# Swapping-Based Partitioned Sampling for better complex density estimation: application to articulated object tracking

Séverine Dubuisson, Christophe Gonzales, Xuan Son Nguyen

# Swapping-Based Partitioned Sampling for better complex density estimation: application to articulated object tracking

Séverine Dubuisson, Christophe Gonzales, Xuan Son Nguyen

Laboratoire d'Informatique de Paris 6 (LIP6/UPMC)
4 place Jussieu, 75005 Paris, FRANCE
firstname.name@lip6.fr

**Abstract.** In this paper, we propose to better estimate high-dimensional distributions by exploiting conditional independences within the Particle Filter (PF) framework. We first exploit Dynamic Bayesian Networks to determine conditionally independent subspaces of the state space, which allows us to independently perform propagations and corrections over smaller spaces. Second, we propose a *swapping* process to transform the weighted particle set provided by the update step of PF into a "new particle set" better focusing on high peaks of the posterior distribution. This new methodology, called *Swapping-Based Partitioned Sampling*, is successfully tested and validated for articulated object tracking.

## 1 Introduction

Dealing with high-dimensional state and observation spaces is a major concern for many research communities. There exist essentially two ways to tackle high-dimensional problems: either reduce the dimension of the state space/search space by approximation or exploit conditional independences naturally arising in the state space to partition the latter into low-dimensional spaces. In this paper, we chose the latter and focus on articulated object tracking. Actually, it is an important computer vision task for a wide variety of applications including gesture recognition, human tracking and event detection. However, tracking articulated structures with accuracy and within a reasonable time is challenging due to the high dimensionality of the state and observation spaces. In the optimal filtering context, the goal of tracking is to estimate a state sequence $\{\mathbf{x}_t\}_{t=1,\dots,T}$ whose evolution is specified by a dynamic equation $\mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_{t-1}, \mathbf{n}_t^{\mathbf{x}})$ given a set of observations. These observations $\{\mathbf{y}_t\}_{t=1,\dots,T}$, are related to the states by $\mathbf{y}_t = \mathbf{h}_t(\mathbf{x}_t, \mathbf{n}_t^{\mathbf{y}})$. Usually, $\mathbf{f}_t$ and $\mathbf{h}_t$ are vector-valued and time-varying transition functions, and $\mathbf{n}_t^{\mathbf{x}}$ and $\mathbf{n}_t^{\mathbf{y}}$ are noise sequences, independent and identically distributed. All these equations are usually considered in a probabilistic way and their computation is decomposed in two main steps. First the prediction of the density function $p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}$ with $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ the prior density related to transition function $\mathbf{f}_t$, and then a correction step $p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ with $p(\mathbf{y}_t|\mathbf{x}_t)$ the likelihood density

related to the measurement function $\mathbf{h}_t$. When functions $\mathbf{f}_t$ and $\mathbf{h}_t$ are linear, or linearizable, and when distributions are Gaussian or mixtures of Gaussians, sequence $\{\mathbf{x}_t\}_{t=1,\dots,T}$ can be computed analytically by Kalman, Extended Kalman or Unscented Kalman Filters [4]. Unfortunately, most vision tracking problems involve nonlinear functions and non-Gaussian distributions. In such cases, tracking methods based on Particle Filters (PF) [4, 6], also called Sequential Monte Carlo Methods (SMC), can be applied under very weak hypotheses: their principle is not to compute the parameters of the distributions, but to approximate these distributions by a set of $N$ weighted samples $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}$, also called *particles*, corresponding to hypothetical state realizations. As optimal filtering approaches do, PF consists of two main steps: (i) a prediction of the object state in the scene (using previous observations), that consists of propagating the set of particles $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}$ according to a *proposal function* $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_t)$, followed by (ii) a correction of this prediction (using a new available observation), that consists of *weighting* the particles w.r.t. a *likelihood function*, so that $w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) \frac{p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_t)}$, with $\sum_{i=1}^{N} w_t^{(i)} = 1$. Particles can then be resampled, so that those with highest weights are duplicated, and those with lowest weights are removed. The estimation of the posterior distribution is then given by $\sum_{i=1}^{N} w_t^{(i)} \delta_{\mathbf{x}_t^{(i)}}(\mathbf{x}_t)$, where $\delta_{\mathbf{x}_t^{(i)}}$ are Dirac masses centered on particles $\mathbf{x}_t^{(i)}$. There exist many models of PF, each having its own advantages. Unfortunately, the computational cost of PF highly depends on the number of dimensions of the state space and, for large state and observation spaces, it may be unrealistically high due to the large number of particles needed to approximate the distributions and to the costs of computing weights $w_t^{(i)}$.

In this paper, we propose to exploit conditional independences in the state space to transform by *swapping* processes the weighted particle set provided by the correction step of PF into a "new particle set" better focusing on high peaks of the posterior distribution. This enables to deal with high-dimensional state spaces by reducing the needed number of particles while increasing the accuracy of the estimation of the probability distribution of the tracked object's state. This paper is organized as follows. Section 2 gives a short overview of the existing approaches that try to solve the high-dimensionality problem by exploiting conditional independences to decompose probabilistic computations. Section 3 recalls the Partitioned Sampling approach and, then, details our approach. Section 4 gives experimental results on challenging synthetic video sequences. Finally, concluding remarks and perspectives are given in Section 5.

## 2 Exploiting conditional independences for tracking

It has been shown in [12] that the number of particles needed to track an object grows exponentially with the dimension of the state space of this object. For problems of articulated object or multiple object tracking, the state space may have very high dimensions, which makes PF unusable for real-time tracking.

Several methods aim at reducing the number of necessary particles by exploiting conditional independences in the state space to divide it into small parts.

Partitioned Sampling (PS) [11] is one of the most popular among these methods. It exploits the fact that, in many problems, both the system dynamics and the likelihood function are decomposable over small subspaces. The key idea, that will be described in Section 3, is then to substitute the application of one PF over the whole state space by a sequence of applications of PF over these small subspaces, thus significantly speeding-up the process. However, for the articulated object tracking purpose, PS suffers from numerous resampling steps that increase noise and decrease the tracking accuracy over time.

The same kind of decomposition is exploited in [9] in the context of a general PF for Dynamic Bayesian Networks (DBN). Here, the proposal distributions of the prediction step is decomposed as the product of the conditional distributions of all nodes of the current time slice in the network. The prediction step then follows the topological order of the nodes of the current time slice of the DBN and uses for each node its conditional probability as the proposal distribution. This allows to integrate the current observations into the proposal distribution.

In [15], the sampling idea of [9] is combined with that of resampling proposed in [11] to create a PF algorithm fitted for DBNs. This algorithm can be seen as a generalization of PS. By following a DBN topological order for sampling and by resampling the particles each time an observed node is processed, particles with low likelihood for one subspace are discarded just after the instantiation of this subspace due to the resampling step, whereas particles with high likelihood are multiplied. This has the same effect as *weighted resampling* in PS.

One of the most recent and promising approach that uses a decomposition technique is the nonparametric Belief Propagation algorithm [17, 8]. It combines the PF framework with the well-known Loopy Belief Propagation algorithm [14] for speeding-up computations (but at the expense of approximations). It has been successfully applied on many problems of high dimensions [16, 2, 7]

Another popular approach is the Rao-Blackwellized Particle Filter for DBN (RBPF) [5]. By using a natural decomposition of the conditional probability, RBPF decomposes the state space into two parts that fulfill the following condition: the conditional distribution of the second part given the first part can be estimated using classical techniques such as Kalman filter. The distribution of the first part is then estimated using PF and the conditional distribution of the second part given the first one is estimated using Kalman filter. As the dimension of the first part is smaller than that of the whole state space, the sampling step of particle filter for the first part needs fewer particles and the variance of the estimation can be reduced. Though RBPF is very efficient for reducing the high dimension of the problem, it can not be applied on all DBNs because the state space cannot always be decomposed into two parts fulfilling the condition.

The framework introduced in [3] is somewhat related to ours. This is a parallel PF for DBNs that uses the same decomposition of the joint probability as a Bayesian Network (BN) to reduce the number of particles required for tracking. The state space is divided into several subspaces that are in some respect

relatively independent. The particles for these subspaces can then be generated independently using different proposal densities. This approach offers a very flexible way of choosing the proposal density for sampling each subspace. However the definition of different subspaces requires the DBN to have a particular independence structure, limiting the generalization of this algorithm. In our paper, we address more general problems where no such independences hold. We focus on PS [11, 12] for its simplicity and generalization potential. In [1], PS was proved to be one of the best algorithm for tracking problems of high dimension. We believe that PS can be improved by better exploiting the independences in DBNs. This idea will be presented in the next section.

## 3 Proposed approach

### 3.1 Partitioned Sampling (PS)

PS is an effective Particle Filter (PF) designed for tracking complex objects with large state space dimensions using only a reduced number of particles. Its key idea is to divide the state space into an appropriate set of partitions and to apply sequentially a PF on each partition, followed by a specific "weighted resampling" ensuring that the sets of particles represent the joint distribution of the whole state space and are focused on its peaks.

Let $g : \mathcal{X} \mapsto \mathbb{R}$ be any strictly positive continuous function, with $\mathcal{X}$ the state space. Given a set of particles $\mathcal{P}_t = \{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ with weights $w_t^{(i)}$, weighted resampling proceeds as follows: let $\rho_t$ be defined as $\rho_t(i) = g(\mathbf{x}_t^{(i)}) / \sum_{j=1}^N g(\mathbf{x}_t^{(j)})$ for $i = 1, \ldots, N$. Select independently indices $k_1, \ldots, k_N$ according to probability $\rho_t$. Finally, construct a new set of particles $\mathcal{P}'_t = \{\mathbf{x'}_t^{(i)}, w'_t{}^{(i)}\}_{i=1}^N$ defined by $\mathbf{x'}_t^{(i)} = \mathbf{x}_t^{(k_i)}$ and $w'_t{}^{(i)} = w_t^{(k_i)} / \rho_t(k_i)$. MacCormick [10] shows that $\mathcal{P}'_t$ represents the same probability distribution as $\mathcal{P}_t$ while focusing on the peaks of $g$.

PS's key idea is to exploit some decomposition of the system dynamics w.r.t. subspaces of the state space in order to apply PF only on those subspaces. This leads to a significant reduction in the number of particles needed for tracking. So, assume that state space $\mathcal{X}$ can be partitioned as $\mathcal{X} = \mathcal{X}^1 \times \cdots \times \mathcal{X}^P$ as well as observation space $\mathcal{Y} = \mathcal{Y}^1 \times \cdots \times \mathcal{Y}^P$. For instance, a system representing a hand could be defined as $\mathcal{X}^{\text{hand}} = \mathcal{X}^{\text{palm}} \times \mathcal{X}^{\text{thumb}} \times \mathcal{X}^{\text{index}} \times \mathcal{X}^{\text{middle}} \times \mathcal{X}^{\text{ring}} \times \mathcal{X}^{\text{little}}$. Assume in addition that the dynamics of the system follows this decomposition, i.e., that:

$$f_t(\mathbf{x}_{t-1}, n_t^{\mathbf{x}}) = f_t^P \circ f_t^{P-1} \circ \cdots \circ f_t^2 \circ f_t^1(\mathbf{x}_{t-1}), \tag{1}$$

where $\circ$ is the usual function composition operator and where each function $f_t^i : \mathcal{X} \mapsto \mathcal{X}$ modifies the particles' states only on subspace $\mathcal{X}^i$ [1].

The PF scheme consists of resampling particles, of propagating them using proposal function $f_t$ and, finally, of updating their weights using the observations at hand. Here, the same result is achieved by substituting the $f_t$ propagation

[1] Note that, in [10], functions $f_t^i$ are more general since they can modify states on $\mathcal{X}^i \times \cdots \times \mathcal{X}^p$. However, in practice, particles are often propagated one $\mathcal{X}^j$ at a time.

step by the sequence of applications of the $f_t^i$ as given in Eq. (1), each one followed by a weighted resampling that produces new particles sets focused on the peaks of a function $g$. To be effective, PS thus needs $g$ to be peaked with the same region as the posterior distribution restricted to $\mathcal{X}^i$. When the likelihood function decomposes as well on subsets $\mathcal{X}^i$, i.e., when:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{i=1}^{P} p^i(\mathbf{y}_t^i|\mathbf{x}_t^i), \tag{2}$$

where $\mathbf{y}_t^i$ and $\mathbf{x}_t^i$ are the projections of $\mathbf{y}_t$ and $\mathbf{x}_t$ on $\mathcal{Y}^i$ and $\mathcal{X}^i$, weighted resampling focusing on the peaks of the posterior distribution on $\mathcal{X}^i$ can be achieved by first multiplying the particles' weights by $p^i(\mathbf{y}_t^i|\mathbf{x}_t^i)$ and, then, by performing a usual resampling. Note that Eq. (2) naturally arises when tracking articulated objects. This leads to the condensation diagram given in Fig. 1, where operations "$*f_t^i$" refer to propagations of particles using proposal function $f_t^i$ as defined above, "$\times p_t^i$" refer to the correction steps where particle weights are multiplied by $p^i(\mathbf{y}_t^i|\mathbf{x}_t^i)$ (see Eq. (2)), and "$\sim$" refer to usual resamplings. MacCormick and Isard show that this diagram produces mathematically correct results [12].

### 3.2 Swapping-Based Partition Sampling (SBPS)

The hypotheses used by PS can best be explained on a dynamic Bayesian network (DBN) representing the conditional independences between random variables of states and observations [13]. Assume for instance that an object to be tracked is composed of 3 parts: a torso, a left arm and a right arm. Let $\mathbf{x}_t^1, \mathbf{x}_t^2, \mathbf{x}_t^3$ represent these parts respectively. Then, the probabilistic dependences between these variables and their observations $\mathbf{y}_t^1, \mathbf{y}_t^2, \mathbf{y}_t^3$, can be represented by the DBN of Fig. 2. In this figure, Eq. (2) implicitly holds because, conditionally to states $\mathbf{x}_t^i$, observations $\mathbf{y}_t^i$ are independent of the other random variables. In addition, the probabilistic dependences between substates $\mathbf{x}_t^1, \mathbf{x}_t^2, \mathbf{x}_t^3$ suggest that the dynamics of the system is decomposable on $\mathcal{X}^1 \times \mathcal{X}^2 \times \mathcal{X}^3$. As a consequence, the condensation diagram of Fig. 1 can be exploited to track the object.

Through the $d$-separation criterion [14], DBNs offer a strong framework for analyzing probabilistic dependences among sets of random variables. By this criterion, it can be remarked that, on Fig. 2, $\mathbf{x}_t^2$ and $\mathbf{x}_t^3$ are independent conditionally to $(\mathbf{x}_t^1, \mathbf{x}_{t-1}^2)$ and $(\mathbf{x}_t^1, \mathbf{x}_{t-1}^3)$ respectively. As a consequence, for each particle, PS's propagations/corrections over subspaces $\mathcal{X}^2$ and $\mathcal{X}^3$ can be performed independently since, in this case, $\mathbf{x}_t^1$ and $\mathbf{x}_{t-1}$ are known and fixed. This suggests the new condensation diagram of Fig. 3.

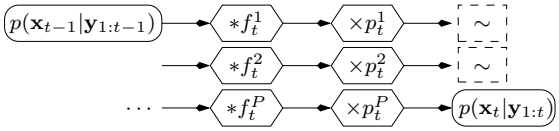


**Fig. 1.** Partitioned Sampling condensation diagram.

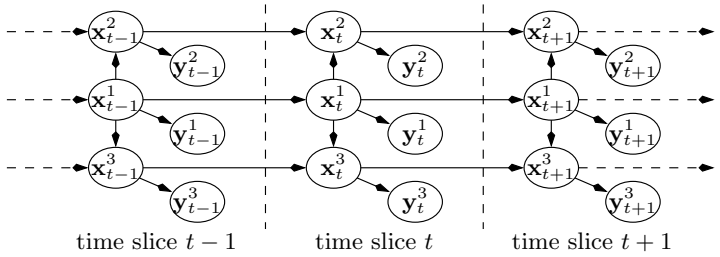


**Fig. 2.** A Dynamic Bayesian network for body tracking.

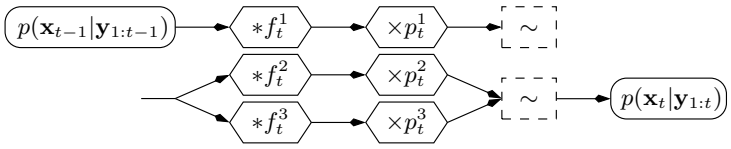


**Fig. 3.** Condensation diagram exploiting conditional independences.

**Proposition 1.** *The set of particles resulting from the Particle Filter of Fig. 3 represents probability distribution* $p(\mathbf{x}_t|\mathbf{y}_{1:t})$.

*Proof.* Propagations performed in parallel concern subspaces that are probabilistically independent. So, they produce the same result as if they were performed sequentially. Hence, the only difference between PS and the PF of Fig. 3 is that fewer resamplings are performed. But resamplings do not change the probability distributions represented by the particle sets. Hence the result. □

There are two major differences between PS and the PF of Fig. 3: the latter performs fewer resamplings, thus introducing less noise in the particle set and, more importantly, it enables to produce better fitted particles by swapping their subparts. Actually, consider again our body tracking example and assume that we generated the 3 particles $\mathbf{x}_t^{(i)}$ of Fig. 4.a where $\mathcal{X}^1$ is the middle part of the object and $\mathcal{X}^2$ and $\mathcal{X}^3$ are its left and right parts respectively, and where the shaded areas represent the object's true state. According to the DBN of Fig. 2, for fixed values of $\mathbf{x}^1_{1:t}$, the sets of left and right parts of the particles represent $p(\mathbf{x}^2_t, y^2_{1:t}|\mathbf{x}^1_{1:t})$ and $p(\mathbf{x}^3_t, y^3_{1:t}|\mathbf{x}^1_{1:t})$ respectively (summing out variables $\mathbf{x}^2_j, \mathbf{x}^3_j$ from the DBN). Hence, after permuting the values of the particles on $\mathcal{X}^2$ (resp. $\mathcal{X}^3$) for a fixed value of $\mathbf{x}^1_{1:t}$, distribution $p(\mathbf{x}^2_t, y^2_{1:t}|\mathbf{x}^1_{1:t})$ (resp. $p(\mathbf{x}^3_t, y^3_{1:t}|\mathbf{x}^1_{1:t})$) remains unchanged. A fortiori, this does not affect the representation of the joint posterior distribution $\int p(\mathbf{x}^1_{1:t}, y^1_{1:t}) p(\mathbf{x}^2_t, y^2_{1:t}|\mathbf{x}^1_{1:t}) p(\mathbf{x}^3_t, y^3_{1:t}|\mathbf{x}^1_{1:t}) d\mathbf{x}^1_{1:t-1} = p(\mathbf{x}_t, \mathbf{y}_{1:t})$. On Fig. 4.a, particles $\mathbf{x}_t^{(1)}$ and $\mathbf{x}_t^{(3)}$ have the same state on $\mathcal{X}^1$. Thus their right parts can be permuted, resulting in the new particle set of Fig. 4.b. Remark that we substituted 2 particles, $\mathbf{x}_t^{(1)}$ and $\mathbf{x}_t^{(3)}$, which had low weights due to their bad estimation of the object's right or left part states, by one particle $\mathbf{x}_t'^{(1)}$ with a high weight and another one $\mathbf{x}_t'^{(3)}$ with a very low weight. After

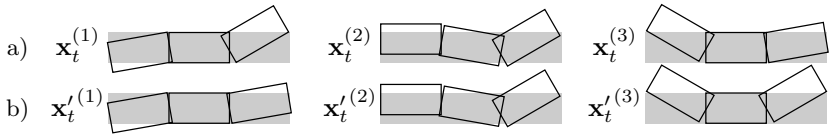


**Fig. 4.** The particle swapping scheme: a) before swapping; b) after swapping

resampling, the later will most probably be discarded and, therefore, swapping will have focused particles on the peaks of the posterior distribution. Note however that not all permutations are allowed: for instance, none can involve particle $\mathbf{x}_t^{(2)}$ because its center part differs from that of the other particles.

Let us now formulate SBPS. Assume again that state space $\mathcal{X}$ is decomposed as $\mathcal{X} = \mathcal{X}^1 \times \cdots \times \mathcal{X}^P$ and that the probabilistic dependences between all random variables $\mathbf{x}_t^i$ and $\mathbf{y}_t^i$, $i = 1, \ldots, P$, are represented by a DBN $\mathcal{G}$. Let $\{P_1, \ldots, P_K\}$ be a partition of $\{1, \ldots, P\}$ such that, for all $i$, $\{\mathbf{x}_t^j\}_{j \in P_i}$ are mutually independent conditionally to $(\cup_{h=1}^{i-1} \cup_{k \in P_h} \mathbf{x}_t^k) \cup \mathbf{x}_{t-1}$. Such sets are easily identified by $d$-separation on DBN $\mathcal{G}$ [14]. By definition, after processing $P_1, \ldots, P_{i-1}$, all the variables of each set $P_i$ can be processed independently. Denote the elements of $P_i$ by $\{i_{P_i}^1, \ldots, i_{P_i}^{k_i}\}$. Then, the SBPS algorithm can be described by the condensation diagram of Fig. 5, where operations "$\rightleftharpoons^{P_i}$" refer to the particle subpart swappings briefly described previously. Remark that, after the resampling operation of part $P_i$, the high weighted particles will be duplicated, which will enable swapping when processing next part $P_{i+1}$. Swappings need however to be further formalized. Let $\mathbf{pa}(X_t^i)$ denote the parents of node $X_t^i$ in $\mathcal{G}$ in time slice $t$ and let $\mathrm{Link}(X_t^i)$ be the set of nodes in all time slices such that there exists an undirected path in $\mathcal{G}$ linking them to $X_t^i$ while not passing through any node in $(\mathbf{pa}(X_t^i))_{1:t}$. Assume now that SBPS was executed up to (but not including) operation $\rightleftharpoons^{P_k}$. Let $r \in P_k$ be some part of the object. Then, for each value of $(\mathbf{pa}(\mathbf{x}_t^r))_{1:t}$, substates $\mathbf{x}_t^r$ of the particles represent $p(\mathbf{x}_t^r, \mathbf{y}_{1:t}^r | \mathbf{pa}(\mathbf{x}_t^r)_{1:t})$. Thus, permuting substates $\mathbf{x}_t^r$ among particles with the same value of $(\mathbf{pa}(\mathbf{x}_t^r))_{1:t}$ does not change this distribution. However, if $\mathbf{x}_t^s$ is a child of $\mathbf{x}_t^r$, then not permuting similarly substates $\mathbf{x}_t^s$ of these particles changes $p(\mathbf{x}_t^s, \mathbf{y}_{1:t}^s | \mathbf{pa}(\mathbf{x}_t^s)_{1:t})$, hence resulting in incorrect computations. More generally, it is easily shown that all the values of the substates in $\mathrm{Link}(\mathbf{x}_t^r)$ (and only those values) need be permuted to ensure that no conditional probability is affected by the swapping. By conditional independences w.r.t. $\mathbf{pa}(\mathbf{x}_t^r)_{1:t}$, the product of all these distributions is the joint probability. So, operation $\rightleftharpoons^{P_k}$ refers to permuting some values of $\mathrm{Link}(\mathbf{x}_t^r)$ for some $r \in P_k$ and among particles having the same substate $\mathbf{pa}(\mathbf{x}_t^r)_{1:t}$. In practice, whenever $\mathbf{pa}(\mathbf{x}_t^r)_t$ is identical for two particles, the continuous nature of the state space make it highly probable that one particle is a copy of the other due to resampling. Hence, their $\mathbf{pa}(\mathbf{x}_t^r)_{1:t}$ values should also be identical. So, our implementation approximates the posterior distributions by performing swapping for fixed values of $\mathbf{pa}(\mathbf{x}_t^r)_t$ instead of $\mathbf{pa}(\mathbf{x}_t^r)_{1:t}$.

**Proposition 2.** *The set of particles resulting from SBPS represents* $p(\mathbf{x}_t | \mathbf{y}_{1:t})$.

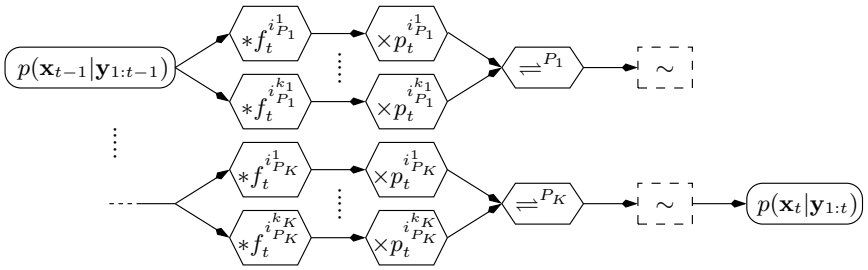


**Fig. 5.** Swapping-based Partitioned Sampling condensation diagram.

*Proof.* If we do not swap particles, the proof follows from Proposition 1. If some swapping occurs, say on substate $\mathbf{x}_t^r$, then, as mentioned previously, distribution $p(\mathbf{x}_t^r, \mathbf{y}_{1:t}^r|\mathbf{pa}(\mathbf{x}_t^r)_{1:t})$ remains unchanged. By definition, swapping also occurs on the neighbors $\mathbf{x}_t^s$ of $\mathbf{x}_t^r$ and their neighbors, so that the conditional distribution of $\mathbf{x}_t^s$ remains unchanged. By induction on neighborhoods in $\mathcal{G}$, no conditional distribution is ever affected by swapping and the result follows. □

Finally, let us show how $\rightleftharpoons^{P_k}$ can determine attractive swappings, i.e., how high-peaked regions can be discovered. For simplicity, we will only describe swapping on the example of Fig. 6, but the principle can easily be generalized. Assume that $\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \mathcal{X}^3 \times \mathcal{X}^4$, where parts are defined from left to right. In addition, assume that $P_1 = \{2\}$, $P_2 = \{3\}$ and $P_3 = \{1, 4\}$, i.e., $P_3$ corresponds to the extremal sides of the object. Let us describe operation $\rightleftharpoons^{P_3}$. Just before propagating part $P_3$, SBPS has constructed particles with equal weights (due to its resamplings). In the rectangles of parts $\mathcal{X}^2$ and $\mathcal{X}^3$, identical letters indicate identical substate values (e.g., particles 1 and 2 have the same value on $\mathcal{X}^2$). Just before executing $\rightleftharpoons^{P_3}$, particles have been propagated on the extremal sides of the object (resulting in Fig. 6.a) and operations $\times p_t^1$ and $\times p_t^4$ have updated their weights. These weights (unormalized for clarity reasons) are displayed inside the rectangles corresponding to $\mathcal{X}^1$ and $\mathcal{X}^4$ and the total weights of the particles are shown on their right side (According to the DBN, these weights are equal to $p^1(\mathbf{y}_t|\mathbf{x}_t^{1,(i)}) \times p^4(\mathbf{y}_t|\mathbf{x}_t^{4,(i)})$). To find the best swappings, we exploit the data structure given in Fig. 6.b: the circle nodes correspond to the values of the particles on $\mathbf{pa}(\mathbf{x}_t^1)_t$ and $\mathbf{pa}(\mathbf{x}_t^4)_t$. The values within rectangles correspond to the set of values and weights of the particles on $\mathbf{x}_t^1$ and $\mathbf{x}_t^4$ for each value of their parents $\mathbf{pa}(\mathbf{x}_t^1)_t$ and $\mathbf{pa}(\mathbf{x}_t^4)_t$. Finally, there exists an edge between two circles if and only if there exists at least one particle with the values of both circles. For instance, edge $(A, D)$ is induced by particle 2. By definition, all the rectangle values that are attached to a given circle (e.g., 5 and 2) can be swapped since they have the same parent values in the DBN. As there exists an edge between two circles if and only if there exists a particle with both circle values, we can conclude that any value attached to one such circle can be combined by swapping with any value attached to the other circle. For instance, 5, which is attached to $A$, can be combined with 1 (attached to $C$) or 4 or 2 (attached to $D$). To get the best

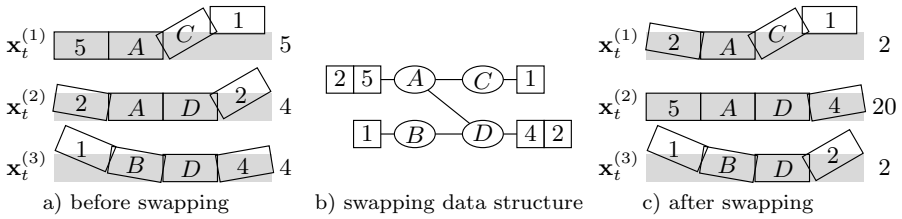


**Fig. 6.** Details on the swapping process.

particle, we shall only consider combinations with the highest rectangle values, hence, for the pair $(A, D)$, we shall only consider combining 5 with 4. The same shall be done for nodes attached to $B$ and the first new particle constructed is the one with the highest product (here $5 \times 4$ from pair $(A, D)$). Once this combination has been used, remove values 5 and 4 from the graph and iterate. When no more rectangle is attached to a circle, this one is removed from the graph. This process is fast and very effective to produce high-weight particles.

## 4 Experimental results

We have tested our method and compared it to PS on synthetic video sequences because we wanted to highlight its interest in terms of dimensionality reduction and tracking accuracy without having to take into account specific properties of images (noise, *etc.*). Moreover, it is possible to simulate specific motions and then to test and compare with accuracy our method with PS. For that, we have generated our own synthetic video sequences, each one containing 300 frames, showing two different kinds of articulated objects: chains or squids. A chain is the concatenation of $P$ colored rectangles ($P$ is also called the length of the object), and a squid is made of two chains crossing in their middle part: in a sense, a chain can be defined by a central rectangle, and two tentacles starting from this central part, while a squid has four tentacles starting from its central part. Chains and squids are translating and distorting over time, see examples of squids in Fig. 10 and 11, and of a chain in Fig. 8. The goal, here, is to observe the capacity of PS and SBPS to deal with articulated objects composed of a varying number of parts and subject to weak or strong motions.

The tracked articulated object is modeled by a set of $P$ rectangles whose corners are labeled $C_1, \ldots, C_4$. The state space contains parameters describing each rectangle, and is defined by $\mathbf{x}_t = \{\mathbf{x}_t^1, \mathbf{x}_t^2, \ldots, \mathbf{x}_t^P\}$, with $\mathbf{x}_t^i = \{x_t^i, y_t^i, \theta_t^i\}$, where $(x_t^i, y_t^i)$ denotes the coordinates of the center of the $i^{\text{th}}$ rectangle, and $\theta_t^i$ is its orientation, $i = 1, \ldots, P$. A particle $\mathbf{x}_t^{(j)} = \{\mathbf{x}_t^{1,(j)}, \mathbf{x}_t^{2,(j)}, \ldots, \mathbf{x}_t^{P,(j)}\}$, $j = 1, \ldots, N$, is thus a possible configuration of an articulated object. In the first frame, particles are uniformly generated around the object. During the prediction step, particles are propagated following a random walk whose variance has been manually chosen. The weights of the particles are then computed using the current observation (i.e. the current frame). Finally, the particle's weights

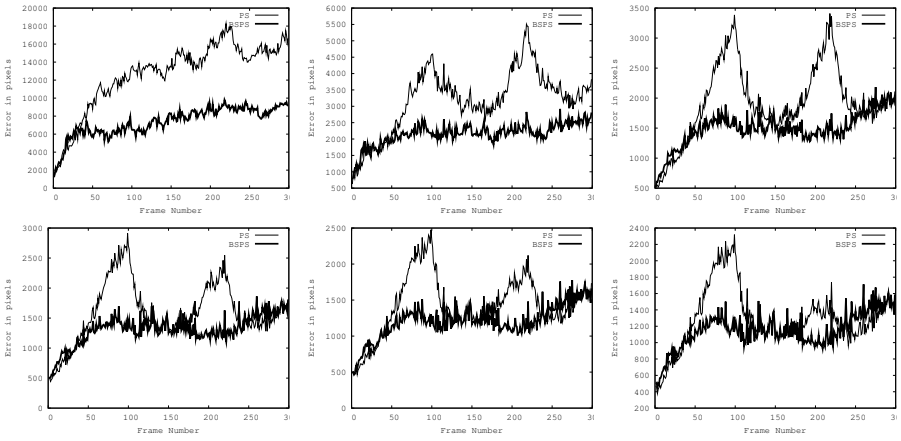


**Fig. 7.** Tracking errors for PS and SBPS approaches for a chain object of length $P = 11$ with, from left to right and from top to bottom, $N = 5, 10, 20, 30, 40, 50$ particles. Motions in frames $[50, 100]$ and $[150, 200]$ are stronger than in the other frames.

are given by $w_t^{(j)} = w_{t-1}^{(j)} p(\mathbf{y}_t|\mathbf{x}_t^{(j)}) \propto w_{t-1}^{(j)} e^{-\lambda d^2}$, with $d$ the Bhattacharyya distance between the histograms of the target (prior) and the reference (previously estimated) regions. Parameter $\lambda$ was set to 50 in all our tests.

In both approaches, the articulated object's joint distribution is estimated by starting from its center part. PS then propagates and corrects particles part after part to derive a global estimation of the object. SBPS considers tentacles of objects as totally independent, and thus propagates, swaps and corrects simultaneously in all tentacles. PS and SBPS are compared by measuring the tracking error as the distance between the ground truth and the estimated articulated object at each instant. This distance is given by the sum of the Euclidean distances between each corner $C_i$ of each estimated rectangle and its corresponding corner $C_i$ of the same rectangle in the ground truth. All the results presented in this paper correspond to a mean over 100 runs.

Our first test concerns the tracking of a chain object of length $P = 11$. To test the stability of our approach, we have generated video sequences in which motions during two specific temporal intervals (frames $[50, 100]$ and $[150, 200]$) are strong. Comparative results of tracking errors of PS and SBPS are reported in Fig. 7, for different numbers $N$ of particles. We can see on these graphs that SBPS always outperforms PS, which shows its stability especially when the motion becomes strong: the tracking error drastically increases for PS whereas that of SBPS is relatively stable. Visual results of tracking are shown on Fig. 8 with $N = 30$ particles for this object: the estimation of the articulated object is represented by the concatenated white rectangles. Over 100 runs, the tracking error resulting from our approach was reduced by 19% as compared to PS.

Table 1 summarizes all the tests performed on different video sequences showing chains of length $P = 3, 5, 7, 9, 11$ or squid of length $P = 5, 9, 13, 17, 21$,

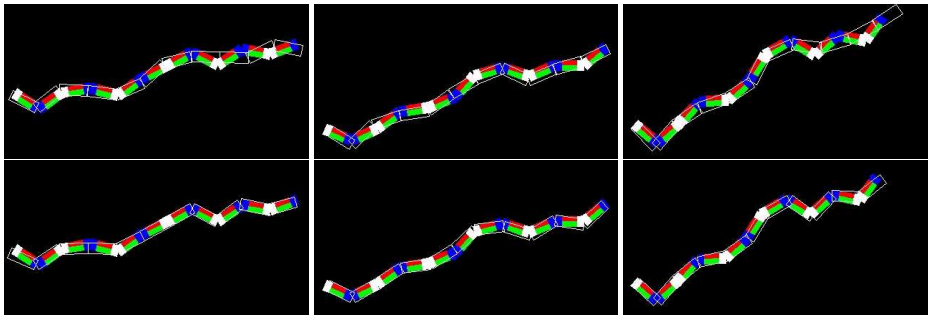

**Fig. 8.** Zooms on tracking results obtained for PS (top line) and SBPS (bottom line) on frames 100, 150 and 200, for a chain object of length $P = 11$ with $N = 30$ particles. White articulated objects represent the mean estimations of the articulated object. Mean tracking error: 1670 pixels for PS, and 1286 pixels for SBPS.

for values $N = 5, 10, 20, 30, 40, 50$. Tracking errors (in pixels) over all the sequences are reported for both approaches, and the percentage of reduction of tracking error obtained with our approach, denoted by $\searrow$ %, is computed as $\left(1 - \frac{SBPS}{PS}\right) \times 100$. As another example, we also reported the tracking errors for a squid object of size $P = 17$ in Fig. 9. We can see that our approach always decreases the tracking error. This is especially noticeable for high values of $P$. Even for small objects ($P = 3$) and a large number of particle ($N = 50$), which should be highly sufficient to provide good tracking results, SBPS outperforms PS, decreasing the error by 7%. The visual tracking results for a squid object of size $P = 5$ using $N = 5$ particles are given in Fig. 10: mean tracking error is 1454 pixels for PS , and 503 pixels for SBPS. All these results show why exploiting both independence between the different object's parts and subpart swapping is highly efficient: "much better" particles are constructed which, in turn, allows to better estimate the joint probability of the articulated object. This advantage is illustrated by Fig. 11 for a squid object of length $P = 13$ (frames 50, 100, 150 and 200): we have voluntarily used a small number of particle for tracking ($N = 5$) and have drawn into the frames the "best" particle, i.e. the one with the highest weight. If we compare PS (top line) and SBPS (bottom line), we can see the benefit of swapping: unlike the best particle of SBPS, that of PS totally misses the articulated object.

Finally, Table 2 reports the mean computation times (in seconds), over 100 runs, for tracking two different objects: a chain of length $P = 9$ and a squid of length $P = 17$, depending on the number $N$ of particles. Of course, we can see that PS is faster than SBPS, but previous tests show that SBPS requires fewer particles to provide a tracking as good as PS. For instance, for a chain of length $P = 9$, using SBPS and $N = 20$, in 2.31 seconds, we get similar tracking results than those with PS using $N = 30$ (in 3.22 seconds). Similarly, for a squid of length $P = 17$, using SBPS and $N = 20$, in 4.11 seconds, we get similar tracking results than those with PS using $N = 40$ (in 8.16 seconds).

**Table 1.** Comparison of tracking mean errors (in pixels) over all the sequences obtained by PS and SBPS depending on the object (chain or squid), its length $P$, and the number $N$ of particles. $\searrow$ %= $\left(1 - \frac{SBPS}{PS}\right) \times 100$ is the error reduction percentage using SBPS.

| | | Chain | | | | | Squid | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $P=3$ | $P=5$ | $P=7$ | $P=9$ | $P=11$ | $P=5$ | $P=9$ | $P=13$ | $P=17$ | $P=21$ |
| $N=5$ | PS | 514 | 1565 | 3440 | 8546 | 12666 | 1999 | 18473 | 37710 | 66659 | 77864 |
| | SBPS | 469 | 1480 | 1706 | 6638 | 7374 | 812 | 6408 | 9125 | 12397 | 13075 |
| | $\searrow$ % | **-9%** | **-6%** | **-50%** | **-33%** | **-42%** | **-60%** | **-76%** | **-76%** | **-82%** | **-84%** |
| $N=10$ | PS | 187 | 315 | 1302 | 1528 | 3199 | 289 | 894 | 1862 | 2339 | 7786 |
| | SBPS | 167 | 293 | 1044 | 1215 | 2161 | 193 | 627 | 746 | 1407 | 2225 |
| | $\searrow$ % | **-11%** | **-7%** | **-20%** | **-21%** | **-33%** | **-34%** | **-30%** | **-60%** | **-40%** | **-72%** |
| $N=20$ | PS | 136 | 193 | 949 | 1529 | 1944 | 153 | 596 | 519 | 1046 | 2610 |
| | SBPS | 125 | 185 | 813 | 1192 | 1495 | 114 | 428 | 405 | 696 | 1374 |
| | $\searrow$ % | **-9%** | **-5%** | **-15%** | **-23%** | **-24%** | **-26%** | **-29%** | **-22%** | **-34%** | **-48%** |
| $N=30$ | PS | 123 | 164 | 819 | 1313 | 1606 | 120 | 510 | 404 | 772 | 1919 |
| | SBPS | 112 | 160 | 706 | 1069 | 1309 | 97 | 377 | 327 | 527 | 1127 |
| | $\searrow$ % | **-9%** | **-3%** | **-14%** | **-19%** | **-19%** | **-20%** | **-27%** | **-20%** | **-32%** | **-42%** |
| $N=40$ | PS | 115 | 159 | 768 | 1199 | 1440 | 108 | 467 | 349 | 666 | 1615 |
| | SBPS | 108 | 147 | 671 | 997 | 1211 | 91 | 351 | 287 | 460 | 1016 |
| | $\searrow$ % | **-7%** | **-8%** | **-13%** | **-17%** | **-16%** | **-16%** | **-25%** | **-18%** | **-31%** | **-38%** |
| $N=50$ | PS | 112 | 141 | 735 | 1109 | 1306 | 102 | 426 | 317 | 592 | 1534 |
| | SBPS | 105 | 138 | 648 | 956 | 1151 | 88 | 337 | 265 | 428 | 943 |
| | $\searrow$ % | **-7%** | **-3%** | **-12%** | **-14%** | **-12%** | **-14%** | **-21%** | **-17%** | **-28%** | **-39%** |

## 5 Conclusion

We have introduced a new framework, *Swapping-Based Partitioned Sampling*, exploiting conditional independences to simultaneously propagate, correct and swap particles in independent subspaces. As a result, the particle sets produced are more concentrated on high peaks of the posterior distribution than in the classical Partition Sampling. Thus, our estimations of the probability densities of the tracked object are more accurate. Empirical tests have shown that SBPS always outperforms PS, especially in cases where the object motion is strong and when the dimension of the state space is high (i.e., the number of parts is large). There still remains to validate our approach on real video sequences.

## References


1. Bandouch, J., Engstler, F., Beetz, M.: Evaluation of Hierarchical Sampling Strategies in 3D Human Pose Estimation. In: BMVC. pp. 925–934 (2008)
2. Bernier, O., Cheung-Mon-Chan, P., Bouguet, A.: Fast nonparametric belief propagation for real-time stereo articulated body tracking. Computer Vision and Image Understanding 113, 29–47 (2009)
3. Besada-Portas, E., Plis, S.M., Cruz, J.M., Lane, T.: Parallel subspace sampling for particle filtering in dynamic Bayesian networks. In: ECML PKDD. pp. 131–146 (2009)
4. Chen, Z.: Bayesian filtering: from Kalman filters to particle filters, and beyond (2003)

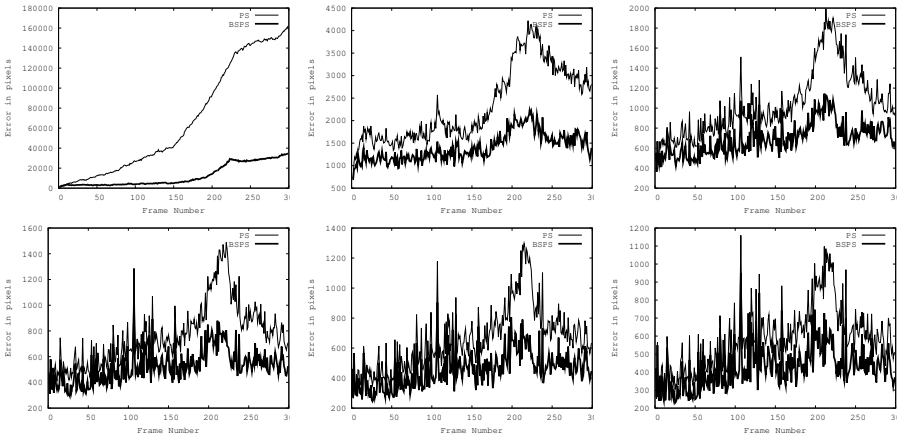


**Fig. 9.** Tracking errors for PS and SBPS approaches for a squid object of length $P = 17$ with, from left to right and from top to bottom, $N = 5, 10, 20, 30, 40, 50$ particles.


5. Doucet, A., de Freitas, N., Murphy, K.P., Russell, S.J.: Rao-Blackwellised particle filtering for dynamic Bayesian networks. In: UAI. pp. 176–183 (2000)
6. Gordon, N., Salmond, D.J., Smith, A.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proceedings of Radar and Signal Processing 140(2), 107–113 (1993)
7. Ihler, A., Fisher, J., Iii, J.W.F., Willsky, A., Moses, R.: Nonparametric belief propagation for self-calibration in sensor networks. In: ISIPSN. pp. 225–233 (2004)
8. Isard, M.: PAMPAS: real-valued graphical models for computer vision. In: CVPR. pp. 613–620 (2003)
9. Kanazawa, K., Koller, D., Russell, S.: Stochastic simulation algorithms for dynamic probabilistic networks. In: UAI. pp. 346–35 (1995)
10. MacCormick, J.: Probabilistic modelling and stochastic algorithms for visual localisation and tracking. Ph.D. thesis, Oxford University (2000)
11. MacCormick, J., Blake, A.: A probabilistic exclusion principle for tracking multiple objects. In: ICCV. pp. 572–587 (1999)
12. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: ECCV. pp. 3–19 (2000)
13. Murphy, K.: Dynamic Bayesian Networks: Representation, Inference and Learning. Ph.D. thesis, UC Berkeley, Computer Science Division (2002)
14. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufman Publishers (1988)
15. Rose, C., Saboune, J., Charpillet, F.: Reducing particle filtering complexity for 3D motion capture using dynamic Bayesian networks. AAAI pp. 1396–1401 (2008)
16. Sigal, L., Isard, M., Sigelman, B.H., Black, M.J.: Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In: NIPS. pp. 1539–1546 (2003)
17. Sudderth, E.B., Ihler, A.T., Isard, M., Freeman, W.T., Willsky, A.S.: Nonparametric belief propagation. Communications of ACM 53, 95–103 (2010)

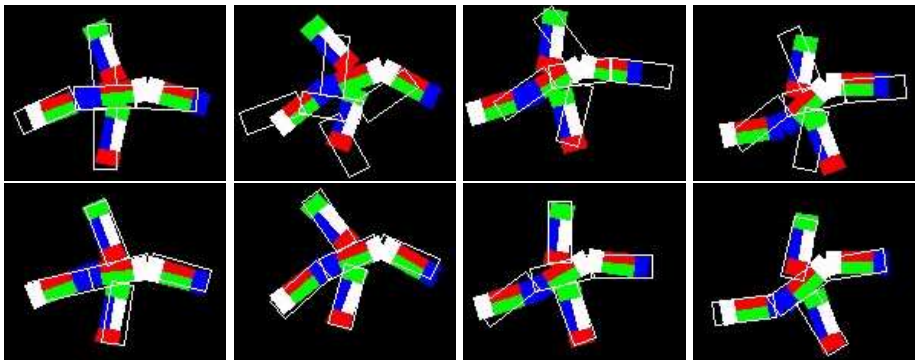

**Fig. 10.** Zooms on tracking results obtained for PS (top line) and SBPS (bottom line) on frames 50, 100, 200 and 250, for a squid of length $P = 5$ with $N = 5$ particles. White articulated objects represent the mean estimations of the articulated object. Mean tracking error: 1454 pixels for PS, and 403 pixels for SBPS.

**Table 2.** Computation times (in sec.) of PS and SBPS for the tracking of a chain and of a squid of lengths $P = 9$ and $P = 17$ respectively, for different values of $N$.

| | | $N = 5$ | $N = 10$ | $N = 20$ | $N = 30$ | $N = 40$ | $N = 50$ |
|---|---|---|---|---|---|---|---|
| Chain: $P = 9$ | PS | 0.70 | 1.21 | 2.18 | 3.22 | 4.2 | 5.15 |
| | SBPS | 0.72 | 1.30 | 2.31 | 3.45 | 4.43 | 5.41 |
| Squid: $P = 17$ | PS | 1.18 | 2.12 | 4.01 | 6.02 | 8.16 | 10.28 |
| | SBPS | 1.20 | 2.15 | 4.11 | 6.21 | 8.37 | 10.69 |

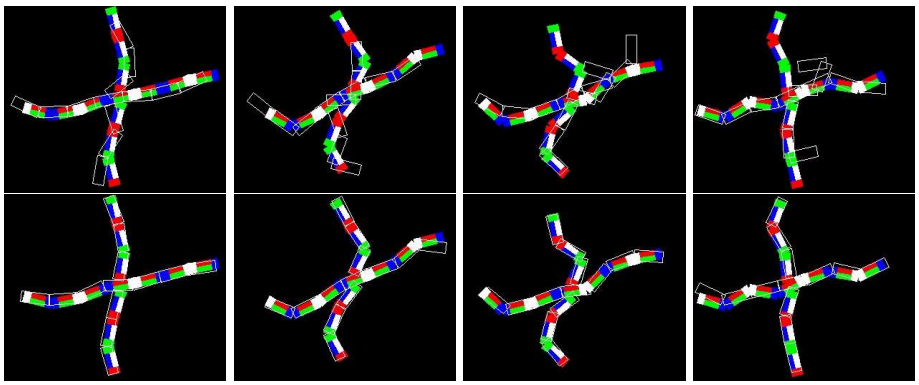

**Fig. 11.** Zooms on the best particles (i.e. with the highest weight) of PS (top line) and SBPS (bottom line) approaches, drawn in white for a squid object of length $P = 13$, with $N = 5$. From left to right, frames 50, 100, 150 and 200.