Automated Planning and Policy Learning for Surveillance Missions[*]

Sara Bernardini and **Maria Fox** and **Derek Long**
Department of Informatics
King's College London, UK
firstname.lastname@kcl.ac.uk

Abstract

The *Search And Tracking* (SAT) problem is concerned with enabling an autonomous UAV to search for a mobile target and tracking it after it is found. Since state-of-the-art probabilistic approaches to SAT suffer high computational cost that makes them unsuitable for complex problems, we propose an alternative technique based on *automated planning*: we track the target reactively while it is in view and plan a recovery strategy that relocates the target every time it is lost by using a high-performing planning tool. In this paper, we first place our work on SAT in the context of our long-term research goal: considering SAT as a case study of a broader range of surveillance missions characterised by high uncertainty and restricted resources, our approach to generating plans for a single UAV marks a first step towards coordinating multiple intelligent vehicles engaged in complex surveillance tasks through *plan-based policy learning*. We then describe our approach to SAT in detail and present a simulation that validates its potential. Finally, we discuss extensions to modern planners required to enable them to reason about SAT tasks, such as the ability to model obstacles within their physical environment and to handle complex spatial and geometrical constraints.

1 Introduction

Over the past ten years there has been growing interest in using autonomous agents for surveillance applications. Surveillance problems are characterised by two kinds of agents: observers and targets. Observers might be mobile (e.g., UAVs), or fixed (e.g., CCTV cameras). Targets might be aware that they are being observed, and possibly evasive, or not. Examples of surveillance problems are: (i) *Search And Tracking*, which is the problem of searching for mobile targets and tracking them after they are found; (ii) *Intelligence Gathering*, in which observers are mobile and the targets correspond to interesting sites to be found, recorded and communicated; and (iii) *Hazard Identification*, in which observers are a mixture of mobile and fixed and targets are physical flaws in components being observed, or environmental readings that exceed safety levels.

At the heart of surveillance problems lies the need to plan complex sequences of behaviours that achieve surveillance goals, which are typically expressed in terms of gathering as much information as possible given the constraints, and communicating findings to a human operator. However, observers operate in situations in which there is very little stability, information is changing rapidly and decisions about what action to perform and how to coordinate with other observers must be made almost instantaneously. To be effective in such situations, observers must be so highly trained that they can react without spending time reasoning about alternative courses of action. Hence, surveillance problems give rise to many challenges including the management of uncertainty in an unpredictable environment, the handling of restricted resources and the communication of commitments and requests between multiple heterogeneous observers.

Although forward planning is certainly required in order to avoid the observers behaving in a purely reactive (and therefore easily distracted) manner, planning problems involving metric resources, continuous time and concurrency, as would be required in the solution of non-trivial surveillance problems, are time-consuming to solve. This complexity is greatly exacerbated if uncertainty is captured explicitly within the planning domain models. Online planning, and plan repair in the case of failure, are feasible in stable situations, but they take too long in situations that are changing rapidly. Online planning also requires significant on-board computational resources, which are often not available in surveillance vehicles. Planning under uncertainty cannot therefore be done online in situations typical of many surveillance problems, where computational resources are limited and rapid responses are frequently required.

Since online planning and planning under uncertainty are both unrealistic for large-scale, fast-moving surveillance problems, we propose to follow an alternative approach based on *plan-based policy learning* via Monte Carlo sampling. Under the assumption that time and resources are available offline to train effective policies, we can sample many instances of the stochastic problem, each instance being a challenging temporal and metric planning problem. We can then solve each instance using a high-performing planner, and apply a classifier to learn a policy as a mapping from states to actions, using the set of solutions as input. Hence, instead of on-board planning, we propose offline

[*]The authors gratefully acknowledge support from EPSRC grant EP/J012157/1 and from BAE Systems Ltd.

planning and policy-learning and on-board policy execution. In this context, a *policy* can be seen as a small piece of software that compiles a vast planning experience, requires negligible computation to execute and can equip agents with the means of making intelligent on-the-fly decisions. The effectiveness of this approach has been already demonstrated in two single-agent cases: management of the loading of multiple batteries (Fox, Long, & Magazzeni 2011), and the control of an autonomous underwater vehicle following the edge of a patch in the coastal waters of the Monterey Bay (Fox, Long, & Magazzeni 2012) (see Section 7 for details). The resulting policies are very high-performing in terms of robustness to the high degree of uncertainty that often occurs in the physical execution environment. The high quality of the policies is a result of the investment in offline planning. Furthermore, if a metric, temporal and continuous planner is used in this offline training phase, the obtained policies are much more expressive than those that can be learned using other techniques in the literature.

In order to scale up the approach taken in the batteries and patch-following cases to the management of multi-target and multi-observer surveillance tasks, we are faced with three significant challenges:

1. *Planning:* at the heart of the policy-learning strategy is the ability to formulate surveillance tasks as planning problems and plan high quality solutions to the deterministic sampled instances of such problems. In the scenarios considered above, a planner capable of reasoning with suitably rich models involving temporal, spatial and continuous constraints is required.
2. *Policy learning:* after generating and solving thousands of planning instances, a classification process is used to obtain a policy from these solutions. Although we plan to use off-the-shelf classifiers for this task (e.g. the WEKA toolset (Hall *et al.* 2009)), a good set of features needs to be identified in order to obtain high-quality and robust policies. This is usually a time-consuming task as identifying a good set of state variables is one of the most challenging technical aspects of learning policies.
3. *Policy integration and switching:* since the different observers need to coordinate their actions, possibly with the help of a human operator, a mechanism to integrate policies is required. In addition, if a complex surveillance task is divided into several stand-alone tasks, the best results might be obtained by learning policies for each sub-task, with the observer switching between policies as it moves between sub-tasks.

In this paper, we give an high-level overview of our approach to address the first challenge, planning, and use SAT missions as a case study for surveillance applications. We choose SAT among the possible surveillance applications because the ability to search for and track a moving target is common to a number of autonomous UAV missions. The interested reader can find a full technical account of our approach to SAT in (Bernardini *et al.* 2013). In addition, we discuss the other two challenges, policy-learning and policy-integration, which are the goals of our future work.

The paper is organised as follows. In Sections 2, we present the SAT problem. In Section 3, we examine as a SAT task decomposes into separate phases and explore the role of planning amongst them. We identify a planning problem that arises in this mission and show how it can be modelled and solved using generic planning technology. In Sections 4, we present a simulation that we developed for demonstrating the potential of our technique and the results that we obtained. In Section 5, we discuss possible extensions to our current approach to SAT. Section 6 presents how we intend to carry out policy learning and integration in detail. Finally, Sections 7 and 8 present related work and conclusions.

2 Search and Tracking Missions

As explained in detail in (Bernardini *et al.* 2013), we consider SAT missions in which the target is a vehicle being sought and tracked through a mixed urban, suburban and rural landscape and the observer is a single fixed-wing UAV. The objective of the mission is to follow the target to its destination. A possible interpretation of this scenario is that a police drone is tracking the car of a suspected criminal target that is trying to reach a certain destination.

We make the following assumptions on the SAT mission: (i) The low-level control of the UAV is managed by subsystems underpinning the abstract deliberation level of our plans and policies. (ii) The UAV is equipped with imaging systems to observe the target. Observation is susceptible to error and interference from terrain (in urban, suburban, forested and mountainous areas the probability of spotting the target is reduced). (iii) The imager has two modes: wide-angle mode ($180°$) used to increase the area being scanned at the cost of a lower probability of successfully observing the target; and narrow-angle mode ($90°$) in which the viewing area is reduced, but the probability of detecting the target is higher. (iv) A faster moving target in the viewing zone is considered easier to spot. (v) The target follows the road networks and does not perform significant evasive actions.

In general, a SAT mission proceeds in two phases, which interleave until the target stops or until the observer acknowledges it has lost the target irrevocably:

- *Tracking:* the UAV flies in a standard pattern over the target, observing its progress; and
- *Search:* the UAV has lost the target and flies a series of manoeuvres intended to rediscover the target.

Once the target is rediscovered, the UAV switches back to tracking mode.

Tracking is managed by a *reactive controller*: the problem is simply one of matching the flight path to the target path. In general, UAVs fly much faster than road vehicles drive, but fixed-wing UAVs cannot hover. Therefore, the flight path of a fixed-wing UAV in tracking mode is a circle of fixed radius centred on the target. The radius cannot be greater than the range of the imaging equipment and cannot be shorter than the turning radius of the UAV. We assume that the UAV flies in a mid-range circle between these extremes. As the target moves the circle moves with it, so the flight path of the UAV describes a spiralling pattern over the ground.

If the UAV fails to observe the target, it must attempt to rediscover it. How this is achieved depends on the time since the observer last observed the target. For a short period after

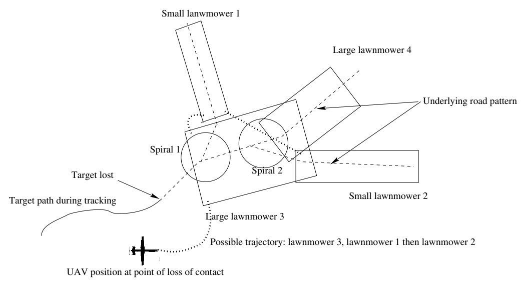
Small lanwmower 1
Large lawnmower 4
Underlying road pattern
Spiral 1
Spiral 2
Target lost
Small lawnmower 2
Target path during tracking
Large lawnmower 3
Possible trajectory: lawnmower 3, lawnmower 1 then lawnmower 2
UAV position at point of loss of contact

Figure 1: Initial state and search plan for the SAT mission.

losing the target, the UAV continues to track at low confidence, i.e. it assumes that the target is still progressing along the same route, and it follows this route for a time in the hope of picking up the target at a later point. However, after a period, it will be necessary to make a more systematic effort to rediscover the target by directing the search into specific places. At this point, the following information is available to the UAV: the current state includes information about the last known location and velocity of the target, the average velocity over the period the target has been tracked, the map of the terrain and the current position of the UAV. Based on this information, it is possible to formulate the task of rediscovering the target as a *planning problem* (see Figure 1): a search plan is constructed from a set of candidate search patterns that can be arranged in a sequence to attempt to optimise the likelihood of rediscovering the target. If, while flying this search plan, the target is rediscovered, the observer switches back to tracking mode.

3 Search as Planning

We propose that the UAV exploits standard *search patterns* to try to find the target. We use these patterns as the building blocks for a search plan that attempts to maximise the expectation of rediscovering the target. In particular, we consider two standard search patterns:

- *Spiral pattern:* we use spirals for covering areas of high density road network, e.g. urban or suburban terrain.
- *Lawnmower pattern:* we use rectangular lawnmowers for covering more elongated areas corresponding to a major road and including some possible side roads.

The challenge is to decide how to distribute these search patterns over the terrain in order to find the target. This selection of search patterns can be seen as a *planning problem*: if each search pattern is assigned a value corresponding to the expectation of finding the target in a search of that area, the planner can choose the sequence of search patterns that maximises the accumulated expectation of rediscovery. Note that the value associated with a search pattern is a function of time, since the target will not be found in an area that is far from its last known location until sufficient time has passed for the target to have reached the area, while it is unlikely that the target will be found in the area once sufficient time has passed for it to have driven through the area.

We exploit the period in which the UAV tracks the predicted location of the target to perform planning. Once the plan has been formulated and dispatched, the UAV executes the search patterns following the sequence specified by the plan, linking patterns together with a series of flight paths. As soon as the target is found, the plan is abandoned and the target is tracked. Replanning occurs if the target is lost again at a later time. So, in our approach, the plan is relevant until the target remains undiscovered, and somewhat counter-intuitively, "plan-failure" corresponds to the situation in which the target is found. Despite the inherent uncertainty in the situation, the problem is deterministic, since the plan is constructed entirely under the assumption that the target remains undiscovered.

3.1 Planning Domain

The domain model for the search problem has a very simple structure: there is a flight action that allows the UAV to fly from one waypoint to another and there are actions allowing the UAV to perform each of the basic search patterns. We use spiral searches and small and large lawnmowers, although we are currently adding new patterns to handle obstacles in the physical environment (see Section 5). The search pattern actions all have similar forms: they have an entry and an exit waypoint and the effect is to move the UAV and to increase the reward, which is the accumulated expectation of finding the target. The actions are durative and their duration is fixed in the problem instance to be the correct (computed) value for the execution of the corresponding search. The search patterns can only be executed so that they coincide with a period during which the target could plausibly be in the area the pattern covers. This is calculated by considering the minimum and maximum reasonable speeds for the target and the distance from where the target was last observed. While we refer the reader to (Bernardini *et al.* 2013) for a full description of the SAT domain, we report here the specification of the action `doSpiral` as an example of how search actions are modelled in PDDL2.1:

```
(:durative-action doSpiral
:parameters (?from ?to - waypoint
                    ?p - spiral )
:duration (=?duration (timefor ?p))
:condition (and
         (at start (beginAt ?from ?p))
         (at start (endAt ?to ?p))
         (at start (at ?from))
         (at end (active ?p)))
:effect (and
         (at end (at ?to))
         (at start (not (at ?from)))
         (at end (increase (reward)
                  (rewardof ?p)))))
```

3.2 Planning Problem

The problem has no goal, but the plan metric measures the value of the plan in terms of the accumulated expectation of finding the target. A few examples of problems of this sort have been considered before, e.g. one variant of the satellite observation problem used in the 3rd International Planning Competition (Long & Fox 2003) had this character.

To create the initial states for the planning problems, we have to manage two tasks: (i) identifying *candidate search patterns*; and (ii) assigning appropriate *rewards* to them. The first task is made difficult by the fact that there are infinitely many patterns that could be used, while the second

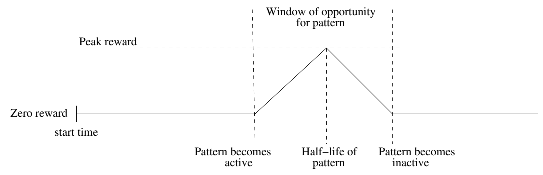

Figure 2: Time-dependent reward function used to assign rewards to search patterns.

is made difficult because of the lack of knowledge about the intentions of the target.

To address the first problem we observe that the planner can only consider a finite subset of search patterns and, since we want to perform planning in real time, is limited to being able to consider a reasonably small number of candidates. Therefore, we generate a sample of possible search patterns by randomly selecting a set of shapes (circles for spirals and large and small rectangular areas for lawnmowers) and placing them onto the general search area. There are three steps involved in this (see (Bernardini *et al.* 2013) for additional technical details on each step):

1. *Search area selection:* we first select a subset of the general search area in which the target is more likely to be rediscovered. This area is identified on the basis of the last known location of the target and its average bearing over the period the target has been observed. Search patterns are generated only within this area.
2. *Sampling:* once the relevant search area is identified, we then sample points using a probability distribution laid over this area, which is based on the density of roads across the sector, the terrain type and the distance from the last known location of the target.
3. *Search pattern generation:* finally, we decide the type of pattern to use for each point. We favour spirals in the part of the search closest to the origin, where spirals give good coverage of the area in which the target might be found, and lawnmowers in rural areas or areas of lower road density, where spirals are likely to cover significant areas of little value in the search.

As for the second problem, i.e. *reward assignment*, we associate with each pattern a reward by using a time-dependent function. The shape of this function, shown in Figure 2, is constructed to represent an approximate lifted Gaussian distribution, with no reward until the target could plausibly have arrived at the search area and no reward after the target is unlikely to be still present in the area. Between these extremes, the reward peaks at the point where the target would be in the centre of the search pattern if driving at average speed. This time-dependent reward function is managed by *timed-initial fluents* in the problem specification that change the reward of the patterns as time progresses.

3.3 Planning and Plan-Based Policy

We use a version of POPF (Coles *et al.* 2010), called OPTIC (Benton, Coles, & Coles 2012), designed to perform anytime, cost-improving search. We use a time-bounded search of 10 seconds given that we are in a time-critical situation. The planner will typically find a first solution very easily, since the empty plan is already a feasible solution, but it will then spend the additional time improving on this by adding further search patterns to the plan, or trying different collections of patterns. The plans produced in this way are monotonically improving, so the final plan produced is the one we select for execution. We have chosen OPTIC because it is very fast at producing its first solution and provides an any-time improvement behaviour. We have found that, on average, OPTIC produces around 6 plans in its 10 second window per problem instance, and the last of these is selected for execution.

In our current version we have implemented a *static policy* which is based on replanning. The observer enters a planning phase every time it loses the target. We allow the observer 10 seconds for planning at each of these points. We currently "freeze" time while the observer is planning because the simulation runs at approximately 50 times real time, but the CPU requirements for planning have to be met in real time. As we shall discussed in Section 6, we aim to later replace this planning behaviour with an essentially instantaneous action selection using a *learned policy*.

4 SAT Simulation and Experimental Results

In order to evaluate the behaviour of our plan-based approach to SAT, we have developed a simulation. This was built in consultation with our industrial collaborators and is intended to provide an appropriately abstracted view of the problem. The key abstraction is that our dispatcher identifies waypoints and turning circles for the UAV according to the flight path and search pattern being executed, but we do not consider control of flight surfaces or altitude.

The area of operations of the simulation is a part of Scotland of about 100 kilometres square, with Glasgow and Edinburgh approximately defining its lower corners. Terrain types were defined by hand, along with an approximate road network for the major roads and rural minor roads. The simulation determines success in spotting and tracking the target, according to terrain, speed and discrepancy between anticipated and actual target positions. The target follows a path acquired using Google Maps, using a configurable origin and destination. This information is also used to decide what speed is appropriate for the target, based on distance between waypoints in the route proposed by Google Maps and terrain type. The simulation integrates the planner and displays the stages of the planning process. Figures 3 and 4 show an example of an initial state and a plan, respectively.

The simulation tool offers various opportunities for interaction, including redirecting the target, repositioning the observer, speeding and slowing the simulation and modifying parameters that govern spotting probabilities, flight dynamics, target behaviour and so on. These values were chosen to broadly represent characteristics of a prototypical UAV as described to us by our industrial collaborators.

By using our simulation, we conducted experiments to compare plan-based search with a static policy proposed by our industrial collaborators. Here, we give a brief overview of the experimental results that we have obtained, while we

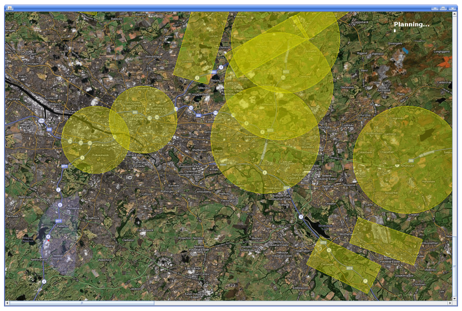

Figure 3: A screenshot showing an initial state: rectangles and circles are search patterns that the planner will consider.

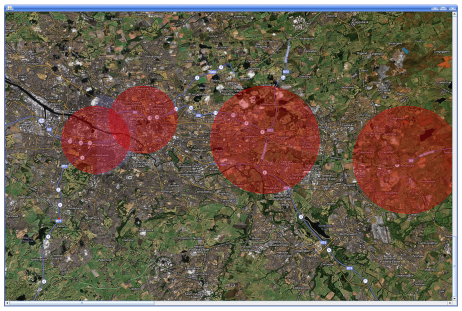

Figure 4: The search patterns that have been selected by the planner ready for execution.

refer the reader to (Bernardini *et al.* 2013) for a more comprehensive description of them.

The static policy works as follows: the observer tracks the target until it has lost sight of it. It continues to track the predicted location of the target for about three minutes. If it has not rediscovered the target, the observer then executes a fixed sequence of search patterns. It first performs a spiral search around the point where it lost the target and then executes a large lawnmower pattern over a 20 kilometre square area. We use a configuration of our plan-based search that tracks the predicted location of the target for the same period as the static policy, before planning and executing a search plan. We generated 20 routes and executed the simulation on each route 1000 times (the simulation has a non-deterministic spotting model and target behaviour), for each of the 2 strategies (a total of 40000 runs). The simulation begins with the target undetected, but in the search arc of the observer. simulation does not use a search plan in this first stage, so failure at this point leads to an early abort. We discount these runs (less than 0.5%) in our analysis. The metric that we use here to evaluate the strategies is the proportion of runs in which the target is tracked to its destination, as shown in Figure 5. The plan-based search strategy is consistently better than the fixed strategy. The static policy has an overall success rate of under 50%, while the plan-based search yields better than 60% success rate. There are a few cases where the static policy appears to do better than the plan-based search and we are still investigating the reason for this. In general, on shorter routes the

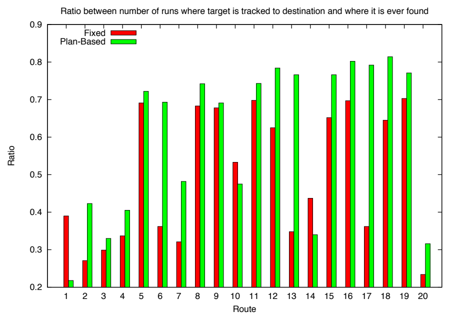

Figure 5: Proportion of successfully tracked targets.

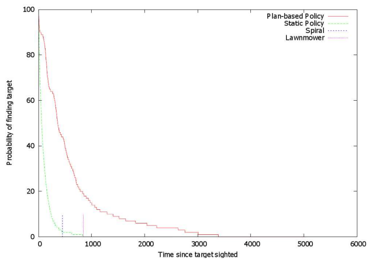

Figure 6: The probability of recapturing the target over time

plan-based approach appears to perform worse, which appears to be because the search patterns are biased towards an assumption that the target is driving to a distant location.

We also analysed the data to find the probability distribution, over time, of relocating the target after losing it, considering only the cases in which it was successfully relocated. Figure 6 shows how this probability changes over time: the planned search is far more robust than the fixed search, offering significant probability of rediscovery of the target even after half-an-hour. Oh the other hand, the static policy only finds the target after it has been lost for 10 minutes in less than 4% of cases. The figure shows the probability of finding the target at a particular time after losing it. A slightly different question is how likely it is that the static policy will find the target *once it has entered the lawnmower search*. We found that to be approximately 13%, which indicates that the lawnmower is not as ineffective as Figure 6 might suggest. We found that the plan-based search very rarely finds the target after the fourth search pattern in a plan.

Our results clearly demonstrate a better performance using planning than using the static policy, but there remains considerable scope for improvement. There is weakness in the way we position the time windows around search patterns and we are investigating how to improve that.

5 Obstacles and Spatial Reasoning

We are currently working on extending our approach to SAT in two main directions: (i) including the effects of *obstacles* that prevent the UAV from tracking the target; (ii) modelling and handling *qualitative spatial constraints*.

5.1 Obstacles

Our current technique and simulation assume that the UAV is free to fly everywhere with no restrictions. However, a more realistic model of the problem needs to include *obstacles*, which are regions that the UAV cannot violate or regions that the UAV can enter with an associated risk of not being able to track the target or even to fly safely. Obstacles can be used to represent a number of entities, from buildings to weather conditions, and might present different characteristics: hard vs. soft, dynamic vs. static, permeant vs. temporary, see-through vs. opaque and big vs. small. In addition, we need to distinguish whether the target is able or not to drive in an area considered an obstacle for the UAV. For example, a military no-fly-zone is an obstacle for the UAV, but not for the target. Clearly, an evasive target might enter such areas to elude pursuit. Following Meuleau *et al.* (2009), we identify the following categories of obstacles: (i) Urban development; (ii) Terrain (vegetation, mountains, etc.); (iii) Adverse weather conditions (rain, showers, thunderstorms, etc.); and (iv) Prohibited or restricted areas (commonly called "no-fly zones"). Urban development, terrain, and no-fly zones are hard, static and permanent obstacles. They cannot be violated and can be incorporated in the model of the environment. Weather conditions are soft, dynamic and temporary obstacles. They need to be modelled in real-time and a risk model needs to be associated with them. Since these four types of obstacles are columnar and we do not consider altitude in our simulation, we model obstacles as *convex polygons*. This choice is common to a number of approaches that consider obstacles (e.g. (Meuleau *et al.* 2009)). Currently, we have considered hard static obstacles of small-medium size with respect to our 100 km square grid, while addressing soft obstacles with an associated risk model is one of our future goals.

When dealing with an environment with obstacles, we need to solve the following issues: (i) how to react to the target entering an area that is an obstacle for the UAV, while the UAV is in tracking mode; (ii) how to treat obstacles during the search pattern generation phase; and (iii) how to navigate between obstacles.

The size and the ability to see inside the obstacle influence how we tackle these issues. If the target enters an area corresponding to an obstacle while the UAV is in tracking mode, then two situations can occur. If the obstacle is small enough to be within the observation range of the UAV's imaging system, then the UAV can simply keep tracking the target if the obstacle is see-through or execute a circular holding pattern in the vicinity of the obstacle, if the obstacle is opaque. If the obstacle is bigger than the observation range, then the UAV identifies the most probable exit side for the target, flies to one of the vertices of such side and does a *patrol search pattern* along it. The most probable exit side is calculated based on the density and significance of roads across the sector, the terrain type and the distance from the symmetry axis and from the last known location of the target.

As for the creation of the search patterns in the presence of obstacles, if the size of the obstacle is smaller than the observation range of the UAV, then we create a circular search pattern centred at the centre of the polygon. If the obstacle is not opaque, the UAV could potentially see the position of the target inside the obstacle and continue to track it at a distance. On the other hand, if the size of the obstacle is larger than the observation range of the UAV, the UAV will not be able to observe inside the obstacle regardless if it is opaque or not. Hence, we follow the sides of the polygon associated with the pattern more precisely in order to maximise the chances to rediscover it. We create a *patrol search pattern* that goes along the sides of the polygon. The entry waypoint for the patrol search pattern is one of the vertices of the most probable exit side, which is calculated as described above for the tracking mode case.

With regard to navigation, recently there has been considerable progress on path planning and obstacle avoidance (see Choset *et al.* (2005) for a survey) and several techniques have been proposed to efficiently solve this problem including probabilistic roadmap (Choset *et al.* 2005) and Rapidly-exploring Random Trees (RRTs) (Cheng, Shen, & LaValle. 2001). Currently, we are working under the assumption that there are only a few obstacles in our 100 km square grid and the path planning is managed by the low-level control of the UAV so that we do not need to explicitly address it. However, if we want to deal with environments cluttered with obstacles, we might need to incorporate a path planning algorithm in the search pattern generation phase. We are exploring the idea of using RRTs to this end.

5.2 Spatial Reasoning

The surveillance scenarios that we consider involve polygons to represent obstacles, spirals and rectangles to represent search patterns, segments to represents roads. In order to plan a course of action for the UAV, we need to be able to reason about the relative positions of these geometrical entities as well as the UAV position in relation to them. Currently, spatial constraints, and indeed geometric and trigonometric functions, cannot be modelled in any of the languages in the standard planning language family, PDDL, neither they can be handled by any state-of-the-art planner, including advanced planners such as OPTIC or COLIN. We aim to extend our planning technology to cope with these novel features introduced in modelling surveillance problems.

We have investigated different ways of representing and reasoning about spatial knowledge and focused on qualitative techniques. *Qualitative* spatial reasoning is concerned with representing and reasoning about spatial knowledge without using exact numerical values. Spacial information is represented by using a finite number of possible *relationships*. A variety of approaches to qualitative spatial reasoning has been proposed (see Cohn *et al.* (2001) for a survey). The fundamental ontological choices that distinguish the various approaches are: (i) what it the primary spatial entity among points and regions of space; and (ii) what qualitative aspects are considered among topology, orientation, distance, direction, size, shape and their combinations. Although we are still exploring several possibilities, we identified two approaches that meet our requirements:

- *Region Connection Calculus* (RCC), which is a first-order theory for representing topological relationships between regions of space (Randell, Cui, & Cohn 1992). RCC is based on a single primitive relation: given two spatial regions x and y, the relation $C(x, y)$ means that x and y are *connected* (i.e. they have one common point).

RCC-8 is a specialisation of RCC that focuses on 8 topological base relations of special importance: (i) x disconnected from y, (ii) x externally connected to y, (iii) x partially overlaps y, (iv) x is equal to y, (v) x is tangential proper part of y, (vi) x is non-tangential proper part of y, (vii) and (viii) are the inverse of the previous two. Constraint-based reasoning techniques originally developed for qualitative temporal reasoning can be used to reason in RCC-8 as well as RCC-8 can be regarded as the spatial counterpart of the temporal (one-dimensional) interval algebra introduced by Allen (Allen 1983).

- *Double-Cross Calculus* (DC-calculus), which is characterised by points as the primary entity and orientation and direction as the qualitative spatial features (Freksa 1992). This calculus allows one to specify the qualitative position of one point with respect to an oriented line segment and express concepts as "to the right of", "behind of", "beyond of". More formally, DC-calculus defines the direction of a located point to a reference point with respect to a perspective point. Three axes are used: one is specified by the perspective point and the reference point, the other two axes are orthogonal to the first one and are specified by the reference point and the perspective point, respectively. These axes define 15 different ternary base relations: left-forward, straight-front, right-forward, left-perpendicular, straight-second-point, right-perpendicular, left-centre, straight-centre, right-centre, left-line, straight-same-location, right-line, left-back, straight-back, right-back. Since this calculus allows one to combine and evaluate path descriptions, DC-calculus has been used to support navigation in a qualitative way (e.g. (Scivos & Nebel 2001)). For example, we can make the following statements: (i) from point A go to point B and make a right turn aiming forward to a point C; (ii) from point B go to point C and make a perpendicular right turn going to point D. From these two descriptions, it is possible to infer that points A and D cannot be identical. The satisfiability problem of constraint systems over DC relations is NP-hard, but the reasoning problem is solvable in PSPACE.

We plan to use *specialised external solvers* to manage these kinds of constraints, instead of doing this inside the planner. This approach is common to a number of other complex high-level control architectures that need to handle spatial reasoning (e.g., (Doherty, Kvarnström, & Heintz 2009)). We are currently experimenting with two publicly available solvers. The first is GQR (Generic Qualitative Reasoner), a solver for binary qualitative constraint networks, developed by the University of Freiburg (2013). GQR takes a calculus description and one or more constraint networks as input, and solves the networks using path consistency and backtracking. RCC-8 is predefined in GQR. The second solver, SparQ, was developed by the University of Bremen (2013) as a collection of tools for qualitative spatial reasoning and directly supports DC-calculus.

The extensions needed to PDDL+ (Fox & Long 2006) to capture RCC-8 and DC-calculus constructs as well as how to integrate answers from the external solver into a developing plan is ongoing work. One of the most difficult technical questions concerns the development of heuristic functions that will guide search effectively in the complex problem spaces produced by metric, temporal and spatial models. While powerful heuristic functions for temporal planning and for continuous planning have been successfully developed (Coles *et al.* 2009; Benton, Coles, & Coles 2012), it is an open research challenge to try to extend these to work effectively in solving much more complex planning problems involving spatial constraints.

6 Policy Learning and Policy Integration

We see our approach to generating plans for a single UAV involved in a SAT mission as a first step towards coordinating multiple intelligent vehicles engaged in complex surveillance tasks through *plan-based policy learning*. In this section, we discuss our long-term research goal in this respect.

Policy-learning is a powerful way to manage *uncertainty*, and it has been explored in a range of different ways including reinforcement learning (Sutton & Barto 1998) and sample-based approaches such as policy rollout (Bertsekas & Castanon 1999). In other work, management of uncertainty in decision-making has been addressed directly, for example in planning under uncertainty (Thiebaux *et al.* 2006), plan execution and repair (Woods *et al.* 2009) and in the use of techniques like hindsight optimisation (Yoon *et al.* 2008). In reinforcement learning and planning techniques, the uncertainty is modelled *explicitly*, resulting in very large state spaces. Although these methods are very powerful, they are resource intensive and systems relying on on-board plan generation are typically not agile or capable of rapid response. Sample-based approaches are much more promising in dynamic situations because they avoid modelling uncertainty explicitly in the representation of the problem, and instead rely on the power of offline learning to cover the space of possibilities. An implicit representation of the uncertainty is required to arrive at a suitable probability distribution from which to sample. These approaches are based on the principles of *Monte Carlo simulation* (Rubinstein & Kroese 2007).

In work on sample-based approaches such as policy rollout, policies are randomly generated and then "rolled out" against real examples of a class of control problems. When the policy is unable to recommend an action, or recommends a poor action, it is improved by adding the missing (state, action) pair or associating the state with a better choice of action. On each iteration, a poor choice of action is replaced by a better option, leading to different states being visited. Following a process of iterative improvement, in which each improvement propagates its effect, the policy ends up being able to respond well to any state. Although promising behaviour has been obtained using this method in simple scenarios (ones that can be modelled in propositional languages) (Yoon, Fern, & Givan 2007), the techniques have not been shown to scale to reasoning about complex models with time, metric resources and continuous effects, all of which typify many real problems.

We propose to use a completely new and much more powerful approach to Monte Carlo-based policy learning, in which we benefit from using a planner to find high-quality solutions to each of the sampled cases, and then a classifier to associate actions with the states encountered in these samples. The policy rollout process is similar to a random walk in the space of policies, starting from an arbitrary initial policy. By contrast, our approach leads to a high quality policy as a result of our investment in offline planning. Furthermore, we plan to use state-of-the-art metric, temporal and continuous planning technology in this offline training phase, resulting in policies that are much more expressive than those that can be learned by using other techniques in the literature. There are *two phases* to this approach:

1. A very large number of samples are taken of initial states of the problem to be solved. Each of these samples constitutes one fully observable case which contains combinatorial choice and (depending on the domain of the problem) might require reasoning about continuous quantities such as time and metric process effects. These cases can be solved using *deterministic planning techniques* as demonstrated by the SAT case. This first phase, therefore, consists of generating and solving thousands of instances to produce thousands of high quality solutions to a representative sample of instances of the problem.
2. The second phase is to obtain a policy from these solutions by means of a *classification process*. Using an off-the-shelf classifier, such as the J48 classifier from WEKA (Hall *et al.* 2009), we can learn a *decision-tree* representation of a policy that captures the vast planning experience embedded in the training data. The quality of the policy depends on the state variables that were chosen to record the planning decisions during training. Usually, it is necessary to experiment with different variables in order to achieve a high level of discrimination in the partitioning of the variables during classification. Identifying a good set of state variables is one of the most challenging technical aspects of this approach to policy-learning.

Although the training phase is expensive in terms of time and computational resources, we see this as an offline process which is not strongly resource-bounded. The classification phase produces a *policy* in the form of a decision tree, the execution of which does not require significant computational resource. The benefit of this approach is that, following the offline training phase, we can produce a software encoding (which can even be realised in hardware) of a policy that stands alone and that can be demonstrated to exhibit very robust behaviour across unseen examples of the control problem at hand.

An important aspect of the class of surveillance problems being considered is that multiple heterogeneous agents often need to *cooperate* in the achievement of tasks. In such a situation, each agent needs a quick and search-free way of deciding what is the best thing to do next, in the context of the actions of other agents. If each agent is executing its own policy independently of the others, there might be unwanted side-effects (such as vehicles not being in the right places at the right times to execute planned joint activities). If the agents are cooperating in the achievement of a task then their behaviours occasionally need to be synchronised. Choreographing these interactions is usually managed by the *human operator*. Another crucial task can also be assign to the human operator. If a complex surveillance task can be broken down into several stand-alone tasks, the best results might be obtained by learning policies for each task component. This means that an agent will need to *switch* between policies as it moves between sub-tasks. Instead of learning single, task-specific policies for each agent, the agents might select between alternative policies, depending on their highest priority task. When an agent switches between policies, which we call *mode-switching*, the choice of which policy to select next can be made by the human operator. We assume that it will be necessary for agents to signal to the human operator when tasks have been completed, to call for assistance in a task and to make commitments about the parts of the task for which they are responsible. Therefore, it is necessary to define an *mixed-initiative* framework in which multi-agent surveillance will be performed, specifying a *communication language* enabling the agents to convey commitments and requests to the operator, and a control interface allowing the operator to coordinate the agents and perform mode-switching.

The policy learning approach focusses on providing a rapid response capability in situations where an agent must decide quickly between alternative actions which are expensive to evaluate in detail. It is less suitable in situations where careful plan-execution monitoring is required (for example, in the dismantling of dangerous equipment). Here, offline planning combined with closely monitored execution and expert plan modification are better suited to the task. Potentially, our approach could be combined with plan execution and repair in cases where both rapid response and carefully monitored behaviour are required.

7 Related Work

7.1 Probabilistic Approaches to SAT

Over the past ten years there has been growing interest in efficient SAT. An approach that has been extensively explored relies on the use of *Bayesian techniques*. Although research originally considered the two areas of searching and tracking separately and focused on scenarios with a single target and a single vehicle (Bourgault, Furukawa, & Durrant-Whyte 2006), the field has rapidly evolved and a unified approach to SAT has been proposed by Furukawa *et al.* (2006) for complex problems featuring multiple targets and observers.

The probabilistic approach to SAT relies on the use of Recursive Bayesian Estimation (RBE) techniques that recursively update and predict the Probability Density Function (PDF) of the target's state with respect to time, under the assumption that the prior distribution and the probabilistic motion model of the target are known. The target is usually assumed to be subjected to external disturbances and not to move on the basis of its own intentions (e.g., a UAV might search for and track life-rafts drifting with wind and current). As RBE techniques are computationally expensive, several approaches have been explored to compute them efficiently, including grid-based methods (Bourgault, Furukawa, & Durrant-Whyte 2006), particle filters (Chung & Furukawa 2006), and element-based techniques (Furukawa, Durrant-Whyte, & Lavis 2007).

Based on the calculated PDF, the search control problem is solved in a greedy fashion over a very short planning horizon; typically, a one-step lookahead horizon is adopted. This myopic planning approach is used to control the computational cost of the technique, which quickly becomes intractable as the number of lookahead steps, the size of the search area, or the number of dimensions of the search space, increases. A unified objective function is used for both search and tracking, allowing a vehicle to switch from one mode to the other while maintaining information gained during the previous phases.

Probabilistic-based SAT has proven successful for problems involving stationary targets or targets moving in small geographical areas, simple motion models in which the targets do not show any evading or intentional behaviour, static search spaces, and short-term missions. Whenever these assumptions are not satisfied, especially for SAT operations over large geographical areas, complex target motion models and long-term operations like those we are interested in, RBE techniques perform poorly due to the high computational cost of accurately maintaining a large state space that includes all the possible positions of the moving targets.

7.2 Orienteering Problem

From a theoretical point of view, our formulation of SAT as a planning problem resembles the *Orienteering Problem with Time Windows* (OPTW) (Kantor & Rosenwein 1992). In a classical orienteering problem (OP), a set of vertices with associated rewards is given as well as a starting and an ending vertex. For all the vertices, the amount of time t_{ij} needed to travel from vertex v_i to vertex v_j is known. Finally, a maximum time budget T_{max} is established. The goal of the OP is to determine a path that visits a subset of the vertices in the available time T_{max} in order to maximise the total collected reward. In the time window variant of the OP, each vertex is associate with a time window and a visit to that vertex can only start during that window. Considering our planning problem, the set of search patterns corresponds to the set of vertices of the OPTW problem, whereas the time slots in which the search patterns are active correspond to the OPTW time windows. As in the OPTW, we also want to maximise the total reward in the available amount of time (limited by the window of the latest possible search). However, in our case and differently from the OPTW, the planner can choose to visit each location more than once and needs to decide which search pattern to use at each location. In the context of planning, the OP has been used to provide suitable heuristic advice on the goals and the goal order that should be considered by a planner that deals with over-subscription problems (Smith 2004).

7.3 Plan-based Policy Learning

The policy-learning approach has been successfully applying to the control of single agents acting under uncertainty in two distinct domains. The first is in managing the loading of multiple independent *battery cells*, in order to maximise their lifetime and reduce switching between batteries (Fox, Long, & Magazzeni 2011). Experimental results suggest that this strategy can reduce the number (and weight) of batteries required to service loads, and this might have important impact in the design of battery-powered field equipment. The second is in controlling an *underwater vehicle* in executing an exploratory mission at sea (Fox, Long, & Magazzeni 2012). In particular, the mission was to track the edge of a large patch in the water, such as an algal bloom, which is moving and is subject to disturbances caused by the ocean dynamics. Tracking a patch involves following the circumference of the patch while it is moving. A plan-based policy can decide how to move based on previous experience with tracking similar patches during training. By sampling many instances of bloom shapes, and planning to track them successfully, it is possible to learn policies that exhibit excellent tracking behaviour in unseen situations.

One of the great advantages of planning is that it can be seen as a rapid *prototyping tool*, since generic PDDL models are easier to build than specialised solutions. Nevertheless, there are challenges in building good planning domain models. For example, the battery domain model is complex, because the battery is subject to the non-linear process of recovery. If the battery is allowed to rest at intervals, then more charge becomes available through the recovery effect. This complex continuous behaviour makes planning very challenging. The task is to extend the lifetimes of the batteries as far as possible by carefully timing the use and rest periods. A good plan is one that gets as close as possible to the maximum lifetime of the batteries, and does so with the smallest number of switches. The continuous planner COLIN(Coles *et al.* 2009) was used to generate the deterministic plans.

8 Conclusions and Future Work

We have presented a planning approach to SAT, viewing the search problem as a planning problem in which search patterns must be selected and sequenced to maximise the expectation of rediscovering the target. Our results indicate that the approach is promising and certainly outperforms static search strategies. By using this approach we have been able to tackle SAT problems on a large scale — a 100 km square area represents a significant challenge to the problem of search, far beyond the capabilities of current alternatives.

Besides performance, we see other advantages in the use of a plan-based approach to SAT. When following a plan, the behaviour of the UAV is predictable and well understood. A plan can be used as a common medium of exchange between the UAV and human observers, allowing safer interaction between the UAV and other air traffic. In addition, although there is clearly a possibility to construct more specialised planning for the SAT problem, an important benefit of the use of a generic planner is that we can readily modify the collection of search actions, add alternative actions and otherwise extend the domain model. This flexibility is particularly important during prototyping and development.

Finally, in this paper, we have discussed our approach to SAT in the context of our long-term research goal, which focuses on automatically learning efficient, light-weight plan-based policies for the high level operation of multiple intelligent vehicles engaged in surveillance in highly dynamic, and potentially hostile, situations. The effectiveness of this approach has been demonstrated in two single-agent cases (the battery and the patch-following domains) and we now aim to scale up the approach to the multi-agent coordination problem, addressing the challenges that arise when many agents are coordinating in solving a problem that requires the integration of multiple policies.

References

Allen, J. 1983. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM* 26(11):832–843.

Benton, J.; Coles, A.; and Coles, A. 2012. Temporal Planning with Preferences and Time-Dependent Continuous Costs. In *Proceedings of the Twenty Second International Conference on Automated Planning and Scheduling (ICAPS-12)*.

Bernardini, S.; Fox, M.; Long, D.; and Bookless, J. 2013. Autonomous Search and Tracking via Temporal Planning. In *Proceedings of the 23st International Conference on Automated Planning and Scheduling (ICAPS-13)*.

Bertsekas, D. P., and Castanon, D. A. 1999. Rollout Algorithms for Stochastic Scheduling Problems. *Journal of Heuristics* 2:89–108.

Bourgault, F.; Furukawa, T.; and Durrant-Whyte, H. F. 2006. Optimal Search for a Lost Target in a Bayesian World. In *Field and Service Robotics*, volume 24 of *Springer Tracts in Advanced Robotics*. Springer Berlin. 209–222.

Cheng, P.; Shen, Z.; and LaValle., S. M. 2001. RRT-based trajectory design for autonomous automobiles and spacecraft. *Archives of Control Sciences* 11(3-4):167–194.

Choset, H.; Burgard, W.; Hutchinson, S.; Kantor, G.; Kavraki, L. E.; Lynch, K.; and Thrun, S. 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press.

Chung, C. F., and Furukawa, T. 2006. Coordinated Search-and-Capture Using Particle Filters. In *Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision*.

Cohn, A. G., and Hazarika, S. M. 2001. Qualitative Spatial Representation and Reasoning: An Overview. *Fundamenta Informaricae* 46(1-2):2–32.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2009. Temporal Planning in Domains with Linear Processes. In *Proceedings of the Joint Conference on Artificial Intelligence (IJCAI)*, 1671–1676.

Coles, A. J.; Coles, A. I.; Fox, M.; and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling*.

Doherty, P.; Kvarnström, J.; and Heintz, F. 2009. A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems. *Autonomous Agents and Multi-Agent Systems* 19:332–377.

Fox, M., and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *Journal of Artificial Intelligence Research* 27.

Fox, M.; Long, D.; and Magazzeni, D. 2011. Automatic Construction of Efficient Multiple Battery Usage Policies. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling, ICAPS 2011, Freiburg, Germany June 11-16, 2011*. AAAI.

Fox, M.; Long, D.; and Magazzeni, D. 2012. Plan-Based Policy-Learning for Autonomous Feature Tracking. In *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS 2012, Atibaia, São Paulo, Brazil, June 25-19, 2012*. AAAI.

Freksa, C. 1992. Using orientation information for qualitative spatial reasoning. *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space* 19:162–178.

Furukawa, T.; Bourgault, F.; Lavis, B.; and Durrant-Whyte, H. F. 2006. Recursive Bayesian Search-and-tracking using Coordinated UAVs for Lost Targets. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA 2006)*, 2521–2526.

Furukawa, T.; Durrant-Whyte, H. F.; and Lavis, B. 2007. The element-based method — theory and its application to Bayesian search and tracking. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, 2807–2812.

Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations* 11(1).

Kantor, M. G., and Rosenwein, M. B. 1992. The Orienteering Problem with Time Windows. *Journal of the Operational Research Society* 43(6):629–635.

Long, D., and Fox, M. 2003. The 3rd International Planning Competition: Results and Analysis. *Journal of Artificial Intelligence Research (JAIR)* 20:1–59.

University of Bremen. 2013. *SparQ*. accessed January 30, 2013.

University of Freiburg. 2013. *GQR (Generic Qualitative Reasoner)*. accessed January 30, 2013.

Meuleau, N.; Plaunt, C.; Smith, D.; and Smith, T. 2009. An Emergency Landing Planner for Damaged Aircraft. In *Twenty-First Innovative Applications of Artificial Intelligence Conference*.

Randell, D. A.; Cui, Z.; and Cohn, A. G. 1992. A spatial logic based on regions and connection. In *Proceedings of the 3rd International Conference on Knowledge Representation and Reasoning*, 165176. Morgan Kaufmann, San Mateo.

Rubinstein, R. Y., and Kroese, D. P. 2007. *Simulation and the Monte Carlo Method*. Wiley, 2 edition.

Scivos, A., and Nebel, B. 2001. Double-crossing: Decidability and computational complexity of a qualitative calculus for navigation. In *Proceedings of the International Conference on Spatial Information Theory: Foundations of Geographic Information Science*, 431–446. London, UK, UK: Springer-Verlag.

Smith, D. E. 2004. Choosing Objectives in Over-Subscription Planning. In *Proceedings of the 14th International Conference on Automated Planning & Scheduling (ICAPS 2004)*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Thiebaux, S.; Gretton, C.; Slaney, J. K.; Price, D.; and Kabanza, F. 2006. Decision-Theoretic Planning with non-Markovian Rewards. *Journal of Artificial Intelligence Research* 25:1774.

Woods, M.; Shaw, A.; Barnes, D.; Price, D.; Long, D.; and Pullan, D. 2009. Autonomous science for an ExoMars Rover-like mission. *Journal of Field Robotics* 46(4):358–390.

Yoon, S. W.; Fern, A.; Givan, R.; and Kambhampati, S. 2008. Probabilistic Planning via Determinization in Hindsight. In *Proceedings of the Conference of Association for Advancement of AI (AAAI)*, 1010–1016.

Yoon, S. W.; Fern, A.; and Givan, R. 2007. Using Learned Policies in Heuristic-Search Planning. In *Proceedings of the Joint Conference on Artificial Intelligence (IJCAI)*, 2047–2053.