Real-Time Robotic Search using Structural Spatial Point Processes

Olov Andersson[*]
Dept of Computer and Info Science
Linköping University
olov.andersson@liu.se

Per Sidén[*]
Dept of Computer and Info Science
Linköping University
per.siden@liu.se

Johan Dahlin
Kotte Consulting AB
uni@johandahlin.com

Patrick Doherty
Dept of Computer and Info Science
Linköping University
patrick.doherty@liu.se

Mattias Villani
Dept of Computer and Info Science
Linköping University and
Department of Statistics
Stockholm University
mattias.villani@gmail.com

Abstract

Aerial robots hold great potential for aiding Search and Rescue (SAR) efforts over large areas, such as during natural disasters. Traditional approaches typically search an area exhaustively, thereby ignoring that the density of victims varies based on predictable factors, such as the terrain, population density and the type of disaster. We present a probabilistic model to automate SAR planning, with explicit minimization of the expected time to discovery. The proposed model is a spatial point process with three interacting spatial fields for i) the point patterns of persons in the area, ii) the probability of detecting persons and iii) the probability of injury. This structure allows inclusion of informative priors from e.g. geographic or cell phone traffic data, while falling back to latent Gaussian processes when priors are missing or inaccurate. To solve this problem in real-time, we propose a combination of fast approximate inference using Integrated Nested Laplace Approximation (INLA), and a novel Monte Carlo tree search tailored to the problem. Experiments using data simulated from real world Geographic Information System (GIS) maps show that the framework outperforms competing approaches, finding many more injured in the crucial first hours.

[*]Equal contribution.

1 INTRODUCTION

There is rising interest in employing robots such as unmanned aerial vehicles (UAV) for search and rescue operations. Such rescue robotics is an emerging research area (Murphy et al., 2008), tackling a range of problems from automating logistics, aid delivery and communications deployment, down to sensor and motion planning of individual robots.

Here we focus on how to improve victim search during disasters by learning and planning with probabilistic models in real-time. The aim is an automated and near-optimal solution for surveying disaster zones that can be used both as a component of a larger robotic system, or as decision support for first responders. Locating injured victims early is crucial in reducing suffering and mortality rates. Many disasters such as earthquakes, flooding, and even terrorist attacks have a multitude of victims and can cover large areas, where manual approaches take valuable time away from rescuers. We seek a more hands-off probabilistic solution, where humans encode their domain knowledge via priors, and are then free to focus on rescue efforts while the aerial robots do what they do best - scout large areas.

We propose to leverage probabilistic models that reflect the problem structure, as well as approximate inference and planning to solve the search problem within stipulated real-time requirements. To this end we separately model the population density, injury probability and detection probability. In conjunction with a terrain-based exploration time, inferring these factors allows us to optimize the sequence of exploratory moves to minimize

discovery time. A priori, a densely populated area is likely to contain more victims than a sparsely populated one, and a field or road is both faster and easier to explore than a forest. In scenarios such as earthquakes, areas near buildings are likely to contain more victims than roads. During a terrorist attack, whatever early information exists can be encoded and explored first. Without further information, the system should sample high population areas, and if it stumbles on victims, *learn* a local adjustment in the injury model and focus on that area. We therefore both allow strong priors on these variables, and can revert back to a spatial process when the priors are uninformative or inaccurate. For example, one can automatically pull estimates of population density from GIS covariates and learn both global and local deviations during the search.

However, real-time inference and search in such spatial problems is very challenging for three main reasons: i) only a small part of the point pattern is observed early in the search, ii) the parameter space in a spatial model is high-dimensional and real-time sequential inference is therefore computationally challenging, and iii) optimal search is a computationally hard planning problem under uncertainty, which we need to solve in real-time.

The main contributions of this paper are:

- A powerful structured spatial probabilistic model for the disaster search domain with the possibility to make effective use of prior information
- Real-time inference via deterministic Integrated Nested Laplace Approximation (INLA), so the model can be updated online
- A real-time approximation to the search planning problem using a novel variant of Monte-Carlo tree search (MCTS) with action abstractions ("jumps").

To the best of our knowledge, this is the first work on real-time learning of structured probabilistic models for the disaster search problem. While there is considerable work on informed exploration in robotics, it stems from mapping or monitoring tasks (Morere et al., 2017; Singh et al., 2009), which ignores the structure of the disaster response problem.

Bayesian approaches have previously been used in conventional search efforts for missing persons (Kratzke et al., 2010). Automating these with UAVs have been considered in Waharte and Trigoni (2010). However, as these search for one entity they mainly rely on strong priors. Large-scale disaster response situations allow us to both draw on a rich set of data from GIS, and learn based on a large number of observations.

Just the planning part of the problem is itself computationally hard, and exhaustive coverage search or heuristics seem popular in practice (Rudol and Doherty, 2008; Goodrich et al., 2008; Xu et al., 2011). Waharte and Trigoni (2010) found heuristics the best choice for real-time operation. In the planning literature, MCTS was also recently proposed for disaster response (Baker et al., 2016), but without the proposed model learning or action abstractions in planning.

The paper is organized as follows. We first define the proposed structured probabilistic model and its use of spatial point processes. In section 3, we describe how INLA is used for real-time approximate inference. The search planning problem and the MCTS approximation is introduced in section 4, and finally we present results on a range of search and rescue scenarios in section 5.

2 SPATIAL POINT PROCESS FOR DISASTER RESPONSE

Learning a spatial process for real-time search is a difficult inference problem since only a small part of the point pattern is observed at any given point in time. We therefore develop a structural spatial point process model that allows us to complement the observed point pattern with prior knowledge, for example about the terrain or cell phone traffic in the search area.

Let $\mathbf{y}_i$ denote a two-dimensional vector with spatial coordinates for the ith data point, e.g. the location of the ith person. Let $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)$ denote the spatial *point pattern* over a region of interest $S \subset \mathbb{R}^2$, for example the observed spatial locations of n individuals over a rectangular region. The simplest example of a *point process* model for such data is the homogeneous Poisson process for which points are uniformly distributed over S with constant intensity λ.

Disaster response scenarios have a more complex *marked point pattern* where a given person i) may or may not be detected by a searching UAV, and ii) may or may not be injured. We propose a model built up by three interacting spatial fields:

- the population intensity $\lambda(\mathbf{s})$,
- the detection probability $r(\mathbf{s})$,
- the probability of being injured $q(\mathbf{s})$,

where $\mathbf{s} \in S$. Figure 1 displays a sample realization from the model over a map of the town of Gamleby in Sweden. The following subsection gives the details for each of the fields, and how specific prior information can be used in each of the three parts of the process.

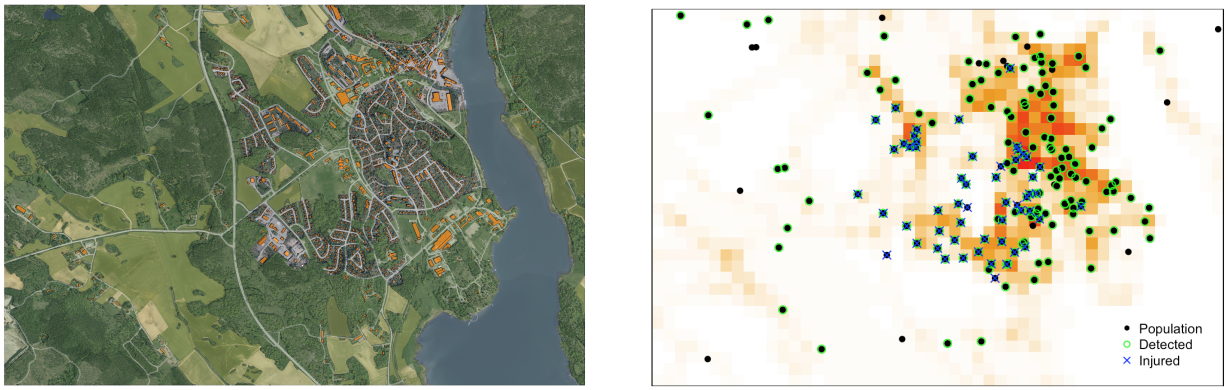

Figure 1: Left: Map of the town of Gamleby, Sweden with buildings (orange), forest (dark green), fields (light green), roads (light grey) and water (dark grey) marked out. Right: a sample realization from the model showing the population intensity overlaid by persons in the area (filled dots), detected persons (green circles) and injured persons (blue crosses). There is increased population intensity at buildings and roads, decreased population density in water, lower detection probability in the forest and increased probability of injury due to an explosion in the southwest part of the town. The displayed points have been thinned out by a factor 10 for visualization purposes.

2.1 POPULATION MODEL

Let $\mathbf{Y}^\star = (\mathbf{y}_1^\star, \ldots, \mathbf{y}_n^\star)$ denote the point pattern of persons over a region S, and let $N_{y^\star}(\tilde{S})$ denote the number of persons in the subset $\tilde{S} \subset S$. We model the point pattern $\mathbf{Y}^\star$ by a Log Gaussian Cox Process (LGCP, Møller et al. (1998))

$$N_{y^\star}(\tilde{S})|\lambda \sim \text{Poisson}\left(\int_{\mathbf{s}\in\tilde{S}} \lambda(\mathbf{s})d\mathbf{s}\right),$$

with log intensity surface given by

$$\log\lambda(\mathbf{s}) = \alpha_\lambda + \mathbf{x}_\lambda^\top(\mathbf{s})\boldsymbol{\beta}_\lambda + \xi_\lambda(\mathbf{s}),$$

where α_λ is an intercept, $\mathbf{x}_\lambda(\mathbf{s})$ is a vector with spatial covariates, $\boldsymbol{\beta}_\lambda$ are regression coefficients and $\xi_\lambda(\mathbf{s})$ is a zero mean Gaussian Process (GP) over S. The spatial covariates $\mathbf{x}_\lambda(\mathbf{s})$ could contain any spatial prior information that helps explain the population intensity λ, for example the location of buildings and water; such information is readily available from GIS systems. The remaining part of λ is modeled as a GP $\xi_\lambda(\mathbf{s})$ with a smooth kernel, as people tend to cluster together. We will throughout this paper focus on GPs with kernels from the Matérn family with smoothness parameter $\nu = 1$ (Rasmussen and Williams, 2006, Ch. 4). Finally, the Poisson distribution can be replaced by some other suitable distribution, in particular the more flexible Negative Binomial distribution which is part of the standard INLA library.

2.2 DETECTION MODEL

The interpretation of $\mathbf{Y}^\star$ are all persons that could be possibly observed by a UAV flying at low height and at good sighting conditions. In practice, conditions may not be perfect, and we model the persons actually observed $\mathbf{Y}$ through the detection probability $r(\mathbf{s})$ of observing a person at point $\mathbf{s}$, generating a thinned Poisson process

$$N_y(\tilde{S})|r,\lambda \sim \text{Poisson}\left(\int_{\mathbf{s}\in\tilde{S}} r(\mathbf{s})\lambda(\mathbf{s})d\mathbf{s}\right),$$

where $N_y(\tilde{S})$ denotes the number of observed persons in $\tilde{S}$. The detection probability $r(\mathbf{s})$ is modeled with

$$\log r(\mathbf{s}) = \mathbf{x}_r^\top(\mathbf{s})\boldsymbol{\beta}_r,$$

where $\mathbf{x}_r(\mathbf{s})$ contain prior information about for example terrain type that might affect visibility. Technically, $r(\mathbf{s})$ is a detection *rate*, since positive values of $\mathbf{x}_r^\top(\mathbf{s})\boldsymbol{\beta}_r$ leads to values for $r(\mathbf{s})$ greater than 1. However, it can be interpreted as a detection probability when $\mathbf{x}_r^\top(\mathbf{s})\boldsymbol{\beta}_r$ is non-negative for all $\mathbf{s}$, which, if not already the case, could always be achieved by re-balancing the model, through increasing α_λ and modifying $\mathbf{x}_r^\top(\mathbf{s})$ and $\boldsymbol{\beta}_r$.

2.3 INJURY MODEL

We assume that when the disaster strikes all persons have a spatially varying probability $q(\mathbf{s})$ of being injured. Let $\mathbf{w}$ be a binary vector of length n with $w_i = 1$ iff the ith detected person is injured, and $w_i = 0$ otherwise. This is an example of marked point pattern where each observed point is marked by a binary variable. We assume a geostatistical marking process where the marking process (injured) is independent of the point pattern process (pattern of detected persons) and assumed to follow

$$w_i|q \sim \text{Bernoulli}\left(q(\mathbf{y}_i)\right),$$

where

$$\log\left(\frac{q(\mathbf{s})}{1-q(\mathbf{s})}\right) = \alpha_q + \mathbf{x}_q^\top(\mathbf{s})\boldsymbol{\beta}_q + \xi_q(\mathbf{s}),$$

where α_q is an intercept, $\mathbf{x}_q(\mathbf{s})$ are spatial covariates, $\boldsymbol{\beta}_q$ are regression coefficients and $\xi_q(\mathbf{s})$ is a GP. The covariates $\mathbf{x}_q(\mathbf{s})$ could for example be large near buildings in an earthquake scenario.

3 ONLINE LEARNING USING INLA

The Integrated Nested Laplace Approximation (INLA, Rue et al. (2009)) is a fast and memory efficient approximate Bayesian learning algorithm which we show can be successfully applied to our model for sequential learning under real-time constraints.

3.1 MODEL FORMULATION ON LATTICE

When learning the model, we assume that the domain $(0, x_{1,max}) \times (0, x_{2,max})$ is rectangular and has been split up into $a_1 \cdot a_2$ equally sized rectangles, each with area $\Delta = \frac{x_{1,max}x_{2,max}}{a_1 a_2}$. This approach was previously used for spatial point process models with INLA by Illian et al. (2012), and makes it clear that the parameter space of spatial models quickly becomes very high-dimensional since there is a parameter to be learned in each of the $a_1 \cdot a_2$ rectangles. Define n_{ij} as the number of detected persons in cell s_{ij} and m_{ij} as the number of detected injured persons in s_{ij}. Conditional on the latent fields, the joint distribution of the number of detected persons and injured in visited cell s_{ij} is given by

$$n_{ij}|z_\lambda, z_r \sim \mathsf{Poisson}\left(\Delta \exp\left(z_{\lambda,ij} + z_{r,ij}\right)\right)$$

$$m_{ij}|n_{ij}, z_q \sim \mathsf{Binomial}\left(n_{ij}, \frac{\exp\left(z_{q,ij}\right)}{1+\exp\left(z_{q,ij}\right)}\right)$$

where $z_{\lambda,ij}$, $z_{r,ij}$, and $z_{q,ij}$ are representative values of $Z_\lambda(s) \equiv \log\lambda(\mathbf{s})$, $Z_r(s) \equiv \log r(\mathbf{s})$ and $Z_q(s) \equiv \log(q(\mathbf{s})/(1-q(\mathbf{s})))$ in s_{ij}. Inference is simplified by observing that $\alpha_{q,ij}$ and $\boldsymbol{\beta}_q$ only enter the binomial injury model and that the log links in both the population and the detection models imply that

$$z_{\lambda,ij} + z_{r,ij} = \alpha_\lambda + \mathbf{x}_{\lambda,ij}^\top\boldsymbol{\beta}_\lambda + \mathbf{x}_{r,ij}^\top\boldsymbol{\beta}_r + \xi_{\lambda,ij}.$$

To avoid identification problems between $\boldsymbol{\beta}_\lambda$ and $\boldsymbol{\beta}_r$ we use different covariates in the population process $\mathbf{x}_\lambda$ and in the detection process $\mathbf{x}_r$.

3.2 INTEGRATED NESTED LAPLACE APPROXIMATION

Inference in models with high-dimensional spatial random fields is a challenging problem, and exact methods such as Markov Chain Monte Carlo (MCMC) are much too slow for real-time learning. Variational inference is a common method in machine learning for fast Bayesian inference, but is well known to underestimate posterior uncertainty, particularly in high-dimensional spatial problems, see e.g. Rue et al. (2009).

INLA (Rue et al., 2009) is a framework for fast accurate approximation of Bayesian posterior distributions in the class of latent Gaussian models. INLA is by now a standard method for spatial problems in the statistical literature, but is rarely used in Robotics. To describe the class of latent Gaussian models, let $\mathbf{z} \in \mathbb{R}^d$ be a (high-dimensional) vector of Gaussian variables with prior $\pi(\mathbf{z}|\boldsymbol{\theta}) = N(\mathbf{0}, Q(\boldsymbol{\theta})^{-1})$, where $Q(\boldsymbol{\theta})$ is a sparse precision (inverse covariance) matrix. In our case $\mathbf{z}$ contains the discretized Gaussian random fields $\xi_{\lambda,ij}$ and $\xi_{q,ij}$ as well as the fixed effects α_λ, $\boldsymbol{\beta}_\lambda$, $\boldsymbol{\beta}_r$, α_q and $\boldsymbol{\beta}_q$. The (low-dimensional) vector of hyperparameters $\boldsymbol{\theta}$ contains the variances and length-scales of the Matérn kernel functions for $\xi_{\lambda,ij}$ and $\xi_{q,ij}$. Further, let $\mathbf{y}$ here denote the n observations, in our case the set of all detected persons n_{ij} or injured persons m_{ij} in visited cells. INLA assumes these observations are independent conditional on the latent variables $\mathbf{z}$, with likelihood function

$$\pi(\mathbf{y}|\mathbf{z}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \pi(y_i|z_i, \boldsymbol{\theta}). \tag{1}$$

Our likelihood for the Poisson model for n_{ij}, and for the binomial model for $m_{i,j}$, are clearly both of the form (1).

INLA uses an intricate mix of several Laplace approximations for the high-dimensional $\mathbf{z}$ combined with numerical integration of the low-dimensional hyperparameters $\boldsymbol{\theta}$ to approximate the marginal posteriors of the latent variables z_i and the joint posterior of the hyperparameters $\boldsymbol{\theta}$. The basic INLA approximation is of the form

$$\pi(z_i|\mathbf{y}) \approx \int \tilde{\pi}(z_i|\boldsymbol{\theta}, \mathbf{y})\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}, \tag{2}$$

where $\tilde{\pi}(z_i|\boldsymbol{\theta}, \mathbf{y})$ is obtained by marginalizing a Laplace approximation (Tierney and Kadane, 1986) of $\pi(\mathbf{z}|\boldsymbol{\theta}, \mathbf{y})$ and

$$\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y}) = \left.\frac{\pi(\mathbf{z}, \boldsymbol{\theta}, \mathbf{y})}{\tilde{\pi}_G(\mathbf{z}|\boldsymbol{\theta}, \mathbf{y})}\right|_{\mathbf{z}=\mathbf{z}^\star(\boldsymbol{\theta})},$$

where $\tilde{\pi}_G(\mathbf{z}|\boldsymbol{\theta}, \mathbf{y})$ is a Gaussian approximation to the full conditional posterior of $\mathbf{z}$ and $\mathbf{z}^\star$ is the mode of $\mathbf{z}$ for a given $\boldsymbol{\theta}$. The integral in (2) is performed numerically by summing over a set of carefully selected support points in $\boldsymbol{\theta}$, see Rue et al. (2009) for details. Finally, we can produce predictions $\mathbb{E}[y_i^*|\mathbf{y}]$ by integrating over the approximation $\pi(z_i|\mathbf{y})$, where y_i^* in this case may be e.g. the number of injured m_{ij}^* in non-visited cells.

The use of *nested* Laplace approximations makes INLA extremely accurate for latent Gaussian models, see Rue et al. (2009) and Teng et al. (2017) for some evidence for LGCP models. In particular, INLA has been shown to be much more accurate than variational approximations.

INLA is typically orders of magnitude faster than Markov Chain Monte Carlo (MCMC) and Hamiltonian Monte Carlo (HMC) (Rue et al., 2009; Teng et al., 2017), and can therefore be successfully applied in a real-time context. By exploiting the sparsity of the precision matrix $Q(\boldsymbol{\theta})$ that results from conditional independencies that appear naturally in spatial problems and efficient reordering schemes (Rue and Held, 2005), INLA scales favorably as $O(d^{3/2})$ in 2D, where d is the total number of cells where the fields are evaluated. Moreover, since our focus here is on $\pi(z_i|\mathbf{y})$ rather than hyperparameter inference per se, we will use the Empirical Bayes (EB) to optimize wrt $\boldsymbol{\theta}$. This gives additional speed-ups since we can also benefit from a warm start with excellent initial values from the previous search iteration, followed by a fast update of the posterior $\pi(z_i|\mathbf{y}, \hat{\boldsymbol{\theta}}_{EB})$.

The core of INLA is implemented in C++ with the convenient r-inla interface to the statistical programming language R, see Rue et al. (2017) for details.

4 PLANNING EXPLORATORY MOVES

Finding the best sequence of exploratory moves is a computationally hard problem. Simple coverage patterns (Goodrich et al., 2008; Rudol and Doherty, 2008; Xu et al., 2011), or heuristics (Waharte and Trigoni, 2010) are common in the literature.

As we employ a sophisticated probabilistic model, it is natural to view the decision problem as a partially observable Markov decision process, or POMDP (Åström, 1965). Only the visited regions are observed, which we use to infer a belief over the unobserved cells using the spatial point process.

Formally, we define the search state of our problem to be $\boldsymbol{\phi}_t = (\mathrm{Bel}_t(\mathbf{M}), \mathbf{p}_t)$. The matrix $\mathbf{M} = (m_{ij})$ consists of the number of injured in each of the Δ-sized cells. We use the notation $\mathrm{Bel}_t(\mathbf{M})$ for the posterior distribution of $\mathbf{M}$ at time t. As $\mathrm{Bel}_t(\mathbf{M})$ are probability distributions, this is technically a belief-augmented MDP formulation (c.f. Thrun et al. (2005)). We also define $\mathbf{p} \in \{s_{ij} : \forall\,(i,j)\}$ as the position of the UAV in the cell grid, which can be considered known by GPS at the length-scales of our cell size. The UAV has to sequentially decide on which cell to explore next, in the form of actions $a_t \in \mathcal{A} = \{s_{ij} : \forall\,(i,j)\}$ that map to exploration of cell s_{ij}. Cell exploration takes time $T_{ij} = \mathbf{x}_{T,ij}^\top \boldsymbol{\beta}_T$, where $\mathbf{x}_{T,ij}^\top$ are spatial covariates for terrain type which may overlap with those used in the detection model, and $\boldsymbol{\beta}_T$ are user-supplied estimates of their respective exploration time. Each exploratory action updates the UAV position $\mathbf{p}_{t_{i+1}}$ and its belief over injured in the map $\mathrm{Bel}_{t_{i+1}}(\mathbf{M})$.

However, the UAV is unable to teleport between cells, it additionally takes time $T_f \cdot \mathrm{dist}(\mathbf{p}, s_{ij})$ to reach a non-adjacent cell s_{ij}. Clearly, a sequential exploration of adjacent cells takes less time but may not discover the most injured, resulting in a difficult trade-off.

Finally, we want to solve the optimal exploration problem

$$\arg\min_{\pi(\boldsymbol{\phi})} \; \mathbb{E}_{\boldsymbol{\tau}|\pi(\boldsymbol{\phi})}[c(\boldsymbol{\tau})],$$

where $c(\boldsymbol{\tau})$ is a cost function and $\boldsymbol{\tau} = \{\boldsymbol{\phi}_{t_i} ... \boldsymbol{\phi}_{t_N}\}$ is the trajectory through the belief-augmented state space $\boldsymbol{\phi}_t$, from current time t_i until all cells have been explored at time t_N. The trajectory is uncertain as the injuries $\mathbf{M}$ are partially unknown at decision time. By taking action $a_t = \pi(\boldsymbol{\phi}_t)$, data $o_t = (n_{ij}, m_{ij})$ will be observed and beliefs will be updated by $\mathrm{Bel}_{t_{i+1}}(\mathbf{M}) = f(\mathrm{Bel}_{t_i}(\mathbf{M}), a_{t_i}, o_{t_i})$, where the transition function f updates the posterior of the injured in non-visited cells according to the spatial point process model.

While maximizing information is popular in POMDP formulations of search (c.f. Morere et al. (2017); Waharte and Trigoni (2010)), not all information is equally useful in our structured model, e.g. one may have large uncertainty but low expectation in sparsely populated areas. At each time t_i we instead minimize the total remaining harm to victims,

$$c(\boldsymbol{\tau}) = \int_{t_i}^{t_N} \sum_{ij} m_{ij}^*(t) h(t)\, dt,$$

where $m_{ij}^*(t)$ is the number of *unexplored* injured in cell s_{ij} at time t, and $h(t)$ is the rate of harm. This is integrated with the trapezoid rule over the varying durations of the discrete sequence of actions. For convenience and without loss of generality, in the following we assume the rate of harm (e.g. mortality rate) is constant while undiscovered, $h(t) \propto 1$.

Unfortunately, the complexity of solving POMDPs is doubly-exponential (Thrun et al., 2005). Even with the search discretized into a grid and a choice of four adjacent cells to explore next, this problem quickly grows infeasible for real-world sizes.

Monte-Carlo tree search (MCTS) is an approximate solution to discrete sequential decision making problems. In its purest form, MCTS is a tree search algorithm that treats the problem of finding good branches (actions) as a sequence of bandit problems solved by the UCB (Upper

Confidence Bounds) algorithm. The resulting UCT algorithm (Kocsis and Szepesvári, 2006) effectively treats finding good plans as an exploration problem in itself. By preferentially sampling from promising branches, the effective branching factor can be significantly reduced.

While most well-known for its successes in the game of Go (Browne et al., 2012; Silver et al., 2017), MCTS has recently found uses in other applications. Morere et al. (2017) used MCTS to plan belief trajectories in a POMDP for environment monitoring. However, POMDP planning is still very expensive, and they could only afford to plan three steps ahead. The reason is that for each tested action in the search tree, uncertain outcomes have to be sampled, and new beliefs inferred.

To create an effective MCTS algorithm for our search domain we employ three modifications, i) incorporating action-abstractions, a small set of long-range moves $\mathcal{A}_{\max}$ directly to the cell s_{ij} with the maximum posterior expected number of unexplored injured, ii) a certainty equivalence assumption where random variables are replaced with their expected values, and iii) using receding-horizon planning with warm-starts and a fast domain-specific cost-to-go approximation. The posterior expected number of unexplored injured $\mathbb{E}\left[m^*_{ij}|o_{t_1},\dots,o_{t_i}\right]$ are computed based on the spatial point process model as described in section 3.

We test two variants. The first, simply called `MCTS`, can only move through adjacent squares $\mathcal{A}_{\texttt{MCTS}} = \{s_{ij} : \text{adj}(\mathbf{p}, s_{ij})\}$, to explore, or if explored, fly through. The second, denoted `MCTSJump`, additionally includes the long-range moves from i) in the first time step t_i of each plan, i.e. $\mathcal{A}_{\texttt{MCTSJump},t_i} = \mathcal{A}_{\max} \cup \mathcal{A}_{\texttt{MCTS}}$. An example search pattern from `MCTSJump` can be seen in Figure 2.

These action-abstractions ("jumps") can be seen as a type of option policies (Stolle and Precup, 2002; Subramanian et al., 2016) in reinforcement learning. Adding more actions always increases the branching factor, which results in a difficult trade-off. We found that using the cells with top ten expected number of injured resulted in a considerable performance increase over using a random or equally spaced subset.

The certainty equivalence assumption $\text{Bel}_{t_{i+1}}(\mathbf{M}) \approx f(\delta_{\mathbb{E}(\mathbf{M})}(\mathbf{M}), a_{t_i}, o_{t_i})$, where $\delta_a(\cdot)$ is the Dirac spike at a, allows us to forego sampling from the outcomes to update the belief model, which is the main bottleneck of planning in POMDPs. A similar assumption was used by Baker et al. (2016). This is a strong assumption, because plans are *evaluated* on the premise that the future is predictable, which means it will not value recourse, the possibility to later change the plan if it turns out worse than expected. However, it still replans at each step, and due to the spatial correlation in our problem, a poor outcome also impacts adjacent cells, limiting recourse.

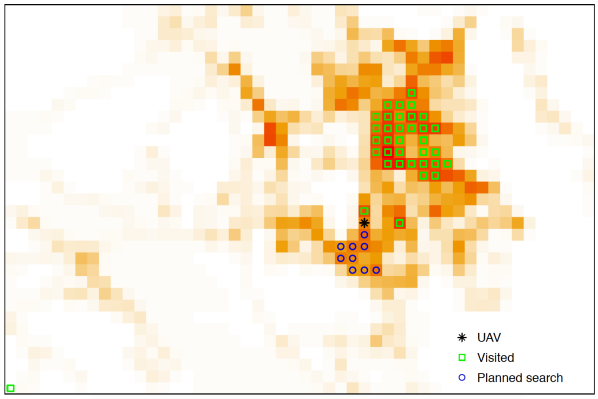

Figure 2: Search scenario using `MCTSJump`, overlaid on heatmap for expected number of detectable injured.

Finally, the receding-horizon formulation from iii) is standard in control, where it is sometimes known as model-predictive control. By cutting the planning horizon from t_N to t_H and adding a domain-specific approximation of the remaining cost $c(\boldsymbol{\tau}) = c(\boldsymbol{\tau}_{t_i..t_H}) + \hat{c}_{t_H..t_N}(\boldsymbol{\phi}_{t_H})$, computation can be greatly decreased. Warm-starts from the previously best plan at t_{i-1} also allows computation to be amortized over several iterations. Here we just assume the cost-to-go decreases linearly to zero as for an ideal lawnmower pattern. This allows comparison of plans of different duration, such as jump moves from i).

As MCTS is an any-time algorithm we give it a fixed 3 second compute budget. It is implemented in C++ and evaluates about 100 000 plans. As it does not assume any fixed search pattern or observation order, it also allows human operators the flexibility to take control if needed.

5 EXPERIMENTS

Here we test the proposed real-time probabilistic SAR modeling and MCTS exploration algorithm. We use real-world GIS data from the Swedish government, which offer easy access to a wealth of data (Lantmäteriet, 2019). We selected a 4.0x2.7km area around the town Gamleby seen in Figure 1. It contains a variety of terrain and is within a proposed UAV test zone, where we may be permitted to test the algorithm with real UAVs.

Data was simulated using the structural spatial point process model in section 2, discretized to 50x33 lattice cells for search and model inference, each about the size of a soccer field. We assume the UAV can fly at a speed of 10m/s for long-range moves. The cell explore times

Table 1: Scenario Settings. Covariates and spatial fields in data generating process and inferred model. B=buildings, R=roads, W=water, F=forest, G_i=Gaussian no i, S=spatial field. Deviations from truth in red.

		Scenario A	Scenario B	Scenario C	Scenario D
Truth	Population	B	$B + R + W + S$	$B + R + W + S$	$B + R + W + S$
	Detection	–	F	F	F
	Injury	–	$B + S$	G_1	$G_2 + G_3$
Model	Population	S	$B + R + W + S$	$B + R + W + S$	$B + R + W + S$
	Detection	–	F	F	F
	Injury	–	$B + S$	$G_1 + S$	$G_2 + S$

Table 2: Time until half of injured have been found (minutes)

	Scenario A	Scenario B	Scenario C	Scenario D
Lawnmower coverage pattern	835	1126	164	271
Globally greedy (on prior)	2287	167	102	279
MCTS (on prior)	572	120	88	171
MCTS (learning)	273	108	88	121
MCTSJump (learning)	**241**	**87**	**69**	**100**

are set to $\boldsymbol{\beta_T} = (1, 2, 0.5, 0.5, 0.75)$ minutes for terrain covariates "buildings", "forest", "road", "field", and "water" respectively. This reflects a fast overhead search with extra cost for difficult terrain such as forests needing multiple angles. The detection covariate for forest was similarly set lower. This rapid search pace also highlights the importance of real-time performance, and why we capped MCTS to 3 seconds. For reference, the entire inference and planning loop in our prototype implementation takes about 5 seconds on a Core i7 CPU, imposing minimal overhead on the search.

To the best of our knowledge, this is the first principled attempt at probabilistic modeling of the full structure of the disaster response search problem, in real-time or otherwise. We show the benefit of real-time learning on a range of scenarios that map to different real-world disasters. On the planning side, most principled optimization-based approaches typically scale poorly (Waharte and Trigoni, 2010; Morere et al., 2017) to the large scenarios we envision. Simple maximum coverage approaches (Goodrich et al., 2008; Rudol and Doherty, 2008; Xu et al., 2011) seem most common in the literature, which we include as baseline. As our mission area is rectangular, a simple baseline coverage algorithm is a "lawnmower" pattern. In terms of covering the largest area in the shortest time, this is optimal. As heuristics were suggested in e.g. Waharte and Trigoni (2010), we also compare against a globally greedy strategy that always selects the cell with highest expected number of injured. Finally, MCTS has been proposed for search planning in Baker et al. (2016). While they did not list computation time, their determinization approach appear similar to our basic MCTS variant without learning, which we use as the final baseline.

In the following we test four scenarios reflecting different types of real-world disasters and the level of prior information available. A summary of the scenarios, the covariates used in the data generating process, as well as those used for the inference models, can be seen in Table 1. Each scenario is replicated 15 times from different seeds, except for the first one, which used 30. In all cases, we attempted to use reasonable values for the parameters in the data generating process, the fixed effect parameters $\{\alpha_\lambda, \boldsymbol{\beta}_\lambda, \boldsymbol{\beta}_r, \alpha_q, \boldsymbol{\beta}_q\}$, and the GP hyperparameters $\{\boldsymbol{\theta}_\lambda, \boldsymbol{\theta}_q\}$, by sampling realizations of the spatial point process and comparing to real-world expectations. When the same covariates were also used for inference, we used normal priors for the fixed effects, centered around the true value but with fairly high variance. We also tested the robustness of our methods to various inference priors, and found only minor differences in the results, see the supplementary material. For the GP hyperparameters, we used rather non-informative Gamma priors for the range and marginal precision of the Matérn fields, using the parameterization of Lindgren et al. (2011).

The results are shown in Table 2. In summary, leveraging real-time learning with the proposed model and `MCTSJump` planning outperforms all baselines ($p < 0.05$ t-test). Learning is especially crucial when prior knowledge is lacking or partial, as seen in Scenario A and D. Interestingly, tree-search algorithms like `MCTS` subsume locally-greedy algorithms if planning horizon is taken to zero, and should be a dominating strategy as the planning horizon is increased. Further, our pro-

posed MCTSJump algorithm will also always include the globally-greedy choice in its list of action abstractions, and can therefore be seen as a principled way of solving this trade-off as an optimization problem, rather than as a heuristic needing tuning to a scenario.

5.1 SCENARIO A: NO PRIOR INFORMATION

In this scenario we assume the model does not have access to any useful spatial covariates. While in practice some information tends to be available, this was designed to test the capability of the model to fall back to the spatial fields, to cover for unexpected situations.

The ground truth is a population distribution drawn from GIS building covariates not available to the agent, see Table 1. For simplicity we ignore the injury part of the model and focus only on maximizing the number of people found in this scenario. As can be seen from the results in the top row of Figure 3 and Table 2, our model with MCTS and MCTSJump significantly outperforms other strategies. Just relying on the spatial field was sufficient to capture the natural clustering in population data. In this case however, the improvements offered by long-range moves ("jumps") was not statistically significant, which is not surprising considering the spatial correlation captured by the field only gives local information.

5.2 SCENARIO B: EARTHQUAKE

Here we simulate a classical earthquake scenario. We generate population using all five GIS covariates, as well as a spatial field. As this is an earthquake, both the building covariate and a spatial field was used to draw realizations of injured people. We use the same structure of the model for inference, and reasonably uninformative priors. Figure 3 and Table 2 show that by drawing on the GIS covariates, even corrupted by a spatial field, our algorithm is significantly faster than in A, faster than competing strategies, and outperforms lawnmower patterns by a wide margin. This also showcases the advantage of MCTSJump, which outperformed regular MCTS by drawing on the injury covariate to make informed jumps directly to the urban areas. It also uses jumps to complete the map, while local MCTS can miss some areas.

5.3 SCENARIO C: TERRORIST ATTACK - KNOWN SITE

In this scenario there has been a localized terrorist attack, represented by a concentration in the injury field southwest of town. We show that using the proposed model, this can easily be encoded on the fly by first-responders, via e.g. a Gaussian shaped covariate centered on the reported site. Figure 3 shows similar but even more targeted behavior than in Scenario B.

5.4 SCENARIO D: TERRORIST ATTACK - ONE SITE UNKNOWN

Finally, we showcase all the capabilities of the structured model by extending Scenario C. In this case, there is a terrorist attack with one site encoded by a Gaussian shaped covariate. However, early information during catastrophes is often incomplete. In this case there is also a second attack site unknown to us. The results show the model quickly picks up on this. In particular, MCTSJump flies to and explores the a priori known site, then without further information will jump around and scout high population areas. At some point it stumbles on injured near the second site, the spatial field quickly learns the local anomaly in injury probability, and the planner focuses on that area. An example of this is shown in the supplementary video material[1] and Figure 4.

6 CONCLUSIONS

We present a new probabilistic framework for victim search during a disaster response based on real-time learning and decision making with a structural spatial point process. The model is built from spatially referenced components on which there is usually ample prior information: i) the distribution of persons, ii) the probability of detecting a person, and iii) the probability of injury. Learning spatial processes and acting on them in real-time is a hard problem. We propose a novel combination of approximate Bayesian learning using INLA and an MCTS strategy adapted to the search problem.

We assess the empirical performance on simulated scenarios on a real map with publicly available GIS data, displaying that both prior information and learning can be efficiently used by our model to outperform related strategies. We also show that the spatial fields can fill in for missing prior information in a very adaptable manner.

The framework proposed here can be extended in many interesting directions, for example to dynamic problems where the intrinsic state variables evolve over time, such as disasters involving gas leakage or rescue operations at sea. While this work indicates that just cleverly using one UAV can make a large difference, in future work we also intend to extend it to search with a team of real UAVs.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) and the WASP Autonomous Research Arenas funded by the Knut and Alice Wallenberg Foundation, as well as the Swedish Foundation for Strategic Research (SSF) project Symbicloud, and the ELLIIT Excellence Center at Linköping-Lund for Information Technology.

[1] https://youtu.be/wyD0O5hF5tE

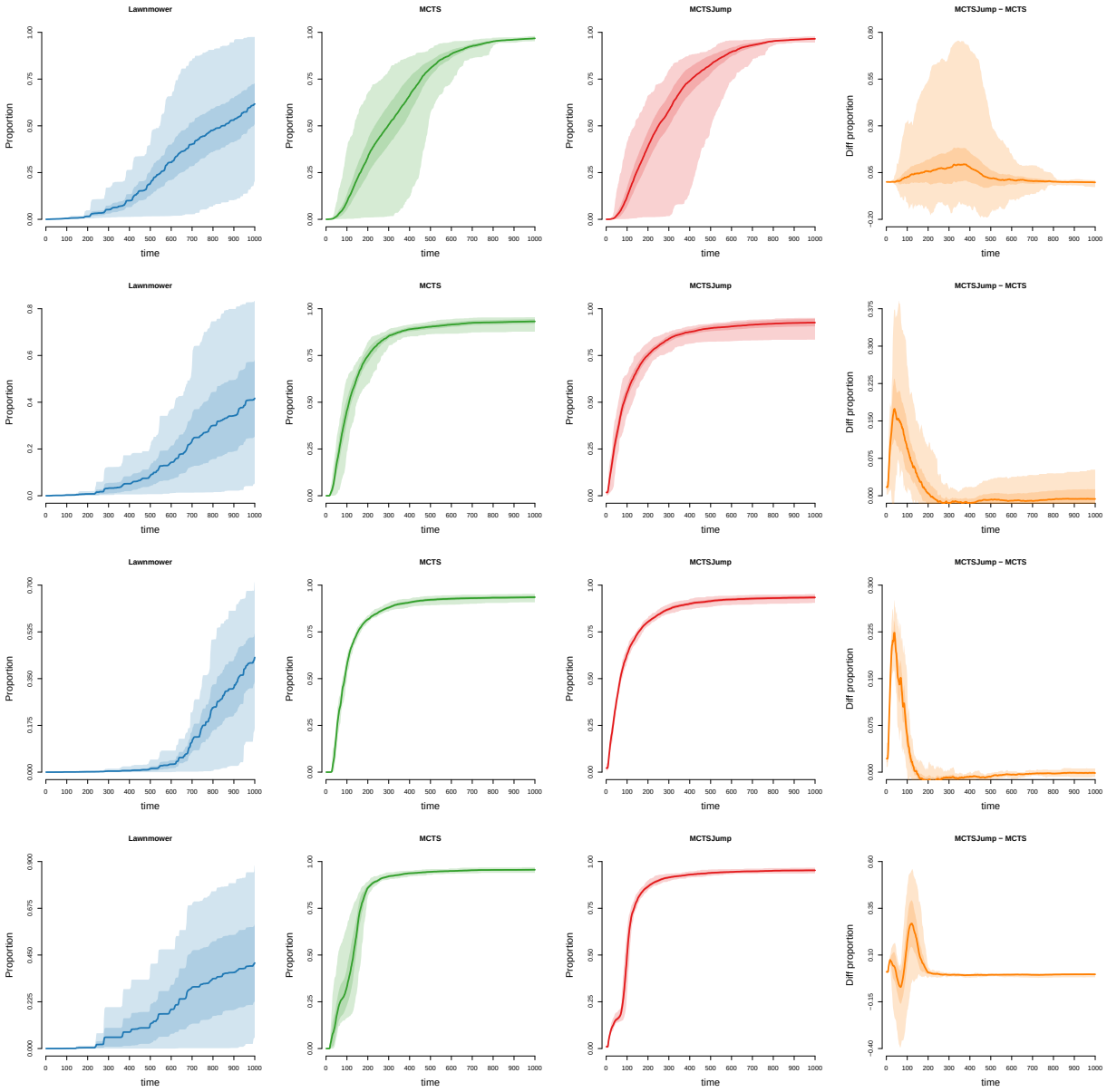

Figure 3: Comparing the proportion of injured found as a function of search time (minutes) for the different strategies (lawnmower, `MCTS`, and `MCTSJump`). The rows correspond to each of the four scenarios A to D (top down). The graphs show the mean proportion of injured found as a function of search time (solid line) for the three strategies over 30 replicates, as well as the 95% confidence bands for the mean (darker regions) and 95% predictive bands for individual proportions in individual replicated datasets (lighter regions). The final column shows that same properties, but for the differences in proportions between MCTS and MCTSjump.

References

Åström, K. (1965). Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174 – 205.

Baker, C., Ramchurn, G., Teacy, L., and Jennings, N. (2016). Factored Monte-Carlo tree search for coordinating UAVs in disaster response. In *ICAPS Proceedings of the 4th Workshop on Distributed and Multi-Agent Planning (DMAP-2016)*.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Goodrich, M. A., Morse, B. S., Gerhardt, D., Cooper, J. L., Quigley, M., Adams, J. A., and Humphrey, C. (2008). Supporting wilderness search and rescue using a camera-equipped mini UAV. *Journal of Field Robotics*, 25(1-2):89–110.

Illian, J. B., Sørbye, S. H., and Rue, H. (2012). A toolbox for fitting complex spatial point process models using integrated nested Laplace approximation (INLA). *The Annals of Applied Statistics*, pages 1499–1530.

Kocsis, L. and Szepesvári, C. (2006). Bandit based Monte-Carlo planning. In *European conference on machine learning*, pages 282–293. Springer.

Kratzke, T. M., Stone, L. D., and Frost, J. R. (2010). Search and rescue optimal planning system. In *2010 13th International Conference on Information Fusion*, pages 1–8. IEEE.

Lantmäteriet (2019). Lantmäteriet open geodata. https://lantmateriet.se/en/maps-and-geographic-information/open-geodata/. Accessed: 2019-03-08.

Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The SPDE approach. *Journal of the Royal Statistical Society Series B*, 73(4):423–498.

Møller, J., Syversveen, A. R., and Waagepetersen, R. P. (1998). Log Gaussian Cox processes. *Scandinavian Journal of Statistics*, 25(3):451–482.

Morere, P., Marchant, R., and Ramos, F. (2017). Sequential Bayesian optimization as a POMDP for environment monitoring with UAVs. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6381–6388. IEEE.

Murphy, R. R., Tadokoro, S., Nardi, D., Jacoff, A., Fiorini, P., Choset, H., and Erkmen, A. M. (2008). Search and rescue robotics. In Siciliano, B. and Khatib, O., editors, *Springer Handbook of Robotics*, pages 1151–1173. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*. MIT Press Cambridge, MA.

Rudol, P. and Doherty, P. (2008). Human body detection and geolocalization for UAV search and rescue missions using color and thermal imagery. In *2008 IEEE aerospace conference*, pages 1–8. Ieee.

Rue, H. and Held, L. (2005). *Gaussian Markov Random Fields: Theory and Applications*. Chapman and Hall/CRC.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society: Series B*, 71(2):319–392.

Rue, H., Riebler, A., Sørbye, S. H., Illian, J. B., Simpson, D. P., and Lindgren, F. K. (2017). Bayesian computing with INLA: a review. *Annual Review of Statistics and Its Application*, 4:395–421.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature*, 550(7676):354.

Singh, A., Krause, A., Guestrin, C., and Kaiser, W. J. (2009). Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755.

Stolle, M. and Precup, D. (2002). Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, pages 212–223. Springer.

Subramanian, K., Scholz, J., Isbell, C. L., and Thomaz, A. L. (2016). Efficient exploration in Monte Carlo tree search using human action abstractions. In *FILM Workshop at NIPS 2016*. NIPS.

Teng, M., Nathoo, F., and Johnson, T. D. (2017). Bayesian computation for log-Gaussian Cox processes: a comparative analysis of methods. *Journal of Statistical Computation and Simulation*, 87(11):2227–2252.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic robotics*. MIT press.

Tierney, L. and Kadane, J. B. (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86.

Waharte, S. and Trigoni, N. (2010). Supporting search and rescue operations with UAVs. In *2010 Interna-*

tional Conference on Emerging Security Technologies, pages 142–147. IEEE.

Xu, A., Viriyasuthee, C., and Rekleitis, I. (2011). Optimal complete terrain coverage using an unmanned aerial vehicle. In *2011 IEEE International conference on robotics and automation*, pages 2513–2519. IEEE.