Model-Based 6D Visual Object Tracking with Impact Collision Models

Maarten M. J. Jongeneel, Alexandre Bernardino, Nathan van de Wouw, Alessandro Saccon

 DOI: https://doi.org/10.23919/ACC53348.2022.9867622

Model-Based 6D Visual Object Tracking with Impact Collision Models

Maarten Jongeneel[1], Alexandre Bernardino[2], Nathan van de Wouw[3], and Alessandro Saccon[4]

***Abstract*— 3D Object tracking is an essential technique in computer vision and has many fields of application. In this study, the focus lies on tracking objects impacting the environment. We show that state-of-the-art methods lose track of objects in this context and we investigate how to overcome this problem by adding prior information regarding the object and surface where collision is expected to occur. For illustration purposes and application relevance, we focus on the case of a box impacting a surface, which is, e.g., encountered in robot tossing in logistics applications. We model the effects of impacts and friction in a motion model, and consider the state of the box to evolve in a Lie group. We present an object tracking algorithm, based on an Unscented Particle Filter, for systems whose state lives in a Lie group and incorporate this motion model. The observations are taken from a single RGB camera and make use of the known 3D model of the object and color characteristics to predict its appearance in the 2D image. We quantitatively evaluate the effectiveness of our proposed methods by means of simulations on synthetic images.**

***Index Terms*— Visual 3D Object Tracking, Nonlinear Dynamics, Particle Filter, Nonsmooth Mechanics.**

I. INTRODUCTION

Estimating the position and orientation of objects in a 3D space from RGB images is an important problem in robotic applications. In robotic manipulation, for example, accurate pose estimation is advantageous for executing advanced grasping tasks. In this paper, we focus on tracking rigid objects that collide with the environment. This can be of particular interest in real-world applications such as robot batting [1] or robot soccer [2]. For sake of brevity, we focus specifically on box-shaped objects as this is also relevant for logistics applications, e.g., for robot tossing[5]. Our approach, however, can be extended to objects with other geometries.

To the best of the authors knowledge, there is no specific work in literature focusing on 3D object tracking in the presence of collisions. A substantial body of work has been recently published on 6D object pose estimation from single-shot, potentially multi-view, RGB(D) camera images. Relevant examples include [3], [4], [5], [6] and references therein. Making use of latent pose or shape representations provided by variational autoencoders, these methods allow to obtain quite robust pose estimates, also in the presence of partial occlusions, for known objects [3], [4], [5] or objects within a known class [6]. These single-shot pose estimation techniques or more traditional methods such as RAPiD-like approaches [7], [8], can then be used as measurement in feature-based or model-based tracking algorithms as shown in [9], [7], [10]. Instead of just employing an instantaneous 6D pose measurement at each instant of time, these tracking algorithms make use of both current and past observations $\mathbf{y}_{1:t}$ [7], thereby improving tracking performance even further. In these cases, the problem of object tracking becomes the problem of estimating the *filtering density*, denoted as $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ [11], recursively in time. In addition to orientation and position parameters, the object state $\mathbf{x}_t$ then also includes linear and angular velocity parameters.

Often, the filtering density is unknown, and there are numerous approximation methods, with the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and the Particle Filter (PF) as the most prominent ones. The EKF and UKF are only valid if the posterior distribution can be closely approximated by a Gaussian distribution and remains unimodal. On the contrary, the PF maintains several hypotheses over time, which allows to approximate the full posterior distribution of systems with non-linear dynamics and non-Gaussian state distributions [12], [11], as is the case for our specific case of tossing a box impacting a surface. Impacts, in particular, lead to nonsmooth behavior, a rather extreme case of nonlinearity with sudden state jumps. We therefore base our approach on the PF.

In a PF, a set of particles $\{\mathbf{x}_t^{(i)}\}_{i=1}^N$ are sampled from a *proposal distribution*, $q(\mathbf{x}_t^{(i)}|\mathbf{y}_{1:t})$, to approximate the filtering density. The quality of this approximation is reflected by so-called unnormalized importance weights $\tilde{w}_t^{(i)}$, which are given as the ratio between the true filtering density and the proposal distribution. Ideally, the proposal distribution is as close as possible to the true filtering density [11] meaning that the choice of proposal distribution is of high importance for the approximation of the filtering density.

Within the context of visual object tracking with Particle Filters, we consider two distinct open problems. First, in literature (see, e.g., [13], [14]), the choice is often made to use the prior distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as a proposal, such that the particles from the previous state $\mathbf{x}_{t-1}$ are directly propagated to the next time step through a *motion model*, without the use of the available observation $\mathbf{y}_t$. Consequently, it may happen that the complete set of particles moves away from the area with high likelihood, resulting in

[1]Maarten Jongeneel (corresponding author) has conducted this work during his MSc project at the Department of Mechanical Engineering, Eindhoven University of Technology (TU/e), The Netherlands (m.j.jongeneel@tue.nl)

[2]Alexandre Bernardino is with the Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisbon, Portugal (alex@isr.tecnico.ulisboa.pt)

[3]Nathan van de Wouw is with the Department of Mechanical Engineering, Eindhoven University of Technology (TU/e), The Netherlands (n.v.d.wouw@tue.nl)

[4]Alessandro Saccon is with the Department of Mechanical Engineering, Eindhoven University of Technology (TU/e), The Netherlands (a.saccon@tue.nl)

[5]This work was partially supported by the Research Project I.AM. through the European Union H2020 program under GA 871899.

degeneracy of the particle set [15]. Second, in literature of visual object tracking, the adopted motion model is typically a constant velocity model [9], [16], [17]. To the best of the authors' knowledge, there does not exist literature on object tracking algorithms exploring the possibility of incorporating prior information regarding the object and surface that are expected to get in contact, potentially undergoing impacts (*nonsmooth behavior*).

To tackle the first problem, we propose to use the *Unscented Particle Filter* (UPF) [15], which uses a UKF that takes the current observation into account to create a proposal distribution. As the 6D pose of an object in 3D space does not evolve in a vector space, but rather on a Lie group, we propose a version of the UPF for a system whose state lives in a Lie group. Although literature exists for particle filters on Lie groups [9], [18] or Unscented Kalman filters on Lie Groups [19], to the best of the authors knowledge, there does not exist literature on the UPF on Lie Groups. We name this new variant of the filter the *Geometric Unscented Particle Filter* (GUPF), which can be seen as the first contribution of this paper.

For dealing with the second problem (nonsmooth behavior), we investigate the consequences of incorporating impact and friction on a nonsmooth dynamical motion model in the GUPF that more accurately describes the nonsmooth dynamics of the object. In a numerical study, we will show that, given the shape of the box, the post-impact velocity is highly sensitive to the pre-impact velocity and pose, and tracking algorithms based on constant velocity models typically fail to track the object accurately beyond the impact. Integrating a nonsmooth model in an object tracking strategy is our second contribution.

The paper has the following structure. In Section II, we introduce notation and review stochastic uncertainty models on Lie groups. In Section III, we detail the Geometric Unscented Particle Filter. In Section IV, we apply the proposed GUPF on box tossing, explicitly providing the motion model and the observation likelihood function. In Section V, we verify our proposed methods by means of simulations, while Section VI concludes with a summary.

II. Mathematical Preliminaries

We assume the reader is familiar with Unscented Kalman Filters [15], Particle Filters [11], and matrix Lie groups [20], [18]. We will use calligraphic letters to denote a Lie group $\mathcal{G}$, and fraktur letters to denote its corresponding Lie algebra $\mathfrak{g}$. Since $\mathcal{G}$ is not a vector space, we cannot apply the usual approach of additive noise. We follow the approach of [21], [22] and define the wrapped Gaussian probability distribution $g \sim \mathcal{N}_L(\bar{g}, \mathbf{\Sigma})$ for the random variable $g \in \mathcal{G}$ as

$$g := \bar{g}\mathrm{Exp}(\xi),\ \xi \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \tag{1}$$

with $\bar{g} \in \mathcal{G}$, $\xi \in \mathfrak{g}$, $\mathbf{\Sigma} \in \mathfrak{g} \otimes \mathfrak{g}$, and where $\mathcal{N}$ denotes the ordinary Normal distribution in Euclidean space and $\otimes$ the tensor product. The sample mean $\bar{g}$ and covariance $\mathbf{\Sigma}$ can be obtained from a sample set $\{g_i\}_{i=1}^N \in \mathcal{G}$ as detailed in [23], [24]. This procedure is shown in Algorithm 1 and will be used in Section III. Note that in Algorithm 1, θ is a threshold value and an initial guess for $\bar{g}$ should be given.

Algorithm 1 Weighted mean $\bar{g}$ and covariance $\mathbf{\Sigma}$ computation on the Lie group $\mathcal{G}$

Input: $\bar{g}$, $\{g_i\}_{i=1}^N$, $\{W_i^{(m)}\}_{i=1}^N$, $\{W_i^{(c)}\}_{i=1}^N, \theta$

Compute the mean:
repeat
  $\xi_i \leftarrow \mathrm{Log}\left(\bar{g}^{-1} g_i\right),\ i = 1, \ldots, N$
  $\bar{\xi} \leftarrow \sum_{i=1}^N W_i^{(m)} \xi_i$
  $\bar{g} \leftarrow \bar{g}\mathrm{Exp}(\bar{\xi})$
until $\|\bar{\xi}\| < \theta$
Compute the covariance:
$\xi_i = \mathrm{Log}\left(\bar{g}^{-1} g_i\right),\ i = 1, \ldots, N$
$\mathbf{\Sigma} = \sum_{i=1}^N W_i^{(c)} \xi_i \xi_i^T$

Output: $\bar{g}$, $\mathbf{\Sigma}$

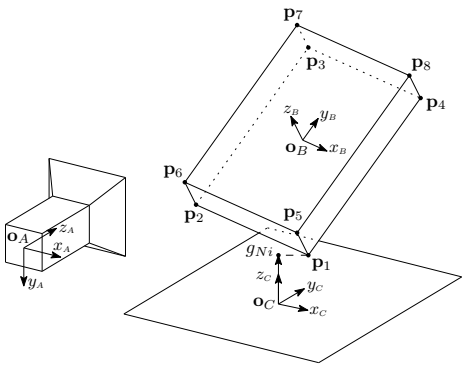


Fig. 1: Illustration of the box, about to make contact with a surface. The gap function is indicated by g_{Ni}, together with the box frame B, contact surface frame C, and inertial frame A.

The state (pose-velocity pair) of a rigid body will be denoted by $\mathbf{x}_t := ({}^A\mathbf{R}(t)_B, {}^A\mathbf{o}(t)_B, {}^B\boldsymbol{v}(t)_{A,B}, {}^B\boldsymbol{\omega}(t)_{A,B}) \in \mathcal{S}$, where we define the state-group as $\mathcal{S} := \mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3_\times$, with $\mathbb{R}^3_\times$ denoting the cross product Lie algebra, equivalent to $\mathfrak{so}(3)$. Note that with ${}^A\mathbf{R}(t)_B$ we denote the coordinate transformation from frame B to frame A and with ${}^A\mathbf{o}(t)_B$ we express the coordinates of the origin of frame B w.r.t. frame A. For further details about this notation, see [25]. We define the pose-group as $\mathcal{P} := \mathrm{SO}(3) \times \mathbb{R}^3$, and denote the Lie algebras corresponding to $\mathcal{S}$ and $\mathcal{P}$ as $\mathfrak{s}$ and $\mathfrak{p}$, respectively. See also Fig. 1 for a schematic overview of the problem, where frame C defines the contact surface, frame B the body-fixed frame of the box located at its center, frame A the inertial frame, $\mathbf{p}_i$ the contact points of the box, and g_{N_i} the gap functions, as we will describe in Section IV. We write all positions and velocities of the box frame B with respect to the inertial frame A, such that we omit the sub- and superscripts of A and B for the sake of brevity, that is, e.g., we will write $\mathbf{v}$ for ${}^B\mathbf{v}_{A,B}$.

III. The Geometric Unscented Particle Filter

This section will describe the newly proposed Geometric Unscented Particle Filter (GUPF). We closely follow the procedure of the UPF in [15] and stress that the novelty

of our proposed GUPF lies in the fact that we generalize the UPF for systems whose state evolves in a Lie group.

We assume an initial state distribution $p(\mathbf{x}_0)$, defined by a wrapped Gaussian with mean $\bar{\mathbf{x}}_0 \in \mathcal{S}$ and covariance $\boldsymbol{P}_0^{\mathbf{x}} \in \mathfrak{s} \otimes \mathfrak{s}$. We thus sample a set of $\{\mathbf{x}_0^{(i)}\}_{i=1}^N$ particles according to $\mathbf{x}_0^{(i)} = \bar{\mathbf{x}}_0 \operatorname{Exp}\left(\boldsymbol{\chi}^{(i)}\right)$ with $\boldsymbol{\chi}^{(i)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P}_0^{\mathbf{x}})$, such that these particles represent the initial state distribution. At time $t-1$ (including the initial time), we consider a single particle $\mathbf{x}_{t-1}$ where for the sake of brevity we have discarded the superscript $(\cdot)^{(i)}$. We follow the standard approach of [15] and augment the state mean and covariance with the process noise and observation noise parameters. To this end, we define the augmented-state group $\mathcal{A} \coloneqq \mathcal{S} \times \mathcal{S} \times \mathcal{P}$ and the augmented state as $\mathbf{x}^{\mathbf{a}} = (\mathbf{x}, \mathbf{m}, \mathbf{n}) \in \mathcal{A}$, where $\mathbf{x} \in \mathcal{S}$ is the state, $\mathbf{m} \in \mathcal{S}$ the process noise, and $\mathbf{n} \in \mathcal{P}$ the observation noise. The Lie algebra of $\mathcal{A}$ is defined as $\mathfrak{a} \coloneqq \mathfrak{s} \otimes \mathfrak{s} \otimes \mathfrak{p}$. We will define the augmented state mean and covariance in the tangent space at a point $\mathbf{x}_{t-1}^{\mathbf{a}} = (\mathbf{x}_{t-1}, \mathbf{m}, \mathbf{n}) \in \mathcal{A}$ as

$$\bar{\boldsymbol{\chi}}_{t-1}^{\mathbf{a}} = \mathbf{0} \in \mathfrak{a}, \;\; \boldsymbol{P}_{t-1}^{\mathbf{a}} = \operatorname{diag}(\boldsymbol{P}^{\mathbf{x}}, \boldsymbol{Q}, \boldsymbol{R}) \in \mathfrak{a} \otimes \mathfrak{a}, \quad (2)$$

respectively, where $\boldsymbol{P}^{\mathbf{x}} \in \mathfrak{s} \otimes \mathfrak{s}$ is the state covariance, $\boldsymbol{Q} \in \mathfrak{s} \otimes \mathfrak{s}$ the process noise covariance, and $\boldsymbol{R} \in \mathfrak{p} \otimes \mathfrak{p}$ the observation noise covariance. This means the sigma points are defined in the Lie algebra $\mathfrak{a}$ (representing the tangent space of $\mathcal{A}$ at the point $\mathbf{x}_{t-1}^a$ via left translation). Since this is a vector space, the sigma points can be simply computed as in [15], which gives $\boldsymbol{\chi}_{t-1}^{\mathbf{a}} = \left[\bar{\boldsymbol{\chi}}_{t-1}^{\mathbf{a}} \;\; \bar{\boldsymbol{\chi}}_{t-1}^{\mathbf{a}} \pm \sqrt{(n_a + \lambda)\boldsymbol{P}_{t-1}^{\mathbf{a}}}\right]$, where $\lambda = \alpha^2(n_a + \zeta) - n_a$ with ζ and α scaling parameters to scale the sigma points and n_a denoting the dimension of the augmented state, which in our case is $n_a = 30$. As a result, each sigma point j can be written as $\boldsymbol{\chi}_{j,t-1}^{\mathbf{a}} = [(\boldsymbol{\chi}_{j,t-1}^{\mathbf{x}}); (\boldsymbol{\chi}_{j,t-1}^{\mathbf{m}}); (\boldsymbol{\chi}_{j,t-1}^{\mathbf{n}})]$, where ; denotes row concatenation and where $\boldsymbol{\chi}_{j,t-1}^{\mathbf{x}} \in \mathfrak{s}$ is the state component, $\boldsymbol{\chi}_{j,t-1}^{\mathbf{m}} \in \mathfrak{s}$ the process noise component, and $\boldsymbol{\chi}_{j,t-1}^{\mathbf{n}} \in \mathfrak{p}$ the observation noise component of the sigma point. Each sigma point is then weighted such that the true mean and covariance of the particle are fully captured by a set of weighted sigma points denoted by $\left\{\boldsymbol{\chi}_{j,t-1}^{\mathbf{a}}, W_j^{(m)}, W_j^{(c)}\right\}_{j=0}^{2n_a+1}$, where $W_j^{(m)} \in \mathbb{R}$ are the weights of the mean and $W_j^{(c)} \in \mathbb{R}$ are the weights of the covariance as in [15] and Algorithm 1.

A. Creating the prior and proposal distributions

We now propagate the state component of the sigma points $\boldsymbol{\chi}_{j,t-1}^{\mathbf{x}} \in \mathfrak{s}$, to obtain the predicted state sigma points $\mathbf{x}_{j,t|t-1} \in \mathcal{S}$. Propagation is obtained through the motion model $f : \mathcal{S} \to \mathcal{S}$, and details of this model will be given in Section IV. For now, we assume that this function is known, such that, for every sigma point $j \in [0, 2n_a]$,

$$\mathbf{x}_{j,t|t-1} = f\left(\mathbf{x}_{t-1} \operatorname{Exp}\left(\boldsymbol{\chi}_{j,t-1}^{\mathbf{x}}\right)\right) \operatorname{Exp}\left(\boldsymbol{\chi}_{j,t-1}^{\mathbf{m}}\right). \quad (3)$$

Note that, in the Euclidean case, the noise parameters are incorporated by simple addition. As in our case the state evolves in a Lie group, this procedure is done by post-multiplication of the exponential map of the noise parameters, as is illustrated by the last term of (3). The predicted particle state mean and covariance can now be computed by following the procedure of Algorithm 1, using $\mathbf{x}_{0,t|t-1}$ as initial guess for the mean. In this way, we obtain for each particle its *prior distribution*, given by $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}_L(\mathbf{x}_{t|t-1}, \boldsymbol{P}_{t|t-1}^{\mathbf{x}})$.

As in [15], the next step is to incorporate a measurement. In our case, such a measurement gives the 3D position and orientation of the box, such that $\mathbf{Z}_t = (\mathbf{R}, \mathbf{o}) \in \mathcal{P}$. This measurement could be the result of some 3D model-based pose estimation algorithm [7]. Our proposed method computes the measurement as a *weighted mean* from the predicted set of particles $\mathbf{x}_{t|t-1}^{(i)}$ obtained by (3) by considering their pose components $\mathbf{H}^{(i)} = (\mathbf{R}^{(i)}, \mathbf{o}^{(i)}) \in \mathcal{P}$ and evaluating them through a *likelihood function* $\mathcal{L} : \mathcal{P} \to \mathbb{R}$ according to

$$L^{(i)} = \mathcal{L}(\mathbf{H}^{(i)}). \quad (4)$$

For the specific case of rectangular boxes, we will detail this function in Section IV. To compute the measurement from a weighted mean on the pose group, we use as initial guess

$$\tilde{\mathbf{Z}}_t = \underset{i=1,\ldots,N}{\arg\max} \left(\mathcal{L}\left(\mathbf{H}^{(i)}\right)\right) \in \mathcal{P}, \quad (5)$$

and follow the same procedure as in Algorithm 1, substituting $W_i^{(m)}$ with the normalized likelihood $\tilde{L}^{(i)}$ associated to the i-th particle. This average is taken as the measurement $\mathbf{Z}_t \in \mathcal{P}$. Each sigma point $\mathbf{x}_{j,t|t-1}$ in (3) is now evaluated through the observation model $h : \mathcal{S} \to \mathcal{P}$ together with the noise component of the sigma point to obtain the predicted pose

$$\mathbf{H}_{j,t|t-1} = h\left(\mathbf{x}_{j,t|t-1}\right) \operatorname{Exp}\left(\boldsymbol{\chi}_{j,t-1}^{\mathbf{n}}\right) \in \mathcal{P}. \quad (6)$$

Using Algorithm 1, we compute the mean of the points $\mathbf{H}_{j,t|t-1}$ by using $\mathbf{H}_{0,t|t-1}$ as initial guess. This results in $\bar{\mathbf{H}}_{t|t-1}$, at which we consider a tangent space and we map the predicted poses obtained in (6) to this space resulting in $\mathcal{H}_{j,t|t-1} \in \mathfrak{p}$. The observation noise covariance and state-observation cross-covariance are then given by

$$\boldsymbol{P}_{t|t-1}^{\mathbf{zz}} = \sum_{j=0}^{2n_a} W_j^{(c)} \left(\mathcal{H}_{j,t|t-1}\right)\left(\mathcal{H}_{j,t|t-1}\right)^T \in \mathfrak{p} \otimes \mathfrak{p} \quad (7)$$

and

$$\boldsymbol{P}_{t|t-1}^{\mathbf{xz}} = \sum_{j=0}^{2n_a} W_j^{(c)} \left(\boldsymbol{\chi}_{j,t|t-1}^{\mathbf{x}}\right)\left(\mathcal{H}_{j,t|t-1}\right)^T \in \mathfrak{s} \otimes \mathfrak{p}, \quad (8)$$

respectively. The innovation is then computed by incorporating the measurement at time t as in [19] such that

$$\boldsymbol{\rho} = \operatorname{Log}\left(\left(\bar{\mathbf{H}}_{t|t-1}\right)^{-1} \mathbf{Z}_t\right) \in \mathfrak{p}. \quad (9)$$

The state mean is now updated according to

$$\mathbf{x}_t = \mathbf{x}_{t|t-1} \operatorname{Exp}\left(\left(\boldsymbol{P}_{t|t-1}^{\mathbf{xz}} \left(\boldsymbol{P}_{t|t-1}^{\mathbf{zz}}\right)^{-1} \boldsymbol{\rho}\right)\right), \quad (10)$$

and the state covariance is then updated (cf. [19]) to obtain

$$\boldsymbol{P}_t^{\mathbf{x}} = \boldsymbol{P}_{t|t-1}^{\mathbf{x}} - \boldsymbol{P}_{t|t-1}^{\mathbf{xz}} \left(\boldsymbol{P}_{t|t-1}^{\mathbf{xz}} \left(\boldsymbol{P}_{t|t-1}^{\mathbf{zz}}\right)^{-1}\right)^T \quad (11)$$

with $\boldsymbol{P}_t^{\mathbf{x}} \in \mathfrak{s} \otimes \mathfrak{s}$. As a result, the proposal distribution for each particle is given by $q(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}_L\left(\mathbf{x}_t, \boldsymbol{P}_t^{\mathbf{x}}\right)$.

B. Computing the weights for each particle

We now sample each particle from its proposal distribution according to $\tilde{\mathbf{x}}_t = \mathbf{x}_t \mathrm{Exp}(\boldsymbol{\chi})$ with $\boldsymbol{\chi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P}_t^{\mathbf{x}})$ as in (1). The prior distribution, the proposal distribution, and the likelihood function are then evaluated at this sampled particle, from which the unnormalized weight can be computed using the *importance function* as in [11]. The final state estimate is obtained by computing a weighted mean of all the particles, following Algorithm 1, where we consider the particle $\tilde{\mathbf{x}}_t^{(i)}$ with the highest weight as the initial guess. Next, all particles are *resampled* according to their weight as in [15].

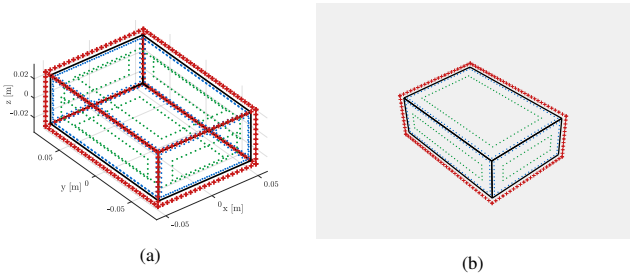
IV. Likelihood function and motion model

In this section, we describe how we apply the GUPF for our specific case of tracking an impacting box. Therefore, we describe the likelihood function used in (4), and derive the motion model used in (3).

A. Color-based likelihood computation

The observation likelihood function is a measure for the compatibility between the observed image and the object pose. Instead of using an approach as in [10], which would require the training of an auto-encoder, we use an approach similar to [13], [26], where an accurate and computationally efficient method is proposed for likelihood computation of a particle from image data. This method utilizes sets of 3D points, generated around a geometric model of the object, which are a function of the object's dimensions and the particle's pose with respect to the camera.

Fig. 2a shows the geometric model of the object and shows the different sets of generated 3D points. In total, we generate thirteen sets of points for each particle: six sets of points laying on each face of the object (in green); six sets of points laying on each face of the object near the edges (in blue), and one set of points just outside the edges of the object (in red). These sets of points are projected to the image plane, resulting in corresponding sets of 2D points in image coordinates. Fig. 2b shows the projection of these sets of points to the image plane, given a state of the particle. Note that we disregard all the self-occluded points, i.e., the red points laying inside the contouring edges of the box, and blue and green points laying on surfaces that are not visible from the camera for this specific state. We then compute color histograms for each set of 2D points using the HSI color space as in [13], resulting in the color histograms $\boldsymbol{h}_i^{\text{cin}}$ (blue points) and $\boldsymbol{h}_i^{\text{face}}$ (green points) for each visible face i, and $\boldsymbol{h}^{\text{cout}}$ (red points). Furthermore, each face has an a-priori created reference color histogram $\boldsymbol{h}_i^{\text{ref}}$ computed from the known object texture maps. High likelihood values should then be assigned to particles if, for each visible face, the similarity between $\boldsymbol{h}_i^{\text{cin}}$ and $\boldsymbol{h}_i^{\text{ref}}$ as well as the similarity between $\boldsymbol{h}_i^{\text{face}}$ and $\boldsymbol{h}_i^{\text{ref}}$ is high, while at the same time the similarity between $\boldsymbol{h}_i^{\text{cin}}$ and $\boldsymbol{h}^{\text{cout}}$ is low. These similarities are computed using the Bhattacharyya similarity function, as described in [27], and we denote them by $\boldsymbol{\jmath}_1$, $\boldsymbol{\jmath}_2$, and $\boldsymbol{\jmath}_3$, respectively. Since the histograms are normalized, the similarities are in the range $\boldsymbol{\jmath} \in [0, 1]$, where $\boldsymbol{\jmath} = 1$ refers to a perfect match between histograms. We define



(a) (b)

Fig. 2: Generated 3D points around the geometric model (a) and their projection onto the image plane for a given state (b).

$$\mathscr{D} = \frac{\kappa_1 (1 - \boldsymbol{\jmath}_1) + \kappa_2 (1 - \boldsymbol{\jmath}_2) + \kappa_3 \boldsymbol{\jmath}_3}{\kappa_1 + \kappa_2 + \kappa_3} \tag{12}$$

with κ_1, κ_2, and κ_3 scaling parameters. In the ideal case, all terms in the numerator are zero when the particle is representing the true state of the object and the background and object colors are exclusive. Finally, the likelihood is modeled as a Laplacian distribution as in [26] to obtain $p(\mathbf{y}_t | \mathbf{x}_t^{(i)}) \propto e^{-\frac{|\mathscr{D}|}{\epsilon}}$, where ϵ is a scaling parameter.

B. Equations of motion

We now derive the motion model f used in (3). For the sake of brevity, we will consider only a single particle $\mathbf{x}_{t-1}$ at a given time $t-1$. We write its pose and velocity with respect to the inertial frame A in terms of the transformation matrix $\mathbf{H} = [\mathbf{R}, \mathbf{o}; \mathbf{0}, 1] \in \mathrm{SE}(3)$ and the *left-trivialized* velocity $\mathbf{v} = [\boldsymbol{v}; \boldsymbol{\omega}] \in \mathbb{R}^6$ as in [25]. This means that we write the state as $\mathbf{x}_{t-1} = (\mathbf{H}(t-1), \mathbf{v}(t-1))$ and compute the predicted state $\mathbf{x}_{t|t-1} = (\mathbf{H}(t), \mathbf{v}(t))$. To derive the equations of motion and unilateral contact and impact dynamics, we closely follow the framework of Glocker in [28]. We write the equations of motion on the level of momenta, to allow for inclusion of impulsive forces and discontinuities in the friction force, such that

$$\mathbb{M} d\mathbf{v}(t) + \mathbf{v}(t) \bar{\times}^* \mathbb{M} \mathbf{v}(t) dt = \mathbf{f}(t) dt + \mathbf{W}_N d\mathbf{P}_N + \mathbf{W}_T d\mathbf{P}_T, \tag{13}$$

where $\mathbb{M}$ denotes the inertia tensor, $\bar{\times}^*$ the dual-cross product on $\mathbb{R}^6$ as in [25], and $\mathbf{f}(t)$ the generalized wrench containing the generalized forces and torques applied to the center of mass of the body (except for the contact forces), which here involve the gravitational forces. Furthermore, $d\mathbf{P}_N$ and $d\mathbf{P}_T$ are respectively the differential measures of the momenta associated to the contact/impact in normal direction and tangential direction, respectively. These differential measures consist of a Lebesgue measurable part $\boldsymbol{\lambda} dt$ which denotes the non-impulsive part of the contact/friction force and an atomic part $\boldsymbol{\Lambda} d\eta$, denoting the impulsive part of the contact/friction force [28], such that $d\mathbf{P}_N = \boldsymbol{\lambda}_N dt + \boldsymbol{\Lambda}_N d\eta$ and $d\mathbf{P}_T = \boldsymbol{\lambda}_T dt + \boldsymbol{\Lambda}_T d\eta$. Furthermore, we can write the kinematics as

$$\dot{\mathbf{H}}(t) = \mathbf{H}(t) \mathbf{v}^{\wedge}(t), \tag{14}$$

with $(\cdot)^{\wedge}$ the *hat*-operator as in [25].

C. Modeling unilateral contact, impact, and friction

Consider again Fig. 1, where frame C defines the location and orientation of the contact surface and this frame is oriented such that the unit vector $\vec{z}_c$ defines the normal to the plane. We consider the eight vertices of the box to be the only potential contact points and denote them by $\mathbf{p}_i$. The contact distance, referred to as the *gap function*, is the distance in z-direction of C between the point $\mathbf{p}_i$ and the origin of C, defined as

$$g_{Ni} \coloneqq {}^A\mathbf{z}_C^T \Big(({}^A\mathbf{o}_B + {}^A\mathbf{R}_B {}^B\mathbf{p}_i) - {}^A\mathbf{o}_C \Big), \qquad (15)$$

and graphically depicted in Fig. 1. Under the assumption that the body and surface are rigid, we can write the constitutive law for unilateral contact using Signorini's contact law [28] using the *proximal point formulation* as in [29], such that

$$\boldsymbol{\lambda}_N = \text{prox}_{\mathcal{C}_N} (\boldsymbol{\lambda}_N - r\mathbf{g}_N) \quad \text{with} \quad r > 0, \qquad (16)$$

where $\mathcal{C}_N = \{\boldsymbol{\lambda}_N \in \mathbb{R}^{n_c} \mid \boldsymbol{\lambda}_N \geq \mathbf{0}\}$ is the set of admissible normal forces, $\mathbf{g}_N$ the column of normal contact distances, and n_c is the number of contact points. We will use Newton's impact law, given as $\gamma_{Ni}^+ = -e_N \gamma_{Ni}^-$, to relate the pre-impact normal velocity γ_{Ni}^- to the post-impact normal velocity γ_{Ni}^+ through the normal coefficient of restitution e_N [28]. We use Newton's model due to its simplicity and ease of implementation, but stress that it can lead to inconsistent results when applied in complex contact situations that can occur in real-world collision scenarios [30], [31]. The normal velocity is given by the time derivative of (15) such that $\gamma_{Ni} = \dot{g}_{Ni}$ almost everywhere, and we write it in terms of the left-trivialized velocity such that $\gamma_{Ni} = \mathbf{w}_{Ni}^T \mathbf{v} \in \mathbb{R}$, where $\mathbf{w}_{Ni}^T$ is the row-vector containing the force directions of the contact impulses of $\mathbf{p}_i$. We introduce the variable $\xi_{Ni} = \gamma_{Ni}^+ + e_{Ni}\gamma_{Ni}^-$ to formulate an impact law that relates the impulsive force Λ_{Ni} to γ_{Ni}^+ via a complementarity condition. This gives

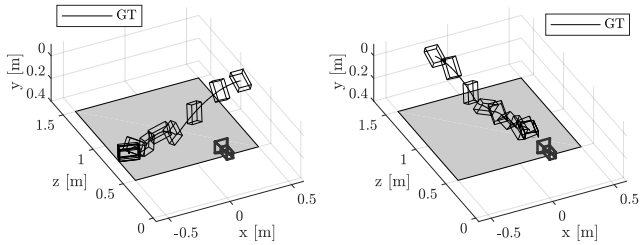
$$\Lambda_{Ni} = \text{prox}_{\mathcal{C}_N} (\Lambda_{Ni} - r\xi_{Ni}) \;\; \forall i \in \mathcal{I}_c, \qquad (17)$$

where $\mathcal{I}_c$ is the set of closed contacts, $\mathcal{C}_N = \mathbb{R}^+$, and $r > 0$. In order to model dry friction, we will use a set-valued Coulomb friction model as in [28]. We will define the tangential velocity $\boldsymbol{\gamma}_{Ti}$ as the velocity of $\mathbf{p}_i$ with respect to C in x- and y-direction of the contact plane such that $\boldsymbol{\gamma}_{Ti} = \mathbf{w}_{Ti}^T \mathbf{v} \in \mathbb{R}^2$, where $\mathbf{w}_{Ti}^T \in \mathbb{R}^{2\times n}$ is the matrix containing the force directions of the friction forces acting on $\mathbf{p}_i$. We can formulate the set-valued force law for the non-impulsive part of isotropic Coulomb friction as follows:

$$\boldsymbol{\lambda}_{Ti} = \text{prox}_{\mathcal{C}_{Ti}} (\boldsymbol{\lambda}_{Ti} - r\boldsymbol{\gamma}_{Ti}), \qquad (18)$$

where $\mathcal{C}_{Ti} = \{\boldsymbol{\lambda}_{Ti} \mid \|\boldsymbol{\lambda}_{Ti}\| \leq \mu\lambda_{Ni}\} \quad \forall i \in \mathcal{I}_c$ and $r > 0$. To account for impulsive friction forces associated with a certain amount of restitution, we write the set-valued force laws for Coulomb-friction in terms of momenta. Therefore, we introduce the variable $\boldsymbol{\xi}_{Ti} = \boldsymbol{\gamma}_{Ti}^+ + e_T \boldsymbol{\gamma}_{Ti}^- \in \mathbb{R}^2$ where e_T denotes the tangential coefficient of restitution. The set-valued force law for $\boldsymbol{\xi}_{Ti}$ and $\boldsymbol{\Lambda}_{Ti}$ then becomes

$$\boldsymbol{\Lambda}_{Ti} = \text{prox}_{\mathcal{C}_{Ti}} (\boldsymbol{\Lambda}_{Ti} - r\boldsymbol{\xi}_{Ti}) \qquad (19)$$



(a) (b)

Fig. 3: Ground truth trajectories. First scenario (a) and second scenario (b).

with $\mathcal{C}_{Ti} = \{\boldsymbol{\Lambda}_{Ti} \mid \|\boldsymbol{\Lambda}_{Ti}\| \leq \mu\Lambda_{Ni}\} \quad \forall i \in \mathcal{I}_c$ and $r > 0$.

D. Numerical integration using time-stepping

We use a timestepping scheme as in [29] for numerical integration of the equations of motion. The advantage of the time-stepping method is that, when integrating the equations of motion (13) over a time-interval possibly containing one or more time instants at which impulsive loads occur, the contributions of the smooth forces and impulsive forces are both taken into account, which avoids the need for event detection of the impacts and allows to perform simulation beyond Zeno events [29]. At the beginning of each time step, the pose $\mathbf{H}(t-1)$ of the object and the velocity $\mathbf{v}(t-1)$ are known from $\mathbf{x}_{t-1}$, either from initial conditions or computed from the previous time step. The velocity and pose at the end of the time-step are computed using (13) and (14), respectively. The values for $d\mathbf{P}_N$ and $d\mathbf{P}_T$ are found by solving the so-called *contact problem*, defined by the set-valued force laws of (16), (17), (18), and (19). We use an augmented Lagrangian approach as discussed in [29] to solve this contact problem, due to its simplicity and effectiveness in our specific case of a single rigid body. In summary, the time-stepping scheme consists of the following steps:

1. The pose $\mathbf{H}(t-1)$ of the object and the left trivialized velocity $\mathbf{v}(t-1)$ are known at the beginning of the time step from the state $\mathbf{x}_{t-1}$;
2. The system matrices $\mathbf{f}$, $\mathbf{w}_{Ni}$, and $\mathbf{w}_{Ti}$ are computed, from which γ_{Ni} and γ_{Ti} follow;
3. The values for $\mathbf{v}$, $\boldsymbol{\lambda}_N$, $\boldsymbol{\lambda}_T$, $\boldsymbol{\Lambda}_N$, and $\boldsymbol{\Lambda}_T$ at the end of the time step are computed using an augmented Lagrangian approach as in [29];
4. The pose of the object is updated to the next time step according to (14) and form together with the velocity at the end of the time-step the state $\mathbf{x}_{t|t-1} = (\mathbf{H}(t), \mathbf{v}(t))$, which can be rewritten to an element of $\mathcal{S}$.

V. Numerical validation

We validate the proposed method by simulations[1] and compare its performance to that of a state-of-the-art tracking algorithms presented in [9], [16] and [17]. These works use a standard with a constant velocity motion model, and therefore differ from our proposed method both in filtering technique and motion model. We consider two tossing

[1]The source code is available at https://gitlab.tue.nl/robotics-lab-public/impact-aware-object-tracking.

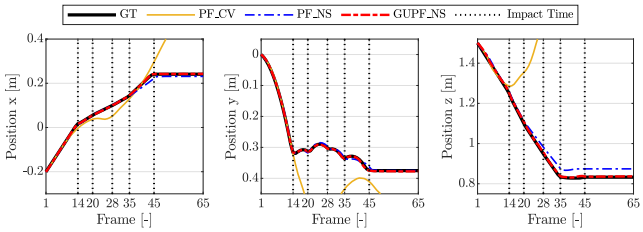


Fig. 4: Tracking results of the position for the various algorithms of the second scenario. The x-position (left), y-position (middle), and z-position (right) of the results are plotted together with the ground truth trajectory.

TABLE I: Resulting RMS values and error reduction of the second scenario.

Method	$\|e_{\mathbf{o}}\|_{RMS}$ [m]	Red. [%]	$\|e_{\mathbf{R}}\|_{RMS}$ [deg]	Red. [%]
PF_CV	0.931	-	130.6	-
PF_NS	0.044	95.3	13.2	89.9
GUPF_NS	0.007	99.2	2.0	98.4

scenarios for which ground truth trajectories and artificial images are created. In the first trajectory, the motion of the object is parallel to the camera image, while in the second trajectory the object moves towards the camera. This allows us to test the effect of camera orientation with respect to the object's motion. This means that the two trajectories can be seen as two extreme cases, and we assume that other camera orientations will give similar results. We can write these trajectories by means of the pose of the box as $\mathbf{GT}_t = ({}^A\mathbf{R}(t)_B^{GT}, {}^A\mathbf{o}(t)_B^{GT})$ and Fig. 3 shows these trajectories, where the black lines indicate the trajectories of the center of mass of the box. Furthermore, for the first and every fifth frame the pose of the box is shown and the camera is illustrated. Assuming a camera frame rate of 60FPS, we create a total of 65 artificial images for each trajectory.

In [9], [16], and [17], a constant velocity model is used to describe the state transition on SE(3). In this constant velocity model, no preference is given to any direction of motion such that the equations of motion can be written as

$$\mathbf{H}(t_e) = \mathbf{H}(t_a)\mathrm{Exp}\left(\Delta t \mathbf{v}^{\wedge}(t_a) + \sqrt{\Delta t}\mathbf{a}^{\wedge}\right), \tag{20}$$

$$\mathbf{v}^{\wedge}(t_e) = \frac{1}{\Delta t}\mathrm{Log}\left(\mathbf{H}(t_a)^{-1}\mathbf{H}(t_e)\right), \tag{21}$$

where $\mathbf{a} \in \mathbb{R}^6$ is a zero-mean white noise vector, t_a and t_e represent respectively the time at the beginning and end of the time-step, and $\Delta t = 1/60$ seconds, corresponding to the assumption of a 60fps camera. We implement this motion model in a PF on Lie groups, as in [9], [32], [33], and refer to this as the PF with Constant Velocity model (PF_CV). Similarly, we implement our proposed nonsmooth motion model in a PF, and refer to this as the PF with Nonsmooth model (PF_NS). Finally, we implement our proposed motion model in the GUPF, and refer to this as GUPF_NS.

In the simulations described in this section, we use a set of 500 particles for each simulation. To initialize the filter, one could use the initial ground truth state as initial state for each

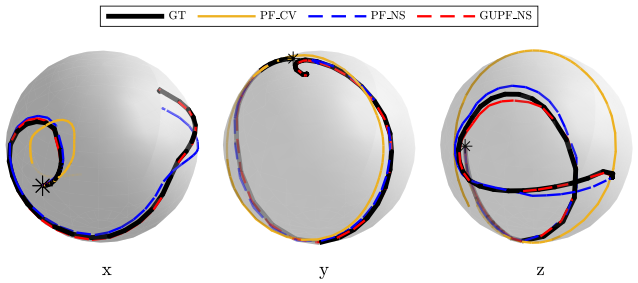


Fig. 5: Tracking results of the orientation for the various algorithms. We plot the evolution of the x, y, and z axis of the coordinate frame B by plotting the first, second, and third column of the rotation matrix ${}^A\mathbf{R}(t)_B$ on the unit sphere. The asterisks indicate the starting point of the simulation.

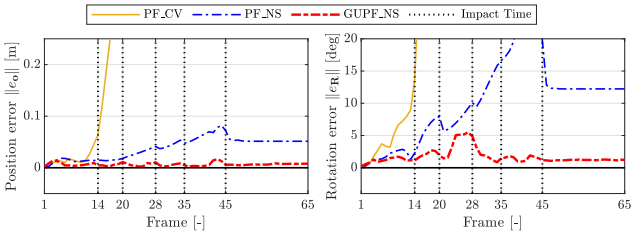


Fig. 6: Tracking errors for the various algorithms for the second scenario.

of these particles. However, since the ground truth data and the filter are based on the same motion model, this would give the filter perfect knowledge of the motion of the object. Instead, we sample the 500 particles from a distribution centered around the ground truth state such that each particle is initialized with a pose and velocity that is different from the ground truth. To this end, we take the initial ground truth state as the initial state mean and sample a set of 500 particles around that mean using (1). The artificial images are used as input to the different algorithms for likelihood computation and obtaining a measurement at each time-step. As a result, these algorithms give as output the estimated position and orientation of the box frame B with respect to the camera coordinate frame A. We can plot both the resulting position and orientation from the different algorithms together with the ground truth and this is shown in Figures 4 and 5. We have plotted the results for the second scenario of Fig. 3 only, due to space limitations, but both scenarios give qualitatively similar results. To visualize the orientation in Fig. 5, we plot the trajectories of the x, y, and z axis of the coordinate frame B by plotting the first, second, and third column of the rotation matrix ${}^A\mathbf{R}(t)_B$ on the unit sphere, where the asterisks indicate the starting point of the simulation. Next, we define the error in position between the results of the filter and the ground truth as $e_{\mathbf{o}}$. We define the error in orientation between the results of the filters and the ground truth as $e_{\mathbf{R}}$, which gives the error as an element of $\mathfrak{so}(3)$. We can take its norm, denoted by $\|e_{\mathbf{R}}\|$, to obtain the angular error in radians, which can then easily be converted into degrees. We have plotted these results in Fig. 6.

In Fig. 4, we observe that the PF_CV loses track of the box at frame 14, right after the first impact. The constant velocity model is unable to predict the abrupt changes in

velocity imposed by the impacts, and the particles drift away from the true state. In Fig. 4 and Fig. 5, we observe that the PF_NS already provides superior tracking performance over the PF_CV, as a result of incorporating the proposed motion model. However, when considering Fig. 6, we clearly see the benefit of the GUPF_NS, which outperforms the PF_NS by taking the current observation into account, therefore bringing particles closer to the true state. Table I shows the RMS errors for the different filters and shows for the PF_NS and GUPF_NS the error reduction with respect to the PF_CV. This shows that both PF_NS and the GUPF_NS have superior tracking performance with respect to the PF_CV.

VI. CONCLUSIONS

In this paper, we have shown that state-of-the-art tracking algorithms, making use of standard Particle Filters and constant velocity models, tend to lose track of an object that experiences abrupt changes in velocity imposed by impacts. To tackle this problem, we propose to incorporate potential collision priors within a suitable filter and create a motion model which describes the nonsmooth dynamics of an impacting object. We incorporate this model in a PF and show that, even though tracking performance increases, the PF still loses track of the object, as it does not take current observations into account. We therefore propose a new filter, the Geometric Unscented Particle Filter, which is an extension of the UPF for systems whose state lives in a Lie group, and apply this filter with the proposed nonsmooth motion model. We show, in simulations, that our proposed methods outperform the state of the art in terms of tracking accuracy. In current research, we are applying the proposed methods on a real setup to obtain experimental confirmation of the findings obtained in simulation.

REFERENCES

[1] Y.-B. Jia, M. Gardner, and X. Mu, "Batting an in-flight object to the target," *The International Journal of Robotics Research*, vol. 38, no. 4, pp. 451–485, 2019.

[2] S. Olufs, F. Adolf, R. Hartanto, and P. Plöger, "Towards Probabilistic Shape Vision in RoboCup: A Practical Approach," in *RoboCup 2006: Robot Soccer World Cup X* (G. Lakemeyer, E. Sklar, D. G. Sorrenti, and T. Takahashi, eds.), (Berlin, Heidelberg), pp. 171–182, Springer Berlin Heidelberg, 2007.

[3] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic, "CosyPose: Consistent multi-view multi-object 6D pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[4] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images," 2019.

[5] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. W. Jr., A. Rodriguez, and J. Xiao, "Multi-view Self-supervised Deep Learning for 6D Pose Estimation in the Amazon Picking Challenge," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[6] E. Sucar, K. Wada, and A. Davison, "NodeSLAM: Neural Object Descriptors for Multi-View Shape Reconstruction," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2020.

[7] V. Lepetit and P. Fua, "Monocular Model-based 3D Tracking of Rigid Objects," *Foundations and Trends in Computer Graphics and Vision*, vol. 1, pp. 1–89, Jan. 2005.

[8] C. Harris, "Tracking with Rigid Objects." MIT Press, 1992.

[9] C. Choi and H. I. Christensen, "Robust 3D visual tracking using particle filtering on the special Euclidean group: A combined approach of keypoint and edge features," *The International Journal of Robotics Research*, vol. 31, no. 4, pp. 498–519, 2012.

[10] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao–Blackwellized Particle Filter for 6-D Object Pose Tracking," *IEEE Transactions on Robotics*, pp. 1–15, 2021.

[11] A. Doucet, N. De Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Springer New York, 1st. ed., 2001.

[12] O. Cappe, S. J. Godsill, and E. Moulines, "An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[13] M. Taiana, J. Santos, J. Gaspar, J. Nascimento, A. Bernardino, and P. Lima, "Tracking Objects with Generic Calibrated Sensors: An Algorithm Based on Color and 3D Shape Features," *Robot. Auton. Syst.*, vol. 58, pp. 784–795, Jun. 2010.

[14] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe, "A Boosted Particle Filter: Multitarget Detection and Tracking," in *Computer Vision - ECCV 2004*, (Berlin, Heidelberg), pp. 28–39, Springer Berlin Heidelberg, 2004.

[15] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. Wan, "The Unscented Particle Filter," in *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS '00, (Cambridge, USA), pp. 563–569, Aug. 2000.

[16] J. Kwon and K. Lee, "Monocular SLAM with Locally Planar Landmarks via Geometric Rao-Blackwellized Particle Filtering on Lie Groups," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1522–1529, Jun. 2010.

[17] Y. K. Baik, J. Kwon, H. S. Lee, and K. M. Lee, "Geometric particle swarm optimization for robust visual ego-motion estimation via particle filtering," *Image and Vision Computing*, vol. 31, no. 8, pp. 565–579, 2013.

[18] G. Marjanovic and V. Solo, "An engineer's guide to particle filtering on matrix Lie groups," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3969–3973, 2016.

[19] M. Brossard, S. Bonnabel, and J. Condomines, "Unscented Kalman filtering on Lie groups," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2485–2491, Sep. 2017.

[20] V. Varadarajan, *Lie Groups, Lie Algebras, and Their Representations*, vol. 102. Spinger-Verlag New York, 1984.

[21] T. D. Barfoot and P. T. Furgale, "Associating Uncertainty With Three-Dimensional Poses for Use in Estimation Problems," *IEEE Transactions on Robotics*, vol. 30, pp. 679–693, Jun. 2014.

[22] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups*, vol. 2. Birkhäuser New York, NY, USA, 2012.

[23] M. Zeestraten, *Programming by Demonstration on Riemannian Manifolds*. PhD thesis, University of Genova, 2017.

[24] G. Dubbelman, *Intrinsic statistical techniques for robust pose estimation*. PhD thesis, University of Amsterdam, Sep. 2014.

[25] S. Traversaro and A. Saccon, *Multibody dynamics notation (version 2)*. Technische Universiteit Eindhoven, Nov. 2019. DC2019.100.

[26] M. Taiana, J. Nascimento, J. Gaspar, and A. Bernardino, "Sample-Based 3D Tracking of Colored Objects: A Flexible Architecture," in *British Machine Vision Conference (BMVC)*, Jan. 2008.

[27] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 142–149, 2000.

[28] C. Glocker, *Set-Valued Force Laws: Dynamics of Non-Smooth Systems*, vol. 1. Springer-Verlag Berlin Heidelberg, Jan. 2001.

[29] R. Leine and H. Nijmeijer, *Dynamics and bifurcations of non-smooth mechanical systems*, vol. 18 of *Lecture Notes in Applied and Computational Mechanics*. Germany: Springer-Verlag Berlin Heidelberg, 2004.

[30] Y. Wang and M. T. Mason, "Two-Dimensional Rigid-Body Collisions With Friction," *Journal of Applied Mechanics*, vol. 59, pp. 635–642, 09 1992.

[31] T. K. Uchida, M. A. Sherman, and S. L. Delp, "Making a meaningful impact: modelling simultaneous frictional collisions in spatial multibody systems," *Proc Math Phys Eng Sci*, vol. 471, May 2015.

[32] J. Kwon, M. Choi, F. C. Park, and C. Chun, "Particle filtering on the Euclidean group: framework and applications," *Robotica*, vol. 25, p. 725–737, Jun. 2007.

[33] A. Chiuso and S. Soatto, "Monte Carlo filtering on Lie groups," in *Proceedings of the IEEE Conference on Decision and Control*, vol. 1, pp. 304 – 309, Feb. 2000.