Efficient Monte Carlo sampler for detecting parametric objects in large scenes

Yannick Verdie, Florent Lafarge

Efficient Monte Carlo sampler for detecting parametric objects in large scenes

Yannick Verdié and Florent Lafarge

INRIA Sophia Antipolis, France
{yannick.verdie,florent.lafarge}@inria.fr

Abstract. Point processes have demonstrated efficiency and competitiveness when addressing object recognition problems in vision. However, simulating these mathematical models is a difficult task, especially on large scenes. Existing samplers suffer from average performances in terms of computation time and stability. We propose a new sampling procedure based on a Monte Carlo formalism. Our algorithm exploits Markovian properties of point processes to perform the sampling in parallel. This procedure is embedded into a data-driven mechanism such that the points are non-uniformly distributed in the scene. The performances of the sampler are analyzed through a set of experiments on various object recognition problems from large scenes, and through comparisons to the existing algorithms.

1 Introduction

Markov point processes constitute an object-oriented extension of traditional Markov Random Fields (MRF). Whereas MRFs address labeling problems on static graphs, Markov point processes can tackle object recognition problems by directly manipulating parametric entities on dynamic graphs. These probabilistic models introduced by Baddeley *et al.* [1] exploit random variables whose realizations are configurations of parametric objects, each object being assigned to a point positioned in the scene. The number of objects is itself a random variable, and thus must not be estimated or specified by an user. Another strength of Markov point processes is their ability to take into account complex spatial interactions between the objects and to impose global regularization constraints.

1.1 Point processes for vision problems

Many recent works exploiting point processes have been proposed to address a large variety of computer vision problems [2–9]. The growing interest in these probabilistic models is motivated by the need to manipulate parametric objects interacting in scenes. Parametric objects can be defined in discrete and/or continuous domains. They usually correspond to geometric entities, *e.g.* segments, rectangles, circles or planes, but can more generally be any type of multi-dimensional function. A point process requires the formulation of a probability

density in order to measure the quality of a configuration of objects. The density is typically defined as a combination of a term assessing the consistency of objects to the data, and a term taking into account spatial interactions between objects in a Markovian context. The optimal configuration maximizing this density is usually searched for by a Monte Carlo based sampler capable of exploring the whole configuration space, in most cases a Markov Chain Monte Carlo (MCMC) algorithm [10, 11].
Descombes *et al.* [2] propose a point process for counting populations from aerial images, each entity being captured by an ellipse. Ge *et al.* [3] present a point process for a similar application, but dedicated to crowd detection from ground-based photos, for which objects are defined as a set of body shape templates learned from training data. Multi-view images are used by Utasi *et al.* [9] to detect people by a point process in 3D where the objects are specified by cylinders. Sun *et al.* [8] and Lacoste *et al.* [7] propose point processes for extracting line networks from images by taking into account spatial interactions between lines to favor the object connexion. Lafarge *et al.* [4] present a general model for extracting different types of geometric features from images, including line, rectangles or disks. A mixture of object interactions are also considered such that the process can address different problems ranging from population counting to line network extraction, through to texture recognition. Lieshout [5] develops a point process for tracking rectangular objects from video. A mono-dimensional point process is proposed by Mallet *et al.* [6] for modeling 1D-signals by mixtures of parametric functions while imposing physical constraints between the modes.

1.2 Motivations

The results obtained by these point processes are particularly convincing and competitive, but the performances remain limited in terms of computation time and convergence stability, especially on large scenes. These drawbacks explain why industry has been reluctant until now to integrate these mathematical models in their products. Indeed, the point processes presented in Section 1.1 emphasize complex model formulations by proposing objects parametrically sophisticated [3, 4], advanced techniques to fit objects to the data [9], and non-trivial spatial interactions between objects [6, 8]. However, these works have only slightly addressed the optimization issues from such complex models.
Jump-Diffusion algorithms [4, 12, 13] have been designed to speed-up the MCMC sampling by inserting diffusion dynamics for the exploration of the continuous subspaces. These samplers are unfortunately restricted to specific density forms. Data considerations have also been used to drive the sampling with more efficiency [14] for image segmentation problems. Some works have also proposed parallelization procedures by using multiple chains simultaneously [15] or decomposition schemes in configurations spaces of fixed dimension [16, 17]. However they are limited by border effects, and are not designed to perform on large scenes. In addition, the existing decomposition schemes cannot be used for configuration spaces of variable dimension. A mechanism based on multiple creation and destruction of objects has been also developed for addressing population

counting problems [2, 9]. Nevertheless object creations require the discretization of the point coordinates which induces a significant loss of accuracy.
These alternative versions of the conventional MCMC sampler globally allow the improvement of optimization performances in specific contexts.That said, the gains in terms of computation time remain weak and are usually realized at the expense of solution stability. Finding a fast efficient sampler for general Markov point processes clearly represents a challenging problem.

1.3 Contributions

We present solutions to address this problem and to drastically reduce computation times while guarantying quality and stability of the solution. Our algorithm presents several important contributions to the field.
Sampling in parallel - Contrary to the conventional MCMC sampler which makes the solution evolve by successive perturbations, our algorithm can perform a large number of perturbations simultaneously using a unique chain. The Markovian property of point processes is exploited to make the global sampling problem spatially independent in a neighborhood.
Non uniform point distributions - Point processes mainly use uniform point distributions which are computationally easy to simulate, but make the sampling extremely slow. We propose an efficient mechanism allowing the modifications, creations or removals of objects by taking into account information on the observed scenes. Contrary to the data-driven solutions proposed by [3] and [14], our non-uniform distribution is not built directly from image likelihood, but is created via space-partitioning trees to ensure the sampling parallelization.
Efficient GPU implementation - We propose an implementation on GPU which significantly reduces computing times with respect to existing algorithms, while increasing stability and improving the quality of the obtained solution.
3D object detection model from point cloud - To evaluate the algorithm in 3D, we propose an original 3D point process to detect parametric objects from point clouds. This model is applied to tree recognition from laser scans of large urban and natural environments. To our knowledge, it is the first point process sampler to date to perform in such highly complex state spaces.

2 Point Process background

A point process describes random configurations of points in a continuous bounded set K. Mathematically speaking, a point process Z is a measurable mapping from a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ to the set of configurations of points in K such that

$$\forall \omega \in \Omega, p_i \in K, Z(\omega) = \{p_1, ..., p_{n(\omega)}\} \tag{1}$$

where $n(\omega)$ is the number of points associated with the event ω. We denote by $\mathcal{P}$, the space of configurations of points in K. The most natural point process is the homogeneous Poisson process for which the number of points follows a

discrete Poisson distribution whereas the position of the points is uniformly and independently distributed in K. Point processes can also provide more complex realizations of points by being specified by a density $h(.)$ defined in $\mathcal{P}$ and a reference measure $\mu(.)$ under the condition that the normalization constant of $h(.)$ is finite:

$$\int_{\boldsymbol{p}\in\mathcal{P}} h(\boldsymbol{p})d\mu(\boldsymbol{p}) < \infty \tag{2}$$

The measure $\mu(.)$ having the density $h(.)$ is usually defined via the intensity measure $\nu(.)$ of an homogeneous Poisson process. Specifying a density $h(.)$ allows the insertion of data consistency, and also the creation of spatial interactions between the points. In particular, the Markovian property can be used in point processes, similarly to random fields, to create a spatial independence of the points in a neighborhood. Note also that $h(.)$ can be expressed by a Gibbs energy $U(.)$ such that

$$h(.) \propto \exp -U(.) \tag{3}$$

From points to parametric objects - What makes point processes attractive for vision is the possibility of marking each point p_i by additional parameters m_i such that the point becomes associated with an object $x_i = (p_i, m_i)$. We denote by $\mathcal{C}$, the corresponding space of object configurations where each configuration is given by $\boldsymbol{x} = \{x_1, ..., x_{n(\boldsymbol{x})}\}$. For example, a point process on $K \times M$ with $K \subset \mathbb{R}^2$ and the additional parameter space $M =]-\frac{\pi}{2}, \frac{\pi}{2}] \times [l_{min}, l_{max}]$ can be seen as random configurations of 2D line-segments since an orientation and a length are added to each point (see Fig. 1). Such point processes are also called marked point processes in the literature.
The most popular family of point processes corresponds to the Markov point processes of objects specified by Gibbs energies on $\mathcal{C}$ of the form

$$\forall \boldsymbol{x} \in \mathcal{C}, \;\; U(\boldsymbol{x}) = \sum_{x_i \in \boldsymbol{x}} D(x_i) \; + \sum_{x_i \sim x_j} V(x_i, x_j) \tag{4}$$

where $\sim$ denotes the symmetric neighborhood relationship of the Markov point process, $D(x_i)$ is a unitary data term measuring the quality of object x_i with respect to data, and $V(x_i, x_j)$, a pairwise interaction term between two neighboring objects x_i and x_j. The $\sim$relationship is usually defined via a limit distance ϵ between points such that

$$x_i \sim x_j = \{(x_i, x_j) \in \boldsymbol{x}^2 : i > j, ||p_i - p_j||_2 < \epsilon\} \tag{5}$$

In the sequel, we consider Markov point processes of this form. Note that this energy form has similarities with the standard labeling energies for MRFs. As explained in [18], our problem can be seen as a generalization of these models.

Simulation - Point processes are usually simulated by a Reversible Jump MCMC sampler [10] to search for the configuration which minimizes the energy U. This sampler consists of simulating a discrete Markov Chain $(X_t)_{t\in\mathbb{N}}$ on

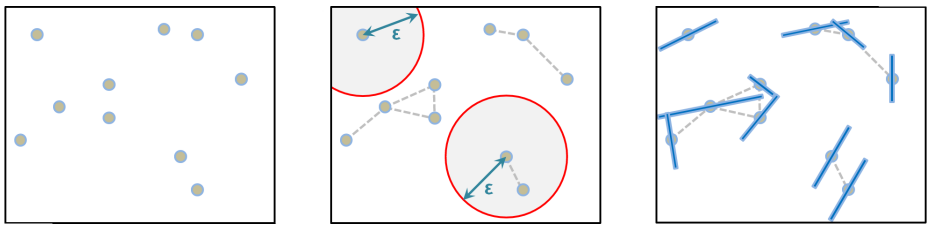


Fig. 1. From left to right: realizations of a point process in 2D, of a Markov point process, and of a Markov point process of line-segments.The grey dashed lines represent the pairs of points interacting with respect to the neighboring relationship which is specified here by a limit distance ϵ between two points.

the configuration space $\mathcal{C}$, converging towards an invariant measure specified by U. At each iteration, the current configuration $\boldsymbol{x}$ of the chain is locally perturbed to a configuration $\boldsymbol{y}$ according to a density function $Q(\boldsymbol{x} \to .)$, also called a kernel. The perturbations are local, which means that $\boldsymbol{x}$ and $\boldsymbol{y}$ are close, and differ by no more than one object. The configuration $\boldsymbol{y}$ is then accepted as new state of the chain with a certain probability depending on the energy variation between $\boldsymbol{x}$ and $\boldsymbol{y}$, and a relaxation parameter T_t. The kernel Q can be formulated as a mixture of sub-kernels Q_m chosen with a probability q_m such that

$$Q(\boldsymbol{x} \to .) = \sum_m q_m Q_m(\boldsymbol{x} \to .) \tag{6}$$

Each sub-kernel is usually dedicated to specific types of moves, as the creation/removal of an object (Birth and Death kernel) or the modification of parameters of an object (*e.g.* translation, dilatation or rotation kernels). The kernel mixture must allow any configuration in $\mathcal{C}$ to be reached from any other configuration in a finite number of perturbations (irreductibility condition of the Markov chain), and each sub-kernel has to be reversible, *i.e.* able to propose the inverse perturbation.

Algorithm 1 RJMCMC sampler [10]

1- Initialize $X_0 = \boldsymbol{x}_0$ and T_0 at $t = 0$;
2- At iteration t, with $X_t = \boldsymbol{x}$,

- Choose a sub-kernel $\mathcal{Q}_m$ according to probability q_m
- Perturb $\boldsymbol{x}$ to $\boldsymbol{y}$ according to $Q_m(\boldsymbol{x} \to .)$
- Compute the Green ratio

$$R = \frac{\mathcal{Q}_m(\boldsymbol{y} \to \boldsymbol{x})}{\mathcal{Q}_m(\boldsymbol{x} \to \boldsymbol{y})} \exp\left(\frac{U(\boldsymbol{x}) - U(\boldsymbol{y})}{T_t}\right) \tag{7}$$

- Choose $X_{t+1} = \boldsymbol{y}$ with probability $\min(1, R)$, and $X_{t+1} = \boldsymbol{x}$ otherwise

The RJMCMC sampler is controlled by the relaxation parameter T_t, called the temperature, depending on time t and approaching zero as t tends to infinity. Although a logarithmic decrease of T_t is necessary to ensure the convergence to the global minimum from any initial configuration, one uses a faster geometric decrease which gives an approximate solution close to the optimum [1].

3 New sampling procedure

3.1 Simultaneous multiple perturbations

The conventional RJMCMC sampler performs successive perturbations on objects. Such a procedure is obviously long and fastidious, especially for large scale problems. A natural idea but still unexplored for Markov point processes consists in sampling objects in parallel by exploiting their conditional independence outside the local neighborhood. Such a strategy implies partitioning the space K so that simultaneous perturbations are performed at locations far enough apart to not interfere and break the convergence properties.

From sequential to parallel sampling - Let $(X_t)_{t\in\mathbb{N}}$, be a Markov chain simulating a Markov point process, and $\{c_s\}$ be a partition of the space K, where each component c_s is called a cell. Two cells c_s and $c_{s'}$ are said *independent on* X if the transition probability for any random perturbation falling in c_s and at any time t does not depend on either objects or perturbations falling in $c_{s'}$, and vice versa. One can easily check that the transition probability of two successive perturbations falling in independent cells under the temperature T_t is equal to the product of the transition probabilities of each perturbation under the same temperature [18]. In other words, realizing two successive perturbations on independent cells at the same temperature is equivalent to performing them in parallel.

Ensuring cell independence - Two independent cells must be located at a minimum distance from each other. As illustrated in Fig. 2, this distance must take into account both the width of the neighboring relationship, *i.e.* ϵ, and the length of the biggest move allowed as object perturbation, denoted by $\delta_{\max}$. Independence between two cells c_s and $c_{s'}$ is then guaranteed if

$$\min_{p\in c_s,\, p'\in c_{s'}} ||p-p'||_2 \geq \epsilon + 2\delta_{\max} \tag{8}$$

Independent cell gathering - The natural idea consists of partitioning the space K into a regular mosaic of cells with size greater than or equal to $\epsilon + 2\delta_{\max}$. The cells can then be regrouped into $2^{\dim K}$ sets such that each cell is adjacent to cells belonging to different sets. Fig. 2 illustrates the regrouping scheme for $\dim K = 2$ and $\dim K = 3$. This regrouping scheme ensures the mutual independence between all the cells of a same set. In the sequel, such a set is called a *mic-set* (set of Mutually Independent Cells).

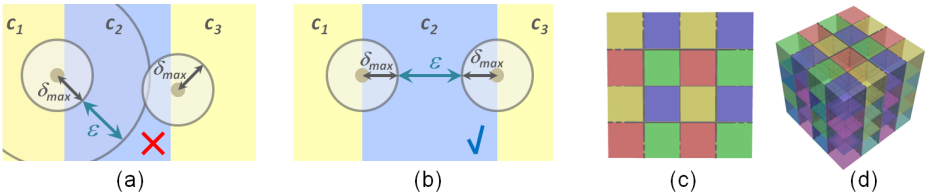


Fig. 2. Independence of cells. (a) the width of the cell c_2 is not large enough to ensure the independence of the cells c_1 and c_3: the two grey points in c_1 and c_3 cannot be perturbed at the same time. (b) the cells c_1 and c_3 are independent as Eq. 8 is satisfied. (c) in dimension two, the cells are squares regrouped into 4 mic-sets (yellow, blue, red and green). Each cell is adjacent to cells belonging to different mic-sets. (d) in dimension three, the cells are cubes regrouped into 8 mic-sets.

3.2 Non-uniform point distributions

Sampling objects in parallel via a regular partitioning, as illustrated on Fig. 2-(c)&(d), is however not optimal because the spatial point distribution is necessarily uniform and does not take into account the characteristics of observed scenes. To overcome this problem, a non-regular partitioning of the scene is created by exploiting data-based knowledge.

Non-uniform kernel from uniform sub-kernels - Mixtures of sub-kernels are frequently used to simulate point processes by MCMC dynamics, each sub-kernel corresponding to a perturbation type (*e.g.* birth and death, translation, rotation, etc). However, the idea consisting of accumulating sub-kernels with spatial restrictions to create non-uniform point distributions has not been exploited in the literature. Let $\{c_s\}^{(1)}, .., \{c_s\}^{(L)}$, be L partitions of the space K such that $\{c_s\}^{(i)}$ is a subdivided partition of $\{c_s\}^{(i-1)}$. The L partitions define a space-partitioning tree, denoted $\mathcal{K}$ and whose levels each correspond to a partition of K. By associating with each cell contained in $\mathcal{K}$ a uniform sub-kernel spatially restricted to the subspace supporting this cell, a non-uniform kernel can be created by accumulation, as defined in Eq. 6 and illustrated in Fig. 3.

Data-driven space-partitioning tree - A regular 1-to-$2^{\dim K}$ hierarchical subdivision scheme is considered to build a partitioning tree, typically a quadtree in dimension two and an octree in dimension three. The subdivision of the cells is driven by the data. We assume that a class of interest in K, in which the objects have a high probability to belong to, can be roughly distinguished from the data. The extraction of such a class is not addressed here, and is supposed to be done by a segmentation algorithm of the literature adapted to the considered application. A cell at a given level of the tree is divided into $2^{\dim K}$ cells at the next level if it overlaps with the given class of interest. The hierarchical decomposition is stopped when the minimal size of the cell becomes inferior to $\epsilon + 2\delta_{\max}$, *i.e.* when the cell independence condition (Eq. 8) is not longer valid. The partitioning tree allows the creation of a non uniform point distribution

naturally and efficiently. Indeed, the density progressively decreases when moving far from the class of interest as shown in Fig. 3, while being ensured to be non-null. The point distribution is thus not directly affected when the class of interest is inaccurately extracted.

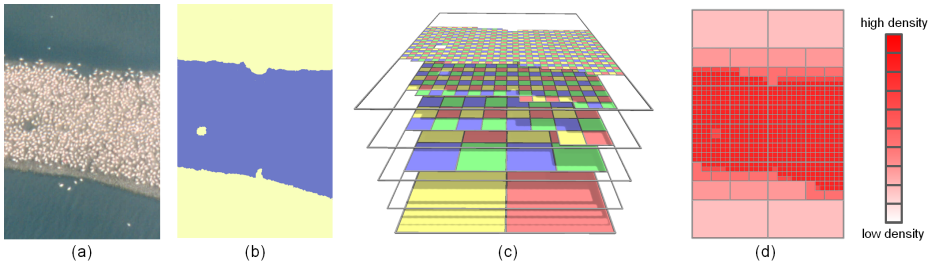


Fig. 3. Space partitioning tree in dimension two. (b) A class of interest (blue area) is estimated from (a) an input image. (c) A quadtree is created so that the levels are recursively partitioned according to the class of interest. Each level is composed of four mic-sets (yellow, blue, red and green sets of cells) to guaranty the sampling parallelization. (d) The cell accumulation at the different levels designs a non uniform distribution. Note how the distribution naturally describes the class of interest by progressively decreasing the density when moving away.

Kernel formulation - Given a space-partitioning tree $\mathcal{K}$ composed of L levels, and $2^{\dim K}$ mic-sets for each level, we can formulate a general kernel Q as a mixture of uniform sub-kernels $Q_{c,t}$, each sub-kernel being defined on the cell c of $\mathcal{K}$, by the perturbation type $t \in \mathcal{T}$, such that

$$\forall \boldsymbol{x} \in \mathcal{C},\ Q(\boldsymbol{x} \rightarrow .) = \sum_{c \in \mathcal{K}} \sum_{t \in \mathcal{T}} q_{c,t}\, Q_{c,t}(\boldsymbol{x} \rightarrow .) \tag{9}$$

where $q_{c,t}$ is the probability of choosing sub-kernel $Q_{c,t}(\boldsymbol{x} \rightarrow .)$, given by

$$q_{c,t} = \frac{\Pr(t)}{\#\text{cells in } \mathcal{K}} \tag{10}$$

Four types of kernels are usually considered in practice so that we have $\mathcal{T} = \{\text{birth and death, translation, rotation, scale}\}$. The switching kernel can also be used when objects have several possible types. Note that the kernel Q is reversible as a sum of reversible sub-kernels. Note also that this kernel allows us to visit the whole configuration space $\mathcal{C}$ since it is guaranteed by the sub-kernels of the coarsest level of $\mathcal{K}$.

3.3 Sampler formulation

The kernel defined in Eq. 9 is embedded into the MCMC dynamics so that the new sampler allows the association of multiple perturbations performed in parallel with relevant non-uniform point distributions based on data-driven space

Algorithm 2 Our parallel sampler

1-Initialize $X_0 = \boldsymbol{x}_0$ and T_0 at $t = 0$;
2-Compute a space-partitioning tree $\mathcal{K}$;
3-At iteration t, with $X_t = \boldsymbol{x}$,

- Choose a mic-set $S_{mic} \in \mathcal{K}$ and a kernel type $t \in \mathcal{T}$ according to probability $\sum_{c \in S_{mic}} q_{c,t}$
- For each cell $c \in S_{mic}$,
  - Perturb $\boldsymbol{x}$ in the cell c to a configuration $\boldsymbol{y}$ according to $Q_{c,t}(\boldsymbol{x} \to .)$
  - Calculate the Green ratio

$$R = \frac{\mathcal{Q}_{c,t}(\boldsymbol{y} \to \boldsymbol{x})}{\mathcal{Q}_{c,t}(\boldsymbol{x} \to \boldsymbol{y})} \exp\left(\frac{U(\boldsymbol{x}) - U(\boldsymbol{y})}{T_t}\right) \tag{11}$$

  - Choose $X_{t+1} = \boldsymbol{y}$ with probability $\min(1, R)$, and $X_{t+1} = \boldsymbol{x}$ otherwise

partitioning trees. Algorithm 2 details the sampling procedure.
Note that the temperature parameter is updated after each series of simultaneous perturbations such that the temperature decrease is equivalent to a cooling schedule by plateau in a standard sequential MCMC sampling. Note also that the hierarchical partitioning of K protects the sample from mosaic effects. In practice, the sampling is stopped when no perturbation has been accepted during a certain number of iterations.

4 Experiments

The proposed algorithm has been implemented on GPU for $\dim K = 2$ and $\dim K = 3$, and tested on various problems including population counting, line-network extraction from images, and object recognition from 3D point clouds. Note that additional results and comparisons can be found in [18], as well as details on energy formulations.

4.1 Implementation

The algorithm has been implemented on GPU using CUDA. A thread is dedicated to each simultaneous perturbation so that operations are performed in parallel for each cell of a mic-set. The sampler is thus all the more efficient as the mic-set contains many cells, and generally speaking, as the scene supported by K is large. Moreover, the code has been programmed to avoid time-consuming operations. In particular, the threads do not communicate between each other, and memory coalescing permits fast memory access. The memory transfer between CPU and GPU has also been minimized by indexing the parametric objects. The experiments presented in this section have been performed on a 2.5 Ghz Xeon computer with a Nvidia graphics card (Quadro 4800, architecture 1.3).

4.2 Point processes in 2D

The algorithm has been evaluated on population counting problems from large-scale images using a point process in 2D, *i.e.* with $\dim K = 2$. The problem presented in Fig. 4 consists in detecting migrating birds to extract information on their number, their size and their spatial organization. The point process is marked by ellipses which are simple geometric objects defined by a point (center of mass) and 3 additional parameters, and are well adapted to capture the bird contours. The energy is specified by a unitary data term based on the Bhattacharyya distance between the radiometry inside and outside of the object, and a pairwise interaction penalizing the strong overlapping of objects [18].

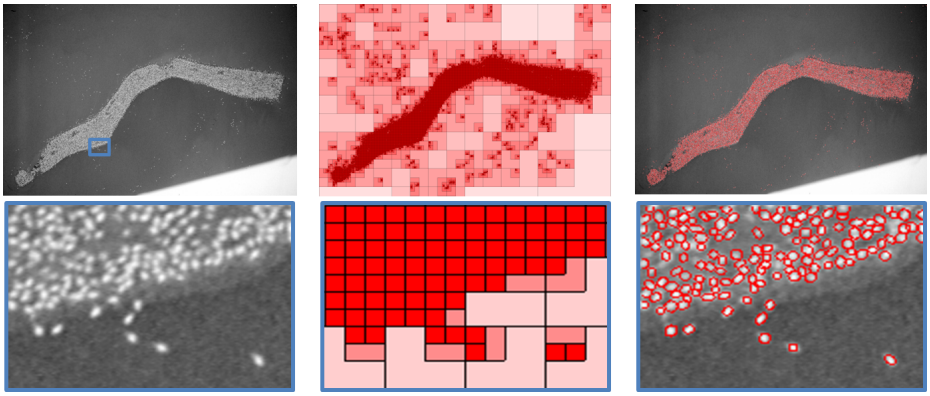
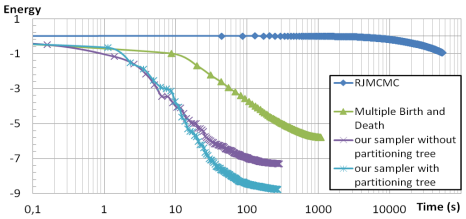


Fig. 4. Bird counting by a point process of ellipses. (right) More than ten thousand birds are extracted by our algorithm in a few minutes from (left) a large scale aerial image. (middle) A quad-tree structure is used to create a non-uniform point distribution. Note, on the cropped parts, how the birds are accurately captured by ellipses in spite of the low quality of the image and the partial overlapping of birds.

Computation time, quality of the reached energy, and stability are the three important criteria used to evaluate and compare the performance of samplers. As shown on Fig. 5, our algorithm obtains the best results for each of the criteria compared to the existing samplers. In particular, we reach a better energy (-8.76 *vs* -5.78 for [2] and -2.01 for [10]) while significantly reducing computation times (269 sec *vs* 1078 sec for [2] and 2.8×10^6 sec for [10]). Note that, for the reasons mentioned in Section 4.1, this gap in performance increases when the input scene becomes larger. Fig. 5 also underlines an important limitation of the reference point process sampler for population counting [2] compared to our algorithm. Indeed, the discretization of the object parameters required in [2] causes approximate detection and localization of objects which explains the average quality of the reached energy. The stability is analyzed by the coefficient of variation, defined as the standard deviation over mean and known to be a relevant statistical measure for comparing methods having different means. Our



	Coefficient of variation		
	energy	time	#objects
RJMCMC [10]	7.3%	4.2%	1.7%
multiple birth and death [2]	5.0%	2.1%	1.3%
our sampler without partitioning tree	7.4%	6.2%	1.6%
our sampler with partitioning tree	4.4%	1.8%	1.1%

Fig. 5. Performances of the various samplers. (left) The graph describes the energy decrease over time from the bird image presented in Fig. 4. Time is represented using a logarithmic scale. Note that the RJMCMC algorithm [10] is so slow that the convergence is not displayed on the graph (2.9×10^7 iterations are required versus 1.8×10^4 for our algorithm). (right) The table presents the coefficients of variation of the energy, time and number of objects reached at the convergence over 50 simulations.

sampler provides a better stability than the existing algorithms. The impact of the data-driven space partitioning tree is also measured by performing tests with uniform point distributions. The performances decrease but remain better than the existing algorithms. In particular, the sampler loses stability, and the objects are detected and located less accurately than with the partitioning tree.

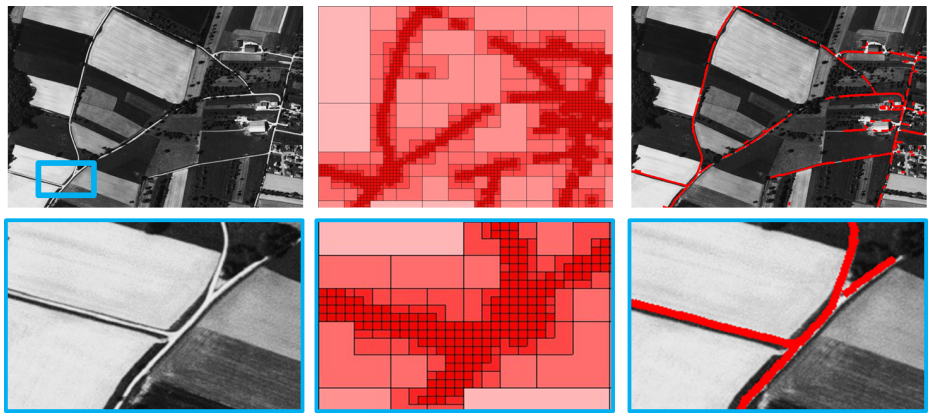


Fig. 6. Line-network extraction by a point process of line-segments. (middle) Even if the point distribution is roughly estimated, (right) the road network is recovered (red segments) by our algorithm in 16 seconds from (left) a satellite image. Note that, as shown on the close-up, some parts of the network can be omitted when roads are hidden by trees at some locations: existing methods also encounter difficulties in such cases.

The algorithm has also been tested on line-network extraction from images. The parametric objects here are line-segments defined by a point (center of mass) and two additional parameters (length and orientation). Contrary to the population counting model, the pairwise potential includes a connexion interaction for linking the line-segments. Figure 6 shows a road network extraction result obtained from a satellite image whereas Table 1 provides elements of comparisons with

existing methods. The result quality in terms of road under/over-detection is globally similar to existing approaches (higher than [4] and [19], and lower than [7]), but our algorithm significantly improves the computing times. 16 seconds are required in our case, compared to 7 minutes by a Jump-Diffusion algorithm [4], 155 minutes for a RJMCMC method [7], and 60 minutes for an Active Contour based approach [19].

	our sampler	Jump-Diffusion [4]	RJMCMC [7]	Active contour [19]
Time (minute)	0.26	6.35	155	60
Over-detection	0.45%	2.06%	0.06%	2.28%
Under-detection	32.7%	52.7%	17%	58.9%
Representation	line-segment	line-segment	line-segment	pixels

Table 1. Comparisons with existing line-network extraction methods from the image presented in Fig. 6.

4.3 Point processes in 3D

We tested our algorithm with $\dim K = 3$ on an original object recognition problem from laser scans. The goal is to extract trees from unstructured point clouds containing a lot of outliers, noise and other different objects (buildings, ground, cars, fences, wires, *etc*), and to recognize their shapes and types. The objects associated with the point process correspond to a library of different 3D-templates of trees detailed in [18]. The unitary data term of the energy measures the distance from points to a 3D-template, whereas the pairwise interaction takes into account constraints on object overlapping as well as on tree type competition. Compared to the former applications, the configuration space $\mathcal{C}$ is of higher dimension since the objects are parametrically more complex. This allows our algorithm to exploit more deeply its potential. The rotation kernel is not used here since the objects are invariant by rotation. However, we use a switching kernel in order to exchange the type of an object.
Fig. 7 shows results obtained from laser scans of large urban and natural environments. 30 (respectively 5.4) thousand trees are extracted in 96 (resp. 53) minutes on the 3.7km^2 mountain area (resp. 1km^2 urban area) from 13.8 (resp. 2.3) million input points. The computation times can appear high, but finding non-trivial 3D-objects in such larges scenes by point processes is a challenge which, to our knowledge, has not been achieved until now due to the extreme complexity of the state space. Note also that the performances could be improved by reducing the space $\mathcal{C}$ with a 3D-point process on manifolds, *i.e.* where the z-coordinate of points is determined by an estimated ground surface.
Evaluating the detection quality with accuracy for this application is a difficult task since no ground truth exists. As illustrated on the cropped part in Fig. 7, we have manually indexed the trees on different zones from aerial images acquired with the laser scans. The objects are globally well located and fitted to the input points with few omissions, even when trees are surrounded by other types of urban entities such as buildings. The non-overlapping constraint of the energy

allows us to obtain satisfactory results for areas with high tree density. Errors frequently occur in distinguishing the tree type in spite of the tree competition term of the energy.

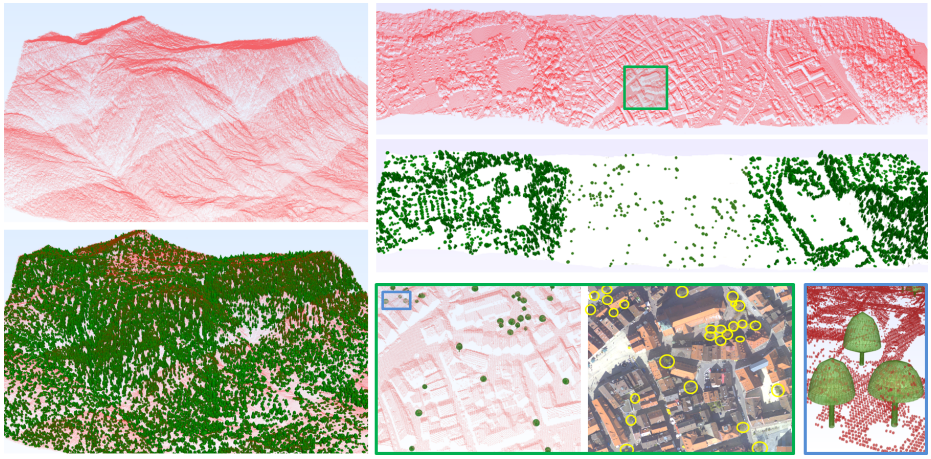


Fig. 7. Tree recognition from point clouds by a 3D-point process specified by 3D-parametric models of trees. Our algorithm detects trees and recognizes their shapes in large-scale (left, input scan: 13.8M points) natural and (top right, input scan: 2.3M points) urban environments, in spite of other types of urban entities, e.g. buildings, car and fences, contained in input point clouds (red dot scans). An aerial image is joined to (bottom right) the cropped part to provide a more intuitive representation of the scene and the tree location. Note, on the cropped part, how the parametric models fit well to the input points corresponding to trees, and how the interaction of tree competition allows the regularization of the tree type in a neighborhood.

5 Conclusion

We propose a new algorithm to sample point processes whose strengths lean on the exploitation of Markovian properties to enable the sampling to be performed in parallel, and the integration of a data-driven mechanism allowing efficient distributions of the points in the scene. Our algorithm improves the performances of the existing samplers in terms of computing times and stability, especially on large scenes where the gain is very important. It can be used without particular restrictions, contrary to most samplers, and even appears as an interesting alternative to the standard optimization techniques for MRF labeling problems. In particular, one can envisage using the model proposed in Section 4.3 to extract any type of parametric objects from large 3D-point clouds.
In future works, it would be interesting to implement the algorithm on other GPU architectures more adapted to the manipulation of 3D data structures,

e.g. architecture 2.0, so that the performances for point processes in 3D could be improved.

Acknowledgments. This work was partially funded by the European Research Council (ERC Starting Grant "Robust Geometry Processing", Grant agreement 257474). The authors thank the French Mapping Agency (IGN) and the Tour du Valat for provinding the datasets.

References

1. Baddeley, A.J., Lieshout, M.V.: Stochastic geometry models in high-level vision. Journal of Applied Statistics **20** (1993)
2. Descombes, X., Minlos, R., Zhizhina, E.: Object extraction using a stochastic birth-and-death dynamics in continuum. JMIV **33** (2009)
3. Ge, W., Collins, R.: Marked point processes for crowd counting. In: CVPR, Miami, U.S. (2009)
4. Lafarge, F., Gimel'farb, G., Descombes, X.: Geometric feature extraction by a multi-marked point process. PAMI **32** (2010)
5. Lieshout, M.V.: Depth map calculation for a variable number of moving objects using markov sequential object processes. PAMI **30** (2008)
6. Mallet, C., Lafarge, F., Roux, M., Soergel, U., Bretar, F., Heipke, C.: A marked point process for modeling lidar waveforms. IP **19** (2010)
7. Lacoste, C., Descombe, X., Zerubia, J.: Point processes for unsupervised line network extraction in remote sensing. PAMI **27** (2005)
8. Sun, K., Sang, N., Zhang, T.: Marked point process for vasculartree extraction on angiogram. In: EMMCVPR, Ezhou, China (2007)
9. Utasi, A., Benedek, C.: A 3-D marked point process model for multi-view people detection. In: CVPR, Colorado Springs, U.S. (2011)
10. Green, P.: Reversible Jump Markov Chains Monte Carlo computation and Bayesian model determination. Biometrika **82** (1995)
11. Hastings, W.: Monte Carlo sampling using Markov chains and their applications. Biometrika **57** (1970)
12. Han, F., Tu, Z.W., Zhu, S.: Range image segmentation by an effective jump-diffusion method. PAMI **26** (2004)
13. Srivastava, A., Grenander, U., Jensen, G., Miller, M.: Jump-Diffusion Markov processes on orthogonal groups for object pose estimation. Journal of Statistical Planning and Inference **103** (2002)
14. Tu, Z., Zhu, S.: Image Segmentation by Data-Driven Markov Chain Monte Carlo. PAMI **24** (2002)
15. Harkness, M., Green, P.: Parallel chains, delayed rejection and reversible jump mcmc for object recognition. In: BMVC, Bristol, U.K. (2000)
16. Byrd, J., Jarvis, S., Bhalerao, A.: On the parallelisation of mcmc-based image processing. In: IEEE International Symposium on Parallel and Distributed Processing, Atlanta, U.S. (2010)
17. Gonzalez, J., Low, Y., Gretton, A., Guestrin, C.: Parallel Gibbs sampling: From colored fields to thin junction trees. Journal of Machine Learning Research (2011)
18. Verdié, Y., Lafarge, F.: Towards the parallelization of Reversible Jump Markov Chain Monte Carlo algorithms for vision problems. Research report 8016, INRIA (2012)
19. Rochery, M., Jermyn, I., Zerubia, J.: Higher order active contours. IJCV **69** (2006)