Published in final edited form as:

Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit. 2011 June ; 2011: 2873–2880. doi:10.1109/CVPR.2011.5995535.

Particle Filter with State Permutations for Solving Image Jigsaw Puzzles

Xingwei Yang[1], **Nagesh Adluru**[2], and **Longin Jan Latecki**[1]

Xingwei Yang: xingwei@temple.edu; Nagesh Adluru: adluru@wisc.edu; Longin Jan Latecki: latecki@temple.edu

[1]Department of Computer and Information Sciences, Temple University, Philadelphia

[2]Department of Biostatistics and Medical Informatics, University of Winsconsin-Madison

Abstract

We deal with an image jigsaw puzzle problem, which is defined as reconstructing an image from a set of square and non-overlapping image patches. It is known that a general instance of this problem is NP-complete, and it is also challenging for humans, since in the considered setting the original image is not given. Recently a graphical model has been proposed to solve this and related problems. The target label probability function is then maximized using loopy belief propagation. We also formulate the problem as maximizing a label probability function and use exactly the same pairwise potentials. Our main contribution is a novel inference approach in the sampling framework of Particle Filter (PF). Usually in the PF framework it is assumed that the observations arrive sequentially, e.g., the observations are naturally ordered by their time stamps in the tracking scenario. Based on this assumption, the posterior density over the corresponding hidden states is estimated. In the jigsaw puzzle problem all observations (puzzle pieces) are given at once without any particular order. Therefore, we relax the assumption of having ordered observations and extend the PF framework to estimate the posterior density by exploring different orders of observations and selecting the most informative permutations of observations. This significantly broadens the scope of applications of the PF inference. Our experimental results demonstrate that the proposed inference framework significantly outperforms the loopy belief propagation in solving the image jigsaw puzzle problem. In particular, the extended PF inference triples the accuracy of the label assignment compared to that using loopy belief propagation.

1. Introduction and Problem Formulation

As shown in [5] the jigsaw puzzle problem is NP-complete if the pairwise affinity among jigsaw pieces is unreliable. Following [2], we focus on reconstructing the original image from square and non-overlapping patches. This type of puzzles does not contain the shape information of individual pieces, which is quite important to determine the pairwise affinities among them. This makes the problem more challenging, since it is more difficult to evaluate pairwise affinities among puzzles. This is different from most of the previous approaches [14, 9, 18, 22], where the shape of the puzzle pieces is utilized. While [2] also considers priors on the target image layout, we do not assume any prior knowledge on the image layout. Thus, only local image content information of the puzzle pieces is available in our framework, e.g., see Fig. 1.

Now we briefly review the PF inference. We begin with a classical tracking example. A robot is moving around and taking images at discrete time intervals. The images form a sequence of observations $Z = (z_1, \ldots, z_m)$, where z_t is an image taken at time t. With each observation z_t there is associated a hidden state x_t. In our example the value of x_t is the robot pose (its 2D position plus orientation). The goal of PF inferences, is to derive the most likely sequence of the hidden states, i.e., to find a state vector $x_{1:m} = (x_1, \ldots, x_m)$ that maximizes the posterior $p(x_{1:m}/Z)$. We observe that here the observations are ordered following their time stamps. In PF inference, this order is utilized to sequentially infer the values of states x_t for $t = 1, \ldots, m$. Now imagine that the robot's clock broke and the time stamps are random. Thus, we are given a set of observations $Z = \{z_1, \ldots, z_m\}$, they are indexed but their index is irrelevant. Of course, we can still associate state x_t with observation z_t, but the set of observations is not ordered, and consequently, the corresponding states x_t are not ordered. Thus, we deal with unordered observations. This is exactly the scenario of the image jigsaw puzzle problem, e.g., see Fig. 1. We are given m square puzzle pieces described by a set of m observations $Z = \{z_1, \ldots, z_m\}$. Each observation z_t describes part of the original image depicted on piece t and is given by a vector of features, which are the color values of the pixels on piece t in our experimental results. The puzzle pieces are numbered with index t, but their numbering is random like the numbers in Fig. 1(b). The value of the state x_t of puzzle piece t is a location of an empty square in the square grid, e.g., the value of x_t is the index of an empty square in the square gird shown in Fig. 1(c). Our goal is to determine the state vector $x_{1:m}$ that maximizes the posterior probability $p(x_{1:m}/Z)$. Since the original image is not provided, this probability is determined based on pairwise appearance consistency of the local puzzle images, i.e., the posterior distribution is a function of how well adjacent pieces fit together once they are placed on the grid. In other words, a vector of grid locations $x_{1:m}$ maximizes $p(x_{1:m}/Z)$ if the puzzle pieces placed at these locations form the most consistent image. We observe that the posterior distribution $p(x_{1:m}/Z)$ usually is very complicated and has many local maxima. This is particulary the case when the local image information of the puzzle pieces is not very descriptive.

Our main contribution is a new PF inference framework that works in this scenario. In the proposed framework we extend PF to handle the situations where we have unordered set of observations that are given simultaneously. One of our key ideas is the fact that it is possible to extend the importance sampling from the proposal distribution so that different particles explore the state space along different dimensions. Then the particle resampling allows us to automatically determine most informative orders of observations (as permutations of state space dimensions). Consequently, we can use a rich set of proposal functions in the process of estimating the posterior distribution.

The classical PF framework has been developed for sequential state estimation like tracking [13, 19] or robot localization [20, 7]. There, the observations arrive sequentially and are indexed by their time stamps, as our tracking example illustrates. It is possible to apply the classical PF framework as stochastic optimization to solve this problem by utilizing a fix order of states. However, by doing so, we would have selected an arbitrary order, and the puzzle construction may fail because of the selected order and would require extremely large number of particles. Our framework on the other hand can work with fewer particles because

each particle explores different order. This gives us a rich set of proposal distributions as opposed to having one fixed. Moreover, the observations are given simultaneously at the same time. Hence, there is no reason to favor any particular order without utilizing this fact.

In our experimental results, we compare the solutions obtained by the proposed inference framework to the solutions of the loopy believe propagation under identical settings on the dataset from [2]. In particular, we use exactly the same dissimilarity-based compatibility of puzzle pieces. The proposed PF inference significantly outperforms the loopy believe propagation in all evaluation measures. The main measure is the accuracy of the label assignment, where the difference is most significant. The accuracy using loopy believe propagation is 23.7% while that using the proposed PF inference is 69.2%.

The rest of the paper is organized as follows. After introducing the preliminaries in §2, our key extensions for permuted PF are explained in §3 and §4. §5 provides implementation details. §6 shows and evaluates the experimental results not only the dataset from [2], but also an extended dataset. §7 describes related approaches.

2. Particle Filter Preliminaries

In this section we present some preliminary facts about Particle Filters (PFs). They will be utilized in the following sections when we introduce the proposed framework. Given is a sequence of observations $Z = (z_1, \ldots, z_m)$, i.e., the observations are ordered. Our goal is to maximize the posterior distribution $p(x_{1:m} / Z)$, i.e., to find the values $\hat{x}_t$ of states x_t such that

$$\hat{x}_{1:m} = \underset{x_{1:m}}{\operatorname{argmax}}\, p(x_{1:m}|Z), \quad (1)$$

where $x_{1:m} = (x_1, \ldots, x_m) \in \mathcal{X}^m$ is a state space vector and each state x_t has a corresponding observation z_t for $t = 1, \ldots, m$.

This goal can be achieved by approximating the posterior distribution with a finite number of samples in the framework of Bayesian Importance Sampling (BIS). Since it is usually difficult to draw samples from the probability density function (pdf) $p(x_{1:m}|Z)$, samples are drawn from a proposal pdf q, $x_{1:m}^{(i)} \sim q(x_{1:m}|Z)$ for $i = 1, \ldots, N$. Then approximation to the density p is given by

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

where $\delta_{x_{1:m}^{(i)}}(x_{1:m})$ denotes the delta-Dirac mass located at $x_{1:m}^{(i)}$ and

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

are the importance weights of the samples. Typically the sample $x_{1:m}^{(i)}$ with the largest weight $w^{(i)}$ is then taken as the solution of (1).

Since it is still computationally intractable to draw samples from q due to high dimensionality of $x_{1:m}$, Sequential Importance Sampling (SIS) is usually utilized. In the classical PF approaches, samples are generated recursively following the order of dimensions in state vector $x_{1:m} = (x_1, \ldots, x_m)$:

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

for $t = 1, \ldots m$, and the particles are built sequentially $x_{1:t}^{(i)}=(x_{1:t-1}^{(i)}, x_t^{(i)})$ for $i = 1, \ldots, N$. The subscript t in q_t indicates from which dimension of the state vector the samples are generated. Since q factorizes as

$$q(x_{1:m}|Z)=q_1(x_1|Z)\prod_{t=2}^{m} q_t(x_t|x_{1:t-1}, Z), \quad (5)$$

we obtain that $x_{1:m}^{(i)} \sim q(x_{1:m}|Z)$. In other words, by sampling recursively $x_t^{(i)}$ from each dimension t according to (4) we obtain a sample from $q(x_{1:m}/Z)$ at $t = m$.

Since at a given iteration we have a *partial* state sample $x_{1:t}^{(i)}$ for $t < m$, we also need an evaluation procedure of this partial state sample. For this we observe that the weights can be recursively updated according to [21]:

$$w(x_{1:t}^{(i)})=\frac{p(z_t|x_{1:t}^{(i)}, z_{1:t-1})p(x_t^{(i)}|x_{1:t-1}^{(i)})}{q_t(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t})} w(x_{1:t-1}^{(i)}). \quad (6)$$

The above equation is derived from (3) using Bayes rule. Consequently, when $t = m$, the weight $w(x_{1:m}^{(i)})$ of particle (i) recursively updated according to (6) is equal to $w^{(i)}$ (defined in (3)). Hence, at $t = m$, we obtain a set of weighted (importance) samples from $p(x_{1:m}/Z)$, which is formally stated in the following theorem [4]:

Theorem 2.1

Under reasonable assumptions on the sampling (4) and weighting functions (6) given in [4], $p(x_{1:m}|Z)$ can be approximated with weighted samples $\{x_{1:m}^{(i)}, w(x_{1:m}^{(i)})\}_{i=1}^{N}$ with any precision if N is sufficiently large. Thus, the convergence in (7) is almost sure:

$$p(x_{1:m}|Z)=\lim_{N\to\infty}\sum_{i=1}^{N} w(x_{1:m}^{(i)})\delta_{x_{1:m}^{(i)}}(x_{1:m}). \quad (7)$$

In many applications, the weight equation (6) is simplified by making a common assumption that $q_t(x_t^{(i)}|x_{1:t-1}^{(i)}, z_{1:t})=p(x_t^{(i)}|x_{1:t-1}^{(i)})$, i.e., we take as the proposal distribution the conditional pdf of the state at time t conditioned on the current state vector $x_{1:t-1}^{(i)}$. This assumption simplifies the recursive weight update to

$$w(x_{1:t}^{(i)})=w(x_{1:t-1}^{(i)})p(z_t|x_{1:t}^{(i)}, z_{1:t-1}), \quad (8)$$

and implies that the samples are generated from

$$x_t^{(i)}\sim p_t(x|x_{1:t-1}^{(i)}). \quad (9)$$

Analogous to (4) p_t in (9) indicates the dimension of the state space from which the samples are generated.

Now we summarize the derived **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 (9) (a special case of (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 (8) (a special case of (6)).
3. **Resampling:** Sample with replacement N new particles form the current set of N particles

$$\{x_{1:t}^{(i)}|i=1,\ldots,N\}$$

according to their weights. We obtain a set of new particles $x_{1:t}^{(i)}$ for $i = 1, \ldots, N$, and renormalize their weights to sum to one. This procedure is a variant of Sampling Importance Resampling (SIR) [21]. It is an important

part of any PF algorithm, since resampling prevents weight degeneration of particles.

3. Key Extension to Permuted States

As stated above, the standard SIS in Eq. 9 and particle evaluation in Eq. 8 utilize the sequential order of the states $x_{1:m} = (x_1, \ldots, x_m)$. Of course, this is the best choice in many applications where the order is determined naturally by the time stamp of the observations. In contrast, the proposed approach is aimed at scenarios where no natural order of observations is given and the observations Z are initially known as in the image jigsaw puzzle problem.

The key idea of the proposed approach is not to utilize the fix order of the states $x_{1:m} = (x_1, \ldots, x_m)$ induced by the order of observations Z, but instead explore different orders of the states $(x_{i_1}, \ldots, x_{i_m})$ such that the corresponding sequences of observations $(z_{i_1}, \ldots, z_{i_m})$ is most informative. In particular, we do not follow the order of indices of observations in Z. This way we are able to utilize the the most informative observations first allowing us to use a rich set of proposal functions. To achieve this we modify the first step of the PF algorithm so that the importance sampling is performed for every dimension not yet represented by the current particle. Intuitively, for example, if the first puzzle piece has a local image very similar to many other puzzle pieces and the second puzzle piece has a very distinctive local image that matches only a few other pieces, then our approach will first process the second puzzle piece, since it is more informative.

To formally define the proposed sampling rule, we need to explicitly represent different orders of states with a permutation $\sigma : \{1, \ldots, m\} \rightarrow \{1, \ldots, m\}$. We use the shorthand notation $\sigma(1:t)$ to denote $(\sigma(1), \sigma(2), \ldots, \sigma(t))$ for $t \leq m$. Each particle (i) now can have a different permutation $\sigma^{(i)}$ of the puzzle pieces *in addition* to their locations. Thus the particles are now represented as $x^{(i)}_{\sigma(1:t)}$. We drop the superscript (i) of $\sigma^{(i)}$ in the context of a particle which already carries the index (i). For example, Fig. 1(c) shows the configuration of a particle at time $t = 2$, where puzzle pieces numbered 3 and 1 in Fig. 1(b) are placed at locations (a) and (b), correspondingly. Hence $\sigma^{(i)}(1:2) = (3, 1)$ and $x^{(i)}_{\sigma(1:2)} = (a, b)$. Thus, a sequence of states $x_{\sigma(1:t-1)}$ visited before time t may be any subsequence $(i_1, \ldots, i_{t-1})$ of $t - 1$ different numbers in $\{1, \ldots, m\}$.

We are now ready to formulate the proposed importance sampling. At each iteration $t \leq m$, for each particle (i) and for each $s \in \overline{\sigma^{(i)}(1:t-1)}$, we sample

$$x^{(i)}_s \sim p_s(x | x^{(i)}_{\sigma(1:t-1)}), \quad (10)$$

where $\overline{\sigma^{(i)}(1:t-1)} = \{1, \ldots, m\} \backslash \sigma^{(i)}(1:t-1)$, i.e., the indices in $1:m$ that are not present in $\sigma^{(i)}(1:t-1)$ for $t \leq m$. The subscript s at the posterior pdf p_s indicates that we sample

values for state *s*. We generate at least one sample for each state $s \in \overline{\sigma^{(i)}(1{:}t-1)}$. This means that the single particle $x^{(i)}_{\sigma(1:t-1)}$ is multiplied and extended to several follower particles $x^{(i)}_{\sigma(1:t-1),s}$. Consequently, at iteration $t < m$ particle (*i*) has $m - t + 1$ followers. Each follower is a sample from a different dimension of the state (i.e., represents a location of a different puzzle piece). Going back to our toy puzzle example, we recall that the current state vector of particle (*i*) in Fig. 1(c) at time $t = 2$ is $x^{(i)}_{\sigma(1:2)} = (a, b)$, where $\sigma^{(i)}(1:2) = (3, 1)$. For sampling at time $t = 3$, we have $\overline{\sigma^{(i)}(1{:}t-1)} = (2, 4, 5, 6)$. Consequently, we sample four followers of particle (*i*) in (10), one for each state $s = 2, 4, 5, 6$, where $x^{(i)}_2$ is the sampled location of puzzle piece 2, $x^{(i)}_4$ is the sampled location of puzzle piece 4, and so on.

In contrast, in the standard application of rule (9), at each iteration *t* particle (*i*) has one follower. Even when sometimes each particle (*i*) has many followers, all followers are samples from the same state, since there is only one unique state at time *t*. For our toy example, this means for particle (i), only locations of say puzzle piece 2 are sampled and not those of puzzle piece 4, since a fixed order of the state dimensions is followed in the classical setting.

We do not make any Markov assumption in (10), i.e., the new state $x^{(i)}_s$ is dependent on all previous states $x^{(i)}_{\sigma(1:t-1)}$ for each particle (*i*).

4. Particle Filter with State Permutations

Now we are ready to outline the proposed **PF with state permutations (PFSP)** algorithm. In addition to the change is in the importance sampling step, the other two steps are also modified. 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 $x^{(i)}_s$ of particle (*i*) from each dimension $s \in \overline{\sigma^{(i)}(1{:}t-1)}$ according to (10), which we repeat here for completeness,

$$x^{(i)}_s \sim p_s(x | x^{(i)}_{\sigma(1:t-1)}), \quad (11)$$

and set $x^{(i,s)}_{\sigma(1:t)} = (x^{(i)}_{\sigma(1:t-1)}, x^{(i)}_s)$ and $\sigma^{(i,s)}(t) = s$, which means that $\sigma^{(i,s)}(1 : t) = (\sigma(1 : t-1), s)$. As stated before, we drop the superscript (*i; s*) in $x^{(i,s)}_{\sigma(1:t)}$, since it is already present as the particle index.

2. **Importance weighting/evaluation:** An individual importance weight is assigned to each follower particle $x^{(i,s)}_{\sigma(1:t)}$ according to

$$w(x^{(i,s)}_{\sigma(1:t)}) = w(x^{(i)}_{\sigma(1:t-1)}) p(z_s | x^{(i,s)}_{\sigma(1:t)}, z_{\sigma^{(i)}(1:t-1)}), \quad (12)$$

3. **Resampling:** Sample with replacement N new particles form the current set of $N \times (m - t + 1)$ particles

$$\{x^{(i,s)}_{\sigma(1:t)} | i = 1, \ldots, N, s \in \overline{\sigma^{(i)}(1:t-1)}\}. \quad (13)$$

according to the weights. Thus, we obtain a set of new particles $\{x^{(i)}_{\sigma(1:t)}\}_{i=1}^{N}$. We also renormalize their weights to sum to one. This is a variant of the standard Sampling Importance Resampling (SIR) step [21] as in the classical PF framework.

We observe that the particle weight evaluation in (12) is analogous to (8) in that the conditional probability of observation z_s is a function of two corresponding sequences of observations and states plus the state x_s. The only difference is that the sequences are determined by the permutation $\sigma^{(i)}(1 : t - 1)$.

Sampling more than one follower of each particle and reducing the number of followers by resampling is known in the PF literature as prior boosting [10, 1]. It is used to capture multi-modal likelihood regions. The resampling in our framework plays an additional and a very crucial role. It selects the the most informative orders of states. Since the weights of $w(x^{(i,s)}_{\sigma(1:t)})$ are determined by the corresponding order of observations $z_{\sigma^{(i)}(1:t-1)}$, and the resampling uses the weights to selects new particles $x^{(i)}_{\sigma(1:t)}$, the resampling determines the order of state dimensions. Consequently, the order of state dimensions is heavily determined by their corresponding observations, and this order may be different for each particle (i). This is in strong contrast to the classical PF, where observations are considered only in one order Z.

The fact that each particle explores a possibly different order of dimensions $\sigma^{(i)}(1 : m)$ is extremely important for the proposed PFSP, since it allows for use of rich set of proposal functions with fewer number of particles. However, at $t = m$ all state dimensions are present in each sample $x^{(i)}_{\sigma(1:m)}$. Hence we can reorder the sequence of state dimensions $\sigma^{(i)}(1 : m)$ to form the original order $1 : m$ by applying the inverse permutation $(\sigma^{(i)})^{-1}$ and obtain $x^{(i)}_{1:m} = x^{(i)}_{\sigma^{-1}\sigma(1:m)}$, i.e., the state values are sorted according to the original state indices $1 : m$ in each sample (i). In analogy to Theorem 2.1, we state the following:

Theorem 4.1

Under reasonable assumptions on the sampling (11) and weighting functions (12) given in [4], $p(x_{1:m}|Z)$ can be approximated with weighted samples $\{x_{1:m}^{(i)}, w(x_{\sigma(1:m)}^{(i)})\}_{i=1}^{N}$ with any precision if N is sufficiently large. Thus, the convergence in (14) is almost sure:

$$p(x_{1:m}|Z)=\lim_{N\to\infty}\sum_{i=1}^{N} w\left(x_{\sigma(1:m)}^{(i)}\right)\delta_{x_{1:m}^{(i)}}(x_{1:m}). \quad (14)$$

Proof—Due to Th. 2.1, we only need to show that $\{x_{1:m}^{(i)}, w(x_{\sigma(1:m)}^{(i)})\}_{i=1}^{N}$ represent weighted samples from $p(x_{1:m}|Z)$.

The key observation is that p and q are probabilities on joint distribution of m random variables, and as such the order of the random variables is not relevant. This follows from the fact that a joint probability is defined as the probability of the intersection of the sets representing events corresponding to the value assignments of the random variables, and set intersection is independent of the order of sets. Consequently, we have for every permutation σ

$$p(x_{\sigma(1:m)}|Z)=p(x_{1:m}|Z) \quad (15)$$

$$q(x_{\sigma(1:m)}|Z)=q(x_{1:m}|Z) \quad (16)$$

According to the proposed importance sampling (11), $x_{\sigma(1:m)}^{(i)}$ is a sample from $q(x_{\sigma(1:m)}|Z)$. Consequently, by (16), $x_{1:m}^{(i)}=x_{\sigma^{-1}\sigma(1:m)}^{(i)}$ is a sample from $q(x_{1:m}^{(i)}|Z)$ for each particle (i).

By the weight recursion in (12), and by (15) and (16)

$$w\left(x_{\sigma(1:m)}^{(i)}\right)=\frac{p(x_{\sigma(1:m)}^{(i)}|Z)}{q(x_{\sigma(1:m)}^{(i)}|Z)}=\frac{p(x_{1:m}^{(i)}|Z)}{q(x_{1:m}^{(i)}|Z)}. \quad (17)$$

Thus $\{x_{1:m}^{(i)}, w(x_{\sigma(1:m)}^{(i)})\}_{i=1}^{N}$ represent weighted samples from $p(x_{1:m}|Z)$. □

5. Implementation Details

In order to utilize the derived PF algorithm to solve the jigsaw puzzle problem, we need to design the proposal pdf in (11) and the conditional pdf of a new observation in (12). Both are detailed in this section.

Given are a set of m puzzle pieces $P = \{1, \ldots, m\}$ and a rectangular grid with m empty squares $G = \{g_1, \ldots, g_m\}$, e.g., see Fig. 1(b,c). In order to solve the image jigsaw puzzle we need to assign locations on G to the puzzle pieces in P. The observation associated with each puzzle piece (of size $K \times K$) is the color information of the partial image depicted on it, i.e., z_i is a $K \times K \times 3$ matrix of pixel color values and the set of observations is $Z = \{z_1, \ldots, z_m\}$.

A sample particle at time $t \leq m$ is given by $x_{\sigma(1:t)} = (x_{\sigma(1)}, \ldots, x_{\sigma(t)})$, where $\sigma(i) \in P$ and $x_{\sigma(i)} \in G$. This means the puzzle piece $\sigma(i)$ is placed on the grid square with index $x_{\sigma(i)}$. The corresponding observations $z_{\sigma(1:t)} = (z_{\sigma(1)}, \ldots, z_{\sigma(t)})$ represents the color information of the partial images on the puzzle pieces. In this section we drop the particle index (i), since all definitions apply to each particle.

We now define an affinity matrix A representing the compatibility of the local images on the puzzle pieces. It is a 3D matrix of size $m \times m \times 4$ with the third dimension being an adjacency type, since two puzzle pieces can be adjacent in four different ways: left/right, right/left, top/bottom, and bottom/top, which we denote with LR, RL, TB, and BT.

In order to be able to compare our experimental results to the results in [2] we define A following the definitions in [2]. They first define a dissimilarity-based compatibility D. Given two puzzle pieces j and i, D measures dissimilarity between their images z_j, z_i by summing the squared LAB color differences along their boundary, e.g., the left/right (LR) dissimilarity is defined as

$$D(j,i,LR) = \sum_{k=1}^{K} \sum_{c=1}^{3} (z_j(k,u,c) - z_i(k,v,c))^2, \quad (18)$$

where u indexes the last column of z_j and v indexes the first column of z_i. Finally, the affinity of the LR connection is given by

$$A(j,i,LR) = \exp\left(-\frac{D(j,i,LR)}{2\delta^2}\right), \quad (19)$$

where δ is adaptively set as the difference between the smallest and the second smallest D values between puzzle piece i and all other pieces in P, see [2] for more details.

Proposal and weights

The proposal distribution $p_s(x/x_{\sigma(1:t-1)}) : G \rightarrow \mathbb{R}$ is a discrete probability distribution of placing puzzle piece s on each grid square x. $p_s(x/x_{\sigma(1:t-1)}) = 0$ if x is occupied or is not adjacent to any square in $\sigma(1 : t-1)$. Now say x is free and is adjacent and is to the right of grid square $x_{\sigma(j)}$ for some $j = 1, \ldots, t$. Then

$$p_s(x|x_{\sigma(1:t-1)}) \propto A(s, \sigma(j), RL). \quad (20)$$

Hence this probability is proportional to the LR similarity between puzzle pieces s and $\sigma(j)$. The definition is analogous for the other three adjacency relations LR, TB, BT. If square x is adjacent to more than one grid squares in $\{x_{\sigma(j)}|j = 1, \ldots, t\}$, then $p_s(x|x_{\sigma(1:t-1)})$ is proportional to the product of the corresponding A values.

Let x_s be a sample from (11) at time t, and as above x_s is adjacent and is to the right of grid square $x_{\sigma(j)}$ for some $j = 1, \ldots, t$. The difference is that x_s is occupied now with the puzzle piece s. Then

$$p(z_s|x_{\sigma(1:t)}, z_{\sigma(1:t-1)}) \propto A(s, \sigma(j), RL). \quad (21)$$

The definition is analogous for the other three adjacency relations LR, TB, BT. If square x_s is adjacent to more than one grid squares in $\{x_{\sigma(j)}|j = 1, \ldots, t\}$, then $p(z_s|x_{\sigma(1:t)}, z_{\sigma(1:t-1)})$ is proportional to the product of the corresponding A values.

To summarize the proposal distribution is a function of how well puzzle piece s fits to the already placed pieces and assigns the probability of placing s to all grid squares, while in the evaluation we already know the grid location of puzzle piece s as well as its adjacent squares. We then use this information to compute the evaluation probability according to A. Hence, both the proposal and evaluation of a given particle are functions of how well adjacent pieces fit together following the order in which the pieces have been added.

For a given image jigsaw puzzle with m pieces, the time complexity of the proposed inference framework is $O(m^2N)$, where N is the number of particles. It follows form the fact that at iteration $t < m$ particle (i) has $m - t + 1$ followers. We set the number of particles $N = 10$ in all our experiments described in the next section.

6. Experimental Results

We compare the image jigsaw puzzle solutions obtained by the proposed PF inference framework to the solutions of the loopy believe propagation used in [2] under identical settings. We used the software released by the authors of [2] to obtain their results and also to compute the affinities defined in Section 5 used in our approach. The results are compared on the dataset provided in [2], which we call MIT Dataset. It is composed of 20 images. In addition, we also consider an extended dataset composed of 40 images, i.e., we added 20 images. As we will see below, the results of both methods on the original and extended datasets are comparable. Our implementations will be made publicly available on an accompanying website.

The experimental results in [2] are conducted in two different settings: with and without any prior on the target image layout. In [3] the prior of the image layout is given by a low resolution version of the original image. [2] weakens this assumption to a statistics of the possible image layout. We focus on the results without any prior of the image layout. Consequently, we focus on a harder problem, since we only use the pairwise relations

between the image patches, given by pair-wise compatibilities of located puzzle pieces as defined in Section 5.

In the probabilistic framework in [2], a puzzle piece is assigned to each grid location. In our PF framework, it is more natural to assign a grid location to each puzzle piece. The solutions of both methods are equivalent, since a final puzzle solution is a set of m pairs composed of (puzzle piece, grid location), where m is the number of the puzzle pieces. We call such pairs the solution pairs.

We use three types of evaluation methods introduced in [2]. Each method focuses on different aspects of the quality of the obtained puzzle solutions. The most natural and strictest one is **Direct Comparison**. It simply computes the percentage of correctly placed puzzle pieces, i.e., for a puzzle with m pieces, Direct Comparison is the number of correct solution pairs divided by m. A less stricter measure is **Cluster Comparison**. It tolerates an assignment error as long as the puzzle piece is assigned to a location that belongs to a similar puzzle piece. The puzzle pieces are first clustered into groups of similar pieces. Moreover, due to lack of prior knowledge of target image, the reconstructed image may be shifted compared to the ground truth image. Therefore, a third measure called **Neighbor Comparison** is used to evaluate the label consistency of adjacent puzzle pieces independent of their grid location, e.g., the location of two adjacent puzzle pieces is considered correct if two puzzle pieces are left-right neighbors in the ground truth image and they are also left-right neighbors in the inferred image. Neighbor Comparison is the fraction of correct adjacent puzzle pieces. This measurement does not penalize the accuracy as long as the adjacent patches in original image are adjacent in the reconstructed image.

The results on the MIT Dataset are shown in Table 1 and on the extended dataset in Table 2. The proposed PF inference framework significantly outperforms the loopy believe propagation in all three performance measures. Moreover, the reconstruction accuracy (according to the most natural measure, Direct Comparison) of the original images by our algorithm is improved three times.

In order to demonstrate that the considered image jigsaw puzzle problem is also very challenging to humans, we show some example results in Fig. 2. There we show the original images, but we would like to emphasize that the original images are not used during the inference. Fig. 2 also demonstrates that the reconstructed images obtained by the proposed algorithm compare very favorably to the results of [2]. In order to demonstrate the dynamic of the proposed PF inference, we show reconstructed images of the best particle at different times (iterations) in Fig. 3.

Both methods are initialized with one anchor patch, i.e., with one correct (puzzle piece, grid location) pair. We always assign a correct image patch to the top left corner of the image. In all our experiments we divide each test image into 108 square patches resulting in $m = 108$ puzzle pieces.

7. Related Work

The first work on Jigsaw Puzzle Problem was reported in [8]. Since shape is an important clue for accurate pairwise relation, many methods [14, 9, 18, 22] focussed on matching distinct shapes among jigsaw pieces to solve the problem. The pairwise relations among jigsaw pieces are measured by the fitness of shapes. There also exist approaches that consider both the shape and image content [16, 17, 23]. Most methods solve the problem with a greedy algorithm and report results on just one or few images. Our problem formulation only considers the image content following Cho et. al [2].

Particle filters (PF) are also known as sequential Monte Carlo methods (SMC) for model estimation based on simulation. There is large number of articles published on PF and we refer to two excellent books [6, 15] for an overview. PF can be viewed as a powerful inference framework that is utilized in many applications. One of the leading examples is the progress in robot localization and mapping based on PF [21]. Classical examples of PF applications in computer vision are contour tracking [12] and object detection [11]. All these approaches utilize PF in the classical tracking/filtering scenario with a pre-defined order of states and observations. To our best knowledge, the proposed PF framework with state permutations is novel and has not been considered before by other authors.

8. Conclusions and Future Work

We introduce a novel inference framework for solving image jigsaw puzzle problem. Our key contribution is an extension of the PF framework to work with unordered observations. Weighted particles explore the state space along different dimensions in different orders, and state permutations that yield most descriptive proposal functions are selected as new particles. By exploiting the equivalence of importance sampling under state permutations, we prove that the obtained importance samples represent samples from the original target distribution. We evaluate the performance of the proposed PF inference on a problem of image jigsaw puzzles. As the experimental results demonstrate, it significantly outperforms the loopy belief propagation. Image jigsaw puzzle problem is an instance of labeling (assignment) problem. Therefore, our future work will focus on a broader spectrum of labeling problems.

Acknowledgments

The work was supported by the NSF under Grants IIS-0812118, BCS-0924164, OIA-1027897, the AFOSR Grant FA9550-09-1-0207, the DOE Award 71498-001-09 and a CIBM-MIR fellowship from UW-Madison. The authors would like to thank Taeg Sang Cho for providing the code for his method in CVPR 2010.

References

1. Carpenter J, Clifford P, Fearnhead P. Building robust simulation-based filters for evolving data sets. 1999 2877.
2. Cho TS, Avidan S, Freeman WT. A probabilistic image jigsaw puzzle solver. CVPR. 2010 2873, 2874, 2877, 2878, 2879, 2880.
3. Cho TS, Butman M, Avidan S, Freeman WT. The patch transform and its applications to image editing. CVPR. 2008 2878.

4. Crisan D, Doucet A. A survey of convergence results on particle filtering methods for practitioners. IEEE Transactions on Signal Processing. 2002; 50(3):736–746. 2875, 2877.
5. Demaine ED, Demaine ML. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. Graphs and Combinatorics. 2007 2873.
6. Doucet, A.; Freitas, ND.; Gordon, N. Sequential Monte Carlo Methods in Practice. Springer Verlag; 2001. 2879
7. Fox, D.; Thrun, S.; Dellaert, F.; Burgard, W. Particle filters for mobile robot localization. In: Doucet, A.; de Freitas, N.; Gordon, N., editors. Sequential Monte Carlo Methods in Practice. Springer Verlag; New York: 2000. 2874
8. Freeman H, Garder L. Apictorial jigsaw puzzles: the computer solution of a problem in pattern recognition. IEEE TEC. 1964; 13:118–127. 2879.
9. Goldberg D, Malon C, Bern M. A global approach to automatic solution of jigsaw puzzles. Symposium on Computational Geometry. 2002 2873, 2879.
10. Gordon N, Salmond D, Smith A. Novel approach to nonlinear/non-gaussian bayesian state estimation. Radar and Signal Processing, IEE Proceedings of. Apr.1993 140:107–113. 2877.
11. Ioffe S, Forsyth D. Probabilistic methods for finding people. Int J Comput Vision. Jun.2001 43:45–68. 2879.
12. Isard M, Blake A. Contour tracking by stochastic propagation of conditional density. Proc European Conf on Computer Vision, ECCV. 1996:343–356. 2879.
13. Khan Z, Balch T, Dellaert F. An mcmc-based particle filter for tracking multiple interacting targets. ECCV. 2004 2874.
14. Kong W, Kimia BB. On solving 2d and 3d puzzles using curve matching. CVPR. 2001 2873, 2879.
15. Liue, J. Monte Carlo strategies in Scientific Computing. Springer Verlag; 2001. 2879
16. Makridis M, Papamarkos N. A new technique for solving a jigsaw puzzle. ICIP. 2006 2879.
17. Nielsen TR, Drewsen P, Hansen K. Solving jigsaw puzzles using image features. PRL. 2008 2879.
18. Radack G, Badler N. Jigsaw puzzle matching using a boundary-centered polar encoding. CGIP. 1982 2873, 2879.
19. Smith K, Gatica-Perez D, Odobez JM. Using particles to track varying numbers of interacting people. CVPR. 2005 2874.
20. Thrun S. Particle filters in robotics. Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI). 2002 2874.
21. Thrun, S.; Burgard, W.; Fox, D. Probabilistic Robotics. The MIT Press; Cambridge: 2005. 2875, 2876, 2879
22. Wolfson H, Schonberg E, Kalvin A, Lamdam Y. Solving jigsaw puzzles by computer. Annals of Operations Research. 1988 2873, 2879.
23. Yao FH, Shao GF. A shape and image merging technique to solve jigsaw puzzles. PRL. 2003 2879.

Figure 1.

The goal is to build the original image (a) given the jigsaw puzzle pieces (b). The original image is not known, thus, it needs to be estimated given the observations shown in (b). The empty squares in (c) form possible locations for the puzzle pieces in (b).

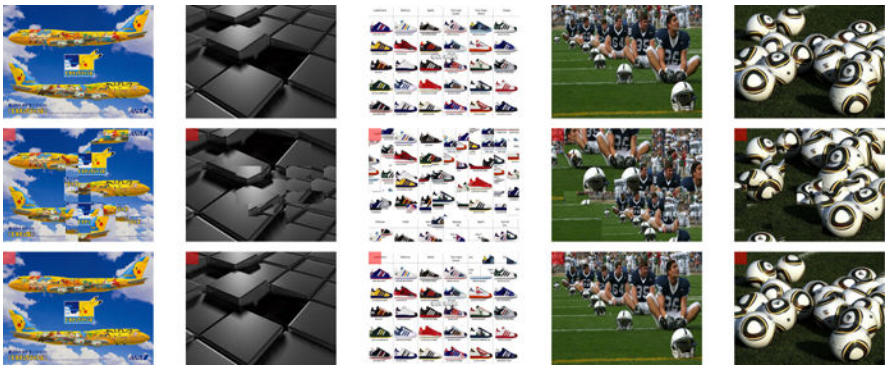

Figure 2.
First row: the original images. Second row: the jigsaw puzzle solutions of [2]. Third row: our solutions.

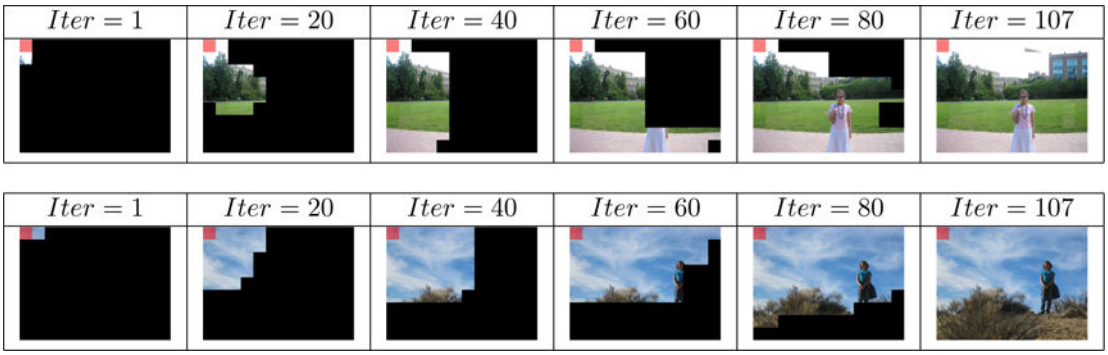

Figure 3.
The reconstructed images of the best particles at different iterations.

Table 1

Experimental results on MIT Dataset.

	[2]	Our algorithm
Direct Comparison	0.2366	0.6921
Cluster Comparison	0.4657	0.7810
Neighbor Comparison	0.6628	0.8620

Table 2

Experimental results on the extended dataset.

	[2]	Our algorithm
Direct Comparison	0.2137	0.7097
Cluster Comparison	0.4500	0.8018
Neighbor Comparison	0.6458	0.8770