Robust point-mass filters on manifolds

Henrik Tidefelt [*,⋆] **Thomas B. Schön** [*,⋆⋆]

[*] *Division of Automatic Control, Linköping University, Sweden (e-mail:* `{tidefelt,schon}@isy.liu.se`*).*

Abstract: Robust state estimation for states evolving on compact manifolds is achieved by employing a point-mass filter. The proposed implementation emphasizes a sane treatment of the geometry of the problem, and advocates separation of the filtering algorithms from the implementation of particular manifolds.

Keywords: State estimation, manifold, non-parametric, point-mass filter

1. INTRODUCTION

State estimation on manifolds is commonly performed by embedding the manifold in a linear space of higher dimension, combining estimation techniques for linear spaces with some projection scheme [Brun et al., 2007, Törnqvist et al., 2008, Crassidis et al., 2007, Lo and Eshleman, 1979]. Obvious drawbacks of such schemes are that computations are carried out in the wrong space, and that the arbitrary choice of embedding has an undesirable effect on the projection operation. Another common approach is to let the filter run in a linear space of local coordinates on the manifold. Drawbacks include the local nature of coordinates, the non-linearities introduced by the curved nature of the manifold, and the dependency on the choice of coordinates. Despite the drawbacks of these two approaches, it should be admitted that they work well for many "natural" choices of embeddings and local coordinates, as long as the uncertainty about the state is concentrated to a small — and hence approximately flat — part of the manifold. Still, the strong dependency on embeddings and local coordinates suggests that the estimation algorithms are not defined within the appropriate framework. The Monte-Carlo technique called the *particle filter* lends itself naturally to a coordinate-free formulation (as in Kwon et al. [2007]). However, the stochastic nature of the technique makes it unreliable, and addressing this problem motivates the word *robust* in the title of this work. With a growing geometric awareness among state estimation practitioners, geometrically sound algorithms tailored for particular applications are emerging. A very common application is that of orientations of physical objects (for instance, Lee and Shin [2002]), and this is also a guiding application in our work.

Our interest in this work is to examine how robust state estimation on compact manifolds of low dimension can be performed while honoring the geometric nature of the problem. The robustness should be with respect to uncertainties which are *not* concentrated to a small part of the manifold, and is obtained by using a non-parametric representation of stochastic variables on the manifold. By *honoring the geometric nature* we mean that we intend to minimize references to embeddings and local coordinates in our algorithms. We say *minimize* since, under a layer of abstraction, we too will employ embeddings to implement the manifold structure, and local coordinates are the natural way for users to interact with the filter. Still, the proposed framework for state estimation can be characterized by the abstraction barrier that separates the details of the embedding from the filter algorithm. For example, in the context of estimation of orientations, rather than speaking of filters for unit quaternions or rotation matrices, this layer of abstraction enables us to simply speak of filters for $\mathcal{SO}(3)$ — both unit quaternions and rotation matrices may be used to implement the low-level details of the manifold structure, but this is invisible to the higher-level estimation algorithm.

Pursuing non-parametric filtering in curved space comes at some computational costs compared to the linear space setting. Most notably, equidistant meshes do not exist, but on the other hand our restriction to compact manifolds means that the whole manifold can be "covered" by a mesh with finitely many nodes. One of the practical benefits of the proposed non-parametric filter is the ability to dynamically adapt the mesh to enhance the degree of detail in regions of interest, for instance, where the probability density is high.

The proposed *point-mass*-based solution for filtering in curved space has three main components:

- Compute — and possibly update — a tessellation of the manifold.
- Implement measurement and time updates. This requires a system model which, unlike when filtering in Euclidean space, cannot have additive noise on the state.
- Provide the user with a point estimate. There is always the option to compute a cheap *extrinsic* estimate (typically the extrinsic mean), but honoring geometric reasoning in this work, we also look into *intrinsic* estimates.

Where our general treatment lacks detail, we will include specialization to the case of spheres.

Terminology. By *manifold*, we refer to a differentiable, Riemannian manifold. Loosely speaking, a (contravariant) *vector* is a velocity on the manifold, belonging to the *tangent space* (which is a vector space) at some point on the manifold, and is basically valid only at that point. A curve on the manifold which locally connects points along the shortest path between the points, is called a *geodesic*, and the *exponential map* maps vectors to points on the manifold in such a way that, for a vector $\boldsymbol{v}$ at p, the curve $t \mapsto \mathrm{e}_p^{t\boldsymbol{v}}$ has velocity $\boldsymbol{v}$ at $t = 0$, and is a

⋆ With financial support from the Swedish Research Council.

⋆⋆This work was supported by the strategic research center MOVIII, funded by the Swedish Foundation for Strategic Research, SSF.

geodesic. When needed, we shall assume that the manifold is *geodesically complete*, meaning that the exponential map shall be defined for all vectors. We recommend Frankel [2004] for an introduction to these concepts from differential geometry. A *tessellation* of the manifold is a set $\{ R^i \}_i$ of subsets of the manifold, such that "there is no overlay and no gap" between regions; the union of all regions shall be the whole manifold, and the intersection of any two regions shall have measure zero. We shall additionally require that each region R^i be simply connected.

Notation. The manifold on which the estimated state evolves is denoted $\mathcal{M}$. We make no distinction in notation between ordinary and stochastic variables; x my refer both to a stochastic variable over the manifold and a particular point on the manifold. The probability of a statement, such as $x \in R$, is written $\mathrm{P}(x \in R)$. The probability density function for a stochastic variable x is written f_x. When conditioning on a variable taking on a particular value, we usually drop the stochastic variable from the notation; for instance, $\mathsf{f}_{x|y}$ is a shorthand for $\mathsf{f}_{x|Y=y}$, where the distinction between the stochastic variable, Y, and the value it takes, y, had to be made clear. The distance in the induced Riemannian metric, between the points x and y, is written $\mathrm{d}(x, y)$. The symbol δ is used to denote the Dirac delta "function". A Gaussian distribution over a vector space, with mean m and covariance C, is denoted $\mathcal{N}(m, C)$, and if the variable x is distributed according to this distribution, we write $x \sim \mathcal{N}(m, C)$. (The covariance is a symmetric, positive semidefinite, linear mapping of pairs of vectors to scalars, and it should be emphasized that a covariance is basically only compatible with vectors at a certain point on the manifold.) In relation to plans for future work, we should also mention that group structure on the manifold is *not* used in this work, although such manifolds, *Lie groups*, are often a suitable setting for estimation of dynamic systems.

2. BACKGROUND AND RELATED WORK

For models with continuous-time dynamics, the evolution of the probability distribution of the state is given by the Fokker-Planck equation, and a great amount of research has been aimed at solving this partial differential equation under varying assumptions and approximation schemes. Daum [2005] gives a good overview that should be accessible to a broad audience. In the present discrete-time setting, the corresponding relation is the Chapman-Kolmogorov equation. It tells how the distribution of the state at the next time step (given all available measurements up till the present) depends on the distribution of the state at the current time step (given all available measurements up till the present) and the *process noise* in the model. Let $y_{0..t}$ be the measurements up to time t, and $x_{s|t}$ be the state at time s given $y_{0..t}$. Conditioned on the measurements $y_{0..t}$, and using that x_{t+1} is conditionally independent of $y_{0..t}$ given x_t, the Chapman-Kolmogorov equation states the familiar

$$\mathsf{f}_{x_{t+1|t}}(x_{t+1}) = \int \mathsf{f}_{x_{t+1}|x_t}(x_{t+1})\, \mathsf{f}_{x_{t|t}}(x_t)\, \mathrm{d}x_t \qquad (1)$$

In combination with Bayes' rule for taking the information in new measurements into account,

$$\mathsf{f}_{x_{t|t}}(x_t) = \frac{\mathsf{f}_{x_{t|t-1}}(x_t)\, \mathsf{f}_{y_t|x_t}(y_t)}{\mathsf{f}_{y_{t|t-1}}(y_t)} \qquad (2)$$

this describes exactly the equations that the discrete-time filtering problem is all about.

To mention just a few references for the particular application of filtering on $\mathcal{SO}(3)$, a filter for random walks on the tangent bundle (with the only system noise being additive noise in the Lie algebra corresponding to velocities) was developed in Chiuso and Soatto [2000], a quaternion representation was used with projection and a Kalman filter adapted to the curved space in Choukroun et al. [2006], and Lee et al. [2008] proposes a method to propagate uncertainty under continuous time dynamics in a noise-free setting. The particle filter approach in Kwon et al. [2007] has already been mentioned.

A solid account of the most commonly used methods for filtering on $\mathcal{SO}(3)$ is provided by Crassidis et al. [2007]. In Lo and Eshleman [1979] the authors presents an interesting representation of probability density functions on $\mathcal{SO}(3)$, making use of exponential Fourier densities.

3. DYNAMIC SYSTEMS ON MANIFOLDS

The filter is designed to track the discrete time stochastic process x, evolving on some manifold of low dimension. That the dimension is low is instrumental to enabling the use of filter techniques that, in higher dimensions, break down performance-wise due to the *curse of dimensionality* [Bergman, 1999, section 5.1]. We use discrete time models in the form

$$x_{t+1} \sim W_{g(x_t, u_t)}$$
$$y_t \sim V_{x_t}$$

where $W_{g(x_t, u_t)}$ is the random distribution of process noise taking values on the manifold, u_t is a known external input, and the measurement y_t is distributed according to the random distribution V_{x_t}. Not being aware of a standard name for a distribution over the manifold, parameterized by a point on the same manifold, we shall use *distribution field* for $W_\bullet$ (here, the bullet indicates that there is a free parameter — for a fixed value of this parameter, we have an ordinary random distribution).

For example, the measurement equation could be given by

$$V_{x_t} = \mathcal{N}(h(x_t), C_{\mathrm{y}}(x_t))$$

That is, we have additive Gaussian white noise added to the nominal measurements $h(x_t)$, and we allow the noise covariance to depend on the state.

A less general example of the dynamic equation could be to combine Gaussian distributions with the exponential map

$$x_{t+1} \sim \exp \mathcal{N}\big(0, C_{g(x_t, u_t)}\big)$$

Here, $\mathcal{N}\big(0, C_{g(x_t, u_t)}\big)$ is our way of denoting a zero mean Gaussian distribution of vectors at $g(x_t, u_t)$. However, (without the structure of a Lie group) the simplicity of this expression is misleading, since the Gaussian distributions at different points on the manifold are defined in different tangent spaces. Hence, a common matrix will not be sufficient to describe the covariance in all points.

To really obtain simple equations for the dynamic equation, we may employ distributions that only depend on the distance

$$\mathsf{f}_{x_{t+1}}(x_{t+1}) = f_{\mathrm{d}}(\mathrm{d}(x_{t+1}, g(x_t, u_t)))$$

4. POINT-MASS FILTER

The main idea of the point-mass filter is to model the probability distribution of the state x being estimated as a sum of weighted Dirac delta functions. The Dirac deltas are located at fixed positions in a uniform grid, and the idea dates back to the

seminal work by Bucy and Senne [1971]. When the filter is run, a sequence of such random variables will be produced and there is a need to distinguish between the variables before and after measurement and time updates, recall the notation introduced in section 2.

Readers familiar with the particle filter will notice many similarities to the proposed filter, but should also pay attention to the differences. To mention a few, the proposed filter is deterministic (and in this sense robust), does not require resampling, associates each probability term (compare *particle*) with a region in the domain of the estimated variable, and calculates with the volumes of these regions. One notable drawback compared to the particle filter is that when the estimated probability is concentrated to a small part of the domain, the particle filter will automatically adapt to provide estimates with smaller uncertainty, while the proposed filter would require a non-trivial extension to do so.

In this section, we first discuss the representation of stochastic variables, and then turn to deriving equations for the time and measurement updates, expressed using the proposed representation.

4.1 Point-mass distributions on a manifold

In this section, we consider how any random variable on the manifold may be represented, and omit time subscripts to keep notation clear. That the idea is termed *point-mass* is due to the sometimes used assumption that the probability is distributed discretely at certain points. Written using the Dirac delta, the probability density function for x is then given by

$$\mathsf{f}_x(\, x \,) = \sum_i p^i \, \delta(\, x - x^i \,)$$

where the sum is over some finite number of points with probability p^i located at x^i. While this makes several operations on the distribution feasible, which would be extremely computationally demanding using other models, this is clearly very different from what we would expect the density function to look like.

To be able to make other interpretations of the pairs $(\, p^i, \, x^i \,)$, each such pair needs to be associated with a region R^i of the probability space, and we require that the set of regions, $\left\{ \, R^i \, \right\}_i$, be a tessellation. Let $\mu^i = \mu\left(\, R^i \, \right)$, where $\mu(\, \bullet \,)$ measures volume.

Given a tessellation $\left\{ \, R^i \, \right\}_i$ (of cardinality N), a more relaxed interpretation of the probabilities p^i is obviously

$$\mathrm{P}\big(\, x \in R^i \, \big) = p^i \tag{3}$$

and a more realistic model of the distribution is that it is piecewise constant;

$$\mathsf{f}_x(\, x \,) = \sum_{i \,:\, x \in R^i} \frac{p^i}{\mu^i}$$

Given the tessellation, including the μ^i, it is clear that the numbers p^i my be replaced by $f^i \triangleq \frac{p^i}{\mu^i}$. Since this is a more natural representation of piecewise constant functions in general, we choose to use this also for the probability density function estimate. For completeness, we state the above equations again, now using f^i instead of p^i:

$$\mathrm{P}\big(\, x \in R^i \, \big) = f^i \, \mu^i \tag{4}$$

$$\mathsf{f}_x(\, x \,) = \begin{cases} \displaystyle\sum_i f^i \, \mu^i \, \delta(\, x - x^i \,) \,, & \text{(Point-mass)} \\ \displaystyle\sum_{i \,:\, x \in R^i} f^i \,, & \text{(Piecewise constant)} \end{cases} \tag{5}$$

The point-mass filter is a *meshless* method in that it does not make use of a connection graph describing neighbor relations between the nodes x^i. (A connection graph is implicit in the tessellation, but it is not used.) While meshless methods in many finite element method applications would use interpolation (of, for instance, Sibson or Laplace type, see Sukumar [2003] for an overview of these) instead of the piecewise constant (5), our choice makes it easy to ensure that the density is non-negative and integrates to 1. Furthermore, both computation of the interpolation itself, and use of the interpolated density, would drastically increase the computational cost of the algorithm.

It turns out that computing good tessellations is a major task of the implementation of point-mass filters on manifolds, just like mesh generation is a major task when using finite element methods. It may also be a time-consuming task, but a basic implementation may do this once for all, offline. Since the number of regions greatly influences the runtime cost of the filter, a tessellation computed offline will have to be rather coarse. For models where large uncertainty is inherent in the filtering problem, this may be sufficient, but if noise levels are low and accurate estimation is theoretically achievable, the tessellation should be adapted to have smaller regions in areas where the probability density is high. [1]

If each region R^i is given as the set of points being closer to x^i than to all other $x^{j \neq i}$, the tessellation is called a *Voronoi diagram* of the manifold (in case of the 2-sphere, see for instance Augenbaum and Peskin [1985], Na et al. [2002]). Since this will make the point-mass interpretation more reasonable, it seems to be a desirable property of the tessellation, although a formal investigation of this strategy remains a relevant topic for future research.

4.2 Measurement update

Just as for particle filters, the measurement update is a straightforward application of Bayes' rule. To incorporate a new measurement of the random variable $y \sim V_x$ modeling the output, we have

$$\begin{aligned} \mathrm{P}\big(\, x \in R^i \, | \, y \, \big) &= \frac{\mathsf{f}_{y | x \in R^i}(\, y \,) \, \mathrm{P}\big(\, x \in R^i \, \big)}{\mathsf{f}_y(\, y \,)} \\ &\approx \frac{\mathsf{f}_{y | x = x^i}(\, y \,) \, \mathrm{P}\big(\, x \in R^i \, \big)}{\mathsf{f}_y(\, y \,)} \end{aligned}$$

where the measurement prior $\mathsf{f}_y(\, y \,)$ need not be known since it is a common factor to all probabilities on the mesh, and will just act as a normalizing constant. Converting to our favorite representation f^i, adding time indices, conditioning on $y_{0..t-1}$, and using conditional independence of y_t and $y_{0..t-1}$ given x_t, this reads

$$f^i_{t|t} = \frac{\mathrm{P}\big(\, x_t \in R^i \, | \, y_{0..t} \, \big)}{\mu^i} \approx \frac{\mathsf{f}_{y_t | x_t = x^i}(\, y_t \,) \; f^i_{t|t-1}}{\mathsf{f}_{y_{t|t-1}}(\, y_t \,)}$$

[1] This statement is based on intuition; it is a topic for future research to provide a theoretical foundation for how to best adapt the tessellation.

By defining

$$\mathrm{BayesRule}(\, f,\, g\,) \triangleq \frac{f\, g}{\int f\, g}$$

and noting that the result will always be a proper probability distribution (and hence integrate to 1, just as the result of the BayesRule operator) we can write:

$$\mathsf{f}_{x_{t|t}} = \mathrm{BayesRule}\big(\, \mathsf{f}_{x_{t|t-1}},\, \mathsf{f}_{y_t \,|\, x=\bullet}(\, y_t\,)\,\big)$$

Note how the volumes of regions enter the computation of the BayesRule operator:

$$\mathrm{BayesRule}(\, f,\, g\,)\,(\, x^i\,) \approx \frac{f(\, x^i\,)\, g(\, x^i\,)}{\sum_j f(\, x^j\,)\, g(\, x^j\,)\, \mu^j} \tag{6}$$

4.3 Time update

The time update can be described by the relation

$$\mathrm{P}\big(\, x_{t+1} \in R^i\,\big) = \int_{\mathcal{M}} \int_{R^i} \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}(\, x'\,)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x'\, \mathrm{d}x_t$$

In the filtering application, the stochastic entities in this relation will be conditioned on $y_{0..t}$, but since the conditioning is the same on both sides, it may be dropped for the sake of a more compact notation in this section. By the mean value theorem, we find

$$\mathrm{P}\big(\, x_{t+1} \in R^i\,\big) = \int_{\mathcal{M}} \mu^i\, \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}(\, x'\,)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x_t$$

for some $x' \in R^i$, and dividing both sides by μ^i and fitting the region in a shrinking ball centered at x^i, we obtain

$$\frac{\mathrm{P}\big(\, x_{t+1} \in R^i\,\big)}{\mu^i} \to \mathsf{f}_{x_{t+1}}\big(\, x^i\,\big)$$

and

$$\int_{\mathcal{M}} \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}(\, x'\,)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x_t \to \int_{\mathcal{M}} \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}\big(\, x^i\,\big)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x_t$$

Hence we obtain the Chapman-Kolmogorov equation (1) in the limit,

$$\mathsf{f}_{x_{t+1}}\big(\, x^i\,\big) = \int_{\mathcal{M}} \mathsf{f}_{x_t}(\, x_t\,)\; \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}\big(\, x^i\,\big)\; \mathrm{d}x_t$$

and this we make the definition of the convolution:

$$\mathsf{f}_{x_{t+1}} = \mathsf{f}_{x_t} * \mathsf{f}_{W_{g(\, \bullet,\, u_t\,)}}$$

The convolution of a distribution field and a probability density function is a new probability density function. We shall think of the time update as implementing this relation.

By approximating the probability density functions as constant over small regions (assuming all the regions R^i are *small*), we get the time update approximation

$$\begin{aligned}
\mathrm{P}\big(\, x_{t+1} \in R^i\,\big) &= \int_{\mathcal{M}} \int_{R^i} \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}(\, x'\,)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x'\, \mathrm{d}x_t \\
&\approx \mu^i \sum_j \int_{R^j} \mathsf{f}_{W_{g(\, x_t,\, u_t\,)}}\big(\, x^i\,\big)\; \mathsf{f}_{x_t}(\, x_t\,)\; \mathrm{d}x_t \\
&\approx \mu^i \sum_j \mathsf{f}_{W_{g(\, x^j,\, u_t\,)}}\big(\, x^i\,\big)\; \mathrm{P}\big(\, x_t \in R^j\,\big)
\end{aligned}$$

This is readily converted to an implementation of the convolution (here, the conditioning is written out for future reference):

$$\begin{aligned}
f^i_{t+1|t} &= \frac{\mathrm{P}\big(\, x_{t+1} \in R^i \mid y_{0..t}\,\big)}{\mu^i} \\
&\approx \sum_j \mathsf{f}_{W_{g(\, x^j,\, u_t\,)}}\big(\, x^i\,\big)\; \frac{\mathrm{P}\big(\, x_t \in R^j \mid y_{0..t}\,\big)}{\mu^j}\, \mu^j \\
&= \sum_j \mathsf{f}_{W_{g(\, x^j,\, u_t\,)}}\big(\, x^i\,\big)\; f^j_{t|t}\, \mu^j
\end{aligned} \tag{7}$$

5. POINT ESTIMATES

The distinction between *intrinsic* and *extrinsic* was introduced in Srivastava and Klassen [2002], where a mean value of a distribution on a manifold was estimated by first estimating the mean of the distribution of the manifold embedded in Euclidean space, and then projecting the mean back to the manifold. This, they termed the *extrinsic estimator*. In contrast, an *intrinsic estimator* was defined without reference to an embedding in Euclidean space. While this may seem a hard contrast at first, Brun et al. [2007] shows that both kinds of estimates may be meaningful from a maximum likelihood point of view, for some manifolds with "natural embedding".

5.1 Intrinsic point estimates

A common intrinsic generalization of the usual mean in Euclidean space is defined as a point where the variance obtains a global minimum, where the variance "only" requires a distance to be defined:

$$\mathsf{Var}_x(\, x\,) \triangleq \int \mathrm{d}(\, x',\, x\,)^2\; \mathsf{f}_x(\, x'\,)\; \mathrm{d}x' \tag{8}$$

Unfortunately, such a mean may not be unique, but if the support of the distribution is compact, there will be at least one.

Other intrinsic point estimates may also be defined, but since the motivation for discussing point estimates here is to illustrate that algorithms aimed at computing intrinsic point estimates based on the proposed probability density representation can be defined, other estimates are not discussed further here.

Since distributions with a globally unique minimum may be arbitrarily close to distributions with several distinct global minimums, it is our understanding that schemes based on local search, devised to find one *good* local minimizer, are reasonable approximations of the definition. Hence, there are two tasks to consider; implementation of the local search, and a scheme that uses the local search in order to find a *good* local minimizer.

To implement a local search, one must be able to compute search directions and to perform line searches. For this, we rely on the exponential map, which allows these tasks to be carried out in the tangent space of the current search iterate. The search direction used is steepest descent computed using finite difference approximation, although more sophisticated methods exist in the literature [Pennec, 2006]. The details of the scheme that makes use of the local search are omitted due to space constraints.

5.2 Extrinsic point estimates

The extrinsic mean estimator proposed in Srivastava and Klassen [2002] is *defined* by replacing the *distance* $\mathrm{d}(\, x',\, x\,)$ in (8) by the distance obtained by embedding the manifold in Euclidean space and measuring in this space instead. It is argued

that if the support of the distribution is small, this should give results similar to the intrinsic estimate. However, considering how arbitrary the choice of embedding is, it is clear that the procedure as a whole is rather arbitrary as well. (Nevertheless, a good embedding seems likely to produce useful results, see for instance the examples in Srivastava and Klassen [2002].)

6. ALGORITHM AND IMPLEMENTATION

The final component to discuss before putting the theory of the previous sections together in an algorithm, is how tessellations are computed. In this section, we do this, present the algorithm in a compact form, and include some notes on the software design and implementation.

6.1 Base tessellations (of spheres)

To be more specific about how a base tessellation may be computed, we have considered how this can be done for spheres, but the technique we employ does not only work for spheres.

The first step is to generate the set of points x^i. We omit details due to space constraints.

The remaining steps are general and do not only apply to spheres. First, equations for the half-space containing the manifold and being bordered by the tangent space at each point x^i is computed. This comes down to finding a base for the space orthogonal to the tangent space at x^i — for spheres, this is trivial. The intersection of these half-spaces is a polytope with a one-to-one correspondence between facets and generating points. (We rely on existing software here, please refer to section 6.4 at this point.) Projecting the facets towards the origin will generate a tessellation, and for spheres this will be a Voronoi tessellation if the "natural" embedding is used. Each region is given by the set of projected vertices of the corresponding polytope facet.

As part of the tessellation task, the volume of each region must also be computed. For the 2-sphere this can be done exactly thanks to the simple formula giving the area of the region bounded inside the geodesics between three points on the sphere [Beger, 1978, p 198]. In the general case we approximate the volumes on the manifold by the volume of the polytope facets. (Note that a facet can be reconstructed from the projected vertices by projecting back to the (embedded) tangent space at the generating point.) For spheres the ideal total volume is known, and any mismatch between the sum of the volumes of the regions and the ideal total volume is compensated by scaling all volumes by a normalizing constant.

6.2 Summary of the algorithm

In the following algorithm, the numbers $f^i_{t|t-1}$ are the (approximate) values of the probability density function at the point x^i, at time t given the measurements from time 0 to time $t-1$. The numbers $f^i_{t|t}$ are the (approximate) values of the probability density function at time t, given also the measurements available at time t.

Initialization

Compute a tessellation with regions R^i of the manifold. Assign a representative point x^i to each region, and measure the volumes μ^i. In case of spheres, see section 6.1.

Let $f^i_{0|-1}$ be the a priori distribution, that is, assign a non-negative value to each $f^i_{0|-1}$ so that $\sum_i f^i_{0|-1}\,\mu^i = 1$.

Process measurements

for $t = 0,\ 1,\ 2,\ \ldots$

Compute a point prediction from $f_{t|t-1}$, for instance, by minimizing (8).

Use the measurements y_t to compute $f_{t|t}$ using $\mathrm{BayesRule}$, see (6) for details.

Compute a point estimate from $f_{t|t}$, for instance, by minimizing (8).

Make a time update to compute $f_{t+1|t}$ using (7).

Possibly update the tessellation. (Details are subject for future work.)

end

6.3 Software design

Our implementation is written in C++ for fast execution. Still, there is a strong emphasis on careful representation of the concepts of geometry in the source code. Perhaps most notably, a manifold is implemented as a C++ type, and allows elements to be handled in a coordinate-free manner. By providing a framework for writing coordinate-free algorithms, we try to guide algorithm development in a direction that makes sense from a geometric point of view.

Other concepts of geometric relevance that are represented in the software design are:

- Scalar functions, that is, mappings from a manifold to the set of real numbers.
- Coordinate maps, that is, invertible mappings from a part of the manifold to tuples of real numbers.
- Tangent spaces, that is, the linear spaces of directional derivatives at a certain point of the manifold. As with the manifold elements, elements of the tangent spaces are handled in a coordinate-free manner. The basic means for construction of tangents is to form the partial derivative with respect to a coordinate function.
- Euclidean spaces are implemented as special cases of manifolds.

6.4 Supporting software

A very important part of the tessellation procedure for spheres and other manifolds with a convex interior seen in the embedding space, are the conversions between polytope representations. That is, given a set of bounding hyperplanes, we want a vertex representation of all the faces, and given a set of vertices, we want the corresponding set of hyperplanes. I our work, these tasks were carried out using *cddlib* [Fukuda, 2008], distributed under the *GNU general public licence*.

Although several algorithms for computing the volume of polytopes of arbitrary dimension exist [Büeler et al., 2000], no freely available implementation compatible with C++ was found. We would like to encourage the development, the sharing, and the advertisement of such software. The authors' implementation for this task is a very simple triangulation-based recursion scheme.

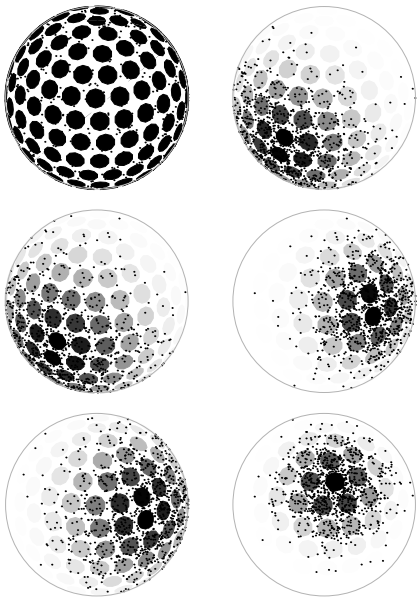

Fig. 1. Estimated probability density function. Left: predictions before a measurement becomes available. Right: estimates after measurement update. Rows correspond to successive time steps. Patches are colored proportional to the density in each region, and random samples are marked with dots. It is seen how the uncertainty increases when time is incremented, and decreases when a measurement becomes available, and that the uncertainty decreases over time as the information from several measurements is fused.

7. EXAMPLE

To illustrate the proposed filtering technique, a manifold of dimension 2 was chosen so that the probability distributions are amenable to illustration. We consider the bearing-tracking problem in 3 dimensions, that is, the state evolves on the 2-sphere. This may be a robust alternative to tracking the position of an object when range-information cannot be determined reliably. It is also a good example to mention when discussing models without dynamics (velocities are not part of the state), since the lack of (Lie) group structure makes the extension to dynamic models non-trivial. As an example of a bearing-sensor in 3 dimensions, we may consider a camera and an object recognition algorithm, which returns image coordinates in each image frame, which are then converted to the three components of a unit vector in the corresponding direction. The example is about the higher-level considerations of the filtering problem, and not the low-level details of implementing the manifold at hand.

The deterministic part of the dynamic equation, g, does not depend on any external input, and just maps any state to itself. The noise in the equation is given by a von Mises-Fischer distribution field (see the overview Schaeben [1992]) with concentration parameter $\kappa = 12$ everywhere. The three scalar measurements are (inaccurately) assumed to have independent Gaussian additive measurement noise, with $\sigma = 0.4$.

Given an initial state, a simulation of the model equations is run, resulting in a sequence of measurements. The manifold is tessellated into $N = 200$ approximately equally sized regions, and the filter is initialized with a uniform probability density. The probability density estimate is then updated as measurements are made available to the filter. The result is illustrated in figure 1.

8. CONCLUSION AND FUTURE WORK

We have shown that point-mass filters can be used to construct robust filters on compact manifolds. By separating the implementation of the low-level manifold structure from the higher-level filter algorithm, we are able to formulate and implement much of the algorithm without reference to a particular embedding. The technique has been demonstrated by considering a simple application on the 2-sphere.

Future work includes application to $\mathcal{SO}(3)$, that is, the manifold of orientations, adaptation of the tessellation, and utilizing Lie group structure when available. In order to cope with the substantial increase of dimension that would result from augmenting the state of our models to also include physical quantities such as angular momentum, the filter should be tailored to tangent or cotangent bundles.

REFERENCES

Jeffrey M. Augenbaum and Charles S. Peskin. On the construction of the voronoi diagram on a sphere. *J. Comp. Phys.*, 59(2):177–192, June 1985.

William H. Beger, editor. *CRC Handbook of mathematical sciences*. CRC Press, Inc., 5th edition, 1978.

Niclas Bergman. *Recursive Bayesian estimation — Navigation and tracking applications*. PhD thesis, Linköping University, May 1999.

Anders Brun, Carl-Fredrik Westin, Magnus Herberthson, and Hans Knutsson. Intrinsic and extrinsic means on the circle — a maximum likelihood interpretation. In *IEEE Conf. Acou., Sp., Sig. Proc.*, Honolulu, Hawaii, USA, April 2007.

R. S. Bucy and K. D. Senne. Digital synthesis of nonlinear filters. *Automatica*, 7:287–298, 1971.

Benno Büeler, Andreas Enge, and Komei Fukuda. *Polytopes: Combinatorics and computation*, chapter Exact volume computation for polytopes: A practical study. Number 29 in DMV Seminar. Birkhäuser, 2000.

Alessandro Chiuso and Stefano Soatto. Monte Carlo filtering on Lie groups. *Proc. 39th IEEE Conf. Decis. Contr.*, pages 304–309, December 2000.

Daniel Choukroun, Itzhack Y. Bar-Itzhack, and Yaakov Oshman. Novel quaternion Kalman filter. *IEEE Trans. Areo. Elec.*, 42(1):174–190, January 2006.

John L. Crassidis, F. Landis Markley, and Yang Cheng. Survey of nonlinear attitude estimation methods. *J. Guid. Contr. Dyn.*, 30(1):12–28, January 2007.

Fred Daum. Nonlinear filters: beyond the kalman filter. *IEEE Aero. Elec. Sys. Mag.*, 20(8:2):57–69, 2005.

Theodore Frankel, editor. *The geometry of physics — an introduction*. Cambridge University Press, 2nd edition, 2004.

Komei Fukuda. *cddlib reference manual, version 0.94*. Institute for Operations Research and Institute of Theoretical Computer Science, ETH Zentrum, CH-8092 Zurich, Switzerland, 2008. URL `http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html`.

Junghyun Kwon, Minseok Choi, F. C. Park, and Changmook Chun. Particle filtering on the Euclidean group: framework and applications. *Robotica*, 25: 725–737, 2007.

Jehee Lee and Sung Yong Shin. General construction of time-domain filters for orientation data. *IEEE Trans. Vis. Comp. Graph.*, 8(2):119–128, April 2002.

Taeyoung Lee, Melvin Leok, and Harris McClamroch. Global symplectic uncertainty propagation on SO(3). *Proc. 47th IEEE Conf. Decis. Contr.*, December 2008.

James Ting-Ho Lo and Linda R. Eshleman. Exponential fourier densities on so(3) and optimal estimation and detection for rotational processes. *SIAM J. Appl. Math.*, 36(1):73–82, February 1979.

Hyeon-Suk Na, Chung-Nim Lee, and Otfried Cheong. Voronoi diagrams on the sphere. *Comp. Geom.*, 23:183–194, 2002.

Xavier Pennec. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *J. Math. Imag. Vis.*, 25(1):127–154, July 2006.

Helmut Schaeben. "Normal" orientation distributions. *Textu. Stre. M.-struct.*, 19(4):197–202, 1992. J. changed name in 2008 from *Textu. M.-struct.*

Anuj Srivastava and Eric Klassen. Monte Carlo extrinsic estimators of manifold-valued parameters. *IEEE Trans. Sig. Proc.*, 50(2):299–308, February 2002.

N. Sukumar. Voronoi cell finite difference method for the diffusion operator on arbitrary unstructured grids. *Int. J. Num. Meth. Eng.*, 57(1):1–34, May 2003.

David Törnqvist, Thomas B. Schön, Rickard Karlsson, and Fredrik Gustafsson. Particle filter SLAM with high dimensional vehicle model. *J. Int. Robo. Sys.*, 2008. Accepted for publication.