Particle Filter Estimation of Posterior Densities

Anonymous CVPR submission

Paper ID ****

Abstract

1. Background

Our goal is to compute a posterior distribution $p(X_1, \ldots, X_m \mid Z)$, where $(X_1, \ldots, X_m)$ is a vector of random variables (RVs) and $Z = \{z_1, \ldots, z_m\}$ is a set of observations. This will allow us to find value assignments $X_t = \widehat{x}_t$ for $t = 1, \ldots, m$ to RVs that maximize this posterior:

$$\widehat{x}_{1:m} = \underset{x_{1:m}}{\operatorname{argmax}}\, p(X_1 = x_1, \ldots, X_m = x_m \mid Z), \quad (1)$$

where $x_{1:m} = (x_1, \ldots, x_m) \in \mathcal{X}^m$ is a state space vector. As it is commonly the case, we will abbreviate

$$p(X_1 = x_1, \ldots, X_m = x_m \mid Z) = p(x_{1:m} \mid Z).$$

We will achieve our goal by approximating the posterior distribution with a final number of samples in the framework of Bayesian Importance Sampling (BIS). Since it is usually difficult to draw samples from the pdf $p(x_{1:m}|Z)$, we will draw samples $x^{(i)}_{1:m} \sim q(x_{1:m}|Z)$ for $i = 1, \ldots, N$ from a proposal pdf q, form which samples are easily generated. Then approximation to the density p is given by

$$p(x_{1:m}|Z) \approx \sum_{i=1}^{N} w^{(i)} \delta(x_{1:m} - x^{(i)}_{1:m}), \quad (2)$$

where δ is the Dirac delta function and

$$w^{(i)} = \frac{p(x^{(i)}_{1:m}|Z)}{q(x^{(i)}_{1:m}|Z)} \quad (3)$$

are normalized weights (so that they sum to one).

Since it is still hard to draw samples from q due to high dimensionality of $x_{1:m}$, Sequential Importance Sampling (SIS) is usually utilized. Following the order of dimensions in the vector of RVs $X = (X_1, \ldots, X_t)$ samples are generated

$$x^{(i)}_t \sim q(x_t|x_{1:t-1}, Z) = q(x_t|x_{1:t-1}, z_{1:t}) \quad (4)$$

for $t = 2, \ldots m$, and the particles are built sequentially $x^{(i)}_{1:t} = (x^{(i)}_{1:t-1}, x^{(i)}_t)$ for $i = 1, \ldots, N$. Since q factorizes as

$$q(x_{1:m}|Z) = q(x_1|Z) \prod_{t=2}^{m} q(x_t|x_{1:t-1}, Z), \quad (5)$$

we obtain that $x^{(i)}_{1:m} \sim q(x_{1:m}|Z)$.

The weights are recursively updated according to

$$w(x^{(i)}_{1:t}) = \frac{p(z_t|x^{(i)}_{1:t}, z_{1:t-1}) p(x^{(i)}_t|x^{(i)}_{1:t-1})}{q(x^{(i)}_t|x^{(i)}_{1:t-1}, z_{1:t})}\, w(x^{(i)}_{1:t-1}). \quad (6)$$

We show now that by recursively updating the weights according to (6) for $t = 2, \ldots, m$, the weight $w(x^{(i)}_{1:m})$ of particle (i) is equal to $w^{(i)}$.

$$w(x^{(i)}_{1:t}) = \frac{p(x^{(i)}_{1:t}|Z)}{q(x^{(i)}_{1:t}|Z)} = \frac{p(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)\; p(x^{(i)}_{1:t-1}|Z)}{q(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)\; q(x^{(i)}_{1:t-1}|Z)}$$

$$= \frac{p(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)}{q(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)}\, w(x^{(i)}_{1:t-1}) \quad (7)$$

$$= \frac{p(Z|x^{(i)}_{1:t})}{p(Z|x^{(i)}_{1:t-1})} \frac{p(x^{(i)}_t|x^{(i)}_{1:t-1})}{q(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)}\, w(x^{(i)}_{1:t-1}) \quad (8)$$

Eq. (8) follows from (7) by Bayes rule interchanging $x^{(i)}_t$ and Z in $p(x^{(i)}_t|x^{(i)}_{1:t-1}, Z)$. It remains to show that the first fraction in (8) is equal to $p(z_t|x^{(i)}_{1:t}, z_{1:t-1})$. This is true, since by keeping in mind that $Z = z_{1:t}$ and by factorizing we obtain

$$p(Z|x^{(i)}_{1:t}) = p(z_t|x^{(i)}_{1:t}, z_{1:t-1})\; p(z_{1:t-1}|x^{(i)}_{1:t-1})$$
$$= p(z_t|x^{(i)}_{1:t}, z_{1:t-1})\; p(Z|x^{(i)}_{1:t-1}) \quad (9)$$

We have just derived the SIS theorem:

Theorem. By sampling particles according to (4) and weighting them according (6), we obtain a set of weighted samples form $p(x_{1:m}|Z)$ once $t = m$. Consequently, we

can approximate $p(x_{1:m}|Z)$ with any precision if the number of particles N is sufficiently large. Thus, we can write

$$p(x_{1:m}|Z) \approx \sum_{i=1}^{N} w(x_{1:t}^{(i)})\delta(x_{1:m} - x_{1:m}^{(i)}), \qquad (10)$$

A common assumption underlying the equations (4) and (6) is that there exist two known functions f and h such that

$$x_t = f(x_{t-1}) + u_t \qquad (11)$$
$$z_t = h(x_t) + v_t \qquad (12)$$

where both u_t and v_t are mutually independent and identically distributed sequences with known probability density functions (pdfs). Often they are assumed to be Gaussians that model state prediction noise u_t and observation noise v_t. (The assumption in (11) can be replied by a weaker one: $x_t = f(x_{1:t-1}) + u_t$, i.e., that we can determine x_t if we know all previous states $x_{1:t-1}$. For the simplicity of presentation we will use (11).

The equation (6) can be simplified by making a common assumption that $q(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t}) = p(x_t^{(i)}|x_{1:t-1}^{(i)})$, which yields

$$w(x_{1:t}^{(i)}) = w(x_{1:t-1}^{(i)})p(z_t|x_{1:t}^{(i)}, z_{1:t-1}), \qquad (13)$$

and, consequently, the samples are generated from

$$x_t^{(i)} \sim p(x|x_{1:t-1}^{(i)}), \qquad (14)$$

i.e., the current observations $z_{1:t-1}$ have no direct influence on the samples. The influence of observations on the samples is introduced when particles are resampled according to their weights, which is a common step in PF algorithms, since particle weights depend on observations.

As a summery of this section, we outline a **standard PF algorithm**. For every time step $t = 1, \ldots, m$ and for every particle $i = 1, \ldots, N$ execute the following three steps:
1) **Importance sampling / proposal:** Sample followers of particle (i) according to (4) and set $x_{1:t}^{(i)} = (x_{1:t-1}^{(i)}, x_t^{(i)})$.
2) **Importance weighting / evaluation:** An importance weight is assigned to each particle $x_{1:t}^{(i)}$ according to (6).
3) **Resampling:** Sample with replacement N new particles form the current set of particles according to the weights. We obtain a set of new particles $x_{1:t}^{(i)}$ for $i = 1, \ldots, N$ all with weight of $1/N$. This procedure is called Sampling Importance Resampling (SIR) approach ¡cite¿.

2. New Approach

We illustrate our key ideas on an example of multi robot localization. A team of m robots obtained m observations $Z = \{z_1, \ldots, z_m\}$ by exploring their environment, where observation z_t comes from robot t. For example, if the robots are equipped with laser range scanners, then each z_t could be a vector of laser range readings representing distances to the closest obstacles from the robot t. Each RV X_t describes robot poses, i.e., its values x_t represent coordinates and the heading direction of the robot t. Our goal is to determine the state vector of robot poses $(x_1, \ldots, x_m)$ in a given top view map of the environment that maximizes the posterior distribution $p(X_1, \ldots, X_m|Z)$. In other words, $(x_1, \ldots, x_m)$ is a vector of most likely robot poses given the measurements Z.

It is possible to apply the classical PF robot localization by utilizing the order of the observations $Z = \{z_1, \ldots, z_m\}$, which follows the numbering of the robots. Hence the follower for each particle (i) is determined by importance sampling from the proposal distribution, i.e., sample $x_t^{(i)} \sim p(x|x_{1:t-1}^{(i)})$ and the particle weight is updated based on the evaluation of the observation z_t according to the recursive formula in Eq. 13. However, by doing so, we would have selected an arbitrary order, and in particular, if the robot localization task fails, it could be due to the selected order. Would we have selected a different order, the localization task could have been successful. Moreover, the observations $Z = \{z_1, \ldots, z_m\}$ are collected simultaneously at the same time. i.e., after each robot completed its exploration. Consequently, there is no reason to favor any particular order without utilizing further information.

In the proposed approach, the order of the observations is not predetermined, in particular, we do not follow the order of indices of the observations in Z. Our key idea is to utilize the PF framework to determine the most informative order of the observations. This way we are able to simultaneously find the most informative order and to utilize the observations in the order of their informativeness. Intuitively, it make sense, for example, if the first robot took its laser readings in the middle of a along a long corridor and the second robot at the entrance to a room with many distinctive features, then our approach will first process the laser readings obtained by the second robot, since they are more informative.

We stress that the SIS in Eq. 4 and particle evaluation in Eq. 6 utilize the sequential order of the RVs reflected in the order of dimensions in the state space $(x_1, \ldots, x_m)$. In many applications, this order is determined naturally by the time stamp of the observations, e.g., a single robot is collecting laser measurements at consecutive time points, in which case x_t denotes the robot pose at time t. The goal of our work is to extend SIS to applications in which there is no natural order of observations like the case of multi robot localization.

The key idea of the proposed approach is not to utilize the fix order of the dimensions, but instead compute the best possible order of the dimensions $(x_{i_1}, \ldots, x_{i_m})$ (or equivalently RVs) so that the corresponding sequence of observa-

tions $Z = (z_{i_1}, \ldots, z_{i_m})$ is most informative. For this we extend the underlying assumption (11) to the existence of a sequence of functions

$$x_s = f_{s|k}(x_k) + u_{s,k} \text{ forall } s, k \in \{1, \ldots, m\}, \quad (15)$$

which means that all RVs depend on each other, and each function $f_{s|k}$ can determine the value of variable s for any given value x_k of variable k. In our multi robot mapping example, $f_{s|k}$ allows to determine the pose of robot s when we know the pose of robot k. Consequently, (15) means that each robot knows the relative pose of the other robots. In comparison, in the standard assumption (11), we have $f = f_{t|t-1}$, i.e., we only can determine the value of the next state.

We observe that now the sequence of states visited before time t is not a sequence of consecutive numbers $(1, \ldots, t-1)$ but any subsequence $(i_1, \ldots, i_{t-1})$ formed by $t-1$ different numbers in $\{1, \ldots, m\}$, and the function $f_{s|k}$ allows us to determine not only the value of the next state but values of all remaining states. Due to noise factor $u_{s,k}$, each value x_s is an estimate or prediction of a true unknown value. Since a given state x_s can be determined based on all the states in the current sequence, we can combine all the predictions $f_{s|i_k}(x_{i_k}) + u_{s,i_k}$ for $k = 1, \ldots, t-1$ and improve the accuracy of the estimation of the state x_s.

The proposed sampling is as follows. At iteration t, the assumption (15) allows us to generate $m - t + 1$ samples from

$$x_{i_s}^{(i)} \sim p_{i_s|i_{t-1}}(x|x_{i_{1:t-1}}^{(i)}) \text{ for } s \in \{t, \ldots, m\}. \quad (16)$$

Hence at iteration t particle (i) has $m - t + 1$ followers. For example, if $m = 5$ and $x_{i_{1:3}}^{(i)} = x_{1,5,3}^{(i)}$, then particle (i) will have two followers $x_2^{(i)}$ and $x_4^{(i)}$. With reference to our multi robot example, when we determined the locations of robots $1, 3, 5$, we consider the two remaining robots 2 and 4 as the next robot whose pose we want to determine by particle (i). The poses of robots 2 and 4 are determined with reference to the pose of robot 3 as the last element of the sequence 1, 5, 3. Of course, we repeat an analogous process for each particle (i) for $i = 1, \ldots, N$, where N is the number of particles.

In contrast, in the standard application of rule (14), at each iteration t particle (i) has one follower. Even when sometimes each particle (i) has many followers, all followers are in the same dimension, which means that we only determine possible locations of say robot 2 by the followers and do not consider locations of robot 4 for particle (i), since a strict order of the state dimensions is followed in the classical setting.

3. Particle Filter with Static Observations

Our derivation is analog to the PF derivation, but it differs fundamentally, since unlike the standard PF framework, the observations Z do not arrive sequentially, but are available at once. To simplify the notation we replace the double indexing of the state variables $x_{i_s}^{(i)}$ with a bijection (onto and one-to-one function) $< \cdot >^{(i)}: \{1, \ldots, m\} \to \{1, \ldots, m\}$. Although we may have a different bijection for each particle, we will drop the index (i) from $< 1 : t >^{(i)}$, since the state variables carry the particle index. For example, we will denote $(x_{i_1}^{(i)}, x_{i_2}^{(i)}, x_{i_3}^{(i)}) = x_{i_{1:3}}^{(i)}$ as $x_{<1:3>}^{(i)}$.

We derive now the recursive weight update formula for the static observations Z. For every t from 2 to m, we have

$$\begin{aligned} w(x_{<1:t>}^{(i)}) &= \frac{p(x_{<1:t>}^{(i)} \mid Z)}{q(x_{<1:t>}^{(i)} \mid Z)} \\ &= \frac{p(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)\, p(x_{<1:t-1>}^{(i)}|Z)}{q(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)\, q(x_{<1:t-1>}^{(i)}|Z)} \\ &= \frac{p(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)}{q(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)}\, w(x_{<1:t-1>}^{(i)}) \\ &= \frac{p(Z|x_{<1:t>}^{(i)}) p(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)})}{p(Z|x_{<1:t-1>}^{(i)}) q(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)}\, w(x_{<1:t-1>}^{(i)}) \end{aligned} \quad (17)$$

To obtain the last equation, we apply Bayes rule to decompose $p(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z)$ that interchanges $x_{<t>}^{(i)}$ and Z.

As it is often the case in PF applications, we assume that $q(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)}, Z) = p(x_{<t>}^{(i)}|x_{<1:t-1>}^{(i)})$. Using this simple exploration based proposal the weight recursion in (17) becomes:

$$w(x_{<1:t>}^{(i)}) = \quad = w(x_{<1:t-1>}^{(i)}) \frac{p(Z|x_{<1:t>}^{(i)})}{p(Z|x_{<1:t-1>}^{(i)})} \quad (18)$$

By recursive substitution of weights in (18), i.e., by applying (18) to $w(x_{<1:t-1>}^{(i)}), w(x_{<1:t-2>}^{(i)}), \ldots, w(x_{<1:2>}^{(i)})$, we obtain

$$\begin{aligned} w(x_{<1:t>}^{(i)}) &= w(x_{<1:t-2>}^{(i)}) \frac{\cancel{p(Z|x_{<1:t-1>}^{(i)})}}{p(Z|x_{<1:t-2>}^{(i)})} \frac{p(Z|x_{<1:t>}^{(i)})}{\cancel{p(Z|x_{<1:t-1>}^{(i)})}} \\ &= \cdots = w(x_{<1>}^{(i)}) \frac{p(Z|x_{<1:t>}^{(i)})}{p(Z|x_{<1>}^{(i)})} \end{aligned} \quad (19)$$

Under the assumption that all particles have the same initial weight $w(x_{<1>}^{(i)})$ and the same initial observation probability $p(Z|x_{<1>}^{(i)})$ for $i = 1, \ldots, N$, we obtain

$$w(x_{<1:t>}^{(i)}) = p(Z|x_{<1:t>}^{(i)}) \quad (20)$$

Since for $t = m$ we have $w(x^{(i)}_{<1:m>}) = w(x^{(i)}_{1:m})$, we obtain that the weights computed by the recursive formulas (18) - (20) are equal to the weights in Eq. (3).

For comparison, the corresponding weight update in the standard PF framework ([5]) is

$$w(x^{(i)}_{1:t}) = w(x^{(i)}_{1:t-1})\, p(z_t|x^{(i)}_{1:t-1}, x_t), \qquad (21)$$

where z_t denotes the new observations obtained at time t. Because our observations Z do not have any natural order, Z cannot be expressed as a sequence of observations. We do not make any Markov assumption in the proposed formula (20), i.e., the new state $x^{(i)}_{<t>}$ is dependent on all previous states $x^{(i)}_{<1:t-1>}$ for each particle (i).

We outline now the proposed PF algorithm with static observations (PFSO). For every time step $t = 1, \ldots, m$ and for every particle $i = 1, \ldots, N$ execute the following three steps:

1) **Importance sampling / proposal:** Sample followers of particle (i) for $s \in \{1, \ldots, m\} \setminus < 1 : t-1 >$

$$x^{(i)}_s \sim p_{s|<t-1>}(x|x^{(i)}_{<1:t-1>}) \qquad (22)$$

and set $x^{(i)}_{<1:t-1>,s} = (x^{(i)}_{<1:t-1>}, x^{(i)}_s)$. This step is possible by assumption (15). We sample one follower for each RV X_s whose dimension index s is not in $< 1 : t-1 >$. Consequently, we have $m - t + 1$ followers at step t. In the first iteration ($t = 1$) we generate m samples

$$x^{(i)}_{<1>} = x^{(i)}_s \sim p_s(x) \;\text{ for }\; s \in \{1, \ldots, m\}. \qquad (23)$$

2) **Importance weighting/evaluation:** An individual importance weight is assigned to each follower of each particle according to Eq. 20.

3) **Resampling:** At the sampling step we sample more followers than the number of particles. Thus we have a larger set of particles $x^{(i)}_{<1:t-1>,s}$ for $i = 1, \ldots, N$ and

$$s \in \{1, \ldots, m\} \setminus < 1 : t-1 >$$

from which we sub-sample N particles and assign qual weights to all of them as in the standard Sampling Importance Resampling (SIR) approach ¡cite¿. We obtain a set of new particles $x^{(i)}_{<1:t>}$ for $i = 1, \ldots, N$. The resampling is not performed in the last step, i.e., when $t = m$.

Theorem. The PFSO algorithm computes an approximation of Eq. 2, i.e., for $t = m$ the particles $x^{(i)}_{<1:t>}$ with $i = 1, \ldots, N$ provide an approximation to the posterior $p(x_{1:m}|Z)$ for sufficiently large N, i.e.,

$$p(x_{1:m}|Z) \approx \sum_{i=1}^{N} w(x^{(i)}_{1:m})\delta(x_{1:m} - x^{(i)}_{1:m}). \qquad (24)$$

Proof. Our proof is based on Eq. (24) with weights defined in Eq. (3). We observe that the weights computed by the recursive formulas (18) - (20) are equal to the weights in Eq. (3), since for $t = m$ we have $w(x^{(i)}_{<1:m>}) = w(x^{(i)}_{1:m})$.

The Sampling Importance Resampling (SIR) replaces weighted particles with N particles with the weight equal to $1/N$, which provides an approximation to the same target pdf. This proves the theorem.

The fact that we can consider more than one follower of each particle and reduce the number of followers by resampling is known in the PF literature and is referred to as prior boosting [2, 1]. It is used to capture multi-modal likelihood regions. We stress that the resampling plays in our framework an additional and a very crucial role. It selects the the most informative random variables (i.e., state space dimensions) as followers of particles. Since the weight of $x^{(i)}_{<1:t-1>,s}$ is determined by the observations Z, and the resampling uses the weights to selects a follower $x_{<t>} = x_s$ from not yet considered dimensions

$$s \in \{1, \ldots, m\} \setminus < 1 : t-1 >,$$

the resampling determines the order of RVs, i.e., the bijection $< t >$ for $t = 1, \ldots m$. Consequently, the order of RVs is heavily determined by Z, and this order may be different for each particle (i). This is in strong contrast to the classical PF, where observations Z have no influence on the order of RVs, which is fixed.

The weight (20) of each particle is based on the evaluation how the predicted observations $h(x^{(i)}_{<1:t-1>,s})$ differ from the current observations $z_{<1:t-1>,s}$. Consequently, in the proposed approach, the weights of different particles are evaluated with respect to different observations. Thus, in the proposed approach the value of the weight depends on two factors,

- how descriptive a given vector of observations $z_{<1:t-1>,s}$ is and
- how good the prediction $h(x^{(i)}_{<1:t-1>,s})$ of observation $z_{<1:t-1>,s}$ is.

4. Rao-Blackwellized Particle Filter

The following derivation is inspired by Rao-Blackwellized particle filter often used in SLAM [3]. Our goal is to derive the posterior $p(x_{1:m}, \mu|Z)$ not only over the set of stats $x_{1:m}$ but also over possible shape models μ. To make this estimation possible, we use the following factorization

$$p(x_{1:t}, \mu|Z) = p(\mu|x_{1:t}, Z)\, p(x_{1:t}|Z) \qquad (25)$$

for $t = 1, \ldots, m$. This factorization allows us to first estimate only the set of states and then compute the shape

model. The key assumption here is that the shape model is a deterministic function of the set of states. Hence, the posterior over models $p(\mu|x_{1:t}, Z)$ can be computed analytically since $x_{1:t}, Z$ are known. This technique is often referred to as Rao-Blackwellization.

By substituting Eq. (25) to Eq. (24), we obtain

$$p(x_{1:m}, \mu|Z) \approx p(\mu|x_{1:m}, Z) \sum_{i=1}^{N} w(x_{1:m}^{(i)}) \delta(x_{1:m} - x_{1:m}^{(i)})$$

$$= \sum_{i=1}^{N} p(\mu|x_{1:m}, Z) w(x_{1:m}^{(i)}) \delta(x_{1:m} - x_{1:m}^{(i)}) \quad (26)$$

By denoting $u(x_{1:m}^{(i)}) = p(\mu|x_{1:m}^{(i)}, Z) w(x_{1:m}^{(i)})$, we obtain

$$p(x_{1:m}, \mu|Z) \approx \sum_{i=1}^{N} u(x_{1:m}^{(i)}) \delta(x_{1:m} - x_{1:m}^{(i)}), . \quad (27)$$

Consequently, if we modify the weight (20) of each particle to

$$u(x_{<1:t-1>}^{(i)}) = p(\mu|x_{<1:t-1>}^{(i)}, Z) p(Z|x_{<1:t-1>}^{(i)}), \quad (28)$$

we obtain a PF algorithm that estimates the posterior $p(x_{1:t}, \mu|Z)$.

5. Improved Importance Sampling from MRFs

In this section we will develop the key motivational story behind our work i.e. sampling from the posterior distribution of a Markov Random Field. As is well known in computer vision community many low level problems have been modeled using MRFs. A limited work is done in modeling mid-level and high-level problems using MRFs not because they cannot be modeled in such a way but the inference process becomes computationally hard. In fact for a general MRF, inference becomes NP-hard. This is because inference in MRFs is closely related to assignment problems. Except for simple costs of assignments the assignment problem quickly turns out to be intractable. Our contribution is to be able to infer to a reasonable degree on the problem instances of contour grouping and object recognition where we can exploit certain results from importance sampling theory to effectively navigate the exponential space of assignments. this has applications in clustering, object recognition, contour grouping and any general applications where inferences can be modeled using Markov Random Fields or Conditional Random Fields.

6. Markov Random Fields

Let $S = \{s_1, s_2, \ldots, s_m\}$ be a family of random variables (RVs), which take the values $e_i \in E = \{e_1, e_2, \ldots, e_n\}$. E is called a set of labels. $s_i = e_i$ denotes the random event that the RV s_i gets e_i assigned. If f is a short hand for $(s_1 = e_1, s_2 = e_2, \ldots, s_m = e_m)$ and S forms a MRF as per definitions in [**?**]:

$$p(f) = \frac{1}{Z} e^{-\frac{1}{T}U(f)} \quad (29)$$

where Z is the normalizing constant and T is the temperature parameter determining the sharpness of the distribution: high-temperature makes all configurations equally likely since the effect of $U(f)$ goes down. Simply put $e^{-\frac{1}{T}U(f)} \to 1$ as $T \to \infty$.

At given T and particular U we can sample "patterns" of assignments by sampling from p.

The state space to be explored to understand the patterns of the posterior p is n^m. At lower Ts it finding Maximum A Posteriori (MAP) estimates of p has important applications in high-level vision problems. Thus one of the goals of sampling from MRF is:

$$\widehat{f} = \underset{f \in E^S}{\operatorname{argmax}}\, p(f) \quad (30)$$

There have been many sampling algorithms like Gibbs sampler, Hot Coupling ([4]), Tree sampling, Swendsen-Wang sampling etc. But most of them assume restrictive conditional independences. Recently Hamze et. al. proposed a very generic importance sampling method called Large Flip Importance Sampling (LFIS) to sample from the posterior [**?**]. The main motivation for their approach comes from N-Fold Way (NFW, [**?**]) and Tabu search ([**?**]) where they use heuristics to improve the sampling of the exponential state space using memory and heuristics to design good moves in the state space. Since the moves are no-longer MCMC in the traditional sense they introduce importance weights to the distinct states visited by N copies of the sampler. Independently there has been an application of similar strategy using particle filters with static observations in [**?**]. In this paper we combine the strengths of both the approaches and present an improved sampler that employs better weigthing scheme and navigational strategy to explore state space so as to compute MAP in an efficient way.

We first present a brief overview of both the approaches and then combine both into an integrated approach.

6.1. Large Flip Importance Sampling

Sampling algorithms used in computer vision often tend to use different terminology which might makes things a bit difficult to understand convergence complexity issues. Hence, even though the Gibbs sampler is the simplest MCMC sampler for MRF we would like to present the algorithm below so as to make the connections between different approaches explicit. where f_i indicates the random event $s_i = e_i$. The above algorithm shows that the invariance kernel at a particular iteration above, for MCMC type

Algorithm 1 Gibbs sampler

1: Initialize a sample: $f^{(0)} = \left(s_1^{(0)} = e_1^{(0)}, s_2^{(0)} = e_2^{(0)}, \ldots, s_m^{(0)} = e_m^{(0)}\right)$.
2: **for** $i = 1$ to T **do**
3: Draw $f_1^{(i)} \sim p(f_1 | f_2^{(i-1)}, f_3^{(i-1)}, \ldots, f_m^{(i-1)})$.
4: Draw $f_2^{(i)} \sim p(f_2 | f_1^{(i)}, f_3^{(i-1)}, \ldots, f_m^{(i-1)})$.
5: Draw $f_m^{(i)} \sim p(f_m | f_1^{(i)}, f_2^{(i)}, \ldots, f_{m-1}^{(i)})$.
6: **end for**
7: Return $\{f^{(1)}, f^{(2)}, \ldots, f^{(T)}\}$.

convergence is defined as:

$$K(f, f') = \frac{1}{m} \sum_{i=1}^{m} p(f_i | f_{M \setminus i}) \tag{31}$$

The convergence condition says

$$\| K^{(T)}(f^{(0)}, f') - p(f') \| \to 0, \text{ as } T \to \infty \tag{32}$$

As can be seen from above there are two computational complexity issues in a sampling algorithm viz. (1) simulation complexity, (2) optimization complexity. Simulation complexity is because of the computations needed in drawing the samples while the optimization complexity involves T. The longer one runs the algorithm the closer one gets to the optimum. Of course in the most general case T can be exponentially large. Most practical algorithm designs involve in reducing T so as to visit non-trivial and important states of E^S as soon as possible without wasting computing resources. Such design algorithms are called *event driven* MCMC approaches. Motivated from NFW (algorithm below) LFIS was developed to address the "cycling" problem of NFW. where $\widehat{f}^{(i)} = f^{(\Theta_i \ldots \Theta_{i+1}-1)}$ i.e. the above algorithm avoids visiting duplicate states and the actual mixing time of the equivalent Gibbs sampler is Θ_T. Hence NFW effectively reduced the mixing time by cleverly covering more states in less time. Geometric$(p_{\text{flip}}(\widehat{f}_1^{(i-1)}, \widehat{f}_2^{(i-1)}, \ldots, \widehat{f}_m^{(i-1)}))$ is the probability that the $\widehat{f}^{(i-1)}$ changes over the next iteration. $\nu(j, e, \widehat{f}^{(i-1)})$ is the discrete posterior probability of the joint event "variable j assumes value e" given that a change in state occurred, i.e. $\widehat{f}^{(i)} \neq \widehat{f}^{(i-1)}$. Essentially the NFW simulates $T - 1$ *flips*.

Algorithm 2 N-Fold Way

1: Initialize a sample: $\widehat{f}^{(0)} = \left(s_1^{(0)} = e_1^{(0)}, s_2^{(0)} = e_2^{(0)}, \ldots, s_m^{(0)} = e_m^{(0)}\right)$.
2: **for** $i = 1$ to T **do**
3: Draw $\tau \sim$ Geometric$(p_{\text{flip}}(\widehat{f}_1^{(i-1)}, \widehat{f}_2^{(i-1)}, \ldots, \widehat{f}_m^{(i-1)}))$.
4: Draw $\widehat{f}^{(i)} \sim \nu(j, e, \widehat{f}^{(i-1)})$.
5: Set $\Theta_i = \Theta_i + \tau$.
6: **end for**
7: Return $\{\widehat{f}^{(1)}, \widehat{f}^{(2)}, \ldots, \widehat{f}^{(T)}\}$.

One reason for longer mixing times for some posteriors is the conditionals involved in the kernel can make the state not change i.e. $f = f'$ for a long time. NFW exploits the clever perspective of "simulating" the long waiting time if the conditionals for change of state space visited can be computed. The main problem involved is in computing this "change conditionals" which if not computed using exhaustive flips can result in "cycling" of the states i.e. same states get visited over and over again instead of visiting new states. LFIS avoids this by having explicit memory and running N copies of the sampler. After the run it computes importance weights to each of the unique states visited as follows:

$$w^{(i)} = \frac{p(\widetilde{f}^{(i)})}{\sum_{i=1}^{S_L} p(\widetilde{f}^{(i)})} \tag{33}$$

where $S_L \subset E^S$. These importance samples represent the posterior $p(f|S_L)$ instead of $p(f)$. Obviously if the algorithm is run long enough $S_L \to E^S$ and the samples represent the posterior $p(f)$.

Algorithm 3 Large Flip Importance Sampling

1: Initialize a sample: $\widehat{f}^{(0)} = \left(s_1^{(0)} = e_1^{(0)}, s_2^{(0)} = e_2^{(0)}, \ldots, s_m^{(0)} = e_m^{(0)}\right)$.
2: $k = 1$.
3: $\Gamma_k \sim \mathcal{U}[\Gamma_{\min}, \Gamma_{\max}]$
4: $n = 0$.
5: $F_0^k = \phi$.
6: **for** $i = 1$ to T **do**
7: Draw $\widehat{f}^{(i)} \sim \nu(j, e, \widehat{f}^{(i-1)})$.
8: $\mathcal{F}_n^k = \mathcal{F}_{n-1}^k \cup \widehat{f}^{(i)}$.
9: $n = n + 1$.
10: **if** $n = \Gamma_k$ **then**
11: $k = k + 1$.
12: $\Gamma_k \sim \mathcal{U}[\Gamma_{\min}, \Gamma_{\max}]$
13: $n = 0$.
14: $F_0^k = \phi$.
15: **end if**
16: **end for**
17: Pick all the *unique* states, $\{\widetilde{f}^{(i)}\}_{i=1}^{T}$, from $\{\mathcal{F}_{\Gamma_k}^k\}$ for all available k.
18: Assign importance weights to those unique states as per Eq. (33).
19: Return $\{(\widetilde{f}^{(1)}, w^{(1)}), (\widetilde{f}^{(2)}, w^{(2)}), \ldots, (\widetilde{f}^{(T)}, w^{(T)})\}$.

6.2. Particle Filters with Static Prior

Another recent work that uses importance sampling for sampling from MRF is in [**?**]. Owing to the similarity to the notations used in robot mapping, they call it particle filters with static prior. There the idea is to sample states *sequentially* and in such a way that samples build-up the assignments bottom-up using clever conditionals. Below we summarize that approach using the terminology used in this paper. $f_{<1:t>}$ denotes t random variables are instantiated not necessarily s_1 through s_t. Also $p(f_{<t>}|f_{<1:t-1>})$ denotes the conditional probability distribution of instantiating *a* random variable which has not been instantiated so far.

Assume there are N particles. Let's take a closer look at the journey of one particle. By definition $f_{<1:m>} \equiv f$ since f has m random variables. Since there are only m random variables the equivalent of number of iterations (T) in Gibbs sampler needed is only m. The importance samples $\{(f^{(i)}_{<1:m>}, w^{(i)}_{<1:m>})\}$ represent the posterior $p(f_{<m>}|f^{(i)}_{<1:m-1>})$. But there are three key observations:

Algorithm 4 Particle filter with static prior

1: **for** $i = 1$ to N **do**
2: Draw $f^{(i)}_{<1>} \sim p(f_{<1>})$.
3: Set $w^{(i)}_{<1>} = p(f^{(i)}_{<1>})$.
4: Draw $f^{(i)}_{<2>} \sim p(f_{<2>}|f_{<1>}(i))$.
5: Set $f^{(i)}_{<1:2>} = \{f^{(i)}_{<2>}, f^{(i)}_{<1>}\}$.
6: Set $w^{(i)}_{<1:2>} = w^{(i)}_{<1>} * p(f^{(i)}_{<2>}|f_{<1>}(i))$. ⋮
7: Draw $f^{(i)}_{<m>} \sim p(f_{<m>}|f^{(i)}_{<1:m-1>})$.
8: Set $f^{(i)}_{<1:m>} = \{f^{(i)}_{<m>}, f^{(i)}_{<1:m-1>}\}$.
9: Set $w^{(i)}_{<1:m>} = w^{(i)}_{<1:m-1>} * p(f^{(i)}_{<m>}|f^{(i)}_{<1:m-1>})$.
10: **end for**
11: Return $\{(f^{(1)}_{<1:m>}, w^{(1)}_{<1:m>}), (f^{(2)}_{<1:m>}, w^{(2)}_{<1:m>}), \ldots, (f^{(N)}_{<1:m>}, w^{(N)}_{<1:m>})\}$.

- If N is large enough and the conditionals are designed properly the samples would represent non-trivial patterns in the posterior and since in [**?**] the goal was to identify good matching between model and image contours the samples were able to serve the purpose.

- The samples are weighted *incrementally* using recursive importance weighting unlike LFIS where the samples are weighted at the end of the sampling process. For derivations of the recursive importance weighting please refer to [**?**].

- Although the samples are not generated in a standard MCMC fashion like Gibbs sampler the importance weighting and the conditional distributions result in useful samples. This is the key basis for LFIS also.

Acknowledgments

This work was supported in part by NSF IIS-0812118 and by DOE DE-FG52-06NA27508 grants. N. Adluru is supported by CIBM and MIR at the UW-Madison.

References

[1] J. Carpenter, P. Clifford, and P. Fearnhead. Building robust simulation-based filters for evolving data sets. Technical report, Dept. of Statistics, University of Oxford, 1999. 4

[2] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings of*, volume 140, pages 107–113, April 1993. 4

[3] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Trans. on Robotics*, 23:34–46, 2007. 4

[4] F. Hamze and N. de Freitas. Hot coupling: A particle approach to inference and normalization on pairwise undirected graphs of arbitrary topology. In *NIPS*, pages 1–8, 2005. 5

[5] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press Cambridge, 2005. 4