Resource Constrained Exploration in Reinforcement Learning

Jen Jen Chung, Nicholas R. J. Lawrance and Salah Sukkarieh
Australian Centre for Field Robotics (ACFR)
The University of Sydney
NSW, 2006, Australia
Email: {j.chung, n.lawrance, salah}@acfr.usyd.edu.au

***Abstract*—This paper examines temporal difference reinforcement learning (RL) with adaptive and directed exploration for resource-limited missions. The scenario considered is for an energy-limited agent which must explore an unknown region to find new energy sources. The presented algorithm uses a Gaussian Process (GP) regression model to estimate the value function in an RL framework. However, to avoid myopic exploration we developed a resource-weighted objective function which combines an estimate of the future information gain using an action rollout with the estimated value function to generate directed explorative action sequences. The results show that under this objective function, the learning agent is able to continue exploring for better state-action trajectories when platform energy is high and follow conservative energy gaining trajectories when platform energy is low.**

I. Introduction

Adaptive and informative exploration in RL can improve the rate of learning and this is especially critical when the learning agent must operate under limited resources, such as limited platform energy. Including resource constraint considerations into the action selection can enable long term operation and learning by scaling the exploration and exploitation according to the available platform resources.

In terms of information-based control and planning under uncertainty, active and informative path planning algorithms developed by Singh et al. [9] and Hollinger et al. [6] have exploited the submodularity property of information gain to select the most informative set of sensing locations. These exploration approaches have typically been used for informative mapping, but can be incorporated into RL problems where the task is to learn a resource gathering policy in an unknown environment.

Various informative exploration implementations have also been a feature of RL strategies. Dearden et al. [3] maintained probability distributions over the value function; Engel et al. [4] and Engel et al. [5] demonstrated the use of GP modelling for value function approximation and noted that the GP covariance could provide confidence bounds on the value estimate to direct exploration; Brochu et al. [1] outlined methods of combining the GP covariance and estimate to form acquisition functions; and more recently, Chung et al. [2] applied information gain rollout to produce a nonmyopic information value to direct exploration.

The standard goal of RL problems is to learn a good policy as the number of observations increase (generally with the aim of generating optimal policies in the limit), however, they do not consider the cost of taking observations. The problem that we are concerned with draws a tighter coupling between what is observed and how the agent may continue to make observations. In this work, the goal of the agent is to learn resource gathering trajectories in the state-action space, in order to do so it must spend that resource to explore the space and gather information.

This raises the question of how to represent exploration and exploitation value quantitatively in the RL framework and how to consolidate these two in an adaptive and nonmyopic way.

As shown in Chung et al. [2], by approximating the value function with a GP, the GP covariance can be manipulated to measure the information value of future actions. However this cannot be naïvely combined with the RL value function since the ability to explore is now dependent on the available resources. An adaptive weighting factor is introduced to represent the current level of available resource and this is used to scale exploration against exploitation.

The resource constrained RL algorithm was tested on a soaring glider problem where the agent must search for energy gaining trajectories while expending energy during exploratory flight. These results were compared against the directed information gain algorithm presented in Chung et al. [2].

In the following section, we outline the value function approximation method using GPs. Section III describes the computation of the information measure using the GP covariance and includes details of the rollout method for nonmyopic exploration. The resource constrained exploration-exploitation objective function is introduced in Section IV, with the soaring glider simulation setup and results presented in Section V and Section VI, respectively. Concluding statements and suggestions for future work are given in Section VII.

II. Gaussian Processes for Value Function Approximation

Traditional RL methods were designed for use on problems with discrete state-action spaces, requiring learning to take place over multiple revisits to any particular state-action pair. Value function approximation techniques have been commonly applied to extend the RL framework to approximate the value of state-action pairs that have not yet been visited.

While any function approximation technique can be used within the RL framework to model the value function, it is desirable to choose a method that is best able to generalise the available observations to the full state-action space. The interested reader is directed to Sutton and Barto [10] for a survey of common RL function approximation techniques. In the following discussions and experiments we build upon a SARSA(λ) RL framework using a GP to model the value function.

GP modelling is a function approximation technique that not only provides a continuous estimate of the value function but also quantifies the uncertainty of the estimate across the function space in the form of a mean and a covariance, respectively, see Rasmussen and Williams [8]. The training inputs $X_N = \{\mathbf{x}_i\}_{i=1}^N$ to the GP are the observed state-action pairs,

$$\mathbf{x}_i = [\mathbf{s}_i, \mathbf{a}_i], \quad (1)$$

and the training targets $\mathbf{y}_N = \{y_i\}_{i=1}^N$ are the corresponding Q-values which are updated on the fly as the eligibility trace decays,

$$y_i = Q(\mathbf{s}_i, \mathbf{a}_i). \quad (2)$$

For a test point $\mathbf{x}_*$, covariance function k and covariance matrix $K = K(X, X)$, the estimated mean value $\bar{Q}_*$ and covariance $\operatorname{cov}(Q_*)$ are

$$\begin{aligned}\bar{Q}_* &= \mathbb{E}\left[Q(\mathbf{x}_*)\,|X, \mathbf{y}, \mathbf{x}_*\right] \\ &= K(\mathbf{x}_*, X)\left[K + \sigma_n^2 I\right]^{-1}\mathbf{y},\end{aligned} \quad (3)$$

$$\operatorname{cov}(Q_*) = K(\mathbf{x}_*, \mathbf{x}_*) - K(\mathbf{x}_*, X)\left[K + \sigma_n^2 I\right]^{-1} K(X, \mathbf{x}_*). \quad (4)$$

Simply stated, the GP approximation assumes that state-action pairs close to one another in the covariance function space will have similar associated Q-values. This assumption can be applied to problems with continuous state and/or action spaces, as well as in discrete state-action problems where the transition function implies some sense of continuity, such as in some grid search problems. Prior assumptions regarding the properties of the value function surface can also be incorporated via judicious design of the GP covariance function. For example, in the following experiments we make the assumption of stationarity in the value function and use the squared exponential covariance function to compute the GP:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T M (\mathbf{x} - \mathbf{x}')\right), \quad (5)$$

where σ_f^2 is the process variance, M is a diagonal matrix with positive elements equal to $\mathbf{l}^{-2}$, and $\mathbf{l} = [l_1, l_2, \ldots, l_n]$ are the length scales in each dimension of the training input vector. While it is possible to train these hyperparameters online, offline training on a set of simulation data was preferred since the initial stages of learning tended to generate large changes in the value function which would lead to instability.

III. Informative Exploration

The goal of exploration in RL is to reduce the uncertainty in the value function estimate. Traditional methods for invoking exploration actions, such as ε-greedy sampling, are susceptible to local minima and fail to actively direct exploration to areas where information of the value function is sparse. Given that we are using a GP to approximate the value function, it is intuitive to use the GP variance measure to quantify and compare the information gain of possible state-action observations. Furthermore, we can achieve a nonmyopic measure of information value by considering the discounted information gain of all the future observations which are possible after executing an action. This can be estimated using the rollout method proposed by Chung et al. [2], which is described briefly in Section III-B.

A. Measuring Information Gain

The GP variance over the state-action space represents a bounding volume around the estimated value function surface. Each consecutive observation results in a reduction in this volume and this reduction represents the information gain of the corresponding state-action observation,

$$V_{bound_N} = \int_{x_{n_a}}^{x_{n_b}} \cdots \int_{x_{1_a}}^{x_{1_b}} \operatorname{cov}([x_1, \ldots, x_n]\,|X_N) \mathrm{d}x_1 \ldots \mathrm{d}x_n, \quad (6)$$

$$I_{gain} = V_{bound_N} - V_{bound_{N+1}}. \quad (7)$$

The training set X_N consists of n-dimensional state-action pairs, with dimensions $[x_1, \ldots, x_n] = [s_1, \ldots, s_u, a_1, \ldots, a_v]$ (typically, $v = 1$); furthermore, $X_{N+1} = X_N \bigcup \mathbf{x}_{N+1}$.

Since the squared exponential covariance function given in (5) is an integrable function, an analytical solution to (6) can be found,

$$\begin{aligned}V_{bound_N} &= \int_{x_{n_a}}^{x_{n_b}} \cdots \int_{x_{1_a}}^{x_{1_b}} k([x_1, \ldots, x_n], [x_1, \ldots, x_n]) - \\ &\quad k([x_1, \ldots, x_n], X_N) K_{XX}^{-1} k(X_N, [x_1, \ldots, x_n]) \mathrm{d}x_1 \ldots \mathrm{d}x_n \\ &= \sigma_f^2 \prod_{m=1}^{n}(x_{m_b} - x_{m_a}) - \sigma_f^4 \left(\frac{\sqrt{\pi}}{2}\right)^n \prod_{m=1}^{n} l_m \\ &\quad \times \sum_{i,j}\left(K_{XX}^{-1}\right)_{i,j} \exp\left[-\sum_{m=1}^{n}\left(\frac{X_{m_i} - X_{m_j}}{2 l_m}\right)^2\right] \times \\ &\quad \prod_{m=1}^{n}\left(\operatorname{erf}\left(\frac{x_{m_b} - \frac{X_{m_i} + X_{m_j}}{2}}{l_m}\right) - \operatorname{erf}\left(\frac{x_{m_a} - \frac{X_{m_i} + X_{m_j}}{2}}{l_m}\right)\right),\end{aligned} \quad (8)$$

where k is the squared exponential covariance function, $K_{XX}^{-1} = \left[K(X, X) + \sigma_n^2 I\right]^{-1}$ is the inverse of the covariance matrix of all the observed locations including additive white noise drawn from $\mathcal{N}\left(0, \sigma_n^2\right)$.

The limits of integration can be chosen to incorporate the entire space or only a local portion of it, e.g. a reachable set within a finite time horizon. In the following experiments, the integral is taken over the entire state-action space.

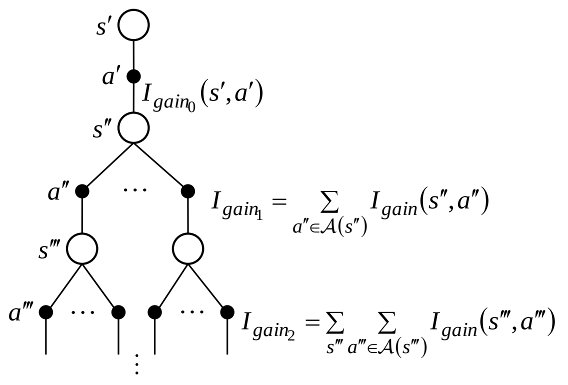

s'
a'
$I_{gain_0}(s',a')$
s''
a''
$I_{gain_1} = \sum_{a'' \in \mathcal{A}(s'')} I_{gain}(s'',a'')$
s'''
a'''
$I_{gain_2} = \sum_{s'''} \sum_{a''' \in \mathcal{A}(s''')} I_{gain}(s''',a''')$

Fig. 1: The rollout method introduced in Chung et al. [2]. The information gain of each rollout level considers all the reachable state-actions at that level. The total information gain is a discounted sum of the information gain of each level.

B. Nonmyopic Exploration via Rollout

The information *reward* of taking an observation at a particular state-action is simply computed as (7), however approaching this from an RL perspective, it is more meaningful to consider the nonmyopic information *value*, i.e. the total future information gain possible due to performing a particular action. There is one key difference to note between the RL state-action value and this information value: the RL reward for any state-action is constant whereas the information reward for reobserving a particular state action decreases over the number of observations. The main consequence of this is that the information reward cannot be directly included in the computation of Q, and similarly, information gain credit cannot be meaningfully assigned back along the eligibility trace. Instead, we take the approach of looking at all the possible future state-action observations leading out from a particular action and use the discounted sum of those rewards to compute the information value.

The information gain rollout technique is illustrated in Fig 1. The information value sums the information gained from the next propsed transition, $s', a' \to s''$ where $a' \in \mathcal{A}(s')$ and $\mathcal{A}(s')$ is the set of available actions from state s', with the discounted information gained from all possible future state-actions rolled out from s'' up to a threshold discount factor. Given a discount parameter γ_r, the total information gain of an action a is computed as

$$I_{a_{total}} = I_{gain_0} + \gamma_r I_{gain_1} + \gamma_r^2 I_{gain_2} + \ldots + \gamma_r^p I_{gain_p}, \quad (9)$$

where p is the highest integer for which γ_r^p is greater than the discount threshold γ_{thres}.

IV. Action Selection

With both the information value and the state-action value, the action selection task now becomes one of consolidating the two competing objectives of *exploration* and *exploitation*. There are a number of factors to consider when combining these two values into a single objective function: firstly, the scale of each value may be very different, resulting in one dominating the other if combined naïvely. Secondly, as noted in Section III-A, having defined the information reward as the change in the GP variance volume, the value of information naturally decreases as more observations are taken. Finally, given a system subject to resource constraints, such as available platform energy, the desire to explore or exploit should adapt according to the current available resources. These issues will be addressed in the following subsections.

A. Reward Tradeoff

The role of the action selection objective function is to rank the available actions in a way which reflects the objectives of the learning task, which in this case is to discover the optimal trajectory through the state-action space. Since it is only the ranking which ultimately determines which action is executed, it is useful to normalise the values when combining them as this also removes the problem of their differences in scaling. That is, let

$$\hat{Q}_i = \frac{\bar{Q}_i}{\max |\bar{Q}_i|}, \quad (10)$$

$$\hat{I}_i = \frac{I_{i_{total}}}{\max |I_{i_{total}}|}, \quad (11)$$

where i is the current timestep. Note that the value estimate component lies within $[-1, 1]$ while the information gain component lies within $(0, 1]$.

A side-effect of this normalisation is that the natural decline in information value as more observations are taken is also eliminated. Generally it is desirable to maintain this decline in the final action selection objective function to encourage the learning agent to explore more at the start and less later on when the number of observations increases. Chung et al. [2] demonstrated that this decline in information value can be reintroduced as a decreasing weighting factor on the normalised information value, the overall effect of this was to scale the agent's behaviour from exploratory to exploitative as learning progressed.

B. Resource Limitations

RL research has typically dealt with problems where the single goal is to efficiently find an optimal state-action trajectory for an agent as its number of observations increases to infinity. To the authors' best knowledge, limitations on available resources such as platform energy, etc. have largely been neglected from such problems in existing research. However, when the RL problem is a resource seeking mission, the reward and the ability to continue collecting rewards become tightly coupled and such limitations can no longer be ignored. In the following experiments we consider a soaring glider learning to gather energy from a wind field. In order to do so it must expend energy to explore the space for profitable flight trajectories. The resource in this problem is the available platform energy, which determines the continuation or (critical) termination of a mission. For simplicity, in the following discussions we will focus solely on platform energy as the resource and reward of interest.

Algorithm 1 eGP-SARSA(λ)

```
1:  Initialise $\theta$                                  ▷ GP hyperparameters
2:  $\mathbf{e} \leftarrow \mathbf{0}$                   ▷ Initialise trace
3:  $s, a \leftarrow$ initial state and action
4:  $\bar{Q}_a \leftarrow 0$                              ▷ Initial value estimate
5:  $E_{init} \leftarrow$ initial energy
6:  for each step $i$ do
7:      $e(s,a) \leftarrow 1$                             ▷ Replacing traces
8:      Take action $a$, observe reward, $r$, next state, $s'$, and
        current energy $E_i$
9:      $\delta \leftarrow r - \bar{Q}_a$
10:     for all $a'' \in \mathcal{A}(s')$ do
11:         $\bar{Q}_{a''} \sim \mathcal{GP}_Q$            ▷ GP approximation for $Q$
12:         $I_{a''_{total}} \leftarrow I_{gain_0} + \gamma_r I_{gain_1} + \ldots + \gamma_r^p I_{gain_p}$
13:     end for
14:     $\omega \leftarrow \max\left(0, \min\left(1, \frac{E_i}{E_{init}}\right)\right)$   ▷ Resource limited
                                                          exploration weight
15:     $\hat{Q}_{a''} \leftarrow \frac{\bar{Q}_{a''}}{\max|\bar{Q}_{a''}|}$   ▷ Normalised state-action value
16:     $\hat{I}_{a''} \leftarrow \frac{I_{a''_{total}}}{\max|I_{a''_{total}}|}$   ▷ Normalised information value
17:     $J_{a''_e} \leftarrow \hat{Q}_{a''} + \omega \hat{I}_{a''}$
18:     $a' \leftarrow \arg\max_{a''} J_{a''_e}$
19:     $\delta \leftarrow \delta + \gamma \bar{Q}_{a'}$
20:     if $(s,a)$ is a new state then
21:         Append $(s,a)$ to $\mathcal{GP}_Q$ training inputs $X$
22:     end if
23:     $\mathbf{y} \leftarrow \mathbf{y} + \alpha\delta\mathbf{e}$   ▷ Update $\mathcal{GP}_Q$ training targets
24:     if retrain hyperparameters then
25:         $\theta \leftarrow \arg\min_\theta -\log p(\mathbf{y}|X,\theta)$   ▷ Minimise the
                                                          negative log marginal likelihood
26:     end if
27:     $\mathbf{e} \leftarrow \gamma\lambda\mathbf{e}$
28:     $s \leftarrow s'$
29:     $a \leftarrow a'$
30: end for
```

We propose a dynamic scaling of the exploration behaviour according to the available platform energy using the weighting factor,

$$\omega = \max\left(0, \min\left(1, \frac{E_i}{E_{max}}\right)\right), \tag{12}$$

combined with (10) and (11) to produce the resource contrained exploration-exploitation objective function:

$$J_{i_e} = \hat{Q}_i + \omega \hat{I}_i, \tag{13}$$

where E_i is the available platform energy at timestep i and E_{max} is the maximum amount of energy that can be stored on the platform (e.g. corresponding to a maximum altitude). The effect of this objective function is to increase the influence of the information value when platform energy is high so that the agent tends to explore areas of the state-action space which have high uncertainty. When platform energy is low, the state-action value dominates the action selection objective function so that the agent tends to exploit areas of the state-action space which are believed to produce high energy reward, thereby replenishing the resource.

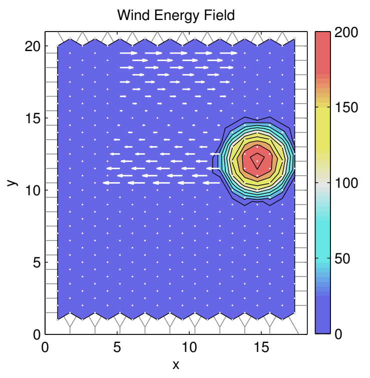


Fig. 2: Wind energy field for the soaring glider simulation. A thermal is centred at $(17, 12)$ and a wind shear field is present between the limits of $5 \leq x \leq 15$ and $10 \leq y \leq 20$.

The full eGP-SARSA(λ) algorithm is presented in Algorithm 1. For each action selection, Algorithm 1 requires simulation of the motion model and estimation of the resulting information gain across a tree with branching factor in the number of possible actions and search depth imposed by the limit γ_r.

V. Experimental Setup

We revisit the soaring glider problem presented by Chung et al. [2]. The glider, representing the learning agent, operates in a discretised hexagonal grid world with an underlying wind energy field shown in Fig. 2. The cell location and platform heading constitute the glider state, (x, y, ψ), where the headings are discretised along the six faces of the hexgrid cells. Note that since the platform heading is an angular state, care must taken to ensure that the wraparound condition is enforced when computing the squared distances in the GP covariance function (5).

At each decision instance, the glider has the choice of three actions: turn left, go straight or turn right, as shown in Fig. 3. Headings are defined as 0° north, then 60°, 120°, etc., travelling in an anti-clockwise direction from north. The glider is only able to travel within the defined search space and receives a penalty for choosing actions which attempt to move it beyond the boundaries.

The reward functions for the hexgrid example are based on a simple glider aircraft model. During flight in no wind

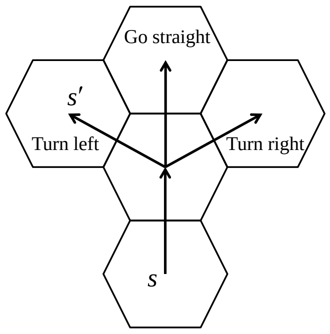


Fig. 3: State transition for each available action. Each action involves an initial forward step, the following step is either into the cell on a bearing of 60° to the left or right, or into the cell directly ahead.

the aircraft is assumed to lose energy at a constant rate while travelling forwards with an additional penalty when turning. This is equivalent to a glider travelling at a constant airspeed which loses energy in proportion to the lift to drag ratio $\frac{L}{D}$, the vehicle mass m and the acceleration due to gravity g. For an electric vehicle these can be considered constant values during flight. An additional penalty is applied to turns due to the drag from increased wing loading. With the fixed turn angles of $60°$ in the hex grid example, the additional turn penalty is approximated as 1.3 times the steady-level energy loss over the same distance. The resulting cost function for a movement action between cells spaced distance d apart is

$$r_{move} = -\frac{mgd}{\frac{L}{D}} \times \begin{cases} 1 & \text{if action = go straight} \\ 1.3 & \text{otherwise.} \end{cases} \qquad (14)$$

The energy sources in the field are either rising air (static soaring) or wind gradient (dynamic soaring) sources. A rising air energy source is placed at $(17, 12)$ with a defined core wind speed w_t and radius l_t. Energy is gained from rising air simply by flying through it (independent of heading) and is proportional to the strength of the wind. At a distance d_t from a thermal the reward is

$$r_{therm} = \begin{cases} 1 - \left(\frac{d_t}{l_t}\right)^2 & \text{if } d_t < l_t \\ 0 & \text{if } d_t \geq l_t. \end{cases} \qquad (15)$$

Dynamic soaring is the process of collecting energy by moving through a spatial wind gradient which effectively increases the airspeed of the aircraft. By flying cyclic patterns through wind shear it is possible to continuously increase airspeed. Typically, dynamic soaring is performed in a vertical wind gradient (with respect to altitude) rather than the planar wind gradient used here. Whilst an in-depth discussion of dynamic soaring is beyond the scope of this paper, suitable descriptions can be found in Lawrance [7], Wood [12] and Weimerskirch et al. [11]. The dynamic soaring sources are modelled as a planar linear shear gradient, as noted by the wind vectors in Fig. 2. Energy capture from a wind gradient is due to the increased air-relative kinetic energy from the increase in airspeed from the wind gradient. For a linear wind gradient $\frac{\partial W_x}{\partial y}$, the kinetic power is

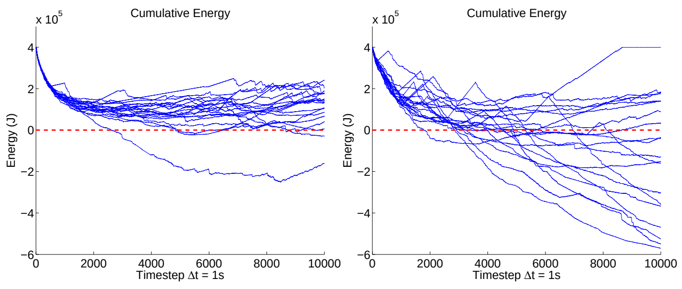
$$\frac{dE_{kinetic}}{dt} = \frac{1}{2} m \left(\frac{\partial W_x}{\partial y} V^2 \cos\psi \sin\psi \right). \qquad (16)$$

This is effectively the projection of the wind gradient in the airspeed direction, for this simulation the resulting reward function for travelling distance d through a linear gradient $\frac{\partial W_x}{\partial y}$ at heading ψ is

$$r_{shear} = \frac{1}{2} m d \sin\psi \cos\psi \frac{\partial W_x}{\partial y} \left(2V + d \sin\psi \cos\psi \frac{\partial W_x}{\partial y} \right). \qquad (17)$$

Finally, the reward function includes a penalty term for flight outside the specified area. The edge penalty is twice the magnitude of the strongest wind energy source,

$$r_{edge} = \begin{cases} -2 \max W & \text{if outside flight area} \\ 0 & \text{otherwise.} \end{cases} \qquad (18)$$



(a) eGP-SARSA(λ) (b) iGP-SARSA(λ)

Fig. 4: Cumulative energy over the course of the simulation for each of the 20 trials. Energies below the threshold value of 0 represent a critical failure.

The resulting reward function is the sum of these four components,

$$r = r_{move} + r_{therm} + r_{shear} + r_{edge}. \qquad (19)$$

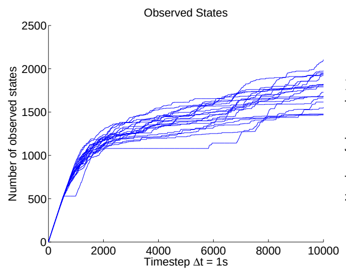
In addition to the reward function, the energy gained from thermalling and dynamic soaring as well as the energy lost due to drag are used to compute the remaining platform energy. In the experiments that follow, the agent is restricted to an upper energy bound defined by the maximum energy and a lower bound of zero which represents a critical failure. For each experiment, the glider began in a randomly selected state with a randomly selected action at maximum energy. The maximum energy was chosen to be the amount of energy required to fly straight and level for half of the total simulation time. Considering the additional energy penalties for turning and attempting to exit the field, it is not expected that a random policy will be able to maintain positive platform energy levels over the duration of the simulation.

VI. Results

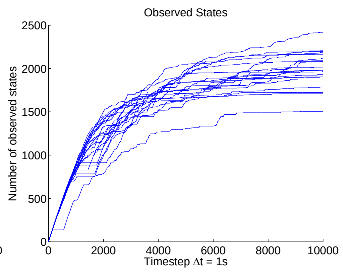
The glider experiment was repeated over 20 trials of $10^4 s$ simulations to compare the performance of the eGP-SARSA(λ) algorithm to the iGP-SARSA(λ) algorithm presented in Chung et al. [2]. Over the 20 trials, 5 of the eGP-SARSA(λ) trials and 14 of the iGP-SARSA(λ) trials failed to maintain positive energy levels.

The eGP-SARSA(λ) objective function adaptively preferences exploitation or exploration according to platform energy, encouraging the agent to exploit energy gaining trajectories when platform energy is low, while the iGP-SARSA(λ) only applies a decline in the information (exploration) value over time. The difference is highlighted in Fig. 4 where the spread of the cumulative energy plots for the iGP-SARSA(λ) cases decline rapidly, while the majority of the eGP-SARSA(λ) cases are able to stay above the threshold value.

The number of observed state-actions over each trial are shown in Fig. 5. The plots suggest that the number of newly observed state-actions reach an asymptotic value in the iGP-SARSA(λ) trials, whereas there appears to be a general trend towards a constant increase in new observations as the eGP-SARSA(λ) simulations progress. The asymptotic exploration behaviour of the iGP-SARSA(λ) is expected since the objective function decreases the information value to 0 over the



(a) eGP-SARSA(λ)



(b) iGP-SARSA(λ)

Fig. 5: The number of observed state-action pairs over the course of the simulation for each of the 20 trials. Since the information value is decreased to 0 in the iGP-SARSA(λ) trials, exploration declines in the latter stages of the mission, whereas the eGP-SARSA(λ) trials continue to explore at a constant rate over the entire mission.

course of the simulation. However the learning behaviour of the eGP-SARSA(λ) algorithm may be more desirable in cases where a final mission time is not given, rather the goal is to continue operation until some terminal condition (such as a threshold platform energy) is triggered.

Two energy gaining flight trajectories from a pair of trials are shown in Fig. 6. Given the complexity of the field and the reward function, there are many flight plan loops that result in net energy gain, although some loops are more efficient than others, as can be seen in Fig. 6.

VII. Conclusions and Future Work

The results presented in this paper demonstrated that introducing resource constraint considerations into the objective function of the RL algorithm produced comparable exploration capabilities whilst satisfying other critical platform constraints. However, the proposed eGP-SARSA(λ) algorithm was unable to provide hard guarantees on resource thresholding. A future research direction is to consider the relationship between the initial rates of learning and the required initial energy to provide threshold energy guarantees.

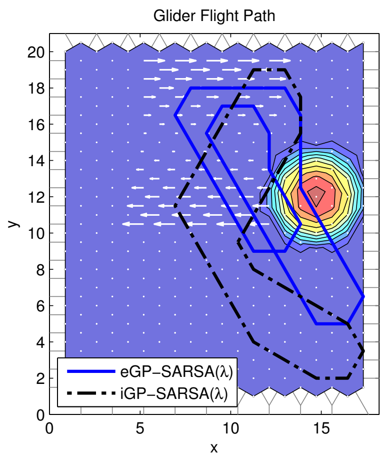


Fig. 6: Final energy gaining flight plan loops for one set of trials. The eGP-SARSA(λ) flight path (solid line) maintains the flight loop within and around the wind energy sources, making use of both the wind shear and thermal energy. The iGP-SARSA(λ) flight path (dot-dash) only exploits the wind shear energy source. A large section of the loop also lies far outside of the energy sources, which would reduce the overall energy gain efficiency.

References

[1] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical report, University of British Columbia, 2010. URL http://arxiv.org/abs/1012.2599.

[2] J. J. Chung, N. R. J. Lawrance, and S. Sukkarieh. Gaussian processes for informative exploration in reinforcement learning. In *2013 IEEE International Conference on Robotics and Automation*, pages 2618–2624, 2013.

[3] R. Dearden, N. Friedman, and S. Russell. Bayesian Q-learning. In *Proceedings of the National Conference on Artificial Intelligence*, pages 761–768, 1998. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.1256.

[4] Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: The Gaussian process approach to temporal difference learning. In *Proceedings of the 20th International Conference on Machine Learning*, pages 154–161, 2003. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.11.2648.

[5] Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 201–208, 2005. URL http://dl.acm.org/citation.cfm?id=1102377.

[6] G. A. Hollinger, B. Englot, F. S. Hover, U. Mitra, and G. S. Sukhatme. Active planning for underwater inspection and the benefit of adaptivity. *The International Journal of Robotics Research*, 32(1):3–18, 2013. URL http://ijr.sagepub.com/content/32/1/3.short.

[7] N. R. J. Lawrance. *Autonomous Soaring Flight for Unmanned Aerial Vehicles*. PhD thesis, The University of Sydney, 2011. URL http://db.acfr.usyd.edu.au/content.php/292.html?publicationid=934&displaypage=1.

[8] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

[9] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34(2):707–755, 2009. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.160.904.

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

[11] H. Weimerskirch, T. Guionnet, J. Martin, S. A. Shaffer, and D. P. Costa. Fast and fuel efficient? Optimal use of wind by flying albatrosses. *Proceedings of the Royal Society of London - Biological Sciences*, 267(1455):1869–1874, 2000. URL http://rspb.royalsocietypublishing.org/content/267/1455/1869.short.

[12] C. J. Wood. The flight of albatrosses (A computer simulation). *Ibis*, 115(2):244–256, 1972. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1474-919X.1973.tb02640.x/abstract.