

A Group Theoretic Metric for Robot State Estimation Leveraging Chebyshev Interpolation

Varun Agrawal¹, Frank Dellaert¹

Abstract—We propose a new metric for robot state estimation based on the recently introduced $SE_2(3)$ Lie group definition. Our metric is related to prior metrics for SLAM but explicitly takes into account the linear velocity of the state estimate, improving over current pose-based trajectory analysis. This has the benefit of providing a single, quantitative metric to evaluate state estimation algorithms against, while being compatible with existing tools and libraries. Since ground truth data generally consists of pose data from motion capture systems, we also propose an approach to compute the ground truth linear velocity based on polynomial interpolation. Using Chebyshev interpolation and a pseudospectral parameterization, we can accurately estimate the ground truth linear velocity of the trajectory in an optimal fashion with best approximation error. We demonstrate how this approach performs on multiple robotic platforms where accurate state estimation is vital, and compare it to alternative approaches such as finite differences. The pseudospectral parameterization also provides a means of trajectory data compression as an additional benefit. Experimental results show our method provides a valid and accurate means of comparing state estimation systems, which is also easy to interpret and report.

I. INTRODUCTION

State estimation is an integral part of many robotic systems. For feedback-loop based controllers, having access to accurate state estimates allows the controller to generate the appropriate control law at a particular time instance. The state vector used in such controllers generally consists of the rotation & translation which represents the pose, and the linear and angular velocity, representing the first derivative of the pose. The state vector can be extended to include other elements, such as the biases in an inertial state estimator, but we focus on the general case where we have an inertial sensor to measure the angular velocity, and thus we only need to estimate the pose and linear velocity.

To improve state estimation, it is vital to be able to measure and compare various approaches, and benchmark them accordingly. Almost all current metrics used for evaluating state estimation algorithms are rooted in SLAM or Visual Odometry, and don't take into account the linear velocity aspect of the state, a key issue in these metrics. Some examples of such metrics are Relative Pose Error (RPE), Absolute Trajectory Error (ATE) [1], and Discernible Trajectory Error (DTE) [2]. Zhang and Scaramuzza [3] consider the linear velocity in their state definition, however they report final metrics for the rotation, translation and velocity independently instead of as a single comprehensive

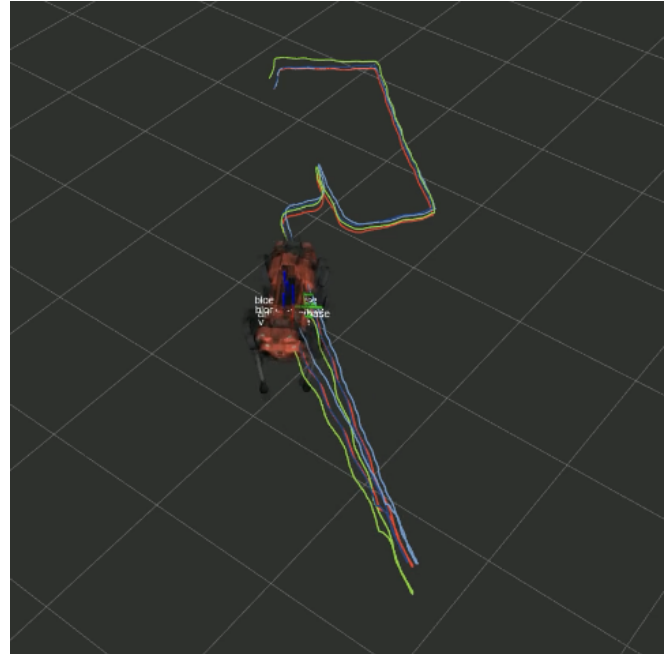


Fig. 1: The ability to compare state estimation techniques with a singular value can aid in various design choices of a robot controller, especially those highly dependent on good state estimates, e.g. for dynamic walking. Here two state estimate trajectories are visualized for qualitative comparison.

metric. As such, the state estimator performance evaluation is missing a crucial aspect required by the robot controller.

A single metric for evaluation which takes into account all aspects of the state is vital to ensure fair comparison and overall performance. For example, the F1-score used in machine learning and information retrieval [4] is a comprehensive summary statistic that takes into account both the precision and recall of the classification model. Since the F1-score singularly captures the performance of the model, it is the *de facto* metric reported in information retrieval and supervised classification literature.

In this work, we propose a singular metric which encapsulates all elements of the generalized state vector. Our metric uses the recently proposed Lie Group $SE_2(3)$ [5], and thus is easy to compute and add to existing software libraries. We refer to this metric as the **Absolute State Error (ASE)**.

Since collecting ground truth data for linear velocity estimation is difficult, we also propose a method for computing the ground truth velocity from the true translation data (generally captured with a motion capture device). While most prior work computes the linear velocity using the method of finite differences [6], our proposed method uses

¹Institute for Robotics and Intelligent Machines and School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA. {varunagrawal, frank.dellaert}@gatech.edu

differentiation via interpolation. There is a wide choice of polynomials to use as the basis model, such as the Fourier series, B-splines, Bernstein polynomials, or Chebyshev polynomials. In our work, we fit the translation data to Chebyshev polynomials, since they have many of the same benefits of splines [7], are known to have optimal performance in a least-square sense, and computing the polynomial derivative is an efficient matrix-vector product [8]. To specify that our metric uses a Chebyshev polynomial, we denote this instantiation as the **Chebyshev Absolute State Error** (C-ASE).

II. RELATED WORK

We review the related literature on state estimation, taking into account the error metrics used. All the works report the generalized state vector, with additional components that are problem or platform specific. Since the related literature on state estimation is quite large, we focus on more recent results in drone and legged robot state estimation.

Mourikis and Roumeliotis [9] is amongst the first state estimation works based on visual-inertial sensor fusion. They report their state estimates as orientation, position and velocity independently without the use of metrics, and without explicitly stating how the true velocity was recorded in hardware experiments. Similarly, Lippiello *et al.* [10] use a simulation environment for their experiments and don't report any metrics, rather they show results qualitatively. Eckenhoff *et al.* [11] only report orientation and position Root Mean Square Error (RMSE), and for experiments with velocity they use the Gazebo simulator. Svacha *et al.* [12] propose an estimator for attitude (orientation) and velocity estimation, each result of which is reported independently with RMSE and Standard Deviation metrics. However, they don't specify how the velocity is computed, other than the data is collected using a motion capture setup. Delmerico *et al.* [13] provide a dataset for high speed autonomous drone racing with only the rotation and position provided. Svacha *et al.* [14] use the platform-provided VIO system to obtain velocity estimates which are used as ground truth substitutes.

For legged robots, Bloesch *et al.* [15], [16] used numerical differentiation via finite differences from the motion capture system's translation data. Rotella *et al.* [17] perform estimation on a humanoid robot with results reported only in simulation. Nobili *et al.* [18] use a specified velocity of 0.5 m/s, and a custom metric called Drift per Distance Traveled (DDT). DDT considers the translation drift over time, assuming that errors in rotation and velocity will carry over into the translation estimates. Hartley *et al.* [19] use SimMechanics for evaluating estimation results in simulation. On hardware experiments however, they only report Relative Position Error and show results qualitatively. In their follow up work, Hartley *et al.* [20] they use the same metric only for translation and use the Cumulative Distribution Function as a measure of translation drift. Wisth *et al.* [6], [21], Buchanan *et al.* [22], [23], Agrawal *et al.* [24] and Kim *et al.* [25] only report the ATE and RPE, without reporting velocity estimation results. Dhédin *et al.* [26] run trajectories with specified horizontal linear velocities. They

report the RPE for pose estimates, and plot the distribution of the estimated velocities to those of a baseline EKF with access to the motion capture data so show correlations in the estimates. Similar to [6], [25], Yang *et al.* [27] don't report velocity estimates, but whilst claiming to report the Relative Pose Error from those papers, they erroneously report the Relative Translation Error, which only evaluates the translation estimates. Fourmy *et al.* [28] use a Kalman Filter to compute the base velocity which is used as the baseline, since they are primarily concerned with Center of Mass velocity and inertia estimation. Probably the most interesting case, Camurri *et al.* [29] use the estimator in a feedback control loop to run walking experiments. Their estimator's state (and particularly velocity) estimates are validated by the performance of the controller. This has the problem of requiring a full locomotion stack, slow iteration times, and difficulty repeating experiments accurately

Polynomial interpolation has seen prior applications in the field of trajectory and state estimation. Sommer *et al.* [30] used B-splines to represent trajectories and compute the trajectory derivatives to obtain accurate velocity and acceleration estimates. Agrawal and Dellaert [31] used Chebyshev polynomials for representing both the state and the dynamics of a quadrotor in a pseudospectral fashion. They leveraged the Chebyshev differentiation matrix and the quadrotor dynamics model to constrain the valid dynamics states. Similarly, Zhu and Wu [32] estimated the coefficients of the Chebyshev polynomial used to represent the trajectory. All these papers perform evaluation in simulation.

From this section, it is apparent that there exists a need for a metric that evaluates all parts of the generalized state vector jointly, as well as an efficient means to compute the linear velocity for hardware experiments.

III. PRELIMINARIES

A. Lie Group and Lie Algebra

A Lie group $GL_n(\mathbb{R})$ is a group which is also a differentiable manifold. It is the set of all $n \times n$ non-singular real matrices where the group operation is matrix multiplication.

The special orthogonal group in 3D is the Lie group $SO(3) = \{\mathbf{R} \in GL_3(\mathbb{R}) \mid \mathbf{R}\mathbf{R}^T = \mathbf{I}, \det\mathbf{R} = +1\}$. It represents all 3D rotation matrices, with the group operation as matrix multiplication, and the inverse is the matrix transpose. The tangent space at the identity for this manifold is the Lie algebra $\mathfrak{so}(3)$ which is given by the space of all 3×3 skew symmetric matrices. The *vec* and *hat* operators are used to convert the matrix to a corresponding vector and vice-versa [33]. The *exponential map* associates an element of the Lie algebra to the Lie group $\mathbf{R}: \mathfrak{so}(3) \rightarrow SO(3)$. Conversely, the *logarithm map* converts the Lie group element to the corresponding Lie algebra $SO(3) \rightarrow \mathfrak{so}(3)$. For more details, we refer the reader to [34].

Similarly, the 3D special Euclidean group is given by $SE(3) = \{\mathbf{T} \in GL_4(\mathbb{R}) \mid \mathbf{R} \in SO(3), \mathbf{p} \in \mathbb{R}^3\}$ and is the set of rigid transformations on \mathbb{R}^3 . The group operation, inverse, exponential map and logarithm map are similar to $SO(3)$,

with the addition of the translation vector as an additional element of the matrix operations [33], [34].

B. Trajectory Evaluation

Trajectory evaluation first arose as a means to benchmark and evaluate SLAM systems [1]. It is important to not only understand how accurate the system is globally, but also how much it varies in a local sense. For this reason, Sturm *et al.* [1] proposed the Absolute Trajectory Error (ATE) to measure the global consistency, and the Relative Pose Error (RPE) to measure the local accuracy over a given window length.

Given the estimated trajectory $\mathbf{P}_1, \dots, \mathbf{P}_n \in \text{SE}(3)$ and the true trajectory $\mathbf{Q}_1, \dots, \mathbf{Q}_n \in \text{SE}(3)$, the Root Mean Square Error (RMSE) for RPE is given by

$$\text{RMSE}(\mathbf{E}_{1:N}, \Delta) = \left(\frac{1}{M} \sum_{i=1}^M \|\text{trans}(\mathbf{E}_i)\|^2 \right)^{1/2} \quad (1)$$

$$\mathbf{E}_i = (\mathbf{Q}_i^{-1} \mathbf{Q}_{i+\Delta})^{-1} (\mathbf{P}_i^{-1} \mathbf{P}_{i+\Delta}) \quad (2)$$

and for ATE is similarly

$$\text{RMSE}(\mathbf{F}_{1:N}) = \left(\frac{1}{N} \sum_{i=1}^N \|\text{trans}(\mathbf{F}_i)\|^2 \right)^{1/2} \quad (3)$$

$$\mathbf{F}_i = \mathbf{Q}_i^{-1} \mathbf{S} \mathbf{P}_i \quad (4)$$

where *trans* specifies the translational components of the transform, Δ is the fixed time interval, $M = N - \Delta$, and \mathbf{S} is the trajectory alignment transform.

IV. STATE ESTIMATION METRIC

A. State Definition

To derive our metric, it is important we define our state vector. Different types of robots can have varying state vector definitions, hence we focus on the general case which is required for robot controllers. We make the assumption that an inertial sensor is available to directly measure the angular velocity of the robot. For a general robotic system, we define the generalized state vector as

$$\mathbf{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i] \quad (5)$$

where $\mathbf{R}_i \in \text{SO}(3)$ is the rotation, $\mathbf{p}_i \in \mathbb{R}^3$ is the translation vector, and $\mathbf{v}_i \in \mathbb{R}^3$ is the linear velocity at time index i .

B. $\text{SE}_2(3)$ Group

Our metric leverages the $\text{SE}_2(3)$ Lie group as proposed by [5]. This group is also termed as “direct spatial isometries”. Similar to how $\text{SE}(3)$ extends $\text{SO}(3)$ with the addition of the translation vector, $\text{SE}_2(3)$ analogously extends $\text{SE}(3)$ with the linear velocity vector.

We tweak the $\text{SE}_2(3)$ group definition in [5] to be equivalently defined as

$$T = \begin{pmatrix} \mathbf{R} & \mathbf{X} & \mathbf{V} \\ 0_{1,3} & 1 & 0 \\ 0_{1,3} & 0 & 1 \end{pmatrix} \quad (6)$$

where $0_{1,3}$ is a 3 dimensional zero row vector.

For our metric, we need to define the group composition and inverse actions. The group composition action is defined as

$$T_1 * T_2 = \begin{pmatrix} \mathbf{R}_1 \mathbf{R}_2 & \mathbf{R}_1 \mathbf{X}_2 + \mathbf{X}_1 & \mathbf{R}_1 \mathbf{V}_2 + \mathbf{V}_1 \\ 0_{1,3} & 1 & 0 \\ 0_{1,3} & 0 & 1 \end{pmatrix} \quad (7)$$

and the inverse operation is defined as

$$T^{-1} = \begin{pmatrix} \mathbf{R}^{-1} & -\mathbf{R}^{-1} \mathbf{X} & -\mathbf{R}^{-1} \mathbf{V} \\ 0_{1,3} & 1 & 0 \\ 0_{1,3} & 0 & 1 \end{pmatrix} \quad (8)$$

The definition of $\text{SE}_2(3)$ thus aids us in defining closed form group operations which are also efficient to compute.

While we don’t explicitly use the Lie algebra aspects of the $\text{SE}_2(3)$ group in our metric, these properties are beneficial for other aspects, such as alignment as discussed below.

C. Alignment

The trajectories first need to be aligned since they can have various ambiguities and gauge freedoms, such as being in different coordinate frames. The transformation required to perform the trajectory alignment is denoted by \mathbf{S} . To handle these ambiguities, we follow the alignment process detailed in Zhang and Scaramuzza [3] since it provides formulae for each component of our generalized state vector.

Let the similarity transformation be parameterized by $S = \{s, \mathbf{R}, \mathbf{t}\}$, where $s \in \mathbb{R}$, $\mathbf{R} \in \text{SO}(3)$, and $\mathbf{t} \in \mathbb{R}^3$. S performs the alignment transformation as

$$\mathbf{R}'_i = \mathbf{R} \mathbf{R}_i, \quad \mathbf{p}'_i = s \mathbf{R} \mathbf{p}_i + \mathbf{t}, \quad \mathbf{v}'_i = s \mathbf{R} \mathbf{v}_i$$

An additional benefit of the Lie group formulation of our approach is that the trajectory alignment can be performed as optimization on the manifold. By leveraging the Lie algebra, we can extend the exposition of Salas *et al.* [35] for the optimal alignment, while taking into consideration all components of the generalized state vector.

For the rest of our discussion, we assume ambiguity resolution and trajectory alignment has already been performed, and the alignment transformation is denoted by \mathbf{S} .

D. Absolute State Error

We denote our proposed metric as *Absolute State Error* to represent the global nature of the metric as well as fact that it explicitly models state estimates. Given an estimated state trajectory $\mathbf{P}_{1:N} \in \text{SE}_2(3)$ and the ground truth states as $\mathbf{Q}_{1:N} \in \text{SE}_2(3)$, alongwith the alignment transformation \mathbf{S} we can compute the Absolute State Error at time step i as

$$\mathbf{E}_i = \mathbf{Q}_i^{-1} \mathbf{S} \mathbf{P}_i \quad (9)$$

To evaluate the state estimates over all time indices, we compute the root mean square error similar to [36]:

$$\text{RMSE}(\mathbf{E}_{1:N}) = \left(\frac{1}{N} \sum_{i=1}^N \|\mathbf{E}_i - \mathbf{I}_{5 \times 5}\|_F^2 \right)^{1/2} \quad (10)$$

where F is the Frobenius matrix norm.

The observant reader will notice that the error metric is of a form similar to the *Absolute Trajectory Error* proposed by Sturm *et al.* [1]. This is advantageous since existing software libraries and tools can be easily extended to use our metric for state estimation.

V. CHEBYSHEV POLYNOMIAL INTERPOLATION

In this section, we briefly review the Chebyshev series and interpolation using Chebyshev polynomials of the second kind (henceforth referred to as Chebyshev polynomials for brevity). We primarily follow the exposition in [37], to which we refer the reader for more details on numerical analysis using Chebyshev polynomials for function approximation.

A. Chebyshev Series & Polynomials

A Chebyshev series is an infinite series akin to the Fourier series which forms an orthogonal basis defined on a unit circle. It is defined for functions $f(x)$ on the interval $[-1, 1]$, though it can be scaled to arbitrary bounds (*e.g.* via min-max normalization), and the series decomposition is given by

$$f(t) = \sum_{k=0}^{\infty} a_k T_k(t) \quad \text{with} \quad a_k = \frac{2}{\pi} \int_{-1}^1 \frac{f(t) T_k(t)}{\sqrt{1-t^2}} dt \quad (11)$$

with the factor $2/\pi$ changing to $1/\pi$ for $k = 0$. Above, T_k is the k th Chebyshev polynomial, defined as the projection of a cosine function to the midline $[-1, 1]$ of the unit circle.

$$T_k(t) \triangleq \cos(k \arccos(t)), \quad -1 \leq t \leq 1 \quad (12)$$

Given an arbitrary real function f on $[-1, 1]$, we can find the best approximation to f with a Chebyshev polynomial of degree N . This is due to the Weierstrass Approximation Theorem [37]) which we recapitulate below

Theorem 1 (Weierstrass Approximation Theorem): Let f be a continuous function on $[-1, 1]$, and let $\epsilon > 0$ be arbitrary. Then there exists a polynomial p such that

$$\|f - p\| < \epsilon \quad (13)$$

A generalized proof can be found in [38]. Thus, we can find the best approximation to any function by interpolating at the Chebyshev-Gauss-Lobatto (or simply Chebyshev) points, defined as:

$$t_j \triangleq \cos(j\pi/N), \quad 0 \leq j \leq N \quad (14)$$

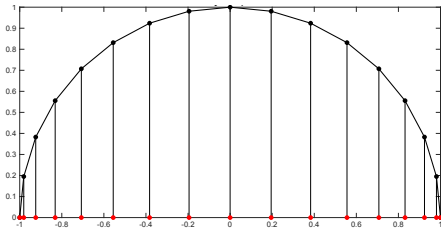


Fig. 2: Chebyshev points $\cos(k\pi/n)$ for degree $n = 16$. They are obtained by projecting a regular grid on the unit circle onto the x-axis.

Interpolation at Chebyshev points ensures efficient convergence of the approximation and avoids issues such as

Runge's phenomenon [39] which causes undesirable oscillation of the interpolant at the endpoints. Furthermore, given the interpolant values, we can make use of the Fast Fourier Transform (FFT) to efficiently obtain the coefficients a_k of the truncated Chebyshev series (the Chebyshev series upto a specified degree N). A visualization is provided in figure 2.

B. Barycentric Interpolation

Given the samples of f at the $N + 1$ Chebyshev points f_j , we can evaluate any point in f efficiently using the Barycentric Interpolation formula [39]. Computationally, the Chebyshev points provide an advantage over equidistant sampled points due to the simplicity of the resulting Barycentric weights, giving us the following interpolation formula

$$f(x) = \sum_{j=0}^N \frac{(-1)^j f_j}{x - x_j} / \sum_{j=0}^N \frac{(-1)^j}{x - x_j} \quad (15)$$

with $f(x) = f_j$ if $x = x_j$, and the summation terms for $j = 0$ and $j = N$ being multiplied by $1/2$.

An additional benefit of Barycentric Interpolation is that it yields a linear form which can be computed via matrix-vector products. Since this is a linear operation, we can reparameterize (15) as an efficient inner product $f(x) = \mathbf{f} \cdot \mathbf{w}$ where \mathbf{f} is a vector of all the values of f at the Chebyshev points, and \mathbf{w} is an $(N + 1)$ vector of Barycentric weights.

C. Differentiation via Interpolation

While finite differences is a common approach for computing derivatives, it is superseded by *Differentiation via Interpolation*. Here, an interpolating polynomial is fit to the samples from the function $f(x)$ (in the least-squares sense) and the derivative of the interpolant is used for computing $f'(x)$. In general, it can be shown that finite differences is a special case of differentiation via interpolation [40]. Furthermore, since polynomial interpolation is a global method, it has better precision and smoothing properties compared to local methods such as (centered) finite differences [41].

Spectral collocation methods, commonly used in the study of differential equations [42], yield an efficient process for obtaining derivatives of the approximating polynomial via the differentiation matrix. Thus, the derivatives of the interpolating polynomial can be efficiently computed via a matrix-vector product. This is useful when performing optimization as we can compute the derivatives of arbitrary functions for (almost) free.

D. Fitting to Noisy Data

The key idea is to express the least-squares polynomial fit in terms of the pseudo-spectral parameterization. Pseudo-spectral parameterization is a way of parameterizing a polynomial completely by its values at specific points, instead of with polynomial coefficients. Given the m vector data samples \mathbf{Z} at discrete time indices, we can minimize the following objective to get optimal the polynomial fit $p^*(\mathbf{X})$ at the pseudo-spectral points:

$$p^*(\mathbf{X}) = \arg \min_{\mathbf{w}} \sum_{i=1}^m \|\mathbf{z}_i - \mathbf{X} \cdot \mathbf{w}_i\|_{\Omega}^2 \quad (16)$$

The main mechanism used is the barycentric interpolation formula 15 which predicts the state $\mathbf{x}(i)$ at any arbitrary time i as $\mathbf{x}_i = \mathbf{X} \cdot \mathbf{w}_i$, with Ω referring to the covariance of the measuring device. As a least-squares objective 16 can be minimized using non-linear programming techniques.

VI. COMPUTING TRUE LINEAR VELOCITY

In order to evaluate state estimation algorithms, it is important to collect ground truth pose and linear velocity data. While motion capture devices and laser trackers are often used for collecting pose data, linear velocity data is not as straightforward to obtain.

We propose to use differentiation via interpolation to estimate the ground truth linear velocity. While most approaches compute the linear velocity using the method of finite differences, differentiation via interpolation is theoretically advantageous, as discussed in section V-C. Given the ground truth translation data from the motion capture device, we use this data as our noisy measurements in 16. By performing a nonlinear least squares optimization, we can obtain a pseudo-spectral parameterization \mathbf{C} of an interpolating polynomial of degree N which represents the translation data. We can use the covariance of the motion capture device as the values for ω to specify our belief in the device's accuracy.

Finally, to compute the true linear velocity, we multiply the values of the polynomial with the $(N + 1) \times (N + 1)$ differentiation matrix \mathbf{D}_N , as defined in [43, p. 53]:

$$\mathbf{v} = \dot{\mathbf{p}} = \mathbf{D}_N \mathbf{C} \mathbf{w}_i \quad (17)$$

VII. EXPERIMENTAL RESULTS

We experimentally validate our Chebyshev interpolation scheme and the use of our proposed C-ASE metric. For the interpolation scheme, we show results in simulation on autonomous driving and quadruped walking.

A. Autonomous Driving

In the case of autonomous driving, we collect trajectories from two different simulators, the CARLA simulator [45], and a simulated version of the AutoRally project [46]. This allows us to have access to both ground truth translation and linear velocity data. We collected a trajectory from each simulator by having a human control the vehicle for an arbitrary amount of time. By letting the trajectory vary in length and possible states, we wish to show the generality and accuracy of our polynomial interpolation approach.

Figs. 3 and 4 show the qualitative results of Chebyshev polynomial interpolation on the translation data. For each trajectory's position data, we correspondingly compute the linear velocity as per section V-C and plot that against the true velocity along with the result of using centered finite differences. From the plots, we can see that the differentiation via interpolation scheme performs at least as well as finite differences, while also possessing a *smoothing* effect on the velocity data, *i.e.*, it avoid extreme peaks.

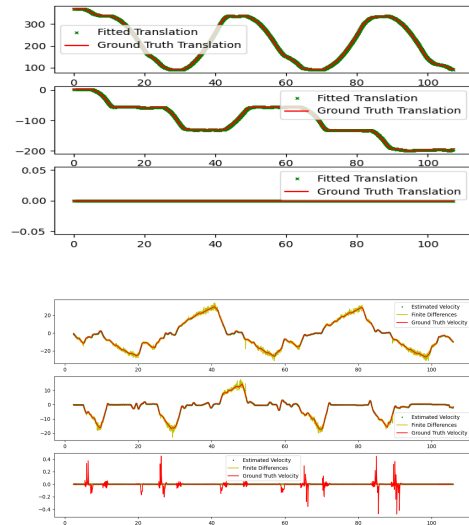


Fig. 3: Chebyshev polynomial fit to the trajectory collected from the CARLA simulator. The ground truth translation and linear velocities are shown as red lines, and the fitted translation and computed velocity are shown as green dots. Result from (centered) finite differences is also shown for comparison. The degree of the polynomial was arbitrarily chosen as 400.

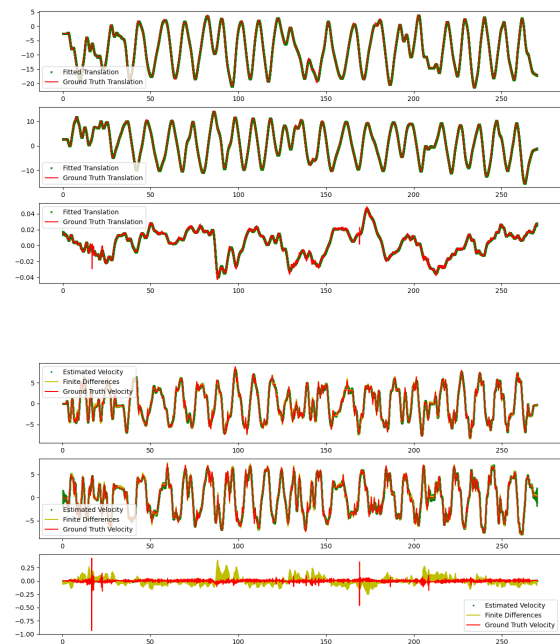


Fig. 4: Similar to the CARLA simulator, we qualitatively show the results of Chebyshev interpolation and differentiation on the AutoRally data.

For quantitative comparison, we compute the Root Mean Square Error (RMSE) between the fitted translation and the true translation, as well as the computed trajectory and the ground truth trajectory. This allows us to examine the effectiveness of our approach at representing the data, as well as how well differentiation via interpolation can perform. For the CARLA trajectory, the RMSE between the polynomial-computed velocity and the true velocity is 0.1860 compared

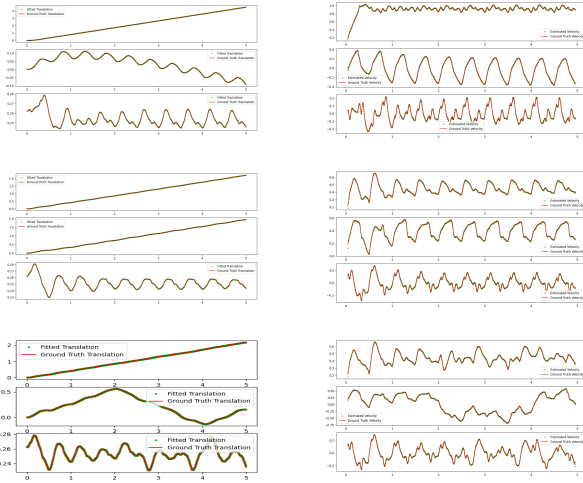


Fig. 5: Chebyshev polynomial fit to the translation and velocity data of an A1 quadruped walking in a diagonal left-front trajectory. The ground truth data is collected in the Pybullet simulation environment [44].

to the RMSE when using finite differences 1.2025. We see a similar trend for the AutoRally trajectory. Table I provides a summary of the experimental results.

It is important to note that the above interpolation was performed with a view of demonstrating correctness, rather than finding the optimal polynomial degree. A higher degree polynomial would lead to lower RMSE, and due to the pseudospectral nature, would still provide significant data compression in terms of representation compared to storing the trajectory in its raw format.

B. Legged Locomotion

To further demonstrate the generality of our approach, we apply the differentiation via interpolation scheme to trajectories of a walking quadruped robot. We consider 3 trajectories: an A1 robot walking in a straight line, in a diagonal line, and in a zig-zag motion. From fig. 5 we see similar results to the autonomous driving experiments, where the translation data is fit accurately, and the computed linear velocity has low RMSE compared to the ground truth. The quantitative summary of the results is provided in table I.

TABLE I: Experimental Results from the CARLA simulator and the AutoRally simulator for autonomous driving, and the A1 robot in the Pybullet simulator for quadruped walking.

Trajectory	Time (s)	Transl. RMSE	Vel. RMSE	Degree
CARLA	107.141	0.0438	0.1860	400
AutoRally Sim	270.23	0.3651	1.0586	256
A1 Straight	5.0	9.3101e-05	0.0121	200
A1 Diagonal	5.0	8.6859e-05	0.0114	200
A1 Zig-Zag	5.0	8.3434e-05	0.0111	200

C. Comparative Analysis

Finally, we test our metric using two state estimators on robot hardware over a 50 second trajectory. For a fair comparison, we use the Bloesch state estimator [15] and the Two State Implicit Filter (TSIF) estimator [47], both of

which are developed by the same group, with the latter shown to have better performance. This is used to verify whether our metric accurately captures the performance gap between these two filtering based state estimators when run on the Anybotics ANYmal C quadruped [48]. Ground truth pose data is collected with the use of motion capture, and the polynomial degree is 128.

TABLE II: Comparison of two filtering based state estimators from the same research group, demonstrating our C-ASE metric accurately captures the difference in performance. Note that the C-ASE metric is dimensionless.

Estimator	RMSE	STD	Median
Bloesch Filter	0.562489	0.226617	0.487763
TSIF	0.297052	0.152732	0.267481

As can be seen from table II, the TSIF estimator has significantly lower C-ASE RMSE when compared to the Bloesch filter, reflecting that our metric indeed captures the difference in performance accurately.

VIII. CONCLUSION

In this work, we have proposed a new metric for robot state estimation evaluation, which leverages the properties of the recently proposed $SE_2(3)$ Lie group. Due to its group properties, this metric explicitly takes into account the linear velocity component of the generalized state vector, while providing a singular comparative value, and is easy to incorporate into existing software libraries and pipelines.

To facilitate the use of this metric, we also show how ground truth linear velocity can be accurately and efficiently computed from translation data obtained from motion capture devices. Leveraging polynomial interpolation using the Chebyshev series of the second kind and differentiation via interpolation, we experimentally validate the accuracy for multiple robotic platforms, and show better results compared to the more common centered finite differences method. Moreover, we leverage a pseudospectral parameterization which provides large data compression in terms of trajectory representation as an additional benefit.

We hope our proposed metric provides a unified approach to benchmarking state estimation systems, and encourages further research in approximation methods for robot trajectory representation. This will allow a fair comparison of such systems, and hopefully lead to further progress in the field of robotic state estimation.

REFERENCES

- [1] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 573–580. DOI: 10.1109/IROS.2012.6385773.
- [2] S. H. Lee and J. Civera, *What's wrong with the absolute trajectory error?* 2023. arXiv: 2212.05376 [cs.RO].

- [3] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7244–7251. DOI: 10.1109/IROS.2018.8593941.
- [4] C. J. Van Rijsbergen, *Information retrieval. 2nd ed.* 1979.
- [5] A. Barrau and S. Bonnabel, "A mathematical framework for imu error propagation with applications to preintegration," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 5732–5738. DOI: 10.1109/ICRA40945.2020.9197492.
- [6] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, Nov. 2019.
- [7] J. H. Park, "Chebyshev approximation of discrete polynomials and splines," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 1999.
- [8] P. Hoffman and K. C. Reddy, "Numerical differentiation by high order interpolation," *SIAM Journal on Scientific and Statistical Computing*, vol. 8, no. 6, pp. 979–987, 1987. DOI: 10.1137/0908079. eprint: <https://doi.org/10.1137/0908079>. [Online]. Available: <https://doi.org/10.1137/0908079>.
- [9] A. Mourikis and S. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007, pp. 3565–3572.
- [10] V. Lippiello, G. Loianno, and B. Siciliano, "Mav indoor navigation based on a closed-form solution for absolute scale velocity estimation using optical flow and inertial data," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 3566–3571. DOI: 10.1109/CDC.2011.6160577.
- [11] K. Eickenhoff, P. Geneva, and G. Huang, "Closed-form preintegration methods for graph-based visual-inertial navigation," *The International Journal of Robotics Research*, vol. 38, no. 5, pp. 563–586, 2019. DOI: 10.1177/0278364919835021. eprint: <https://doi.org/10.1177/0278364919835021>. [Online]. Available: <https://doi.org/10.1177/0278364919835021>.
- [12] J. Svacha, G. Loianno, and V. Kumar, "Inertial yaw-independent velocity and attitude estimation for high-speed quadrotor flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1109–1116, 2019. DOI: 10.1109/LRA.2019.2894220.
- [13] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza, "Are we ready for autonomous drone racing? the uzh-fpv drone racing dataset," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6713–6719. DOI: 10.1109/ICRA.2019.8793887.
- [14] J. Svacha, J. Paulos, G. Loianno, and V. Kumar, "Imu-based inertia estimation for a quadrotor using newton-euler dynamics," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3861–3867, 2020. DOI: 10.1109/LRA.2020.2976308.
- [15] M. Bloesch, M. Hutter, M. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots: Consistent fusion of leg kinematics and imu," in *Robotics: Science and Systems VIII*. IEEE, 2013, pp. 17–24.
- [16] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 6058–6064. DOI: 10.1109/IROS.2013.6697236.
- [17] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal, "State estimation for a humanoid robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 952–958. DOI: 10.1109/IROS.2014.6942674.
- [18] S. Nobili, M. Camurri, V. Barasuol, M. Focchi, D. Caldwell, C. Semini, and M. Fallon, "Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots," in *Proceedings of Robotics: Science and Systems*, Robotics: Science and Systems Foundation, 2017.
- [19] R. Hartley, J. Mangelson, L. Gan, M. Ghaffari Jaididi, J. M. Walls, R. M. Eustice, and J. W. Grizzle, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2018, pp. 4422–4429. [Online]. Available: <https://arxiv.org/pdf/1712.05873.pdf>.
- [20] R. Hartley, M. Ghaffari Jaididi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 3783–3790. [Online]. Available: <https://arxiv.org/pdf/1803.07531.pdf>.
- [21] D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2020.
- [22] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in *5th Annual Conference on Robot Learning*, 2021. [Online]. Available: <https://openreview.net/forum?id=a5ZiDzL0enJ>.
- [23] R. Buchanan, V. Agrawal, M. Camurri, F. Dellaert, and M. Fallon, "Deep imu bias inference for robust visual-inertial odometry with factor graphs," *IEEE Robotics*

- and *Automation Letters*, vol. 8, no. 1, pp. 41–48, 2023. DOI: 10.1109/LRA.2022.3222956.
- [24] V. Agrawal, S. Bertrand, R. Griffin, and F. Dellaert, “Proprioceptive state estimation of legged robots with kinematic chain modeling,” in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, 2022, pp. 178–185. DOI: 10.1109/Humanoids53995.2022.10000099.
- [25] Y. Kim, B. Yu, E. M. Lee, J.-h. Kim, H.-w. Park, and H. Myung, “Step: State estimator for legged robots using a preintegrated foot velocity factor,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4456–4463, 2022.
- [26] V. Dhédin, H. Li, S. Khorshidi, L. Mack, A. K. C. Ravi, A. Meduri, P. Shah, F. Grimmering, L. Righetti, M. Khadiv, and J. Stueckler, “Visual-inertial and leg odometry fusion for dynamic locomotion,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 9966–9972. DOI: 10.1109/ICRA48891.2023.10160898.
- [27] S. Yang, Z. Zhang, Z. Fu, and Z. Manchester, “Cerberus: Low-drift visual-inertial-leg odometry for agile locomotion,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 4193–4199. DOI: 10.1109/ICRA48891.2023.10160486.
- [28] M. Fourmy, T. Flayols, P.-A. Leziart, N. Mansard, and J. Solà, “Contact forces preintegration for estimation in legged robotics using factor graphs,” May 2021, pp. 1372–1378. DOI: 10.1109/ICRA48506.2021.9561037.
- [29] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, “Pronto: A multi-sensor state estimator for legged robots in real-world scenarios,” *Frontiers in Robotics and AI*, vol. 7, 2020.
- [30] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, “Efficient derivative computation for cumulative b-splines on lie groups,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, 2020, pp. 11145–11153. DOI: 10.1109/CVPR42600.2020.01116. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.01116>.
- [31] V. Agrawal and F. Dellaert, “Continuous-time state & dynamics estimation using a pseudo-spectral parameterization,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi’an, China: IEEE Press, 2021, 426–432. DOI: 10.1109/ICRA48506.2021.9561318. [Online]. Available: <https://doi.org/10.1109/ICRA48506.2021.9561318>.
- [32] M. Zhu and Y. Wu, “Chevopt: Continuous-time state estimation by chebyshev polynomial optimization,” *IEEE Transactions on Signal Processing*, vol. 70, pp. 3136–3147, 2022. DOI: 10.1109/TSP.2022.3183435.
- [33] J. Solà, J. Deray, and D. Atchuthan, “A micro lie theory for state estimation in robotics,” *CoRR*, vol. abs/1812.01537, 2018. arXiv: 1812.01537. [Online]. Available: <http://arxiv.org/abs/1812.01537>.
- [34] Y. Wang and G. S. Chirikjian, “Nonparametric second-order theory of error propagation on motion groups,” *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1258–1273, 2008, PMID: 20333324. DOI: 10.1177/0278364908097583. eprint: <https://doi.org/10.1177/0278364908097583>. [Online]. Available: <https://doi.org/10.1177/0278364908097583>.
- [35] M. Salas, Y. Latif, I. D. Reid, and J. Montiel, “Trajectory alignment and evaluation in slam: Horns method vs alignment on the manifold,” in *Robotics: Science and Systems Workshop: The problem of mobile sensors*, sn, 2015, pp. 1–3.
- [36] M. Grupp, *Evo: Python package for the evaluation of odometry and slam*. <https://github.com/MichaelGrupp/evo>, 2017.
- [37] L. N. Trefethen, *Approximation theory and approximation practice*. Siam, 2013.
- [38] M. H. Stone, “The generalized weierstrass approximation theorem,” *Mathematics Magazine*, vol. 21, no. 4, pp. 167–184, 1948, ISSN: 0025570X, 19300980. [Online]. Available: <http://www.jstor.org/stable/3029750> (visited on 03/06/2024).
- [39] J. Berrut and L. Trefethen, “Barycentric lagrange interpolation,” *SIAM Review*, vol. 46, no. 3, pp. 501–517, 2004. [Online]. Available: <http://dx.doi.org/10.1137/S0036144502417715>.
- [40] T. A. Driscoll and R. J. Braun, *Fundamentals of Numerical Computation*. Siam, 2017, vol. 154.
- [41] K. Ahnert and M. Abel, “Numerical differentiation of experimental data: Local versus global methods,” *Comput. Phys. Commun.*, vol. 177, pp. 764–774, 2007. [Online]. Available: <https://api.semanticscholar.org/CorpusID:15129086>.
- [42] M. Hussaini, D. Kopriva, and A. Patera, “Spectral collocation methods,” *Applied Numerical Mathematics*, vol. 5, no. 3, pp. 177–208, 1989, ISSN: 0168-9274. DOI: [https://doi.org/10.1016/0168-9274\(89\)90033-0](https://doi.org/10.1016/0168-9274(89)90033-0). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0168927489900330>.
- [43] L. N. Trefethen, *Spectral methods in MATLAB*. SIAM, 2000, vol. 10.
- [44] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” 2016.
- [45] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

- [46] B. Goldfain, P. Drews, C. You, M. Barulic, O. Velev, P. Tsiotras, and J. M. Rehg, "Autorially: An open platform for aggressive autonomous driving," *IEEE Control Systems Magazine*, vol. 39, no. 1, pp. 26–55, 2019.
- [47] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The Two-State Implicit Filter - Recursive Estimation for Mobile Robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 573–580, 2017, ISSN: 2377-3766. DOI: 10.1109/LRA.2017.2776340.
- [48] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44. DOI: 10.1109/IROS.2016.7758092.