Active sensing of spatiotemporal phenomena with a UAV

Martin Asenov

Master of Science by Research
Center for Doctoral Training in
Robotics and Autonomous Systems
School of Informatics
University of Edinburgh
2017

Abstract

Actively steering a drone to maximize the information collected from an on-board sensor is largely unused despite the development of necessary technology. In this report we discuss the different challenges associated with active sensing on a UAV and present example applications. We use Gaussian Processes to model the underlying phenomena in the observed field.

In the first set of experiments, we try to detect and localize multiple light sources by using simple light sensor mounted on the UAV. We also demonstrate an integration of our system with a novel broadband spectrometer sensor for detection and characterization of gases developed at Heriot-Watt University.

The work presented here provides the foundations for using UAV as a general sensing platform in an intelligent way aiming to maximize the information collected during flight.

Acknowledgements

First and foremost, I would like to thank my supervisor Dr. Subramanian Ramamoorthy for his support and guidance.

Next, I want to thank Prof. Derryck T. Reid, Carson Vogt and Marius Rutkauskas for their help on the project, as well as all Ph.D. students from InSpace for the numerous discussions.

I would also like to thank my parents for their unconditional support and understanding.

This work was supported in part by the EPSRC Center for Doctoral Training in Robotics and Autonomous System as well as the Defence Science and Technology Laboratory (Dstl).

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Martin Asenov*)

Table of Contents

Chapter 1

Introduction

1.1 Motivation

Sensing and gathering information about different physical phenomena is a key requirement for environmental and industrial applications. Examples include measuring temperature, wind speed and gases. Drones have seen increased usage as a sensing platform for gathering those types of readings as the technology matures.

Typically, any processing of the data is done offline after the UAV has finished collecting data. A problem that arises is that, due to battery limitations, often the flight time is relatively short. This is an issue when one wants to make measurements across a big field, as it is not possible to record enough data within the time constraints. Possible solutions include recharging or replacing the drone battery once it runs out or using multiple drones. However, those solutions bring their own set of problems. Using multiple drones can be costly and synchronization between them is needed. Returning to a base station for power in order to acquire more measurements could be ineffective, especially when measuring dynamical processes. By the time the battery is replaced or recharged, the dynamics of the observed process could have changed. Moreover, in certain applications it is critical to find the extreme points of the measured phenomena in the fastest way possible.

An alternative solution to gathering data is to exploit the structure of the observed problem. Physical phenomena often have spatial and temporal correlations. For example, when measuring the temperature, it is quite likely that it will be very similar at the same location after 5 minutes, or at a location a few meters away at the same time. Although, wind and gases in an open space have certain dynamics, similar assumptions can be made to a certain extent. Often useful prior knowledge about the observed

phenomena are available - what are the average sensor readings one should expect to get, how much do they vary, etc. Finally, it is also possible to measure the noise of the sensor, given a constant sensory input.

1.2 Project

The project aims to explore the possibility of using a drone as an active sensing platform, incorporating biases about the observed phenomena. In many natural events, for example, one can exploit temporal and spatial correlations. This allows to fit a probability model over the entire area of interest and estimate mean and variance at each point. With that knowledge, the next prediction point can be picked effectively, based on mean extrema, variance extrema or combination of both. Within very few sensor readings, taking into account the imposed structure and efficient sampling, the entire field of interest could be mapped or the global extrema found.

In order to effectively model a specified field we incorporated those assumptions and structure of the problem into a Gaussian Processes Regression. In addition, for optimal selection of the next prediction point, a Bayesian optimisation framework was integrated. To test this approach we used an off-the-shelf light sensor mounted on an UAV in order to locate multiple sources of light. A broadband gas spectrometer sensor, developed by collaborators from Heriot-Watt University, was tested as well.

The outcomes of this projects can be summarized as follows:

- Developed a framework for autonomous flights and active sensing based on Gaussian Processes and Bayesian optimisation
- Software integration of a chemical and a light sensor within the UAV platform
- Performed autonomous active sensing flights with the light sensor
- Critically evaluated existing methods on our drone system and discovered areas for potential improvement

1.3 Realization and Document Structure

The rest of the dissertations is divided into 5 sections.

Chapter 2

This chapter provides background information about Gaussian Processes and Bayesian optimisation. It describes how they are constructed and following sections give more details about different choices when using those two frameworks and their significance.

Chapter 3

This chapter comments on previous related work in active sensing of different natural phenomena using a variety of robotics platforms. The usage of drones as sensing platform is discussed. Finally, the chapter gives information previous work in detection hazardous gases.

Chapter 4

This chapter describes the design decision for our drone, on-board computer and the differences between the two sensors, light and chemical, used in the project. It then describes the software stack developed and some of the logistics of conducting flights, which motivated the conducted set of experiments.

Chapter 5

This chapter describes the experiments conducted and their significance.

Chapter 6

The project is summarized in this chapter and future work discussed.

Chapter 2

Background

Gaussian Processes are heavily used within the active sensing community. This chapter aims to describe some of their useful properties [1]. Different aspect are discussed including model selection, sampling strategy, optimisations when doing inference.

2.1 Gaussian Processes

2.1.1 Definitions

In this section Gaussian Processes ($\mathcal{GP}$) are described by providing a few definitions, describing their mathematical model and arguing why they are a good approach to model the specified problem.

Definition 2.1.1 *A Gaussian process is a collection of random variables with the property that the joint distribution of any finite subset is a Gaussian.*

Definition 2.1.2 *A Gaussian process is fully specified by a mean and a covariance function.*

Definition 2.1.3 *A Gaussian process is a distribution of functions with certain properties, defined by its mean and covariance kernel functions.*

The first definition expresses the mathematical structure of a $\mathcal{GP}$ - it is nothing more than a multivariate Gaussian (normal) distribution, with the property that any selection of its random variables is a normal distribution as well. The second definition gives a better idea of how the distribution can be used to model regression problems.

[1] Parts of the analysis were summarized from my report for ASR.

The first part of the second definition explains how to construct the multivariate Gaussian - by a mean and covariance function. The mean function is used for generating the mean vector of the distribution, while the covariance function generates the covariance matrix. After defining the mean vector and covariance matrix of a multivariate Gaussian, one can draw samples from it. Each sample drawn is visualised in a special way and represents a function. This is exactly what is expressed in the third definition. Given a mean and covariance kernel, one can construct a distribution of functions. Those function have special properties, based on the mean and covariance functions used for generating the multivariate Gaussian.

2.1.2 Construction

$\mathcal{GP}$ are constructed by using a Gaussian distribution, which is defined in eq.2.1 in the univariate case and eq.2.2 in the multivariate case.

$$N(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.1}$$

$$N(x|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{2.2}$$

The univariate Gaussian has two hyperparameters - the mean μ and variance σ^2. The mean is the center of mass of the distribution, while the variance controls how the probability mass is distributed. Alternative way to think about the variance is what is the certainty about the mean value - the higher the variance the more samples are further away from the mean. Similarly, this can be extended to multivariate Gaussian, where there is a mean vector with the mean value for each dimension. As for the variance, this is expressed in a covariance matrix, where in addition to the variance for each dimension there is also the correlations between them.

The next step in constructing $\mathcal{GP}$ is drawing samples from a chosen Gaussian distribution and visualizing them in a specific way. An example approach is shown in fig.2.1, but there are many other and more efficient ways of sampling a multivariate normal distribution [5] [22]. Going back to def.2.1.3 of the $\mathcal{GP}$, one can use the multivariate normal distribution in order to sample functions. A sample from an n-dimensional Gaussian has the form of $X_1, X_2, \ldots X_n$, which is a point in n-dimensional space. Each point can be plotted, with its dimension "number label" on the X axis and the values for those dimensions on the Y axis. In this way each sample is represented

by n points, which if $n \to \infty$ is exactly a function - mapping between one set to another set.

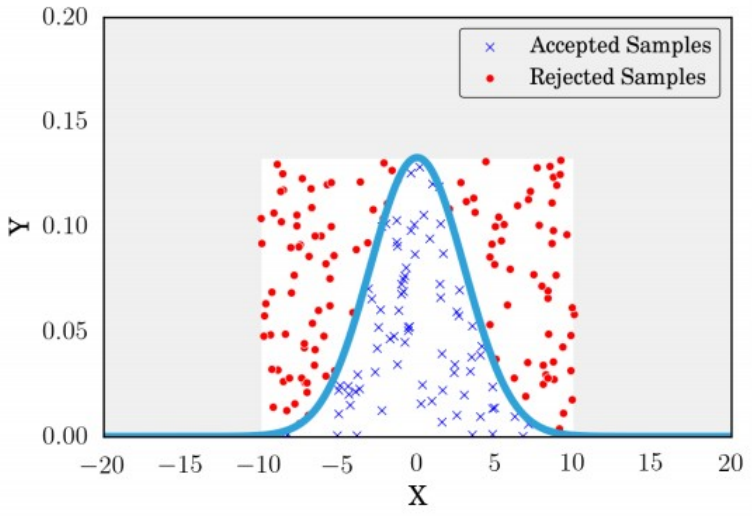


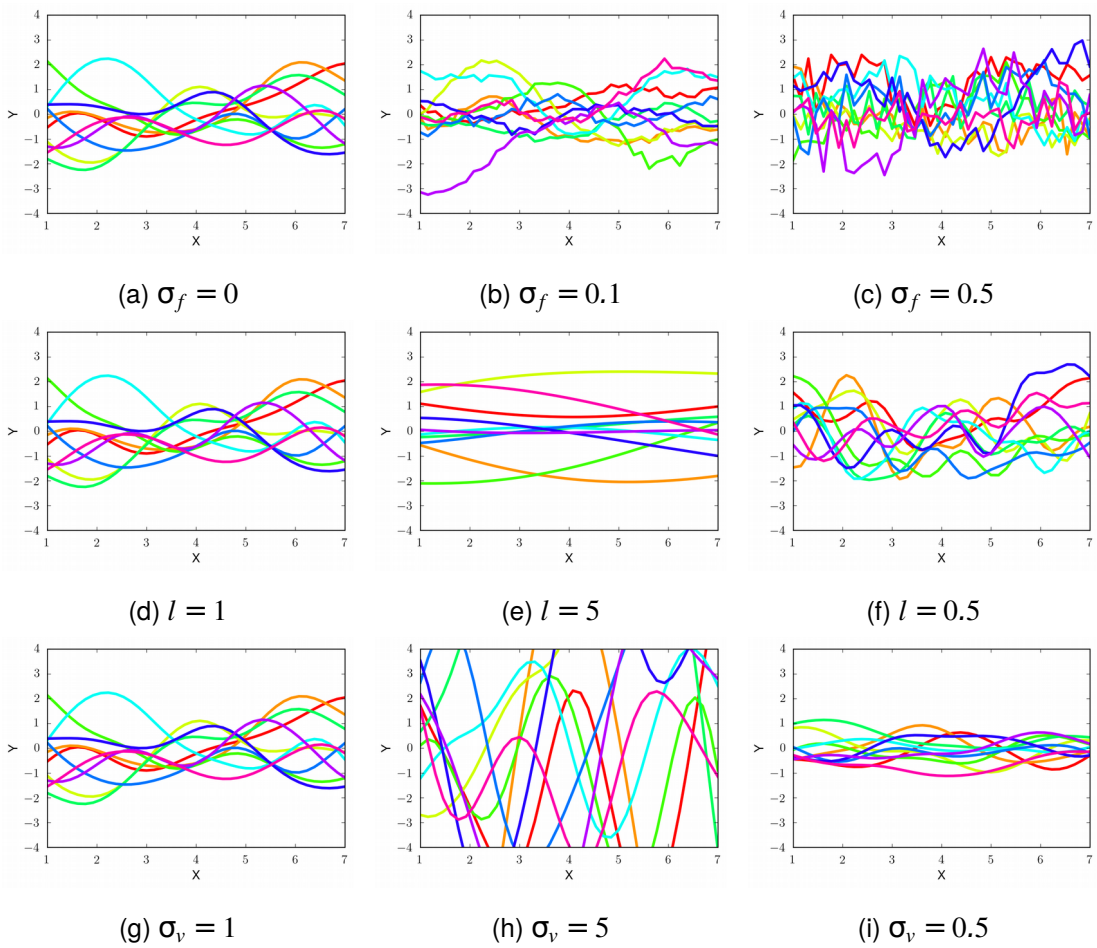
Figure 2.1: Rejection Sampling.

To specify the properties of a distribution of functions, a mean and covariance kernels must be defined. The mean kernel is used to generate the mean vector of the Gaussian and the covariance kernel to generate the covariance matrix. The most common kernel used is the squared exponential kernel for the covariance matrix, defined in eq.2.3 and eq.2.4 and zero function for the mean vector. By using this kernel one can generate smooth functions.

$$cov(y_n, y_{n'}) = k(x_n, x_{n'}) + \sigma_y^2 \delta_{nn'} \tag{2.3}$$

$$k(x_n, x_{n'}) = \sigma_f^2 exp(-\frac{1}{2l^2}(x_n - x_{n'})^2) \tag{2.4}$$

The kernel is general in the sense that it can generate as big of a covariance matrix as wanted for a $\mathcal{GP}$. The dimension of the covariance matrix is often misinterpreted to be dependent on the number of observations. In fact this is not the case, it is just the accuracy with which the function is represented. The squared exponential kernel is defined by three hyperparameters - noise level, horizontal lengthscale and vertical lengthscale. The effect they have on the generated functions can be seen in fig.2.2.



(a) $\sigma_f = 0$ (b) $\sigma_f = 0.1$ (c) $\sigma_f = 0.5$

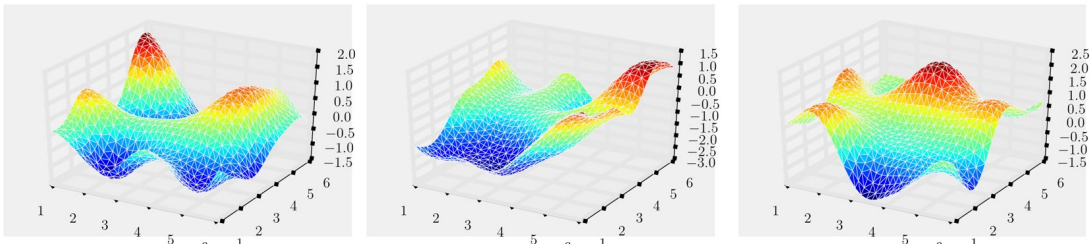
(d) $l = 1$ (e) $l = 5$ (f) $l = 0.5$

(g) $\sigma_v = 1$ (h) $\sigma_v = 5$ (i) $\sigma_v = 0.5$

Figure 2.2: Effect of varying different hyperparameters of the squared exponential kernel. (a), (b) and (c) Effect on varying the noise level σ_f. As we increase σ_f we introduce more noise in our functions. (d), (e) and (f) Effect on varying the horizontal length scale l. As we increase l the functions become smoother. (g), (h) and (i) Effect on varying the vertical length scale σ_v. As we increase σ_v the functions start to take wider range of values.

So far in the discussion the focus was on describing how to model functions in the one dimensional case. However, the kernel function can be easily extended in order to generate functions taking higher dimensional input. In the example of squared exponential, the only modifications one has to do is in eq.2.4 which becomes eq.2.5. This becomes relevant in the project as modelling of two to four dimensional functions is needed.

$$k(x_n, x_{n'}) = \sigma_f^2 exp(-\sum_{d=1}^{D} \frac{1}{2l^2}(x_{dn} - x_{dn'})^2) \tag{2.5}$$



(a) (b) (c)

Figure 2.3: Modelling $2D$ regression using Gaussian Processes and squared exponential kernel. By extending the kernel as in eq.2.5 one can model higher dimensional functions. As before every sampled function, the surfaces in this case, are one sample from a multidimensional normal distribution visualized as described before.

Going back to def.2.1.3, this section has successfully described $\mathcal{GP}$ as a distribution of functions dependant on a covariance and on a mean kernel. This can be expressed as in eq.2.6, where m is the mean kernel and k is the covariance kernel.

$$f \sim GP(m, k) \tag{2.6}$$

2.1.3 Inference

So far I described how to create a distribution of functions with certain properties. In order to use $\mathcal{GP}$ in problems like regression, one still has to define how the functions go through the data. The Bayes' theorem is used, taking advantage of the fact that the normal distribution is self-conjugate, allowing to compute a closed-form solution of the posterior.

Eq.2.6 can be expressed in an alternative form as in eq.2.7, where $m = 0$ and $\Sigma(X_*, X_*)$ is the resulting matrix of the squared exponential kernel k.

$$f_* \sim N(0, \Sigma(X_*, X_*)) \tag{2.7}$$

A set of data points $\{x_n, y_n\}_{n=1}^{N}$ is introduced. In a vector form this can be represented as $\{X, f\}$. Using X_* and X, the joint Gaussian distribution can be defined as in eq.2.8.

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} \Sigma(X, X) & \Sigma(X, X_*) \\ \Sigma(X_* X) & \Sigma(X_*, X_*) \end{bmatrix}\right) \tag{2.8}$$

The final step to compute the posterior, is to calculate the conditional distribution of functions as in eq.2.9. Since the two Gaussian distributions are conditioned, the posterior will also be a Gaussian distribution.

$$f_*|X_*, X, f \sim N(\mu_{f_*}, \Sigma_{f_*}) \tag{2.9}$$

Calculating the mean vector μ_{f_*} and covariance matrix Σ_{f_*} is derived in great details in [3] (sec. 2.3.1). The results can be seen in eq.2.10, 2.11.

$$\mu_{f_*} = \Sigma(X_*, X)\Sigma(X, X)^{-1} f \tag{2.10}$$

$$\Sigma_{f_*} = \Sigma(X_*, X_*) - \Sigma(X_*, X)\Sigma(X, X)^{-1}\Sigma(X, X_*) \tag{2.11}$$

A simple example of nonlinear regression with $\mathcal{GP}$ can be seen in fig.2.4.

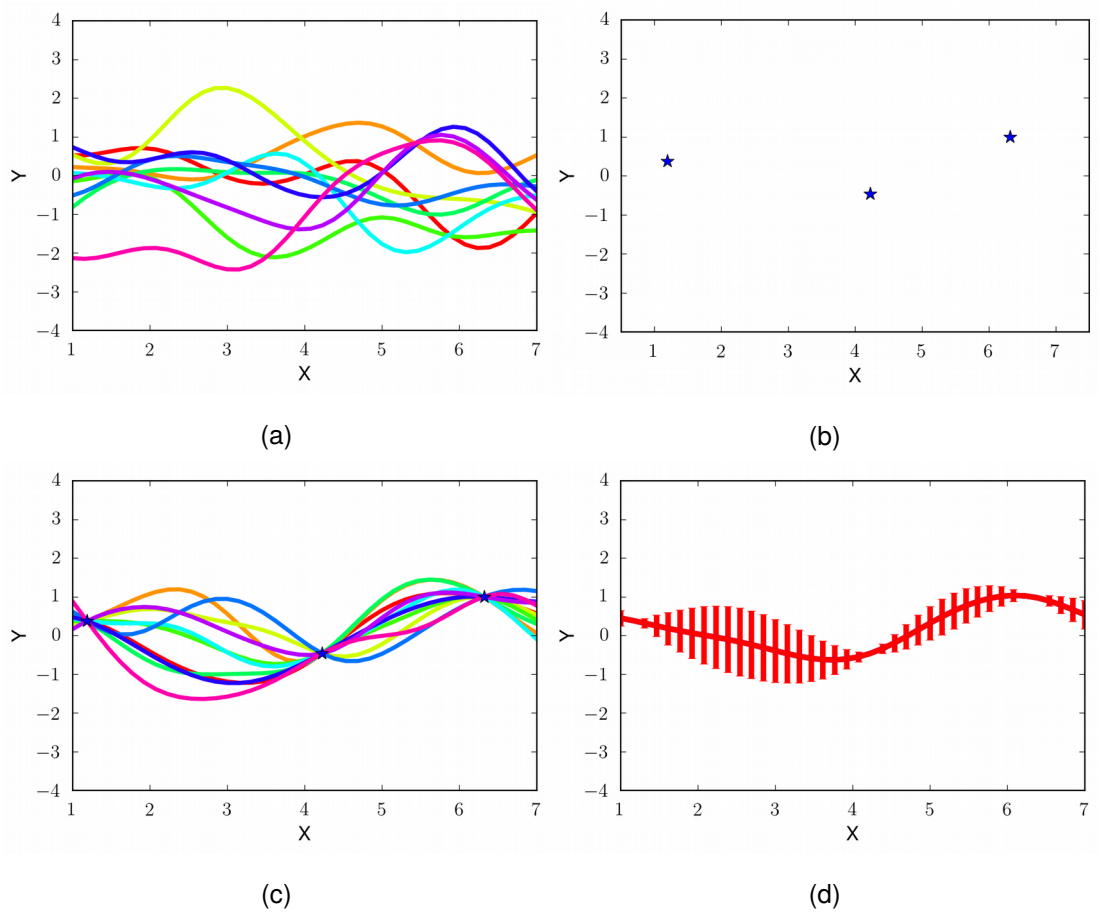


Figure 2.4: Inference in $\mathcal{GP}$. (a) Samples from the prior distribution of functions (b) Introduced data (c) Samples from the posterior distribution of functions. The functions preserved their smooth properties, while fitting the data (d) As more function samples of the posterior are drawn, one can measure the uncertainty of the observed field.

Again this can be easily extended to a higher dimensional data. In fig.2.5 an example of 2D regression can be seen. All three of the surfaces go through the two data points while preserving their smooth properties. Within this project inference in 3D hyperplanes is used.

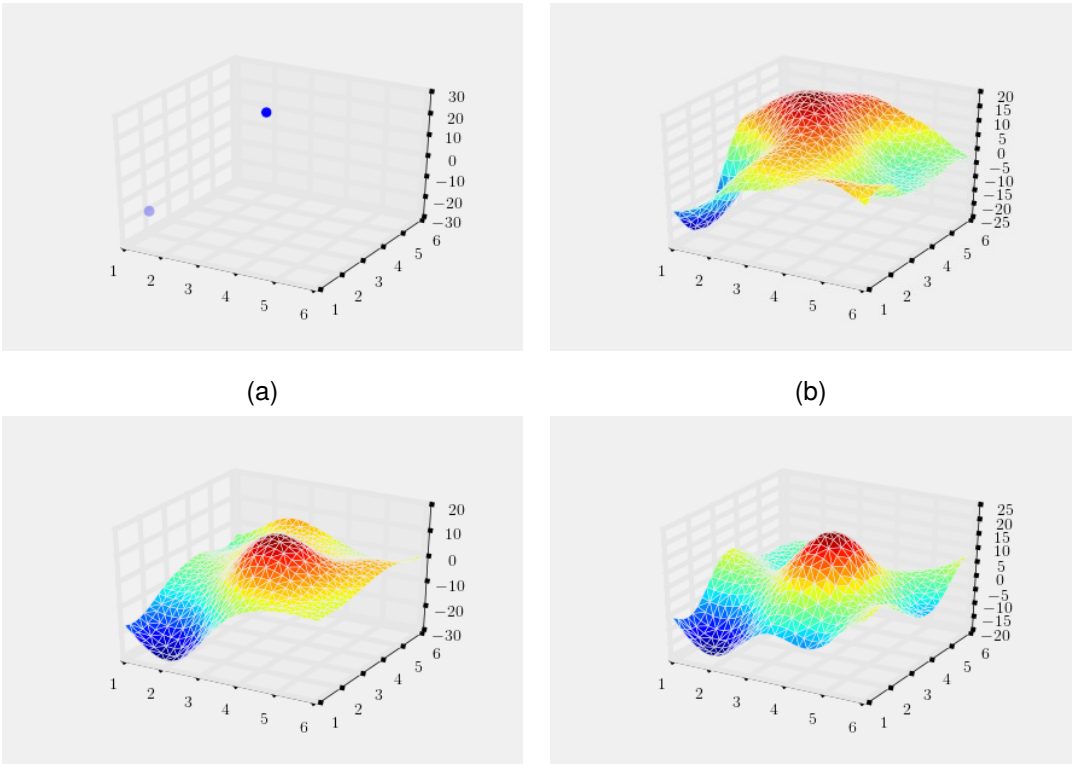


Figure 2.5: Inference in $2D$ $\mathcal{GP}$. (a) Introduced data points (b), (c), (d) Functions samples drawn from the posterior. All the smooth surfaces go through the introduced points (red and blue peaks).

2.1.4 Model selection, hyperparameter estimation and Bayesian optimization

There are multiple choices one could make when constructing a $\mathcal{GP}$ regression model - model selection, hyperparameter estimation, sampling strategy, optimisation techniques for speeding up the inference, etc.

First, the type of kernel has to be selected. This characterises the types of functions that are going to be sampled - their local and global structure, whether they are periodic, etc. In the problems modelled in this report, the assumption made is that the concentration of gases/intensity of light is a smooth function of time and space. When a measurement is taken, it is likely that the measurement will be similar after a couple of seconds, or if it is taken at some small distance away. Therefore, the squared exponential kernel is going to be used. This is further supported by collecting some information before running the active sensing experiments.

Second, one has to select the hyperparameters of the kernel. In the case of square exponential kernel there are three hyperparameters - noise level σ_f, horizontal lengthscale l and vertical lengthscale σ_v. The first approach is to select the hyperparameters based on prior knowledge. The noise of the sensor can be estimated, by providing it with constant signal and by measuring the variation. The horizontal lengthscale could be modelled based on the knowledge about how much does the concentration/intensity of the measured process varies with respect to time and space. Finally, one can specify the vertical legthscale as the range of the sensor and update the mean kernel to be the most likely value. An alternative approach is to use a pilot experiment, where some data is collected. Then the probability of the model given the observed data is maximised [4]. This can be expressed in eq.2.12 [28]. The problem with this approach is that the hyperparameters could vary from experiment to experiment. Ideally, hyperparameters selection should be optimised as the measurements are taken. Traditionally, two approaches have been used - Monte Carlo and maximum likelihood estimation (MLE) [16].

$$\log p(y|X, \theta) = \frac{1}{2} y^T K_y^{-1} y - \frac{1}{2} \log |K_y| - \frac{n}{2} \log 2\pi \tag{2.12}$$

The log likelihood has to be maximised, or equivalently the loss function in eq.2.13 has to be minimised.

$$L(\theta) = \frac{1}{2} \log |K_y| + \frac{1}{2} y^T K_y^{-1} y \tag{2.13}$$

However, this could prove to be expensive due to the inversion of the n-dimensional K_y matrix, where n is the number of measurements. In order to address this issue, multiple optimisation techniques have been proposed. Some approaches include Scaled Conjugate Gradients [27], BFGS [16] as well as its more memory efficient version L-BFGS [18], Newton's method or often referred as the iteratively reweighed least squares (IRLS) algorithm in the context of Gaussian Processes [11]. The conjugate gradients method treats the problem of solving $K_y^{-1} y$ as a quadratic programming problem [16]. This is based on the realisation that a problem $A^{-1} b$ can be converted to $Ax = b$. Considering the function

$$\varphi(x) = \frac{1}{2} x^T A x - x^T b \tag{2.14}$$

the minimum value of the function is for $x = A^{-1} b$. Having redefined the problem, the method of steepest descent can be used to solve it [8]. Similarly, BFGS and Newton's

methods are also used for optimising the quadratic programming problem.

So far in this chapter I introduced how to construct a Gaussian Processes regression, how to choose a kernel and optimise the hyperparameters. Having a specified model, one can go and take readings, updating the beliefs along the way. The last remaining, and very important question, is how to select the next point to make a measurement. This is in fact the core question in active sensing. How to optimally place n static sensor in order to measure a specified field was discussed in [14]. This is different from our problem of active sensing, but it gives good intuition of how this can be approached in a dynamic setting. Probably the simplest approach is to solve the task as an art-gallery problem [10]. In this case, an UAV should uniformly take readings across the observed space. Alternative approach is to place sensor at places with highest entropy, thus minimizing the entropy at the unobserved places. This expressed in mathematical terms can be seen in eq.2.15.

$$\begin{aligned} \mathcal{A}^* = argmin_{\mathcal{A}\subset\mathcal{V}:|\mathcal{A}|=k} H(\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}|\mathcal{X}_{\mathcal{A}}) &= argmin_{\mathcal{A}\subset\mathcal{V}:|\mathcal{A}|=k} \mathcal{H}(\mathcal{X}_{\mathcal{V}}) - \mathcal{H}(\mathcal{X}_{\mathcal{A}}) \\ &= argmax_{\mathcal{A}\subset\mathcal{V}:|\mathcal{A}|=k} \mathcal{H}(\mathcal{X}_{\mathcal{A}}) \end{aligned} \tag{2.15}$$

Formulated as a static optimal placement of n sensors, this is a NP-hard problem. However, a greedy approach could be used where sensors are added one by one. This of course directly relates to the active sensing problem and could be reused. The main criticism of this approach is that it's indirect. The entropy only at the sensors' location is considered, rather than how they affect the quality of the prediction over the whole field. Instead one would ideally want to increase the mutual information between all the taken readings as in eq.2.16. Here again, this is hard to compute as it is a NP-complete problem, but greedy algorithms exist that directly map to active sensing.

$$\mathcal{A}^* = argmax_{\mathcal{A}\subset\mathcal{V}:|\mathcal{A}|=k} H(\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}) - H(\mathcal{X}_{\mathcal{V}\setminus\mathcal{A}}|\mathcal{X}_{\mathcal{A}}) \tag{2.16}$$

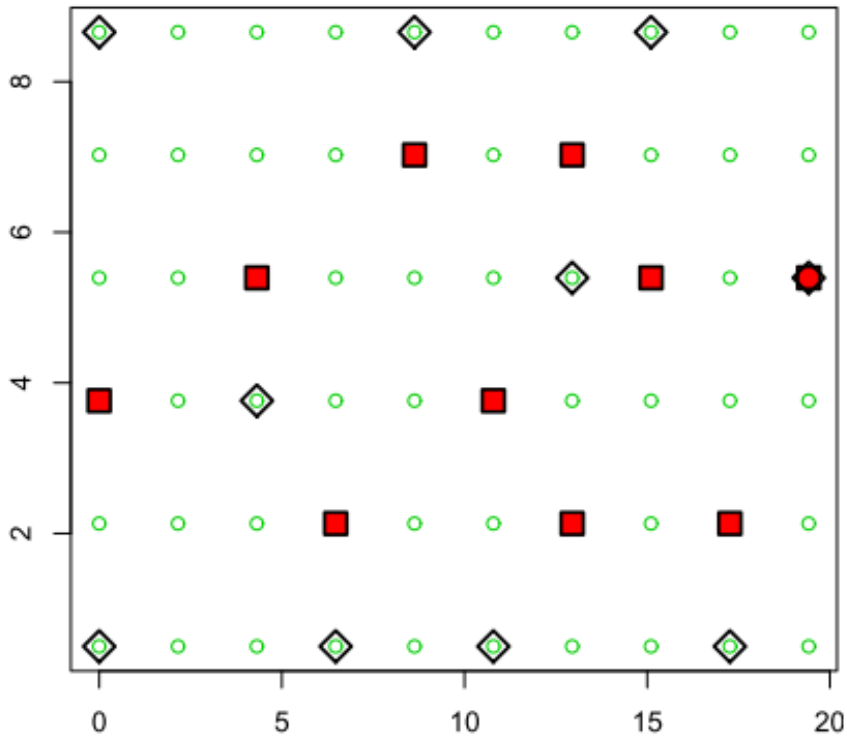


Figure 2.6: Sensor placements using the entropy criteria (diamonds) and mutual information (squares). The entropy criteria tends to prefer sensor placements along the edge of the observed phenomena, which does not necessary reduce the most the overall uncertainty. (figure taken from [14])

In certain cases, one might be interested in finding the global maxima/minima in addition to mapping the whole observed field. As such a Bayesian optimisation can be used. One can employ different strategies to achieve this, some of which include expected improvement (EI) [32], upper (lower) confidence bound (UCB) [32] and maximum probability of improvement (MPI) [15]. UCB (eq.2.17) is direct trade off between the exploitation and exploration principle. If $k = 0$, it is pure exploitation, always going to the highest value of the current estimate. Alternatively, when k is high, the function selects places with high uncertainty regardless of where the estimated maximum of the function is.

$$\alpha_{UCB}(x;\theta,\mathcal{D}) = \mu(x;\theta,\mathcal{D}) + k\sigma(x;\theta,\mathcal{D}) \tag{2.17}$$

EI and MPI both try to find a more optimal value, but with different criteria. EI (eq.2.18) maximises the expected improvement, which is the product of the improvement and the probability of that improvement.

$$\alpha_{EI}(x;\theta,\mathcal{D}) = \int_{y} \max(0, y_{best} - y)\, p(y|x;\theta,\mathcal{D})\, dy \tag{2.18}$$

On the other side, MPI (eq.2.19 and eq.2.20) finds a point with the highest probability of improvement, regardless of how big this improvement is.

$$\alpha_{MPI}(x;\theta,\mathcal{D}) = p(f(x) < y_{best}) = \varphi(\gamma(x)) \tag{2.19}$$

$$\gamma(x) = \sigma(x;\theta,\mathcal{D})^{-1}(\mu(x;\theta,\mathcal{D}) - y_{best}) \tag{2.20}$$

Chapter 3

Related work

Building upon Chapter 2, this chapter describes related work in active sensing using different robotics platforms. It summarizes different applications and optimisation techniques being used. The chapter describes different specific applications of drone sensing and some of the challenges and benefits in comparison to other platforms[1]. Finally, a short description of spectroscopy and previous related work in hazardous gas sensing is presented.

3.1 Active sensing of spatiotemporal phenomena

The work conducted in this report can be summarized as active sensing of spatiotemporal phenomena, so it is useful to provide the two definitions, namely def.3.1.1 and def.3.1.2.

Definition 3.1.1 *Spatiotemporal phenomena is an event relating to, or existing in both space and time.*

Definition 3.1.2 *Active sensing, or active information gathering, is collecting the most useful information about a problem and then using it to do inference with the goal of maximizing the accuracy of the inference while minimizing the quantity of information gathered.*

Using Gaussian Processes ($\mathcal{GP}$) regression for modelling spatiotemporal phenomena has been an active area of research in recent years. An example of monitoring a spatiotemporal phenomena and their dynamics is discussed in [31]. The phenomenon

[1] Parts of the analysis were summarized from my reports for ASR and RRP

discussed in the paper is water quality in rivers and lakes. The goal specified is to maximize the information collected, while taking into account the limitations of the sensor devices and robots being used. Hence, $\mathcal{GP}$ are a good choice since they can quantify the amount of information they have collected. Moreover, they can pick locations they want to go to next, based on the places with higher uncertainty.

In modeling spatiotemporal phenomena a common problem seems to be few incorrect extreme measurements leading to inaccurate model [19]. $\mathcal{GP}$ tend to reproduce a field around those few extreme measurements, while predictions being low in desirable locations. However, using log-measurements mitigates the issue as it removes extremity and skewness.

Another key idea in conducting inference with $\mathcal{GP}$ is lazy evaluation [14]. This is achieved by continuously picking a place with high uncertainty - conducting measurements - making inference and updating the model - picking a place with high uncertainty - etc. Usually correlation decreases exponentially as the distance between points increases. However, often variables apart from each other are dependent. By exploiting locality in kernel functions one can achieve significant speedup of the algorithm [14]. $\mathcal{GP}$ also prove to be effective in modeling spatiotemporal phenomena over long periods of time [13].

There have also been interests in using UAV mounted sensors [1] [12] [24]. Examples of this include coastal wetland mapping, flood and wildfire surveillance, tracking oil spills, urban studies, and Arctic ice investigations [12]. However, most of the work is focused around data collection with the processing down offline after the flight is finished. Currently, not a lot of work is done on the problem of active sensing, where the UAV is processing the data online, updating an underlying model and actively steering to achieve a certain goal.

There are different aspects of active sensing that one could investigate, but in this report the focus is on how can the most accurate predictions be achieved with the fewest measurements possible. This is dictated from two reasons - battery capacity of the drone and time needed to take a measurement. Moreover, it is also investigated how to find the source of contamination the fastest in addition to having the most accurate prediction of the observed field everywhere.

An area of great importance, when doing active sensing on a drone, is the speed of the inference, as there is limited flight time. There are multiple ways, one can optimise a $\mathcal{GP}$. For example, one way is to have a more efficient inference, like Conjugate Gradients [27], BFGS [16] and Newton's method [11]. Another way of improvement

is to have an efficient way of selecting new points for prediction. Again here different methods have been proposed like expected improvement (EI) [32], upper (lower) confidence bound (UCB)[32] and maximum probability of improvement (MPI) [15]. All of them essentially provide a trade-off between exploration and exploitation.

Doing regression with $\mathcal{GP}$ has a couple of useful properties. First of all, unlike neural nets for example, $\mathcal{GP}$ is a non-parametric model. Learning in parametric models, like backpropagation, consists of many iterations. In $\mathcal{GP}$ the learning of a model consists of only setting the hyperparameters and calculating a conditional probability distribution of two Gaussians. Moreover, $\mathcal{GP}$ provides a framework for quantifying uncertainty. This proves to be of great importance in different fields where only a limited number of noisy measurements can be taken.

In this report, squared exponential kernel is discussed. However, there are a wide variety of other kernels, which can generate and model different types of functions. In fact, one can also combine different functions to compose a kernel, the only requirement being that the kernel has to generate positive definite covariance matrix [28]. $\mathcal{GP}$ can be used for classification similarly to neural networks by applying sigmoid nonlinearity [20]. An interesting theoretical result is that $\mathcal{GP}$ can simulate shallow neural networks with one hidden layer with an infinite number of neurons [33]. This combined with the fact that any function can be represented using a shallow neural network, suggests that $\mathcal{GP}$ is indeed a powerful model [6]. They've been some recent advances to scale $\mathcal{GP}$ similar to neural networks by introducing hierarchical structure to improve learning [7]. Another work aims to combine $\mathcal{GP}$ with neural networks, aiming to get the expressibility and robustness from neural networks, while incorporating the uncertainty estimates from the $\mathcal{GP}$ [34]. However this is beyond the scope of this project.

3.2 Spectroscopy

The main application that motivated this project was detection of hazardous gases using spectroscopy. The purpose of a spectrometer is to measure the emission spectrum of a light source. The beam is focused through a slit, dispersed and a sensor on the other side detects the different wavelengths. This creates a function, where there is an intensity value for each of the different wavelengths. As gases absorb light in different parts of the spectrum, by using a broadband spectrometer, one can detect and measure the intensity of variety of gases. The infrared spectra of more than 60 explosive

compounds was analyzed in [25]. In this study, it was shown that explosives can be distinguished by their absorption in different wavelengths regions - alcohols showing a strong band near $2.9\mu m$ and amines near $3.0\mu m$. It can also be seen that it might be possible to detect common explosives like TNT, PETN, TATP and RDX from their absorption spectra in the $2.6\mu m$ to $3.6\mu m$ band [29]. CO_2 can be used as a proxy for running experiments due to safer testing as well as strong absorption at the $2.5mm$ band.

Research by collaborators from Heriot-Watt [2] has led to the development of mid-infrared Fourier-transform spectroscopy for standoff detection of liquids on surfaces [35] and the detection of aerolized airborne hazards [21]. However, in both of those cases, they relied on powerful and large femtosecond laser system, incompatible with a UAV system. In the system used in the project they replaced it with conventional thermal light source and investigated what level of performance can be maintained.

While single gas sensors are widely available, integrating a broadband spectrometer within a UAV have not been done yet. It is a challenging problem due to the many constraints imposed - weight, power, computational resources, etc. While this report would not focus on the actual development of the sensor, some of those constraints and the mode of operation of the sensor would be important for structuring and developing the active sensing framework. The main limitation when using such a sensor is the number of measurement one could make. The UAV is heavier due to all the equipment and extra power is being drawn from the different parts of the sensors and the on-board computer. Moreover, a few seconds of computational time are required in order to compute the fourier transform of the signal. This essentially limits the system to taking only a couple of dozen readings, before running out of battery.

[2] Pr.Derryck T.Reid, Marius Rutkauskas

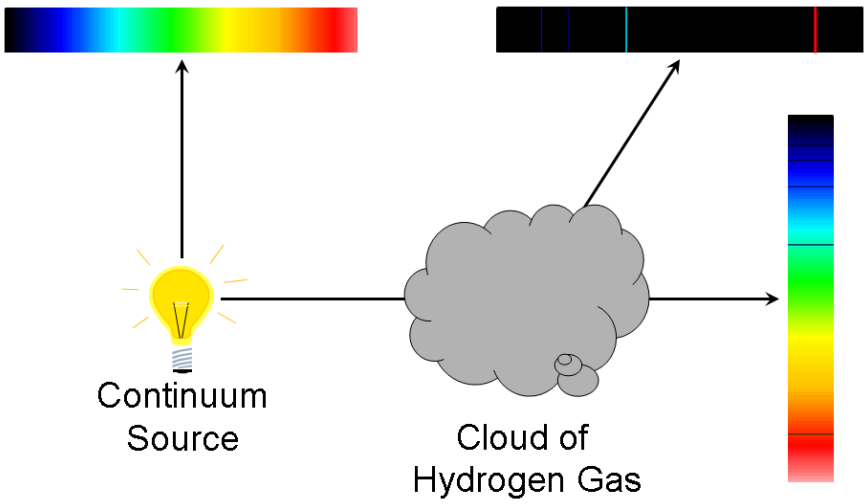


Figure 3.1: Schematic description of gas spectroscopy. A constant light source is pointed through a set of mirrors, decomposing it into a spectrum. A detector at the end measures the intensity of the different parts of the spectrum. As the light passes through different types of gases, the spectrum changes depending on the type and concentration of the gas. (Figure taken from [30])

Chapter 4

Design and Methods

This chapter discusses the choice of hardware and sensors used in addition to software framework developed. First of all, the specification of the drone and on-board computer are discussed in terms of requirements for developing an active sensing framework. Throughout the report two sensors are used, off-the-shelf light sensor and broadband spectrometer developed by collaborators at HWU. The differences between the two sensors are discussed and how they affect the modelling assumptions. Finally, the chapter describes the software stack developed in order to conduct experiments.

4.1 UAV platform

In order to perform the necessary experiments a robust UAV platform is needed, capable of carrying certain weight, outputting current for external hardware as well as being fully programmable from on-board computer. The platform of our choice was DJI's Matrice 100 quadcopter. Some of the specifications most relevant to the project are 1.2kg weight allowance, overall weight of 3.6kg and flight time of about 10 minutes. When using off-the-shelf sensor, the weight allowance is reasonable. However, in the case of the broadband spectrometer sensor, this turned to be one of the main limiting factors. As components of the sensor are decreased in size and weight, more noise is introduced, making the collection a sensible readings harder. Moreover, as the weight of the UAV is increased, the flight dynamics change, which could impede the sampling strategy. So far it was assumed that the cost of getting to a place is negligible. As the drone gets heavier, the time of flight shortens. Therefore it might be useful to take into account the cost flying to a point within the sampling strategy.

Figure 4.1: DJI Matrice M100. The Matrice M100 is one of the few drones well suited for the project. It has full ROS support, UART connection for control from an on-board computer, external power connection from the battery, ability to carry up to 1.2kg payload.

4.2 On-board computer

In the experiments presented two on-board computers were used, Raspberry Pi 3 and Nvidia TX1. The choice of on-board computer is mostly agnostic. There must be a connection from the on-board computer to the drone's controller, in this project a UART was used. The developed framework does not have any special computational requirements, neither there is need for strong Wifi signal as all the processing is done on the drone itself.

4.3 Sensors

The specific application the project explores is automatic detection of hazardous gases, using the chemical sensor developed at HWU. However, due to number of difficulties associated with conducting experiments with such a sensor, in most of the work described here, a cheap off-the-shelf light sensor was used. There are some similarities between the two sensors, but also some important differences. In terms of similarities both of the sensors are capable of measuring intensities - of either gases or light - which work equally well within the framework. However, there are some subtle differences that make the chemical sensor more challenging to use in comparison to the

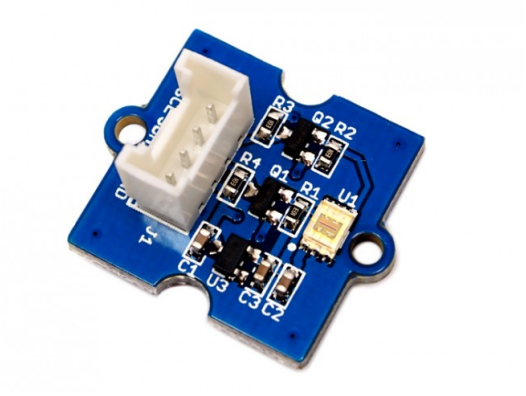
light one. When acquiring data with the light sensor, one needs to take only one reading - of the current intensity of the light. The processes of acquiring useful reading with the chemical sensor is more involved. First of all, at least two hundred samples have to be acquired. Next a Fast Fourier Transform (FFT) is computed. The result is a spectrum of the light within certain wavelengths. From there on, one has to compute how the changes in the specific wavelengths are affected by the concentration of the gas measured. Due to the logistics difficulties with using explosive gases, in the set of experiments CO_2 was used. First of all, the gas is non-lethal [1] making testing easier. Secondly, the concentration of CO_2 can be clearly observed by the absorption at $25mm$ band. Apart from these issues, the other challenge is dynamics of gases. When flying outside air have certain dynamics based on the wind speed and direction. Moreover ,the propellers introduce local air dynamics that could also affect the measurements [23].

4.3.1 Light sensor

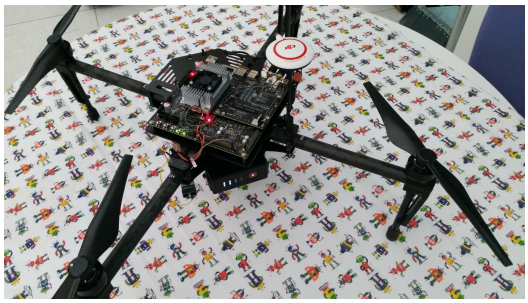
For the light experiments an off-the-shelf low cost light sensor was used. The sensor connects to oscilloscope for acquiring data, which then connects to the on-board computer. In addition to removing the complexity of modelling the dynamics of gases, the sensor also conveniently outputs a single number, representing the intensity of the light. There are two disadvantages of this sensor. First of all, when used during day time, the light from the sun is too strong, and the sensor always gives maximum value. Secondly, the accuracy of the sensor is not very high, because it is able to detect only a few dozen levels of intensity of light. Moreover, the sensor has to be in close proximity to the flash light in order to get a strong reading.

[1] Unless in high concentration. When conducting experiments, at least two CO_2 meters were carried, measuring potentially dangerous level of concentration

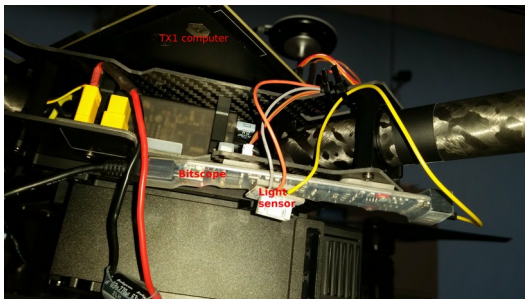
(a)



(b)



(c)

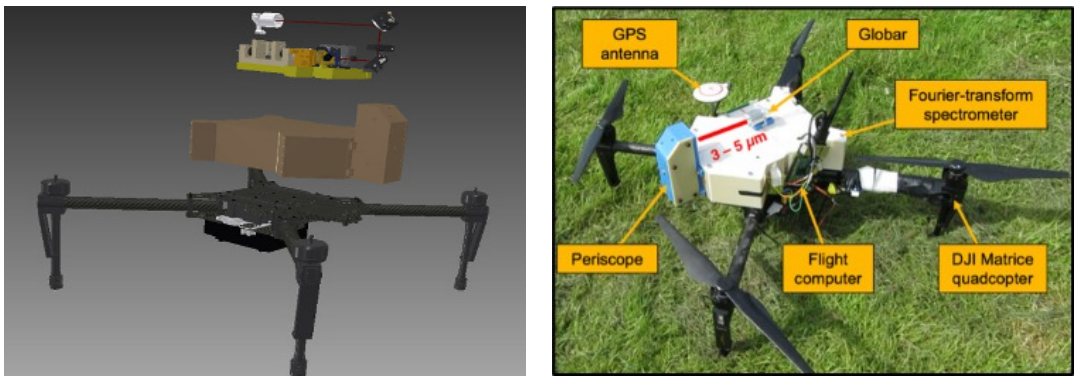


(d)

Figure 4.2: Hardware setup for the light experiments. (a) Bitscope oscilloscope used for acquiring the data from the sensor (b) Light sensor (c) Full setup of the on-board computer connected and powered from the drone (d) Sensor and Bitscope mount point

4.3.2 Broadband spectrometer

In the experiments, the possibility of using a broadband spectrometer for detection of different gases was also explored. The design of the sensor can be seen in fig.4.3.



(a) (b)

Figure 4.3: Hardware setup for chemical experiments. (a) Design of the broadband spectrometer on the drone (b) The sensor mounted on the actual drone (figure taken from collaborators)

This introduces a whole set of other problems as already discussed - modelling dynamics of the wind, modelling dynamics of the air flow from the propellers, extra processing required for acquiring the signal, classifying the type of gas and it's concentration based on the signal, etc. The dynamics of the air flow around a hovering UAV is exemplified in fig.4.4. Based on all those difficulties, it is important to start with a simpler problem of active sensing like light. But, at the same time it is necessary to keep in mind all those issues, when developing different algorithms.

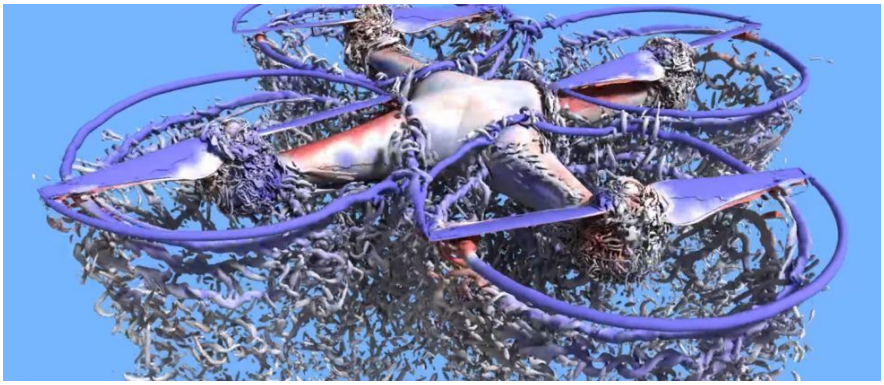


Figure 4.4: Aerodynamics of a small drone. NASA presented work when they explore the aerodynamics of a similar DJI Phantom 3 drone. (figure taken from [23])

4.4 Software stack

The first step in order to conduct the experiments was developing the necessary software stack. For the Gaussian Processes Regression, the GPy library [9] was used. The Matrice 100 has full ROS [26] support including commands like take off, go to waypoint, land, etc. In order to incorporate Bayesian optimisation, the GPyOpt library [2] was also integrated.

For simulation purposes DJI's simulator was used as seen in fig.4.5. This allows to verify that the drone is connected with the on-board computer, the commands are received and executed, different libraries are correctly integrated and are running when needed. The main limiting factor when running things in simulation is that it is hard to get meaningful data from the light sensor connected to the Bitscope. For that reason, it is also important to conduct experiments with Gaussian Processes with synthetic data in order to evaluate different options. This ensured that the final system would have as little errors as possible. However, despite all the simulations there were multiple challenges when conducting real flights - noise in the sensor, dropping wifi connection, flight dynamics, etc.

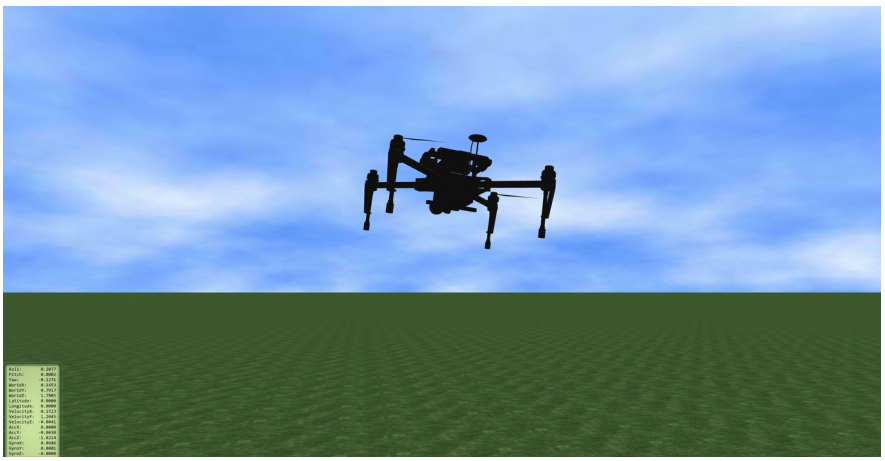


Figure 4.5: Testing in simulation. The simulator was enough for testing the API of the drone and shows a proof of concept that the developed software works.

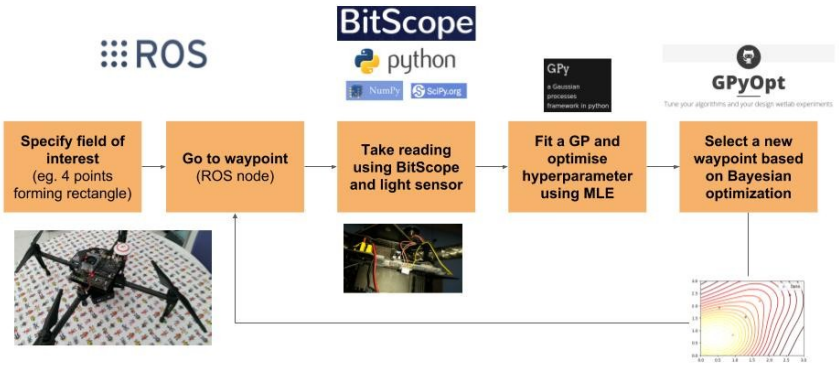


Figure 4.6: Software stack of the project. During the project ROS, GPy and GPyOpy were used for the development of the on-board software. After specifying the field of interest, a close loop is executed of taking a measurement, fitting a Gaussian Processes model, selecting a new point to explore, flying to it, etc.

4.5 Conducting test flights

Doing experiments with the drone was challenging for number of reasons. First of all, one has to find a suitable place for flying - away from people, with shortly cut grass or other flat surface, clear view of the sky for GPS acquisition, no light apart from the one coming from the flash lights used for testing, weak wind and no rain.

Heriot-Watt was chosen as a main place for flying most convenient as it has a few grass fields nearby and it is close to the lab where the drone is stored. An issue was the length of the grass, but a wooden plate to taking off and landing as seen in fig.4.7 was used.

Figure 4.7: Conducting experiments during the night. When conducting light experiments, we waited until sunset and used number of industrial flash lights to generate signal for the sensor.

When flying a drone a number of safety rules should be fulfilled. To ensure safe work, the person navigating the drone should be able to manually take off and land the drone, return the drone from short and long distance, even when he has lost track of the current orientation of the drone. The Matrice 100 is fairly easy to fly due to the controller provided with the drone. It has three different modes - F-mode, where the drone is controlled from the on-board computer (TX1 or raspberry pi in this case), A-mode, where the drone maintains a constant height, and P-mode, where the drone also accounts for the wind and maintains a constant position. This makes flying easy as when the drone is in P-mode, the drone hovers without any input from the controller. Moreover, when running experiments one could switch from F-mode to P-mode, which acts as a kill switch, immediately stopping the drone in place. As part of the safety procedure we always fly the drone manually first, then fly the drone to the edges of specified field to check that we are in safe distance from other objects and people, before proceeding to run the full experiments.

Finally, once out in the wild, one also has to connect to the on-board computer to do changes on the fly as well as run commands. For that a hotspot from an Android phone was used, to which the on-board computer automatically connects. As the drone flies around, this sometimes breaks the ssh connection, killing the currently running

program. A terminal multiplexer (tmux) was used in order to address this issue. It was also useful as one can reattach to the running process, even if the connection is lost mid-air.

Chapter 5

Experiments

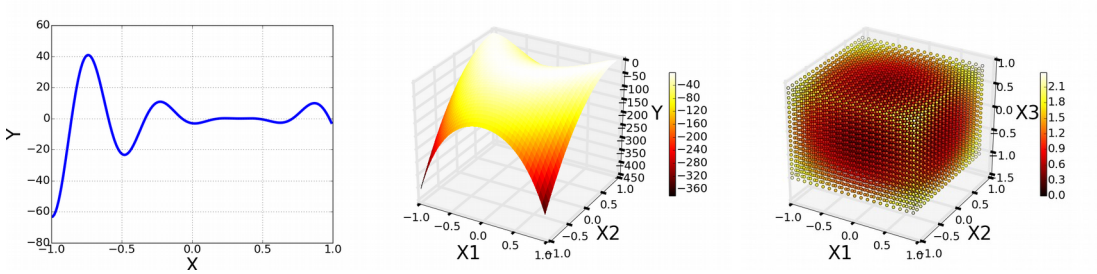
This chapter provides a thorough description of the experiments undertaken and their purpose. The goal for this project is to create an active sensing drone framework to be used for modelling various types of spatiotemporal phenomena. In order to develop and test the system in depth both simulation and real world experiments were used. The results from this section include benchmarks on three synthetic functions, demonstration of active source seeking and evaluation of different approaches using in-flight collected data.

5.1 Experiments with synthetic data

For the experiments with synthetic data three different functions were used to evaluate the performance of different optimisation algorithms and acquisition functions. The same random seed was used in order to have a fair comparison. As discussed in previous chapters, the goal is to model the observed field as well as find the global minima. Three evaluation metrics were used - the minimum of the function found, the average absolute-difference error between the posterior mean and the true values of the function and the time needed for computation. This gives a good estimate of how fast one can find the global minimum, how good is the overall prediction of the function and what is the performance. Three different aspects of the model are benchmarked - the number of samples used to estimate the function, the optimisation algorithm used for computing the posterior and the type of acquisition function. The functions tested are smooth, with multiple local minima, in order to make the evaluation as similar as possible to the real experiments. The example function could be seen in fig.5.1 with their benchmark results in table5.1,5.2,5.3.

		BFGS			SCG			TNC		
		min	err	time	min	err	time	min	err	time
LCB	1	-47.67	10.26	1.38	-15.80	10.26	1.04	-49.07	10.26	0.76
	3	-47.67	10.27	1.03	-48.64	10.30	1.56	-49.07	10.28	1.06
	5	-49.21	10.57	1.33	-49.64	10.43	1.77	-49.07	10.57	1.41
	10	-50.45	10.54	1.92	-50.44	10.45	2.81	-50.38	10.54	3.13
	20	-50.46	10.67	4.60	-50.46	10.69	6.36	-50.46	10.67	4.54
EI	1	-48.33	10.26	1.43	-27.81	10.44	1.05	-47.41	10.26	0.94
	3	-48.33	10.30	1.70	-30.96	10.57	1.63	-48.12	10.29	1.28
	5	-50.43	10.34	2.04	-50.32	10.64	2.29	-50.45	10.33	1.63
	10	-50.43	10.30	5.44	-50.32	10.62	13.27	-50.45	10.30	6.01
	20	-50.45	10.38	12.81	-50.46	10.64	22.35	-50.45	10.38	19.33
MPI	1	-21.18	10.26	0.73	-27.31	10.26	0.83	-20.88	10.26	0.72
	3	-23.15	10.44	0.98	-29.52	10.52	1.59	-22.82	10.43	1.44
	5	-24.57	11.69	1.37	-31.82	11.53	7.28	-24.57	11.85	1.54
	10	-43.35	11.70	2.04	-43.35	11.56	26.56	-43.35	11.70	2.60
	20	-50.45	10.91	10.30	-50.46	10.95	89.07	-50.45	10.90	10.32

Table 5.1: Benchmark values for the $1D$ regression problem $(6x-2)^2 sin(12x-4)$



(a) (b) (c)

Figure 5.1: Example test functions for Gaussian Processes. A number of increasingly more complex functions were used in order to evaluate how well active sensing works. (a) $1D$ regression example - $(6x-2)^2 sin(12x-4)$ (b) $2D$ regression example - $100(x_1 - x_0^2)^2 + (x_0 - 1)^2$ (c) $3D$ regression example - $\sum_{i=0}^{3} x_i sin(x_i) + 0.1x_i$

Observing the benchmarking data several conclusions can be drawn. In terms of optimisation methods BFGS is the fastest, outperforming both SCG and TNC. This comes to no surprise, as the efficiency of BFGS has been shown before [17].

		BFGS			SCG			TNC		
		min	err	time	min	err	time	min	err	time
LCB	1	-214.14	58.61	3.92	-313.22	58.62	5.93	-159.41	58.62	2.52
	3	-404.00	59.46	4.01	-400.00	58.94	8.08	-273.69	58.83	2.84
	5	-404.00	59.17	4.73	-400.00	58.87	4.01	-404.00	59.00	2.98
	10	-404.00	59.98	3.40	-404.00	60.20	5.84	-404.00	59.98	3.08
	20	-404.00	60.32	3.62	-404.00	60.36	4.93	-404.00	60.14	4.27
EI	1	-110.31	58.61	1.83	-229.20	58.62	3.52	-110.31	58.62	1.71
	3	-282.71	59.33	2.12	-368.91	59.66	2.75	-282.09	59.32	4.16
	5	-404.00	59.16	9.15	-404.00	59.77	4.10	-404.00	59.24	2.63
	10	-404.00	60.03	9.03	-404.00	59.70	13.04	-404.00	59.96	12.14
	20	-404.00	59.41	18.38	-404.00	59.42	48.22	-404.00	59.40	15.81
MPI	1	-112.73	58.71	1.51	-121.99	58.62	3.29	-110.31	58.66	1.77
	3	-118.37	59.29	1.76	-129.45	59.54	2.82	-115.82	59.39	1.92
	5	-139.78	59.62	2.01	-150.28	60.45	7.30	-128.85	60.07	2.42
	10	-200.50	61.42	4.85	-358.22	60.32	20.65	-259.00	60.72	5.44
	20	-400.00	60.81	10.46	-400.00	59.77	87.07	-400.00	60.50	12.40

Table 5.2: Benchmark values for the $2D$ regression problem $100(x_1 - x_0^2)^2 + (x_0 - 1)^2$

		BFGS			SCG			TNC		
		min	err	time	min	err	time	min	err	time
LCB	1	0.64	0.92	0.38	0.64	0.89	1.80	0.64	0.92	0.57
	3	0.22	0.97	1.00	0.41	0.96	2.23	0.21	0.97	1.05
	5	0.22	1.00	2.15	0.41	0.91	2.77	0.21	1.00	1.27
	10	-0.00	1.04	2.24	0.03	1.07	3.98	-0.00	0.88	2.51
	20	-0.00	0.54	5.01	-0.01	0.56	78.19	-0.01	0.36	6.03
EI	1	0.63	0.82	0.78	0.64	0.92	1.50	0.63	0.82	1.66
	3	0.41	0.99	1.34	0.64	0.78	1.44	0.41	0.99	1.28
	5	0.09	1.04	1.90	0.48	0.88	2.06	0.09	1.04	2.01
	10	0.00	0.89	8.28	0.02	0.81	16.08	0.00	0.89	8.03
	20	-0.01	0.65	41.19	-0.01	0.62	82.12	-0.01	0.65	33.81
MPI	1	0.64	0.82	0.65	0.64	0.93	0.87	0.64	0.82	0.60
	3	0.58	0.90	0.93	0.59	0.92	1.58	0.59	0.90	2.69
	5	0.53	0.89	1.27	0.55	0.92	2.34	0.54	0.89	1.37
	10	0.30	0.85	7.77	0.29	0.85	22.39	0.31	0.86	8.27
	20	0.08	0.68	25.55	0.09	0.59	124.30	0.09	0.69	30.21

Table 5.3: Benchmark values for the $3D$ regression problem $\sum_{i=0}^{3} x_i sin(x_i) + 0.1 x_i$

Expectedly, as more samples are used, a better minima of the function is found. An interesting result from the data is that the accuracy of our posterior mean in comparison to the true values of the function does not improve a lot as more samples are used in the $1D$ and $2D$ example, but it does in the $3D$. There are two reasons for that. First, the hyperparameters of the kernel are estimated as samples from the functions are taken. Thus, depending on the complexity of the function and the location of the samples taken, one might need more than 20 to correctly estimate the hyperparameters of the kernel. Second, even though it is a higher dimensional regression problem, within the evaluated interval, the $3D$ regression example has simpler structure than the other two problems. It has only one local minima as seen in fig.5.2.

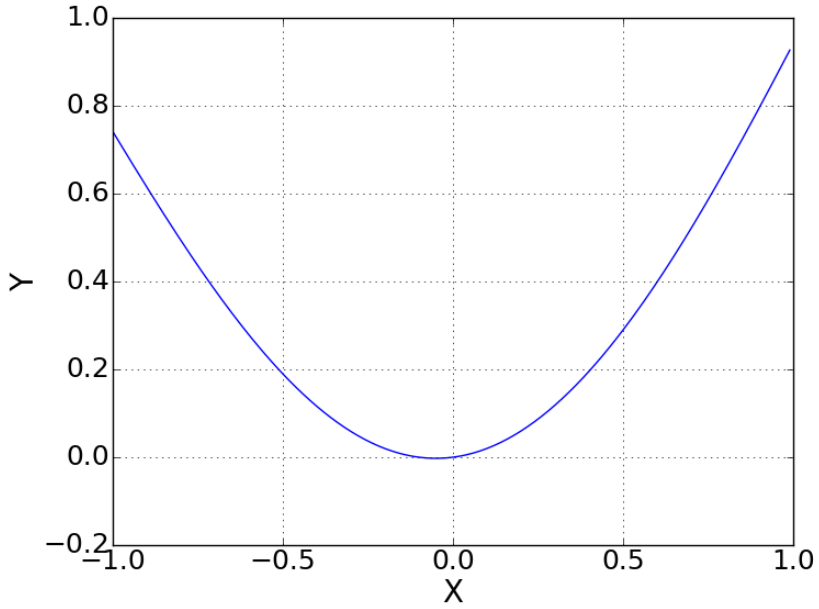
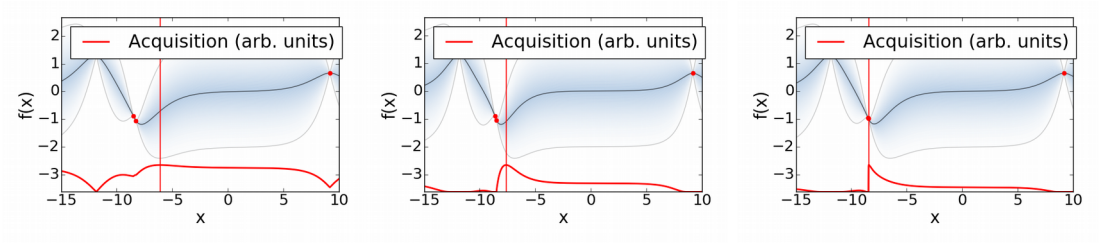


Figure 5.2: Plot of $x_i sin(x) + 0.1x$ for the interval $[-1, 1]$. Even though it is a 3D regression problem like $\sum_{i=0}^{3} x_i sin(x_i) + 0.1x_i$, its structure is not as complicated as other $1D$ and $2D$ problems. Since different x_i are not multiplied together, a similarly looking smooth function with one local minima is presented, regardless of how big i is.

Finally, the different acquisition functions seem to be giving similar results. Despite their different criteria, they often suggest similar point, which can be seen from fig.5.3.

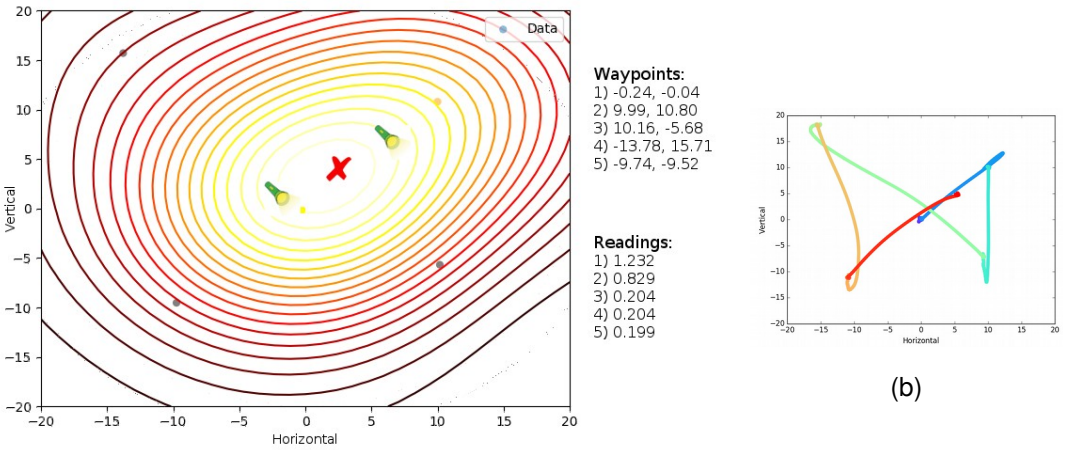


(a) (b) (c)

Figure 5.3: Acquisition functions. In this plots the values of the acquisition function are shown in red at the bottom. The function used is $sin(x)$. Three random sample points are taken, before the acquisition function is computed. The next selected point is marked by a vertical red line. In this example, despite the different criteria used, they all select a similar next point for prediction. (a) LCB (b) EI (c) MPI

5.2 Initial experiments with light sensor

The first experiment consisted of flying to 5 waypoints, fitting a GP model and then flying to the point of the highest mean. The hyperparameters of the kernel were manually set $(l = 9.5, \sigma_f = 0, \sigma_v = 1)$, and for the acquisition model the point with the highest posterior mean was used. The model outputs a sensible choice for next point of exploration, right in between the two flash lights. From this initial experiments a few observations could be made. First of all, the light sensor in this case performs well, mostly due to the fact that there was no light coming from anywhere else, but the flash lights. Also the DJI's API for flying to a specific point seemed to be inaccurate. This turned out to be due to the way it is implemented - as a one way request, without any callback for conformation, nor guarantees for accuracy. Based on that an improved version of the API was implemented, by dynamically monitoring current velocity and position until the drone have arrived at the desired waypoint, sending multiple request for the wanted goal until it is within a specified distance from it.



(a) (b)

Figure 5.4: Initial light experiment. In this experiment 2 flash lights were placed as seen in (a). The drone flies to five locations, gets readings, fits a Gaussian Process model and selects the point of highest posterior mean (X cross). The recorded trajectory can be seen in (b) and the video of the flight can be found at http://youtu.be/HVn5LSjqtOY

5.3 Full active sensing with Bayesian optimisation

In the next set of experiments a full Bayesian optimisation framework is integrated in addition to the Gaussian Process modelling as in fig.4.6. The drone flies at a constant height (6 meters) and tries to map a $2D$ field of 10 by 10 meters and locates a source of light. In this experiment, the BFGS was used as an optimisation algorithm for it's good performance and EI to focus on exploitation and source seeking, rather than exploration. The result can be seen in fig.5.5. The framework correctly maps locations of high and low intensity of light. This revealed another limitations of our hardware - the accuracy of our light sensor. In this specific experiment the light sensor returned only two values - 0.55 and 0.67 - depending on where it was close or not to a light sensor.

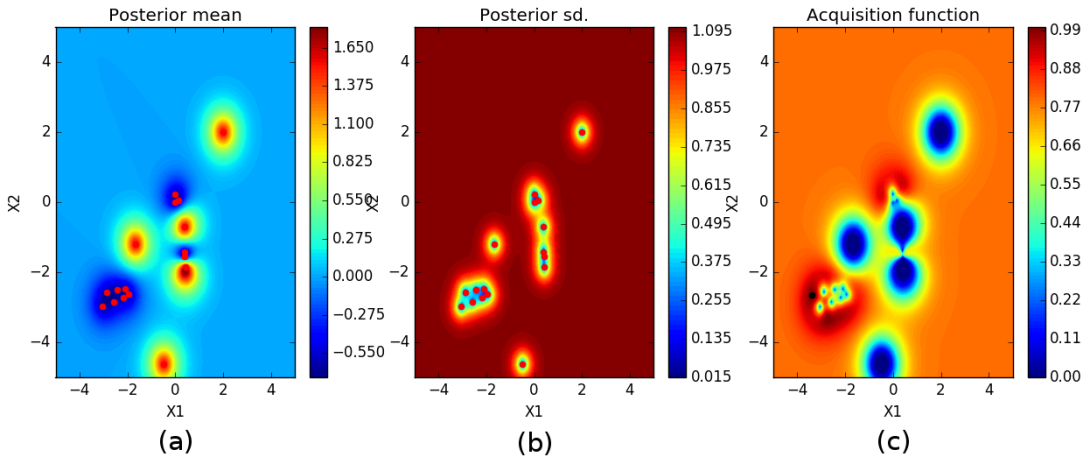


Figure 5.5: Experiment with Bayesian optimization. (a) Posterior mean over the observed region - the higher the value, the lower the intensity of light. The red dots show from where the drone took readings. The flash light is location in the bottom left, close to where the cluster of readings is. (b) Posterior standard deviation over the observed space. Close to where readings were taken, there is low standard deviation, increasing for points further away. (c) Values of the acquisition function over the observed space. The big blue clusters represent places, where there was low intensity of the light. It is interesting to also note the area in the bottom left - the drone shouldn't fly to the same locations it has been already, but also it is good to fly nearby as it got high intensity of light there, and it has strategy to exploit.

Next, a manual flight was conducted, collecting the position of the drone and readings of the sensor for every position, which can be seen in fig.5.6. By flying manually, the drone could safely go closer to the light source, making the data more diverse. However, the sensor still has only about 10 different values for the intensity of the light. So far in this section the assumption that the intensity of light is a smooth function of space was used, thus using squared exponential kernel. Having recorded data from the UAV this can be checked empirically if it is true. As seen in fig.5.7 as the number of sequential recorded light readings increases, the average variance increases, which confirms our assumption. The implemented framework is capable of doing the same modelling process in a $3D$ space as well. Moreover, as seen in sec.5.1 the inference process does not take significantly more time, and depending on the problem, it might not even take more samples than a $2D$ regression problem. However, two limiting factors prevented experiments of full 3D light source seeking. The light sensor had limited accuracy, especially at the height at which autonomous flight were conducted.

On the other side, it is not safe to fly the drone below certain height, due to lack of any obstacle avoidance and sometimes noisy positioning. Instead, data is collected by flying the drone manually, and predictions are evaluated by constraining the model to take samples and make predictions only within the recorded trajectory.

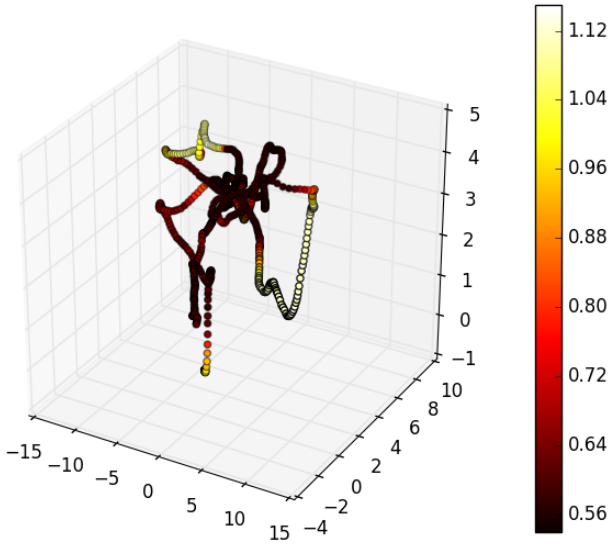


Figure 5.6: Trajectory of light readings during the night.

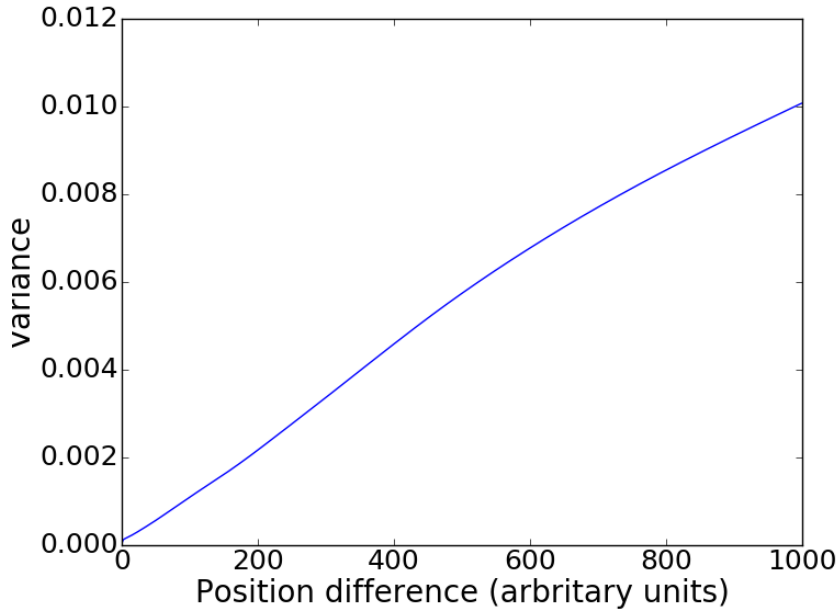


Figure 5.7: Variance in the intensity of light between far away positions.

5.4 Experiments with light sensor data

Evaluating the performance of our model of the field while flying with the drone is hard, mostly due to the limited battery life. In a typical flight one can only go to a couple of dozen waypoints. Moreover, in order to get diverse and meaningful data, the drone has to fly on a very low height autonomously, which is not safe at this stage of the project. Thus, in order to evaluate the performance of the model, data is collected and algorithms evaluated offline. In order to achieve this, the domain of the Gaussian Processes is restricted to be the recorded trajectory of the flight. The example data collected can be seen in fig.5.8. The benchmarks from the data can be seen in table5.4. The minimum was located by using any optimization algorithm or acquisition function, with results being good across the board. The overall error seems to not be decreasing, which could be accounted to the inaccurate sensor and the difficulty of finding good hyperparameters for the model.

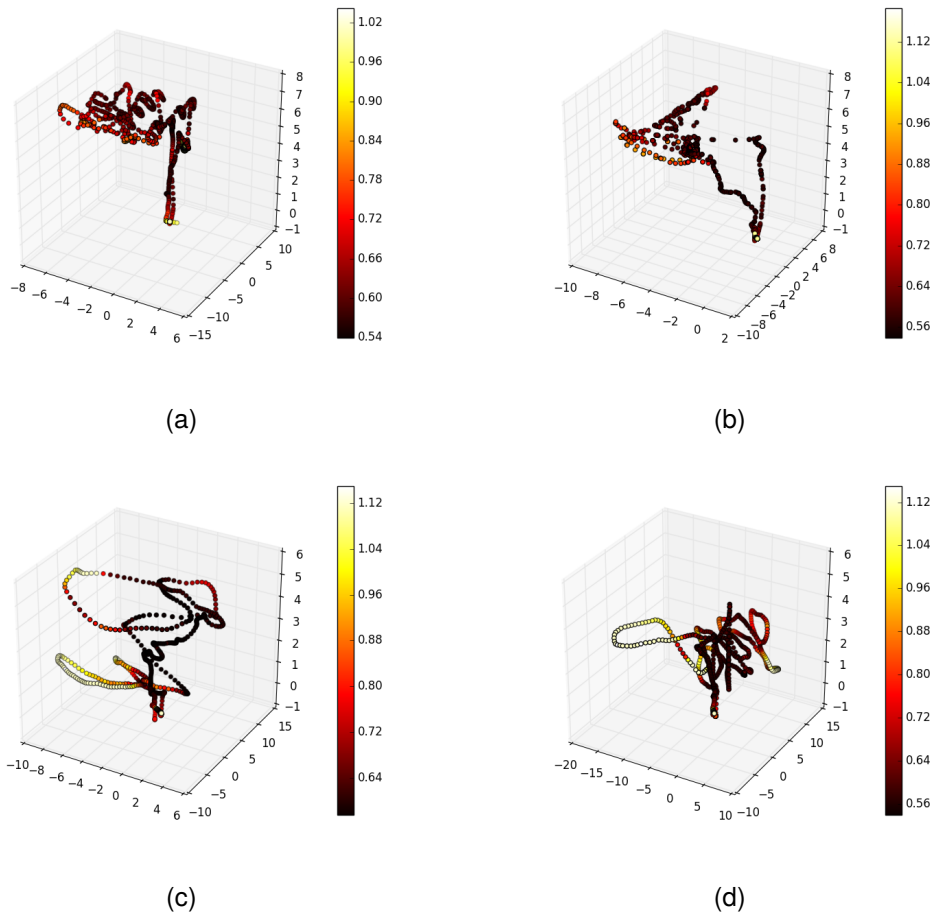


Figure 5.8: Recorded in-flight data of light intensities.

		BFGS			SCG			TNC		
		min	err	time	min	err	time	min	err	time
LCB	1	-0.99	0.70	5.87	-0.99	0.63	6.03	-1.00	0.71	5.85
	3	-0.99	0.67	5.95	-1.00	0.65	6.41	-1.00	0.68	6.15
	5	-0.99	0.67	5.94	-1.00	0.67	6.09	-1.00	0.65	6.32
	10	-1.03	0.78	6.38	-1.10	0.87	7.02	-1.03	0.77	6.74
	20	-1.09	0.81	6.59	-1.10	0.87	7.82	-1.09	0.79	6.93
EI	1	-0.99	0.70	5.55	-0.99	0.63	6.49	-1.00	0.71	7.30
	3	-0.99	0.67	8.53	-1.00	0.65	5.90	-1.00	0.68	5.64
	5	-0.99	0.67	5.65	-1.00	0.67	6.08	-1.00	0.65	5.79
	10	-1.03	0.78	6.17	-1.10	0.87	7.83	-1.03	0.77	8.68
	20	-1.09	0.81	7.95	-1.10	0.87	7.77	-1.09	0.79	7.51
MPI	1	-0.99	0.70	6.77	-0.99	0.63	8.36	-1.00	0.71	6.48
	3	-0.99	0.67	5.62	-1.00	0.65	6.47	-1.00	0.68	7.02
	5	-0.99	0.67	9.31	-1.00	0.67	6.69	-1.00	0.65	5.80
	10	-1.03	0.78	6.05	-1.10	0.87	6.73	-1.03	0.77	6.13
	20	-1.09	0.81	6.50	-1.10	0.87	7.77	-1.09	0.79	7.00

Table 5.4: Benchmark values for the in-flight recorded data (average across the results for the 4 flights)

5.5 Experiments with chemical sensor

Some preliminary experiments with the broadband spectrometer were conducted. However, there were multiple challenges during the integration of the sensor within the drone, including power, computation and most critically noise. The sensor was able to detect multiple different gases and liquids. However, when in flight the motors introduce noise within the detector. Filtering the noise out was tried, but it was discovered that the noise has similar frequency to the rotation of the motors, thus making filtering hard. Nevertheless, an example flight with the full stack of hardware required for the broadband spectrometer was done and can be seen in fig.5.9. However, more work needs to be done, until this sensor can be incorporated within an the active sensing framework.

Figure 5.9: Sensing of CO_2. A successful hardware and software integration with the broadband spectrometer sensor was demonstrated. However, due to strong noise caused from the vibrations of the motors, it was not possible to get good signal and integrate the sensor within the described framework.

5.6 Evaluation

The presented results achieve a couple of milestones towards the development of full autonomous active sensing framework on a drone.

First, the performance of Gaussian Processes on three different regression problems was evaluated, by comparing different optimisation algorithms for inference, acquisition functions and number of samples. The model was able to locate the global minima only with a few measurements, however to achieve accurate mapping of the field, more samples were needed. The BFGS optimisation algorithm was shown to be the fastest and more accurate across the board. Interestingly, it was observed that often the overall accuracy of the prediction does not improve as more samples were taken. This can be attributed to the fact that estimation of the hyperparameters was performed as readings were taken. As some of the functions had complex shape, it might take more than 20 samples to correctly estimate the hyperparameters and decrease the error. It was also demonstrated that a higher dimensional problem is not necessarily harder to optimise, when it has a simple structure. Finally, it was shown that different acquisition functions often give similar results and example was given why this is the case.

Next, the framework to work within a drone for active seeking of source light at a constant height. Two examples were shown, one with a Gaussian Process with fixed hyperparameters, and one with the full stack of Bayesian optimisation and hyperparameters estimation. Both of those examples showed promising results, but also limiting factors for conducting full $3D$ light source seeking experiment. More specifically, the flight controller of the drone needs further improvements, integrating a system of obstacle detection as well as more precise sensor.

Example data was also collected, by flying the drone manually. This was then evaluated similarly to the synthetic data, by restricting the model to sample the space only within the recorded trajectory. Similar performance of different choices for optimisation algorithms, acquisition functions and number of samples were shown. Throughout the experiments, it became even more evident, that a more accurate sensor is needed.

Finally, full hardware and software integration of a broadband spectrometer sensor for detecting and categorizing multiple gases was demonstrated. Due to multiple problems, most notably noise in the sensor, integration with active sensing capability was not possible.

Chapter 6

Conclusion

6.1 Summary

The report aims to present an unified overview of using drone as an active sensing framework for monitoring different spatiotemporal phenomena. The goals considered in the report were mapping a field of interest and finding global extrema points. The example problem considered was detecting hazardeous gases using broadband spectrometer, but a general framework was presented that can work regardless of the type of sensor used. Several important results were presented. It was shown that Gaussian Processes can model variety of different functions and can find global minima. Next, an integration with a drone was shown to work for mapping a light field and finding flash lights on the ground. A full hardware and software integration with a novel broadband spectrometer sensor was demonstrated. However, noise in the sensor introduced from the motors during flight, prevented integration of the sensor with the active sensing framework. Finally, benchmarks of synthetic and real in-flight collected data was conducted, comparing different aspects of the presented model like optimisation method, acquisition functions and number of samples used.

6.2 Future Work

Through work on the project several key areas of improvement can be noted.

Autonomous flight

During the project it was found that the flight controllers commands lack acknowledgement or conformation for the arrival of the drone at the specified waypoint. An improved version of the API was implemented, however the positioning of the drone,

especially while flying between waypoints, was still noisy. This resulted in the drone flying below its specified height, making it unsafe to fly at low heights. The company manufacturing the drone recently released an updated version of their SDK, so it would be interesting if there are any improvements in that regard. Also as another safety measure it would be good to integrate an obstacle avoidance system as the one in fig.6.1.

Figure 6.1: DJI guidance obstacle avoidance system.

Sensors

A major stumbling block within the project were the sensors used. The light sensor was simple and convenient, but with limited accuracy. The broadband spectrometer sensor yields interesting future applications, but was proven to be difficult to implement within a drone platform. The light sensor nevertheless provided a convenient way of running experiments, by eliminating some of the challenges of modelling dynamics of wind and air flow around propellers. Thus it would be nice to run the same set of experiments with higher quality light sensor. Secondly, an off-the-shelf CO_2 sensor could be used. This will still ease the problem of experimenting, as we won't have to deal with the complex acquiring of the signal from the broadband spectrometer, but being able to test how the introduced dynamics changed the introduced framework.

Flight locations

By transitioning from light to gas sensor, it would be nice to test first in a location isolated from any wind, and focus only on the dynamics introduced by the drone itself. Thus flying in a close space would be beneficial. However, then the problem of acquiring GPS arises. A good location would be big closed court, but with glass roof. Conveniently, there is a basketball court at HWU which could be used for that

purpose. Once the local dynamics of the propellers are modelled, the problem again can be extended by flying outside and taking wind into account.

Modelling

In this report we focused extensively on the traditional method used throughout for modelling spatiotemporal phenomena - Gaussian Processes. Although a powerful model, sometimes they may lack the expressibility of a model like neural network. It would be an interesting line of research to see if the two models can be combined, especially in the context of modelling local and global wind speed. We see a rise of the GPU on mobile board like the TX1 and TX2, so it would be useful to use a GPU implementation of a GP regression. By having more computational power possible bigger models can be used, which is relevant when we model processes dependant on both time and space.

Flight dynamics

In this work we discretize the problem and treat the UAV as a tool to sample points in 3D space. However, it would be interesting to incorporate velocity in our acquisition function, taking into account our position, speed and direction for a more optimal sampling strategy.

Applications

The system as it is presents a foundation for future development of already useful military applications - prevention and rapid evaluation of critical situations involving explosive gases. The regression model discussed is general and works regardless of the sensor used. As we already showed in the report, substituting the light and chemical sensor and vice versa imposes almost no additional work [1]. It would be an interesting line of research to find and test different types of sensors within this framework. An interesting example would be measuring wind speed. This has different applications in energy and environmental sciences. Apart from building more accurate local weather models, optimal places for placing wind turbines could be explored. Moreover, measuring wind speed does not actually require any additional sensor, as the error gradient from the flight controller could be used to directly estimate the speed and the direction of the wind.

[1] Here we don't discuss the work required for developing practically useful sensor. There are also different power and weight constraints depending on the model of the UAV used

Bibliography

[1] M. Alvarado, F. Gonzalez, A. Fletcher, and A. Doshi. Towards the development of a low cost airborne sensing system to monitor dust particles after blasting at open-pit mine sites. *Sensors*, 15(8):19667–19687, 2015.

[2] T. G. authors. Gpyopt: A bayesian optimization framework in python. `http://github.com/SheffieldML/GPyOpt`, 2016.

[3] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[4] M. Blum and M. A. Riedmiller. Optimization of gaussian process hyperparameters using rprop. In *ESANN*, pages 339–344, 2013.

[5] G. E. Box, M. E. Muller, et al. A note on the generation of random normal deviates. *The annals of mathematical statistics*, 29(2):610–611, 1958.

[6] T. Chen and H. Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

[7] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. Practical learning of deep gaussian processes via random fourier features. *arXiv preprint arXiv:1610.04386*, 2016.

[8] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[9] GPy. GPy: A gaussian process framework in python. `http://github.com/SheffieldML/GPy`, since 2012.

[10] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and vlsi. *Journal of the ACM (JACM)*, 32(1):130–136, 1985.

[11] P. W. Holland and R. E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods*, 6(9):813–827, 1977.

[12] V. V. Klemas. Coastal and environmental remote sensing from unmanned aerial vehicles: An overview. *Journal of Coastal Research*, 31(5):1260–1267, 2015.

[13] R. E. Kopp, A. C. Kemp, K. Bittermann, B. P. Horton, J. P. Donnelly, W. R. Gehrels, C. C. Hay, J. X. Mitrovica, E. D. Morrow, and S. Rahmstorf. Temperature-driven global sea-level variability in the common era. *Proceedings of the National Academy of Sciences*, 113(11):E1434–E1441, 2016.

[14] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

[15] 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*, 86(1):97–106, 1964.

[16] W. Leithead, Y. Zhang, and D. Leith. Efficient gaussian process based on bfgs updating and logdet approximation. *IFAC Proceedings Volumes*, 38(1):1305–1310, 2005.

[17] W. E. Leithead and Y. Zhang. O (n 2)-operation approximation of covariance matrix inverse in gaussian process regression based on quasi-newton bfgs method. *Communications in StatisticsSimulation and Computation* R , 36(2):367–380, 2007.

[18] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[19] K. H. Low, J. M. Dolan, and P. K. Khosla. Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing. In *ICAPS*, pages 233–240, 2009.

[20] D. J. MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[21] L. Maidment, Z. Zhang, M. D. Bowditch, C. R. Howle, and D. T. Reid. Stand-off detection of aerosols using mid-infrared backscattering fourier transform spectroscopy. In *SPIE Security+ Defence*, pages 999507–999507. International Society for Optics and Photonics, 2016.

[22] G. Marsaglia, W. W. Tsang, et al. The ziggurat method for generating random variables. *Journal of statistical software*, 5(8):1–7, 2000.

[23] NASA. Exploring Drone Aerodynamics with Computers. https://www.nasa.gov/image-feature/ames/exploring-drone-aerodynamics-with-computers, Jan 11, 2017. [Online; accessed 11-July-2017].

[24] S. Nebiker, A. Annen, M. Scherrer, and D. Oesch. A light-weight multispectral sensor for micro uavopportunities for very high resolution airborne remote sensing. *The international archives of the photogrammetry, remote sensing and spatial information sciences*, 37(B1):1193–1199, 2008.

[25] F. Pristera, M. Halik, A. Castelli, and W. Fredericks. Analysis of explosives using infrared spectroscopy. *Analytical Chemistry*, 32(4):495–508, 1960.

[26] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, 2009.

[27] J. Quinonero-Candela, C. E. Rasmussen, and C. K. Williams. Approximation methods for gaussian process regression. *Large-scale kernel machines*, pages 203–224, 2007.

[28] C. E. Rasmussen and C. K. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[29] H. resolution spectra of common explosives. Explosives analysis. http://what-when-how.com/forensic-sciences/analysis, 2017. [Online; accessed 5-April-2017].

[30] Richard Pogge. Lecture 24: Matter and Light. http://www.astronomy.ohio-state.edu/pogge/Ast161/Unit4/spectra.html, Oct 19, 2007. [Online; accessed 4-August-2017].

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

[32] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[33] C. K. Williams. Computing with infinite networks. In *Advances in neural information processing systems*, pages 295–301, 1997.

[34] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

[35] Z. Zhang, R. J. Clewes, C. R. Howle, and D. T. Reid. Active ftir-based stand-off spectroscopy using a femtosecond optical parametric oscillator. *Optics letters*, 39(20):6005–6008, 2014.