Active Learning for Robot Exploration Bayesian Optimization for Object Grasping

José Nogueira[1]

***Abstract*— Safe and robust grasping of unknown objects is a major challenge in robotics, which has no general solution yet. A promising approach relies on haptic exploration, where active optimization strategies can be employed to reduce the number of exploration trials. Two critical problems one must consider are non-convex and typically varying smoothness properties of the exploration score and that certain optimal grasps discovered by the optimization procedure may be very sensitive to small deviations of the parameters from their nominal values (unsafe grasps[3]). To reduce the risk of grasp failure, safe grasps should be favored. Therefore, we use Treed Gaussian Processs (TGPs) to model the exploration metric more correctly and propose a new algorithm, Unscented Bayesian Optimization (UBO), that performs efficient optimization while considering uncertainty in the input space, leading to the discovery of safe optima. The results highlight how our method outperforms the classical Bayesian optimization both in synthetic problems and in realistic robot grasp simulations, finding robust and safe grasps after a few exploration trials.**

I. Introduction

Learning how to grasp unknown objects can be performed in two different ways. One strategy relies on gathering extensive amounts of data from multiple sensors and learn features that allow grasping generalization. Those features are mapped to optimal grasp configurations and strategies. However, even for a simple gripper, the amount of data required for proper generalization is very large [1]. A more natural alternative is on-the-fly learning by trial and error [2]. This can be achieved with minimal visual or haptic input and it naturally generalize to multiple objects and tasks. However, on-the-fly learning has two problems. First, it can be expensive. Thus, we need an efficient grasping exploration methodology. Second, it can get lucky. We need to guarantee that the optimal grasp can be repeated in the presence of noise with sufficient quality. A brute-force approach would need to test grasps in many different configurations in search for the best grasping point and, for each configuration, repeat the test many times to average out the robot positioning uncertainty. This is clearly unfeasible in practice, and better search strategies must be devised.

*This work was partially supported by the EU project LIMOMAN [PIEF-GA-2013-628315], Portuguese project FCT [UID/EEA/50009/2013] and Spanish projects DPI2015-65962-R (MINECO/FEDER, UE) and CUD2013-05 (SIRENA).

[1]Institute for Systems and Robotics, Instituto Superior Tecnico, Universidade de Lisboa, Portugal josemscnogueira@gmail.com, alex@isr.tecnico.ulisboa.pt, ljamone@isr.tecnico.ulisboa.pt

[2]Centro Universitario de la Defensa, Zaragoza, Spain rmcantin@unizar.es

[3]small errors during motor execution may turn optimal grasps into bad grasps

In this paper, we follow the trial and error methodology for grasping and directly address the two problems. First, given an object, we must find the configuration that maximize grasp quality with a small budget. Second, we incorporate repeatability errors in the grasp configuration such as optimality is preserved without performing multiple trials. In robotics in general and, for the problem of robot grasping in particular, Bayesian optimization has been one of the most successful and efficient trial-and-error techniques [2], [3], [4], [5], even in the presence of mechanical failures [6]. In this setup, repeatability errors correspond to uncertainty in the input space.

Bayesian optimization [7], [8] is a global optimization technique for black-box functions. Because it is designed for sample efficiency, at the cost of extra computation, it is intended for functions that are expensive to evaluate (in terms of cost, energy, time). The beauty of Bayesian optimization is its capability to deal with general black-box functions, therefore being able to address the grasping problem without any extra information, just the results from previous trials. Bayesian optimization relies on a probabilistic surrogate function that is able to learn about the target function based on previous samples and, therefore, drive future sampling more efficiently. Typically, in Bayesian Optimization, the most commonly used surrogate model is the Gaussian Process (GP), which considers fixed smoothness target function properties throughout the input space. GP's may not correctly model some exploration metrics with varying smoothness behavior (this is the case for grasping optimization). Recently, some authors have begun to use other models which consider varying smoothness properties of data - Heteroscedastic [4] regression models [9], [10] and Heteroscedastic Bayesian Optimization [11], [12]. However, to the authors knowledge, the consideration of heteroscedacity in grasp optimization has yet to be addressed. In addition uncertainty in the input space has not been addressed neither in the grasp planning literature nor in the Bayesian optimization literature. There has been previous works that consider input noise in GP regression [13], however, those methods propagate the input noise to the output space, which may result in unnecessary exploration of the space for the optimization problem. Safe exploration has also been recently addressed within Bayesian optimization, but in that case, the problem is to guarantee that the outcome is above a threshold for each trial [14].

As the main contribution of this paper, we address the

[4]In Bayesian Optimization, it is a concept specifically applied to refer to non-stationarity of the objective function

problem of heteroscedacity and input noise in Bayesian optimization, which is then applied to robot grasping. For dealing with heteroscedacity we use TGPs [12] instead of the standardly used Gaussian Processes (GPs), since they use nonstacionary model parameterization over the input space, which in turn models better varying smoothness behaviors; and for the input noise, we need a system to propagate the noise distribution from the input query through all the models and decisions of our method. We solve this with the *unscented transformation* [15], [16], a method to estimate the results of applying a nonlinear transformation to a probability distribution. Both contributions' experiments will be considered separately from one another for the sake of isolating their improvements in Bayesian Optimization results.

In this paper, we present the *Unscented Bayesian Optimization* (UBO) algorithm. It has the advantages of the sample efficiency from Bayesian optimization and the capability of dealing with input noise during function queries. Applied to grasping, this means that the method can find the optimal grasp while considering the input noise for safety. Furthermore, due to the recent popularity of Bayesian optimization in many areas (e.g. autonomous algorithm tuning [17], robot planning [18], [19], control [20], [21], reinforcement learning [22], [5], sensor networks [23], etc.), this method can directly impact many other fields, that would greatly benefit from an extension to deal with input noise.

II. Bayesian optimization

Consider the problem of finding the optimum (e.g. minimum) of an unknown real valued function $f : \mathbb{X} \to \mathbb{R}$, where $\mathbb{X}$ is a compact space, $\mathbb{X} \subset \mathbb{R}^d, d \geq 1$, with a maximum budget of N evaluations of the target function f. The Bayesian optimization algorithm selects the best query points at each iteration so that the optimization gap $|y^* - y_n|$ is minimum for the available budget. This is achieved by using two ingredients. First, a probabilistic surrogate model: a distribution over the family of functions $P(f)$, where the target function $f()$ belongs, built incrementally using the sample evaluations. Second, a Bayesian decision process, that uses the information captured in the surrogate model to select the next query point in order to maximize the information about the optimum. Therefore, Bayesian optimization can be seen as an active learning approach to find the optimum. Without loss of generality, in the remainder of the paper we assume that the standard surrogate model[1] $P(f)$ is a GP $\mathcal{GP}(\mathbf{x}|\mu, \sigma^2, \boldsymbol{\theta})$ with inputs $\mathbf{x} \in \mathbb{X}$, scalar outputs $y \in \mathbb{R}$ and an associated kernel or covariance function $k(\cdot, \cdot)$ with hyperparameters $\boldsymbol{\theta}$. The hyperparameters are estimated using a Monte Carlo Markov Chain (MCMC) algorithm, i.e.: slice sampling [17], [24], resulting in m samples $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_i\}_{i=1}^m$.

Given at step n a dataset of query points $\mathbf{X} = \{\mathbf{x}_{1:n}\}$ and its respective outcomes $\mathbf{y} = \{y_{1:n}\}$, then the prediction of the GP at a new query point $\mathbf{x}_q$, with kernel k_i conditioned on the i-th hyperparameter sample $k_i = k(\cdot, \cdot|\boldsymbol{\theta}_i)$ is normally distributed, $\widehat{y}(\mathbf{x}_q) \sim \sum_{i=1}^m \mathcal{N}(\mu_i, \sigma_i^2|\mathbf{x}_q)$, where:

$$\begin{aligned} \mu_i(\mathbf{x}_q) &= \mathbf{k}_i(\mathbf{x}_q, \mathbf{X})\mathbf{K}_i^{-1}\mathbf{y} \\ \sigma_i^2(\mathbf{x}_q) &= k_i(\mathbf{x}_q, \mathbf{x}_q) - \mathbf{k}_i(\mathbf{x}_q, \mathbf{X})\mathbf{K}_i^{-1}\mathbf{k}_i(\mathbf{X}, \mathbf{x}_q) \end{aligned} \tag{1}$$

The vector $\mathbf{k}_i(\mathbf{x}_q, \mathbf{X})$ is the cross-correlation of the query point $\mathbf{x}_q$ with respect to the dataset $\mathbf{X}$ and $\mathbf{K}_i = \mathbf{K}_i(\mathbf{X}, \mathbf{X}) + \mathbf{I}\sigma_n^2$ is the Gram matrix corresponding to kernel k_i for the dataset $\mathbf{X}$, with noise variance σ_n^2. The noise term represents the observation noise in stochastic functions [25] or the nugget term for surrogate missmodeling [26]. Note that, because we use a sampling distribution of $\boldsymbol{\theta}$ the predictive distribution at any point $\mathbf{x}$ is a mixture of Gaussians.

To select the next point at each iteration, we use the *expected improvement* criterion [27] as a way to minimize the optimality gap. The expected improvement is the expectation of the improvement function $I(\mathbf{x}) = \max(0, \rho - f(\mathbf{x}))$, where ρ is an incumbent value, usually the *best outcome* until that iteration y_{best} or, for stochastic functions, the best average prediction $\widehat{y}_{best}$. Taking the expectation over the mixture of Gaussians of the predictive distribution, we can compute the expected improvement as:

$$\begin{aligned} EI(\mathbf{x}) &= \mathbb{E}_{p(y|\mathbf{x},\boldsymbol{\theta})}\left[\max(0, \rho - f(\mathbf{x}))\right] \\ &= \sum_{i=1}^{m} \left[(\rho - \mu_i)\,\Phi(z_i) + \sigma_i \phi(z_i)\right] \end{aligned} \tag{2}$$

where ϕ and Φ are the corresponding Gaussian probability density function (PDF) and cumulative density function (CDF) and $z_i = (\rho - \mu_i)/\sigma_i$. In this case, (μ_i, σ_i^2) is the prediction computed with Equation (1).

Finally, in order to reduce initialization bias and improve global optimality, we rely on an initial design of p points based on *Latin Hypercube Sampling* (LHS), as suggested in [28].

III. Treed Gaussian processes

A Treed Gaussian Process (TGP) [12] distinguishes from a GP as it is a partial nonstacionary regression model: it considers changes in the model's parametrization over the input space. In other words, it can model different smoothness target function's behaviors over the input space. The main difference that distinguishes TGPs from other heteroscedastic models [9], [10], [11] is that nonstacionarity is due to the partition of the input space, where each partition has its own hyperparameters, rather than having them vary continuously over the input space.

A. Definition

A TGP can be described as a Decision Tree (fig. 1), where each leaf node corresponds to singular GP with a respective input space compact interval. This means that for any point of the input domain, there is only one GP which models the target function in its corresponding compact interval, according to equations 1.The union of all leaves' intervals represents all the input space and the intersection of any two of these intervals is empty. $\mathcal{L}$ denotes the set of all leaves of the TGP.

[1]Except for the experiments with TGPs

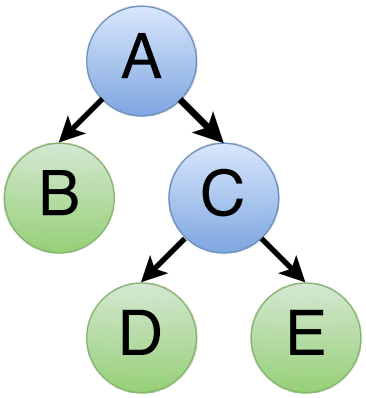


Fig. 1: Treed Gaussian Process

To determine which leaf $\in \mathcal{L}$ governs a specific query $\mathbf{x}$, this query is tested by TGP's non-leaf nodes. In each non-leaf node a binary test is performed to $\mathbf{x}$ to determine to which child node $\mathbf{x}$ belongs. This test can be explicitly written as $h_i(\mathbf{x}) > \tau$, with $i = 1..d$ and d as the number of dimensions of $\mathbf{x}$. i represents a *feature* of $\mathbf{x}$ and τ is called the *threshold*. The function h_i is expressed by:

$$h_i(\mathbf{x}) = x_i, \qquad x_i = i^{th} \text{ component of } \mathbf{x} \tag{3}$$

Each one of these binary outcomes correspond to only one of the child nodes.

B. Tree Construction

To construct the tree which models the current state of the learning process, we start with a tree composed by only one node which governs all the input space. Then, this tree is split recursively until splitting is no longer viable.

We wish to split (if possible) every node into two children nodes, resulting in a overall uncertainty reduction of the original node, while also guarantying that the two new child nodes have a minimum number of samples. This last detail is crucial for hyperparameter optimization, since with a low number of samples, hyperparameter estimation can be compromised.

The uncertainty of a node A is defined as:

$$\mathrm{U}(A) = \frac{1}{|A|} \sum_{y_i \in A} (\bar{y}_A - y_i)^2 \tag{4}$$

where $\bar{y}_A$ is the average of the output of the samples in A, $|A|$ the number of samples and y_i the output of sample i.

Node A is split on *feature* i and *threshold* τ into two child-nodes $A'_{h,\tau}$ and $A''_{h,\tau}$ if the splitting correspond to an overall uncertainty reduction, if it doesn't violate the minimum number of samples per leaf and if it maximizes the following equation:

$$\begin{aligned} \mathrm{I}\left(A, A'_{h,\tau}, A''_{h,\tau}\right) = \mathrm{U}(A) &- \frac{\left|A'_{h,\tau}\right|}{|A|} \mathrm{U}\left(A'_{h,\tau}\right) \\ &- \frac{\left|A''_{h,\tau}\right|}{|A|} \mathrm{U}\left(A''_{h,\tau}\right) \end{aligned} \tag{5}$$

Equations 3 and 5 imply that these splits occur at the sampled points. Therefore we have a finite and discrete set of *features* i and *thresholds* τ from which we wish to maximize eq. 5. It was also shown [12] that this strategy allows to maintain low variance in the vicinity of the splits [2]. In this paper we maximize the previous equation with *brute-force* search.

C. Hyperparameter estimation

We use the log-marginal-likelihood to estimate the hyperparameters for each GP. If we consider a set of samples of one GP, the log-marginal-likehood wields:

$$\begin{aligned} 2\log p\left(\mathbf{y}|\mathbf{x}_{1:t}, \theta\right) = &- \mathbf{y}^{\intercal}\left(\mathbf{K}_t^{\theta} + \sigma^2 I\right)^{-1}\mathbf{y} \\ &- \log\left|\mathbf{K}_t^{\theta} + \sigma^2 I\right| - t\log(2\pi) \end{aligned} \tag{6}$$

For TGPs, an aggregation technique is performed which allows a specific GP associated with leaf $j \in \mathcal{L}$ to optimize hyperparameters with its respective samples as well as the samples from other leafs.

For the sake of notation, let $\mathbf{y}_{(j)}$ denote the data in node j, let $\mathbf{y}_{(j\setminus i)}$ denote the data in node j excluding the data in node i. Let δ^j be the depth of node j such that the root node has depth equal to zero. Let ρ^j be the list of nodes in the path from node j to the root, and let ρ_i^j be the i-th element in the list ρ^j, such that $\rho_0^j = j$ and $\rho_{\delta^j}^j = 0$ (root).

We then consider the *weighted* marginal pseudo-likelihood decomposition [12] as follows:

$$\begin{aligned} p\left(\mathbf{y}|\mathbf{x}_{1:t}, \theta\right) \approx\ & p^{w_0^j}\left(\mathbf{y}_{(j)}|\mathbf{x}_{(\mathbf{j})}, \theta\right) \\ & \times \prod_{i=1}^{|\rho^j|} p^{w_i^j}\left(\mathbf{y}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}|\mathbf{x}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}, \theta\right) \end{aligned} \tag{7}$$

Using eqs. 6 and 7, we obtain the *weighted* log-marginal-likehood:

$$\begin{aligned} \log p\left(\mathbf{y}|\mathbf{x}_{1:t}, \theta\right) = & w_0^j \log p\left(\mathbf{y}_{(j)}|\mathbf{x}_{(\mathbf{j})}, \theta\right) \\ & + \sum_{i=1}^{|\rho^j|} w_i^j \log p\left(\mathbf{y}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}|\mathbf{x}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}, \theta\right) \end{aligned} \tag{8}$$

One should think of the set $\left(\mathbf{y}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}, \mathbf{x}_{\left(\rho_i^j \setminus \rho_{i-1}^j\right)}\right)$ as all samples of the i-th order parent of node j except the samples of the $(i-1)$-th order parent of node j. For example, the 0 order parent of node j is itself, the 1-st order parent is its direct *father* node and the 2-nd order parent would be its *grandfather* node.

The purpose of using this aggregation technique for estimating the hyperparameters is not only to make leafs not totally independent from each other in terms of regression, but also to allow lower values for the minimum number of samples per leaf so hyperparameter optimization is not compromised. But one should notice that by reinforcing this

[2] the surrogate model variance is the lowest at the sampled queries

cross-effect we can also lose the opportunity to model the objective function with more accuracy.

As for the weights w_i^j, we propose a different approach from [12], in light of what was said in the previous paragraph. In their work they use a fixed formula to calculate weights. We explore alterations to this formula and compare their performance. We consider the weights as:

$$w(k)_i^j = \left(\frac{2}{1+\delta^j-\delta^i}\right)^k \tag{9}$$

Higher values of k promote greater independence between leafs when estimating the hyperparameters. In [12] $k=1$. We note the set of all weights over j and i as $\mathbf{w}^k$. We also note, from now forward, $\mathbf{w}^\infty$ as the case where each leaf only uses its own data for hyperparameter estimation, i.e. $w(\infty)_0^j = 1$ and $w(\infty)_i^j = 0, \forall i \neq 0$.

IV. Unscented Bayesian optimization

In this paper, we propose to consider the input noise during the decision process to explore and select the regions that are safe. That is, the regions that guarantee good results even if the experiment/trial is repeated several times. In this respect, our contribution is twofold: we present the *unscented expected improvement* (Sec. IV-B) and the *unscented optimum incumbent* (Sec. IV-C). Both methods are based on the *unscented transformation* (Sec. IV-A), initially developed for tracking and filtering applications [29], [15].

A. Unscented transformation

The unscented transformation is a method to propagate probability distributions through nonlinear transformations with a trade off of computational cost vs accuracy. It is based on the principle that *it is easier to approximate a probability distribution than to approximate an arbitrary nonlinear function* [15]. The unscented transformation uses a set of deterministically selected samples from the original distribution (called *sigma points*) and transform them through the nonlinear function $f(\cdot)$. Then, the transformed distribution is computed based on the weighted combination of the transformed sigma points.

The advantage of the unscented transformation is that the mean and covariance estimates of the new distribution are accurate to the third order of the Taylor series expansions of $f(\cdot)$ provided that the original distribution is a Gaussian prior, or up to the second order of the expansion for any other prior. Fig. 2 highlights the differences between approximating the distribution using sigma points or using standard first-order Taylor linearization. The distribution from the UT is closer to the real distribution. Because the prior and posterior distributions are both Gaussians, the unscented transformation is a linearization method. However, because the linearization is based on the statistics of the distribution, it is often found in the literature as *statistical linearization*.

Another advantage of the unscented transformation is its computational cost. For a d-dimensional input space, the unscented transformation requires a set of $2d+1$ sigma points. Thus, the computational cost is negligible compared

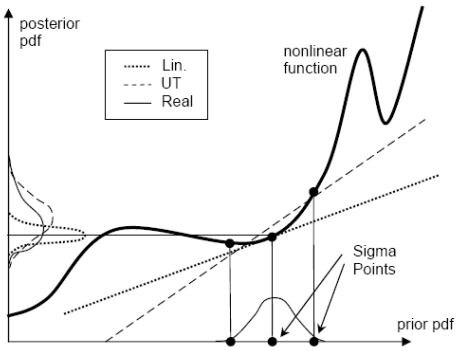


Fig. 2: Propagation of a normal distribution through a nonlinear function. The first order Taylor expansion (dotted) only uses information of the function at the mean point to compute the linear approximation, while the UT (dashed) approaches the function with a linear regression of several sigma points. The actual distribution is the solid one. (Adapted from [16])

to other alternatives to Bayesian approximation such as Monte Carlo, which requires a large number of samples, or numerical integration such as *Gaussian quadrature*, which has an exponential cost on d. Van der Merwe [16] proved that the unscented transformation is part of the more general *sigma point filters*, which achieve similar performance results.

1) Computing the unscented transformation: Assuming that the prior distribution is a Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \Sigma_{\mathbf{x}})$, then the $2d+1$ sigma points of the unscented transformation are computed by

$$\begin{aligned}
\mathbf{x}^0 &= \bar{\mathbf{x}} \\
\mathbf{x}_+^{(i)} &= \bar{\mathbf{x}} + \left(\sqrt{(d+k)\Sigma_{\mathbf{x}}}\right)_i \qquad \forall\, i = 1 \ldots d \\
\mathbf{x}_-^{(i)} &= \bar{\mathbf{x}} - \left(\sqrt{(d+k)\Sigma_{\mathbf{x}}}\right)_i \qquad \forall\, i = 1 \ldots d
\end{aligned} \tag{10}$$

where $(\sqrt{\cdot})_i$ is the i-th row or column of the corresponding matrix square root. In this case, k is a free parameter that can be used to tune the scale of the sigma points. For optimal values of k, see [15]. For these sigma points, the weights are defined as:

$$\begin{aligned}
\omega^0 &= \frac{k}{d+k} \\
\omega_+^{(i)} &= \frac{1}{2(d+k)} \qquad \forall\, i = 1 \ldots d \\
\omega_-^{(i)} &= \frac{1}{2(d+k)} \qquad \forall\, i = 1 \ldots d
\end{aligned} \tag{11}$$

Then, the transformed distribution is $\mathbf{x}' \sim \mathcal{N}(\bar{\mathbf{x}}', \Sigma'_{\mathbf{x}})$, where:

$$\bar{\mathbf{x}}' = \sum_{i=0}^{2d} \omega^{(i)} f(\mathbf{x}^{(i)}) \tag{12}$$

B. Unscented expected improvement

Bayesian optimization is about selecting the most interesting point at each iteration. Usually, this is achieved by a greedy criterion, such as the *expected improvement*, the

upper confidence bound or the *predictive entropy*. These criteria, also denoted *acquisition functions*, select the query point that has the higher potential to become the optimum, assuming that the query is deterministic. However, in our case, the query is a probability distribution due to input noise. Thus, instead of analysing the outcome of the criterion, we are going to analyse the resulting posterior distribution of transforming the query distribution through the acquisition function.

For the purpose of safe Bayesian optimization, we will use the expected value of the transformed distribution as the acquisition function. Therefore, we define the unscented expected improvement as:

$$UEI(\mathbf{x}) = \sum_{i=0}^{2d} \omega^{(i)} EI(\mathbf{x}^{(i)}) \tag{13}$$

where $\mathbf{x}^{(i)}$ and $\omega^{(i)}$ are computed according to equations (10) and (11) respectively. The expected value of the transformed distribution $\bar{\mathbf{x}'} = UEI(\mathbf{x})$ is enough to take a decision considering the risk on the input noise. Anyway, the value of $\Sigma'_{\mathbf{x}}$ represents the output uncertainty and can also be used as meta-analysis tool.

C. Unscented optimal incumbent

The unscented expected improvement can be used to drive the search procedure towards safe regions. However, because the target function is unknown by definition, the sampling procedure can still query good outcomes in unsafe areas. Furthermore, in Bayesian optimization there is a final decision that is independent of the acquisition function employed. Once the optimization process is stopped after sampling N queries, we still need to decide which point is the *best*. Moreover, after every iteration, we need to say which point is the incumbent. If the final decision about the incumbent selects the sample with best outcome $\mathbf{x}^*$ such that $y_{best} = f(\mathbf{x}^*)$ we may select an unsafe query. Instead, we propose to apply the unscented transformation also to the select the optimal incumbent $\mathbf{x}^*$, based on the function outcome $f()$ at the sigma points. This would require additional evaluations of $f()$, but the main idea of Bayesian optimization is to reduce the number of evaluations on $f()$. Instead of evaluating $f()$ at the sigma points, we evaluate the sigma points at the GP surrogate average prediction $\mu()$.

Therefore, we define the unscented outcome (UO) as:

$$UO(\mathbf{x}) = \sum_{i=0}^{2d} \omega^{(i)} \sum_{j=1}^{m} \mu_j(\mathbf{x}^{(i)}) \tag{14}$$

where $\sum_{j=1}^{m} \mu_j(\mathbf{x}^{(i)})$ is the prediction of the GP according to equation (1) integrated over the kernel hyperparameters and at the sigma points of equation (10). Under these conditions, the incumbent of the optimal solution $\mathbf{x}^*$ corresponds to:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \; UO(\mathbf{x}) \tag{15}$$

V. Results

In this section we describe the experiments carried out in this work. Results will be evaluated separately into two main subsections: one for comparing BO-TGP against BO-GP (or simply BO); the other to compare the benefits of the Unscented Bayesian Optimization (UBO) with respect to the classical Bayesian optimization (BO). We opted to use GPs for the Unscented experiments since they are used as standard regression models in BO and it makes comparison more standard for both contributions.

The two main goal are to demonstrate that: by using TGP, we can better model input space regions with different smothness properties, which allows better model regression and improves the learning criterion effectiveness by giving more accurate information about the target function. This makes BO have better results; as for the second contribution, by using the UBO, we minimize the risk of choosing unsafe global optima.

We first illustrate both methods applied to synthetic functions, to clearly visualize the importance of better target function modeling/selecting safe optima. Then, we show the results of doing autonomous exploration of daily life objects with a dexterous robot hand using realistic simulations, reproducing the conditions of a real robot setup.

In this work we have used and extended the BayesOpt software [24] with the proposed methods. For the GP kernel, we used the standard choice of the Matérn kernel with $\nu = 5/2$. Without loss of generality, we assume that the input noise is white Gaussian, isotropic and stationary, i.e. $\mathcal{N}(0, \mathbf{I}\sigma_x)$, although the method can be applied to anisotropic, nonstationary and even non-Gaussian noise.

To reproduce the effect of the input noise, we queried the result of each method using Monte Carlo samples according the input noise distribution at the incumbent point at each iteration $\{y_{mc}^{(i)}(\mathbf{x}^*)\}$. By analyzing the outcome of the samples we can estimate the expected outcome from the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$ and the variability of outcomes $std(y_{mc}(\mathbf{x}^*))$. As we can see in the results, our method is able to provide equal or better expected outcomes while reducing the variability of those same outcomes.

A. Synthetic Functions

The synthetic functions presented here are the 1D Reproducing Kernel Hilbert Space Function (RKHS) [30], Gramacy 2-D Exponential Function (GF) [31] and a specially designed Mixture of 2D Gaussian distributions (GM) (courtesy of Ruben Martinez-Cantin), see Fig.3.

1) BO-GP Vs. BO-TGP: We have performed 100 runs of Bayesian Optimization (BO) for all three functions (RKHS, GF, GM) and the optimization procedure using GP and TGP (see eq. 9).

For RKHS, each run has 5 initial random samples and the optimization performs 45 iterations, with $\sigma_p = 3.93$, $s_l = 8$ [3]; for GF, each run has 20 initial samples and

[3] minimum number of samples per TGP leaf

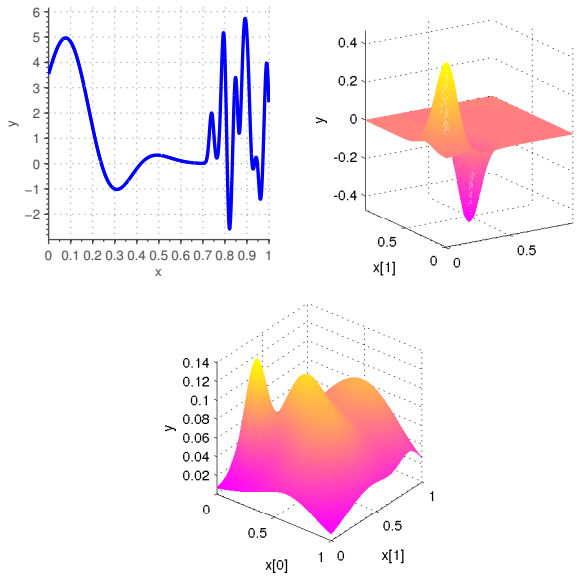
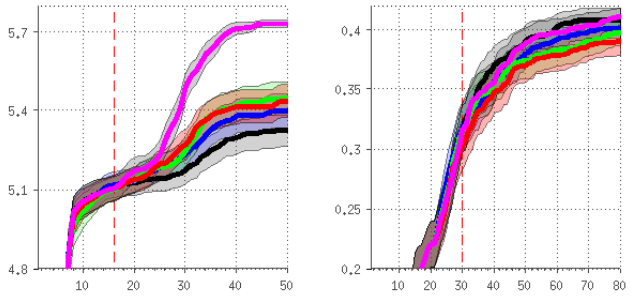
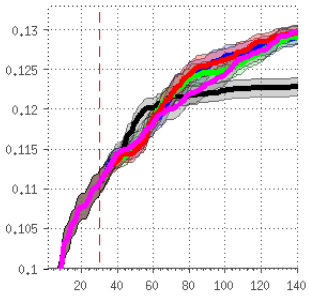


Fig. 3: Left: RKHS function [30]. Right: GF function [31]. Middle: GM function



(a) RKHS (b) GF



(c) GM

Fig. 4: GP Vs. TGP Results. Synthetic Functions. Best Sample observation y over the number of iterations.
Black - Bayesian Optimization using Gaussian Processes (BO-GP). Blue - Bayesian Optimization using Treed Gaussian Processes (BO-TGP) and $\mathbf{w}^1$. Green - BO-TGP and $\mathbf{w}^2$. Red - BO-TGP and $\mathbf{w}^3$. Magenta - BO-TGP and $\mathbf{w}^\infty$. Dashed Red - First TGP node split. Shaded regions represent the standard deviation of the best observation at each iteration.

the optimization performs 40 iterations, with $\sigma_p = 0.2$ [4], $s_l = 15$; for GM, each run has 30 initial samples and the optimization performs 110 iterations, with $\sigma_p = 0.642$, $s_l = 15$. For all functions it was used $\sigma_n = 10^{-6}$.

	$\bar{y}(\mathbf{x}^*) \pm \sigma(y(\mathbf{x}^*))$				
Function	GP	TGP $\mathbf{w}^1$	TGP $\mathbf{w}^2$	TGP $\mathbf{w}^3$	TGP $\mathbf{w}^\infty$
RKHS	5.3279± 0.0642	5.3993± 0.0638	5.4472± 0.0621	5.4375± 0.0627	**5.7305± 0.0128**
GF	0.4081± 0.0093	0.4008± 0.0079	0.3974± 0.0108	0.3900± 0.0118	**0.4106± 0.0074**
GM	0.1228± 0.0011	0.1295± 0.0001	0.1297± $8.1e-04$	0.1302± $8.3e-04$	**0.1309± $6.5e-04$**

TABLE I: Results at the last iteration of the BO process (means and standard deviations over all runs).

Results for these experiments are shown in figure 4 and table I.

From RKHS and GM results we can clearly see that BO-TGP finds the optimal value more frequently (and quicker) than BO-GP. This happens since TGP can have different kernel bandwidths l_i over the input space and, therefore, better estimate the expected value and variance of the objective function $f()$. This improves exploration with the given learning criterion and allows BO to use the sample budget more efficiently. BO may search for other possible local optima instead of wasting samples where variance would be otherwise high (and low expected value) like in GPs.

As for the GF function, BO-TGP doesn't show significant better results than BO-GP. While TGPs help BO by better estimating the objective function and improve multiple local optima exploration, the GF function only has one global/local maximum. GPs may give a worse estimation for GF but, by having higher overall process variance, they tend to make BO reach the single global optimum with similar convergence rate.

2) BO Vs. UBO: We performed 100 runs of each optimization procedure (BO and UBO) for each function (RKHS, GM). We used 100 Monte Carlo samples for $\{y_{mc}^{(i)}(\mathbf{x}^*)\}$. For RKHS each run has 5 initial samples with LHS and the optimization performs 45 iterations. The input noise is set as $\sigma_x = 0.01$. For GM each run has 30 initial samples and the optimization performs 90 iterations. The input noise is set as $\sigma_x = 0.1$. In Fig. 5 and Fig. 6 we show the statistics over the different runs for the evaluation criteria, as a function of the number of iterations. The shaded region represents the 95% confidence interval. Being deterministic functions, we used $\sigma_n = 10^{-6}$ as nugget in both cases.

For both functions, we can observe that UBO quickly overcomes the results of BO. UBO computes less risky solutions, as demonstrated by the higher expected return

[4]Is is common to randomly sample the objective function to estimate σ_p as the standard deviation of those samples. Since this may not be possible for all cases, we assume the worst possible case, where the objective function only two possible values - the global maximum and global minimum. This wields $\sigma_p = \sqrt{\frac{f(\mathbf{x}^+) - f(\mathbf{x}^-)}{2}}$

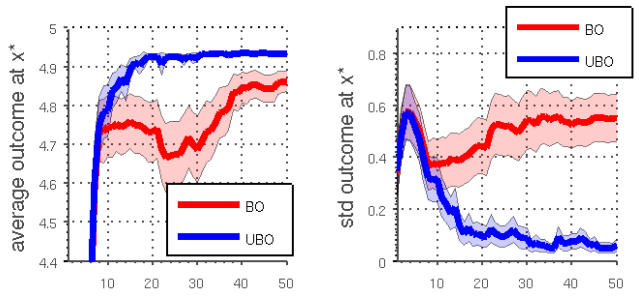


Fig. 5: RKHS Results. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$

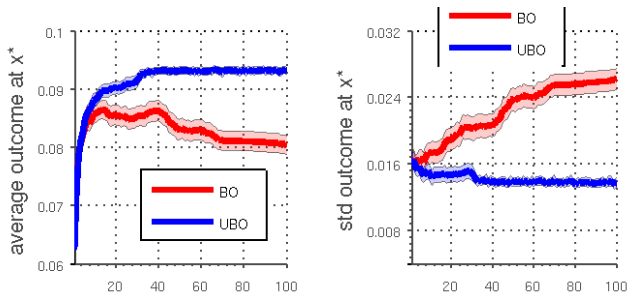


Fig. 6: GM. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$

value and lower standard deviation. In Table III we show the numeric results obtained at the last iteration, as well as the values of the worst sample of the Monte Carlo runs. The worst case for UBO is always more favourable than the worst case for BO by a large margin.

B. Robot Grasp Simulations

We use the Simox simulation toolbox for robot grasping [32]. This toolbox simulates the iCub robot hand grasping arbitrary objects. Given an initial pose for the robot hand and a finger joint trajectory, the simulator runs until the fingers are in contact with the object surface and computes a grasp quality metric based on wrench space analysis. We use a representation of the iCub left hand which can move freely in space and a few static objects (see Fig. 7).

The robot hand is initially placed with palm facing parallel to one of the facets, at a fixed distance of the object bounding box, and the thumb aligned with one of the neighbor facets.

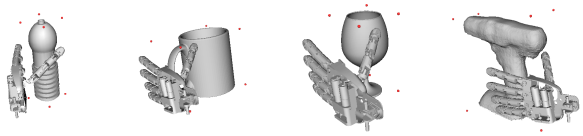


Fig. 7: Objects used in the simulations with corresponding initial robot hand configuration. Left to right: bottle, mug, glass and drill.

This setup uniquely defines the default pose of the hand with respect to the object. The learning goal is then to find the optimal grasp pose by choosing incremental translations and rotations $((\delta_x, \delta_y, \delta_z, \theta_x, \theta_y, \theta_z))$ of the hand's pose with respect to its default pose. A power grasp synergy was adopted for the hand closure.

1) BO-GP Vs. BO-TGP: We have performed 30 runs of BO for all the proposed objects (Water bottle, Mug, Cup of Glass, Drill) and the optimization procedure using GP and TGP (see eq. 9). The water bottle object was evaluated in two different facets (one from the side[1] and one from the top[2]), while all the other objects were only evaluated from one of the sideways facet.

Each run has 30 initial samples and the optimization performs 120 iterations, with $\sigma_p = 0.35$, $\sigma_n = 10^{-4}$. The input search space is composed by $(\delta_x, \delta_y, \delta_z, \theta_x, \theta_y, \theta_z, s_1)$.

	$\bar{y}(\mathbf{x}^*) \pm \sigma(y(\mathbf{x}^*))$				
Object	GP	TGP $\mathbf{w}^1$	TGP $\mathbf{w}^2$	TGP $\mathbf{w}^3$	TGP $\mathbf{w}^\infty$
Bottle[1]	0.5867± 0.0339	**0.6080**± 0.0231	0.6008± 0.0239	0.5775± 0.0320	0.6001± **0.0222**
Bottle[2]	0.5200± 0.0346	0.5567± 0.0260	0.5546± **0.0194**	0.5689± 0.0261	**0.5761**± 0.0255
Mug	0.1704± 0.0138	0.1620± 0.0145	0.1678± 0.0180	**0.1847**± 0.0156	0.1778± **0.0135**
Glass	0.4526± 0.0263	0.4394± **0.0180**	0.4444± 0.0267	**0.4803**± 0.0229	0.4584± 0.0260
Drill	**0.1328**± 0.0095	0.1209± **0.0080**	0.1227± 0.0087	0.1216± 0.0117	0.1239± 0.0096

TABLE II: Results at the last iteration of the BO process (means and standard deviations over all runs).

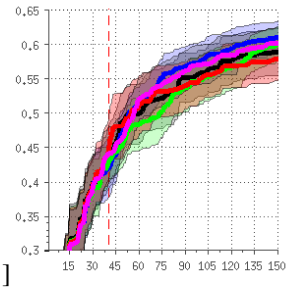
Results for these experiments are shown in figure 8 and table II.

Aside from the experiment from figure 8(e) - Drill - higher values of k seem to have better results overall - specifically $k = 3$ and $k = \infty$ (similar to the synthetic results). Still, it is not possible to conclude that as we increase k we get progressively better in BO-TGP. As k models data cross-correlation between leafs, different k might have different BO-TGP performances according to the objective function's characteristics.

2) BO Vs. UBO: We performed 30 runs of the robotic grasp simulation for each object and each optimization criterion. The robot hand posture with respect to the objects was initialized as shown in Fig. 7. We used 20 samples to compute $\{y_{mc}^{(i)}(\mathbf{x}^*)\}$.

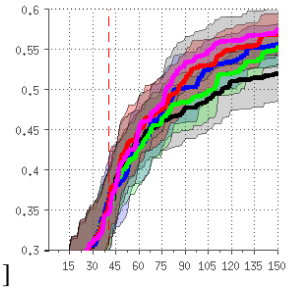
Each run starts with 40 initial samples with LHS and proceeds with 60 iterations of optimization. We assume the grasp quality metric to be stochastic, due to small simulation errors and inconsistencies, and we set $\sigma_n = 10^{-4}$. Also, we assume an input noise $\sigma_x = 0.03$ (note that the input space was normalized in advance to the unit hypercube $[0, 1]^d$). The results are shown in Fig. 9, 10, 11 and 12.

It can be seen that, for the bottle and glass, the UBO method has clear advantages over BO. UBO obtains higher mean values and lower standard deviations. For the drill, UBO eventually overcomes BO after few iterations, which might imply that the unsafe optimum is difficult to find, but it's still reachable within the given budget. Looking at the



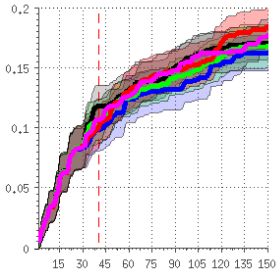
]

(a) Bottle[1

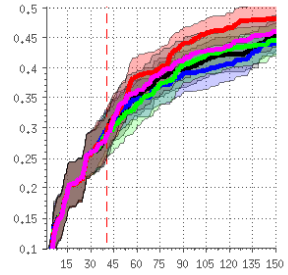


]

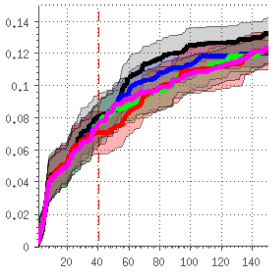
(b) Bottle[2



(c) Mug



(d) Glass



(e) Drill

Fig. 8: GP Vs. TGP Results. Simox Simulation Environment. Black - BO-GP. Blue - BO-TGP and $\mathbf{w}^1$. Green - BO-TGP and $\mathbf{w}^2$. Red - BO-TGP and $\mathbf{w}^3$. Magenta - BO-TGP and $\mathbf{w}^\infty$. Dashed Red - First TGP node split. Shaded regions represent the standard deviation of the best observation at each iteration.

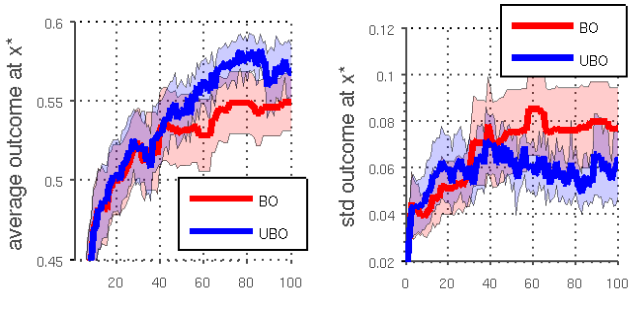


Fig. 9: Bottle. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$

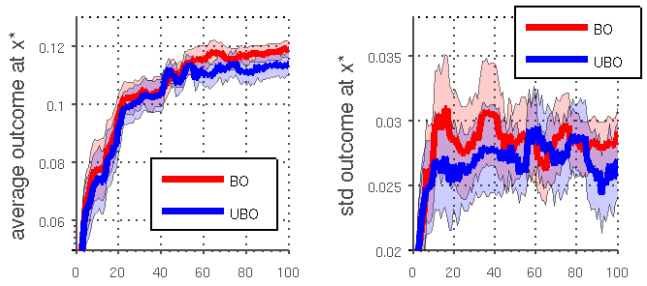


Fig. 10: Mug. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$

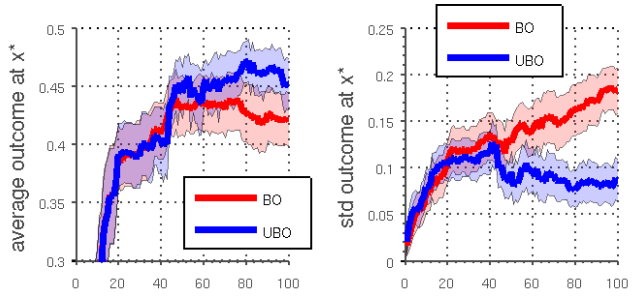


Fig. 11: Glass. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$

quantitative results shown in Table III, we can see that, at the end of the optimization, UBO is better than BO in all criteria, except for the mean output value for the mug. For the mug, the 100 trials are not enough to obtain better mean values. We can see that the mug and drill objects are more challenging due to their non-rotational symmetry. Since the optimization is only done in translation parameters, the method is missing exploration in the rotation degrees of freedom. Furthermore, in the mug case, the facet chosen was the one that contains the handle. Trying to learn a grasp in this setting is much harder than the other cases since, for the same input space volume, the number of configurations which return a good metric is much smaller. This deteriorates GP regression and

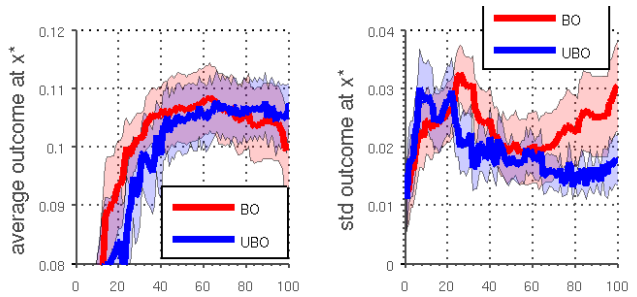
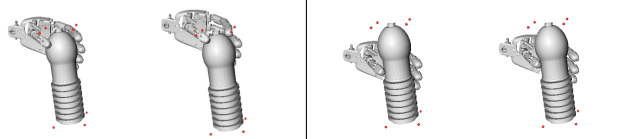


Fig. 12: Drill. Left: Expected outcome at the current optimum $\bar{y}_{mc}(\mathbf{x}^*)$, Right: Variability of the outcome $std(y_{mc}(\mathbf{x}^*))$



(a) $y = 0.439$ (b) $y = 0.377$ (c) $y = 0.413$ (d) $y = 0.418$

Fig. 13: Grasp safety. In this example the best grasp is at an unsafe zone (a), because some bad grasps can be found in its vicinity (b). The unscented Bayesian optimization chooses grasps with lower risk at a safe zone (c) and (d), where performance is robust to input noise.

	$\bar{y}_{mc}(\mathbf{x}^*)$		worst $y_{mc}(\mathbf{x}^*)$		$std(y_{mc}(\mathbf{x}^*))$	
Exp.	BO	UBO	BO	UBO	BO	UBO
RKHS	4.863	**4.934**	2.881	**4.657**	0.554	**0.065**
GM	0.080	**0.093**	0.023	**0.053**	0.027	**0.014**
Bottle	0.550	**0.567**	0.390	**0.430**	0.077	**0.065**
Mug	**0.119**	0.114	0.051	**0.059**	0.029	**0.027**
Glass	0.421	**0.452**	0.080	**0.252**	0.184	**0.087**
Drill	0.101	**0.108**	0.050	**0.068**	0.030	**0.018**

TABLE III: Results at the last iteration of the Bayesian optimization process (means over all runs). In this case, *worst* represents the sample with worst outcome y_{mc}.

hinders Bayesian optimization performance in general. An alternative in this case would be to use a nonstationary kernel [33].

In Fig. 13 we illustrate four grasps at the water bottle explored during the experiments. Two of the grasps are performed in a safe region while the two other are explored at a unsafe region. Although the unsafe zone has one observation with the highest value, it has also higher risk of getting a low value observation in its vicinity.

VI. Conclusion

This article presented novel Bayesian Optimization methods for robust grasping, namely Gaussian Processes Vs. Treed Gaussian Processes and Bayesian Optimization Vs. Unscented Bayesian Optimization. Both contributions were proposed to answer classical BO problems: heteroscedasticity and proper input noise modeling. These problems affect performance of BO in many active learning problems. The specific application here was to find the optimal way to grasp an object (using the Simox simulator).

Results shown that both implementations outperformed classic BO for the synthetic problem cases. BO-TGP achieves better results when the objective functions presents multiple local optima and different smoothness behaviours while Unscented Bayesian Optimization (UBO) finds much *safer* optima when we consider the presence of input noise.

We couldn't conclude that using higher degrees of independency between TGP leaf's data for lack of consistent results. Despite this, our synthetic results (and some from Simox's experiments) suggest that less data correlation between TGP leaf's leads to higher rates of learning convergence.

UBO results demonstrated that using plain BO may lead to inconsistent final optima values when consider input noise. We have shown that using UBO, while considering input noise conditions, leads to better average optima when comparing with BO.

We these two methods with the main purpose of perform *robust and safe grasping* of unknown objects by haptic exploration. The potential interest of both these methods goes beyond grasping or even robotics. Bayesian optimization is currently being used in many applications: engineering, computer sciences, economics, simulations, experimental design, biology, artificial intelligence. In all those fields, there are many situations where input noise, heteroscedasticity or uncertainty may arise, and in which safe optimization is therefore fundamental.

References

[1] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Rob. Res.*, vol. 34, no. 4-5, pp. 705–724, 2015.

[2] O. Kroemer, R. Detry, J. Piater, and J. Peters, "Active learning using mean shift optimization for robot grasping," in *IROS*, 2009.

[3] ——, "Combining active learning and reactive control for robot grasping," *Robotics and Autonomous systems*, vol. 58, no. 9, pp. 1105–1116, 2010.

[4] A. Boularias, J. A. Bagnell, and A. Stentz, "Efficient optimization for autonomous robotic manipulation of natural objects." in *AAAI*, 2014, pp. 2520–2526.

[5] C. Daniel, O. Kroemer, M. Viering, J. Metz, and J. Peters, "Active reward learning with a novel acquisition function," *Autonomous Robots*, vol. 39, no. 3, pp. 389–405, 2015.

[6] A. Cully, J. Clune, D. Tarapore, and J. B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, p. 503507, 2015.

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

[8] E. Brochu, V. Cora, and N. Freitas, "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," 2010, arXiv:1012.2599v1.

[9] Q. Le, A. Smola, and S. Canu, "Heteroscedastic Gaussian process regression," Australian National University, 0200 ACT, Australia, Tech. Rep., 2005.

[10] K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard, "Most likely heteroscedastic Gaussian process regression," University of Freiburg, Tech. Rep., 2007.

[11] S. Kuindersma, R. Grupen, and A. Barto, "Variational Bayesian optimization for runtime risk-sensitive control," University of Massachussets Amherst, Tech. Rep., 2012.

[12] J. Assel, Z. Wang, B. Shahriari, and N. Freitas, "Heteroscedastic treed Bayesian optimizaion," 2015, arXiv:1410.7172v2.

[13] J. McHutchon and C. Rasmussen, "Gaussian process training with input noise," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 1341–1349.

[14] F. Berkenkamp, A. Schoellig, and A. Krause, "Safe controller optimization for quadrotors with Gaussian processes," in *Proc. of the International Conference on Robotics and Automation (ICRA)*, 2016, pp. 491–496.

[15] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, March 2004.

[16] R. van der Merwe, "Sigma-point Kalman filters for probabilistic inference in dynamic state-space models," Ph.D. dissertation, OGI School of Science & Engineering, Oregon Health & Science University, 2004.

[17] J. Snoek, H. Larochelle, and R. Adams, "Practical Bayesian optimization of machine learning algorithms," in *NIPS*, 2012, pp. 2960–2968.

[18] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6136–6143.

[19] R. Martinez-Cantin, N. de Freitas, E. Brochu, J. Castellanos, and A. Doucet, "A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot." *Autonomous Robots*, vol. 27, no. 3, pp. 93–103, 2009.

[20] M. Tesch, J. Schneider, and H. Choset, "Adapting control policies for expensive systems to changing environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[21] R. Calandra, A. Seyfarth, J. Peters, and M. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," *Annals of Mathematics and Artificial Intelligence (AMAI)*, vol. 1 1, pp. 1–19 1–19, 2015.

[22] R. Martinez-Cantin, N. de Freitas, A. Doucet, and J. A. Castellanos, "Active policy learning for robot planning and exploration under uncertainty," in *Robotics: Science and Systems*, 2007.

[23] R. Garnett, M. A. Osborne, and S. J. Roberts, "Bayesian optimization for sensor set selection," in *Information Processing in Sensor Networks (IPSN 2010)*, 2010.

[24] R. Martinez-Cantin, "BayesOpt: A Bayesian optimization library for nonlinear optimization, experimental design and bandits," *Journal of Machine Learning Research*, vol. 15, pp. 3735–3739, 2014.

[25] D. Huang, T. T. Allen, W. I. Notz, and N. Zheng, "Global optimization of stochastic black-box systems via sequential kriging meta-models," *Journal of Global Optimization*, vol. 34, no. 3, pp. 441– 466, 2006.

[26] R. B. Gramacy and H. K. H. Lee, "Cases for the nugget in modeling computer experiments," *Statistics and Computing*, vol. 22, pp. 713–722, 2012.

[27] J. Mockus, *Bayesian Approach to Global Optimization*, ser. Mathematics and Its Applications. Kluwer Academic Publishers, 1989, vol. 37.

[28] A. D. Bull, "Convergence rates of efficient global optimization algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2879–2904, 2011.

[29] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000.

[30] Z. Wang, J. Assel, and N. Freitas, "RKHS 1d function for Bayesian optimization tasks," https://github.com/iassael/function-rkhs, accessed: 2015-10-03.

[31] R. Gramacy and H. Lee, "Bayesian treed Gaussian process models," University of California, Santa Cruz, Tech. Rep., 2006.

[32] V. Vahrenkamp, "Simox: A lightweight simulation and motion planning toolbox for C++," http://simox.sourceforge.net/.

[33] R. Martinez-Cantin, "Locally-biased Bayesian optimization using non-stationary Gaussian processes," in *Neural Information Processing Systems (NIPS) workshop on Bayesian Optimization*, 2015.