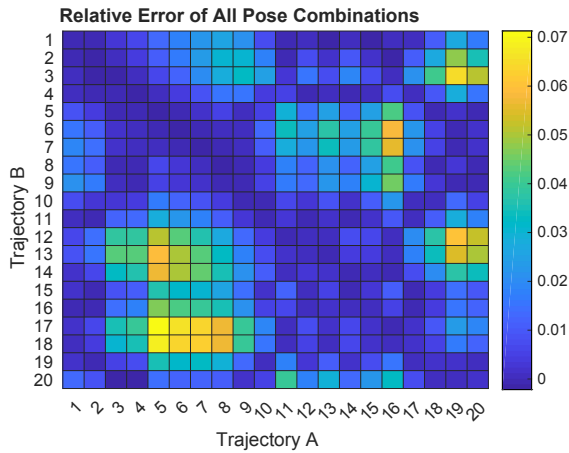


Fig. 2. The relative error of the predicted cost in each pose combination.

TABLE I  
THE COMPUTATIONAL TIME (UNIT: SECONDS)

Num. of Poses	20	50	100	200
By metric overall	2.49116	36.7919	294.033	2457.14
By solving overall	22.5889	460.228	4799.49	42428.2
By metric avg.	0.00623	0.0147	0.0294	0.0614
By solving avg.	0.0565	0.184	0.480	1.06

maximum around 10%, which is sufficient for any further decision makings. At last, with the Matlab code, it takes only 3 seconds to compute the metric for all the  $20 \times 20$  cases, while the corresponding time consumed by solving the problem is 26 seconds.

### B. Computational Time and Relative Error with Different Trajectory Lengths

To quantify the effect of trajectory lengths on the accuracy of the metric, we simulate several trajectories with 20, 50, 100 and 200 poses respectively. As before, we compute the alignment for all possible pose combinations for each case. The relative error between the predicted cost (by the metric) and the real cost (by solving the problem) is reported in Fig. 3. It can be seen from Fig. 3 that for most of the cases, the relative error is still reasonable. However as the length of the trajectory grows, the “unrealistic alignments” (i.e. misalignments) can lead to a substantially amount of outliers in the boxplot. An explanation to this phenomenon is that because the metric is essentially built on the result of linear cases, if the alignment is too far, the linearization point changes dramatically and the linear result is less accurate. However, in general, the predicted cost still provide a reasonable approximation to the real cost.

The computational time is reported in Table I. For each case, we show the overall computational time for all possible pose combinations, as well as the averaged time for obtaining a cost for a single pose combination. Overall, calculating the cost by the metric is at least one magnitude faster than by solving the problem directly. This timing saved here can be extremely beneficially for large problem instances.

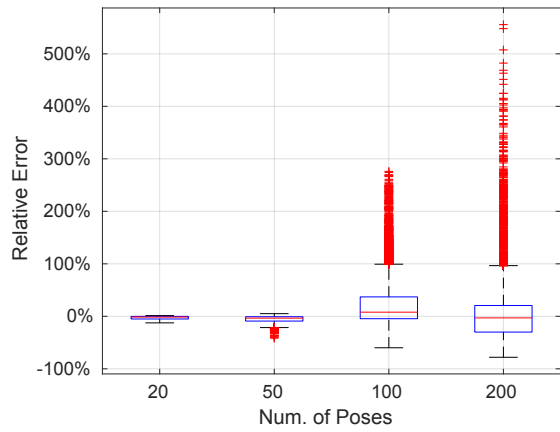


Fig. 3. The relative error of the metric with respect to different lengths of trajectories. Note here we compute all possible pose combinations, for example the number of possible alignments for 200 poses are 40000. This may of course include some ridiculous combinations, which attributes to the outliers (marked in red) in the plot.

## VII. DISCUSSION AND CONCLUSION

In incremental scenarios, it is also possible to compute an approximate solution to the second subproblem (i.e., Eq. (1) (7)), exploiting special structure and sparse linear algebraic techniques [27][28][29][30][31]. While this idea works and is a general practice, the main insight of this paper over previous publications is that: It is actually possible to derive an analytical equation (in closed form) which quantifies the change of optimal values directly. Besides, the proposed method is more advantageous in face of many new measurement candidates, which can share  $\mathbf{x}_1^*$  and  $\text{Cov}(\mathbf{x}_1^*)$ .

In the proposed equations, the computational bottleneck of  $\Delta f$  lies in the calculation of  $\text{Cov}(\mathbf{x}_1^*)$ , which can be mitigated by computing a marginal covariance [32][20], i.e., only computing  $\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T$  explicitly. It is also of great interests to further compare the numerical procedure to calculate an updated solution to the second subproblem, with that to calculate  $\mathbf{A}_2 \text{Cov}(\mathbf{x}_1^*) \mathbf{A}_2^T$ . Further research in this direction will help understanding the connection between updating the solution and updating the optimal value.

In nonlinear cases, the equation is derived by linear approximation. The numerical experiment shows good accuracy in general, but still there are many failure cases as shown in Fig. 3. One explanation is to use the change of linearization points. However, Fig. 1 shows that the approximated  $\Delta f$  is actually quite tolerant to linearization changes (see the twisted trajectories). Therefore it is reasonable to assume that there exists a more sophisticated mechanism that has not been well-understood yet. A theoretical investigation on the accuracy bound of  $\Delta f$  is still required.

To conclude, this paper provides the closed form of the change of optimal values in the incremental minimum norm optimization and least distance optimization in the linear case. The change of optimal values in nonlinear cases are approximated via linearization. These results yield the possibility of using the change of optimal values as a pre-calculated metric, in incremental (or online) applications.

## REFERENCES

- [1] F. Bai, T. Vidal-Calleja, and S. Huang, "Robust incremental SLAM under constrained optimization formulation," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1207–1214, 2018.
- [2] D. J. MacKay, *Information theory, inference and learning algorithms*. Cambridge university press, 2003.
- [3] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active slam," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2080–2087.
- [4] K. Khosoussi, M. Giamou, G. S. Sukhatme, S. Huang, G. Dissanayake, and J. P. How, "Reliable graph topologies for SLAM," *International Journal of Robotics Research*, 2018.
- [5] P. Whaithe and F. P. Ferrie, "Autonomous exploration: Driven by uncertainty," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 193–205, 1997.
- [6] H. J. S. Feder, J. J. Leonard, and C. M. Smith, "Adaptive mobile robot navigation and mapping," *The International Journal of Robotics Research*, vol. 18, no. 7, pp. 650–668, 1999.
- [7] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ international conference on intelligent robots and systems*, vol. 1. IEEE, 2002, pp. 540–545.
- [8] A. J. Davison, "Active search for real-time vision," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1. IEEE, 2005, pp. 66–73.
- [9] T. A. Vidal-Calleja, A. Sanfeliu, and J. Andrade-Cetto, "Action selection for single-camera slam," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 6, pp. 1567–1581, 2010.
- [10] H. Kretschmar and C. Stachniss, "Information-theoretic compression of pose graphs for laser-based slam," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1219–1230, 2012.
- [11] N. Carlevaris-Bianco, M. Kaess, and R. M. Eustice, "Generic node removal for factor-graph slam," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1371–1385, 2014.
- [12] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose slam," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 78–93, 2010.
- [13] H. Johannsson, M. Kaess, M. Fallon, and J. J. Leonard, "Temporally scalable visual slam using a reduced pose graph," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 54–61.
- [14] R. Valencia, M. Morta, J. Andrade-Cetto, and J. M. Porta, "Planning reliable paths with pose slam," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 1050–1059, 2013.
- [15] Y. Wang, R. Xiong, Q. Li, and S. Huang, "Kullback-leibler divergence based graph pruning in robotic feature mapping," in *2013 European Conference on Mobile Robots*. IEEE, 2013, pp. 32–37.
- [16] E. Olson and M. Kaess, "Evaluating the performance of map optimization algorithms," in *RSS Workshop on Good Experimental Methodology in Robotics*, vol. 15, 2009.
- [17] G. Hu, S. Huang, and G. Dissanayake, "Evaluation of pose only slam," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 3732–3737.
- [18] Y. Latif, C. Cadena, and J. Neira, "Robust loop closing over time for pose graph SLAM," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1611–1626, 2013.
- [19] M. C. Graham, J. P. How, and D. E. Gustafson, "Robust incremental slam with consistency-checking," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 117–124.
- [20] F. Bai, T. Vidal-Calleja, S. Huang, and R. Xiong, "Predicting objective function change in pose-graph optimization," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 145–152.
- [21] C. D. Meyer, *Matrix analysis and applied linear algebra*. Siam, 2000, vol. 71.
- [22] F. Bai, S. Huang, T. Vidal-Calleja, and Q. Zhang, "Incremental SQP method for constrained optimization formulation in SLAM," in *Control, Automation, Robotics and Vision (ICARCV), 2016 14th International Conference on*. IEEE, 2016, pp. 1–6.
- [23] G. Dubbelman, I. Esteban, and K. Schutte, "Efficient trajectory bending with applications to loop closure," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 4836–4842.
- [24] G. Dubbelman and B. Browning, "COP-SLAM: closed-form online pose-chain optimization for visual SLAM," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1194–1213, 2015.
- [25] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Springer Science & Business Media, 2011, vol. 2.
- [26] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.
- [27] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, "Methods for modifying matrix factorizations," *Mathematics of computation*, vol. 28, no. 126, pp. 505–535, 1974.
- [28] A. Cassioli, A. Chiavaioli, C. Manes, and M. Sciarone, "An incremental least squares algorithm for large scale linear classification," *European journal of operational research*, vol. 224, no. 3, pp. 560–565, 2013.
- [29] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.
- [30] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
- [31] L. Polok, V. Ila, M. Solony, P. Smrz, and P. Zemcik, "Incremental block cholesky factorization for nonlinear least squares in robotics," in *Robotics: Science and Systems*, 2013, pp. 328–336.
- [32] V. Ila, L. Polok, M. Solony, P. Smrz, and P. Zemcik, "Fast covariance recovery in incremental nonlinear least square solvers," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4636–4643.