Robotics and Autonomous Systems 87 (2017) 120–132

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Incremental sparse GP regression for continuous-time trajectory estimation and mapping

Xinyan Yan [a,*], Vadim Indelman [b], Byron Boots [a]

[a] Institute for Robotics and Intelligent Machines, College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA
[b] Faculty of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel

HIGHLIGHTS

- An incremental sparse GP regression algorithm for STEAM problems is proposed.
- The benefits of GP-based approaches and incremental smoothing are combined.
- The approach elegantly handles asynchronous and sparse measurements.
- Results indicate significant speed-up in performance with little loss in accuracy.

ARTICLE INFO

Article history:
Received 26 January 2016
Accepted 4 October 2016
Available online 14 October 2016

Keywords:
State estimation
Localization
Continuous time
SLAM
Gaussian process regression

ABSTRACT

Recent work on simultaneous trajectory estimation and mapping (STEAM) for mobile robots has used Gaussian processes (GPs) to efficiently represent the robot's trajectory through its environment. GPs have several advantages over discrete-time trajectory representations: they can represent a continuous-time trajectory, elegantly handle asynchronous and sparse measurements, and allow the robot to query the trajectory to recover its estimated position at any time of interest. A major drawback of the GP approach to STEAM is that it is formulated as a *batch* trajectory estimation problem. In this paper we provide the critical extensions necessary to transform the existing GP-based batch algorithm for STEAM into an extremely efficient incremental algorithm. In particular, we are able to vastly speed up the solution time through efficient variable reordering and incremental sparse updates, which we believe will greatly increase the practicality of Gaussian process methods for robot mapping and localization. Finally, we demonstrate the approach and its advantages on both synthetic and real datasets.

1. Introduction & related work

Simultaneously recovering the location of a robot and a map of its environment from sensor readings is a fundamental challenge in robotics [1–3]. Well-known approaches to this problem, such as square root smoothing and mapping (SAM) [4], have focused on regression-based methods that exploit the sparse structure of the problem to efficiently compute a solution. The main weakness of the original SAM algorithm was that it was a *batch* method: all of the data must be collected before a solution can be found. For a robot traversing an environment, the inability to update an estimate of its trajectory online is a significant drawback. In response to this weakness, Kaess et al. [5] developed a critical extension to the batch SAM algorithm, iSAM, that overcomes this problem by *incrementally* computing a solution. The main drawback of iSAM, was that the approach required costly periodic batch steps for variable reordering to maintain sparsity and relinearization. This approach was extended in iSAM 2.0 [6], which employs an efficient data structure called the *Bayes tree* [7] to perform incremental variable reordering and just-in-time relinearization, thereby eliminating the bottleneck caused by batch variable reordering and relinearization. The iSAM 2.0 algorithm and its extensions are widely considered to be state-of-the-art in robot trajectory estimation and mapping.

The majority of previous approaches to trajectory estimation and mapping, including the smoothing-based SAM family of algorithms, have formulated the problem in discrete time [1–4,6,8,9]. However, discrete-time representations are restrictive: they are not easily extended to trajectories with irregularly spaced waypoints or asynchronously sampled measurements. A continuous-time formulation of the SAM problem where measurements constrain the trajectory at any point in time, would elegantly contend with these difficulties. Viewed from this perspective, the robot trajectory is a *function* $\boldsymbol{x}(t)$, that maps any time t to a robot

* Corresponding author.
E-mail address: xinyan.yan@cc.gatech.edu (X. Yan).

http://dx.doi.org/10.1016/j.robot.2016.10.004
0921-8890/

state. The problem of estimating this function along with landmark locations has been dubbed *simultaneous trajectory estimation and mapping* (STEAM) [10].

Tong et al. [11] proposed a Gaussian process (GP) regression approach to solving the STEAM problem. While their approach was able to accurately model and interpolate asynchronous data to recover a trajectory and landmark estimate, it suffered from significant computational challenges: naive Gaussian process approaches to regression have notoriously high space and time complexity. Additionally, Tong et al.'s approach is a *batch* method, so updating the solution necessitates saving all of the data and completely re-solving the problem. In order to combat the computational burden, Tong et al.'s approach was extended in Barfoot et al. [10] to take advantage of the sparse structure inherent in the STEAM problem. The resulting algorithm significantly speeds up solution time and can be viewed as a continuous-time analog of Dellaert's original square-root SAM algorithm [4]. Unfortunately, like SAM, Barfoot et al.'s GP-based algorithm remains a batch algorithm, which is a disadvantage for robots that need to continually update the estimate of their trajectory and environment.

In this work, we provide the critical extensions necessary to transform the existing Gaussian process-based approach to solving the STEAM problem into an extremely efficient incremental approach. Our algorithm elegantly combines the benefits of Gaussian processes and iSAM 2.0. Like the GP regression approaches to STEAM, our approach can model continuous trajectories, handle asynchronous measurements, and naturally interpolate states to speed up computation and reduce storage requirements, and, like iSAM 2.0, our approach uses a Bayes tree to efficiently calculate a *maximum a posteriori* (MAP) estimate of the GP trajectory while performing incremental factorization, variable reordering, and just-in-time relinearization. The result is an online GP-based solution to the STEAM problem that remains computationally efficient while scaling up to large datasets.

The present paper is an extension of the work presented in [12,13]. As a further contribution, in this manuscript we elaborate more on variable re-ordering that is key to making both batch and incremental GP-regression computationally more efficient, and we study the performance of the proposed approach in an additional real-world dataset. Furthermore, we release an efficient implementation of the approach developed herein as open source code.[1]

2. Batch trajectory estimation & mapping as Gaussian process regression

We begin by describing how the simultaneous trajectory estimation and mapping (STEAM) problem can be formulated in terms of Gaussian process regression. Following Tong et al. [11] and Barfoot et al. [10], we represent robot trajectories as functions of time t sampled from a Gaussian process:

$$\boldsymbol{x}(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \mathcal{K}(t, t')), \qquad t_0 < t, t'. \tag{1}$$

Here, $\boldsymbol{x}(t)$ is the continuous-time trajectory of the robot through state-space, represented by a Gaussian process with mean $\boldsymbol{\mu}(t)$ and covariance $\mathcal{K}(t, t')$ functions.

We next define a finite set of measurements:

$$\boldsymbol{y}_i = \boldsymbol{h}_i(\boldsymbol{\theta}_i) + \boldsymbol{n}_i, \quad \boldsymbol{n}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_i), \quad i = 1, 2, \ldots, N. \tag{2}$$

The measurement $\boldsymbol{y}_i$ can be any linear or nonlinear functions of a set of related variables $\boldsymbol{\theta}_i$ plus some Gaussian noise $\boldsymbol{n}_i$. The related variables for a range measurement are the robot state at the corresponding measurement time $\boldsymbol{x}(t_i)$ and the associated landmark location $\boldsymbol{\ell}_j$. We assume the total number of measurements is N, and the number of trajectory states at measurement times is M.

Based on the definition of Gaussian processes, any finite collection of robot states has a joint Gaussian distribution [14]. So the robot states at measurement times are normally distributed with mean $\boldsymbol{\mu}$ and covariance $\mathcal{K}$.

$$\begin{aligned} &\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathcal{K}), \quad \boldsymbol{x} = [\boldsymbol{x}_1^\mathsf{T} \quad \ldots \quad \boldsymbol{x}_M^\mathsf{T}]^\mathsf{T}, \quad \boldsymbol{x}_i = \boldsymbol{x}(t_i) \\ &\boldsymbol{\mu} = [\boldsymbol{\mu}_1^\mathsf{T} \quad \ldots \quad \boldsymbol{\mu}_M^\mathsf{T}]^\mathsf{T}, \quad \boldsymbol{\mu}_i = \boldsymbol{\mu}(t_i), \quad \mathcal{K}_{ij} = \mathcal{K}(t_i, t_j). \end{aligned} \tag{3}$$

Note that any point along the continuous-time trajectory can be estimated from the Gaussian process model. Therefore, the trajectory does not need to be discretized and robot trajectory states do not need to be evenly spaced in time, which is an advantage of the Gaussian process approach over discrete-time approaches (e.g. Dellaert's square-root SAM [4]).

The landmarks $\boldsymbol{\ell}$ which represent the map are assumed to conform to a joint Gaussian distribution with mean $\boldsymbol{d}$ and covariance $\boldsymbol{W}$ (Eq. (4)). The prior distribution of the combined state $\boldsymbol{\theta}$ that consists of robot trajectory states at measurement times and landmarks is, therefore, a joint Gaussian distribution (Eq. (5)).

$$\boldsymbol{\ell} \sim \mathcal{N}(\boldsymbol{d}, \boldsymbol{W}), \qquad \boldsymbol{\ell} = [\boldsymbol{\ell}_1^\mathsf{T} \quad \boldsymbol{\ell}_2^\mathsf{T} \quad \ldots \quad \boldsymbol{\ell}_O^\mathsf{T}]^\mathsf{T} \tag{4}$$

$$\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\eta}, \mathcal{P}), \qquad \boldsymbol{\eta} = [\boldsymbol{\mu}^\mathsf{T} \quad \boldsymbol{d}^\mathsf{T}]^\mathsf{T}, \qquad \mathcal{P} = \begin{bmatrix} \mathcal{K} & \\ & \boldsymbol{W} \end{bmatrix}. \tag{5}$$

To solve the STEAM problem, given the prior distribution of the combined state and the likelihood of measurements, we compute the *maximum a posteriori* (MAP) estimate of the combined state *conditioned* on measurements via Bayes' rule:

$$\begin{aligned} \boldsymbol{\theta}^* \triangleq \boldsymbol{\theta}_{MAP} &= \operatorname*{argmax}_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\boldsymbol{y}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \frac{p(\boldsymbol{\theta})p(\boldsymbol{y}|\boldsymbol{\theta})}{p(\boldsymbol{y})} \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} p(\boldsymbol{\theta})p(\boldsymbol{y}|\boldsymbol{\theta}) = \operatorname*{argmin}_{\boldsymbol{\theta}} (-\log p(\boldsymbol{\theta}) - \log p(\boldsymbol{y}|\boldsymbol{\theta})) \\ &= \operatorname*{argmin}_{\boldsymbol{\theta}} \left(\|\boldsymbol{\theta} - \boldsymbol{\eta}\|_{\mathcal{P}}^2 + \|\boldsymbol{h}(\boldsymbol{\theta}) - \boldsymbol{y}\|_{\boldsymbol{R}}^2 \right) \end{aligned} \tag{6}$$

where the norms are Mahalanobis norms defined as: $\|\boldsymbol{e}\|_{\Sigma}^2 = \boldsymbol{e}^\mathsf{T}\Sigma^{-1}\boldsymbol{e}$, and $\boldsymbol{h}(\boldsymbol{\theta})$ and $\boldsymbol{R}$ are the mean and covariance of the measurements collected, respectively:

$$\boldsymbol{h}(\boldsymbol{\theta}) = [\boldsymbol{h}_1(\boldsymbol{\theta}_1)^\mathsf{T} \quad \boldsymbol{h}_2(\boldsymbol{\theta}_2)^\mathsf{T} \quad \ldots \quad \boldsymbol{h}_N(\boldsymbol{\theta}_N)^\mathsf{T}]^\mathsf{T} \tag{7}$$

$$\boldsymbol{R} = \operatorname{diag}(\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots, \boldsymbol{R}_N). \tag{8}$$

Because both covariance matrices $\mathcal{P}$ and $\boldsymbol{R}$ are positive definite, the objective in Eq. (6) corresponds to a least squares problem. Consequently, if some of the measurement functions $\boldsymbol{h}_i(\cdot)$ are nonlinear, this becomes a nonlinear least squares problem, in which case iterative methods including Gauss–Newton and Levenberg–Marquardt [15] can be utilized; in each iteration, an optimal update is computed given a linearized problem at the current estimate. A linearization of a measurement function at current state estimate $\bar{\boldsymbol{\theta}}_i$ can be accomplished by a first-order Taylor expansion:

$$\boldsymbol{h}_i\left(\bar{\boldsymbol{\theta}}_i + \delta\boldsymbol{\theta}_i\right) \approx \boldsymbol{h}_i(\bar{\boldsymbol{\theta}}_i) + \left.\frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{\theta}_i}\right|_{\bar{\boldsymbol{\theta}}_i} \delta\boldsymbol{\theta}_i. \tag{9}$$

Combining Eq. (9) with Eq. (6), the optimal increment $\delta\boldsymbol{\theta}^*$ at the current combined state estimate $\bar{\boldsymbol{\theta}}$ is

$$\delta\boldsymbol{\theta}^* = \operatorname*{argmin}_{\delta\boldsymbol{\theta}} \left(\|\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta} - \boldsymbol{\eta}\|_{\mathcal{P}}^2 + \|\boldsymbol{h}(\bar{\boldsymbol{\theta}}) + \boldsymbol{H}\delta\boldsymbol{\theta} - \boldsymbol{y}\|_{\boldsymbol{R}}^2 \right) \tag{10}$$

$$\boldsymbol{H} = \operatorname{diag}(\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots, \boldsymbol{H}_N), \qquad \boldsymbol{H}_i = \left.\frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{\theta}_i}\right|_{\bar{\boldsymbol{\theta}}_i} \tag{11}$$

where $\boldsymbol{H}$ is the measurement Jacobian matrix. To solve the linear least squares problem in Eq. (10), we take the derivative with

[1] Please check out the code at https://github.com/XinyanGT/online-gpslam-code.