Continuous-Time State Estimation Methods in Robotics: A Survey

William Talbot, *Graduate Student Member, IEEE*, Julian Nubert, Turcan Tuna, *Graduate Student Member, IEEE*, Cesar Cadena, Frederike Dümbgen, Jesús Tordesillas, Timothy D. Barfoot, and Marco Hutter, *Member, IEEE*

(Survey Paper)

***Abstract*—Accurate, efficient, and robust state estimation is more important than ever in robotics as the variety of platforms and complexity of tasks continue to grow. Historically, discrete-time filters and smoothers have been the dominant approach, in which the estimated variables are states at discrete sample times. The paradigm of continuous-time state estimation proposes an alternative strategy by estimating variables that express the state as a continuous function of time, which can be evaluated at any query time. Not only can this benefit downstream tasks such as planning and control, but it also significantly increases estimator performance and flexibility, as well as reduces sensor preprocessing and interfacing complexity. Despite this, continuous-time methods remain underutilized, potentially because they are less well-known within robotics. To remedy this, this work presents a unifying formulation of these methods and the most exhaustive literature review to date, systematically categorizing prior work by methodology, application, state variables, historical context, and theoretical contribution to the field. By surveying splines and Gaussian process together and contextualizing works from other research domains, this work identifies and analyzes open problems in continuous-time state estimation and suggests new research directions.**

***Index Terms*—Continuous-time (CT), Gaussian process, optimization, sensor fusion, splines, state estimation.**

Received 28 March 2025; accepted 23 June 2025. Date of publication 28 July 2025; date of current version 27 August 2025. This work was supported by the Swiss National Science Foundation (SNSF) [20CH-1_229464/1] through CHIST-ERA under Grant CHIST-ERA-23-MultiGIS-07, ANR JCJC project NIMBLE under Grant ANR-22-CE33-0008 and Grant ANR-22-CE33-0008, and in part by the European Union's Horizon Europe research and innovation programme under the Marie Skłodowska-Curie under Grant 101207106 and Grant 101207106. This article was recommended for publication by Associate Editor M. Ghaffari and Editor J. Civera upon evaluation of the reviewers' comments. *(Corresponding author: William Talbot.)*

William Talbot, Turcan Tuna, Cesar Cadena, and Marco Hutter are with the Robotic Systems Lab (RSL), ETH Zürich, 8092 Zürich, Switzerland (e-mail: wtalbot@ethz.ch; tutuna@ethz.ch; cesarc@ethz.ch; mahutter@ethz.ch).

Julian Nubert is with the Robotic Systems Lab (RSL), ETH Zürich, 8092 Zürich, Switzerland, and also with the Max Planck Institute (MPI) for Intelligent Systems, 70569 Stuttgart, Germany (e-mail: jnubert@ethz.ch).

Frederike Dümbgen is with the Willow, Inria, Computer Science Department of ENS, PSL Research University, 75006 Paris, France (e-mail: frederike.duembgen@inria.fr).

Jesús Tordesillas is with the Institute for Research in Technology, ICAI School of Engineering, Comillas Pontifical University, 28015 Madrid, Spain (e-mail: jtordesillas@comillas.edu).

Timothy D. Barfoot is with the Autonomous Space Robotics Laboratory (ASRL), University of Toronto, Toronto, ON M5S 1A1, Canada (e-mail: tim.barfoot@utoronto.ca).

Digital Object Identifier 10.1109/TRO.2025.3593079

I. Introduction

Autonomous driving, extra-planetary exploration, infrastructure inspection, and environmental monitoring are examples of the complex tasks expected of robotic platforms today. To cope with these diverse challenges, onboard sensors have increased in quantity, variety, and bandwidth, and the requirement for accurate and robust state estimation is more critical than ever. The constraints of these platforms, including cost, size, sensor quality, power consumption, and computational resources, vary significantly. Some platforms, such as commercial drones or handheld devices, are highly constrained, while autonomous cars, construction machines, and others may have access to more onboard compute. As the demand for such diverse robotic systems grows in the coming decades, so will the requirement for general, flexible, and scalable state estimation solutions. A paradigm shift, from discrete to continuous-time, may be the key to achieving this.

Discrete-time (DT): Historically, state estimation techniques have operated in discrete-time, which means the state of the robot or process is estimated at specific moments in time. However, the evolution of the system is not modeled, and intermediate states cannot be inferred. These times must include the measurement times, so regardless of how quickly the system state evolves, DT methods require estimating variables at all of these sample times [1], [2], [3]. The computation of such algorithms scales poorly as the number and frequency of sensors increase, compromising their ability to run in real-time. Several techniques and tricks are employed to work around this issue. One of the most common is to aggregate measurements over time, especially from sensors that capture hundreds or thousands of measurements per second, to form lower frequency *pseudo*measurements. Examples include the accumulation of points from a sweeping light detection and ranging (LiDAR) sensor for interscan or scan-to-map registration [4], [5], [6], [7], [8], feature extraction in event [9] or rolling shutter (RS) cameras, and inertial measurement unit (IMU) preintegration [10], [11], [12], [13], [14]. Motion distortion may occur if the platform moves substantially over this accumulation period. motion distortion correction (MDC) methods such as LiDAR deskewing [15] or RS compensation [16] attempt to correct for this. However, MDC is fundamentally a *chicken-and-egg* problem

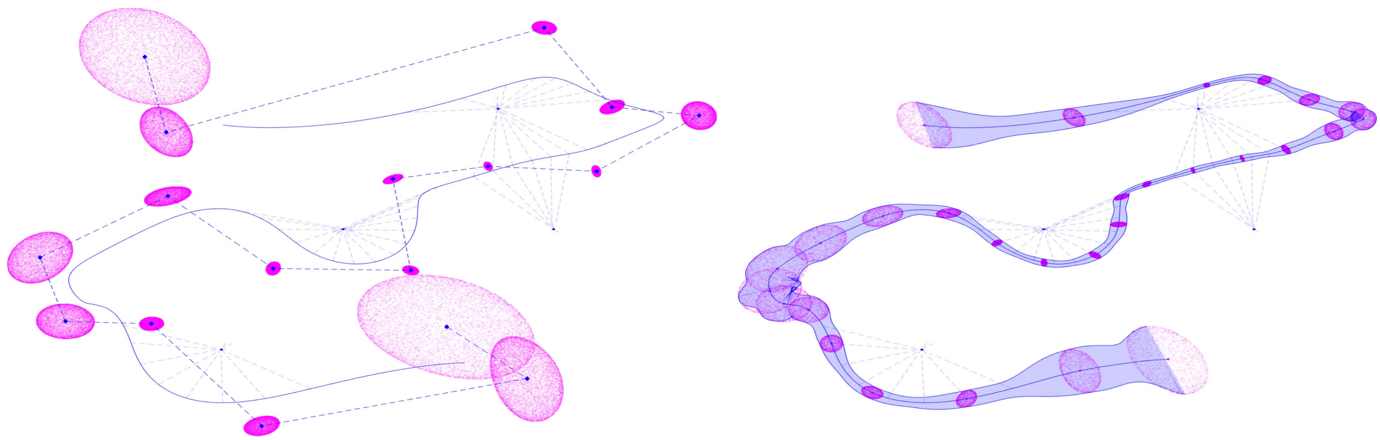

Fig. 1. Example of 2-D localization from noisy accelerometer, gyroscope, and range-bearing measurements (shown projected from interpolated poses) using two popular Continuous-time (CT) methods; B-splines (left), and "exactly sparse" Gaussian process (right). The Laplace approximation for the posterior (6) is obtained via batch optimization. For the uniform cubic B-spline, the $SE(2)$ control points (diamonds) are estimated, while for the uniform *"constant-jerk"* Gaussian process, $SE(2)$ states on the trajectory are estimated. The 3σ uncertainties for the estimated variables (pink) and the interpolated $SE(2)$ trajectories (blue) are shown. Since the latter method supports efficient covariance interpolation in manifold spaces, a 3σ uncertainty envelope is shown around the trajectory.

since the motion-corrected measurements are required to estimate the motion accurately. Consequentially, these methods introduce hard-to-model errors into the system. Some DT systems (e.g., [17]) also achieve state reduction by triggering sensors to capture simultaneously and often have complicated hardware triggering systems and clock synchronization in place to ensure that the timestamps reported by sensors are well-aligned to a central clock. Yet, in systems without proper time synchronization, estimating time offsets between sensor clocks is a complex task for DT state estimators [18], [19].

Continuous-time (CT): Continuous-time methods (see Fig. 1) differ by modeling the underlying process as a continuous function of time and have the defining characteristic that the state can be queried at any time (sometimes within certain bounds). This capability can, on the one hand, be useful for downstream tasks that rely on accurate state estimates, such as planning or control. On the other hand, it allows querying *interpolated* states at measurement times during estimation without requiring additional explicitly defined variables. By providing a layer of abstraction between the time-varying state and the optimization variables, CT methods allow the number of variables to be flexible and their generation to be application and sensor-agnostic. This property facilitates an elegant means of trading off accuracy and computation time while considering the robot's dynamics. Such a system encourages the direct inclusion of asynchronous high-frequency sensor measurements, such as from an IMU or LiDAR, without necessarily creating new estimation variables, avoiding the errors introduced by measurement aggregation. Furthermore, these systems can jointly estimate the time offsets between sensor clocks by shifting the interpolation time [19].

Significance of this work: The CT paradigm could be the key to accurate, robust, and scalable state estimation. Indeed, CT methods are already being leveraged to achieve state-of-the-art performance across a full spectrum of applications. However, they remain relatively niche and less well-understood within the broader robotics community. The field has progressed substantially since the last CT survey almost a decade ago [20]. Other review works have focused more specifically on a single method [19] or sensor modality [21], or on comparative performance [22]. To update the community on modern CT methods and lower the bar of entry to future research, this work makes following contributions:

1) A concise background (see Section II) and consolidated formulations for the established CT methods (see Section III);
2) The most complete survey of CT state estimation to date (until October 2024), categorized by method, application, state variables, historical context, or theoretical contribution, as most appropriate (see Section IV);
3) Identification and analysis of open problems (see Section V);
4) A discussion of the scope of CT methods in other robotics' domains and applications (see Section VI).

II. State Estimation in Robotics

A. Preliminaries

This section provides a brief introduction to the field of state estimation and establishes a common notation for DT estimation and the CT methods introduced in Section III. In the interests of conciseness, a detailed theory of optimization-based state estimation theory will not be covered as has been done in prior works [23], [24], [25], [26]. This work considers state estimation problems, which aim to tractably estimate the probability distribution of a (finite) set of (continuous) random variables that parametrize a system of interest. Lie theory [27], [28], [29] has proven to be a valuable framework for treating these variables accurately in the context of state estimation in robotics; e.g., the $SO(n)$ and $SE(n)$ groups for representing n-dimensional orientations and poses. Because Lie groups are smooth, differentiable manifolds, they have unique tangent spaces, which are vector spaces amenable to the linear algebra operations required during optimization. This work will adopt the notation and *right*-handed conventions of Sola et al. [29].

1) States: A continuously evolving process is defined by two components: *i)* A time-varying state, denoted as $\mathbf{x}(t)$ for time variable t, such as the $SE(3)$ pose of a robot, and *ii)* its time-invariant context, a set of variables denoted as $\boldsymbol{\theta}$, such as sensor calibration or map parameters. Without yet proposing a relationship to $\mathbf{x}(t)$, a set of variables $\mathbf{x} = (\mathbf{x}_0, \ldots, \mathbf{x}_N)$ is defined to describe $\mathbf{x}(t)$. Estimating these variables, collected together as $\boldsymbol{\Theta} = \{\mathbf{x}, \boldsymbol{\theta}\}$, will then be possible by incorporating measurements and prior knowledge.

2) Measurements: A measurement model describes how an observation $\tilde{\mathbf{z}}$ is generated from one or more sensors. The *measurement function* for this model, $\mathbf{z} = \mathbf{h}(\mathbf{x}, \boldsymbol{\theta}, \mathbf{n})$, defines a (measured) observation as a function of the state and associated noise $\mathbf{n}$, usually assumed to be drawn from a Gaussian distribution and independent between measurements. Common examples in robotics include range-bearing, projective geometry, kinematic, and inertial measurement models. A *residual function* is defined to obtain a vector-valued residual $\mathbf{e} = \mathbf{g}(\mathbf{z}, \tilde{\mathbf{z}})$, which is commonly subtraction or the generalized $\ominus$ operator [29, Eq. (26)] for Lie groups. Alternatively, $\mathbf{h}(\cdot)$ and $\mathbf{g}(\cdot)$ may be implicitly combined such as in point-to-plane [5], [30] or on-unit-sphere angular [31] distances.

3) Priors: Prior models capture assumptions about a system before evidence from measurement models is incorporated. They can be formulated in the same way as measurement models, except that $\tilde{\mathbf{z}}$ is not observed but set *a priori*. In practice, a belief about the system's initial conditions (e.g., pose and velocity) is often required to ensure a unique solution in the optimization. A more complex example is planar surface regularization in highly structured environments. As with measurement models, residuals can be defined for such priors.

4) Processes: Process models (a.k.a. system dynamics, motion models) describe the evolution of the state over time. Process models have frequently appeared in DT filters, occasionally in optimization-based approaches, and are a fundamental idea used in some CT methods. Since the process model is known *a priori* and does not require measurements, it can be seen as a type of Bayesian prior. If the process is *Markovian*, then it can be formulated in terms of the current process state $\mathbf{x}(t)$, process inputs $\mathbf{u}(t)$, zero-mean process noise $\mathbf{w}(t)$, and context $\boldsymbol{\theta}$, as a nonlinear time-varying (NTV) stochastic differential equation (SDE) as follows:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta}, \mathbf{u}(t), \mathbf{w}(t)). \tag{1}$$

Commonly, $\mathbf{f}$ does not depend on the time-invariant parameters $\boldsymbol{\theta}$ and is linearized about a current estimate of the other inputs as follows:

$$\dot{\mathbf{x}}(t) \approx \mathbf{F}(t)\mathbf{x}(t) + \mathbf{v}(t) + \mathbf{L}(t)\mathbf{w}(t) \tag{2}$$

$$\text{with } \mathbf{v}(t) := \mathbf{f}(\bar{\mathbf{x}}(t), \mathbf{u}(t), \mathbf{0}) - \mathbf{F}(t)\bar{\mathbf{x}}(t) \tag{3}$$

$$\mathbf{F}(t) := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}(t), \mathbf{u}(t), \mathbf{0}}, \text{ and } \mathbf{L}(t) := \left.\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right|_{\bar{\mathbf{x}}(t), \mathbf{u}(t), \mathbf{0}}. \tag{4}$$

The general solution of this linear time-varying (LTV) SDE [32], [33], given some initial state $\mathbf{x}(t')$ at time t', is as follows:

$$\mathbf{x}(t) = \boldsymbol{\Phi}(t, t')\mathbf{x}(t') + \int_{t'}^{t} \boldsymbol{\Phi}(t, s)\left(\mathbf{v}(s) + \mathbf{L}(s)\mathbf{w}(s)\right) ds. \tag{5}$$

Here, $\boldsymbol{\Phi}(t, t')$ is the *transition matrix* for the process, with properties $\boldsymbol{\Phi}(t, t') = \boldsymbol{\Phi}(t, s)\boldsymbol{\Phi}(s, t')$ for any $s \in [t', t]$, and $\boldsymbol{\Phi}(t, t) = \mathbf{I}$. It can be found analytically for simple LTV-SDEs but can be challenging to find in general, and so without an analytical solution, it must be computed numerically [34], [35], [36].

5) Maximum a posteriori (MAP) estimation: Modern state estimation approaches are fundamentally probabilistic, relying on the principle of Bayesian inference. The application of Bayesian theory to DT state estimation in robotics has been covered thoroughly in prior work [25], [26], [37], [38], and the findings transfer to CT. One powerful consequence of this theory is that under the assumption that all residual errors (e.g., from measurement, prior, and process models) are drawn from zero-mean Gaussian distributions, an optimal estimate (in the MAP sense) of the state is obtainable by minimizing the sum of their squared Mahalanobis distances. Specifically, for M residuals with individual covariances $\boldsymbol{\Sigma}_j$, a Gaussian approximation for the posterior distribution $\mathcal{N}(\boldsymbol{\Theta}_{\text{MAP}}, \boldsymbol{\Sigma}_{\text{MAP}})$ (a.k.a. *Laplace approximation*) is found [39] by solving

$$\begin{aligned} \boldsymbol{\Theta}_{\text{MAP}} &:= \arg\min_{\boldsymbol{\Theta}} \sum_{j=0}^{M} \|\mathbf{e}_j\|^2_{\boldsymbol{\Sigma}_j} \\ \boldsymbol{\Sigma}^{-1}_{\text{MAP}} &:= \left.\frac{\partial^2}{\partial \boldsymbol{\Theta}^2} \sum_{j=0}^{M} \|\mathbf{e}_j\|^2_{\boldsymbol{\Sigma}_j}\right|_{\boldsymbol{\Theta}_{\text{MAP}}}. \end{aligned} \tag{6}$$

B. Discrete-time (DT) State Estimation

In Discrete-time (DT) state estimation, the process state variables $\mathbf{x}$ to be estimated are defined as the process state itself at a specific set of times $(t_0, \ldots, t_N)$, i.e., $\mathbf{x}_i := \mathbf{x}(t_i)$. There are two broad categories of approaches for solving DT state estimation problems: *filtering* and *smoothing*.

1) Filtering: Bayesian filtering adopts a predict and update pattern that relies on the Markov assumption. This assumption states that the evolution of a process, such as robot motion, can be inferred from its current state only and does not depend on how this state was reached. In the *predict* step, the prior belief of the state is propagated according to a process model. Then, in the *update* step, measurements are used to estimate the posterior distribution for the state. Kalman filters (KFs) [40], [41], [42] and information filters (IFs) [37] have proven to be highly effective and efficient, and are still widely used, especially when latency and computational load must be low. If the state has a non-Gaussian distribution, discretized state-space methods such as the histogram filter, or Monte Carlo methods such as the particle filter (PF), which uses samples to represent the state probability distribution, are often used instead [37].

2) Smoothing: Bayesian smoothing estimates the joint probability distribution of variables representing (a window of) the current and previous robot states, typically relying on optimization techniques. Smoothing methods have gained popularity in many robotics problems as they have matured in recent decades regarding robustness, accuracy, and speed. Due to the Markov property, variables involved in the optimization are usually related to only a small subset of (neighboring) variables. These approaches demonstrate speeds compatible with onboard robot

TABLE I
COMPARISON OF CT STATE ESTIMATION METHODS

Specifications	**Linear Interpolation**	**Temporal Splines**	**TGPs**
Interpolation	linear	polynomial	process model
Lie Group Support	✓	✓	✓
Covariance Interpolation	✓	Section V-A6	✓
Extrapolation	✓	[47]	✓
Non-Interpolated Residual Support	✓	✗	✓
Initialization	linear extrapolation	Section V-A5	process model
Process Variables $\mathbf{x}$	explicit states	control points	explicit (Markovian) states
Derivatives	constant 1st derivative	computable (exact)	within state (probabilistic)
Variable $\mathbf{x}_i$ Size	size of $\mathcal{L}$	size of $\mathcal{L}$	(size of $\mathcal{L}$) + (DoF of $\mathcal{L}$) $\times$ (derivatives)
Variables per Interpolation	2	k	2
Design Choices	-	spline type (e.g., B-spline)	process model (e.g., WNOA)
	state representation (e.g., joint/split pose)	state representation (e.g., joint/split pose)	state representation (e.g., joint/split pose)
	-	order k (e.g., k = 4 for cubic)	hyperparameters (e.g., $\mathbf{Q}_C$)
	state times	knots	state times
Characteristics	parametric	parametric	non parametric
	2 TBFs	k TBFs	∞ TBFs (kernel trick)

operation by exploiting this inherent (near diagonal) sparsity through efficient linear algebra tools (e.g., sparse LU, LDL, Cholesky, and QR factorization), which are the basis of efficient optimal solutions of linear problems and fast iterative solvers based on relinearization for nonlinear problems (e.g., Gauss–Newton (GN) [24], Levenberg–Marquardt, steepest descent, Powell's dogleg) [25].

C. Continuous-time (CT) State Estimation

Since processes in the physical world usually change continuously over time, it is natural to seek a continuous-time representation to model them. The fundamental property of CT methods is their ability to query the state $\mathbf{x}(t)$ at any required time t, given a set of variables $\mathbf{x} = (\mathbf{x}_0, \ldots, \mathbf{x}_N)$. The exact relationship between $\mathbf{x}(t)$ and $\mathbf{x}$ depends on the chosen method. Yet, a key advantage of these methods is that the number and temporal placement of variables are flexible. Intuitively, the quantity of variables must scale with the duration of the process and needs to be increased to accurately model more complex dynamics. Notably, these methods do not require variables for every measurement time and can naturally accommodate asynchronous sensor data through their inherent ability to interpolate (or extrapolate). To remain computationally competitive with DT methods, CT methods rely on the *local support* property, meaning that inferring a state at a particular time depends only on a small number of (local) variables.

The robotic research community has experimented with several CT methods for state estimation over the past years. *1)*Interpolation [e.g., Linear interpolation (LI)] and integration methods (see Section III-A) appear frequently, in part due to their simplicity and low-computational complexity. *2)*Temporal splines (see Section III-B), especially B-splines, are among the most popular CT methods to emerge. Through *control points* and *knots*, splines embrace the abstraction of state inference from estimated variables. *3)*"Exactly Sparse" Temporal Gaussian Process (TGPs) [1] (see Section III-C) are a recent, compelling approach. Table I provides a high-level overview. The spline and GP approaches have adopted the optimization-based smoothing paradigm, with few exceptions [45], [46], which may be partly attributed to the way in which these methods are conventionally formulated. Hence, this work focuses primarily on their use in an optimization context.

III. THEORY

A. Interpolation and Integration

When interpolation and integration are discussed in the context of CT state estimation, the state variables precisely represent the process state (i.e., $\mathbf{x}_i := \mathbf{x}(t_i)$). Interpolation is defined as the inference of intermediate states from estimated curves that pass exactly through these explicit states.[2] On the other hand, integration uses rate measurements, such as the angular rate from an IMU, for this inference.

1) Linear interpolation (LI): One of the simplest and fastest interpolation methods is Linear interpolation (LI). The *generalized* linear interpolation of two adjacent states can be written as follows:

$$\mathrm{GLERP}(\mathbf{x}_i, \mathbf{x}_{i+1}, \alpha) = \mathbf{x}_i \boxplus (\alpha \cdot (\mathbf{x}_{i+1} \boxminus \mathbf{x}_i)) \tag{7}$$

where $\boxplus, \boxminus$ are the addition and subtraction operators for composite manifolds [29, Sec. IV], and $\alpha := \frac{t-t_i}{t_{i+1}-t_i} \in [0, 1)$ for interpolation time $t \in [t_i, t_{i+1})$ (outside is extrapolation). For Lie groups, these operators become $\oplus, \ominus$ [29, Eqs. (25) and (26)], which when applied to quaternions [48] or rotation matrices [24] yield *spherical linear interpolation* (SLERP). This has also been applied to $SE(3)$ poses [24]. When the states belong to vector space, this simplifies to

$$\mathrm{LERP}(\mathbf{x}_i, \mathbf{x}_{i+1}, \alpha) := \mathbf{x}_i + \alpha(\mathbf{x}_{i+1} - \mathbf{x}_i) = (1-\alpha)\mathbf{x}_i + \alpha\mathbf{x}_{i+1}. \tag{8}$$

Importantly, LI maintains a constant rate of change ("velocity") while other interpolation methods, such as quaternion linear blending (QLB) [48] do not. LI, therefore, enables rapid inference of the state at any time under this constant-rate assumption, and allows the number of optimized states to be chosen flexibly. While nonlinear interpolation methods exist, they are usually captured within the scope of temporal basis functions (TBFs), the foundation for temporal splines [20] and TGPs [49]. Indeed,

[1]These are not to be confused with *sparse* GPs [43], which address the cubic complexity of GPs by selecting *active points* as a subset of training points or optimizable pseudoinputs that account for the entire training set [44].

[2]However, the term interpolation is frequently overloaded, and so is used more liberally in later sections to describe inference with regression models that do not pass through every data point. This is common in the spline literature, although only some spline types are *interpolating*.

LI is exactly a special case of these methods. Despite the development of more advanced methods, LI remains popular for its simplicity and speed, and may be sufficiently accurate to model many processes.

2) Numerical Integration: Numerical integration in robotics is sometimes discussed in the CT state estimation literature, as it allows inference of intermediate states from rate measurements. However, unlike the other approaches, numerical integration does not provide a closed-form state inference function. In robotics, this is often used for the motion compensation of scanning sensors such as LiDARs or RS cameras, or for *IMU preintegration*. A common method is *Riemann summation* (a.k.a. the *rectangle rule*), which assumes a constant rate of change and, hence, is closely related to LI. When formulated as an ODE, numerically integrating constant rates over short time intervals is known as the *forward Euler method*. Regressing and integrating Gaussian processes (GPs) for IMU preintegration has also been explored (see Section IV-A6).

B. Temporal Splines

A number of different formulations for splines have been presented in the literature [20], [31], [47], [48], [50], [51], [52], [53], [54], [55]. The formulation presented in this work is primarily based on the works of Furgale et al. [20] and Sommer et al. [55]. A *spline* is a function of a scalar input whose resultant curve (in vector or manifold space) is a piecewise polynomial of *degree* m.[3] The generated curve is composed of *segments*. The *order* $k \in \mathbb{Z}^+$ is a fundamental property of a spline guaranteeing certain continuity properties and is related to the degree as $k = m + 1$. In *temporal* splines, the input scalar values represent time.

A set of *control points* $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1, \dots)$ (the estimation variables) and a set of scalar times called *knots* $(t_0, t_1, \dots)$ characterize a temporal spline. Each spline segment depends only on k control points. This is the *local support* property for splines and is key for efficient computation and optimization. The ith polynomial segment (indexing from $i = 0$) depends on a subset of the control points $(\mathbf{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_{i+k-1})$ and is defined on the interval $[t_{i+k-2}, t_{i+k-1})$.[4] A time t lying within this interval can be normalized as follows:

$$u_i(t) = \frac{t - t_{i+k-2}}{t_{i+k-1} - t_{i+k-2}} \in [0, 1). \tag{9}$$

Polynomial powers of this normalized time are then stacked into the vector $\mathbf{u}_i(t) = [1 \quad u_i(t) \quad u_i(t)^2 \quad \dots \quad u_i(t)^m]^T$.

1) Interpolation: Vector Space: Through $\mathbf{M}_i$, the *blending matrix*, one can compute the coefficients $\lambda_{i,j}(t)$ of the *blending vector* $\boldsymbol{\lambda}_i(t) = \mathbf{M}_i \mathbf{u}_i(t) \in \mathbb{R}^k$, that enable interpolation on the ith segment of the curve. Interpolation on the ith segment is simply

$$\mathbf{x}_i(t) = \sum_{j=0}^{k-1} \lambda_{i,j}(t) \mathbf{x}_{i+j}. \tag{10}$$

[3]Technically, splines are a special case of temporal basis functions (TBFs) where the k bases are the polynomials of degree 0 to m.

[4]To capture the degenerate $k = 1$ case in a manner consistent with higher orders, define $t_{-1} := -\infty$, and when no knots are defined, $t_0 := \infty$.

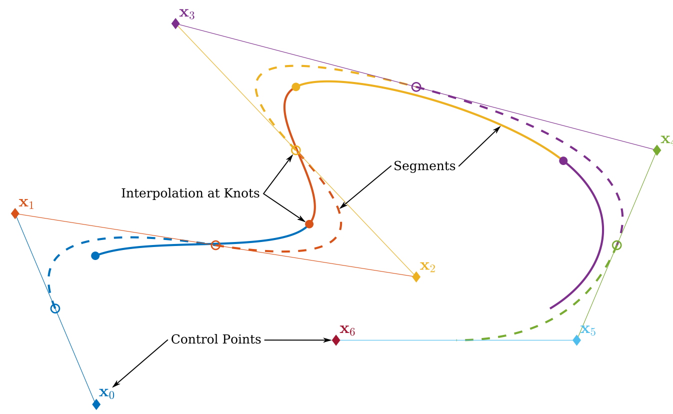

Fig. 2. Uniform vector-space B-splines of different orders. The control points, which are also exactly the points of the degenerate $k = 1$ spline, are shown as diamonds. The linear $k = 2$ spline (thin solid line) is the connection between these control points. The quadratic $k = 3$ spline (dashed line) and cubic $k = 4$ spline (solid line) are also shown. Each segment is colored to correspond with the first of the k control points used in its interpolation.

Fig. 2 illustrates some 2-D vector-space splines.

Lie Groups: To efficiently work with Lie group splines, the *cumulative* interpolation formulation must be used [55]. The *cumulative blending matrix* $\widetilde{\mathbf{M}}_i$ is defined as a cumulative sum over columns of $\mathbf{M}_i$, $\widetilde{m}_{j,n} := \sum_{s=j}^{k-1} m_{s,n}$, and $\widetilde{\lambda}_{i,j}(t)$ are the coefficients of the *cumulative blending vector* $\widetilde{\boldsymbol{\lambda}}_i(t) = \widetilde{\mathbf{M}}_i \mathbf{u}_i(t) \in \mathbb{R}^k$. Then, using the composition operator $\circ$ (matrix multiplication for many groups), the Exp map, and the Log map defined by [29, Eqs. (1), (21), and (22)], the interpolation at t is given as follows:

$$\mathbf{x}_i(t) = \mathbf{x}_i \circ \mathbf{A}_{i,1}(t) \circ \mathbf{A}_{i,2}(t) \circ \dots \circ \mathbf{A}_{i,k-1}(t) \in \mathcal{L} \tag{11}$$

$$\text{with } \mathbf{A}_{i,j}(t) := \text{Exp}(\widetilde{\lambda}_{i,j}(t) \mathbf{d}_j^i) \in \mathcal{L} \tag{12}$$

$$\text{and } \mathbf{d}_j^i := \mathbf{x}_{i+j} \ominus \mathbf{x}_{i+j-1} = \text{Log}(\mathbf{x}_{i+j-1}^{-1} \circ \mathbf{x}_{i+j}) \in \mathbb{R}^d. \tag{13}$$

2) Spline Types: The computation of the as-yet-undefined blending matrix $\mathbf{M}_i$ depends on the spline type, defined by a selection of internal constraints that determine its continuity properties. The temporal *continuity* C^n of a spline refers to its degree-of-differentiability, where n is the number of derivatives that are continuous over the entire input domain of the spline. The ith derivative is a piecewise polynomial of degree $m - i$. Closed-form formulas have been derived for the first three temporal derivatives [55]. $\mathbf{M}_i$ also depends on the knots, which often have *uniform* spacing at some *knot interval*. Otherwise, the spline is *non-uniform*.

B-Splines: The most popular spline used in robotics applications is the *B-spline*, since it has the highest continuity possible; C^{k-2} (not C^{k-1}, e.g., in [19], [55], and [56]). If the B-spline is uniform, then each element at indices (s, n) of the blending matrix can be set using binomial coefficients [55] as follows:

$$m_{s,n} = \frac{\binom{k-1}{n}}{(k-1)!} \sum_{l=s}^{k-1} (-1)^{l-s} \binom{k}{l-s} (k-1-l)^{k-1-n}. \tag{14}$$

TABLE II
COMMON SPLINE TYPES AND THEIR PROPERTIES

Spline	Continuity	Interpolating	Example Applications	Additional Notes
Bézier	C^{k-4}/C^{k-3}	Every m-th	Vector Graphics, Fonts	C^{k-3} if control points are mirrored about knot points.
Hermite	C^{k-4}/C^{k-3}	All	Animation	Velocity set at control points. Special case of Bézier spline.
Kochanek–Bartels	C^{k-4}/C^{k-3}	All	Computer Graphics	Modification of Hermite Spline.
Cardinal	C^{k-3}	All	Animation	Special case of Kochanek–Bartels spline.
Catmull–Rom	C^{k-3}	All	Animation, Smoothing	Special case of Cardinal spline.
Linear	C^0	All	Fast Interpolation	Special case of spline with order k = 2.
B-Spline	C^{k-2}	None in general	Animation, Camera Paths	Minimal support w.r.t. order, smoothness, and domain.
NURBS	C^{k-2}	None in general	Computer-Aided Design (CAD)	Special case of B-spline.

The blending matrix of non-uniform splines depends on $2\mathrm{m}$ knots (not $2\mathrm{k}$, e.g., in [20]). It is the transpose of the *basis matrix* $\mathbf{B}_i$ from Haarbach et al. [48], originally Qin [54], and so it can be computed by transposing the result of Qin's recursive formula. Table II summarizes common spline types.

3) Optimization: In formulating a spline optimization problem, all priors and measurements must be included as *interpolated factors*. This means that interpolation (11) must be used in the residual computation to obtain the state or its temporal derivatives at the required sample times. Therefore, sufficient control points and knots are required to perform interpolation at all prior and measurement times. As a result, these factors in the optimization are associated with *i)* the control points of the spline segment(s) upon which interpolation will be performed, and *ii)* any necessary time-invariant variables. Jacobians for these factors, required for optimization, can be computed analytically [22], [55], [57], [58].

C. Temporal Gaussian Processes (TGPs)

In contrast to splines, which model the state parametrically as a weighted combination of control points, TGPs are nonparametric, using instead a Gaussian process (GP) as follows:

$$\mathbf{x}(t) \sim \mathrm{GP}(\check{\mathbf{x}}(t), \check{\mathbf{P}}(t, t')). \tag{15}$$

Here $\check{\mathbf{x}}(t)$ is the prior mean function and $\check{\mathbf{P}}(t, t')$ is the prior covariance function, or *kernel*. Interestingly, TGPs can be considered a weighted combination of infinitely many temporal basis functions (TBFs), made nonparametric by substituting the mean and covariance with functions (known as the *kernel trick*) [49], [59]. Indeed, despite different design choices, CT methods (based on temporal basis functions (TBFs)) are sometimes related. For example, posterior interpolation with a WNOA prior (see Section III-C2) yields cubic Hermite spline interpolation [24]. As in DT estimation, the estimated process variables $\mathbf{x}$ represent the state at specific times. Similarly to spline knots, the times of these *support states* may be chosen freely (uniformly or non-uniformly).

This method requires the evolution between these support states to be modeled as a LTV-SDE process as introduced in Section II. While this imposes a Markovian restriction on the CT state, augmenting the base state with derivatives to overcome this is usually straightforward, shown later in Section III-C2. Since many systems of interest in robotics can be modeled as LTV-SDEs, this method is quite general.

It is assumed that the process is governed by CT white noise as follows:

$$\mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_C \cdot \delta(t - t')) \tag{16}$$

where $\mathbf{Q}_C$ is the (symmetric, positive-definite) *power-spectral density matrix* and $\delta(\cdot)$ is the *Dirac delta* function. The mean function $\check{\mathbf{x}}(t) = \mathbb{E}[\mathbf{x}(t)]$ can be derived from (5) as follows:

$$\check{\mathbf{x}}(t) = \mathbf{\Phi}(t, t')\check{\mathbf{x}}(t') + \mathbf{v}(t, t') \tag{17}$$

$$\text{with } \mathbf{v}(t, t') := \int_{t'}^{t} \mathbf{\Phi}(t, s)\mathbf{v}(s) ds. \tag{18}$$

From this it is easy to formulate a residual $\mathbf{e}_i$ between two explicitly estimated states at times t_i and t_{i+1} as follows:

$$\mathbf{e}_i = \mathbf{\Phi}(t_{i+1}, t_i)\mathbf{x}(t_i) + \mathbf{v}(t_{i+1}, t_i) - \mathbf{x}(t_{i+1}). \tag{19}$$

The covariance function $\check{\mathbf{P}}(t, t') = \mathrm{Var}[\mathbf{x}(t)]$ can also be derived from (5) (cf. [24], [34], [35], [60]) as follows:

$$\check{\mathbf{P}}(t, t') = \mathbf{\Phi}(t, t_0)\check{\mathbf{P}}_0\mathbf{\Phi}(t', t_0)^T + \begin{cases} \mathbf{\Phi}(t, t')\mathbf{Q}(t', t_0) & t' < t \\ \mathbf{Q}(t, t_0) & t' = t \\ \mathbf{Q}(t, t_0)\mathbf{\Phi}(t', t)^T & t' > t \end{cases} \tag{20}$$

$$\text{with } \mathbf{Q}(t, t') := \int_{t'}^{t} \mathbf{\Phi}(t, s)\mathbf{L}(s)\mathbf{Q}_C\mathbf{L}(s)^T\mathbf{\Phi}(t, s)^T ds. \tag{21}$$

Together, the residual $\mathbf{e}_i$ with corresponding prior covariance $\mathbf{Q}(t_{i+1}, t_i)$ are used as a binary factor between adjacent states, commonly referred to as the *GP (motion) prior factor*.

1) Interpolation. Vector Space:: The key insight is that the process model has been used to define the prior mean and covariance functions of the GP. Because the process model is Markovian, the inverse covariance is *exactly sparse* (block-tridiagonal), and by exploiting this sparsity [34], [35], the interpolated posterior mean and covariance can be derived[5] for $t \in [t_i, t_{i+1})$ as follows:

$$\mathbf{x}(t) = \mathbf{\Lambda}(t)\mathbf{x}(t_i) + \mathbf{\Psi}(t)\mathbf{x}(t_{i+1}) + \mathbf{v}(t, t_i) - \mathbf{\Psi}(t)\mathbf{v}(t_{i+1}, t_i) \tag{22}$$

$$\mathbf{P}(t, t) = \begin{bmatrix} \mathbf{\Lambda}(t) & \mathbf{\Psi}(t) \end{bmatrix} \begin{bmatrix} \mathbf{P}(t_i, t_i) & \mathbf{P}(t_i, t_{i+1}) \\ \mathbf{P}(t_{i+1}, t_i) & \mathbf{P}(t_{i+1}, t_{i+1}) \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda}(t)^T \\ \mathbf{\Psi}(t)^T \end{bmatrix} \tag{23}$$

[5]The GP interpolation equations can also be derived from the perspective of optimal control theory [22, Appendix A].

$$+ \mathbf{Q}(t,t_i) - \boldsymbol{\Psi}(t)\mathbf{Q}(t_{i+1},t_i)\boldsymbol{\Psi}(t)^T$$

$$\text{with } \boldsymbol{\Psi}(t) := \mathbf{Q}(t,t_i)\boldsymbol{\Phi}(t_{i+1},t)^T\mathbf{Q}(t_{i+1},t_i)^{-1} \tag{24}$$

$$: \text{and } \boldsymbol{\Lambda}(t) := \boldsymbol{\Phi}(t,t_i) - \boldsymbol{\Psi}(t)\boldsymbol{\Phi}(t_{i+1},t_i). \tag{25}$$

Importantly, interpolation relies only on the two temporally adjacent states. If $\mathbf{v}(t) = \mathbf{0} \forall t \in [t_i, t_{i+1})$ as in commonly used models, then (22) further simplifies to

$$\mathbf{x}(t) = \boldsymbol{\Lambda}(t)\mathbf{x}(t_i) + \boldsymbol{\Psi}(t)\mathbf{x}(t_{i+1}). \tag{26}$$

Lie Groups: To work with Lie group states, which would yield a nonlinear SDE process in almost all nontrivial cases, the method of *local* LTV-SDEs is used [61], [62], [63], which utilizes the local tangent space of the manifold. Let $\mathbf{x}(t) = \{\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \ddot{\boldsymbol{x}}(t), \dots\}$ be the global state, where $\boldsymbol{x}(t) \in \mathcal{L}$ is a Lie group element (e.g., pose), and $\dot{\boldsymbol{x}}(t) \in \mathbb{R}^d, \ddot{\boldsymbol{x}}(t) \in \mathbb{R}^d$ are its first and second temporal derivatives. Between every pair of explicitly estimated states, $\mathbf{x}_i := \mathbf{x}(t_i)$ and $\mathbf{x}_{i+1} := \mathbf{x}(t_{i+1})$, the NTV-SDE is approximated as an LTV-SDE for $t \in [t_i, t_{i+1})$ through the use of a local mapping as follows:

$$\boldsymbol{\Xi}(t) := \begin{bmatrix} \boldsymbol{\xi}(t) \\ \dot{\boldsymbol{\xi}}(t) \\ \ddot{\boldsymbol{\xi}}(t) \\ \vdots \end{bmatrix} := \begin{bmatrix} \boldsymbol{x}(t) \ominus \boldsymbol{x}(t_i) \\ \mathbf{J}_r(\boldsymbol{\xi}(t))^{-1}\dot{\boldsymbol{x}}(t) \\ \frac{d}{dt}\left(\mathbf{J}_r(\boldsymbol{\xi}(t))^{-1}\right)\dot{\boldsymbol{x}}(t) + \mathbf{J}_r(\boldsymbol{\xi}(t))^{-1}\ddot{\boldsymbol{x}}(t) \\ \vdots \end{bmatrix} \tag{27}$$

where $\mathbf{J}_r(\boldsymbol{\xi}(t))$ is the right Jacobian of the manifold as defined by [29, Eq. (67)].[6] The local-to-global mapping is as follows:

$$\boldsymbol{x}(t) = \boldsymbol{x}(t_i) \oplus \boldsymbol{\xi}(t) \in \mathcal{L} \tag{28}$$

$$\begin{bmatrix} \dot{\boldsymbol{x}}(t) \\ \ddot{\boldsymbol{x}}(t) \\ \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{J}_r(\boldsymbol{\xi}(t))\dot{\boldsymbol{\xi}}(t) \\ \mathbf{J}_r(\boldsymbol{\xi}(t))\left(\ddot{\boldsymbol{\xi}}(t) - \frac{d}{dt}\left(\mathbf{J}_r(\boldsymbol{\xi}(t))^{-1}\right)\dot{\boldsymbol{x}}(t)\right) \\ \vdots \end{bmatrix}. \tag{29}$$

Therefore, one stores and optimizes the global state variables $\mathbf{x}$ but uses the local LTV-SDE for interpolation. Interpolating the state mean at $t \in [t_i, t_{i+1})$ is then a three-step process as follows:

1) Map explicitly estimated global states $\mathbf{x}(t_i), \mathbf{x}(t_{i+1})$ to local states $\boldsymbol{\Xi}(t_i), \boldsymbol{\Xi}(t_{i+1})$ using (27);
2) Interpolate local state $\boldsymbol{\Xi}(t)$ using (22) or (26);
3) Map the interpolated local state $\boldsymbol{\Xi}(t)$ to a global state $\mathbf{x}(t)$ using (28) and (29).

A similar procedure is available for the covariance [36].

2) GP Priors: The white-noise-on-acceleration (WNOA) or *"constant-velocity"* prior is a generic process model and was the first to be used for GP-based CT state estimation [59]. Other priors are discussed in Section IV-C. It assumes the second derivative ("acceleration") is driven by white noise (16). For vector-valued $\boldsymbol{x}(t)$, this is written as follows:

$$\ddot{\boldsymbol{x}}(t) = \mathbf{w}(t) \in \mathbb{R}^d. \tag{30}$$

[6] Note that $\frac{d}{dt}(\mathbf{J}_r(\boldsymbol{\xi}(t))^{-1})$ does not have a general closed form, but can be approximated [64]. A closed-form solution exists for $SO(3)$ [65].

A first-order SDE (satisfying the Markovian condition) is formulated by including the first derivative in the state as follows:

$$\mathbf{x}(t) = \begin{bmatrix} \boldsymbol{x}(t) \\ \dot{\boldsymbol{x}}(t) \end{bmatrix} \in \mathbb{R}^{2d}, \text{ and } \dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\boldsymbol{x}}(t) \\ \ddot{\boldsymbol{x}}(t) \end{bmatrix} \in \mathbb{R}^{2d}. \tag{31}$$

This yields a linear time-invariant (LTI) SDE with

$$\mathbf{F}(t) = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \ \mathbf{v}(t) = \mathbf{0}, \text{ and } \mathbf{L}(t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix}. \tag{32}$$

The transition, covariance, and precision matrices for the LTI-SDE are calculated in closed form (with $\Delta t := t - t'$) as follows:

$$\boldsymbol{\Phi}(t,t') = \begin{bmatrix} \mathbf{I} & \Delta t\mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \in \mathbb{R}^{2d\times 2d} \tag{33}$$

$$\mathbf{Q}(t,t') = \begin{bmatrix} \frac{1}{3}\Delta t^3\mathbf{Q}_C & \frac{1}{2}\Delta t^2\mathbf{Q}_C \\ \frac{1}{2}\Delta t^2\mathbf{Q}_C & \Delta t\mathbf{Q}_C \end{bmatrix} \in \mathbb{R}^{2d\times 2d}, \text{ and} \tag{34}$$

$$\mathbf{Q}(t,t')^{-1} = \begin{bmatrix} 12\Delta t^{-3}\mathbf{Q}_C^{-1} & -6\Delta t^{-2}\mathbf{Q}_C^{-1} \\ -6\Delta t^{-2}\mathbf{Q}_C^{-1} & 4\Delta t^{-1}\mathbf{Q}_C^{-1} \end{bmatrix} \in \mathbb{R}^{2d\times 2d}. \tag{35}$$

Since $\mathbf{v}(t) = \mathbf{0}\forall t$, the computation of the GP prior residual is simplified, and (26) can be used.

If $\boldsymbol{x}(t)$ is a Lie group object, then one must have the global Markov state $\mathbf{x}(t) = \{\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t)\} \in \mathcal{L} \times \mathbb{R}^d$. The local Markov state defined at $t \in [t_i, t_{i+1})$ and its temporal derivative are as follows:

$$\boldsymbol{\Xi}(t) = \begin{bmatrix} \boldsymbol{\xi}(t) \\ \dot{\boldsymbol{\xi}}(t) \end{bmatrix} \in \mathbb{R}^{2d}, \text{ and } \dot{\boldsymbol{\Xi}}(t) = \begin{bmatrix} \dot{\boldsymbol{\xi}}(t) \\ \ddot{\boldsymbol{\xi}}(t) \end{bmatrix} \in \mathbb{R}^{2d}. \tag{36}$$

The local white-noise-on-acceleration (WNOA) prior thus assumes that acceleration *in the local frame* is driven by white noise, as $\ddot{\boldsymbol{\xi}}(t) = \mathbf{w}(t) \in \mathbb{R}^d$. Substituting into (19), the WNOA prior residual (with $\Delta t_{i+1} := t_{i+1} - t_i$) can be written as follows:

$$\mathbf{e}_i = \begin{bmatrix} \Delta t_{i+1}\dot{\boldsymbol{x}}(t_i) - \boldsymbol{x}_{i+1} \ominus \boldsymbol{x}_i \\ \dot{\boldsymbol{x}}(t_i) - \mathbf{J}_r(\boldsymbol{x}_{i+1} \ominus \boldsymbol{x}_i)^{-1}\dot{\boldsymbol{x}}(t_{i+1}) \end{bmatrix}. \tag{37}$$

IV. SURVEY

A. Interpolation and Integration

An overview of the considered works that use interpolation and integration in a CT formulation is provided in Table III.

1) Linear interpolation (LI) in LiDAR and RADAR Systems: Among the earliest works that formulate LI as a CT method are the LiDAR SLAM system by Bosse and Zlot [66] and Bosse et al. [67], which applied the constant-velocity assumption over short time intervals for pose interpolation in their surfel-based LiDAR and LiDAR-inertial simultaneous localization and mapping (SLAM) systems, respectively. Due to its speed, many works have adopted LI for point cloud motion distortion correction (MDC) in LiDAR odometry [4], [5], [68], [69], [70], [71],

TABLE III
LINEAR INTERPOLATION (LI) AND NUMERICAL INTEGRATION CT STATE REPRESENTATION WORKS

Authors	**Year**	**Applications**	**Time Invariant Estimates** State(s)	**Time-Varying Estimates** State(s)	 State Representation(s)
Ait-Aider and Berry [93]	2009	RS Object Tracking	$\mathbf{L}$	$\dot{\mathbf{r}}, \dot{\mathbf{p}}$	Linear Interpolation
Bosse and Zlot [66]	2009	LiDAR SLAM	$\mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation (& Cubic Spline)
Klein and Murray [90]	2009	RS SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3)$	Riemann Summation
Forssén and Ringaby [94]	2010	RSC	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Hong et al. [68]	2010	PC MDC for PC Registration	-	$\mathbf{T} \in SE(3)$	Linear Interpolation
Moosmann and Stiller [69]	2011	PC MDC for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Ringaby and Forssén [95]	2011	RSC	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Karpenko et al. [99]	2011	RSC for Video Stabilization	$\mathbf{K}$	$\mathbf{r} \in SO(3)$	Linear Interpolation
Hedborg et al. [97]	2012	RSC for SfM	$\mathbf{L}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Bosse et al. [67]	2012	LiDAR-Inertial SLAM	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{t}_E, \mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Jia and Evans [96]	2012	RSC for Video Rectification	-	$\mathbf{r} \in SO(3)$	Linear Interpolation
Anderson and Barfoot [92]	2013	LO	-	$\dot{\mathbf{T}}$ $\mathbf{T} \in SE(3)$	Feature Tracking + RANSAC Riemann Summation
Vivet et al. [82]	2013	RADAR MDC for RADAR Odometry Linear Velocity Estimation	-	${}^2\dot{\mathbf{T}}$	Linear Interpolation TBF
Dong and Barfoot [70]	2013	PC MDC for PC Registration for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Li et al. [105]	2013	VIO	-	$\mathbf{r} \in SO(3)$ $\mathbf{p}, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Euler Method (RK) Trapezoidal Rule (RK) Discrete-Time Values
Guo et al. [98]	2014	RS Visual-Inertial SLAM	$\mathbf{L}$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation Discrete-Time Values
Zhang and Singh [4, 5]	2014/2017	PC MDC for PC Registration for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Ceriani et al. [80]	2015	LiDAR SLAM	$\mathcal{M}$	$\mathbf{T} \in SE(3)$	Linear Interpolation
Dubé et al. [104]	2016	LiDAR-Inertial-Wheeled SLAM	$\mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Splines
Droeschel et al. [79]	2017	PC MDC for LiDAR SLAM	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Deschaud [71]	2018	PC MDC for LiDAR SLAM	-	$\mathbf{T} \in SE(3)$	Linear Interpolation
Wang et al. [106]	2018	Stereo VO	-	$\ddot{\mathbf{r}}, \dot{\mathbf{r}}, \ddot{\mathbf{p}}$	Constant-Acceleration Integration
Lowe et al. [107]	2018	LiDAR-Visual-Inertial SLAM	$\mathbf{S_3}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Uniform Linear Spline
Park et al. [86]	2018	LiDAR-Inertial SLAM	$\mathbf{b}_g, \mathbf{b}_a, \mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
[100]	2018	multiSensor Odometry and Localization	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Eckenhoff et al. [101]	2019	MultiCamera VIO and Calibration	$\mathbf{T}_E \in (SU(2) \times \mathbb{R}^3)^n, \mathbf{t}_E, \mathbf{K}^n, \boldsymbol{d}_C^n$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation Discrete-Time Values
Ye et al. [74]	2019	PC MDC for LO	-	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation Discrete-Time Values
Lin and Zhang [72]	2020	PC MDC for PC Registration for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Shan et al. [75]	2020	PC MDC for LiDAR-Inertial SLAM	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Park et al. [89]	2020	LiDAR-Visual Calibration	$\mathbf{T}_E, \mathbf{t}_E$	$\mathbf{T} \in SE(3)$	Linear Interpolation
Park et al. [87]	2021	LiDAR-Visual-Inertial SLAM	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{t}_E, \mathcal{M}_S$	$\mathbf{T} \in SE(3)$ / $SO(3) \times \mathbb{R}^3$	Linear Interpolation (& Uniform Cubic B-Spline)
Burnett et al. [83]	2021	Spinning RADAR MDC	-	${}^2\dot{\mathbf{T}}$	Linear Interpolation
Li et al. [76]	2021	PC MDC for LIO	-	$\mathbf{p}$ $\mathbf{r} \in SU(2), \mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation Discrete-Time Values
Wang and Ma [108]	2021	Visual-Inertial-LiDAR Extrinsic Calibration	$\mathbf{T}_E$	$\mathbf{r} \in SU(2), \dot{\mathbf{p}}$ $\mathbf{p}$ $\mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation Constant-Acceleration Integration Discrete-Time Values
Shan et al. [78]	2021	PC MDC for LiDAR-Visual-Inertial SLAM	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Xu and Zhang [81]	2021	LIO & PC MDC	$\mathbf{g}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Forward/Backward Riemann Summation Discrete-Time Values
Xu et al. [109]	2022	LIO	$\mathbf{g}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Constant Angular Velocity & Acceleration Integration Discrete-Time Values
Wang and Ma [110]	2022	PC MDC for LiDAR-Visual-Inertial SLAM	-	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$	Linear Interpolation
Ramezani et al. [88]	2022	LiDAR-Inertial SLAM	$\mathbf{b}_g, \mathbf{b}_a, \mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation (& Cubic B-Spline)
Dellenbach et al. [84]	2022	PC MDC for PC Registration for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
He et al. [111]	2023	LIO	$\mathbf{g}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{p}}$ $\mathbf{b}_g, \mathbf{b}_a$	Constant Angular Velocity & Acceleration Integration Discrete-Time Values
Vizzo et al. [73]	2023	PC MDC for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Linear Interpolation
Nguyen et al. [77]	2023	PC MDC for LiDAR-Inertial SLAM	-	$\mathbf{r} \in SO(3), \mathbf{p}, \dot{\mathbf{p}}, \mathbf{b}_g, \mathbf{b}_a$	Linear Interpolation
Zheng and Zhu [85]	2023	LO	-	$\mathbf{T} \in SE(3)$	Linear Interpolation
Chen et al. [112]	2023	PC MDC for LIO	-	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, \dot{\mathbf{p}}, \mathbf{b}_g, \mathbf{b}_a$	Constant-Jerk Integration
Zheng and Zhu [30]	2024	LO	-	$\mathbf{T} \in SE(3)$	Linear Interpolation

See Table VII for definitions.

[72], [73], LiDAR-inertial odometry [74], [75], [76], LiDAR-inertial SLAM [77], LiDAR-visual-inertial odometry [78], and inertial-wheeled odometry [79]. Ceriani et al. [80] modeled an $SE(3)$ trajectory with LI in their LiDAR SLAM system, repeatedly correcting for motion distortion in its scan-to-map ICP-based optimizer. FAST-LIO [81] performs a Riemann-style summation of IMU measurements in its LiDAR-inertial odometry (LIO) system, both forward, for state propagation in the Kalman filter (KF), and backward, for point cloud MDC. LI has also been applied to MDC of spinning RADAR [82], [83]. The LiDAr odometry (LO) systems CT-ICP [84] and ECTLO [85] embrace the CT formulation for LiDAR MDC within the residuals of iterative optimization, as does the LiDAR-inertial SLAM system by Park et al. [86] in its surfel and inertial residuals. Later surfel-based SLAM works by the same authors [87], [88] maintain LI as the primary representation for fast inference, but use cubic B-splines to obtain pose corrections. Park et al. [89] applied LI over $SE(3)$ poses to the spatial-temporal calibration of LiDAR-camera systems. Recently, Zheng and Zhu [30] formulated point-to-plane residuals for every point of a scanning LiDAR in their LO system, using LI to estimate the $SE(3)$ pose at the acquisition time of each point.

2) Linear interpolation (LI) in Camera Systems: Many works have also applied LI to camera systems, especially in rolling shutter compensation (RSC). An early example was by Klein and Murray [90], who applied LI to perform RSC within PTAM [91]. The constant velocity was determined using the feature correspondences, and a similar idea was adopted by the RANSAC approach of Anderson and Barfoot [92]. RSC was very common in early CT works, through LI [93], [94], [95], [96], [97], [98] or Riemannian integration of gyroscope measurements [99]. LI has also been used to efficiently infer poses for multi-sensor fusion [100], as well as in multi-camera odometry and calibration systems [100], [101].

3) Arguments for Linear interpolation (LI): A common technique present in many CT LI works [66], [67], [87], [88], [102], [103], [104] is the separation of a base trajectory of discrete poses and a correction trajectory during estimation. The key idea is that when using LI, corrections can be adequately represented

TABLE IV
GAUSSIAN PROCESS IMU MEASUREMENT PREINTEGRATION WORKS

Authors	**Year**	**Contributions**
Gentıl et al. [13]	2018	UPMs
Gentil et al. [14]	2020	GPMs
Gentil et al. [127]	2020	Event-Inertial SLAM with GPMs
Gentil and Vidal-Calleja [128]	2021	$SO(3)$ UGPMs LPMs
Dai et al. [129]	2022	Event-Inertial Odometry and Mapping with LPMs
Gentil and Vidal-Calleja [130]	2023	$SE(3)$ UGPMs Optional Kernel Hyperparameter Tuning
Gentil et al. [131]	2024	LPMs for LiDAR MDC & Dyn. Obj. Detection

at a lower frequency than the poses themselves [87], [104], with few coefficients to be optimized. Furthermore, LI is simple to implement with low-computational cost.

4) Nonlinear Interpolation: Not all nonlinear interpolation strategies are generalized by the temporal splines or TGPs. In the pose interpolation method for SLAM proposed by Terzakis and Lourakis [113], they regressed a quadratic rational function to the previous five or more poses with equality constraints to enforce interpolation. Rational functions take the form of an algebraic fraction where both numerator and denominator are polynomials (i.e., the quotient of quadratic TBFs in this case). Zhu and Wu [114] represented the time-varying state by a *Chebyshev polynomial*, where the temporal base coefficients are regressed through the *collocation method* for batch and sliding-window MAP estimation. They later apply the method to attitude estimation [115]. Given known dynamics, Agrawal and Dellaert [116] used a Chebyshev polynomial basis to represent the CT state and control input, leveraging pseudospectral control theory for estimation.

5) Numerical Integration: Integrating the derivative of a state until a time of interest is sometimes interpreted as CT state estimation. This includes Riemannian summation, which follows the same constant rate assumption as LI. In many IMU integration [81], [105], [117] and IMU preintegration methods [10], [11], [12], [118], [119], analytic equations are derived for the propagation of rotation under a constant angular velocity model, and position under a constant acceleration model. Occasionally, other models such as constant velocity [81] or constant jerk [112] are assumed for inertial position integration. FAST-LIO2 [109] and Point-LIO [111] are two notable Kalman filters with continuous kinematic models achieving state-of-the-art performance. Many other works have employed IMU integration (e.g., in EKFs [117], [120], [121] and UKFs [122], [123]), using a wide variety of numerical integration techniques (e.g., Euler method, trapezoidal rule for ODEs [105], and fourth-order RK [120], [122]). However, they do not use interpolation to infer intermediate states in a CT sense for any purpose. Wang et al. [106] derived discretized integration equations from assumptions of constant linear and angular acceleration over short time segments. Treating these as a CT representation of a camera pose, they formulate reprojection residuals and optimize for the constant rate parameters of these segments, as well as an "abrupt force" parameter to determine when new segments should be created.

6) Gaussian process IMU Measurement Preintegration: In contrast to the other discussed methods, one collection of works, summarized in Table IV, models inertial measurements from an IMU with temporal GPs to form preintegrated measurements. Gentil et al. [13] fitted GPs to the measurements and used them to upsample the IMU measurements to a very high frequency. Assuming constant acceleration, upsampled preintegrated measurements (UPMs) are computed from these GP and a first-order method is provided to correct IMU biases and intersensor time offsets after integration. They applied UPMs to LiDAR-IMU calibration, and later to LiDAR-inertial SLAM [124]. Subsequently [14], they presented Gaussian preintegrated measurements (GPMs), an analytical method for position, velocity, and single-axis rotation estimation based on linear operators applied to GP kernels [125] that does not suffer from numerical integration error. They also derive postintegration biases and time offset corrections for GPMs. They applied UPMs and GPMs to their SLAM system IN2LAAMA [126] and to line-based event-inertial SLAM [127]. In follow-up work, Gentil and Vidal-Calleja [128] proposed unified Gaussian preintegrated measurements (UGPMs) to extend GPMs to $SO(3)$ rotations, and linear preintegrated measurements (LPMs) which use LI for computational speed. Dai et al. [129] proposed a corner-based event-inertial odometry and mapping system using LPMs for pose inference at the event times. Most recently, Gentil and Vidal-Calleja [130] formulated UGPMs for $SE(3)$ poses, fusing the rotation and acceleration computation steps. They proposed initializing UGPMs with LPMs, with an optional kernel hyperparameter tuning step. They demonstrated their efficacy in a wide variety of multi-sensor estimation problems and later use LPMs for LiDAR MDC and dynamic object detection [131]. These GP-based IMU preintegration methods have been shown to be more accurate than other methods [10], [11], especially during fast motion. The cubic complexity of GP regression in these methods might be overcome with exactly sparse kernels (see Section III-C) in future work.

B. Temporal Splines

A chronologically ordered overview of all surveyed methods using temporal splines is given in Table V.

1) Origins and Early Formulations: Splines have received significant attention in robotics as a CT representation of the state. They were first formulated as piecewise-polynomial by Schoenberg in 1946 [188], inspired by "lofting" where thin wooden strips (called splines) were used for aircraft and ship design, including for construction from templates during World War II [189]. While various spline types have been explored in different robotics applications, B-splines, short for *basis* splines, are most popular due to their smoothness properties. Their mathematical foundations were pioneered by Schoenberg, de Boor, and others, with the de Boor–Cox recurrence formula [51], [52] a famous early formulation. From this, Qin [54] was the first to derive a general matrix representation for B-splines. Later, splines were presented as a special case of TBFs where the bases are polynomials [20], [141]. As an alternative to polynomials, wavelet basis functions [50], [134] have been proposed in a hierarchical scheme to account for the varying richness of motion.

TABLE V
SPLINE AND TEMPORAL BASIS FUNCTION (TBF) CT STATE REPRESENTATION WORKS

Authors	Year	Applications	Time-Invariant Estimates	Time-Varying Estimates	
			State(s)	State(s)	State Representation(s)
Kim et al. [132, 133]*	1995	Torque Computation, Animation	-	$\mathbf{r} \in SU(2)$	Bézier, Hermite & B-Spline
Gortler and Cohen [134]*	1995	Theory	-	$\mathbb{R}^n$	Polynomial and Wavelet TBFs
Anderson-Sprecher and Lenth [135]	1996	Bearing-Based Object Tracking	-	${}^2\mathbf{p}$	Cubic B-Spline
Qin [54]*	1998	Theory	-	$\mathbb{R}^n$	B-Spline
Kang and Park [136]	1999	Theory	-	$\mathbf{r}$	Cubic Splines
Crouch et al. [53]	1999	Theory	-	$\mathcal{L}$	Cubic Splines
Jung and Taylor [137]	2001	VIO	- -	$\mathbf{r} \in SO(3)$ $\mathbf{p}$	Riemann Summation Spline
Bibby and Reid [138]	2010	Dynamic SLAM	${}^2\mathcal{M}_O$	${}^2\mathbf{T} \in \mathbb{R}^3, ({}^2\mathbf{T}_D \in \mathbb{R}^3)^D$	Non-Uniform Cubic Splines
Hadzagic and Michalska [139]	2011	Object Tracking	-	${}^2\mathbf{p}$	Non-Uniform Cubic B-Spline
Fleps et al. [140]	2011	Visual-Inertial Calibration	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{T}_E, \mathbf{g}, \dot{\mathbf{p}}_{\text{scale}}$	$\mathbf{p} \in \mathbb{R}^3, \mathbf{r} \in \mathbb{R}^4$	Cubic B-Splines
Furgale et al. [20, 141]*	2012/2015	Theory Visual-Inertial Calibration RS Localization	- $\mathbf{T}_E, \mathbf{g}, \mathbf{L}$ -	$\mathbb{R}^n$ $\mathbf{r} \in \mathbb{R}^3$(CGR), $\mathbf{p}, \mathbf{b}_g, \mathbf{b}_a$ $\mathbf{r} \in \mathbb{R}^3$(CGR), $\mathbf{p}$	TBFs B-Splines B-Splines
Furgale et al. [18]	2013	Visual-Inertial Calibration	$\mathbf{T}_E, \mathbf{t}_E, \mathbf{g}$	$\mathbf{T} \in \mathbb{R}^6$(CGR) $\mathbf{b}_g, \mathbf{b}_a$	TBFs / Quintic B-Spline TBFs / Cubic B-Splines
Lovegrove et al. [142]*	2013	RS SLAM RS-Inertial SLAM	$\boldsymbol{\rho}_d$ $\mathbf{b}_g, \mathbf{b}_a, \mathbf{K}, \mathbf{T}_E, \mathbf{g}, \boldsymbol{\rho}_d$	$\mathbf{T} \in SE(3)$	Cubic B-Spline
Anderson et al. [143, 144]	2013/2015	LiDAR SLAM	$\mathbf{L}$	$\dot{\mathbf{T}}$ $\mathbf{T} \in SE(3)$	TBFs / Cubic B-Spline Riemann Summation
Oth et al. [145]	2013	RS Calibration	t_{RS}	$\mathbf{r} \in \mathbb{R}^3, \mathbf{p} \in \mathbb{R}^3$	TBFs / Uniform Cubic B-Spline
Sheehan et al. [146]	2013	LiDAR Localization	-	$\mathbf{T} \in \mathbb{R}^6$(RPY)	Cubic Catmull-Rom Spline
Zlot and Bosse [102, 103]	2013/2014	LiDAR-Inertial Odometry	$\mathcal{M}_S$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Uniform Linear / Cubic B-Spline
Rehder et al. [147]	2014	LiDAR-Visual-Inertial Calibration	$\mathbf{T}_E, \mathbf{T}_E, \mathbf{g}$	$\mathbf{T} \in \mathbb{R}^6$(CGR), $\mathbf{b}_g, \mathbf{b}_a$	TBFs / Quintic B-Splines
Alismail et al. [148]	2014	PC MDC for PC Registration for LO	-	$\mathbf{T} \in \mathbb{R}^6$(CGR)	Uniform Linear to Quintic B-Spline
Anderson et al. [50]	2014	TE	-	$\mathbf{T} \in \mathbb{R}^6$	Polynomial and Wavelet TBFs
Patron-Perez et al. [149]	2015	RS SLAM RS-Inertial SLAM	$\boldsymbol{\rho}_d$ $\mathbf{b}_g, \mathbf{b}_a, \mathbf{K}, \mathbf{T}_E, \mathbf{g}, \boldsymbol{\rho}_d$	$\mathbf{T} \in SE(3)$	Cubic B-Spline
Kerl et al. [150]	2015	RS RGB-D Odometry	-	$\mathbf{T} \in SE(3)$	Uniform Cubic B-Spline
Mueggler et al. [151]	2015	Event Camera Localization	-	$\mathbf{T} \in SE(3)$	Uniform Cubic B-Spline
Rehder et al. [152]	2016	Extrinsic Calibration Theory LiDAR-Visual-Inertial Calibration	$\mathbf{t}_E$ $b_r, \mathbf{T}_E, \mathbf{t}_E, \mathbf{g}, \mathbf{L}_\pi$	- $\mathbf{T} \in \mathbb{R}^6$(AA) $\mathbf{b}_g, \mathbf{b}_a$	TBFs Quintic B-Splines Cubic B-Splines
Rehder et al. [153]	2016	Multi-IMU Visual-Inertial Calibration	$\mathbf{S}_g, \mathbf{M}_g, \mathbf{G}_g, \mathbf{S}_a,$ $\mathbf{M}_a, \mathbf{T}_E, \mathbf{t}_E, \mathbf{g}$	$(\mathbf{T} \in \mathbb{R}^6\text{(AA)})^N$ $(\mathbf{b}_g \mathbf{b}_a)^N$	Quintic B-Splines Cubic B-Splines
Kim et al. [154]	2016	RS SLAM	$\mathbf{p}_d$	$\mathbf{T} \in SE(3)$	Cubic B-Spline
Sommer et al. [155]*	2016	Theory Attitude Estimation	-	$\mathcal{L}$ $\mathbf{r} \in SU(2)$	B-Spline
Li et al. [156, 157, 158]	2016-2018	Bearing-based Tracking	-	${}^2\mathbf{p}$	Polynomial and Sinusoidal TBFs
Rehder and Siegwart [159]	2017	Visual-Inertial Calibration	$t_e, \boldsymbol{r}_C, \mathbf{S}_g, \mathbf{M}_g, \mathbf{G}_g, \mathbf{S}_a, \mathbf{M}_a,$ $\mathbf{p}_{ia}, \mathbf{r}_{ga}, \mathbf{T}_E, \mathbf{t}_E, \mathbf{g}, \mathbf{m}_o$	$\mathbf{T} \in \mathbb{R}^6$(AA) $\mathbf{b}_g, \mathbf{b}_a$	Uniform Quintic B-Spline Uniform Cubic B-Spline
Vandeportaele et al. [160]	2017	RS Localization	-	$\mathbf{T} \in SE(3)$	Uniform Cubic B-Spline
Droeschel and Behnke [161]	2018	PC MDC, LiDAR SLAM	$\mathcal{M}_S$	$\mathbf{T} \in SE(3)$	Cubic B-Spline
Mueggler et al. [162]	2018	Event-Inertial Localization	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{r}_{\mathbf{g}} \in \mathbb{R}^2$	$\mathbf{T} \in SE(3)$	Uniform Cubic B-Spline
Ovrén and Forssén [163]	2018	RS Visual-Inertial SLAM	$\boldsymbol{\rho}_d$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$	Cubic B-Splines
Ovrén and Forssén [164]	2019	RS Visual-Inertial SLAM	$\boldsymbol{\rho}_d$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, SE(3)$	Cubic B-Spline(s)
Sommer et al. [55]*	2020	Theory TE Visual-Inertial Calibration	- - $\mathbf{b}_g, \mathbf{b}_a, \mathbf{T}_E, \mathbf{g}$	$\mathcal{L}$ $\mathbf{T} \in SO(3) \times \mathbb{R}^3, SE(3)$ $\mathbf{T} \in SO(3) \times \mathbb{R}^3, SE(3)$	B-Spline Cubic to Quintic B-Spline(s) Quartic Uniform B-Spline(s)
Pacholska et al. [165]	2020	Certifiable Range-Based Localization	-	${}^n\mathbf{p}$	Polynomial and Sinusoidal TBFs
Lv et al. [166]	2020	LiDAR-Inertial Calibration	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{T}_E$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Uniform Cubic B-Spline
Hug and Chli [31]	2020	Visual-Inertial SLAM	$\mathbf{b}_g, \mathbf{b}_a, \mathbf{g}, \mathbf{L}$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$	Cubic B-Spline(s)
Lv et al. [167]	2021	PC MDC LIO	- $\mathbf{b}_g, \mathbf{b}_a$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{p}}$ & $\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Riemann Summation & Uniform B-Splines Uniform B-Splines
Ng et al. [168]	2021	Multi-RADAR Inertial Odometry	-	$\mathbf{T} \in SE(3)$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Spline Discrete-Time Values
Yang et al. [169]	2021	Multi-Camera SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3)$	Non-Uniform Cubic B-Spline
Quenzel and Behnke [56]	2021	Surfel Registration for LO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Non-Uniform Quadratic B-Splines
Huang et al. [170]	2021	VO	$\mathbf{L}$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$ $\mathbf{T} \in \mathbb{R}^4$ (position & roll)	Cubic B-Spline
Persson et al. [47]*	2021	VO Visual-Inertial Calibration	- $\mathbf{T}_E$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$	Cubic to Quintic B-Spline
Cioffi et al. [19]	2022	Visual-Inertial-GNSS SLAM	$\mathbf{T}_E, \mathbf{t}_E, \mathbf{g}, \mathbf{L}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \mathbf{b}_g, \mathbf{b}_a$	Cubic to Sextic Uniform B-Splines
Hug et al. [171]	2022	Stereo Visual-Inertial SLAM	$\mathbf{g}, \mathbf{L}$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, \mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines
Tirado and Civera [57]*	2022	Object Tracking	-	$(\mathbf{T}_D \in SE(3))^D$	Non-Uniform Cubic B-Splines
Mo and Sattar [172]	2022	Visual-Inertial SLAM	$\mathbf{K}, \mathbf{r}_{\mathbf{g}} \in \mathbb{R}^2, s, \boldsymbol{\rho}_d$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\boldsymbol{r}_C \in \mathbb{R}^2, \mathbf{b}_g, \mathbf{b}_a$	Cubic Splines Discrete-Time Values
Huai et al. [173]	2022	RS Visual-Inertial Calibration	$t_{\text{RS}}, \{\mathbf{S}_g, \mathbf{M}_g, \mathbf{r}_{ga}\} \in \mathbb{R}^{3\times3},$ $\mathbf{G}_g, \mathbf{S}_a, \mathbf{M}_a, \mathbf{T}_E, \mathbf{t}_E, \mathbf{r}_{\mathbf{g}}$	$\mathbf{T} \in \mathbb{R}^6$(AA), $\mathbf{b}_g, \mathbf{b}_a$	Uniform Quintic B-Splines
Wang et al. [174]	2022	Event-Based VO	-	${}^2\mathbf{p}$	B-Spline
Lang et al. [175]	2022	Visual-Inertial SLAM	$t_{\text{RS}}, \boldsymbol{\rho}_d$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values
Lv et al. [176]	2022	LiDAR-Inertial Calibration	$b_r, s_r, \mathbf{m}_r \in \mathbb{R}^4, \mathbf{S}_g, \mathbf{M}_g,$ $\mathbf{G}_g, \mathbf{S}_a, \mathbf{M}_a, \mathbf{T}_E, \mathbf{t}_{\mathbf{E}}, \mathbf{g}$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values
Lv et al. [177]	2023	LiDAR-Visual-Inertial SLAM	$\mathbf{t}_E, \boldsymbol{\rho}_d$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values
Lang et al. [178, 179]	2023/2024	LVIO	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Non-Uniform Cubic B-Splines Discrete-Time Values
Li et al. [58]*	2023	Theory, UWB-Inertial Localization	$\mathbf{T}_E, \mathbf{r}_{\mathbf{g}}$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3, \mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines
Li [45]	2023	UWB-Inertial Localization	-	$\mathbf{p}$	Uniform Cubic B-Spline
Jung et al. [180]	2023	MDC for Multi-LiDAR Inertial Odometry	-	$\mathbf{T} \in SE(3)$	Uniform Cubic B-Spline
Lu et al. [181]	2023	Event-Inertial Odometry	-	$\mathbf{p}$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values
Nguyen et al. [182]	2024	LIO	-	$\mathbf{r} \in SO(3), \mathbf{p}$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values
Lv et al. [183]	2024	LO	-	$\mathbf{r} \in SO(3), \mathbf{p}$	Uniform Cubic B-Splines
Hug et al. [184]	2024	Odometry Localization	-	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$	Uniform B-/Z-Splines
Li et al. [185]	2024	LiDAR-RADAR-inertial Odometry and Calibration	$\mathbf{T}_E, \mathbf{t}_E$	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform B-Splines Discrete-Time Values
Nguyen et al. [186]	2024	Ground Truth Trajectory Estimation	$\mathbf{b}_g, \mathbf{b}_a$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$	Uniform Quintic B-Spline
Li et al. [187]	2024	LIO	-	$\mathbf{T} \in SU(2) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a$	Uniform Cubic B-Splines Discrete-Time Values

Asterisks (*) indicate works that significantly expanded the mathematical formulation of these methods. See Table VII for definitions.

Sinusoidal temporal bases have also been used in conjunction with polynomials for certifiable range-based localization [165] and object tracking [156], [157], [158].

2) Generalization to Lie Groups: Importantly for robotics, splines were formulated for orientations [132], [133], [136] and then general Lie groups [53]. Despite this, many early applications retained a vector spline representation for orientation or pose [18], [20], [140], [141], [145], [146], [147], [148], [152], [153] before $SE(3)$ splines [142], [149], [151], [154] and general Lie group splines of arbitrary order [155] were applied. Transitioning away from the $\mathbb{R}^3$ formulation for orientation was necessary because of the presence of singularities, the loss of C^{k-2} continuity [132], [133], [142], and because interpolation does not represent a minimum distance in rotation space [142], [160]. For these reasons, the resultant curves under the vector formulation may not realistically approximate motion. Manifold constraints were rarely enforced before the Lie group formulation, although Fleps et al. [140] did so in their SQP approach.

3) Choice of Pose Representation: For 3-D pose estimation using B-splines, a choice exists between the *joint* ($SE(3)$) and *split* ($SO(3) \times \mathbb{R}^3$ or $SU(2) \times \mathbb{R}^3$) representations. Ovrén and Forssén [164] scrutinized this choice for robotic applications and advocate for the split representation in most cases. They showed that under the $SE(3)$ representation, translation, and orientation are coupled in *screw motion*, where implicitly "the pose is manipulated by an internal force and torque," an action called a *wrench* [190], which may be appropriate in some contexts such as robot arm end effector trajectories. In contrast, they argue that the split representation implicitly assumes that an external force and torque actuate the pose. Furthermore, they demonstrate that the analytic acceleration of the joint $SE(3)$ spline is not the kinematic body acceleration in general. Besides the theoretical justifications, they experimentally show that the split representation converges faster and more accurately for their RS visual-inertial SLAM system. Tirado and Civera [57] made a similar argument; however, they advocated for the joint representation as the general case for rigid-body motion. Hug and Chli [31] preferred the split representation, emphasizing the increased computational effort required in the $SE(3)$ representation. For orientation, Haarbach et al. [48] and Li et al. [58] advocated for quaternions ($SU(2)$) over rotation matrices ($SO(3)$). This can be attributed to their space efficiency, requiring four variables instead of nine, and their relative computational efficiency. For the same reasons, Haarbach et al. [48] advocated for the unit dual quaternion $\mathbb{DH}_1$ over $SE(3)$ for B-spline pose interpolation.

4) Modern Advancements: Sommer et al. [55] devised an efficient method for computing Lie group B-spline derivatives, requiring $\mathcal{O}(k)$ instead of $\mathcal{O}(k^2)$ matrix operations for a spline of order k. They further derive a recursive strategy to compute analytic Jacobians for the $SO(3)$ B-spline and its first and second derivatives in $\mathcal{O}(k)$ time. This is extended to other Lie groups [57], [58] with significant computational savings demonstrated over automatic or numerical differentiation [57]. Finally, Johnson et al. [22, Appendix E] provided analytic Jacobians of general Lie group splines up to the second derivative. Hug and Chli [31] contributed analytic Jacobians for the knots, suggesting the possibility that they could be jointly optimized in future work. Recently, Hug et al. [184] used SymForce [191] to obtain analytic B-spline implementations that are much faster than those derived by Sommer et al. [55]. Persson et al. [47] provided a number of novel theoretical contributions: *i)* constant-velocity extrapolation through new basis functions, *ii)* a necessary condition on knot spacing for quaternion splines to express a maximum angular velocity, *iii)* a simple method to derive and compute integrals of spline derivative functions, *iv)* a theoretical and experimental investigation of different spline regularizers, and *v)* an approach for how these integral-type costs should be sampled. Spline regularization has been effective in reducing oscillations [18], [20] and enforcing nonholonomic constraints [170]. In addition, Cioffi et al. [19] demonstrated with batch optimization that B-splines are more accurate than DT for trajectory estimation, especially when estimating for time offsets between sensors.

5) Early Applications in Robotics: Among the first robotic applications of splines were position estimation in object tracking [135] and visual-inertial odometry (VIO) [137]. In another early work, Bibby and Reid [138] represented the vehicle's trajectories and dynamic objects in the environment using cubic interpolating splines in a 2-D dynamic SLAM application. Hadzagic and Michalska [139] used cubic B-splines for 2-D position estimation in ship tracking. Bosse and Zlot [66], [103] used cubic B-splines for trajectory fitting and optimization while keeping LI as their primary CT representation. Later, TBFs were applied to the estimation of robot velocity in a feature-based LiDAR SLAM context [143], [144], using cubic B-splines in the implementation and a Riemann summation of velocity to obtain the pose estimates.

6) Explicit motion distortion correction (MDC): As in LI, B-splines were also used to correct for MDC, including in point cloud registration for LO [148], in a LiDAR SLAM system [161] (extending Droeschel et al. [79]), and RSC for RGB-D odometry [150]. The LO and mapping system presented in Quenzel and Behnke [56] used a temporal B-spline and Gaussian mixture models (GMMs) for local-to-map surfel registration. Lv et al. [167] solved the LiDAR MDC problem for their LIO system by first estimating pose and velocity with IMU integration before fitting a uniform B-spline and using it to correct the motion-distorted point cloud. They use the corrected scans in a feature-based LIO optimization where a B-spline again represents the trajectory. Jung et al. [180] used cubic B-spline interpolation for motion and temporal compensation of multiple LiDARs in their LIO KF.

7) Localization, Odometry, and SLAM: Sheehan et al. [146] used a Catmull–Rom spline for LiDAR localization, and Li et al. [58] used B-splines for UWB-inertial localization. B-splines have been used to process high-rate event camera measurements, which are particularly hard to aggregate for DT systems. This includes event [151] and event-inertial localization [162], event VO [174], and map-free event-inertial ego-velocity estimation [181]. Leveraging prior work [77], Nguyen et al. [182] proposed an iterative, parallelized, on-manifold linear solver for real-time surfel-based LIO. Lv et al. [183] used a "spot uncertainty model" to weight point-to-plane residuals in their LO system. Ng et al. [168] applied B-splines to RADAR-inertial odometry, using radial velocity measurements from multiple radars to form ego-velocity residuals. Adopting uniform cubic B-splines for the trajectory, Li et al. [185] jointly optimized LiDAR point-to-plane, RADAR Doppler velocity, and inertial residuals for multi-sensor odometry. B-splines have also been used as the CT trajectory formulation in RS visual [142], [149], [154], RS visual-inertial [175], visual-inertial [31], [164], [171], multi-camera [169], LiDAR [161], and LiDAR-visual-inertial [177], [178], [179] SLAM.

8) Sensor Calibration: TBFs and B-splines have been used for sensor calibration, including estimation of RS camera line delay [145], [173], [175], visual-inertial [18], and LiDAR-visual-inertial [147], [177] extrinsic calibration, and comprehensive simultaneous intrinsic and extrinsic calibration for LiDAR-visual-inertial [152], multi-IMU visual-inertial [153], and visual-inertial [47], [159] configurations. Uniform quintic B-splines have been applied to RS visual-inertial calibration [173], and uniform cubic B-splines to targetless LiDAR-inertial calibration in structured environments [166], [176].

9) Online Optimization: While many perform batch optimization, some works operate online using *sliding-window* optimization (a.k.a. *fixed-lag smoothing*). Early works use linear splines (equivalent to LI) [102], [103], [104], [107], but Zlot and Bosse [103] also did with cubic B-splines. Persson et al. [47] were among the first to use windowed optimization for real-time odometry. Lang et al. [175] developed B-spline marginalization strategies for RS visual-inertial fixed-lag smoothing, refined by Lv et al. [177] to achieve real-time sliding-window LiDAR-visual-inertial SLAM. More recent spline-based odometry systems have continued to develop marginalization strategies [178], [183], [185], with Lang et al. [178] to first to do so for non-uniform splines. Importantly, the use of per-point residuals in these works allows MDC to occur simultaneously to state estimation [177], [178], [183], [185]. Quenzel and Behnke [56], Mo and Sattar [172] who incorporated inertial measurements through cubic splines, and Tirado and Civera [57] who employed non-uniform cubic B-splines for object tracking, also employ sliding-window optimization. In contrast to these optimization approaches, Li [45] devised a probabilistic recursive filter that maintained the control points for a single cubic B-spline segment in its state and could thus run in real-time with uncertainty estimation. Hug et al. [184] took a different approach, proposing a Gaussian belief propagation (GBP) optimization framework called Hyperion, where factor graph optimization is achieved through efficient message passing between nodes and factors. While usually slower than existing nonlinear least squares (NLLS) solvers such as Ceres [192], GBP computes variable (inverse) uncertainty during optimization instead of as a postoptimization procedure, allowing strategies for targeted updating (e.g., of nonconverged nodes), and is inherently scalable to multi-agent systems.

C. Temporal Gaussian Process (TGPs)

A chronologically ordered overview of all the TGP methods surveyed in this work is presented in Table VI.

1) Early Works: The use of GPs for CT state estimation was first realized by Tong et al. [49], [59], [229], where they presented Gaussian process Gauss–Newton (GPGN), a nonlinear batch optimization method, and applied it to 2-D range-based localization [59] and SLAM [49]. In these early formulations, the connection with TBFs was made explicit in the derivation, with the transition from parametric to nonparametric achieved through the *kernel trick*. Tong et al. [193] subsequently extended GPGN to 3-D for their laser-based visual SLAM system. Crucially, Barfoot et al. [34] and Anderson et al. [35] demonstrated that GPs using the LTV-SDE class of prior have exactly sparse inverse kernel matrices, enabling efficient regression and interpolation. The computation of the inverse kernel matrix is reduced from cubic to linear time, and interpolation can be computed in constant time. When using this prior with a linear measurement model, they show exact equivalence to discrete-time smoothing with identical computational complexity when evaluated at the measurement times [24], [35]. Solving the SLAM problem using this method was called simultaneous trajectory estimation and mapping (STEAM), with Anderson and Barfoot [61] extending the approach to $SE(3)$ poses using local LTV-SDEs (see Section III-C1). Nonlinear models are supported in these works through linearization and iterative optimization.

2) Optimization: Exploiting the sparsity of the formulation, the STEAM problem can be modeled as a factor graph with binary GP prior factors connecting adjacent states [35], [61], enabling batch [61] and sliding-window optimization [220]. Yan et al. [194], [195] used this to develop a GP version of the incremental optimizer iSAM2 [230], and with iGPMP2 [60] applied this incremental GP approach to motion-planning problems. *Sparse interpolated measurement factors* allow measurements to be added between the explicitly estimated states [35], [36], [49], [195], and can be readily used within an incremental Bayes tree optimization scheme [195]. Dong et al. [62], [63] provided a formulation for general matrix Lie groups before integrating it into their simultaneous trajectory estimation and planning (STEAP) framework [197], [198]. Exploring a different direction, Barfoot et al. [205] proposed exactly sparse Gaussian variational inference (ESGVI) to fit a Gaussian to the full Bayesian posterior of the problem states. This approximation ought to be more representative than the Laplace approximation from MAP estimation (6). They devise a Newton-style iterative solver that avoids the need for analytic derivatives through cubature rules and exploits the inherent sparsity of the problem structure to achieve the same computational complexity as MAP estimation. Wong et al. [206] then extended this approach for hyperparameter learning with an expectation maximization (EM) optimization framework.

3) Applications Using the WNOA Prior: The Gauss–Newton (GN) TGP approach [35], [61], [62] has been applied in the context of *teach and repeat*, such as in visual odometry (VO) for UAVs [201], [202], or LiDAR and RADAR odometry [216], [224]. Liu et al. [217] and Wang and Gammell [218] have proposed incremental event-based VO pipelines based on this CT representation, employing the GN algorithm in the underlying optimization. Li et al. [212] demonstrate LiDAR-inertial extrinsic calibration and localization in structured environments. Incremental Bayes-tree-based optimizers [230] have been used in SLAM for agricultural monitoring [196] and inertial-GNSS odometry [214]. Recent works [215], [219] have applied TGPs to FMCW LiDAR odometry. Rather than aggregating points (e.g., for scan registration), Wu et al. [215] formulated point-to-plane and radial-velocity residuals for each point. Yoon et al. [219] combined radial-velocity residuals with a gyroscope for linear estimation of the total DoF ego-velocity, which they integrate into $SE(3)$ poses. Judd and Gammell [203], [204] addressed the *multimotion* estimation problem with TGPs in their occlusion-aware VO pipeline, estimating for the $SE(3)$ ego-motion and trajectories of points on dynamic objects.

4) Applications Using Other Priors: The aforementioned works all use the WNOA prior. Tang et al. [64] proposed the *"constant-acceleration"* white-noise-on-jerk (WNOJ) model instead. They achieve higher accuracies in some cases and prove that when the chosen prior does not match the motion, the state

TABLE VI
TEMPORAL GAUSSIAN PROCESS (TGP) CT STATE REPRESENTATION WORKS

	Year	Applications	Time-Invariant Estimates State(s)	Time-Varying Estimates State(s)	 State Representation(s)
Tong et al. [59]*	2012	Theory Range-Based Localization	- -	$\mathbb{R}^n$ $\{{}^2\mathbf{T}, {}^2\dot{\mathbf{T}}\} \in \mathbb{R}^6$	WNOA TGP
Tong et al. [49]*	2013	Theory Range-Based SLAM	- ${}^2\mathbf{L}$	$\mathbb{R}^n$ ${}^2\mathbf{T} \in \mathbb{R}^3$	WNOA TGP
Tong et al. [193]	2014	Laser-Based Visual SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3)$	WNOA TGP
Barfoot et al. [34]	2014	Range-Based SLAM	${}^2\mathbf{L}$	$\{{}^2\mathbf{T}, {}^2\dot{\mathbf{T}}\} \in \mathbb{R}^6$	WNOA TGP
Yan et al. [194, 195]*	2014/2017	Range-Based SLAM	${}^2\mathbf{L}$	$\{{}^2\mathbf{T}, {}^2\dot{\mathbf{T}}\} \in \mathbb{R}^6$	WNOA TGP
Anderson et al. [35]*	2015	Range-Based SLAM	${}^2\mathbf{L}$	$\{{}^2\mathbf{T}, {}^2\dot{\mathbf{T}}\} \in \mathbb{R}^6$	WNOA TGP
Anderson and Barfoot [61]*	2015	Visual SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Dong et al. [196]	2017	Visual-Inertial-GNSS SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a$	WNOA TGP Discrete-Time Values
Dong et al. [62, 63]*	2017/2018	Theory Range-Based SLAM Inertial Attitude Estimation Monocular Visual SLAM Mobile Manipulator Planning (PR2)	- ${}^2\mathbf{L}$ - $\mathbf{L}$ -	$\mathcal{L}, \mathbb{R}^d$ ${}^2\mathbf{T} \in SE(2), \mathbb{R}^3, {}^2\dot{\mathbf{T}} \in \mathbb{R}^3$ $\mathbf{r} \in SO(3), \dot{\mathbf{r}}$ $\mathbf{T} \in SIM(3), \dot{\mathbf{T}} \in \mathbb{R}^7$ $\mathbf{x} \in SE(2) \times \mathbb{R}^{15}, \dot{\mathbf{x}} \in \mathbb{R}^{18}$	WNOA TGP
Mukadam et al. [197, 198]*	2017/2019	Two-Link Mobile Arm Planning Mobile Manipulator Planning (PR2) Mobile Manipulator Planning (Vector)	-	$\mathbf{x} \in SE(2) \times \mathbb{R}^{2}, \dot{\mathbf{x}} \in \mathbb{R}^{2}$ $\mathbf{x} \in SE(2) \times \mathbb{R}^{15}, \dot{\mathbf{x}} \in \mathbb{R}^{18}$ $\mathbf{x} \in SE(2) \times \mathbb{R}^{6}, \dot{\mathbf{x}} \in \mathbb{R}^{9}$	WNOA TGP
Rana et al. [199]	2018	Learning from Demonstration and Motion Planning	-	$\mathbf{p}, \dot{\mathbf{p}}$	WNOA/Learnt TGPs
Maric et al. [200]	2018	Motion Planning	-	$\mathbf{x} \in \mathbb{R}^n$	WNOA TGP
Warren et al. [201]	2018	VO for Teach & Repeat	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Warren et al. [202]	2018	VO for Teach & Repeat	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Tang et al. [64]*	2019	Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}$	WNOJ TGP
Judd [203], Judd and Gammell [204]	2019/2020	Dynamic SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, (\mathbf{T}_D \in SE(3), \dot{\mathbf{T}}_D)^D$	WNOA TGPs
Barfoot et al. [205]	2020	Wheeled Robot SLAM	${}^2\mathbf{L}$	${}^2\mathbf{p}, {}^2\mathbf{r} \in \mathbb{R}$	WNOA TGP
Wong et al. [206]	2020	Hyperparameter Estimation for LiDAR Localization	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Wong et al. [207]*	2020	LiDAR Localization LO	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}$	Singer TGP
Peršić et al. [208]	2021	Spatiotemporal Calibration from Object Tracking	-	$(\mathbf{p}_D, \dot{\mathbf{p}}_D, \ddot{\mathbf{p}}_D)^D$	WNOJ TGPs
Kapushev et al. [209]	2021	SLAM	${}^2\mathbf{L}$	${}^2\mathbf{T} \in \mathbb{R}^3$	RFF-Gaussian TGP
Judd and Gammell [210, 211]	2021/2024	Dynamic SLAM	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}, (\mathbf{T}_D \in SE(3), \dot{\mathbf{T}}_D, \ddot{\mathbf{T}}_D)^D$	WNOA / WNOJ TGPs
Li et al. [212]	2021	LiDAR-Inertial Extrinsic Calibration and Localization	$\mathbf{T}_E, \mathbf{t}_E$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a$	WNOA TGP Discrete-Time Values
Zhang et al. [213]	2022	LiDAR-Inertial-GNSS Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a, b_{\text{GNSS, t}}, b_{\text{GNSS, v}}, n_{\text{GNSS}}$	WNOJ TGP
Zhang et al. [214]	2022	Inertial-GNSS Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a, b_{\text{GNSS, t}}, b_{\text{GNSS, v}}$	WNOA TGP Discrete-Time Values
Wu et al. [215]	2022	FMCW LiDAR Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Burnett et al. [216]	2022	LiDAR and RADAR Odometry for Teach & Repeat	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Liu et al. [217]	2022	Event-Based SLAM for VO	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Wang and Gammell [218]	2023	Event-Based SLAM for VO	$\mathbf{L}$	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Yoon et al. [219]	2023	FMCW LiDAR-Inertial Odometry	-	$\dot{\mathbf{T}}$	WNOA TGP
Goudar et al. [220]	2023	UWB Localization	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Le Gentil et al. [221]	2023	Event MDC Feature Tracking	-	${}^2\mathbf{T} \in \mathbb{R}^3$	Squared Exponential TGPs
Le Gentil and Vidal Calleja [222]	2023	IMU-Pose Extrinsic Calibration	$\mathbf{T}_E, \mathbf{t}_E$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3$ $\mathbf{b}_g, \mathbf{b}_a, \mathbf{r_g} \in SO(3)$	Gaussian TGP Discrete-Time Values
Zheng and Zhu [223]	2024	LIO	-	$\mathbf{r} \in SO(3), \dot{\mathbf{r}}$ $\mathbf{p}, \dot{\mathbf{p}}, \ddot{\mathbf{p}}$ $\{\mathbf{b}_g\}^{N_g}, \{\mathbf{b}_a\}^{N_a}$	WNOA TGP WNOJ TGP WNOV TGPs
Johnson et al. [22]*	2024	Visual-Inertial Localization	$\mathbf{b}_g, \mathbf{b}_a$	$\mathbf{T} \in SE(3)/SO(3) \times \mathbb{R}^3, \dot{\mathbf{T}}, \ddot{\mathbf{T}}$ $\mathbf{T} \in SE(3)/SO(3) \times \mathbb{R}^3$	WNOJ TGP(s) Uniform Cubic / Quintic B-Spline(s)
Lisus et al. [224]	2024	RADAR Odometry for Teach & Repeat	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Burnett et al. [225]	2024	LiDAR-Inertial and RADAR-Inertial Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a, (\mathbf{r_g} \in SO(3))$	WNOA TGP WNOV TGPs
Barfoot et al. [226]	2024	Certifiable Trajectory Estimation	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}$	WNOA TGP
Zhang et al. [227]	2024	LiDAR-Inertial-GNSS-Velocimeter Odometry	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}$ $\mathbf{b}_g, \mathbf{b}_a, b_{\text{GNSS, t}}, b_{\text{GNSS, v}}$	WNOJ TGP Discrete-Time Values
Nguyen et al. [65]	2024	Visual-Inertial Localization and Calibration UWB-Inertial Localization Multi-LiDAR Localization and Calibration	$\mathbf{T}_E, \mathbf{g}, \mathbf{b}_g, \mathbf{b}_a$	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{T}}, \ddot{\mathbf{T}}$	WNOJ TGPs
Burnett et al. [228]	2024	LIO	-	$\mathbf{T} \in SE(3), \dot{\mathbf{T}}, \ddot{\mathbf{T}}, \mathbf{b}_g, \mathbf{b}_a$	Singer & WNOV TGPs
Shen et al. [46]	2024	Multi-LiDAR Odometry	-	$\mathbf{T} \in SO(3) \times \mathbb{R}^3, \dot{\mathbf{T}}, \ddot{\mathbf{p}}$	Constant Angular Velocity & Acceleration TGP

Stars (*) indicate works that significantly expanded the mathematical formulation of this method. See Table VII for definitions.

estimate will be biased along certain DoF. The WNOJ prior has since been used for online GNSS multi-sensor odometry [213], [227] and target-based multi-sensor extrinsic calibration [208]. Recently, Zheng and Zhu [223] proposed multi-sensor LIO using WNOJ for position, WNOA for orientation, and white-noise-on-velocity (WNOV) (a.k.a. "random-walk") for IMU biases, and achieved state-of-the-art results. As in Wu et al. [215] and their previous work [30], they avoided measurement aggregation (e.g., LiDAR registration, IMU preintegration) while retaining real-time performance through the split pose representation and efficient map management. Burnett et al. [225] also utilized the WNOV prior for IMU biases, demonstrating effective LiDAR-inertial and RADAR-inertial odometry on various datasets. Judd and Gammell [210], [211] compared the WNOJ and WNOA priors in their multimotion visual odometry (MVO) pipeline [203], [204]. Johnson et al. [22] recently were the first to compare TGPs and B-splines in visual-inertial localization experiments. They compared WNOJ TGPs to cubic and quintic B-splines, both with and without motion prior factors, and for both the split and joint pose representations. They concluded that with the same motion models and measurements, and when the degree-of-differentiability matches, temporal splines and GPs are similar in accuracy and solve time. Furthermore, they show that motion prior factors act as regularizers, preventing overfitting to noisy measurements. However, these benefits are significantly diminished when an IMU is available. Zheng and Zhu [30] and Johnson et al. [22] investigated the efficacy of the split pose representation and concluded its primary benefit is the reduction of computation time, both at solve and inference time. From the existing literature, Johnson et al. [22] provided insights into computation timings of these CT method variants, while Cioffi et al. [19] demonstrated that CT solve times can be on par with DT. Most recently, Nguyen et al. [65] released their split pose WNOJ implementation and provide localization and calibration examples for a variety of sensor combinations. They further provided a generalization of white-noise-on-derivative (WNOD) priors for any order and analytic Jacobians for their residuals. Wong et al. [207] investigated the *Singer prior*, which represents latent accelerations as a GP with a Matérn kernel. They argue that the data-driven optimization of an additional parameter enables a better representation of the robot's motion. Burnett et al. [228] utilized the Singer prior for LIO, and comparatively analyzed the use of IMU as a motion model input (for preintegration) and as a measurement.

5) Other Approaches: Shen et al. [46] adapted the TGP method to an EKF for the first time to achieve real-time dropout-resilient multi-LiDAR odometry. They create point-to-voxel residuals with a degeneracy-aware point sampling strategy and an incremental, probabilistic voxel-based map. Gentil et al. [221] proposed a novel method for event camera motion compensation using GPs. By treating camera motion as $SE(2)$ motion in the image frame over short time periods, they modeled each DoF as independent zero-mean GPs with Gaussian kernels. The inducing values (i.e., set of observations) of these are then the hyperparameters of another GP representing a continuous occupancy field of events in the image plane. By optimizing these through log-marginal likelihood maximization, this occupancy field, and hence a continuous distance field as its negative logarithm, can be inferred and used for homography registration and pattern tracking. Gentil and Vidal Calleja [222] used a GP pose representation in the extrinsic calibration of an IMU with a pose measurement reference frame (e.g., motion capture or robotic arm frames). Kapushev et al. [209] proposed a SLAM algorithm evaluated in 2-D which models the pose as a GP, with a random Fourier features (RFF) approximation of the Gaussian kernel and a discretized motion model or smoothing splines for the prior mean.

V. Open Problems

A. Temporal Splines

1) Knot Selection: The optimal selection of knots remains an important unresolved problems of the spline method. The simplest and most common approach is to use uniform splines, tuning the frequency according to the application. While uniform splines benefit from a fixed blending matrix that needs to be computed only once, they implicitly assume that excitation of the state evolution (e.g., robot motion) remains constant. Hence, uniform knots can lead to underfitting or overfitting of the trajectory in settings with varying dynamics [31]. Furthermore, a control point is only well-constrained if there are sufficient measurements on the associated segments, meaning that optimization can be poorly posed if the knot frequency is too high relative to the measurement frequency.

Some works associate knots with the measurement [138], [169], with Bibby and Reid [138] selectively removed them where acceleration and jerk are sufficiently similar. Others [160], [178] increased the knot frequency according to the maximum magnitudes of acceleration and angular velocity (from IMU measurements). Another suggestion is to iteratively add knots until the mean visual reprojection error is sufficiently low [160] or to increase knots until the reprojection residuals within segments agree with an expected value [145]. Anderson et al. [50] used this idea to decide when to increase the active resolution level in their hierarchical formulation, based on work by Gortler and Cohen [134] who applied wavelet detail functions over a base B-spline signal function.

An alternative approach is *spline error weighting* [163], [164], which recognizes that existing methods neglect approximation error arising from model mismatch. By formulating spline fitting in the frequency domain, they devise a method to capture model mismatch and measurement error by setting the residual covariance based on the measurement noise and variance of the error. They further use this frequency analysis to propose a method for selecting a suitable uniform knot frequency based on the fraction of signal energy to be retained.

Several other schemes have been offered. Dubé et al. [104] proposed several distributions for the knots within the sliding window of the optimization. These include an exponential distribution, a frequency analysis distribution based on average correction signal power, and a uniform distribution, potentially in combination. Hug and Chli [31] discussed optimization of the knots themselves and derived Jacobians for them. However, this idea has not yet been pursued experimentally. Regularizing motion terms [20], [22], [47] may avoid the issue of underconstrained control points; however, it may add assumptions conflicting with the implicit spline constraints.

2) Certifiability and Recovery Guarantees: Important open research questions arise because the optimization methods commonly used for estimation may converge to bad local minima or fail to converge altogether. Indeed, depending on the measurement modalities used, the residuals of the factor graph (i.e., optimization cost terms) may be nonconvex. Particularly, if states are confined to manifold spaces, this may lead to nonconvex constraints and exacerbate the problem. Multiple recent works [231], [232], [233], [234] show that nonconvexity may give rise to bad local minima in which local solvers like GN can get stuck. In the face of these convergence issues, the global minimum can still be recovered from a tight but more costly convex relaxation (e.g., for initialization [232]). A related question is that of the uniqueness of the solution. Because splines promote smoothness, fewer measurements may be required to obtain unique state estimates. However, exactly how many are necessary remains an open problem, and failure to meet this may result in ambiguities. Recovery guarantees have been derived for polynomial or sinusoidal basis functions with noiseless range measurements [165], but extending these to more general cases remains open.

3) Complex Processes: The piecewise-polynomial interpolation of splines may be problematic in estimating high-frequency or hybrid dynamic systems, such as those that may occur in the presence of vibration or impacts. Rather than increase knot frequency or spline order, which raises the computational load, Li et al. [187] proposed using the RMSE of the accelerometer measurements to adaptively select between using IMU measurements directly or preintegrating them as a form of low-pass filter. In addition, they apply constant-velocity factors to the roll and pitch when the acceleration is sufficiently low. Resilience to nonpolynomial and hybrid dynamics remains a challenging problem for the spline method.

4) Incremental Optimization: Sliding-window optimization can facilitate real-time operation [56], [57], [102], [103], [104], [107], [172], [175], [177], [178], [182], [183]. Other incremental schemes, such as those based on the Bayes tree [230], have not been explored. Various graphical models have been presented for splines [57], [171], [172], however only very recent works [175], [177], [178], [183] have made serious efforts to represent spline estimation with factor graphs and use these to design marginalization strategies. Leveraging these to incorporate incremental optimizers like iSAM2 [230] could be explored in future work.

5) Initialization: Initialization of control points is rarely discussed in the literature. However, appropriate initialization certainly affects convergence time and optimality. Simple strategies for initializing new control points include using the value of the previous control point, or IMU dead reckoning. However, the state estimate may be far from the control point, especially for highly dynamic motion and high-order splines, and no inverse interpolation function exists to compute control points from the state. One strategy adopted by Huang et al. [170] was to use a spline approximation algorithm from Piegl and Tiller [235] for initialization. Li et al. [185] devised a sensor-specific coarse-to-fine procedure to initialize control points, biases, and extrinsic calibration parameters. Kang and Park [136] offered initialization equations for different parameterizations of orientation splines based on boundary conditions.

6) Covariance Interpolation: It is often desirable to infer the uncertainty of the state at the interpolation time from the uncertainty of the control points. To a limited extent, covariance interpolation for vector-valued splines has been demonstrated [20, Fig. 6] [138]. Bibby and Reid [138] provided a derivation for their non-uniform cubic splines, applied to dynamic SLAM in 2-D, using the Jacobian of the spline interpolation function. State covariance is also estimated by the probabilistic filter proposed by Li [45]. However, no works provide a covariance interpolation formulation for Lie group splines.

B. Temporal Gaussian Process (TGPs)

1) State Time Selection: The TGP method provides flexibility in selecting state times. This is analogous to the spline knot selection problem. However, several differences in the methodology must be considered. First, the process state variables are always constrained through GP prior factors, meaning that TGPs are less sensitive to sensor dropout and do not overfit like splines. In addition, while measurements can be incorporated at any time, there are computational advantages of using noninterpolated factors, achieved by setting the state time to the measurement time. Finally, when using local LTV-SDEs, the approximation error of the linearization is more significant when the changes between states are more considerable. Thus, intuitively, the temporal density of states should be higher when the motion is more dynamic and less when the state changes slowly. Regardless, the approaches proposed for spline knot selection may also apply to TGPs.

A straightforward approach is to set the state times as the measurement times of low-rate sensors, exploiting the efficiency of noninterpolated factors. This is more advantageous when many factors are associated with a single measurement time, such as landmark measurements in global shutter visual SLAM. Adapting the state times based on IMU measurements or residual errors as a surrogate metric for motion change would also be possible. For more theoretical approaches, increasing knots until residual errors match expected values [145] could be applied to state times. Alternatively, the state times—usually fixed dependent variables—could be added as optimization variables, as discussed for knots [31]. Furthermore, residuals could be weighted analogously to spline error weighting [163], [164] to account for the model approximation error uncaptured by the LTV-SDE prior.

2) Certifiability and Recovery Guarantees: As for splines, common measurement models and state constraints may yield nonconvex optimization problems that challenge convergence and global optimality guarantees for local solvers like GN. A series of works on (DT) globally optimal state estimation [233], [234], [236] shows that many MAP problems can be formulated as quadratically constrainted quadratic programs (QCQPs), allowing for tight (convex) semidefinite program (SDP) relaxations. The latter can be used to *globally* solve or certify CT estimation problems.

In contrast with the spline method, GP motion priors can be effortlessly integrated with this paradigm, as shown recently for the examples of CT range-only localization [237] and pose-graph optimization [226]. Indeed, to incorporate a GP motion prior, the state is augmented with the necessary number of temporal derivatives, after which the motion prior regularization always takes the form of additional interstate quadratic cost terms, which can be readily included in the QCQP formulation. However, augmenting the state and regularizing the problem in this way may affect the tightness of the SDP relaxations of the QCQP, which, in turn, can break certifiability. For pose-graph optimization in particular, a significant number of so-called *redundant constraints* are required for tightness at reasonable noise levels when using the WNOA prior [226, Eq. (57)], which breaks typical constraint qualifications and significantly increases the computational complexity of the used SDP solvers. Measures to increase and (*a-priori*) understand tightness and to speed up SDP solvers are crucial for advancing the practicality of these methods.

As for the required number of measurements, TGPs behave differently than splines because of the Bayesian nature of the approach. For example, in the absence of measurements, the method will return a trajectory according to the motion prior. This allows the method to interpolate between states even when no measurements are acquired, while the uncertainty of such interpolations will grow.

3) Complex Processes: Tang et al. [64] showed that selecting a process model prior that does not represent state evolution leads to biased estimation. Thus, it is clear that a prior that most appropriately reflects the state evolution should be chosen. TGPs inherently allows for complex process dynamics and control inputs to regularize the trajectory and guide interpolation. However, only relatively simple models, such as the WNOA or WNOJ priors, have been examined thus far. While efficient to compute and require tuning only a small number of hyperparameters, they do not accurately approximate the dynamics of all systems. Systems with hybrid dynamics or impacts (e.g., legged robotics) may be particularly problematic since impulse-like accelerations induce perceived velocity discontinuities, violating the assumptions of existing priors. Further research is needed into the feasibility and tractability of system-specific priors and control input utilization.

Some recent works have used learning in the modeling of system dynamics for fast online use [238], [239], and it would be reasonable to consider this idea in the computation or augmentation of the prior. A principled method for tuning hyperparameters (e.g., $\mathbf{Q}_C$) is the maximization of the marginal log-likelihood using a training dataset [35] (over hand-tuning [60], [64], [227] or fitting to data [61]), such as with the Singer prior [207]. While this typically requires ground truth data, the EM parameter learning framework extending ESGVI [205] proposed by Wong et al. [206] demonstrated that hyperparameters can be learned from the original noisy measurement data. The Singer prior [207] is an example of an alternative approach to process modeling, whereby an SDE is derived from factorization of the spectral density of a GP prior, such as a kernel from the Matérn family [240], [241]. It remains to be seen what methods could be effectively employed to learn reasonable GP priors for complex processes.

C. Evaluation and Benchmarking

Public benchmarks and appropriate metrics must be established with which CT algorithms can be compared and evaluated. While many works provide comparisons on public benchmarks to other hand-picked methods [30], [76], [177], [178], [180], [182], [223], they are frequently inconsistent, adding ambiguity. Since CT methods can fuse high-rate asynchronous measurements, benchmarks must consider how to provide sensor data and ground truth references. For example, MCD [186] recently leveraged survey-grade maps to optimize a "ground truth" B-spline from LiDAR-inertial data. Existing algorithms are usually compared with aggregated absolute or relative trajectory errors, which are insufficient metrics for understanding their complete behavior. Zhang and Scaramuzza [242] demonstrated the capacity for temporal GPs with Gaussian kernels to represent ground truth trajectories and proposed generalized absolute and relative errors for principled trajectory evaluation.

VI. Applications to Other Domains

A. Temporal Splines

Splines have been extensively used for planning in robotics, including wheeled robots [243], legged locomotion [244], wheel-legged locomotion [245], and manipulation [246], [247]. However, due to their smoothness guarantees, splines are especially appealing for *differentially flat systems*, such as uncrewed aerial vehicles (UAVs) [248], [249]. In a differentially flat system, the states and the inputs can be written as algebraic functions of some variables (called *flat outputs*) and their derivatives. The continuity of splines can guarantee that the UAV's trajectory is dynamically feasible.

Moreover, splines whose basis functions form a partition of unity (i.e., nonnegative, summing to one) exhibit the *convex hull* property, which guarantees that each spline segment is wholly contained within the convex hull of its control points. This is leveraged by many trajectory planning works [250], [251], [252], [253], [254], [255], [256], since it substantially simplifies obstacle avoidance constraints [257]. B-splines are the typical choice of splines for trajectory planning, as they can guarantee smoothness and high continuity by construction without imposing explicit constraints. However, the convex hull of the B-Spline control points is typically much larger than the segment itself. To reduce conservativeness, some works [258] use B-spline control points as decision variables in trajectory planning, but then, the Bézier control points of each interval of this B-spline are used in the obstacle avoidance constraints. The MINVO control points [259], [260] allow for even tighter enclosures and have been leveraged for obstacle avoidance [260].

To a limited degree, shape estimation for *continuum* or flexible robots has also been found to be amenable to spline and TBF theory since they are also 1-D estimation problems, parametrized by arc length instead of time. For example, weighted combinations of basis functions [261], [262] and Bézier curves [263] have been explored.

Splines have also been leveraged to represent specific environment features. For instance, Catmull–Rom splines are used by Qiao et al. [264] to parametrize the road lanes. B-splines are used by Rodrigues et al. [265] to represent unstructured 2-D environments more compactly, and in other works [266], [267], [268] to represent the sensor data in their state estimation frameworks. Spline theory has also been utilized for perception outside the state estimation problems discussed in this work, such as the spatio-temporal representation of event camera feature tracks with Bézier curves [269].

In machine learning, apart from traditional regression [270], [271], splines have also recently found applications in neural networks [272], [273], [274], including recently Kolmogorov–Arnold Networks (KANs) [275], proposed as a promising alternative to the traditional multilayer perception (MLP). Moreover, Yang et al. [276] and Roth et al. [277] used splines as the output of their end-to-end learned neural network-based local planner, trained imperatively through bilevel optimization.

B. Temporal Gaussian Process (TGPs)

GPs have been deployed for a large number of other tasks in robotics. GPs with Gaussian kernels have often been used in Bayesian filters, such as for replacing or enhancing the parametric prediction and observation models [44], [278], [279]. The use of GPs as surrogate observation models has been explored for signal-strength (e.g., WiFi, magnetic flux) state estimation applications [280], [281], [282], [283], and as dynamics prediction models within the control community [284], [285], [286], [287], [288]. In informative path planning applications, GPs have been used for representing scalar environmental fields, such as temperature, salinity, methane concentration, magnetic field intensity, and terrain height [289], [290], [291], [292], [293], [294]. GPs have also seen extensive use in mapping, including occupancy maps [295], [296], [297], [298], [299], [300], [301], [302], [303], [304], [305], [306], implicit surfaces [307], [308], [309], [310], [311], [312], [313], [314], [315], [316], [317], [318], their combination [319], gradient maps [320], [321], [322], [323], and distance fields [221], [315], [318], [324], [325],

TABLE VII
VARIABLE DEFINITIONS

Variable	State Representation(s)	Description
${}^n\mathbf{p}$	$\mathbb{R}^n$	n-Dimensional Position
${}^n\dot{\mathbf{p}}$	$\mathbb{R}^n$	n-Dimensional Linear Velocity
${}^n\ddot{\mathbf{p}}$	$\mathbb{R}^n$	n-Dimensional Linear Acceleration
${}^n\mathbf{r}$	$SO(n), \mathbb{R}^{n(n-1)/2}, SU(2)$	n-Dimensional Orientation
${}^n\dot{\mathbf{r}}$	$\mathbb{R}^{n(n-1)/2}$	n-Dimensional Angular Velocity
${}^n\ddot{\mathbf{r}}$	$\mathbb{R}^{n(n-1)/2}$	n-Dimensional Angular Acceleration
${}^n\mathbf{T}$	$SE(n), {}^n\mathbf{r} \times {}^n\mathbf{p}$	n-Dimensional Pose
${}^n\dot{\mathbf{T}}$	$\mathbb{R}^{n(n+1)/2}$	n-Dimensional Pose Velocity
${}^n\ddot{\mathbf{T}}$	$\mathbb{R}^{n(n+1)/2}$	n-Dimensional Pose Acceleration
$\mathcal{S} = \{\mathcal{S}_E, \mathcal{S}_I\}$	-	Sensor Calibration Variables
$\mathcal{S}_I$	-	Sensor Intrinsics
$\mathbf{K}$	-	Camera Projection Intrinsics
t_{RS}	$\mathbb{R}$	Rolling Shutter Line Delay
t_e	$\mathbb{R}$	Camera Exposure Time
$\boldsymbol{r}_C$	$\mathbb{R}^n$	Camera Response Parameters
$\boldsymbol{d}_C$	$\mathbb{R}^n$	Camera Distortion Parameters
b_r	$\mathbb{R}$	Range Bias
s_r	$\mathbb{R}$	Range Scale
$\mathbf{m}_r$	-	Range Frame Misalignment Parameters
$\mathbf{b}_g$	$\mathbb{R}^3$	Gyroscope Bias
$\mathbf{S}_g$	$\mathbb{R}^{3\times3}$ (diagonal)	Gyroscope Scale
$\mathbf{M}_g$	$\mathbb{R}^{3\times3}$ (triangular)	Gyroscope Axis Misalignment
$\mathbf{G}_g$	$\mathbb{R}^{3\times3}$	Gyroscope G-Sensitivity
$\mathbf{b}_a$	$\mathbb{R}^3$	Accelerometer Bias
$\mathbf{S}_a$	$\mathbb{R}^{3\times3}$ (diagonal)	Accelerometer Scale
$\mathbf{M}_a$	$\mathbb{R}^{3\times3}$ (triangular)	Accelerometer Axis Misalignment
$b_{\text{GNSS, t}}$	$\mathbb{R}$	GNSS Receiver Clock Bias
$b_{\text{GNSS, v}}$	$\mathbb{R}$	GNSS Receiver Velocity Bias
n_{GNSS}	$\mathbb{N}^0$	Number of Satellites
$\mathcal{S}_E = \{\mathbf{T}_E, \mathbf{t}_E\}$	-	Sensor Extrinsics
$\mathbf{T}_E$	$(\mathbf{T})^n$	Inter-sensor Relative Poses
$\mathbf{p}_{ia}$	$\mathbb{R}^3$	IMU-Accelerometer Translation
$\mathbf{r}_{ga}$	$\mathbf{r}$	Gyroscope-Accelerometer Rotation
$\mathbf{t}_E$	$\mathbb{R}^n$	Inter-sensor Time Offsets
$\mathbf{g}$	$\mathbb{R}^3$	Gravity Vector
$\mathbf{r}_\mathbf{g}$	$\mathbb{R}^2, \mathbf{r}$	Gravity Orientation
s	$\mathbb{R}$	Map Scale
${}^n\mathcal{M}$	-	Map in n-Dimensional Space
${}^n\mathcal{M}_S$	-	Surfel Map in n-Dimensional Space
${}^n\mathcal{M}_O$	-	Occupancy Map in n-Dimensional Space
${}^n\mathbf{L}$	$\mathbb{R}^{n\times L}$	Point Landmarks in n-Dimensional Space
$\mathbf{L}_\pi$	$\mathbb{R}^{4\times L}$	Plane Landmarks in 3D Space
$\mathbf{p}_d$	$\mathbb{R}^p$	Pixel/Landmark Depths/Distances
$\boldsymbol{\rho}_d$	$\mathbb{R}^p$	Pixel/Landmark Inverse Depths/Distances
$\mathbf{p}_b$	$\mathbb{R}^{n\times p}$	Pixel/Landmark Bearing Vectors
$\mathbf{m}_o$	$\mathbb{R}^n$	Material Optical Parameters
$\mathbf{T}_D/\dot{\mathbf{T}}_D/\ddot{\mathbf{T}}_D$	$\mathbf{T}/\dot{\mathbf{T}}/\ddot{\mathbf{T}}$	Dynamic Object Pose/Velocity/Acceleration
$\mathbf{p}_D/\dot{\mathbf{p}}_D/\ddot{\mathbf{p}}_D$	$\mathbf{p}/\dot{\mathbf{p}}/\ddot{\mathbf{p}}$	Dynamic Object Position/Velocity/Acceleration

If omitted, n is assumed to be 3 (3-D).

[326]. They have also been used as a representation for local perception and exploration uncertainty [327]. However, these works did not use LTV-SDE priors to obtain exact sparsity in GP regression.[7]

Mukadam et al. [329] were the first to apply exactly sparse TGPs to the problem of motion planning with the GPMP algorithm, in particular for a 7-DoF robotic arm, addressing the large number of states needed by discrete-time planners. This was extended to factor graph optimization, in GPMP2 [330] (later used by Maric et al. [200]), GPMP-GRAPH [331], and iGPMP2 [60], [330], before unification with state estimation in STEAP [197], [198], and learning from demonstration in CLAMP [199]. CLAMP is unique from other works in that the transition matrix $\mathbf{\Phi}(t_{i+1}, t_i)$ and bias $\mathbf{v}(t_{i+1}, t_i)$ are learned for particular tasks from trajectory demonstrations with linear ridge regression. GPMP2 was also extended to dGPMP2 [332], which through a self-supervised end-to-end training framework, allowed the learning of factor covariance parameters, such as $\mathbf{Q}_C$, an idea which could be applied to state estimation in future work. Inspired by these works, Cheng et al. [333] adopted an exactly sparse GP for path planning in an autonomous driving context, parametrizing their WNOJ prior by arc length instead of time. Lilge et al. [334], [335] similarly parametrized by arc length to model the shape of a continuum robot using a white-noise-on-strain-rate (analogous to WNOA) LTV-SDE prior derived from the Cosserat rod model, with state consisting of $SE(3)$ pose and $\mathbb{R}^6$ generalized strain. They recently explore including both temporal and spatial domains jointly [336].

[7] Although, some works [319] attain sparsity in the covariance matrix by using a piecewise kernel that is zero beyond a certain distance [328].

VII. CONCLUSION

This work examined the literature and methods of continuous-time state estimation for robotics. Interpolation and integration techniques remain popular CT methods, especially linear interpolation due to its simplicity and low computation cost. This work has focused more heavily on temporal splines and Gaussian process, which can represent complex process dynamics while remaining computationally competitive. By modeling state evolution explicitly, rather than capturing "snapshots" at measurement times, CT methods offer significant advantages over DT estimation, both for the performance, flexibility, and generalizability of the estimator itself and for downstream planning or control algorithms. By providing introductory theory, a comprehensive survey, and a discussion of open problems and applications in other domains, this work aims to support future CT research, which will become increasingly essential to the field as the number, variety, and complexity of sensors grow in robotics applications.

REFERENCES

[1] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2013, pp. 3923–3929.

[2] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The two-state implicit filter recursive estimation for mobile robots," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 573–580, Jan. 2018.

[3] J. Nubert, S. Khattak, and M. Hutter, "Graph-based multi-sensor fusion for consistent localization of autonomous construction robots," in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 10048–10054.

[4] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst. Conf.*, Berkeley, CA, 2014, pp. 1–9.

[5] J. Zhang and S. Singh, "Low-drift and real-time LiDAR odometry and mapping," *Auton. Robots*, vol. 41, pp. 401–416, 2017.

[6] Q. Li et al., "LO-Net: Deep real-time LiDAR odometry," in *Proc. 2019 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8473–8482.

[7] J. Nubert, S. Khattak, and M. Hutter, "Self-supervised learning of LiDAR odometry for robotic applications," in *Proc. 2021 Int. Conf. Robot. Autom.*, IEEE, 2021, pp. 9601–9607.

[8] T. Tuna, J. Nubert, Y. Nava, S. Khattak, and M. Hutter, "X-ICP: Localizability-aware LiDAR registration for robust localization in extreme environments," *IEEE Trans. Robot.*, vol. 40, pp. 452–471, 2023.

[9] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.

[10] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proc. Robot.: Sci. Syst. Conf.*, 2015, pp. 1–20.

[11] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, Feb. 2017.

[12] M. Brossard, A. Barrau, P. Chauchat, and S. Bonnabel, "Associating uncertainty to extended poses for on lie group IMU preintegration with rotating Earth," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 998–1015, Apr. 2022.

[13] C. L. Gentil, T. Vidal-Calleja, and S. Huang, "3D LiDAR-IMU calibration based on upsampled preintegrated measurements for motion

distortion correction," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 2149–2155.
[14] C. L. Gentil, T. Vidal-Calleja, and S. Huang, "Gaussian process preintegration for inertial-aided state estimation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2108–2114, Apr. 2020.
[15] T. Wu, H. Fu, B. Liu, H. Xue, R. Ren, and Z. Tu, "Detailed analysis on generating the range image for LiDAR point cloud processing," *Electronics*, vol. 10, no. 11, 2021, Art. no. 1224.
[16] S. P. Nicklin, R. D. Fisher, and R. H. Middleton, "Rolling shutter image compensation," in *RoboCup 2006: Robot Soccer World Cup X*. Berlin, Germany: Springer, 2007, pp. 402–409.
[17] R. Bähnemann et al., "Under the sand: Navigation and localization of a micro aerial vehicle for landmine detection with ground penetrating synthetic aperture radar," *Field Robot.*, vol. 2, pp. 1028–1067, 2022.
[18] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2013, pp. 1280–1286.
[19] G. Cioffi, T. Cieslewski, and D. Scaramuzza, "Continuous-time vs. discrete-time vision-based SLAM: A comparative study," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2399–2406, Apr. 2022.
[20] P. Furgale, C. H. Tong, T. D. Barfoot, and G. Sibley, "Continuous-time batch trajectory estimation using temporal basis functions," *Int. J. Robot. Res.*, vol. 34, no. 14, pp. 1688–1710, 2015.
[21] D. Lee, M. Jung, W. Yang, and A. Kim, "LiDAR odometry survey: Recent advancements and remaining challenges," *Intell. Serv. Robot.*, vol. 17, no. 2, pp. 95–118, 2024.
[22] J. Johnson, J. Mangelson, T. Barfoot, and R. Beard, "Continuous-time trajectory estimation: A comparative study between Gaussian process and spline-based approaches," 2024, *arXiv:2402.00399*.
[23] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intell. Transp. Syst. Mag.*, vol. 2, no. 4, pp. 31–43, 2011.
[24] T. D. Barfoot, *State Estimation for Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[25] F. Dellaert et al., "Factor graphs for robot perception," *Found. Trends Robot.*, vol. 6, no. 1/2, pp. 1–139, 2017.
[26] F. Dellaert, "Factor graphs: Exploiting structure in robotics," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 4, pp. 141–166, 2021.
[27] R. Gilmore, *Lie Groups, Lie Algebras, and Some of Their Applications*. Chelmsford, MA, USA: Courier Corporation, 2006.
[28] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, vol. 2. Berlin, Germany: Springer Sci. & Bus. Media, 2011.
[29] J. Sola, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," 2018, *arXiv:1812.01537*.
[30] X. Zheng and J. Zhu, "Traj-LO: In defense of LiDAR-only odometry using an effective continuous-time trajectory," *IEEE Robot. Autom. Lett.*, vol. 9, no. 2, pp. 1961–1968, Feb. 2024.
[31] D. Hug and M. Chli, "HyperSLAM: A generic and modular approach to sensor fusion and simultaneous localization and mapping in continuous-time," in *Proc. 2020 Int. Conf. 3D Vis.*, IEEE, 2020, pp. 978–986.
[32] P. S. Maybeck, *Stochastic Models, Estimation, and Control*. Cambridge, MA, USA: Academic Press, 1982.
[33] R. F. Stengel, *Optimal Control and Estimation*. Chelmsford, MA, USA: Courier Corporation, 1994.
[34] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch continuous-time trajectory estimation as exactly sparse Gaussian process regression," in *Proc. Robot.: Sci. Syst. Conf.*, Citeseer, 2014, pp. 1–10.
[35] S. Anderson, T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression," *Auton. Robots*, vol. 39, pp. 221–238, 2015.
[36] S. W. Anderson, "Batch continuous-time trajectory estimation," Ph.D. dissertation, Univ. Toronto, Toronto, ON, Canada, 2017. [Online]. Available: https://www.proquest.com/dissertations-theses/batch-continuous-timetrajectory-estimation/docview/1937946759/se-2?accountid=27229
[37] S. Thrun, "Probabilistic robotics," *Commun. ACM*, vol. 45, no. 3, pp. 52–57, 2002.
[38] S. Särkkä and L. Svensson, *Bayesian Filtering and Smoothing*, vol. 17. Cambridge, U.K.: Cambridge Univ. Press, 2023.
[39] D. M. Rosen, G. Huang, and J. J. Leonard, "Inference over heterogeneous finite-/infinite-dimensional systems using factor graphs and Gaussian processes," in *Proc. 2014 Int. Conf. Robot. Autom.*, IEEE, 2014, pp. 1261–1268.
[40] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME–J. Basic Eng.*, vol. 82, pp. 35–45, 1960.
[41] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI*, vol. 3068. Bellingham, WA, USA: SPIE, 1997, pp. 182–193.
[42] E. A. Wan and R. V. D. Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. IEEE 2000 Adaptive Syst. Signal Process. Commun. Control Symp.*, IEEE, 2000, pp. 153–158.
[43] J. Quinonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, 2005.
[44] J. Ko and D. Fox, "GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models," *Auton. Robots*, vol. 27, pp. 75–90, 2009.
[45] K. Li, "On embedding B-splines in recursive state estimation," *IEEE Trans. Aerosp. Electron. Syst.*, 2025.
[46] H. Shen et al., "CTE-MLO: Continuous-time and efficient multi-LiDAR odometry with localizability-aware point cloud sampling," *IEEE Trans. Field Robot.*, vol. 2, pp. 165–187, 2025.
[47] M. Persson, G. Häger, H. Ovrén, and P.-E. Forssén, "Practical pose trajectory splines with explicit regularization," in *Proc. 2021 Int. Conf. 3D Vis.*, IEEE, 2021, pp. 156–165.
[48] A. Haarbach, T. Birdal, and S. Ilic, "Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation," in *Proc. 2018 Int. Conf. 3D Vis.*, IEEE, 2018, pp. 381–389.
[49] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian process Gauss–Newton for non-parametric simultaneous localization and mapping," *Int. J. Robot. Res.*, vol. 32, no. 5, pp. 507–525, 2013.
[50] S. Anderson, F. Dellaert, and T. D. Barfoot, "A hierarchical wavelet decomposition for continuous-time SLAM," in *Proc. 2014 Int. Conf. Robot. Autom.*, IEEE, 2014, pp. 373–380.
[51] C. D. Boor, "On calculating with B-splines," *J. Approximation Theory*, vol. 6, no. 1, pp. 50–62, 1972.
[52] C. D. Boor, *A Practical Guide to Splines*, vol. 27. New York, NY, USA: Springer-Verlag, 1978.
[53] P. Crouch, G. Kun, and F. S. Leite, "The De Casteljau algorithm on lie groups and spheres," *J. Dynamical Control Syst.*, vol. 5, no. 3, pp. 397–429, 1999.
[54] K. Qin, "General matrix representations for B-splines," in *Proc. 6th Pacific Conf. Comput. Graph. Appl.*, IEEE, 1998, pp. 37–43.
[55] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative B-splines on lie groups," in *Proc. 2020 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11148–11156.
[56] J. Quenzel and S. Behnke, "Real-time multi-adaptive-resolution-surfel 6D LiDAR odometry using continuous-time trajectory optimization," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2021, pp. 5499–5506.
[57] J. Tirado and J. Civera, "Jacobian computation for cumulative B-splines on SE(3) and application to continuous-time object tracking," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7132–7139, Jul. 2022.
[58] K. Li, Z. Cao, and U. D. Hanebeck, "Continuous-time ultra-wideband-inertial fusion," *IEEE Robot. Autom. Lett.*, vol. 8, no. 7, pp. 4338–4345, Jul. 2023.
[59] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian process Gauss-Newton: Non-parametric state estimation," in *Proc. 2012 Conf. Comput. Robot Vis.*, IEEE, 2012, pp. 206–213.
[60] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time gaussian process motion planning via probabilistic inference," *Int. J. Robot. Res.*, vol. 37, no. 11, pp. 1319–1340, 2018.
[61] S. Anderson and T. D. Barfoot, "Full steam ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE(3)," in *Proc. 2015 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2015, pp. 157–164.
[62] J. Dong, B. Boots, and F. Dellaert, "Sparse Gaussian processes for continuous-time trajectory estimation on matrix lie groups," 2017, *arXiv:1705.06020*.
[63] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse Gaussian processes on matrix lie groups: A unified framework for optimizing continuous-time trajectories," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 6497–6504.

[64] T. Y. Tang, D. J. Yoon, and T. D. Barfoot, "A white-noise-on-jerk motion prior for continuous-time trajectory estimation on SE(3)," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 594–601, Apr. 2019.

[65] T.-M. Nguyen et al., "A third-order gaussian process trajectory representation framework with closed-form kinematics for continuous-time motion estimation," 2025, *arXiv:2410.22931*.

[66] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," in *Proc. 2009 Int. Conf. Robot. Autom.*, IEEE, 2009, pp. 4312–4319.

[67] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-D range sensor with application to mobile mapping," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1104–1119, Oct. 2012.

[68] S. Hong, H. Ko, and J. Kim, "VICP: Velocity updating iterative closest point algorithm," in *Proc. 2010 Int. Conf. Robot. Autom.*, IEEE, 2010, pp. 1893–1898.

[69] F. Moosmann and C. Stiller, "Velodyne SLAM," in *Proc. 2011 IEEE Intell. Veh. Symp. (IV)*, IEEE, 2011, pp. 393–398.

[70] H. Dong and T. D. Barfoot, "Lighting-invariant visual odometry using LiDAR intensity imagery and pose interpolation," in *Field and Service Robotics*. Berlin, Germany: Springer, 2013, pp. 327–342.

[71] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 2480–2485.

[72] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV," in *Proc. 2020 Int. Conf. Robot. Autom.*, IEEE, 2020, pp. 3126–3131.

[73] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "KISS-ICP: In defense of point-to-point ICP–simple, accurate, and robust registration if done the right way," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 1029–1036, Feb. 2023.

[74] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D LiDAR inertial odometry and mapping," in *Proc. 2019 Int. Conf. Robot. Autom.*, IEEE, 2019, pp. 3144–3150.

[75] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2020, pp. 5135–5142.

[76] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

[77] T.-M. Nguyen, D. Duberg, P. Jensfelt, S. Yuan, and L. Xie, "SLICT: Multi-input multi-scale surfel-based LiDAR-inertial continuous-time odometry and mapping," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2102–2109, Apr. 2023.

[78] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled LiDAR-visual-inertial odometry via smoothing and mapping," in *Proc. 2021 Int. Conf. Robot. Autom.*, IEEE, 2021, pp. 5692–5698.

[79] D. Droeschel, M. Schwarz, and S. Behnke, "Continuous mapping and localization for autonomous navigation in rough terrain using a 3D laser scanner," *Robot. Auton. Syst.*, vol. 88, pp. 104–115, 2017.

[80] S. Ceriani, C. Sánchez, P. Taddei, E. Wolfart, and V. Sequeira, "Pose interpolation SLAM for large maps using moving 3D sensors," in *Proc. 2015 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2015, pp. 750–757.

[81] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[82] D. Vivet, P. Checchin, and R. Chapuis, "Localization and mapping using only a rotating FMCW radar sensor," *Sensors*, vol. 13, no. 4, pp. 4527–4552, 2013.

[83] K. Burnett, A. P. Schoellig, and T. D. Barfoot, "Do we need to compensate for motion distortion and doppler effects in spinning radar navigation?," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 771–778, Apr. 2021.

[84] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic LiDAR odometry with loop closure," in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 5580–5586.

[85] X. Zheng and J. Zhu, "ECTLO: Effective continuous-time odometry using range image for LiDAR with small FoV," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2023, pp. 9102–9109.

[86] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic LiDAR fusion: Dense map-centric continuous-time SLAM," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 1206–1213.

[87] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity meets continuous-time: Map-centric dense 3D LiDAR SLAM," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 978–997, Apr. 2022.

[88] M. Ramezani et al., "Wildcat: Online continuous-time 3D LiDAR-inertial SLAM," 2022, *arXiv:2205.12595*.

[89] C. Park, P. Moghadam, S. Kim, S. Sridharan, and C. Fookes, "Spatiotemporal camera-LiDAR calibration: A targetless and structureless approach," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1556–1563, Apr. 2020.

[90] G. Klein and D. Murray, "Parallel tracking and mapping on a camera phone," in *Proc. IEEE 8th Int. Symp. Mixed Augmented Reality*, IEEE, 2009, pp. 83–86.

[91] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 2007 IEEE/ACM Int. Symp. Mixed Augmented Reality*, IEEE, 2007, pp. 225–234.

[92] S. Anderson and T. D. Barfoot, "RANSAC for motion-distorted 3D visual sensors," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2013, pp. 2093–2099.

[93] O. Ait-Aider and F. Berry, "Structure and kinematics triangulation with a rolling shutter stereo rig," in *Proc. 2009 Int. Conf. Comput. Vis.*, IEEE, 2009, pp. 1835–1840.

[94] P.-E. Forssén and E. Ringaby, "Rectifying rolling shutter video from hand-held devices," in *Proc. 2010 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, IEEE, 2010, pp. 507–514.

[95] E. Ringaby and P.-E. Forssén, "Scan rectification for structured light range sensors with rolling shutters," in *Proc. 2011 Int. Conf. Comput. Vis.*, IEEE, 2011, pp. 1575–1582.

[96] C. Jia and B. L. Evans, "Probabilistic 3-D motion estimation for rolling shutter video rectification from visual and inertial measurements," in *Proc. IEEE 14th Int. Workshop Multimedia Signal Process. (MMSP)*, IEEE, 2012, pp. 203–208.

[97] J. Hedborg, P.-E. Forssén, M. Felsberg, and E. Ringaby, "Rolling shutter bundle adjustment," in *Proc. 2012 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, IEEE, 2012, pp. 1434–1441.

[98] C. X. Guo, D. G. Kottas, R. DuToit, A. Ahmed, R. Li, and S. I. Roumeliotis, "Efficient visual-inertial navigation using a rolling-shutter camera with inaccurate timestamps," in *Proc. Robot.: Sci. Syst. Conf.*, 2014, pp. 1–9.

[99] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," Stanford Univ., Stanford, CA, USA, *Comput. Sci. Tech Rep.* CTSR 2011–3, 2011.

[100] P. Geneva, K. Eckenhoff, and G. Huang, "Asynchronous multi-sensor fusion for 3D mapping and localization," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 5994–5999.

[101] K. Eckenhoff, P. Geneva, J. Bloecker, and G. Huang, "Multi-camera visual-inertial navigation with online intrinsic and extrinsic calibration," in *Proc. 2019 Int. Conf. Robot. Autom.*, IEEE, 2019, pp. 3158–3164.

[102] R. Zlot and M. Bosse, "Efficient large-scale 3D mobile mapping and surface reconstruction of an underground mine," in *Field and Service Robotics*. Berlin, Germany: Springer, 2013, pp. 479–493.

[103] R. Zlot and M. Bosse, "Efficient large-scale three-dimensional mobile mapping for underground mines," *J. Field Robot.*, vol. 31, no. 5, pp. 758–779, 2014.

[104] R. Dubé, H. Sommer, A. Gawel, M. Bosse, and R. Siegwart, "Non-uniform sampling strategies for continuous correction based trajectory estimation," in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 4792–4798.

[105] M. Li, B. H. Kim, and A. I. Mourikis, "Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera," in *Proc. 2013 Int. Conf. Robot. Autom.*, IEEE, 2013, pp. 4712–4719.

[106] X. Wang, F. Xue, Z. Yan, W. Dong, Q. Wang, and H. Zha, "Continuous-time stereo visual odometry based on dynamics model," in *Proc. 2018 Asian Conf. Comput. Vis.*, Springer, 2018, pp. 388–403.

[107] T. Lowe, S. Kim, and M. Cox, "Complementary perception for handheld SLAM," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1104–1111, Apr. 2018.

[108] Y. Wang and H. Ma, "Online spatial and temporal initialization for a monocular visual-inertial-LiDAR system," *Sensors*, vol. 22, no. 2, pp. 1609–1620, 2021.

[109] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[110] Y. Wang and H. Ma, "mVIL-fusion: Monocular visual-inertial-LiDAR simultaneous localization and mapping in challenging environments," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 504–511, Feb. 2023.

[111] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-LIO: Robust high-bandwidth light detection and ranging inertial odometry," *Adv. Intell. Syst.*, vol. 5, no. 7, 2023, Art. no. 2200459.

[112] K. Chen, R. Nemiroff, and B. T. Lopez, "Direct LiDAR-inertial odometry: Lightweight LIO with continuous-time motion correction," in *Proc. 2023 Int. Conf. Robot. Autom.*, IEEE, 2023, pp. 3983–3989.

[113] G. Terzakis and M. Lourakis, "Efficient pose prediction with rational regression applied to vSLAM," in *Proc. 2024 Int. Conf. Robot. Autom.*, IEEE, 2024, pp. 11970–11976.

[114] M. Zhu and Y. Wu, "ChevOpt: Continuous-time state estimation by Chebyshev polynomial optimization," *IEEE Trans. Signal Process.*, vol. 70, pp. 3136–3147, 2022.

[115] M. Zhu and Y. Wu, "Inertial-based navigation by polynomial optimization: Inertial-magnetic attitude estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, no. 6, pp. 7772–7783, Dec. 2023.

[116] V. Agrawal and F. Dellaert, "Continuous-time state & dynamics estimation using a pseudo-spectral parameterization," in *Proc. 2021 Int. Conf. Robot. Autom.*, IEEE, 2021, pp. 426–432.

[117] J. D. Hol, T. B. Schön, and F. Gustafsson, "Modeling and calibration of inertial and vision sensors," *Int. J. Robot. Res.*, vol. 29, no. 2/3, pp. 231–244, 2010.

[118] S. Shen, N. Michael, and V. Kumar, "Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs," in *Proc. 2015 Int. Conf. Robot. Autom.*, IEEE, 2015, pp. 5303–5310.

[119] T. Qin, P. Li, and S. Shen, "VINS-MONO: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[120] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1143–1156, Oct. 2008.

[121] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *Int. J. Robot. Res.*, vol. 36, no. 10, pp. 1053–1072, 2017.

[122] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. J. Robot. Res.*, vol. 30, no. 1, pp. 56–79, 2011.

[123] J. Kelly and G. S. Sukhatme, "A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors," in *Proc. 2010 Int. Symp. Exp. Robot.*, Springer, 2014, pp. 195–209.

[124] C. L. Gentil, T. Vidal-Calleja, and S. Huang, "IN2LAMA: Inertial LiDAR localisation and mapping," in *Proc. 2019 Int. Conf. Robot. Autom.*, IEEE, 2019, pp. 6388–6394.

[125] S. Särkkä, "Linear operators and stochastic partial differential equations in Gaussian process regression," in *Proc. 21st Int. Conf. Artif. Neural Netw.*, Espoo, Finland, Springer, 2011, pp. 151–158.

[126] C. L. Gentil, T. Vidal-Calleja, and S. Huang, "IN2LAAMA: Inertial LiDAR localization autocalibration and mapping," *IEEE Trans. Robot.*, vol. 37, no. 1, pp. 275–290, Feb. 2021.

[127] C. L. Gentil, F. Tschopp, I. Alzugaray, T. Vidal-Calleja, R. Siegwart, and J. Nieto, "IDOL: A framework for IMU-DVS odometry using lines," in *Proc. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2020, pp. 5863–5870.

[128] C. L. Gentil and T. Vidal-Calleja, "Continuous integration over so (3) for IMU preintegration," in *Proc. Robot.: Sci. Syst. Conf.*, 2021, pp. 1–10.

[129] B. Dai, C. L. Gentil, and T. Vidal-Calleja, "A tightly-coupled event-inertial odometry using exponential decay and linear preintegrated measurements," in *Proc. 2022 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2022, pp. 9475–9482.

[130] C. L. Gentil and T. Vidal-Calleja, "Continuous latent state preintegration for inertial-aided systems," *Int. J. Robot. Res.*, vol. 42, pp. 874–900, 2023.

[131] C. L. Gentil, R. Falque, and T. Vidal-Calleja, "Real-time truly-coupled LiDAR-inertial motion correction and spatiotemporal dynamic object detection," in *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2024, pp. 12565–12572.

[132] M.-J. Kim, M.-S. Kim, and S. Y. Shin, "AC/sup 2/-continuous B-spline quaternion curve interpolating a given sequence of solid orientations," in *Proc. Comput. Animation*, IEEE, 1995, pp. 72–81.

[133] M.-J. Kim, M.-S. Kim, and S. Y. Shin, "A general construction scheme for unit quaternion curves with simple high order derivatives," in *Proc. 22nd Annu. Conf. Comput. Graph. Interactive Techn.*, 1995, pp. 369–376.

[134] S. J. Gortler and M. F. Cohen, "Hierarchical and variational geometric modeling with wavelets," in *Proc. 1995 Symp. Interactive 3D Graph.*, 1995, pp. 35–ff.

[135] R. Anderson-Sprecher and R. V. Lenth, "Spline estimation of paths using bearings-only tracking data," *J. Amer. Statist. Assoc.*, vol. 91, no. 433, pp. 276–283, 1996.

[136] I. Kang and F. Park, "Cubic spline algorithms for orientation interpolation," *Int. J. Numer. Methods Eng.*, vol. 46, no. 1, pp. 45–64, 1999.

[137] S.-H. Jung and C. J. Taylor, "Camera trajectory estimation using inertial sensor measurements and structure from motion results," in *Proc. 2001 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, IEEE, 2001, pp. II–II.

[138] C. Bibby and I. Reid, "A hybrid SLAM representation for dynamic marine environments," in *Proc. 2010 Int. Conf. Robot. Autom.*, IEEE, 2010, pp. 257–264.

[139] M. Hadzagic and H. Michalska, "A Bayesian inference approach for batch trajectory estimation," in *Proc. Int. Conf. Inf. Fusion*, IEEE, 2011, pp. 1–8.

[140] M. Fleps, E. Mair, O. Ruepp, M. Suppa, and D. Burschka, "Optimization based IMU camera calibration," in *Proc. 2011 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2011, pp. 3297–3304.

[141] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proc. 2012 Int. Conf. Robot. Autom.*, IEEE, 2012, pp. 2088–2095.

[142] S. Lovegrove, A. Patron-Perez, and G. Sibley, "Spline fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras," in *Proc. Brit. Mach. Vis. Conf.*, 2013, pp. 1–12.

[143] S. Anderson and T. D. Barfoot, "Towards relative continuous-time SLAM," in *Proc. 2013 Int. Conf. Robot. Autom.*, IEEE, 2013, pp. 1033–1040.

[144] S. Anderson, K. MacTavish, and T. D. Barfoot, "Relative continuous-time SLAM," *Int. J. Robot. Res.*, vol. 34, no. 12, pp. 1453–1479, 2015.

[145] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proc. 2013 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1360–1367.

[146] M. Sheehan, A. Harrison, and P. Newman, "Continuous vehicle localisation using sparse 3D sensing, kernelised Rényi distance and fast Gauss transforms," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2013, pp. 398–405.

[147] J. Rehder, P. Beardsley, R. Siegwart, and P. Furgale, "Spatio-temporal laser to visual/inertial calibration with applications to hand-held, large scale scanning," in *Proc. 2014 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2014, pp. 459–465.

[148] H. Alismail, L. D. Baker, and B. Browning, "Continuous trajectory estimation for 3D SLAM from actuated LiDAR," in *Proc. 2014 Int. Conf. Robot. Autom.*, IEEE, 2014, pp. 6096–6101.

[149] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *Int. J. Comput. Vis.*, vol. 113, no. 3, pp. 208–219, 2015.

[150] C. Kerl, J. Stuckler, and D. Cremers, "Dense continuous-time tracking and mapping with rolling shutter RGB-D cameras," in *Proc. 2015 Int. Conf. Comput. Vis.*, 2015, pp. 2264–2272.

[151] E. Mueggler, G. Gallego, and D. Scaramuzza, "Continuous-time trajectory estimation for event-based vision sensors," in *Proc. Robot.: Sci. Syst. Conf.*, 2015, pp. 1–9.

[152] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Trans. Robot.*, vol. 32, no. 2, pp. 383–398, Apr. 2016.

[153] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes," in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 4304–4311.

[154] J.-H. Kim, C. Cadena, and I. Reid, "Direct semi-dense SLAM for rolling shutter cameras," in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 1308–1315.

[155] H. Sommer, J. R. Forbes, R. Siegwart, and P. Furgale, "Continuous-time estimation of attitude using B-splines on lie groups," *J. Guid. Control Dyn.*, vol. 39, no. 2, pp. 242–261, 2016.

[156] T. Li, J. Prieto, and J. M. Corchado, "Fitting for smoothing: A methodology for continuous-time target track estimation," in *Proc. 2016 Int. Conf. Indoor Positioning Indoor Navigation (IPIN)*, IEEE, 2016, pp. 1–8.

[157] T. Li, J. M. Corchado, H. Chen, and J. Bajo, "Track a smoothly maneuvering target based on trajectory estimation," in *Proc. Int. Conf. Inf. Fusion*, IEEE, 2017, pp. 1–8.

[158] T. Li, H. Chen, S. Sun, and J. M. Corchado, "Joint smoothing and tracking based on continuous-time target trajectory function fitting," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1476–1483, Jul. 2019.

[159] J. Rehder and R. Siegwart, "Camera/IMU calibration revisited," *Sensors*, vol. 17, no. 11, pp. 3257–3268, 2017.

[160] B. Vandeportaele, P.-A. Gohard, M. Devy, and B. Coudrin, "Pose interpolation for rolling shutter cameras using non uniformly time-sampled

B-splines," in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, SCITEPRESS, 2017, pp. 286–293.
[161] D. Droeschel and S. Behnke, "Efficient continuous-time SLAM for 3D LiDAR-based online mapping," in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 5000–5007.
[162] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1425–1440, Dec. 2018.
[163] H. Ovrén and P.-E. Forssén, "Spline error weighting for robust visual-inertial fusion," in *Proc. 2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 321–329.
[164] H. Ovrén and P.-E. Forssén, "Trajectory representation and landmark projection for continuous-time structure from motion," *Int. J. Robot. Res.*, vol. 38, no. 6, pp. 686–701, 2019.
[165] M. Pacholska, F. Dümbgen, and A. Scholefield, "Relax and recover: Guaranteed range-only continuous localization," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2248–2255, Apr. 2020.
[166] J. Lv, J. Xu, K. Hu, Y. Liu, and X. Zuo, "Targetless calibration of LiDAR-IMU system based on continuous-time batch estimation," in *Proc. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2020, pp. 9968–9975.
[167] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, "CLINS: Continuous-time trajectory estimation for LiDAR-inertial system," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2021, pp. 6657–6663.
[168] Y. Z. Ng, B. Choi, R. Tan, and L. Heng, "Continuous-time radar-inertial odometry for automotive radars," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2021, pp. 323–330.
[169] A. J. Yang, C. Cui, I. A. Bârsan, R. Urtasun, and S. Wang, "Asynchronous multi-view SLAM," in *Proc. 2021 Int. Conf. Robot. Autom.*, IEEE, 2021, pp. 5669–5676.
[170] K. Huang, Y. Wang, and L. Kneip, "B-splines for purely vision-based localization and mapping on non-holonomic ground vehicles," in *Proc. 2021 Int. Conf. Robot. Autom.*, IEEE, 2021, pp. 5374–5380.
[171] D. Hug, P. Bänninger, I. Alzugaray, and M. Chli, "Continuous-time stereo-inertial odometry," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 6455–6462, Jul. 2022.
[172] J. Mo and J. Sattar, "Continuous-time spline visual-inertial odometry," in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 9492–9498.
[173] J. Huai, Y. Zhuang, Y. Lin, G. Jozkow, Q. Yuan, and D. Chen, "Continuous-time spatiotemporal calibration of a rolling shutter camera-IMU system," *Sensors*, vol. 22, no. 8, pp. 7920–7930, 2022.
[174] Y. Wang et al., "Visual odometry with an event camera using continuous ray warping and volumetric contrast maximization," *Sensors*, vol. 22, no. 15, 2022, Art. no. 5687.
[175] X. Lang, J. Lv, J. Huang, Y. Ma, Y. Liu, and X. Zuo, "Ctrl-VIO: Continuous-time visual-inertial odometry for rolling shutter cameras," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11537–11544, Oct. 2022.
[176] J. Lv, X. Zuo, K. Hu, J. Xu, G. Huang, and Y. Liu, "Observability-aware intrinsic and extrinsic calibration of LiDAR-IMU systems," *IEEE Trans. Robot.*, vol. 38, no. 6, pp. 3734–3753, Dec. 2022.
[177] J. Lv, X. Lang, J. Xu, M. Wang, Y. Liu, and X. Zuo, "Continuous-time fixed-lag smoothing for LiDAR-inertial-camera SLAM," *IEEE/ASME Trans. Mechatron.*, vol. 28, no. 4, pp. 2259–2270, Aug. 2023.
[178] X. Lang et al., "Coco-LIC: Continuous-time tightly-coupled LiDAR-inertial-camera odometry using non-uniform B-spline," *IEEE Robot. Autom. Lett.*, vol. 8, no. 11, pp. 7074–7081, Nov. 2023.
[179] X. Lang et al., "Gaussian-LIC: Real-time photo-realistic SLAM with Gaussian splatting and LiDAR-inertial-camera fusion," 2024, *arXiv:2404.06926*.
[180] M. Jung, S. Jung, and A. Kim, "Asynchronous multiple LiDAR-inertial odometry using point-wise inter-LiDAR uncertainty propagation," *IEEE Robot. Autom. Lett.*, vol. 8, no. 7, pp. 4211–4218, Jul. 2023.
[181] X. Lu, Y. Zhou, J. Niu, S. Zhong, and S. Shen, "Event-based visual inertial velometer," 2023, *arXiv:2311.18189*.
[182] T.-M. Nguyen et al., "Eigen is all you need: Efficient LiDAR-inertial continuous-time odometry with internal association," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 5330–5337, Jun. 2024.
[183] Y. Lv, Y. Zhang, X. Zhao, W. Li, J. Ning, and Y. Jin, "CTA-LO: Accurate and robust LiDAR odometry using continuous-time adaptive estimation," in *Proc. Int. Conf. Robot. Autom.*, IEEE, 2024, pp. 12034–12040.
[184] D. Hug, I. Alzugaray, and M. Chli, "Hyperion–A fast, versatile symbolic Gaussian belief propagation framework for continuous-time SLAM," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 215–231.
[185] S. Li, X. Li, S. Chen, Y. Zhou, and S. Wang, "Continuous-time LiDAR/IMU/radar odometry for accurate and smooth state estimation via tightly coupled integration," *IEEE Internet Things J.*, vol. 11, no. 24, pp. 40306–40318, Dec. 2024.
[186] T.-M. Nguyen et al., "MCD: Diverse large-scale multi-campus dataset for robot perception," in *Proc. 2024 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 22304–22313.
[187] J. Li, S. Yuan, M. Cao, T.-M. Nguyen, K. Cao, and L. Xie, "HCTO: Optimality-aware LiDAR inertial odometry with hybrid continuous time optimization for compact wearable mapping system," *ISPRS J. Photogrammetry Remote Sens.*, vol. 211, pp. 228–243, 2024.
[188] I. J. Schoenberg, "Contributions to the problem of approximation of equidistant data by analytic functions: Part A.–On the problem of smoothing or graduation. A first class of analytic approximation formulae," in *I. J. Schoenberg Selected Papers*. Berlin, Germany: Springer, 1988, pp. 3–57.
[189] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. San Mateo, CA, USA: Morgan Kaufmann, 1995.
[190] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL, USA: CRC Press, 2017.
[191] H. Martiros et al., "SymForce: Symbolic computation and code generation for robotics," 2022, *arXiv:2204.07889*.
[192] S. Agarwal and K. Mierle, and The Ceres Solver Team, "Ceres solver," Oct. 2023. [Online]. Available: https://github.com/ceres-solver/ceres-solver
[193] C. H. Tong, S. Anderson, H. Dong, and T. D. Barfoot, "Pose interpolation for laser-based visual odometry," *J. Field Robot.*, vol. 31, no. 5, pp. 731–757, 2014.
[194] X. Yan, V. Indelman, and B. Boots, "Incremental sparse Gaussian process regression for continuous-time trajectory estimation & mapping," *Measurements*, vol. 1, 2014, Art. no. 2.
[195] X. Yan, V. Indelman, and B. Boots, "Incremental sparse GP regression for continuous-time trajectory estimation and mapping," *Robot. Auton. Syst.*, vol. 87, pp. 120–132, 2017.
[196] J. Dong, J. G. Burnham, B. Boots, G. Rains, and F. Dellaert, "4D crop monitoring: Spatio-temporal reconstruction for agriculture," in *Proc. 2017 Int. Conf. Robot. Autom.*, IEEE, 2017, pp. 3878–3885.
[197] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "Simultaneous trajectory estimation and planning via probabilistic inference," in *Proc. Robot.: Sci. Syst. Conf.*, 2017, pp. 1–9.
[198] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "STEAP: Simultaneous trajectory estimation and planning," *Auton. Robots*, vol. 43, pp. 415–434, 2019.
[199] M. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots, "Towards robust skill generalization: Unifying learning from demonstration and motion planning," in *Proc. Intell. Robots Syst.*, 2018, pp. 109–118.
[200] F. Maric, O. Limoyo, L. Petrovic, I. Petrovic, and J. Kelly, "Singularity avoidance as manipulability maximization using continuous time Gaussian processes," 2018, *arXiv:1803.09493*.
[201] M. Warren, M. Paton, K. MacTavish, A. P. Schoellig, and T. D. Barfoot, "Towards visual teach and repeat for GPS-denied flight of a fixed-wing UAV," in *Field and Service Robotics*, Berlin, Germany: Springer, 2018, pp. 481–498.
[202] M. Warren, M. Greeff, B. Patel, J. Collier, A. P. Schoellig, and T. D. Barfoot, "There's no place like home: Visual teach and repeat for emergency return of multirotor UAVs during GPS failure," *IEEE Robot. Autom. Lett.*, vol. 4, no. 1, pp. 161–168, Jan. 2019.
[203] K. Judd, "Unifying motion segmentation, estimation, and tracking for complex dynamic scenes," Ph.D. dissertation, Eng. Sci., Univ. Oxford, Oxford, U.K., 2019.
[204] K. M. Judd and J. D. Gammell, "Occlusion-robust MVO: Multimotion estimation through occlusion via motion closure," in *Proc. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2020, pp. 5855–5862.
[205] T. D. Barfoot, J. R. Forbes, and D. J. Yoon, "Exactly sparse Gaussian variational inference with application to derivative-free batch nonlinear state estimation," *Int. J. Robot. Res.*, vol. 39, no. 13, pp. 1473–1502, 2020.
[206] J. N. Wong, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "Variational inference with parameter learning applied to vehicle trajectory estimation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5291–5298, Oct. 2020.
[207] J. N. Wong, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "A data-driven motion prior for continuous-time trajectory estimation on SE(3)," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1429–1436, Apr. 2020.
[208] J. Peršić, L. Petrović, I. Marković, and I. Petrović, "Spatiotemporal multisensor calibration via Gaussian processes moving target tracking," *IEEE Trans. Robot.*, vol. 37, no. 5, pp. 1401–1415, Oct. 2021.

[209] Y. Kapushev, A. Kishkun, G. Ferrer, and E. Burnaev, “Random fourier features based SLAM,” in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2021, pp. 6597–6602.

[210] K. M. Judd and J. D. Gammell, “Multimotion visual odometry (MVO),” 2021, *arXiv:2110.15169*.

[211] K. M. Judd and J. D. Gammell, “Multimotion visual odometry,” *Int. J. Robot. Res.*, vol. 43, pp. 1250–1278, 2024.

[212] S. Li, L. Wang, J. Li, B. Tian, L. Chen, and G. Li, “3D LiDAR/IMU calibration based on continuous-time trajectory estimation in structured environments,” *IEEE Access*, vol. 9, pp. 138803–138816, 2021.

[213] H. Zhang, F. Widmayer, L. Lünnemann, and D. Abel, “OnlineFGO: Online continuous-time factor graph optimization with time-centric multi-sensor fusion for robust localization in large-scale environments,” 2022, *arXiv:2211.05400*.

[214] H. Zhang, X. Xia, M. Nitsch, and D. Abel, “Continuous-time factor graph optimization for trajectory smoothness of GNSS/INS navigation in temporarily GNSS-denied environments,” *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9115–9122, Oct. 2022.

[215] Y. Wu et al., “Picking up speed: Continuous-time LiDAR-only odometry using doppler velocity measurements,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 264–271, Jan. 2023.

[216] K. Burnett, Y. Wu, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, “Are we ready for radar to replace LiDAR in all-weather mapping and localization?,” *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10328–10335, Oct. 2022.

[217] D. Liu, A. Parra, Y. Latif, B. Chen, T.-J. Chin, and I. Reid, “Asynchronous optimisation for event-based visual odometry,” in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 9432–9438.

[218] J. Wang and J. D. Gammell, “Event-based stereo visual odometry with native temporal resolution via continuous-time Gaussian process regression,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6707–6714, Oct. 2023.

[219] D. J. Yoon et al., “Need for speed: Fast correspondence-free LiDAR-inertial odometry using doppler velocity,” in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2023, pp. 5304–5310.

[220] A. Goudar, T. D. Barfoot, and A. P. Schoellig, “Continuous-time range-only pose estimation,” in *Proc. 2023 Conf. Comput. Robot Vis.*, IEEE, 2023, pp. 29–36.

[221] C. L. Gentil, I. Alzugaray, and T. Vidal-Calleja, “Continuous-time Gaussian process motion-compensation for event-vision pattern tracking with distance fields,” in *Proc. 2023 Int. Conf. Robot. Autom.*, IEEE, 2023, pp. 804–812.

[222] C. L. Gentil and T. V. Calleja, “A Gaussian process approach for IMU to pose spatiotemporal calibration,” in *Proc. 2023 Australas. Conf. Robot. Autom.*, vol. 10, no. 11, 2023, Art. no. 1224.

[223] X. Zheng and J. Zhu, “Traj-LIO: A resilient multi-LiDAR multi-IMU state estimator through sparse Gaussian process,” 2024, *arXiv:2402.09189*.

[224] D. Lisus, K. Burnett, D. J. Yoon, R. Poulton, J. Marshall, and T. D. Barfoot, “Are doppler velocity measurements useful for spinning radar odometry?” *IEEE Robot. Autom. Lett.*, vol. 10, no. 1, pp. 224–231, Jan. 2025.

[225] K. Burnett, A. P. Schoellig, and T. D. Barfoot, “Continuous-time radar-inertial and LiDAR-inertial odometry using a Gaussian process motion prior,” *IEEE Trans. Robot.*, vol. 41, pp. 1059–1076, 2025.

[226] T. D. Barfoot, C. Holmes, and F. Dümbgen, “Certifiably optimal rotation and pose estimation based on the cayley map,” *Int. J. Robot. Res.*, vol. 44, no. 3, pp. 366–387 2025.

[227] H. Zhang, C.-C. Chen, H. Vallery, and T. D. Barfoot, “GNSS/multi-sensor fusion using continuous-time factor graph optimization for robust localization,” *IEEE Trans. Robot.*, vol. 40, pp. 4003–4023, 2024.

[228] K. Burnett, A. P. Schoellig, and T. D. Barfoot, “IMU as an input vs. a measurement of the state in inertial-aided state estimation,” 2024, *arXiv:2403.05968*.

[229] C. H. Tong, “Laser-based 3D mapping and navigation in planetary worksite environments,” *Ph.D. dissertation*, Univ. Toronto, Toronto, ON, Canada, 2013.

[230] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “iSAM2: Incremental smoothing and mapping using the bayes tree,” *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[231] C. Holmes and T. D. Barfoot, “An efficient global optimality certificate for landmark-based SLAM,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1539–1546, Mar. 2023.

[232] A. Goudar, F. Dümbgen, T. D. Barfoot, and A. P. Schoellig, “Optimal initialization strategies for range-only trajectory estimation,” *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2160–2167, Mar. 2024.

[233] H. Yang, J. Shi, and L. Carlone, “TEASER: Fast and certifiable point cloud registration,” *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 314–333, Apr. 2021.

[234] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, “SE-sync: A certifiably correct algorithm for synchronization over the special Euclidean group,” *Int. J. Robot. Res.*, vol. 38, no. 2/3, pp. 95–125, 2019.

[235] L. Piegl and W. Tiller, *The NURBS Book*. Berlin, Germany: Springer Sci. & Bus. Media, 2012.

[236] L. Carlone, G. C. Calafiore, C. Tommolillo, and F. Dellaert, “Planar pose graph optimization: Duality, optimal solutions, and verification,” *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 545–565, Jun. 2016.

[237] F. Dümbgen, C. Holmes, and T. D. Barfoot, “Safe and smooth: Certified continuous-time range-only localization,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 1117–1124, Feb. 2023.

[238] G. Cioffi, L. Bauersfeld, E. Kaufmann, and D. Scaramuzza, “Learned inertial odometry for autonomous drone racing,” *IEEE Robot. Autom. Lett.*, vol. 8, no. 5, pp. 2684–2691, May 2023.

[239] G. Cioffi, L. Bauersfeld, and D. Scaramuzza, “HDVIO: Improving localization and disturbance estimation with hybrid dynamics VIO,” 2023, *arXiv:2306.11429*.

[240] J. Hartikainen and S. Särkkä, “Kalman filtering and smoothing solutions to temporal Gaussian process regression models,” in *Proc. 2010 Int. Workshop Mach. Learn. Signal Process.*, IEEE, 2010, pp. 379–384.

[241] J. Hartikainen and S. Sarkka, “Sequential inference for latent force models,” 2012, *arXiv:1202.3730*.

[242] Z. Zhang and D. Scaramuzza, “Rethinking trajectory evaluation for SLAM: A probabilistic, continuous-time approach,” 2019, *arXiv: 1906.03996*.

[243] M. Elbanhawi, M. Simic, and R. Jazar, “Randomized bidirectional B-spline parameterization motion planning,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 406–419, Feb. 2016.

[244] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, “Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots,” *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2261–2268, Jul. 2018.

[245] M. Bjelonic, P. K. Sankar, C. D. Bellicoso, H. Vallery, and M. Hutter, “Rolling in the deep–hybrid locomotion for wheeled-legged robots using online trajectory optimization,” *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3626–3633, Apr. 2020.

[246] S. Chand and K. L. Doty, “On-line polynomial trajectories for robot manipulators,” *Int. J. Robot. Res.*, vol. 4, no. 2, pp. 38–48, 1985.

[247] A. Piazzi and A. Visioli, “Global minimum-jerk trajectory planning of robot manipulators,” *IEEE Trans. Ind. Electron.*, vol. 47, no. 1, pp. 140–149, Feb. 2000.

[248] M. J. V. Nieuwstadt and R. M. Murray, “Real-time trajectory generation for differentially flat systems,” *Int. J. Robust Nonlinear Control*, vol. 8, no. 11, pp. 995–1020, 1998.

[249] D. Mellinger and V. Kumar, “Minimum snap trajectory generation and control for quadrotors,” in *Proc. 2011 Int. Conf. Robot. Autom.*, IEEE, 2011, pp. 2520–2525.

[250] O. K. Sahingoz, “Generation of Bézier curve-based flyable trajectories for multi-UAV systems with parallel genetic algorithm,” *J. Intell. Robot. Syst.*, vol. 74, no. 1/2, pp. 499–511, 2014.

[251] W. Ding, W. Gao, K. Wang, and S. Shen, “Trajectory replanning for quadrotors using kinodynamic search and elastic optimization,” in *Proc. 2018 Int. Conf. Robot. Autom.*, IEEE, 2018, pp. 7595–7602.

[252] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, “Robust and efficient quadrotor trajectory generation for fast autonomous flight,” *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3529–3536, Oct. 2019.

[253] J. Tordesillas, B. T. Lopez, and J. P. How, “FASTER: Fast and safe trajectory planner for flights in unknown environments,” in *Proc. 2019 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2019, pp. 1934–1940.

[254] J. Tordesillas and J. P. How, “MADER: Trajectory planner in multiagent and dynamic environments,” *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 463–476, Feb. 2022.

[255] J. Tordesillas and J. P. How, “PANTHER: Perception-aware trajectory planner in dynamic environments,” *IEEE Access*, vol. 10, pp. 22662–22677, 2022.

[256] J. A. Preiss, K. Hausman, G. S. Sukhatme, and S. Weiss, “Trajectory optimization for self-calibration and navigation,” in *Proc. Robot.: Sci. Syst. Conf.*, 2017, pp. 1–9.

[257] S. Liu et al., “Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments,” *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1688–1695, Jul. 2017.

[258] L. Tang, H. Wang, P. Li, and Y. Wang, “Real-time trajectory generation for quadrotors using B-spline based non-uniform kinodynamic search,” in *Proc. 2019 IEEE Int. Conf. Robot. Biomimetics*, IEEE, 2019, pp. 1133–1138.

[259] J. Tordesillas and J. P. How, “MINVO basis: Finding simplexes with minimum volume enclosing polynomial curves,” *Comput.-Aided Des.*, vol. 151, 2022, Art. no. 103341.

[260] G. Herron, “Polynomial bases for quadratic and cubic polynomials which yield control points with small convex hulls,” *Comput. Aided Geometric Des.*, vol. 6, no. 1, pp. 1–9, 1989.

[261] E. J. Lobaton, J. Fu, L. G. Torres, and R. Alterovitz, “Continuous shape estimation of continuum robots using X-ray images,” in *Proc. 2013 Int. Conf. Robot. Autom.*, IEEE, 2013, pp. 725–732.

[262] B. Kim, J. Ha, F. C. Park, and P. E. Dupont, “Optimizing curvature sensor placement for fast, accurate shape sensing of continuum robots,” in *Proc. 2014 Int. Conf. Robot. Autom.*, IEEE, 2014, pp. 5374–5379.

[263] S. Song, Z. Li, H. Yu, and H. Ren, “Electromagnetic positioning for tip tracking and shape sensing of flexible robots,” *Sensors*, vol. 15, no. 8, pp. 4565–4575, 2015.

[264] Z. Qiao, Z. Yu, H. Yin, and S. Shen, “Online monocular lane mapping using Catmull-Rom spline,” in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2023, pp. 7179–7186.

[265] R. T. Rodrigues, A. P. Aguiar, and A. Pascoal, “A B-spline mapping framework for long-term autonomous operations,” in *Proc. 2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2018, pp. 3204–3209.

[266] L. Pedraza, D. Rodriguez-Losada, F. Matia, G. Dissanayake, and J. V. Miró, “Extending the limits of feature-based SLAM with B-splines,” *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 353–366, Apr. 2009.

[267] M. Liu, S. Huang, and G. Dissanayake, “A new observation model for B-spline SLAM,” in *Proc. 2009 Australas. Conf. Robot. Autom.*, 2009, pp. 1–8.

[268] M. Liu, S. Huang, G. Dissanayake, and S. Kodagoda, “Towards a consistent SLAM algorithm using B-splines to represent environments,” in *Proc. 2010 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2010, pp. 2065–2070.

[269] H. Seok and J. Lim, “Robust feature tracking in DVS event stream using Bézier mapping,” in *Proc. 2020 IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2020, pp. 1658–1667.

[270] L. C. Marsh and D. R. Cormier, *Spline Regression Models*. Newbury Park, CA, USA: Sage, 2001.

[271] L. M. Sangalli, J. O. Ramsay, and T. O. Ramsay, “Spatial spline regression models,” *J. Roy. Stat. Soc. Ser. B: Stat. Methodol.*, vol. 75, no. 4, pp. 681–703, 2013.

[272] D. Fakhoury, E. Fakhoury, and H. Speleers, “ExSpliNet: An interpretable and expressive spline-based neural network,” *Neural Netw.*, vol. 152, pp. 332–346, 2022.

[273] S. Zhu, L. Zhou, S. Pan, C. Zhou, G. Yan, and B. Wang, “GSSNN: Graph smoothing splines neural networks,” in *Proc. 2020 AAAI Conf. Artif. Intell.*, 2020, pp. 7007–7014.

[274] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, “Neural spline flows,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 7511–7522.

[275] Z. Liu et al., “KAN: Kolmogorov-Arnold networks,” 2024, *arXiv:2404.19756*.

[276] F. Yang, C. Wang, C. Cadena, and M. Hutter, “iPlanner: Imperative path planning,” 2023, *arXiv:2302.11434*.

[277] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, “ViPlanner: Visual semantic imperative learning for local navigation,” in *Proc. 2024 Int. Conf. Robot. Autom.*, IEEE, 2024, pp. 5243–5249.

[278] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models,” in *Proc. 2007 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2007, pp. 1901–1907.

[279] J. Ko and D. Fox, “Learning GP-BayesFilters via Gaussian process latent variable models,” *Auton. Robots*, vol. 30, pp. 3–23, 2011.

[280] B. F. D. Hähnel and D. Fox, “Gaussian processes for signal strength-based location estimation,” in *Proc. Robot.: Sci. Syst. Conf.*, Citeseer, 2006, pp. 1–8.

[281] B. Ferris, D. Fox, and N. D. Lawrence, “WiFi-SLAM using Gaussian process latent variable models,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2007, pp. 2480–2485.

[282] A. Brooks, A. Makarenko, and B. Upcroft, “Gaussian process models for sensor-centric robot localisation,” in *Proc. 2006 Int. Conf. Robot. Autom.*, IEEE, 2006, pp. 56–61.

[283] W. McDonald, C. L. Gentil, and T. Vidal-Calleja, “Global localisation in continuous magnetic vector fields using Gaussian processes,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, IEEE, 2023, pp. 1–5.

[284] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “Gaussian processes and reinforcement learning for identification and control of an autonomous blimp,” in *Proc. 2007 Int. Conf. Robot. Autom.*, IEEE, 2007, pp. 742–747.

[285] D. Nguyen-Tuong and J. Peters, “Local Gaussian process regression for real-time model-based robot control,” in *Proc. 2008 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2008, pp. 380–385.

[286] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, “Gaussian processes for data-efficient learning in robotics and control,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 2, pp. 408–423, Feb. 2015.

[287] J. Boedecker, J. T. Springenberg, J. Wülfing, and M. Riedmiller, “Approximate real-time optimal control based on sparse Gaussian process models,” in *Proc. 2014 IEEE Symp. Adaptive Dyn. Program. Reinforcement Learn.*, IEEE, 2014, pp. 1–8.

[288] W. Cao, A. Capone, R. Yadav, S. Hirche, and W. Pan, “Computation-aware learning for stable control with Gaussian process,” 2024, *arXiv:2406.02272*.

[289] A. Viseras et al., “Decentralized multi-agent exploration with online-learning of Gaussian processes,” in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 4222–4229.

[290] R. Allamraju and G. Chowdhary, “Communication efficient decentralized Gaussian process fusion for multi-UAS path planning,” in *Proc. 2017 Amer. Control Conf.*, IEEE, 2017, pp. 4442–4447.

[291] K.-C. Ma, L. Liu, and G. S. Sukhatme, “Informative planning and online learning with sparse Gaussian processes,” in *Proc. 2017 Int. Conf. Robot. Autom.*, IEEE, 2017, pp. 4292–4298.

[292] R. Mishra, M. Chitre, and S. Swarup, “Online informative path planning using sparse Gaussian processes,” in *Proc. 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans*, IEEE, 2018, pp. 1–5.

[293] D. Jang, J. Yoo, C. Y. Son, D. Kim, and H. J. Kim, “Multi-robot active sensing and environmental model learning with distributed Gaussian process,” *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5905–5912, Oct. 2020.

[294] J. Wakulicz, K. M. B. Lee, C. Yoo, T. Vidal-Calleja, and R. Fitch, “Informative planning for worst-case error minimisation in sparse Gaussian process regression,” in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 11066–11072.

[295] S. O’Callaghan, F. T. Ramos, and H. Durrant-Whyte, “Contextual occupancy maps using Gaussian processes,” in *Proc. 2009 Int. Conf. Robot. Autom.*, IEEE, 2009, pp. 1054–1060.

[296] S. K. Gan, K. Yang, and S. Sukkarieh, “3D path planning for a rotary wing UAV using a Gaussian process occupancy map,” in *Proc. 2009 Australas. Conf. Robot. Autom.*, Citeseer, 2009, pp. 1–6.

[297] S. K. Gan, K. J. Yang, and S. Sukkarieh, “3D online path planning in a continuous Gaussian process occupancy map,” in *Proc. Australian Conf. Field Robot.*, Citeseer, 2009, pp. 1–6.

[298] S. T. O’Callaghan and F. T. Ramos, “Gaussian process occupancy maps,” *Int. J. Robot. Res.*, vol. 31, no. 1, pp. 42–62, 2012.

[299] S. Kim and J. Kim, “Building occupancy maps with a mixture of Gaussian processes,” in *Proc. 2012 Int. Conf. Robot. Autom.*, IEEE, 2012, pp. 4756–4761.

[300] S. Kim and J. Kim, “Continuous occupancy maps using overlapping local Gaussian processes,” in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2013, pp. 4709–4714.

[301] J. Wang and B. Englot, “Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion,” in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 1003–1010.

[302] S. T. O’Callaghan and F. T. Ramos, “Gaussian process occupancy maps for dynamic environments,” in *Proc. 2014 Int. Symp. Exp. Robot.*, Springer, 2016, pp. 791–805.

[303] A. Y. Hata, F. T. Ramos, and D. F. Wolf, “Monte Carlo localization on gaussian process occupancy maps for urban environments,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 2893–2902, Sep. 2018.

[304] M. G. Jadidi, J. V. Miro, and G. Dissanayake, “Warped Gaussian processes occupancy mapping with uncertain inputs,” *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 680–687, Apr. 2017.

[305] Y. Yuan, H. Kuang, and S. Schwertfeger, “Fast Gaussian process occupancy maps,” in *Proc. 2018 Int. Conf. Control Autom. Robot. Vis.*, IEEE, 2018, pp. 1502–1507.

[306] M. G. Jadidi, J. V. Miro, and G. Dissanayake, "Gaussian processes autonomous mapping and exploration for range-sensing mobile robots," *Auton. Robots*, vol. 42, pp. 273–290, 2018.

[307] O. Williams and A. Fitzgibbon, "Gaussian process implicit surfaces," in *Proc. Gaussian Processes Pract.*, 2006, pp. 1–4.

[308] S. Dragiev, M. Toussaint, and M. Gienger, "Gaussian process implicit surfaces for shape estimation and grasping," in *Proc. 2011 Int. Conf. Robot. Autom.*, IEEE, 2011, pp. 2845–2850.

[309] M. G. Castro, T. Peynot, and F. Ramos, "Laser-radar data fusion with Gaussian process implicit surfaces," in *Field and Service Robotics*. Berlin, Germany: Springer, 2013, pp. 1–14.

[310] M. P. Gerardo-Castro, T. Peynot, F. Ramos, and R. Fitch, "Robust multiple-sensing-modality data fusion using Gaussian process implicit surfaces," in *Proc. Int. Conf. Inf. Fusion*, IEEE, 2014, pp. 1–8.

[311] W. Martens, Y. Poffet, P. R. Soria, R. Fitch, and S. Sukkarieh, "Geometric priors for Gaussian process implicit surfaces," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 373–380, Apr. 2017.

[312] S. Caccamo, Y. Bekiroglu, C. H. Ek, and D. Kragic, "Active exploration using Gaussian random fields and Gaussian process implicit surfaces," in *Proc. 2016 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2016, pp. 582–589.

[313] B. Lee, C. Zhang, Z. Huang, and D. D. Lee, "Online continuous mapping using Gaussian process implicit surfaces," in *Proc. 2019 Int. Conf. Robot. Autom.*, IEEE, 2019, pp. 6884–6890.

[314] L. Wu, R. Falque, V. Perez-Puchalt, L. Liu, N. Pietroni, and T. Vidal-Calleja, "Skeleton-based conditionally independent Gaussian process implicit surfaces for fusion in sparse to dense 3D reconstruction," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1532–1539, Apr. 2020.

[315] L. Wu, K. M. B. Lee, L. Liu, and T. Vidal-Calleja, "Faithful euclidean distance field from log-Gaussian process implicit surfaces," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2461–2468, Apr. 2021.

[316] L. Liu, S. Fryc, L. Wu, T. L. Vu, G. Paul, and T. Vidal-Calleja, "Active and interactive mapping with dynamic Gaussian process implicit surfaces for mobile manipulators," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3679–3686, Apr. 2021.

[317] J.-P. A. Ivan, T. Stoyanov, and J. A. Stork, "Online distance field priors for Gaussian process implicit surfaces," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 8996–9003, Oct. 2022.

[318] L. Wu, K. M. B. Lee, C. L. Gentil, and T. Vidal-Calleja, "Log-GPIS-MOP: A unified representation for mapping, odometry, and planning," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 4078–4094, Oct. 2023.

[319] S. Kim and J. Kim, "GPmap: A unified framework for robotic mapping based on sparse Gaussian processes," in *Field and Service Robotics*. Berlin, Germany: Springer, 2015, pp. 319–332.

[320] C. L. Gentil, M. Vayugundla, R. Giubilato, W. Stürzl, T. Vidal-Calleja, and R. Triebel, "Gaussian process gradient maps for loop-closure detection in unstructured planetary environments," in *Proc. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2020, pp. 1895–1902.

[321] R. Giubilato, C. L. Gentil, M. Vayugundla, T. Vidal-Calleja, and R. Triebel, "GPGM-SLAM: Towards a robust SLAM system for unstructured planetary environments with Gaussian process gradient maps," in *Proc. IROS Workshop Planet. Exploration Robots: Challenges Opportunities*, 2020, pp. 1721–1753.

[322] R. Giubilato, C. L. Gentil, M. Vayugundla, M. J. Schuster, T. Vidal-Calleja, and R. Triebel, "GPGM-SLAM: A robust SLAM system for unstructured planetary environments with Gaussian process gradient maps," *Field Robot.*, vol. 2, pp. 1697–1720, 2022.

[323] R. Giubilato et al., "Robust place recognition with Gaussian process gradient maps for teams of robotic explorers in challenging lunar environments," in *Proc. Int. Astronautical Congr.*, 2022, pp. 1–7.

[324] L. Wu, C. L. Gentil, and T. Vidal-Calleja, "Pseudo inputs optimisation for efficient Gaussian process distance fields," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2023, pp. 7249–7255.

[325] C. L. Gentil, O.-L. Ouabi, L. Wu, C. Pradalier, and T. Vidal-Calleja, "Accurate Gaussian-process-based distance fields with applications to echolocation and mapping," *IEEE Robot. Autom. Lett.*, vol. 9, no. 2, pp. 1365–1372, Feb. 2024.

[326] L. Wu, C. L. Gentil, and T. Vidal-Calleja, "VDB-GPDF: Online Gaussian process distance field with VDB structure," *IEEE Robot. Autom. Lett.*, vol. 10, no. 1, Jan. 2025.

[327] M. Ali, H. Jardali, N. Roy, and L. Liu, "Autonomous navigation, mapping and exploration with Gaussian processes," in *Proc. Robot.: Sci. Syst. Conf.*, 2023, pp. 1–13.

[328] A. Melkumyan and F. T. Ramos, "A sparse covariance function for exact Gaussian process inference in large datasets," in *Proc. Int. Joint Conf. Artif. Intell.*, 2009, pp. 1936–1942.

[329] M. Mukadam, X. Yan, and B. Boots, "Gaussian process motion planning," in *Proc. 2016 Int. Conf. Robot. Autom.*, IEEE, 2016, pp. 9–15.

[330] J. Dong, M. Mukadam, F. Dellaert, and B. Boots, "Motion planning as probabilistic inference using Gaussian processes and factor graphs," in *Proc. Robot.: Sci. Syst. Conf.*, 2016, pp. 1–9.

[331] E. Huang, M. Mukadam, Z. Liu, and B. Boots, "Motion planning with graph-based trajectories and Gaussian process inference," in *Proc. 2017 Int. Conf. Robot. Autom.*, IEEE, 2017, pp. 5591–5598.

[332] M. Bhardwaj, B. Boots, and M. Mukadam, "Differentiable gaussian process motion planning," in *Proc. 2020 Int. Conf. Robot. Autom.*, IEEE, 2020, pp. 10598–10604.

[333] J. Cheng, Y. Chen, Q. Zhang, L. Gan, C. Liu, and M. Liu, "Real-time trajectory planning for autonomous driving with Gaussian process and incremental refinement," in *Proc. 2022 Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 8999–9005.

[334] S. Lilge, T. D. Barfoot, and J. Burgner-Kahrs, "Continuum robot state estimation using Gaussian process regression on SE(3)," *Int. J. Robot. Res.*, vol. 41, no. 13/14, pp. 1099–1120, 2022.

[335] S. Lilge, T. D. Barfoot, and J. Burgner-Kahrs, "State estimation for continuum multi-robot systems on SE(3)," *IEEE Trans. Robot.*, vol. 41, pp. 905–925, 2024.

[336] S. Teetaert, S. Lilge, J. Burgner-Kahrs, and T. D. Barfoot, "Space-time continuum: Continuous shape and time state estimation for flexible robots," 2024, *arXiv:2409.12302*.

William Talbot (Graduate Student Member, IEEE) received the B.Eng. degree in mechatronic engineering majoring in space engineering, and the B.Sc. degree majoring in computer science and minoring in physics with the University of Sydney, Sydney, NSW, Australia, the M.Sc. degree in robotics, systems, and control, in 2024, from ETH Zurich, Zurich, Switzerland, where he is currently working toward the Ph.D. degree with the Robotic Systems Laboratory.

His B.Eng. thesis was conducted at the NASA Jet Propulsion Laboratory. His research interests include state estimation, mapping, and physical modeling.

Julian Nubert received the M.Sc. degree in robotics, systems, and control in 2020 from ETH Zurich, Zurich, Switzerland, where he is currently working toward the Ph.D. degree with the Robotic Systems Laboratory.

He is currently with the Max Planck Institute, Stuttgart, Germany, through the MPI ETH Center for Learning Systems. His research interests focus on robust robot perception and how it can be used to deploy mobile robotic systems.

Mr. Nubert was the recipient of the ETH silver medal and was awarded the Willi Studer Prize for his accomplishments during his masters' studies.

Turcan Tuna (Graduate Student Member, IEEE) received the B.Sc. (double major with distinction) degree in mechanical engineering and control and automation engineering from Istanbul Technical University, Istanbul, Türkiye, the M.Sc. degree in robotics, systems, and control, in 2022, from ETH Zurich, Zurich, Switzerland, where he is currently working toward the Ph.D. degree in robust spatial mapping with Robotic Systems Laboratory.

His research interests include developing and deploying robust localization, perception, and mapping frameworks on robotic systems.

Cesar Cadena received the Ph.D. degree in computer science from the University of Zaragoza, Zaragoza, Spain, in 2011.

He is currently a Senior Scientist with ETH Zurich, Zurich, Switzerland. He is interested in providing machines the ability to understand and interact with our ever-changing world. His research interests include robotic scene understanding, both geometry and semantics, covering semantic mapping, data association, place recognition tasks, simultaneous localization and mapping, persistent mapping in dynamic environments, and robot navigation.

Frederike Dümbgen received the B.Sc. and M.Sc. degrees in mechanical engineering, with a minor in computational science and engineering, and the Ph.D. degree in computer and communication sciences from the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2013, 2016, and 2021, respectively.

She conducted her master's thesis with ETH Zurich. From 2022 to 2024, she was a Postdoctoral Researcher with the Robotics Institute, University of Toronto, and from 2024 to 2025, she was a Researcher with the WILLOW team, Inria Paris. In 2026, she will join the Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA, as an Assistant Professor. Her research interests include estimation, control, and advanced optimization for robotics.

Jesús Tordesillas received the Ph.D. degree (under the supervision of Prof. Jonathan P. How) from Aerospace Controls Laboratory, the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2022.

He also did an internship with the NASA-Jet Propulsion Laboratory working for the DARPA Subterranean Challenge. After the Ph.D., he was a Postdoctoral Associate with MIT and later with Robotic Systems Laboratory (led by Prof. Marco Hutter), ETH Zurich. He is currently an Assistant Professor with ICAI, Comillas Pontifical University, Madrid, Spain. His research interests include trajectory planning in dynamic environments, optimization, and constrained deep learning.

Timothy D. Barfoot received the B.A.Sc. degree in engineering science and the Ph.D. degree in aerospace science and engineering from the University of Toronto, Toronto, ON, Canada, in 1997 and 2002, respectively.

He is currently a Professor with the University of Toronto Robotics Institute, Toronto. He is interested in developing methods to allow robotic systems to operate over long periods of time in large-scale, unstructured, 3-D environments, using rich onboard sensing (e.g., cameras and laser rangefinders) and computation. His research interests include guidance, navigation, and control of autonomous systems for a variety of applications.

Marco Hutter (Member, IEEE) received the M.Sc. and Ph.D. degrees in design, actuation, and control of legged robots, from ETH Zurich, Zurich, Switzerland, in 2009 and 2013.

He is an Associate Professor for Robotic Systems at ETH Zurich. His research interests include the development of novel machines and actuation concepts together with the underlying control, planning, and machine learning algorithms for locomotion and manipulation.

Dr. Hutter was the recipient of the ERC Starting Grant, PI of the NCCRs robotics and digital fabrication, PI in various EU projects, and international challenges. He is a cofounder of several ETH Startups including ANYbotics AG and Gravis Robotics AG.