A Closed-Form Constraint Networks Solver for Maximum Likelihood Mapping

Dario Lodi Rizzini*
*Dipartimento di Ingegneria dell'Informazione, University of Parma, Parma, Italy

***Abstract*— Maximum likelihood mapping is one of the approaches applied to simultaneous localization and mapping problems. According to such formulation map estimation corresponds to the configuration that maximizes the likelihood associated to a constraint network representing the map. Several efficient iterative fixed-point techniques have been proposed to solve this optimization problem in practice, but with no regard to the structure of the solution.**

In this paper, we present the closed-form solution for a generic constraint network of planar poses. The fundamental assumption concerns the form of error function and the expression of the corresponding gradient. Through algebraic manipulation, the equations relating position variables to angular parameters are derived. Furthermore, the solution is expressed by an orthogonality condition between the vector of orientation parameters and an affine transformation of the same vector. The proposed algorithm has been implemented and applied to solve the position equations of a simple constraint network.

***Index Terms*— Mapping, Maximum Likelihood**

I. INTRODUCTION

Mapping has become an important task for several robotic applications. An autonomous mobile robot often requires a global representation of the environment in order to perform other tasks like navigation. When such representation is not available, the robot has to build the map using its uncertain motion information and sensor observations. Map building and localization are coupled problems and in literature are often referred to as *Simultaneous Localization and Mapping (SLAM)*.

Maximum Likelihood (ML) is one of the approaches developed to address the SLAM problem and mapping in particular. According to ML, SLAM is formulated as a network of constraints. Relations of robot poses and observations are represented by constraints among nodes in a graphical model. A node of the graph may represent both robot poses and map features (feature-based model) or only robot poses (delayed-state model) [6]. The latter formulation is adopted hence after. The solution of the constraint network corresponds to the configuration of the graph that maximizes its likelihood or, equivalently, minimizes the least square error. While the construction of the constraint network is common to all the approaches and strongly depends on the available sensor data and on data association model, the technique used in the solution of the constraint network identifies each specific ML algorithm.

Early ML methods were only suitable for offline processing, since they required both significant computational capabilities and the availability of all data at the beginning of the computation. Thus, the research on such methods focused on the development of efficient numerical solution and on alternative incremental formulation. Less interest has been devoted to the formal analysis of the problem and to the investigation of its properties.

Lu and Milios [15] first proposed a brute force technique to align range scans, once all the scans have been acquired. Gutman and Konolige [10] improved the construction of the network by introducing map patches representing several scans and robot poses and provided an effective loop detection method based on correlation. Several efficient numeric techniques based on relaxation or on gradient descent have been developed in the following years to compute the optimal network configuration. Gauss-Seidel relaxation has been originally proposed by Duckett *et al.* [5]. *Graphical SLAM* [7] and Multi-level relaxation (MLR) [8] were conceived to speed up the relaxation by introducing respectively star nodes and different levels of resolution. The early gradient descent algorithms were based on conjugate gradient [13, 16]. Stochastic gradient descent (SGD) has been proposed [17] to compute the configuration minimizing least-square error by optimizing one constraint per time and by using an incremental pose representation which enables efficient updates. The smoothing and mapping (SAM) algorithm [4] relies on a QR factorization of information matrix that allows an efficient estimation of the poses of the network nodes with an efficient back-substitution.

Recently, several offline algorithms described above have been adapted to the incremental construction of the map. In [18] an incremental version of SGD is described. The incremental tree network optimizer (Toro) [9] combines several improvements such as tree parameterization, adaptive learning rate and selective update rules. The incremental SAM [11] extends the QR factorization algorithm to integrate new observations when available and to reuse previously computed solutions.

The works referred above propose specific numeric techniques to solve the constraint network, which are usually based on fixed point iteration. Although such methods perform efficiently in practice, little attention has been paid to convergence conditions (see e.g. the brief discussion in [8]) and to the analytic discussion of negative loglikelihood function. Even though the performance of constraint extraction from observations strongly depends on empirical factors (environment conditions, specific sensors, assumptions about sensor model and data association algorithms), the network optimization is suitable for formal evaluation once the constraint network is given.

In our previous paper [14], the equations providing the condition for minima of negative loglikelihood function have been derived and a sketch of their solution has been proposed. The only assumption concerns the form of such function. In this paper, we present a closed form solution of constraint network and an implementation of the solution algorithm. The solution is computed by searching the zeros of negative loglikelihood gradient and by deriving a system of nonlinear equations, under the assumption that constraint covariance matrices are identity matrices. Then, the resulting equations are partitioned into position and angular equations. The first ones are linear and relate position displacement terms to the angular parameters. Position equations illustrate the structural dependence between position coordinates and robot orientations and show how the parameters are analytically affected by the closure of a loop. Thus, position displacements can be substituted into the remaining equations. Angular equations are multivariate polynomial equations with respect to the sines and cosines of pose orientations and require further manipulation to achieve the solution of the problem. In particular, the optimal configuration is found when each pair of angular parameters associated to a pose orientation is orthogonal to the corresponding coordinate pair of an affine transformation of the angular parameters vector. The affine transformation depends on the values of the constraints. The proposed solution has been implemented in `Matlab` script language.

The paper is organized as follows. Section II illustrates the general formulation of the maximum likelihood problem and the gradient of the network error function. Section III derives of a simplified set of equations that are more amenable to a solution. Section IV presents the preliminar results obtained on an example of constraint network. Finally, section V gives conclusion remarks.

II. PROBLEM FORMULATION

This section discusses the general formulation of the maximum likelihood (ML) approach and derives the equations that must be satisfied by the solution. The mapping problem is formulated as a graph, whose nodes correspond to the variables of the map and whose edges represent the constraints between pairs of these variables. The network of constraints is extracted from raw sensor data and hence after we assume that it is a given datum of the problem. Furthermore, in this paper the map consists only of robot poses according to the *delayed-state* representation [6]. The delayed-state approach corresponds to a view-based representation of the environment similar to one adopted in [15], but it can be obtained from a *feature based* map by marginalizing the poses [19].

Let $\mathbf{p} = [p_1 \ \cdots \ p_n]^T$ be the vector of robot poses. Each planar pose $p_i = [x_i, y_i, \theta_i]^T$ is usually represented by two position coordinates x_i and y_i and the orientation θ_i with respect to a global reference frame. The origin pose is conventionally noted as $p_0 = [0,0,0]^T$. In this paper, the pose is represented by a 4 dimensional vector $p_i = [x_i, y_i, c_i, s_i]^T$, where $c_i = \cos\theta_i$ and $s_i = \sin\theta_i$ verify equation

$$g(c_i, s_i) = c_i^2 + s_i^2 - 1 = 0 \tag{1}$$

Let $\hat{\delta}_{ij}$ and Ω_{ij} be respectively the mean and the information matrix of an observation of node j seen from node i corresponding to the constraint $\langle i,j\rangle$. Since poses are represented by 4 parameters, $\hat{\delta}_{ij} = [\hat{x}_{ij}, \hat{y}_{ij}, \hat{c}_{ij}, \hat{s}_{ij}]$ and Ω_{ij} is a 4×4 positive definite symmetric matrix. Let $f_{ij}(\mathbf{x})$ be a function that computes a zero noise observation according to the current configuration of the nodes j and i

$$f_{ij}(\mathbf{p}) = \begin{bmatrix} (x_j - x_i)\ c_i + (y_j - y_i)\ s_i \\ -(x_j - x_i)\ s_i + (y_j - y_i)\ c_i \\ c_j\ c_i + s_j\ s_i \\ s_j\ c_i - c_j\ s_i \end{bmatrix} \tag{2}$$

where $\cos(\theta_j - \theta_i) = c_j\ c_i + s_j\ s_i$ and $\sin(\theta_j - \theta_i) = s_j\ c_i - c_j\ s_i$.

The vector of error variables for constraint $\langle i,j\rangle$ is given by

$$e_{ij}(\mathbf{p}) = f_{ij}(\mathbf{p}) - \delta_{ij} \tag{3}$$

The expression of the error is different for the two parameterizations, but the second one that uses *sine* and *cosine* of orientation is safer due to the periodicity of functions. The error on constraint is a scalar obtained by weighting the error of each component of $e_{ij}(p)$

$$\chi^2_{ij}(\mathbf{p}) = e^T_{ij}(\mathbf{p})\ \Omega_{ij}\ e_{ij}(\mathbf{p}) \tag{4}$$

and the error of the whole network is

$$\begin{aligned} \chi^2(\mathbf{p}) &= \sum_{\langle i,j\rangle \in \mathcal{C}} \chi^2_{ij}(\mathbf{p}) & (5) \\ &= \sum_{\langle i,j\rangle \in \mathcal{C}} e^T_{ij}(\mathbf{p})\ \Omega_{ij}\ e_{ij}(\mathbf{p}) & (6) \end{aligned}$$

where $\mathcal{C}$ is the set of constraints. The argument $\mathbf{p}$ will be omitted when the meaning of equations is not ambiguous.

The network is solved by finding the configuration $\mathbf{p}$ that minimizes Eq. (6). Our aim is to investigate the existence of one or more exact minima and to possibly compute their closed-form expression. The first step is the assessment of critical points of Eq. (6) corresponding to the configurations that make gradient of error null. In the next subsection, the general expression of the gradient is derived.

A. Error Gradient Expression

Since the orientation parameters are subjected to Eq. (1), the addressed optimization problem is constrained and the computation of critical points is performed exploiting the method of *Lagrange multipliers*. Thus, the function to be optimized consists of both the error function Eq. (6) and the conditions provided by Eq. (1)

$$E(\mathbf{p}) = \chi^2(\mathbf{p}) + \sum_{i=1}^{n} \lambda_i\ g_i(c_i, s_i). \tag{7}$$

Then, the target function $E(\mathbf{p})$ is a sum of terms and each term depends only on the parameters of few poses. The expression of a row block of gradient $\nabla_i E = \partial E(\mathbf{p})/\partial p_i$ is a function of the variables of the pose p_i and of the poses p_j connected to node i by a constraint. Let $\mathcal{I}_i$ and $\mathcal{O}_i$ be respectively the sets of the indices connected to pose p_i by an incoming or an outgoing constraint, or formally

$$\mathcal{I}_i = \{j | \langle j,i\rangle \in \mathcal{C}\} \quad (8)$$

$$\mathcal{O}_i = \{j | \langle i,j\rangle \in \mathcal{C}\} \quad (9)$$

Therefore, the gradient row block $\nabla_i\chi$ has the following expression

$$\begin{aligned}\nabla_i E &= \sum_{j\in\mathcal{I}_i} \nabla_i\chi^2_{ji} + \sum_{j\in\mathcal{O}_i} \nabla_i\chi^2_{ij} + \lambda_i\ \nabla_i g_i(c_i,s_i) \quad (10)\\ &= \sum_{j\in\mathcal{I}_i} 2\frac{\partial e^T_{ji}}{\partial p_i}\ \Omega_{ji}\ e_{ji} + \sum_{j\in\mathcal{O}_i} 2\frac{\partial e^T_{ij}}{\partial p_i}\ \Omega_{ij}\ e_{ij}\\ &\quad +2\lambda_i\ [0,0,c_i,s_i]^T \quad (11)\end{aligned}$$

Hence after, the constant 2 is omitted since it does not affect the computation of critical points. The equation Eq. (11) contains the Jacobians of constraint errors e_{ij} and e_{ji} with respect to p_i. The value of Jacobian is different when the constraint is incoming or outgoing w.r.t. pose p_i. Therefore, the expressions of incoming and outgoing Jacobians are given by

$$\frac{\partial e_{ij}}{\partial p_j} = \begin{bmatrix} c_i & s_i & 0 & 0\\ -s_i & c_i & 0 & 0\\ 0 & 0 & c_i & s_i\\ 0 & 0 & -s_i & c_i \end{bmatrix} \quad (12)$$

$$\frac{\partial e_{ij}}{\partial p_i} = \begin{bmatrix} -c_i & -s_i & (x_j-x_i) & (y_j-y_i)\\ s_i & -c_i & (y_j-y_i) & -(x_j-x_i)\\ 0 & 0 & c_j & s_j\\ 0 & 0 & s_j & -c_j \end{bmatrix} \quad (13)$$

The expression of gradient row block is obtained by substituting the expressions of Jacobians and of the constraint error vectors e_{ij}.

B. Gradient of Simplified Error Function

In the general case, the expression of gradient terms $\nabla_j\chi^2_{ij}$ and $\nabla_i\chi^2_{ij}$ are complex and difficult to manipulate. The constraint error functions χ^2_{ij} represent the Mahalanobis distance between the value of constraint with current configuration $\mathbf{p}$ and the required value. When $\Omega_{ij} = I_4$ for each constraint $\langle i,j\rangle$, this distance is equal to the Euclidean one. Such distance mixes metric and angular terms which are not homogeneous, but its expression can be computed and manipulated to achieve some results. Even though such an approach may be questionable, it is worthwhile investigating it in order to gain further insight into the structure of the solution. Gradient terms become:

$$\nabla_j\chi^2_{ij} = \begin{bmatrix} x_{ij} - \hat{x}_{ij}c_i + \hat{y}_{ij}s_i\\ y_{ij} - \hat{x}_{ij}s_i - \hat{y}_{ij}c_i\\ c_j - \hat{c}_{ij}c_i + \hat{s}_{ij}s_i\\ s_j - \hat{c}_{ij}s_i - \hat{s}_{ij}c_i \end{bmatrix} \quad (14)$$

$$\nabla_i\chi^2_{ij} = \begin{bmatrix} -x_{ij} + \hat{x}_{ij}c_i - \hat{y}_{ij}s_i\\ -y_{ij} + \hat{x}_{ij}s_i + \hat{y}_{ij}c_i\\ x^2_{ij}c_i + y^2_{ij}c_i - \hat{x}_{ij}x_{ij} - \hat{y}_{ij}y_{ij}\\ \quad + c_i - \hat{c}_{ij}c_j - \hat{s}_{ij}s_j\\ x^2_{ij}s_i + y^2_{ij}s_i + \hat{y}_{ij}x_{ij} - \hat{x}_{ij}y_{ij}\\ \quad + s_i + \hat{s}_{ij}c_j - \hat{c}_{ij}s_j \end{bmatrix} \quad (15)$$

In particular, Eq. (14) and Eq. (15) represent the gradient of error on the constraint $\langle i,j\rangle$. The two equations depend on the relative position displacements $x_{ij} = x_j - x_i$ and $y_{ij} = y_j - y_i$ and on the angular parameters. Observe that the gradient terms achieved by differentiating with respect to position parameters x_i, y_i, x_j and y_j are linear and equal except for the sign.

III. SOLUTION OF THE CONSTRAINT NETWORK

In the previous section, the expression of gradient terms has been computed under a specific hypothesis on the information matrices of the constraints. Then, the aim is the computation of the solutions, which in this context correspond to the values of the pose vector $\mathbf{p}$ that make the error gradient equal to zero. A solution is not necessarily a minimum of error function Eq. (6). Furthermore, the existence of one or more solutions has not been fully investigated. The system of equations will be manipulated in the following in order to give a constructive answer to these questions.

Let $(\mathcal{N},\mathcal{C})$ be the connected directed graph representing the map, $n = |\mathcal{N}| - 1$ the number of nodes except for origin and $e = |\mathcal{C}|$ the number of constraints or edges. Given a total order over the edges $(\mathcal{C},<)$, we define the $(n+1)\times e$ *output and input incidence matrices*, N^{out} and N^{in} as follows. The element n^{out}_{gh} of N^{out} is equal to 1, when the $h-th$ edge $\langle i_h, j_h\rangle$ comes out of the $g-th$ node (i.e. $i_h = g$), and otherwise it is equal to 0. Similarly, element $n^{\text{in}}_{gh} = 1$, when $j_h = g$, and otherwise it is equal to 0. Furthermore, let the *incidence matrix* N be the difference of the two matrices, $N = N^{\text{in}} - N^{\text{out}}$.

The constraint networks that are considered in this paper must satisfy few assumptions mainly depending on the nature of the mapping problem.

- The constraints $\langle i-1,i\rangle$, with $i = 1,\ldots,n$, belong to the network. This property holds because the graph has been built by a robot exploring the environment and a proper node numbering that respects the sequential growth of the network is chosen like in [17]. The direction of the constraint from the pose with a smaller index to the pose with a larger index is the result of next rule.
- Each constraint is expressed from the point of view of the pose with smaller index. Thus, constraint $\langle i,j\rangle$ contains the parameters of the frame change from i to j with $i < j$. In general, the direction of a constraint $\langle i,j\rangle$ does not affect the result, since information matrix Ω_{ij} can be adjusted. However, the hypotheses previously made on Ω_{ij} do not allow the inversion of a constraint without changing the error function. Even so, the proposed rule on direction is basically consistent with the usual structure of the network.

In order to achieve a solution of $\nabla E(\mathbf{p}) = 0$, the components of each gradient row block Eq. (11) are divided in two parts. The first two equations of each row block are linear with respect to both position and orientation parameters. The last two equations are obtained from the differentiation of error function with respect to the angular parameters and depend also on the Lagrange multipliers. Hence after, the first ones are called *position equations* and the latter *angular equations*. The solution algorithm initially operates on position equations in order to substitute the position terms in angular equations. Then, the angular parameters are solved.

A. Position Equations

The position equations are obtained by combining the terms in the first two rows of Eq. (14) and Eq. (15) according to the graph topology. Since the same operations are performed on the equations for coordinate x and for coordinate y, a vectorial notation is adopted

$$\sum_{j\in\mathcal{I}_i}(-d_{ji}+l_{ji})+\sum_{j\in\mathcal{O}_i}(d_{ij}-l_{ij})=0 \tag{16}$$

where d_{ij} and l_{ij} are defined as

$$d_{ij} = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} = \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} \tag{17}$$

$$l_{ij} = \begin{bmatrix} c_i & -s_i \\ s_i & c_i \end{bmatrix}\begin{bmatrix} \hat{x}_{ij} \\ \hat{y}_{ij} \end{bmatrix} \tag{18}$$

The system of position equations is given by the equations Eq. (16), one for each pose p_i. Our aim is to compute the value of position terms d_{ij} with respect to angular terms l_{ij}, which are considered known terms of the linear system.

An interesting result is achieved by splitting the terms d_{ij} into the position coordinate vectors $d_i = [x_i, y_i]^T$ and $d_j = [x_j, y_j]^T$. The matrix of the new linear system is the *adjacency matrix* of the constraint network [1], after removing node p_0. Thus, the solution of the system can be computed by multiplying the inverse of adjacency matrix and the fixed terms. The values of position displacements x_{ij} and y_{ij} are then obtained by the difference of the positions of the two poses.

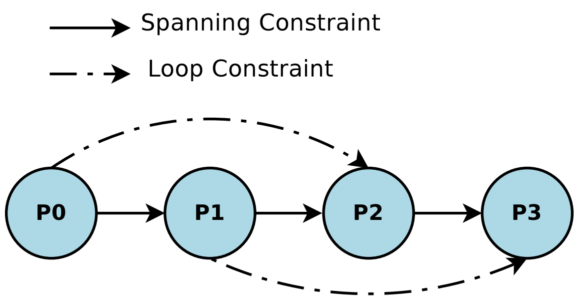

Fig. 1. An example of network with spanning constraints and loop constraints.

However, it would be better to estimate directly the x_{ij} and y_{ij} in order to point out the dependence on angular terms l_{ij}. Constraints are ordered starting from the constraints in the form $\langle i, i+1\rangle$ (called *spanning constraints* hence after) and then adding the other constraints $\langle i, j\rangle$ with $j > i+1$ in lexicographical order (called *loop constraints*). Figure 1 illustrates the difference between the two types of constraints with an example. The *incidence matrix* of the network is the matrix of the linear system of equations. Since the rank of incidence matrix of a connected graph is $n = |\mathcal{N}| - 1$ [1], the missing $n_L = e - n$ equations are provided by loop equations:

$$d_{ij} = \sum_{k=i+1}^{j} d_{k-1,k} \tag{19}$$

Equations (16) and (19) can be manipulated in order to write a matricial expression of position parameters d_{ij} with respect to angular parameters l_{ij}. In particular, let the constraints be ordered according to the sorting criterion

$$\langle i,j\rangle \;<\; \langle g,h\rangle \Leftrightarrow i+(n+1)(j-i) < g+(n+1)(h-g) \tag{20}$$

According to such criterion spanning constraints come before loop constraints.

Let $\mathbf{d}_S = [d_{01}\dots d_{n-1,n}]^T$ be the vector of position displacements of spanning edges, $\mathbf{l}_S$ be the corresponding spanning angular terms, $\mathbf{d}_L = [d_{g_1,h_1}\dots d_{g_{n_L},h_{n_L}}]^T$ be the vector of position displacements of loop edges, $\mathbf{d} = [\mathbf{d}_S\mathbf{d}_L]^T$ and $\mathbf{l} = [\mathbf{l}_S\mathbf{l}_L]^T$ be the complete vectors. Let us define the $n_L \times n$ matrix U whose elements are

$$u_{ij} = \begin{cases} 1 & \text{if } g_i < j \le h_i \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

The above matrix relates each loop edge $\langle g_i, h_i\rangle$ with the spanning edges that belong to the same loop. Matrix U cannot be used directly, since terms d_{ij} and l_{ij} are vector terms. Thus, we introduce $\bar{U} = U \otimes I_2$, where $\otimes$ is the *Kronecker product*. The Kronecker product of an $r_A \times c_A$ matrix A and a $r_B \times c_B$ matrix B is an $r_A r_B \times c_A c_B$ matrix $C = A \otimes B$, whose $r_B \times c_B$ block elements $c_{ij} = a_{ij}B$. Thus, the resulting system of equations is given by

$$\begin{bmatrix} I_{2n} & \bar{U}^T \\ 0 & I_{2n_L}+\bar{U}\bar{U}^T \end{bmatrix}\begin{bmatrix} \mathbf{d}_S \\ \mathbf{d}_L \end{bmatrix} = \begin{bmatrix} I_{2n} & \bar{U}^T \\ \bar{U} & \bar{U}\bar{U}^T \end{bmatrix}\begin{bmatrix} \mathbf{l}_S \\ \mathbf{l}_L \end{bmatrix} \tag{22}$$

The solution of the linear system is given by

$$\begin{aligned}
\begin{bmatrix} \mathbf{d}_S \\ \mathbf{d}_L \end{bmatrix} &= \begin{bmatrix} P_{SS} & P_{SL} \\ P_{LS} & P_{LL} \end{bmatrix}\begin{bmatrix} \mathbf{l}_S \\ \mathbf{l}_L \end{bmatrix} && (23)\\
G &= (I_{2n_L}+\bar{U}\bar{U}^T)^{-1} && (24)\\
P_{SS} &= I_{2n} - \bar{U}^T G \bar{U} && (25)\\
P_{SL} &= \bar{U}^T - \bar{U}^T G \bar{U}\bar{U}^T && (26)\\
P_{LS} &= G\bar{U} && (27)\\
P_{LL} &= G\bar{U}\bar{U}^T && (28)
\end{aligned}$$

Vector $\mathbf{l}$ depends on the parameters c_i and s_i of the poses and it is convenient to express such dependency in the solution. Such dependency can be expressed explicitly with respect to the vector of angular parameters $\mathbf{t} = [c_1\ s_1 \dots c_n\ s_n]^T$ by using a $2e \times 2(n+1)$ matrix Q defined as

$$Q = \left((N^{\text{out}})^T \otimes \text{ones}(2,2)\right) .\!* \left(\text{ones}(1,n+1)\otimes D\right) \tag{29}$$

$$D = \begin{bmatrix} \hat{x}_{i_1 j_1} & \hat{y}_{i_1 j_1} & \dots & \hat{x}_{i_e j_e} & \hat{y}_{i_e j_e} \\ -\hat{y}_{i_1 j_1} & \hat{x}_{i_1 j_1} & \dots & -\hat{y}_{i_e j_e} & \hat{x}_{i_e j_e} \end{bmatrix}^T \tag{30}$$

where .* is the product element-by-element and $ones()$ generates a matrix consisting of 1s with the given size in a `Matlab`-like notation. Thus, we can finally write $\mathbf{d} = P\,Q\,[1\ 0\ \mathbf{t}^T]^T$.

B. Angular Equations

For each pose p_i, a pair of angular equations is defined. These equations combine the terms of incoming and outgoing constraint gradient (respectively Eq. (14) and Eq. (15)) and the terms related to Lagrange multipliers:

$$\begin{aligned}&\sum_{j\in\mathcal{O}_i}(x_{ij}^2 c_i + y_{ij}^2 c_i - \hat{x}_{ij}x_{ij} - \hat{y}_{ij}y_{ij} + c_i - \hat{c}_{ij}c_j - \hat{s}_{ij}s_j)\\ &\qquad + \sum_{j\in\mathcal{I}_i}(c_i - \hat{c}_{ji}c_j + \hat{s}_{ji}s_j) - \lambda_i c_i = 0\end{aligned} \tag{31}$$

$$\sum_{j\in\mathcal{O}_i}(x_{ij}^2 s_i + y_{ij}^2 s_i + \hat{y}_{ij}x_{ij} - \hat{x}_{ij}y_{ij} + s_i + \hat{s}_{ij}c_j - \hat{c}_{ij}s_j) + \sum_{j\in\mathcal{I}_i}(s_i - \hat{s}_{ji}c_j - \hat{c}_{ji}s_j) - \lambda_i s_i = 0 \tag{32}$$

The above equations are difficult to manipulate due to the dependence on Lagrange multipliers and on the square of position displacements x_{ij}^2 and y_{ij}^2. Equations Eq. (31) and Eq. (32) can be combined together in order to achieve a simplified relation. In general, given the polynomials $f_1, f_2, g_1, g_2 \in \mathbb{R}[\mathbf{p}, \lambda]$, if $[\mathbf{p}^*, \lambda^*]^T$ is a zero of both f_1 and f_2, then $[\mathbf{p}^*, \lambda^*]^T$ is also a zero of $g_1 f_1 + g_2 f_2$. Thus, equation $g_1 f_1 + g_2 f_2 = 0$ can be solved, but the achieved solutions must be verified in the original equations.

A new equation is obtained by adding side by side Eq. (31) multiplied by $-s_i$ and Eq. (32) multiplied by c_i.

$$s_i\ \alpha_i + c_i\beta_i = 0 \tag{33}$$

where

$$\begin{aligned}
\alpha_i &= \sum_{j\in\mathcal{I}_i}(\hat{c}_{ji}c_j - \hat{s}_{ji}s_j) + \sum_{j\in\mathcal{O}_i}(\hat{c}_{ij}c_j + \hat{s}_{ij}s_j) + \\
&\quad + \sum_{j\in\mathcal{O}_i}(\hat{x}_{ij}x_{ij} + \hat{y}_{ij}y_{ij}) && (34)\\
\beta_i &= -\sum_{j\in\mathcal{I}_i}(\hat{s}_{ji}c_j + \hat{c}_{ji}s_j) + \sum_{j\in\mathcal{O}_i}(\hat{s}_{ij}c_j - \hat{c}_{ij}s_j) + \\
&\quad + \sum_{j\in\mathcal{O}_i}(\hat{y}_{ij}x_{ij} - \hat{x}_{ij}y_{ij}) && (35)
\end{aligned}$$

Values of $\mathbf{v} = [\alpha_1\ \beta_1 \dots \alpha_n\ \beta_n]^T$ can be written in a more convenient matricial form. The substitution of terms with x_{ij} and y_{ij} is expressed as

$$\mathbf{v} = (L + M\ P\ Q)\begin{bmatrix} 1 \\ 0 \\ \mathbf{t} \end{bmatrix} \tag{36}$$

where

$$\begin{aligned}
M &= \left(N_0^{\text{out}} \otimes \text{ones}(2,2)\right) .* \left(\text{ones}(n,1) \otimes D^{\perp}\right) && (37)\\
D^{\perp} &= \begin{bmatrix} \hat{x}_{i_1 j_1} & \hat{y}_{i_1 j_1} & \dots & \hat{x}_{i_e j_e} & \hat{y}_{i_e j_e} \\ \hat{y}_{i_1 j_1} & -\hat{x}_{i_1 j_1} & \dots & \hat{y}_{i_e j_e} & -\hat{x}_{i_e j_e} \end{bmatrix} && (38)\\
L &= L^{\text{in}}\left((N^{\text{out}})^T \otimes I_2\right) + L^{\text{out}}\left((N^{\text{in}})^T \otimes I_2\right) && (39)\\
L^{\text{in}} &= \left(N_0^{\text{in}} \otimes \text{ones}(2,2)\right) .* \left(\text{ones}(n,1) \otimes C^{\text{in}}\right) && (40)\\
L^{\text{out}} &= \left(N_0^{\text{out}} \otimes \text{ones}(2,2)\right) .* \left(\text{ones}(n,1) \otimes C^{\text{out}}\right) && (41)\\
C^{\text{in}} &= \begin{bmatrix} \hat{c}_{i_1 j_1} & -\hat{s}_{i_1 j_1} & \dots & \hat{c}_{i_e j_e} & -\hat{s}_{i_e j_e} \\ -\hat{s}_{i_1 j_1} & -\hat{c}_{i_1 j_1} & \dots & -\hat{s}_{i_e j_e} & -\hat{c}_{i_e j_e} \end{bmatrix} && (42)\\
C^{\text{out}} &= \begin{bmatrix} \hat{c}_{i_1, j_1} & \hat{s}_{i_1, j_1} & \dots & \hat{c}_{i_e, j_e} & \hat{s}_{i_e, j_e} \\ \hat{s}_{i_1, j_1} & -\hat{c}_{i_1, j_1} & \dots & \hat{s}_{i_e, j_e} & -\hat{c}_{i_e, j_e} \end{bmatrix} && (43)
\end{aligned}$$

N_0^{out} and N_0^{in} are respectively the $n \times e$ submatrix of input and output incidence matrices obtained removing the first row corresponding to origin node p_0. The matrix in Eq. (36) can then be split in a column vector b of known terms (first column vector of $L + MPQ$) and a matrix A (the columns from 3 to $2n$).

After the substitution of x_{ij} and y_{ij} with position equations, there are n quadratic equations in $\mathbf{t}$ and n conditions $c_i^2 + s_i^2 = 1$ $(i = 1, \dots, n)$. Several approaches have been proposed to solve a system of multivariate quadratic polynomial equations, e.g. *Gröbner basis* method [3], resultant techniques like *Dixon resultant* [12], and *Extended Linearization* (XL) [2]. In all these methods, the solution is achieved through complex symbolic computation of a high-degree univariate polynomial. However, the specific form of Eq. (33) suggests that the solutions of the problem correspond to conditions $\alpha_i = \rho_i c_i$ and $\beta_i = -\rho_i s_i$. Thus, the problem is expressed as an affine-parametric system of linear equations

$$\begin{aligned}
R\,\mathbf{t} &= A\,\mathbf{t} - b && (44)\\
R &= \text{diag}([\rho_1\ \ -\rho_1 \dots \rho_n\ \ -\rho_n]) && (45)
\end{aligned}$$

where parameters ρ_i must be chosen to satisfy $c_i^2 + s_i^2 = 1$. In this paper, no formal study of the rank of A and of $A - R$ is presented. The numerical evidence of examples and the considerations on the rank of the matrices used to derive A (in particular, the incidence matrix has rank n) suggest that the rank of the matrix is $2n$, at least for non-degenerate constraints. Currently, this step is performed numerically, but we are actively pursuing an exact condition.

IV. Results

In this section, we present a simple network to illustrate the solution derived in the previous section and a numerical example that partially support the correctness of the approach.

A. Analytic Example

The smallest significant constraint network consists of $n + 1 = 3$ poses and $e = 3$ constraints and contains a single loop. The solution of such network can be manually computed. The gradient equations for p_1 components are the following

$$\begin{aligned}
(-x_{01} + \hat{x}_{01}) + (x_{12} - \hat{x}_{12}\ c_1 + \hat{y}_{12}\ s_1) &= 0\\
(-y_{01} + \hat{y}_{01}) + (y_{12} - \hat{x}_{12}\ s_1 - \hat{y}_{12}\ c_1) &= 0\\
(x_{12}^2\ c_1 + y_{12}^2\ c_1 - \hat{x}_{12}\ x_{12} - \hat{y}_{12}\ y_{12} + &\\
+c_1 - \hat{c}_{12}c_2 - \hat{s}_{12}s_2) + (c_1 - \hat{c}_{01}) &= \lambda_1 c_1\\
(x_{12}^2\ s_1 + y_{12}^2\ s_1 + \hat{y}_{12}\ x_{12} - \hat{x}_{12}\ y_{12} + &\\
+s_1 + \hat{s}_{12}\ c_2 - \hat{c}_{12}\ s_2) + (s_1 - \hat{s}_{01}) &= \lambda_1 s_1
\end{aligned}$$

and those for p_2 components

$$\begin{aligned}
(-x_{02} + \hat{x}_{02}) + (-x_{12} + \hat{x}_{12}\ c_1 - \hat{y}_{12}\ s_1) &= 0\\
(-y_{02} + \hat{y}_{02}) + (-y_{12} + \hat{x}_{12}\ s_1 + \hat{y}_{12}\ c_1) &= 0\\
(c_2 - \hat{c}_{02}) + (c_2 - \hat{c}_{12}\ c_1 + \hat{s}_{12}\ s_1) &= \lambda_2 c_2\\
(s_2 - \hat{s}_{02}) + (s_2 - \hat{s}_{12}\ c_1 - \hat{c}_{12}\ s_1) &= \lambda_2 s_2
\end{aligned}$$

The solution illustrated in previous section has been derived from gradient equations and expressed in the matricial form. For the 3 poses network the specific terms are the following

$$\begin{aligned}
P &= \frac{1}{3}\begin{bmatrix} 2 & -1 & 1 \\ -1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} && (46)\\
Q &= \begin{bmatrix} \hat{x}_{01} & \hat{y}_{01} & 0 & 0 & 0 & 0 \\ -\hat{y}_{01} & \hat{x}_{01} & 0 & 0 & 0 & 0 \\ 0 & 0 & \hat{x}_{12} & \hat{y}_{12} & 0 & 0 \\ 0 & 0 & -\hat{y}_{12} & \hat{x}_{12} & 0 & 0 \\ \hat{x}_{02} & \hat{y}_{02} & 0 & 0 & 0 & 0 \\ -\hat{y}_{02} & \hat{x}_{02} & 0 & 0 & 0 & 0 \end{bmatrix} && (47)\\
M &= \begin{bmatrix} 0 & 0 & \hat{x}_{12} & -\hat{y}_{12} & 0 & 0 \\ 0 & 0 & \hat{y}_{12} & \hat{x}_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} && (48)\\
L &= \begin{bmatrix} \hat{c}_{01} & -\hat{s}_{01} & 0 & 0 & \hat{c}_{12} & \hat{s}_{12} \\ -\hat{s}_{01} & -\hat{c}_{01} & 0 & 0 & \hat{s}_{12} & -\hat{c}_{12} \\ \hat{c}_{02} & -\hat{s}_{02} & \hat{c}_{12} & -\hat{s}_{12} & 0 & 0 \\ -\hat{s}_{02} & -\hat{c}_{02} & -\hat{s}_{12} & -\hat{c}_{12} & 0 & 0 \end{bmatrix} && (49)
\end{aligned}$$

This result is confirmed by direct manipulation of the gradient equations.

B. Testing Position Equations

The proposed algorithm has been implemented as a `Matlab` script to test the equations on a numerical example. A network of 100 poses corresponding to a robot moving on a grid has been randomly generated. A gaussian noise has been added to the constraints to simulate the uncertainty of motion and sensor data. Figure 2 compares the result obtained using

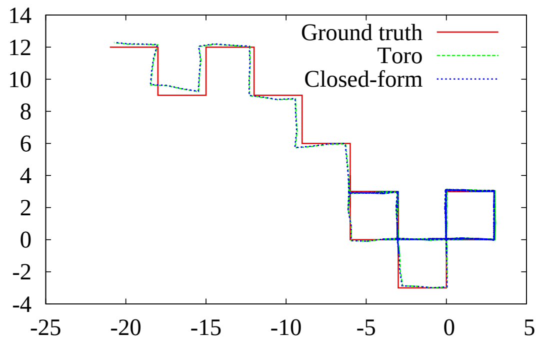

Fig. 2. The solution of a 100 poses constraint network computed by *Toro* and according the position equations with the orientation angles of Toro.

the network solver *Toro* [9] and the position computed using the position equations with the orientation values computed by Toro. Even though the numerical optimizer performs the optimization on the general negative loglikelihood, the two solved networks almost overlap. Therefore, this example suggests that the final result is weakly affected by the choice of a simplified error function and that position equations are correct.

V. CONCLUSION

In this paper, we have derived the closed form solution for a constraint network of planar poses when the error function is given in a particular form. The gradient of error function is computed and the resulting system of nonlinear equations is manipulated in order to achieve a simplified set of relations. The first result is the position components of network configuration with respect to angular parameters. The position terms have been substituted into the remaining equations, which only depends on angular terms after the substitution. Therefore, an orthogonality condition between the vector of orientation parameters and an affine transformation of the same vector has been derived to compute the optimum configuration of the constraint network. Moreover, the proposed algorithm has been implemented and partially tested on an example.

Future works will investigate three open issues. First, a formal evaluation of the existence and the number of solutions of the constraint network is required, possibly formulating constructive conditions related to the proposed solution. Second, we will better investigate the orthogonality condition in order to pursue an exact equation. Finally, we will perform further experiments to compare results with larger constrain networks in order to validate the proposed approach for practical application.

VI. ACKNOWLEDGMENTS

The author gratefully thanks Stefano Caselli and Luca Consolini for the suggestions and the discussion of the results presented in the paper. This reasearch is partially supported by laboratory AER-TECH of Regione Emilia-Romagna, Italy.

REFERENCES

[1] B. Bollobas. *Modern Graph Theory*. Springer Verlag, 1998.
[2] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Proc. of EUROCRYPT 2000*, volume 1807, pages 392–407. Springer Verlag, 2000.
[3] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra 3rd ed.* Springer, 1996.
[4] F. Dellaert and M. Kaess. Square root sam: Simultaneous location and mapping via square root information smoothing. *Int. Journal of Robotics Research*, 25(12):1181–1204, 2006.
[5] T. Duckett, S. Marsland, and J. Shapiro. Fast, on-line learning of globally consistent maps. *Journal of Autonomous Robots*, 12(3):287 – 300, 2002.
[6] R. Eustice, H. Singh, and J.J. Leonard. Exactly sparse delayed-state filters. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2428–2435, 2005.
[7] J. Folkesson and H.I. Christensen. Graphical SLAM-a self-correcting map. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 419–427, 2004.
[8] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):1–12, 2005.
[9] G. Grisetti, D. Lodi Rizzini, C. Stachniss, E. Olson, and W. Burgard. Online constraint network optimization for efficient maximum likelihood map learning. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1880–1885, 2008.
[10] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE Int. Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 318–325, 1999.
[11] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(2):1365–1378, 2008.
[12] D. Kapur, T. Saxena, and L. Yang. Algebraic and geometric reasoning using dixon resultants. In *In ACM ISSAC*, pages 99–107, 1994.
[13] K. Konolige. Large-scale map-making. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, pages 457–463, 2004.
[14] D. Lodi Rizzini. Towards a closed-form solution of constraint networks for maximum likelihood mapping. In *Proc. of the Int. Conf. on Advanced Robotics (ICAR)*, 2009.
[15] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Journal of Autonomous Robots*, 4:333–349, 1997.
[16] M. Montemerlo and S. Thrun. Large-scale robotic 3-D mapping of urban structures. In *Proc. of Int. Sym. on Experimental Robotics (ISER)*, 2004.
[17] E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2262–2269, 2006.
[18] E. Olson, J. Leonard, and S. Teller. Spatially-adaptive learning rates for online incremental SLAM. In *Proc. of Robotics: Science and Systems (RSS)*, Atlanta, GA, USA, 2007.
[19] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005.