
Proceedings Paper:
Szwalek, A., Liu, X., Lyu, C. et al. (2026) Bayesian optimisation for sensor scheduling and tracking with different acquisition functions. In: Proceedings of 2025 IEEE Sensor Data Fusion: Trends, Solutions, Applications (SDF). 2025 IEEE Sensor Data Fusion: Trends, Solutions, Applications (SDF), 24-26 Nov 2025, Bonn, Germany. Institute of Electrical and Electronics Engineers (IEEE). ISBN: 9798331576523. ISSN: 2333-7427. EISSN: 2473-7666.

https://doi.org/10.1109/SDF67080.2025.11331217

Bayesian Optimisation for Sensor Scheduling and Tracking with Different Acquisition Functions

1st Alicja Szwalek
School of Electrical & Electronic Engineering, University of Sheffield
Sheffield, United Kingdom
aszwalek1@sheffield.ac.uk

2nd Xingchi Liu
Scientific Computing Department, Rutherford Appleton Laboratory, Science and Technology Facilities Council,
Didcot, United Kingdom
xingchi.liu@stfc.ac.uk

3rd Chenyi Lyu
School of Electrical & Electronic Engineering, University of Sheffield
Sheffield, United Kingdom
clyu5@sheffield.ac.uk

4th Alasdair Hunter
Dstl, Porton Down, Signal Processing and Data Fusion
Salisbury, United Kingdom
ahunter@dstl.gov.uk

5th Lyudmila Mihaylova
School of Electrical & Electronic Engineering, University of Sheffield
Sheffield, United Kingdom
l.s.mihaylova@sheffield.ac.uk

***Abstract*—Joint target tracking and sensor scheduling includes resource optimisation and gathering the most informative data for purposes such as search and rescue, fire detection and surveillance tasks. For such real-time tasks, the limited access to initial tracking data can challenge the effectiveness of traditional machine learning methods, thereby motivating the development of active sensing strategies. This paper addresses such problems and formulates the joint target tracking and sensor scheduling problems within a Bayesian optimisation framework. The key question that this framework answers is: where to position the sensors in order to accurately track an object. In the considered case study, the sensors are mobile and represented by uncrewed aerial vehicles (UAVs). The active sensing of the environment is based on uncertainty-guided sampling thanks to a Gaussian process representation.**

The main novelty lies in the formulation of the sensor scheduling and tracking within a Bayesian optimisation setting. Under this framework, a detailed comparison of different acquisition functions is carried out, to identify the most suitable solutions for an active sensing problem. Results with respect to accuracy and computational time are reported.

***Index Terms*—Bayesian optimisation, sensor scheduling, UAV, active learning, Gaussian process methods, target tracking, uncertainty quantification, upper confidence bound**

I. Introduction

Target tracking over sensor networks is an important problem and plays a critical role in applications such as surveillance and environmental monitoring. A variety of model-based approaches have been developed, including the Kalman filter, [1] and the particle filter [2], both of which rely on a state-space model to represent the system dynamics and observation process. Consensus Kalman filters [3] have also been a popular method.

Recently, model-free methods have gained significant attention in target tracking. For example, the adaptive kernel learning Kalman filter [4] adapts online without relying on an explicit system model, while some reinforcement learning approaches [5] embedded neural networks to approximate the underlying system dynamics or control policies. Model-based methods are often preferred as they offer lower computational complexity compared with model-free methods. However, in situations where tracking data is limited, such as in the presence of occlusions, model-based methods often struggle to accurately represent the target's expected behaviour, resulting in decreased accuracy or even target loss. This has led to the adoption of techniques such as Bayesian Optimisation (BO) [6], which construct a surrogate model to approximate an expensive-to-evaluate objective function. An acquisition function, derived from this surrogate, is then used to guide the selection of the next sampling point [6]. Gaussian process (GP) regression is often chosen as a surrogate for BO due to its ability to quantify the uncertainty of the predictions and its flexibility as a non-parametric model.

BO has been applied to many areas such as environment monitoring [7], robot navigation [8] and particle accelerator tuning [9]. In [10], a BO-guided method with a GP surrogate was used for target tracking with no prior information. The target is tracked using one or multiple Uncrewed Aerial Vehicles (UAV) equipped with Received Signal Strength (RSS) sensors. The RSS of a moving target is modelled as a black-box function and the Expected Improvement (EI) [11] acquisition function is used to determine the next sampling location.

Acquisition functions can be used to guide a UAV in searching for a target by selecting informative sampling locations. The problem of determining when and where to collect measurements is referred to as *sensor scheduling*. This can be formulated as a stochastic optimal control problem such as the partially observable Markov decision process [12] to consider both short-term and long-term performance. Another approach would be using a myopic or greedy approach, which does not consider long-term performance but tends to achieve

This work is supported by the UK Defence Science and Technology Laboratory (Dstl), part of the UK Ministry of Defence, under project "Trust in Distributed Surveillance Systems"

good performance [13].

A. Main Contributions

This paper builds upon the framework developed in our previous work [10] and investigates the impact of different acquisition functions on target tracking in active sensing settings. Additionally, it explores the effectiveness of using a single-output GP model to guide multiple UAVs in an active sensing setting with different levels of noise. Three different acquisition functions are adopted—namely the upper confidence bound (UCB) [14], the knowledge gradient function (KG) [15] and Thompson sampling (TS) [16]—and are compared to the originally proposed acquisition function, expected improvement (EI).

The rest of the paper is organised as follows. Section II introduces the problem formulation and the background knowledge. The proposed methods are detailed in Section III. The results of the simulations are discussed in section IV followed by conclusions in Section V.

II. Problem Formulation

Consider the sensor scheduling problem of determining the "best", in a certain sense, position of a suite of sensors. The sensors - in our case, UAVs - are collecting information about an object on the ground and the aim of the sensors is to collect the most informative data and track the target on the ground. Based on the acquisition function, we choose where to send the UAVs, using predictions of areas with the highest RSS values, which we assume correspond to the target location. This was illustrated in Figure 1, where the target position is indicated by the peak in the RSS measurement surface.

For the purposes of sensor scheduling and quantifying the informativeness of the data, the task is formulated as a Bayesian optimisation (BO) function.

For clarity, we denote vectors as bold lowercase letters, and matrices as bold uppercase letters.

A. Measurement Equation

The UAV collects measurements denoted by z, which depend on the target location $\mathbf{x}_t$ at time t, with additive measurement noise ϵ. The noise is assumed to be zero-mean Gaussian random variable with variance σ^2_{noise}, namely

$$z = f(\mathbf{x}_t, t) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2_{\text{noise}}), \tag{1}$$

where $f(\cdot)$ is an unknown underlying function and $\mathcal{N}(.)$ denotes a normal distribution.

The measurement equation connecting the observed RSS and the target location can be written as follows:

$$z_{t_i} = z_{0,t_i} - \eta \log_{10}(d_{t_i}) + \epsilon, \tag{2}$$

where z_{t_i} is an RSS measurement at time t_i, z_{0,t_i} is the transmission power of the target, η is the path loss exponent determining how sharp the signal attenuates and d_{t_i} is the Euclidean distance between the current sensor position and the target's position. We treat the area with high RSS values as the target position. Denote the set of measurements received until time t by $\mathbf{z}_t = [z_{t_1}, z_{t_2}, \cdots, z_{t_{n_t}}]^\intercal$ and $\intercal$ represents the transpose operation. The UAVs are collecting RSS data, and by the time t, n_t measurements are received with time stamps $t_1, t_2, \cdots, t_{n_t}$.

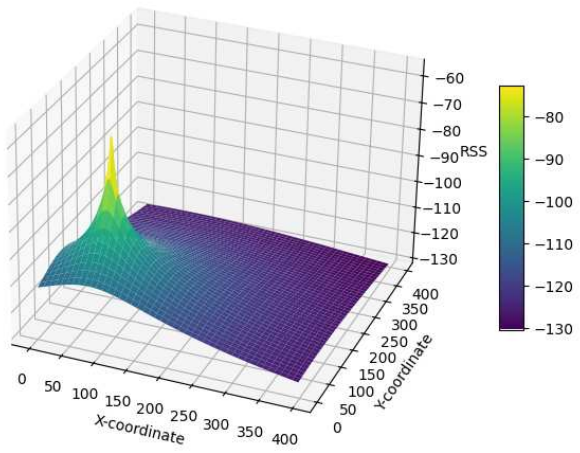


Figure 1. Objective function at a given timestep.

Since $\mathbf{x}_{t_i}$ is the the target location associated with the measurement at time stamp t_i, where $t_i \leq t$, at any time t, we can have a set denoted as $\mathcal{D}_t = \{\mathbf{x}_{t_i}, t_i, z_{t_i}\}_{i=1}^{n_t}$. We will further use the following matrix representation: $\mathbf{X}_t = [\mathbf{x}_1^\top, \mathbf{x}_2^\top, \ldots, \mathbf{x}_{n_t}^\top]$.

B. Gaussian Process (GP) Method

Bayesian optimisation relies on two key components: a surrogate model, typically a GP, which represents our prior belief about the objective function, and an acquisition function, which guides the search for optimal sensor placements. To build a probabilistic model of the objective function $f(\cdot)$, a GP is defined as a distribution over functions:

$$f(\mathbf{x}_t, t) \sim \mathcal{GP}\big(\mu(\mathbf{x}_t, t),\ k\big((\mathbf{x}_t, t), (\mathbf{x}'_t, t')\big)\big). \tag{3}$$

Here, $(\mathbf{x_t}, t)$ and $(\mathbf{x}'_\mathbf{t}, t')$ denote the training and testing input data respectively. $\mathcal{GP}(.)$ denotes a GP with mean function $\mu(x_t, t)$ and covariance function $k((x_t, t), (x'_t, t'))$. The vector $\mathbf{x_t}_i$ represents the spatial location of the $i^{(th)}$ observation at time t_i. Given the data $\mathcal{D}_t$, we define $\mathbf{K}_t \in \mathbb{R}^{n_t \times n_t}$ as a covariance matrix with the $(i,j)^{\text{th}}$ entry as $k((\mathbf{x}_{t_i}, t_i), (\mathbf{x}_{t_j}, t_j))$. In addition, we define $\mathbf{k}_*$ as a vector with the j^{th} entry as $k((\mathbf{x}_{t_j}, t_j), (x_{t_*}, t_*))$.

The posterior distribution at a new input $(\mathbf{x}_*, t_*)$ can be written as:

$$p\left(f(\mathbf{x}_*, t_*) \mid \mathcal{D}_t\right) \sim \mathcal{N}\left(\mu_{\text{post}}(\mathbf{x}_*, t_*),\ k_{\text{post}}((\mathbf{x}_*, t_*), (\mathbf{x}'_*, t'_*))\right), \tag{4}$$

where

$$\mu_{\text{post}}(\mathbf{x}_*, t_*) = \mu(\mathbf{x}_*, t_*) + \mathbf{k}_*^\top \left[\mathbf{K}_t + \sigma^2_{\text{noise}}\mathbf{I}\right]^{-1} \times \left(\mathbf{z} - \mu(\mathbf{X}_t)\right), \tag{5}$$

$$k_{\text{post}}\big((\mathbf{x}_*, t_*), (\mathbf{x}'_*, t'_*)\big) = k\big((\mathbf{x}_*, t_*), (\mathbf{x}'_*, t'_*)\big) - \mathbf{k}_*^\top \left[\mathbf{K}_t + \sigma^2_{\text{noise}}\mathbf{I}\right]^{-1} \mathbf{k}_*. \quad (6)$$

The posterior mean and variance specify the predictive distribution, which captures both the predicted value and the corresponding uncertainty.

C. Kernels

The kernel specifies the prior assumptions about the function through hyperparameters, which represent its characteristics such as smoothness and length-scale. The kernel design for the active sensing setting was proposed in [10] where a spatial-temporal kernel is defined as a sum of three spatial kernels—a squared exponential kernel k_{SE}, an exponential kernel k_{EXP}, and a constant kernel—representing the local stationarity of the RSS map, multiplied by a temporal Matérn kernel k_{MAT}.

This kernel was kept the same as in [10] for the EI algorithm. The Matérn kernel is omitted when UCB, KG, and Thompson sampling are used. In these three cases, the remaining kernels are provided with both spatial and temporal inputs ($\mathbf{x_t}$ and $\mathbf{t}$) as this configuration resulted in better accuracy compared to the EI setting.

For the considered sensor scheduling and tracking task, we define the following composite kernel:

$$k((\mathbf{x_t}, \mathbf{t}), (\mathbf{x'_t}, \mathbf{t}')) = \big(k_{\text{BIAS}}((\mathbf{x_t}, \mathbf{x'_t}), (\mathbf{t}, \mathbf{t}')) + k_{\text{SE}}((\mathbf{x_t}, \mathbf{x'_t}), (\mathbf{t}, \mathbf{t}')) + k_{\text{EXP}}((\mathbf{x_t}, \mathbf{x'_t}), (\mathbf{t}, \mathbf{t}'))\big) \cdot k_{\text{CONST}}((\mathbf{x_t}, \mathbf{x'_t}), (\mathbf{t}, \mathbf{t}')), \quad (7)$$

where

$$k_{\text{SE}}((\mathbf{x_t}, \mathbf{x_t}'), (\mathbf{t}, \mathbf{t}')) = \sigma_t^2 \exp(-\frac{\|\mathbf{x_t} - \mathbf{x_t}'\|^2}{2\ell_t^2}), \quad (8)$$

$$k_{\text{EXP}}((\mathbf{x_t}, \mathbf{x_t}'), (\mathbf{t}, \mathbf{t}')) = \sigma_t^2 \exp(-\frac{\|\mathbf{x_t} - \mathbf{x_t}'\|}{\ell_t}), \quad (9)$$

$$k_{\text{BIAS}}((\mathbf{x_t}, \mathbf{x_t}'), (\mathbf{t}, \mathbf{t}')) = b, \quad (10)$$

$$k_{\text{CONST}}((\mathbf{x_t}, \mathbf{x_t}'), (\mathbf{t}, \mathbf{t}')) = c. \quad (11)$$

Here, σ^2 denotes the amplitude scaling parameter, ℓ the length-scale parameter, b denotes a bias in the measurements and c denotes a scaling constant; $||.||$ denotes the Euclidean norm.

D. Hyperparameter Learning

The learning of the kernel hyperparameters of the GP is achieved by maximising the log marginal likelihood (LML) which enables efficient numerical calculations. Given the data $\mathbf{X}_t, \mathbf{z}_t$, the likelihood $p(\mathbf{z}_t|f(\mathbf{X}_t))$, the prior distribution $p(f(\mathbf{X}_t)|\mathbf{X}_t, \boldsymbol{\theta})$ and the hyperparameters vector $\boldsymbol{\theta}$, the marginal likelihood is defined as follows:

$$p(\mathbf{z}_t \mid \mathbf{X}_t, \boldsymbol{\theta}) = \int p(\mathbf{z}_t \mid f(\mathbf{X}_t))\, p(f(\mathbf{X}_t) \mid \mathbf{X}_t, \boldsymbol{\theta})\, df(\mathbf{X}_t), \quad (12)$$

Here $\mathbf{X}_t, \boldsymbol{\theta}$ comprises the hyperparameters of all kernels specified in (7). The LML can be computed using the closed-form equation:

$$\log p\left(\mathbf{z}_t \mid \mathbf{X}_t, \boldsymbol{\theta}\right) = -\frac{1}{2}\mathbf{z}_t^\top \left[\mathbf{K}_t + \sigma_n^2\mathbf{I}\right]^{-1} \mathbf{z}_t - \frac{1}{2}\log\left|\mathbf{K}_t + \sigma_n^2\mathbf{I}\right| - \frac{n_t}{2}\log(2\pi). \quad (13)$$

Here n denotes the number of observations. The minimisation of the negative LML is performed using gradient-based methods provided by the SciPy optimiser [17] and the GPyTorch library [18], both used in this work. Specifically, we employ the default $Broyden–Fletcher–Goldfarb–Shanno$ algorithm, which typically finds near-optimal values for all hyperparameters at each timestep.

III. Proposed Methods

In this work, we evaluate the performance and suitability of several well-established acquisition functions, namely the EI, UCB, KG, and TS—for their effectiveness in guiding the search using the GP surrogate in a target tracking setting.

In the proposed framework, the UAVs select their sampling positions sequentially, with each UAV's choice informed by the positions chosen by the preceding UAVs. The GP is first trained using 30 randomly sampled positions on the two-dimensional spatial grid. After this initial training, a three-dimensional grid incorporating the time dimension is constructed, marking the start of the UAVs' exploration of the environment by visiting locations selected by the acquisition function. The sensors move in discrete steps, with no explicit motion model or physical constraints such as velocity or inertia.

After each observation, the GP model is rebuilt—meaning a new instance is created using the updated dataset—refining the predictive distribution used to guide the next UAV's decision. This sequential update strategy enables improved performance without requiring joint planning, offering both simplicity and reduced computational cost. Furthermore, performance is enhanced by an optimiser that tunes the GP kernel hyperparameters.

A. Acquisition Functions

Acquisition functions are chosen based on theoretical considerations and physical aspects and they are used in Bayesian optimisation (BO) to guide the search for the optimum based on the uncertainty in the posterior [19]. These functions typically have analytical forms that are inexpensive to evaluate. Even when a closed-form expression is not available, they can be approximated efficiently. In either case, acquisition functions are significantly easier to optimise than the original objective function.

Most existing acquisition functions focus either on maximising the improvement between the current and next query, such as EI or probability of improvement [20], or on maximising knowledge gain by reducing uncertainty, as exemplified by the KG policy [21] and entropy search [22]. TS, on the other hand, draws a sample from the posterior distribution at each candidate point and then selects the point corresponding to the maximum sampled value.

These algorithms balance exploration and exploitation, a trade-off that is fundamental to the effectiveness of BO. Exploration targets regions with high model uncertainty, whereas

exploitation focuses on areas with high predicted objective values.

1) *Expected Improvement* [11] is a widely applied acquisition function due to its simplicity and computational efficiency. The algorithm is able to naturally balance exploration with exploitation, however, in this setting, we follow the common approach to introduce the ξ parameter to control it explicitly.

Following [10], the EI acquisition function used in the active sensing framework is defined as:

$$\mathrm{EI}(\mathbf{x}_t, t) = \begin{cases} \mathcal{I}(\mathbf{x}_t, t)\,\Phi(Z(\mathbf{x}_t, t)) \\ \quad + \sigma(\mathbf{x}_t, t)\,\phi(Z(\mathbf{x}_t, t)), & \sigma(\mathbf{x}_t, t) > 0, \\ 0, & \sigma(\mathbf{x}_t, t) = 0, \end{cases} \tag{14}$$

$$\begin{aligned} \mathcal{I}(\mathbf{x}_t, t) &= \mu(\mathbf{x}_t, t) - z^* - \xi, \\ Z(\mathbf{x}_t, t) &= \frac{\mathcal{I}(\mathbf{x}_t, t)}{\sigma(\mathbf{x}_t, t)}, \end{aligned} \tag{15}$$

where $\mathcal{I}(\mathbf{x}_t, t)$ is the improvement at the candidate point $(\mathbf{x}_t, t)$, z^* is the best observed value and ξ is a parameter used to balance the exploration-exploitation. Here $\Phi(.)$ is the cumulative distribution function reflecting the improvement, $Z(\mathbf{x}_t, t))$ is the standardised improvement, $\sigma(\mathbf{x}_t, t)$ is the predictive standard deviation and $\phi(.)$ is the probability density function.

After calculating the EI for the initial set of sample points, the point with the highest EI value is selected to guide the search toward areas most likely to improve over the current best observation. A prediction is made at this point using the GP model. In each subsequent iteration, the sample set is expanded with the newly selected points, and the acquisition function is recalculated over the updated set. This process ensures that the next point with the highest EI is always selected based on the most recent model and data, continuously refining the search for the optimum.

2) The *Upper Confidence Bound (UCB)* [14] provides theoretical guarantees [23]. It selects where to sample next by maximising the function of predicted mean μ and the predicted standard deviation σ:

$$\mathrm{UCB}(\mathbf{x_t}, \mathbf{t}; \lambda) = \mu(\mathbf{x_t}, \mathbf{t}) + \lambda\sigma(\mathbf{x_t}, \mathbf{t}). \tag{16}$$

Here λ is a confidence parameter that controls the trade-off between exploration and exploitation. The choice of this parameter is a practical problem which has strong influence on the performance of this algorithm.

3) *Knowledge Gradient Policy* [15] is similar to EI as it evaluates the expected impact of a sample. However unlike EI, it does not assume that the solution needs to be a previously evaluated sample. It also evaluates the expected impact of a sample on the entire posterior distribution, not just the posterior at a sampled point [6]. The expected gain in knowledge is represented by the knowledge gradient as follows:

$$\mathrm{KG}_n(\mathbf{x}_t, t) := \mathbb{E}_n\left[\mu^*_{n+1} - \mu^*_n \mid \mathbf{x}_{t+1} = \mathbf{x}_t, t+1 = t\right], \tag{17}$$

where μ^*_n denotes the maximum expected value of the posterior mean after n observations. When we take an additional sample at point x, we obtain an updated posterior with mean μ_{n+1}, and consequently a new maximum predicted value μ^*_{n+1}. The knowledge gradient represents the expected increase in the maximum predicted value, $\mu^*_{n+1} - \mu^*_n$, after sampling at x, therefore, the next sampling point is chosen as the one with the highest KG value.

The standard KG policy assumes stationarity of the unknown function. This condition may be partially relaxed using a batched version of the KG function, called q-KG, and it is discretised for the decision-space [24]. In our implementation, we also use a spatio-temporal kernel to model both space and time. To further improve the performance of KG for tracking a non-stationary target, we apply a sliding time window to focus on more recent data and introduce a local cubic subset around the best prediction, encouraging the KG to prioritise exploitation.

The KG method is significantly more computationally expensive than previously discussed methods. The conventional approach requires optimisation, typically performed via stochastic gradient ascent. This process was simplified by the one-shot KG method [25], which generates fantasy observations from the GP posterior, and approximates the KG in a single forward pass without retraining the GP for every candidate. Unlike the original approach, it evaluates all samples in parallel, making it more scalable.

4) *Thompson Sampling* [16] is another well-established method which can be applied to a wide variety of problems. The algorithm first computes the posterior distribution for all candidate points, and then draws independent samples from the posterior, typically using Monte Carlo (MC) or quasi-Monte Carlo sampling method such as the Sobol sampling [26]. The sample is then maximised to find the next sampling point, which can be defined as follows:

$$\mathrm{TS}(\mathbf{x}_t, t) = \arg\max_{(\mathbf{x}_t, t) \in \mathcal{X}} \tilde{f}(\mathbf{x}_t, t), \quad \tilde{f}(\mathbf{x}_t, t) \sim p\left(f(\mathbf{x}_t, t) \mid \mathcal{D}_t\right). \tag{18}$$

where $\tilde{f}(\mathbf{x}_t, t)$ is a sample from the posterior distribution at $(\mathbf{x}_t, t)$, and $\mathcal{X}$ is an input space over both spatial and temporal domains.

The Sobol sampler [26] uses Sobol sequences which are more evenly distributed than random independent and identically distributed samples, offering improved performance with respect to standard sampling. Since TS does not require solving an optimisation problem, it offers good computational efficiency. However, on large grids, drawing samples from the posterior becomes computationally expensive, so approximate sampling methods should be considered.

IV. Performance Evaluation

A. Implementation and Simulation Settings

A standard GP regression model provided by the GPFlow library [27] is used to evaluate the EI and UCB algorithms, while a BoTorch [28] `CustomKernel` model is employed to

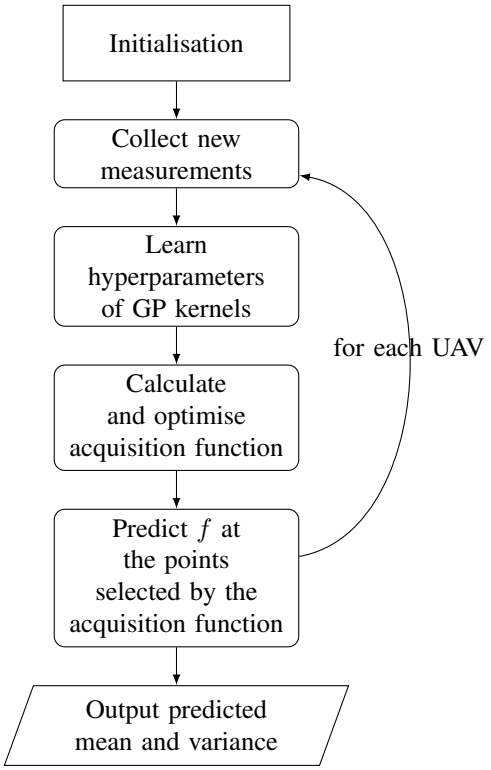


Figure 2. Flowchart of the Bayesian optimisation algorithm

assess the KG and TS algorithms. A spatio-temporal kernel, defined in Equation (7), is used to capture the characteristics of the considered sensor scheduling and tracking task. The parameters of the acquisition functions are selected empirically. For EI, the ξ parameter is set to 0.01. The λ parameter in UCB is set to 1.5.

The simulation setup follows that presented in [10]. The algorithm is evaluated over an area of $400\text{m} \times 400\text{m}$, discretised into a grid with a step size of 3m. The target is initialised at position [50 m, 50 m] and moves linearly with velocities of $1\,m/s$ in both x and y directions. The obtained results are averaged over 100 Monte Carlo simulations with two levels of measurement noise in the RSS sensor (1dB and 5dB).

A single GP algorithm with real-time learning of the hyperparameters of its composite kernel is implemented, as illustrated in Figure 2. The algorithm operates by collecting a five-second data window, training the GP kernels to estimate their hyperparameters, and then computing and maximising the acquisition function to determine the next measurement point. Subsequently, the mean and variance of $f(\mathbf{x})$ are predicted using the GP equations (5) and (6) at the locations selected by the acquisition function. This information is then used to identify the region where the target is most likely to be located. The new measurement is added to the dataset, and the updated dataset is shared among the remaining UAVs to determine their respective sampling locations. Once all UAVs have collected their measurements, the combined dataset is used to retrain the GP, refining the estimate of the target's location.

The optimisation of the KG and TS acquisition functions is performed within the BoTorch framework [28]. For the EI and UCB acquisition functions, the GPFlow model is optimised using a SciPy-based optimiser. In both cases, the kernel hyperparameters are tuned by minimising the LML defined in Equation (13).

We consider the EI acquisition function as proposed in [10] as a baseline and compare the UCB, KG and Thompson sampling algorithms to it. We have kept the kernel as originally proposed in [10] (Equations (10)-(14)) for EI.

Table I
AVERAGE RUNTIMES AND DISTANCE ERRORS FOR DIFFERENT ACQUISITION FUNCTIONS WITH VARYING NUMBERS OF UAVS UNDER TWO NOISE LEVELS.

Acquisition Function	Number of UAVs	Average Runtime (s)		Average Distance Error (m)	
		Noise 1dB	Noise 5dB	Noise 1dB	Noise 5dB
EI	1 UAV	51.11	53.99	54.65	95.38
	2 UAVs	85.65	86.36	35.48	46.16
	3 UAVs	157.98	159.93	27.59	31.58
UCB	1 UAV	364.50	424.32	11.47	60.05
	2 UAVs	1546.92	2669.54	10.06	29.83
	3 UAVs	3094.48	3280.41	9.52	14.92
KG	1 UAV	217.62	262.42	47.64	123.26
	2 UAVs	261.32	512.47	31.95	117.99
	3 UAVs	261.26	261.32	30.04	111.85
Thompson sampling	1 UAV	292.74	242.12	19.23	29.88
	2 UAVs	963.97	937.65	14.28	18.70
	3 UAVs	1610.69	1810.15	12.88	22.63

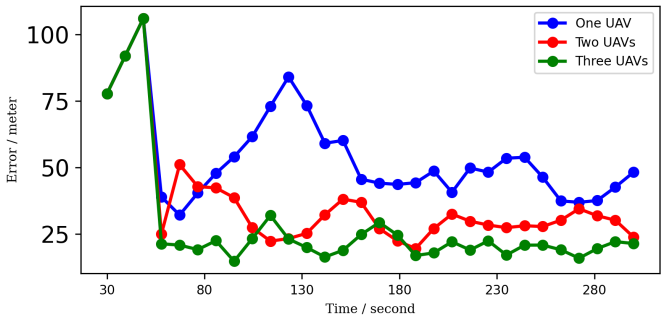
B. Results

The performance of the acquisition functions is assessed in terms of accuracy and runtime under two levels of noise and a varying number of UAVs. In Table I, the numerical results are presented for the benchmark algorithm, EI, compared to UCB, KG and TS. The runtime is measured on an Intel® Core™ i9-14900 CPU with 64 GB RAM.

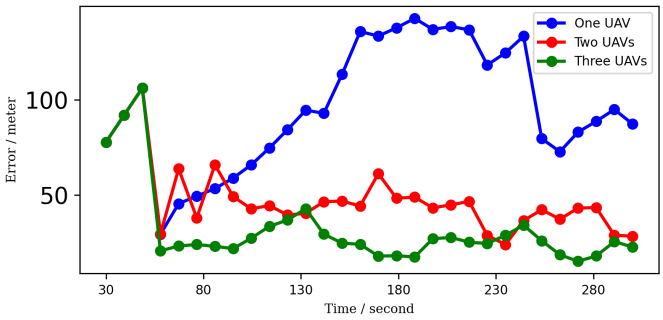
The distance error shown in Table I and Figure 3 is the Euclidean distance between the UAV position and the target position at each timestep, averaged over the entire simulation and then further averaged across 100 simulations.

Table I shows a clear trend indicating that, for most methods, runtime increases significantly as accuracy improves. Furthermore, at a higher noise level, the runtime increases while the accuracy decreases.

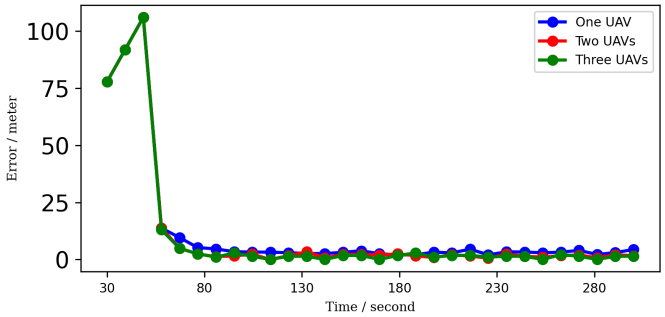
Although EI offers the most favourable trade-off between accuracy and computational cost, maintaining competitive accuracy while achieving the lowest average runtime, UCB achieves the lowest average distance error at the lower noise level, and TS achieves the lowest average distance error at the



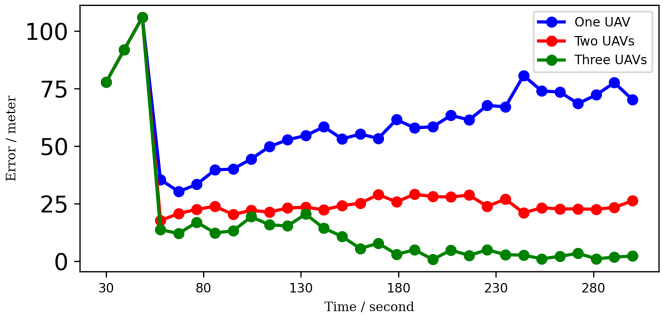
(a) Expected improvement, $\sigma = 1dB$



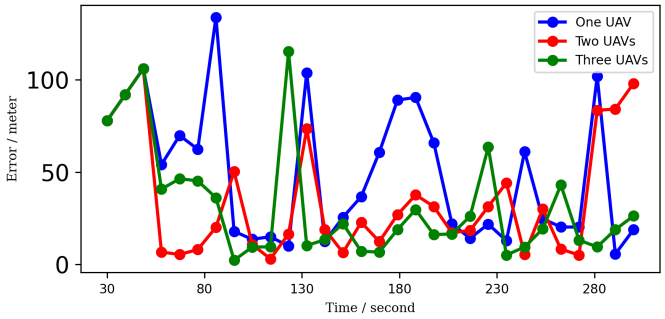
(b) Expected improvement, $\sigma = 5dB$



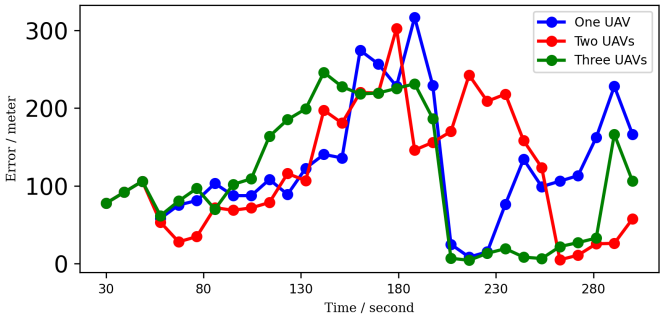
(c) Upper confidence bound, $\sigma = 1dB$



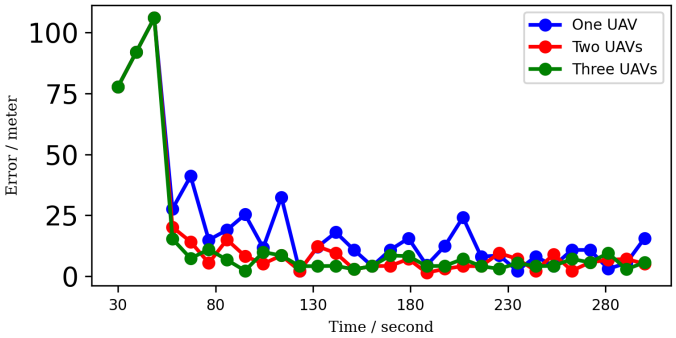
(d) Upper confidence bound, $\sigma = 5dB$



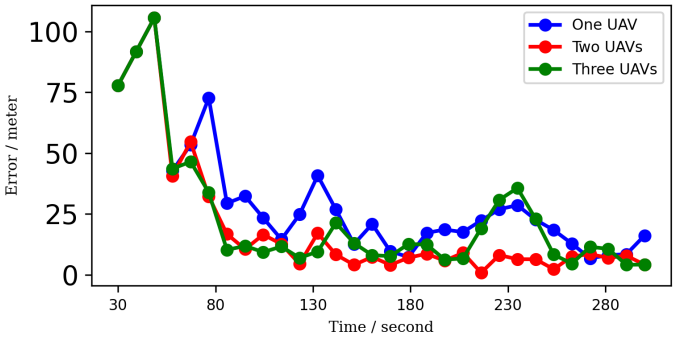
(e) Knowledge gradient, $\sigma = 1dB$



(f) Knowledge gradient, $\sigma = 5dB$



(g) Thompson sampling, $\sigma = 1dB$



(h) Thompson sampling, $\sigma = 5dB$

Figure 3. Distance error between predicted and real target location over time

higher noise level. Across both noise levels, UCB produces results that are on average approximately 54% more accurate than EI, while TS is approximately 60% more accurate. In contrast, KG demonstrates the poorest balance between accuracy and runtime.

The tracking performance of the methods is visualised in Fig. 3, where the distance between the target position and the predicted target position is plotted. For two and three UAVs, the results are averaged into a single line on the graph. It is evident that the first few iterations of all algorithms are performed on the same initial sample set, resulting in identical initial distance errors across all acquisition functions. This persists until the UAVs begin selecting different points based on the new data they collect.

As the noise level increases, the acquisition functions are more likely to select suboptimal locations due to the reduced reliability of the measurements, occasionally causing the UAV to follow local maxima.

V. Conclusions

This work proposes a GP framework for sensor scheduling and target tracking based on a composite kernel and different acquisition functions: 1) EI, 2) UCB, 3) TS and 4) KG. While the first three methods perform reliably and offer different strengths, KG shows reduced effectiveness in an active sensing setting with a non-stationary objective. While the UCB algorithm achieves the lowest average distance error for both levels of noise, the average runtime is significantly higher than that of EI and KG. EI offers the best trade-off between accuracy and runtime. TS offers competitive performance while KG gives the least accurate results.

Future work will focus on multi-task GP methods that consider cross-correlations in the kernel representations and scalability aspects. Different approaches for learning of the kernel hyperparameters will also be considered.

References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[2] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proceedings F: Radar and Signal Processing*, vol. 140, no. 2, 1993, pp. 107–113.

[3] W. Li, Z. Wang, G. Wei, L. Ma, J. Hu, and D. Ding, "A survey on multisensor fusion and consensus filtering for sensor networks," *Discrete Dynamics in Nature and Society*, vol. 2015, pp. 1–12, 2015.

[4] Y. Li, J. Lou, X. Tan, Y. Xu, J. Zhang, and Z. Jing, "Adaptive kernel learning Kalman filtering with application to model-free maneuvering target tracking," *IEEE Access*, vol. 10, pp. 78 088–78 101, 2022.

[5] H. Wu, S. Song, K. You, and C. Wu, "Depth control of model-free AUVs via reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 12, pp. 2499–2510, 2018.

[6] P. I. Frazier, "A tutorial on Bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[7] R. Marchant and F. Ramos, "Bayesian optimisation for intelligent environmental monitoring," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2242–2249.

[8] R. Oliveira, L. Ott, V. Guizilini, and F. Ramos, "Bayesian optimisation for safe navigation under localisation uncertainty," in *Springer Proceedings in Advanced Robotics*, N. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, Eds., 2019, pp. 489–504.

[9] J. Kaiser, C. Xu, A. Eichler, A. Santamaria Garcia, O. Stein, E. Bründermann, W. Kuropka, H. Dinter, F. Mayet, and T. Vinatier, "Reinforcement learning-trained optimisers and Bayesian optimisation for online particle accelerator tuning," *Scientific Reports*, vol. 14, no. 1, p. 15733, 2024.

[10] X. Liu and L. Mihaylova, "Active sensing for target tracking: A Bayesian optimisation approach," in *Proceedings of the 27th International Conference on Information Fusion (FUSION)*. IEEE, 2024, pp. 1–7.

[11] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[12] Y. He and K. P. Chong, "Sensor scheduling for target tracking in sensor networks," in *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC)*, vol. 1. IEEE, 2004, pp. 743–748.

[13] S.-J. Lee, S.-S. Park, and H.-L. Choi, "Potential game-based non-myopic sensor network planning for multi-target tracking," *IEEE Access*, vol. 6, pp. 79 245–79 257, 2018.

[14] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 397–422, 2002.

[15] P. I. Frazier, W. B. Powell, and S. Dayanik, "A knowledge-gradient policy for sequential information collection," *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2410–2439, 2008.

[16] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[17] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[18] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Advances in Neural Information Processing Systems*, 2018.

[19] J. T. Wilson, R. Moriconi, F. Hutter, and M. P. Deisenroth, "The reparameterization trick for acquisition functions," *arXiv preprint arXiv:1712.00424*, 2017.

[20] H. J. Kushner, "A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise," *Journal of Basic Engineering*, vol. 86, no. 1, pp. 97–106, 1964.

[21] I. O. Ryzhov, W. B. Powell, and P. I. Frazier, "The knowledge gradient algorithm for a general class of online learning problems," *Operations Research*, vol. 60, no. 1, pp. 180–195, 2012.

[22] P. Hennig and C. J. Schuler, "Entropy search for information-efficient global optimization," *Journal of Machine Learning Research*, vol. 13, pp. 1809–1837, 2012.

[23] S. Takeno, Y. Inatsu, and M. Karasuyama, "Randomized Gaussian process upper confidence bound with tighter Bayesian regret bounds," in *Proceedings of the International Conference on Machine Learning*, 2023, pp. 33 490–33 515.

[24] D. Lee, "Knowledge gradient: capturing value of information in iterative decisions under uncertainty," *Mathematics*, vol. 10, no. 23, p. 4527, 2022.

[25] J. Ungredda, M. Pearce, and J. Branke, "Efficient computation of the knowledge gradient for Bayesian optimization," *arXiv preprint arXiv:2209.15367*, 2022.

[26] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112, 1967.

[27] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman, "GPflow: A Gaussian process library using tensorflow," *Journal of Machine Learning Research*, vol. 18, no. 40, pp. 1–6, 2017.

[28] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "Botorch: A framework for efficient Monte-Carlo Bayesian optimization," in *Advances in Neural Information Processing Systems, vol. 33*, 2020.