# Multi-Scale activity estimation with spatial abstractions

# Multi-Scale Activity Estimation with Spatial Abstractions

Majd Hawasly[1⋆], Florian T. Pokorny[2], and Subramanian Ramamoorthy[3]

[1] FiveAI Inc., Edinburgh, United Kingdom
m.hawasly@five.ai
[2] KTH Royal Institute of Technology, Stockholm, Sweden
[3] School of Informatics, The University of Edinburgh, Edinburgh, United Kingdom

**Abstract.** Estimation and forecasting of dynamic state are fundamental to the design of autonomous systems such as intelligent robots. State-of-the-art algorithms, such as the particle filter, face computational limitations when needing to maintain beliefs over a hypothesis space that is made large by the dynamic nature of the environment. We propose an algorithm that utilises a hierarchy of such filters, exploiting a filtration arising from the geometry of the underlying hypothesis space. In addition to computational savings, such a method can accommodate the availability of evidence at varying degrees of coarseness. We show, using synthetic trajectory datasets, that our method achieves a better normalised error in prediction and better time to convergence to a true class when compared against baselines that do not similarly exploit geometric structure.

## 1 Introduction

Autonomous agents acting in dynamic environments need the capacity to make predictions about the environment within which they are acting, so as to take actions that are suited to the present world state. Traditionally, tools for *state estimation* are geared to the case wherein uncertainty arises from noise in the dynamics or sensorimotor processes. For example, the particle filter is a state estimation method utilising a nonparametric representation of beliefs over the state space, used extensively in robotics. However, in problems involving spatial activity, e.g., robot navigation, the underlying dynamics are best described in a hierarchical fashion, as movement is not just determined by local physical laws and noise characteristics, but also by longer-term *goals* and *preferences*. This has a few implications for predictive models: we require techniques that 1) accept evidence at varying scales - from very precise position measurements to coarser forms of knowledge, e.g. human feedback, and 2) make predictions at multiple scales to support decision making. These form the primary focus of this paper.

Early models of large-scale spatial navigation [5] considered ways in which multiple representations, ranging from coarse and intuitive topological notions of connectivity between landmarks to a more detailed metrical and control level description of action selection, could be brought together in a coherent framework

⋆ At the time of this study, the first author was with The University of Edinburgh.

and implemented on robots. Other recent methods, e.g. [1, 3], propose ways in which control vector fields could be abstracted so as to support reasoning about larger-scale tasks. While these works provide useful inspiration, the hierarchy in these methods is often statically defined by the designer, while in many applications it is of interest to learn it directly from data, e.g. to enable continual adaptation over time. Also, these approaches are often silent on how best to integrate tightly with Bayesian belief estimates, such as within a particle filter.

There is indeed prior work on the notion of hierarchy in state estimation with particle filters. For instance, Verma et al. [11] define a variable resolution particle filter for operation in large state spaces, where chosen states are aggregated to reduce the complexity of the filter. Brandao et al. [2] devise a subspace hierarchical particle filter wherein state estimation can be run in parallel with factored parallel computation. Other ways to factoring computation exist, e.g. [6, 10], and a hierarchy of feature encodings can be used [13]. However, to the best of our knowledge, no prior method allows tracking a process on multiple scales at once and accepts evidence with variable resolutions.

In this paper, we learn a spatial hierarchy directly from input trajectories, using which we devise a novel construction of a bank of particle filters - one at each scale in a geometric filtration - which maintain consistent beliefs over the trajectories as a whole and, through that, over the state space. We present an agglomerative clustering scheme [7] using the Fréchet distance between trajectories [4] to compute a tree-structured representation of trajectory classes that correspond to incrementally-coarser partitions of the underlying space. This is inspired by persistent homology on trajectories [8, 9] whose output is also such a hierarchical representation. We then define a *linear dynamics* model at each of the levels of the hierarchy based on the subset of trajectories they represent, and show how that can be used with a stream of observations to provide updates to the probability that the system is following the dynamics associated with each of the abstracted trajectory classes. This construction of the filter allows us to fluently incorporate readings of varying resolution if they were accompanied by an indication of the coarseness with which the observation is to be interpreted.

We show that our proposed method performs better than baselines both in terms of normalised error in prediction with respect to the ground truth, and in terms of the time taken for the belief to converge to the true trajectory of a class (where convergence is defined with respect to the resolution of the prediction being considered). We perform experiments with synthetic datasets which brings out the qualitative behaviour of the procedure in a visually intuitive manner.

## 2 Multiscale Hierarchy of Particle Filters

The *Multiscale Hierarchy of Particle Filters* (MHPF) is a bank of *consistent* particle filters defined over *abstractions* of the state space induced by example trajectories. The lowest level of this hierarchy consists of the complete set of trajectories with cardinality equal to the size of the trajectory dataset, while each other abstract level has coarser descriptions of the trajectory shape defined

by equivalence class of similar trajectories for increasing thresholds. With a particle filter defined at each level, this *inclusion* property of the representation allows evidence at various degrees of coarseness to be incorporated into the full bank of filters while maintaining consistency across all levels, see Fig. 2.

**Construction** To create a filtration of spatial abstractions from trajectories we consider agglomerative hierarchical clustering [12] by means of a trajectory distance measure. In this paper, we use the discrete Fréchet distance [4]: for two discretised $d$-dimensional trajectories $\tau_1 : [0, m] \rightarrow \mathbb{R}^d$ and $\tau_2 : [0, n] \rightarrow \mathbb{R}^d$, the distance $\delta_F(\tau_1, \tau_2) = \inf_{\alpha,\beta} \max_{j \leq m+n} \delta_E(\tau_1(\alpha(j)), \tau_2(\beta(j)))$, where $\alpha$ and $\beta$ are discrete, monotonic re-parametrisations $\alpha : [1 : m + n] \rightarrow [0 : m]$, $\beta : [1 : m + n] \rightarrow [0 : n]$ which align the trajectories to each other point-wise, and $\delta_E(.,.)$ is Euclidean distance. Thus, $\delta_F$ corresponds to the maximal point-wise distance between optimal reparameterisations of $\tau_1$ and $\tau_2$, which can be computed efficiently using dynamic programming in $O(mn)$ time [4].

Let $D$ be the distance matrix of the input trajectories, $D_{i,j} = D_{j,i} = \delta_F(\tau_i, \tau_j)$. A *single-linkage* hierarchical agglomerative clustering of $D$ results in a *tree* $\mathcal{T}$ of trajectory clusters in which the leaves are the single trajectories, while every other tree layer is created when the pair with the smallest distance from the previous layer combine together (Fig. 2). If $\tau_i$ and $\tau_j$ are such a pair, we call the new cluster $\tau_{ij}$ a *parent* to its constituents and write $\tau_{ij} = \rho(\tau_i) = \rho(\tau_j)$. Let the *birth index* $b$ be that minimum distance that indexes the creation of a layer (e.g., $b_{ij} = D_{i,j}$ for $\tau_{ij}$), and the *death index* $d$ be the distance at which a cluster is subsumed to its parent (e.g., $d_i = d_j = D_{i,j}$ for $\tau_i$ and $\tau_j$). Let $\mathcal{C}$ be the set of all clusters in $\mathcal{T}$. A class $c_i \in \mathcal{C}$ is *alive* at some index $x$ if $b_i \leq x < d_i$. A *level* in the tree $\mathcal{C}_x \subseteq \mathcal{C}$ at index $x$ contains all the classes that are alive at $x$. Fig. 1 (Left) illustrates an example clustering.

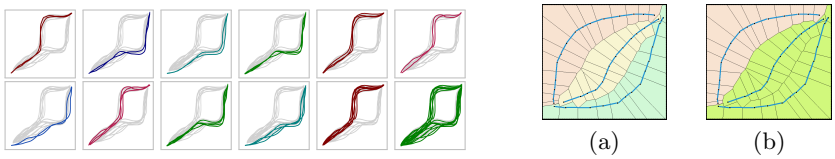



**Fig. 1:** (Left) Trajectory clusters with increasing birth indices of a tree of 14 trajectories using hierarchical single-linkage agglomerative clustering with Fréchet distance. (Right) The intuition behind the probability operations in a toy example 2D domain, where (a) three classes merge into two (b).

Thus, a cluster $c \in \mathcal{C}$ is a collection of qualitatively *similar* trajectories at some *level of resolution* (index) $b$. The *class* of behaviour that $c$ represents could be modelled as a generative model $\mathbf{P}(z'|z, c)$, $z, z' \in \mathbb{R}^d$. With the assumption of no self-intersecting trajectories, we can approximate the dynamics of $c$ at some arbitrary point $z \in \mathbb{R}^d$ using a weighted average of velocity at local points of $c$ in an $\epsilon$-ball around $z$: $B_\epsilon(z) = \{z' \in c : \delta_E(z', z) < \epsilon\}$, where $\epsilon$ relates to the density or sparsity of the trajectories. Hence, $\dot{z} = \frac{1}{\eta} \sum_{z' \in B_\epsilon(z)} \frac{\dot{z}'}{\delta_E(z,z')}$, with normalisation $\eta = \sum_{z' \in B_\epsilon(z)} \frac{1}{\delta_E(z,z')}$. Thus, $z' \sim z + \dot{z} + \gamma(\kappa)$, where $\gamma(\kappa)$ is a noise term related to dynamics noise $\kappa$.

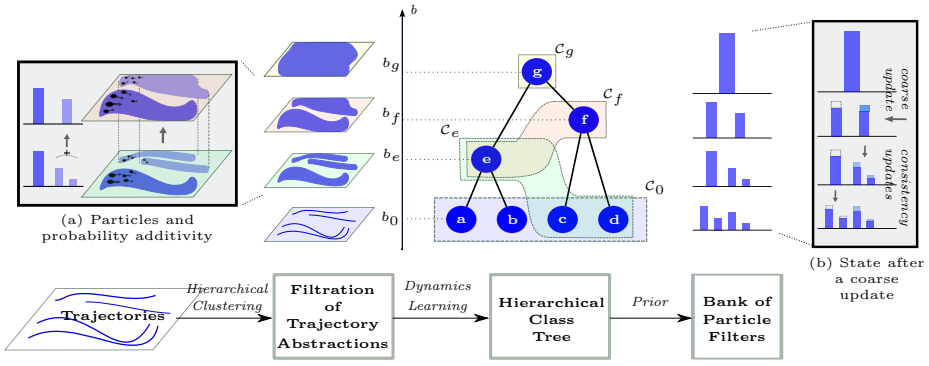



**Fig. 2:** An overview of the approach. Trajectories are hierarchically clustered into a filtration of spatial abstractions (*classes*), organised in a tree structure by birth indices. Shaded areas on the tree show *levels* of the hierarchy, with $\mathcal{C}_0$ being the finest level with single trajectory classes. Inset (a) shows an example particle set and how the tree structure enforces the consistency of class probabilities. The distributions on the right show an example of a consistent estimate across the tree maintained by a bank of particle filters. Inset (b) shows an example of a coarse observation received at one level, and how updates propagate throughout the tree to maintain consistency.

At each level of the tree $\mathcal{C}_b$ we define a particle filter where a *particle* $x^t$ represents a weighted hypothesis of *both* the class of behaviour $c \in \mathcal{C}_b$ and the position $z^t \in \mathbb{R}^d$ at time $t$. We write $(x^t(z^t, c), w^t)$ where $w^t$ is a weight that reflects to what extent the hypothesis of the particle is compatible with evidence. We denote by $X_b$ the set of all particles of the filter at $\mathcal{C}_b$. There are two kinds of observations in MHPF: 1) *position observations* $z^t + \gamma(\psi)$, where $\gamma(\psi)$ is a noise term related to the observation noise parameter $\psi$; and 2) *coarse observations* which provide qualitative evidence regarding the underlying process. Here, we assume that coarse observations can be identified to one of the classes in $\mathcal{C}$.

Algorithm 1 presents the full MHPF procedure. First, the particle set $X_0$ of the filter at $\mathcal{C}_0$ is created by sampling $N$ particles from a prior over initial positions and class assignment from $\mathcal{C}_0$ (individual trajectories) with uniform weights. Denote by $N_i$ the number of particles of class $c_i$, such that $\sum_{c_i \in \mathcal{C}_0} N_i = N$. The prior probabilities of the classes $c_i \in \mathcal{C}_0$ can be computed as $N_i/N$. These probabilities propagate recursively upwards in the tree by additivity: a parent's probability is the sum of its children's probabilities, $\mathbf{P}^t(\bar{c}) = \sum_{\underline{c}=\rho^{-1}(\bar{c})} \mathbf{P}^t(\underline{c})$. By the same principle, the children of a class proportionally inherit their parent's probability when moving down the tree. For the sake of intuition, consider the simple example in Fig. 1 (Right), where a cluster of trajectories can be understood spatially as the union of Voronoi cells of trajectory discretisation. The corresponding probability of this class is the probability of the agent being in that region. Then, when classes merge at some level of resolution their corresponding regions merge, and thus their probabilities are added up. This simple technique guarantees consistency of the filters by design.

With the class probabilities specified, the same number of particles as assigned to the children are sampled for parents, $N_{\bar{c}} = \sum_{\underline{c}=\rho^{-1}(\bar{c})} N_{\underline{c}}$, and this is

**Algorithm 1** Multiscale Hierarchy of Particle Filters

**Require:** Prior over particles, number of basic particles $N$, the depletion parameter $v$, tree structure $\mathcal{T}$

Create $X_0$: sample $N$ particles from a prior over $\mathcal{C}_0 \times \mathbb{R}^d$ with equal weights.
**for** each time step $t > 0$ **do**
  Build the tree probabilities up from $X_0$ and $\mathcal{C}_0$ (Algorithm 2).
  **for** parents $\bar{c}$ of $\mathcal{C}_0$ classes recursively **to** the root of $\mathcal{T}$ **do**
    Sample $N_{\bar{c}}$ particles; $N_{\bar{c}} = \sum_{\underline{c}=\rho^{-1}(\bar{c})} N_{\underline{c}}$, with equal weights
  **end for**
  Sample a new position per particle, $z^t \sim \mathbf{P}(z|z^{t-1}, c)$
  Receive observation $\xi^t$.
  **if** fine observation **then**
    $\mathcal{C}_\xi = \mathcal{C}_0$.
    update $X_0$ weights with Euclidean distance to $\xi^t$: $w^t \propto -\log(\delta_E(., \xi^t))$.
  **else if** coarse observation at tree level $b_\xi$ **then**
    Find all alive classes at $b_\xi$: $\mathcal{C}_\xi = \{c_i \in \mathcal{C} :\ b_i \leq b_\xi < d_i\}$.
    Compute tree distance $\delta_\mathcal{T}(c, \xi^t)$, for all $c \in \mathcal{C}_\xi$.
    Update weights in $X_c$, $c \in \mathcal{C}_\xi$ relative to distance: $w^t \propto -\log(\delta_\mathcal{T}(c, \xi^t))$.
  **end if**
  Rebuild the tree probabilities from $\mathcal{C}_\xi$ and $X_\xi$ (Algorithm 2).
  Update particle weights in $X \setminus X_\xi$ : $w^t = w^{t-1} \frac{\mathbf{P}^t(c)}{\mathbf{P}^{t-1}(c)}$
  Update $X_0$: resample $N\,(1-v)$ particles from $X_0$ based on new weights $w^t$, and $N\,v$ particles uniformly randomly from $\mathcal{C}_0$.
**end for**

**Algorithm 2** Tree Probability Rebuild

**Require:** Tree structure $\mathcal{T}$, tree level $\mathcal{C}_b$, particle set $X_b$

Update the probabilities of $c_i \in \mathcal{C}_b$ from $X_b$ weights: $\mathbf{P}^t(c_i) = \frac{\sum_{c(x)=c_i} w^t(x)}{\sum_{x \in X_b} w^t(x)}$
**for** children of $\mathcal{C}_b$ classes recursively **to** the leaves of $\mathcal{T}$ **do**
  Update child $\underline{c}$ probability relative to its parent $\bar{c}$: $\mathbf{P}^t(\underline{c}) = \mathbf{P}^{t-1}(\underline{c}) \frac{\mathbf{P}^t(\bar{c})}{\mathbf{P}^{t-1}(\bar{c})}$
**end for**
**for** parents of $\mathcal{C}_0$ classes recursively **to** the root of $\mathcal{T}$ **do**
  Update parent $\bar{c}$ probability relative to its children $\underline{c}$: $\mathbf{P}^t(\bar{c}) = \sum_{\underline{c}=\rho^{-1}(\bar{c})} \mathbf{P}^t(\underline{c})$
**end for**

repeated recursively to the top of the tree. Note that, any arbitrary level $\mathcal{C}_b$ of the tree would have exactly $N$ particles with a proper probability distribution. The last stage of the tree construction is to sample new positions for the particles. Note that the class assignment of a particle does not change due to sampling.

**Updates** A coarse observation $\xi \in \mathcal{C}$ with resolution $b_\xi$ targets all the particles from classes that are *alive* at $\mathcal{C}_\xi = \{c_i \in \mathcal{C} |\ b_i \leq b_\xi < d_i\}$. To update these particles, we use the *tree distance* between classes $\delta_\mathcal{T}(.,.)$ which we define as the birth index of the *youngest* shared parent of the two classes in the tree. This measures how large the $\epsilon$-balls around the points of one class need to be to include the other. For example, in Fig. 2, $\delta_\mathcal{T}(e, g) = \delta_\mathcal{T}(c, e) = b_g$. The weight of

a particle is updated relative to the distance of the observation from the particle's class $c$, $w \propto -\log(\delta_{\mathcal{T}}(\xi, c))$. A position observation $\xi \in \mathbb{R}^d$, on the other hand, updates a particle relative to the Euclidean distance between the observation and the particle's position $z$, $w \propto -\log(\delta_E(\xi, z))$.

After updating all particles in $X_\xi$, the probabilities of the corresponding classes in $\mathcal{C}_\xi$ are recomputed as the sum of their particles' normalised weights, then propagate to the rest of the tree as in Algorithm 2. Here, children classes of $\mathcal{C}_\xi$ are updated first recursively relative to their parents' new probabilities, $\mathbf{P}^t(\underline{c}) = \mathbf{P}^{t-1}(\underline{c})\frac{\mathbf{P}^t(\bar{c})}{\mathbf{P}^{t-1}(\bar{c})}$, $\forall \underline{c} = \rho^{-1}(\bar{c})$, then the updates propagate upwards to update all the remaining parents $\mathbf{P}^t(\bar{c}) = \sum_{\underline{c}=\rho^{-1}(\bar{c})} \mathbf{P}^t(\underline{c})$. Then, particle weights are updated to reflect the updated class probabilities, $w^t = w^{t-1}\frac{\mathbf{P}^t(c)}{\mathbf{P}^{t-1}(c)}$, $\forall x \in X \setminus X_\xi$. The final step is to sample $N$ particles from $X_0$ with uniform weights to get the posterior particle set after incorporating the evidence $\xi$. To guard against particle depletion, we replace the classes of a small percentage $v$ of all particles uniformly randomly to classes from $\mathcal{C}_0$.

## 3 Experiments

We evaluate the performance of **MHPF** with $N = 100$ particles in two synthetic 2-dimensional navigation domains, one representing a 2-dimensional configuration space with 33 trajectories, and the other with 13 trajectories (Fig. 3 (Left)). We compare the performance to particle filters without access to the hierarchical structure: **BL1** is a basic particle filter with $N = 100$ particles, each follows the dynamics of a single trajectory (classes $c \in \mathcal{C}_0$); and **BL2** is a particle filter with $N = 100$ particles which all follow the averaging dynamics of the trajectories together with $\kappa$ noise (Note that **BL1** is equivalent to the filter at the bottom layer of MHPF stack, and **BL2** is equivalent to the filter at the top layer.) We use as metrics: 1) the mean squared error of the filter's point prediction, 2) the tree distance of the filter's predicted class to the ground truth, and 3) the time to convergence to the true class. Each experiment is run with 10 randomly-selected ground truth trajectories, reporting averaged scores of 25 repetitions. Trajectories are uniformly discretised, and the length of a trial depends on the number of trajectory points. Observation at time $t$ is generated from the discretised ground truth $z^t \in \mathbb{R}^2$ and the observation noise $\psi$. A fine observation is defined as $z^t + \gamma$ where $\gamma \in [0, \psi] \times [0, \psi]$, while a coarse observation is selected by sampling $n = 10$ points from $\mathcal{N}(z^t, \psi^2)$ then finding the class that is most likely to generate these samples. We use the localised dynamics model as in Section 2 with $\epsilon = b_c$ for some coarse class $c$ and the noise parameter $\kappa$. At the end of every step, $v = 1\%$ of the particles is changed randomly.

In the configuration space dataset, we compute the filter's predicted position at time $t$ as the $w$-weighted average of the particle positions when using fine observations only, and report the average mean squared error (MSE) of the ground truth over time. **MHPF** achieved a mean of 0.27 (standard deviation of 0.04), beating **BL1** 0.38(0.14) and **BL2** 0.53(0.13). Fig. 3 (Right) illustrates

the kind of multi-resolution output MHPF can produce, showing the *maximum a posteriori* (MAP) class in time at different levels of the tree.

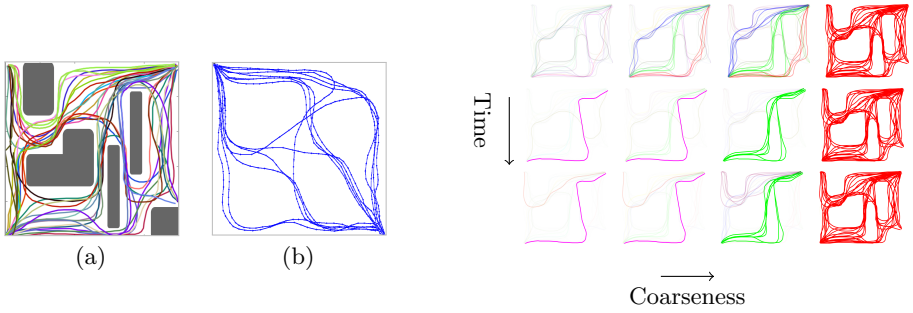
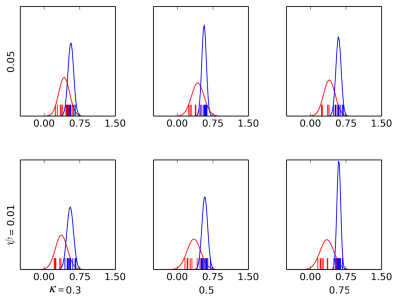
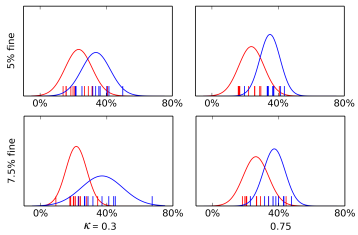



**Fig. 3:** (Left) Datasets used. (Right) Evolution of MHPF prediction. The columns show levels across the tree with the finest at the left, and rows show time steps with the first at the top. Each panel shows the trajectories of the alive classes. The opacity of the line reflects the probability of the class.

Next, we compare **MHPF** with **BL1** using the 13 trajectory dataset in a situation where fine observations are consistently generated, but coarse observations are produced stochastically 50% of the time. We analyse the benefit of this additional knowledge by plotting, in Fig. 4a, the average tree distance of the MAP prediction to the ground truth, with noise parameters ($\kappa = 30\%, 50\%, 75\%$) and ($\psi = 1\%, 5\%$). Finally, when fine observations are only provided for a lead-in period of 5%/ 7.5% of trial length followed by only coarse observations, we show in Fig. 4b the time needed for the tree distance to converge within the 33%-ball of the ground truth with noise parameters $\psi = 1\%$ and $\kappa = 30\%, 75\%$.




**(a)** Tree distance of the ground truth to MAP prediction. Coarse instructions were provided stochastically 50% of the time. The plot shows robustness against noise as dynamics noise varies between 30% (Left) to 75% (Right) of range, and observation noise ranges between 1% (Bottom) to 5% (Top). MHPF converges to a better solution than the baseline (statistically significant at p-value= 0.004).




**(b)** Time needed to reach within 33% of convergence. Fine observations are provided for a lead-in period (5% (Top) and 7.5% (bottom) of trial time). The plot shows the benefit of coarse observations to convergence time. Dynamics noise ranges from 30% (Left) to 75% (Right), and observation noise is set to 1%. MHPF converges faster to the correct solution than the baseline (statistically significant at a p-value = 0.02).

**Fig. 4:** Performance results comparing MHPF (red) and BL1 (blue) - lower is better.

## 4 Conclusion

We propose an estimation and forecasting approach utilising a filtration over trajectories and a correspondingly hierarchical representation of probability distributions over the underlying state space so as to enable Bayesian filtering. A key benefit of our methodology is the ability to incorporate 'coarse' observations alongside the basic 'fine' scale signals. This approach to seamlessly handling inhomogeneity in scale is a benefit in many robotics and sensor networks applications. We demonstrate the usefulness of this technique with experiments that show performance gains over a conventional particle filtering scheme that does not similarly exploit the geometric structure in the hypothesis space.

## References


1. Belta, C., Bicchi, A., Egerstedt, M., Frazzoli, E., Klavins, E., Pappas, G.: Symbolic planning and control of robot motion [grand challenges of robotics]. Robotics Automation Magazine, IEEE 14(1), 61–70 (2007)
2. Brandao, B.C., Wainer, J., Goldenstein, S.K.: Subspace hierarchical particle filter. In: 19th Brazilian Symposium on Computer Graphics and Image Processing, SIBGRAPI'06. pp. 194–204. IEEE (2006)
3. Burridge, R.R., Rizzi, A.A., Koditschek, D.E.: Sequential composition of dynamically dexterous robot behaviors. The International Journal of Robotics Research 18(6), 534–555 (1999)
4. Eiter, T., Mannila, H.: Computing discrete Fréchet distance. Tech. rep., Tech. Report CD-TR 94/64, Technical University of Vienna (1994)
5. Kuipers, B.: The spatial semantic hierarchy. Artificial intelligence 119(1) (2000)
6. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: Vernon, D. (ed.) Computer Vision ECCV 2000, Lecture Notes in Computer Science, vol. 1843, pp. 3–19. Springer Berlin Heidelberg (2000)
7. Müllner, D.: Modern hierarchical, agglomerative clustering algorithms. arXiv preprint arXiv:1109.2378 (2011)
8. Pokorny, F.T., Hawasly, M., Ramamoorthy, S.: Multiscale topological trajectory classification with persistent homology. In: Robotics: Science and Systems (2014)
9. Pokorny, F.T., Hawasly, M., Ramamoorthy, S.: Topological trajectory classification with filtrations of simplicial complexes and persistent homology. The International Journal of Robotics Research (2015)
10. Shabat, G., Shmueli, Y., Bermanis, A., Averbuch, A.: Accelerating particle filter using randomized multiscale and fast multipole type methods. Pattern Analysis and Machine Intelligence, IEEE Transactions on PP(99), 1–1 (2015)
11. Verma, V., Thrun, S., Simmons, R.: Variable resolution particle filter. In: International Joint Conference on Artificial Intelligence (IJCAI). pp. 976–984 (2003)
12. Xu, R., Wunsch, D., I.: Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)
13. Yang, C., Duraiswami, R., Davis, L.: Fast multiple object tracking via a hierarchical particle filter. In: Tenth IEEE International Conference on Computer Vision (ICCV). vol. 1, pp. 212–219 Vol. 1 (2005)