Active Tactile Exploration for Grasping

Filipe Veiga[1] and Alexandre Bernardino[2]

***Abstract*— This paper addresses the problem of robotic grasp optimization. Due to uncertainty both on the robot kinematics, motor control and object perception, it is very hard to analytically compute good grasps and execute them successfully. Our approach is based on searching for the best grasp configurations by iteratively optimizing a suitable grasp criterion. This approach may be compared to human learning stages where infants learn by trial and error what are the best grasping strategies. Initial grasps are often unsuccessful, but after a few trials the system learns to adapt to the uncertainties in the environment.**

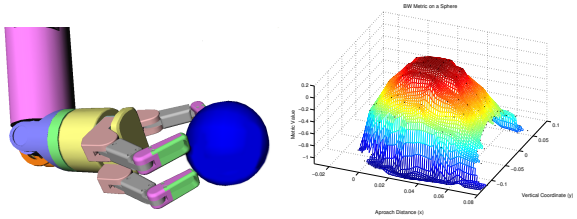

Fig. 1. By sampling the grasp input parameter space the system draws the mapping from grasp input parameters to grasp quality.

I. Introduction

Robot grasp planning and optimization have been hot topics in the last decade and are still under very active research. Most of the proposed approaches rely on off-line sampling strategies: candidate grasps are generated according to some criteria and then ranked with some quality metric obtained in simulation environments. Uniform sampling around heuristic pre-grasps [1], [2], simulated annealing [3], randomized trees [4] and active learning [5] are state-of-the-art techniques to generate grasp candidates. However, such approaches often assume the availability of "perfect" models of the robot kinematics, control and object geometry, which does not apply in practice. The problem is very challenging since it involves the integration of multiple robot skills: object perception, motor control, force and tactile sensing, robot kinematics and dynamics. Each of these skills introduce some sort of uncertainty either due to noise in sensors or modeling (calibration) errors. Because of such errors, deterministic approaches to grasp planning are brittle and prone to failures. Under uncertainty and systematic errors, more precise models of the grasping problem can be obtained by learning approaches. Trial and error approaches can be used to learn models of the grasping process but exploration and feedback strategies must be carefully chosen otherwise the problem may become intractable due to the large dimension of its state-space. In this work we propose a learning approach that tries to emulate how humans use their hands to explore, restrain and manipulate objects, adapting to uncertainties in the models. We propose the use of Bayesian Optimization methods [6] to tackle the problem of efficient exploration. Our work exploits a sequential sampling strategy, where the results from previous trials convey information to guide the next samples. We show experimentally that, depending on object's shape properties, sequential decision may reduce significantly the number of trials necessary to achieve quasi-optimal grasps with respect to random search. Despite being recently used in machine learning applications such as learning robot parameters [7], learning neural network weights [8], finding policies for robot path planning [9], etc, we introduce Bayesian Optimization methods to the robot grasping problem.

We consider in this work robot hands with the ability to detect forces and contact points with objects. Most recent robotic hands have some sort of tactile or force/torque sensing allowing this ability. In these cases grasps can be evaluated by a quality metric through the analysis of the force/torque pairs (wrenches [10]) applied by the hand on the object. Several of these metrics have been proposed in the literature, often using some form of wrench space analysis. The one we use is based on [11] and provides quality measures for both force-closure [10] and non-force-closure grasps. Force-closure grasp quality depends on the highest magnitude wrench that the grasp can resist in any direction. For non-force-closure grasp the quality depends on how distant the grasp is to being force-closure. The evaluation of non-force-closure grasps is not common in grasp planning literature but we find the additional information obtained from non-force-closure grasp assessment to be extremely useful. The proposed method consists in performing consecutive grasp trials, changing the grasp parameters (for instance hand-object relative pose, hand closure strategy, etc) until good grasps are achieved. Each grasp trial should use as much as possible the information obtained in the previous, ones in order to minimize the number of trials to reach a good grasp. The objective is to optimize the grasp parameters such as to obtain the highest quality grasp metric value.

This paper presents our approach and implementation to the problem of grasp planning through the use of Bayesian Optimization Methods. Section II will describe the method in detail. The experimental results obtained are shown in section III.

[1]F. Veiga is with the Institute for Systems and Robotics, Instituto Superior Tecnico, 1049-001 Lisboa, Portugal fveiga at isr.ist.utl.pt

[2]A. Bernardino is with the Institute for Systems and Robotics, Instituto Superior Tecnico, 1049-001 Lisboa, Portugal alex at isr.ist.utl.pt

II. Bayesian Optimization

One of the key points of our method is to define a suitable exploration strategy able to minimize the number of required exploration trials to achieve good grasps. To address this issue we will employ recent results in global sequential optimization using Bayesian methods [6]. The grasp quality metric will be maximized through an search strategy that proposes new trials on regions of the parameter space where either its expected value (exploitation) or its uncertainty (exploration) are large. The outline of the proposed method is as follows:

Let x denote a vector of grasp parameters that we want to optimize. Depending on the capabilities of the robotic manipulators these may include the hand-object relative pose, joint velocities, closure synergies, etc. An initial configuration x_0 is defined for the first robot trial. This can be defined in several ways: either randomly on a certain range depending on the object state or using heuristics from prior knowledge. Each robot grasp trial feeds a Gaussian Process Regression model [12] that models the expected value and variance of the quality metric. These measures can then be predicted easily at any point on the state space. To decide the next point to try, we maximize at each time step a form of Expected Improvement function (EI) with exploitation-vs-exploration control [13]. To optimize the expected improvement function we use the DIRECT algorithm [14]. This is a global optimization method that works by partitioning the space in intervals (DIRECT stands for DIvide RECTangles), estimating the Expected Improvement (EI) function in their center, and choosing, at each time step, the interval where EI can be maximal for any bound on the function derivative (Lipschitz constant). The obtained solution will define the parameters of the next robot trial. This cycle is depicted in Fig. 2.

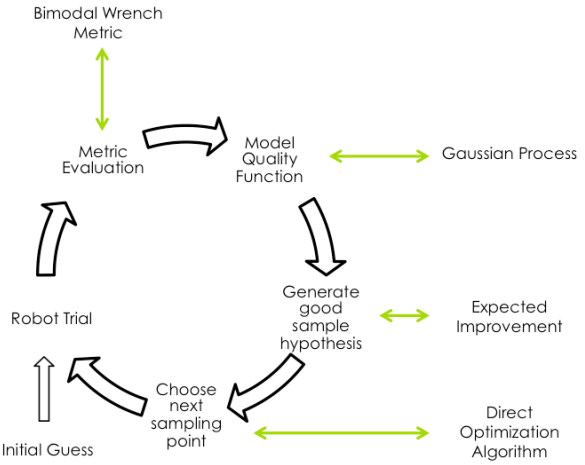

Fig. 2. Complete method diagram.

A. Gaussian Process Regression

A Gaussian process allows the information about the subject of interest to be represented in a way that the learning agent understands. It is used as a means to describe a distribution over functions. As a more formal definition, a Gaussian Process (*GP*) is a collection of random variables, any finite number of which have a joint Gaussian distribution, [12]. It is completely defined by a mean and covariance function pair

$$\mu(x) = \mathbb{E}[f(x)] \tag{1}$$

$$\Sigma(x,x') = \mathbb{E}\left[(f(x)-\mu(x))(f(x')-\mu(x'))\right] \tag{2}$$

and is represented as

$$f(x) \sim GP(\mu(x),\Sigma(x,x')). \tag{3}$$

In the current context the random values represent values from the grasp quality metric function $f(x)$ that the robot wishes to learn. The mean function is the best prediction of the true function given what is known. Also it is initially considered as a zero function since there is no relevant information at the start of the process, but other priors may be used if desired. The covariance function is what gives a sense of proximity or similarity between two points.

A kernel is the general name given to a function K of two arguments mapping a pair of inputs $x \in X$, $x' \in X$ into $\mathbb{R}$. In this work the kernel chosen is a Matérn class covariance function given by

$$K(r) = (1+\frac{\sqrt{3}r}{l})exp(-\frac{\sqrt{3}r}{l}) \tag{4}$$

where

$$r = |x-x'| \tag{5}$$

measures the distance between points x and $x'$and l represents the kernel length. Resorting to kernel functions allows us to create covariance functions where the connectivity between points is directly related with the distance between them. For more information on kernel functions refer to [12].

The representation of a covariance function is what implies a distribution over functions. Our goal is to perform the estimation of $f(x)$ based on the already known function values. To do this it is a simple matter of conditioning the distribution over functions to what is already known

$$F_*|X_*,X,F \sim N(\mu(x),\Sigma(x,x')) \tag{6}$$

$$\mu(x) = K(X_*,X)K(X,X)^{-1}F \tag{7}$$

$$\Sigma(x,x') = K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \tag{8}$$

where X_* is the estimation point set, X is the observation (known) point set, F is the set of observed function values and F_* denotes the estimated function values.

The values $K(X_*,X)$, $K(X,X_*)$, $K(X,X)$ and $K(X_*,X_*)$ are the kernel evaluated between point pairs of the prediction and observation set.

Despite the fact that this is a fairly good estimation of the function it is still somewhat naive. The thought of getting perfect measurements experimentally is an extremely gullible approach. Therefore some observational error has to be taken into account when preforming the estimation. Assuming

additive independent identically distributed Gaussian noise ε with variance σ_n^2 leads to

$$F_*|X_*,X,F \sim N(\mu(x),\Sigma(x,x')) \tag{9}$$

$$\mu(x) = K(X_*,X)(K(X,X)+\sigma_n^2 I)^{-1}F \tag{10}$$

$$\Sigma(x,x') = K(X_*,X_*) - K(X_*,X)(K(X,X)+\sigma_n^2 I)^{-1}K(X,X_*) \tag{11}$$

where I is the identity matrix.

This estimation of the mean function provides us with the robust estimation of the real function that we need. It also provides a measure of the estimation's uncertainty, which will be crucial to the robot's learning.

B. Expected Improvement

With the Gaussian Process providing a model of the function we wish to learn, the next step is to decide what we should do to improve our knowledge of this function. This decision shouldn't be taken lightly since it is what defines the learning rate. To achieve this decision we will use the concept of Expected Improvement [6].

The Gaussian Process provides a global estimation of $f(x)$ based on what is known at the time. Since we are trying to find the maximum value of the function $f(x)$ the natural decision should be to explore regions of the function where a higher maximum is possible. From this idea we define the improvement function as

$$I(x) = \max\{0, f_{n+1}(x) - f^{max}\} \tag{12}$$

where f^{max} is the current maximum value. The function takes on positive values when the prediction is higher than the best value found so far and is set to zero otherwise. Using the improvement function, the new observation point is attained by finding the maximum expected improvement point

$$x = \arg\max_x \mathbb{E}(\max\{0, f_{n+1}(x) - f^{max}\}|\mathscr{D}_n) \tag{13}$$

where $\mathscr{D}_n$ is all that is known at time n

$$\mathscr{D}_n = [f_1, \ldots, f_n] \tag{14}$$

and f_i is the observed value for trial i. This expected improvement can easily be evaluated analytically, [15], through

$$EI(x) = \begin{cases} (\mu(x) - f^{max})\Phi(Z) + \sqrt{\Sigma}\phi(Z) & if\ \Sigma > 0 \\ 0 & if\ \Sigma = 0 \end{cases} \tag{15}$$

where

$$Z = \begin{cases} \frac{\mu(x)-f^{max}}{\sqrt{\Sigma}} & if\ \Sigma > 0 \\ 0 & if\ \Sigma = 0 \end{cases}, \tag{16}$$

Σ is the covariance function and $\phi(.)$ and $\Phi(.)$ respectively denote the probability density function and the cumulative distribution function of the standard Normal distribution. As mentioned in the previous section, the uncertainty measure given by the covariance function Σ plays a huge role on the decision of the next observation. It enables the balancing between exploiting and exploring. When exploring, we should focus on points where the prediction variance is large in order to minimize the global uncertainty. When exploiting one's focus should be the points where the predicted mean function is high so that a higher and more accurate value of the global maximum may be found.

Evaluating the expected improvement through (15) balances the exploration versus exploitation trade-off in an unruly fashion. A more generalized form of the $EI(.)$ has to be found in order to directly control this balance. Lizotte [13] suggests a $\xi \geq 0$ parameter obtaining

$$EI_\xi(x) = \begin{cases} (\mu(x) - (f^{max}+\xi))\Phi(Z_\xi) + \sqrt{\Sigma}\phi(Z_\xi) & if\ \Sigma > 0 \\ 0 & if\ \Sigma = 0 \end{cases} \tag{17}$$

where

$$Z_\xi = \begin{cases} \frac{\mu(x)-(f^{max}+\xi)}{\sqrt{\Sigma}} & if\ \Sigma > 0 \\ 0 & if\ \Sigma = 0 \end{cases}. \tag{18}$$

In this work we use $\xi = \frac{\hat{\sigma}_f^2}{100}$ where $\hat{\sigma}_f^2$ is the Gaussian process estimated variance given by

$$\hat{\sigma}_f^2 = F^T K(X,X)^{-1} F. \tag{19}$$

For a more detailed reading on this topic refer to [6] and [13].

Now we have a procedure to represent the current knowledge, to predict the unknown and to accordingly decide what to do next. Repeating this process will ultimately lead to learning $f(x)$ very efficiently in terms of minimizing the number of observations. In the next section we will show that one more aspect must be considered for the method to be efficient.

C. Direct Optimization Algorithm

To optimize the Expected Improvement function one needs to evaluate it at several points. Kernels must be evaluated at all point pairs possible between the points in the estimated and observation point sets. While the kernel evaluation is simple, the number of points in the estimated point set grows at troubling rates if uniformly sampled through input space. This point set grows at a rate of n^D where n is the number of samples per dimension and D is the number of input parameters. So a simple uniform sampling of the space along all the dimensions in order to find the point with the highest expected improvement is not a good approach.

We turn to a more efficient approach trough the use of the Direct Optimization algorithm, [14]. This algorithm uses a small number of initial predictions to decide how to DIvide the feasible space into smaller RECTangles. The end result is a high discretization of the target function near the function maxima and a low discretization elsewhere.

The Direct algorithm starts by normalizing the function domain into a unit hyper-cube with center c_1

$$\bar{\Omega} = \{x \in \mathbb{R}^N : 0 \leq x \leq 1\}. \tag{20}$$

The algorithm works in this normalized space, only reverting to the original space when making function calls. After evaluating the function at $f(c_1)$ it is time to make the first division of the hyper-cube. The cube is divided into smaller cubes centered at $c_1 \pm \delta e_i$, $i = 1, ..., n$ where δ is one third of the cube length and e_i is the ith unit vector. Direct choses to

leave the best function values in the largest space. As such the first dimension to be divided is chosen by means of

$$\omega_i = \min(f(c_i + \delta e_i), f(c_i - \delta e_i)), \quad 1 \leq i \leq N. \tag{21}$$

The dimension with the smallest ω_i is divided into thirds and the process is repeated for all dimensions on the resulting center hyper-rectangles.

With the hyper-cube division done it is time to find which of the newly generated rectangles/cubes may be potentially optimal. For the optimality test, we test each of the rectangles/cubes for the existence of a Lipschitz Constant $\hat{K} > 0$ that allows

$$f(c_j) - \hat{K} d_j \leq f(c_i) - \hat{K} d_i, \forall i, \tag{22}$$

$$f(c_j) - \hat{K} d_j \leq f_{min} - \varepsilon |f_{min}| \tag{23}$$

where $\varepsilon > 0$ is a positive constant, f_{min} is the current best function value and c_j and d_j are respectively the center of the tested hyper-rectangle/cube and a measure of the dimension of the same rectangle/cube. In (22) we test if the possible variation of the value $f(c_j)$ when traveling inside the respective hyper-rectangle/cube may reach a minimum value when applying the same variation scale to all other $f(c_i)$ navigating on the respective ith rectangle/cube. On the other hand (23) tests if the possible minimum reached on the jth rectangle/cube is of interest when compared with f_{min}. The term $\varepsilon |f_{min}|$ makes sure that the improvement to the minimum value is non trivial.

Now that we know S, the set of all the potentially optimal rectangles/cubes, the only remaining step is to divide all members of this set, evaluate the function at the center of the resulting rectangles/cubes and update f_{min}. An interesting fact is that when dividing an hyper-rectangle, the algorithm always chooses to divide along the longest dimension(s) to ensure that the rectangles shrink on every dimension.

The Direct algorithm repeats the previous operations (except the initialization) until S is empty, meaning no more divisions are of interest. When this stage is reached the f_{min} is the global function minimum. Since we are looking to maximize the expected improvement function, it is only a matter of suppling Direct with negative values of $EI(.)$.

III. Experimental Results

A. Experimental Setup

The experiments performed during the course of this work were done in a simulation environment through the OpenRave simulator [16]. The environment consisted of a manipulator arm, the Barrett hand, and a few objects as shown in Figures 3 and 4.

B. 1D exploration with Bayesian Optimization

We will now show some 1D scans made by changing the grasp's initial position along the approach axis inside a bounded region near the object. The grasp parameter x to optimize in this case is simply the distance of the hand to the object. The goal of these scans is to give a better understanding of how the Gaussian process and the Bayesian

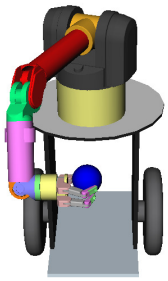

Fig. 3. The experimental setup used for validating the proposed metric.

Fig. 4. The objects.

methods interact. Also to be shown is how the system behaves when the initial random sample is one of the best possible or one of the worst possible. Fig. 5 depicts the latter. It represents the sequence carried out by the method to sample a sphere, when the initial random sample does not even touch the object (top left plot). In red we can see the *GP* mean function which is the best estimation of the metric function at this time, the dashed lines depict the estimation variance at each function point, the red dots are the values collected from the robot trials. The blue function represents the *EI* function that classifies the function space in terms of exploration interest. Even with the setback caused by the bad initial random sample, it only takes 2 iterations of the method in order to find a possible maximum (top right plot). Acknowledging the fact that a region that may contain the maximum has been found, the system focuses the search in it's neighboring points (bottom left plot). Once this region has been exploited, the system resumes it's exploration efforts to assure there are no more regions that may contain better values of $f(x)$. After confirming that it has found the function maximum value the exploration stops (bottom right plot).

The second case that will be shown is the opposite case. Fig. 6 shows the sequence followed by the method while sampling a star prism when the initial random sample is one possible maximum (top left plot). As the function is still completely unknown to the system, excluding the random sample, it is impossible for it to realize that the first sample is actually a possible maximum and so it proceeds with the exploration. After 3 iterations the system finally realizes the potential of the first sample (top right plot) examining the neighboring region (bottom left plot). The exploration is resumed and 2 more interest areas are found before the systems stops (bottom right plot).

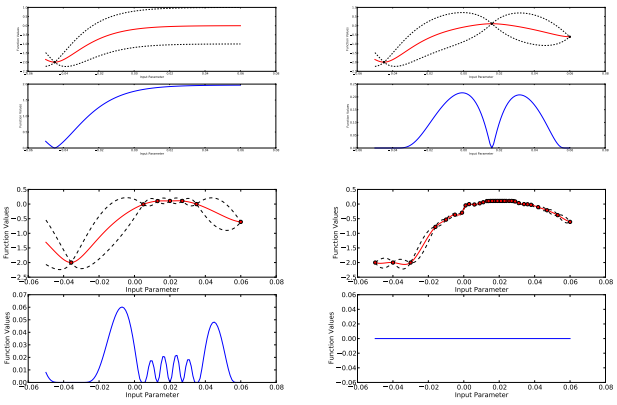

Fig. 5. 1D scan of a sphere, computed by sampling along the input parameter.

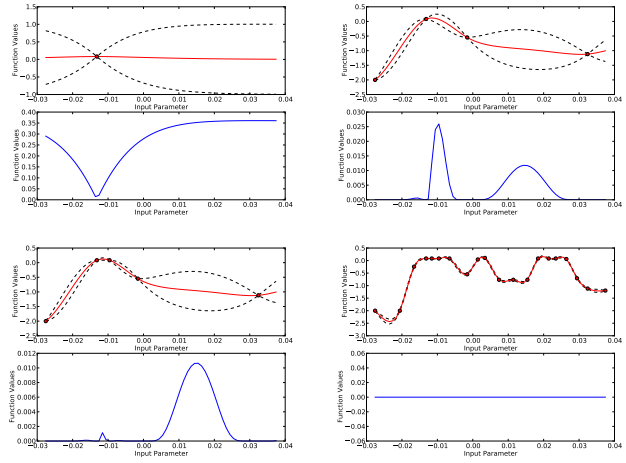

Fig. 6. 1D scan of a star prism, computed by sampling along the input parameter.

C. 2D exploration with Bayesian Optimization

Let's now scan the object in the 2-dimensional space. The results are shown in figures 7 to 9. The parameters to optimize are now the distance to the object (x) and the vertical offset with respect to the object's center (y). Also in order to test the method's performance under the worst possible conditions, we assume that the observations are noisy.

As shown in table I, it is clear that the number of necessary trials is different depending on the object. Also depicted in table I are the number of trials needed in order to find a value that differs from the global maximum by less than 5%, 10% and 20% respectively from left to right.

Although only stoping when the expected improvement no longer displays interesting values, the method could have settled for much less samples and still provide a fairly good approximation of the global maximum.

D. Bayesian optimization versus random sampling

We now make a base comparison between the performance of the Bayesian optimization method and random sampling

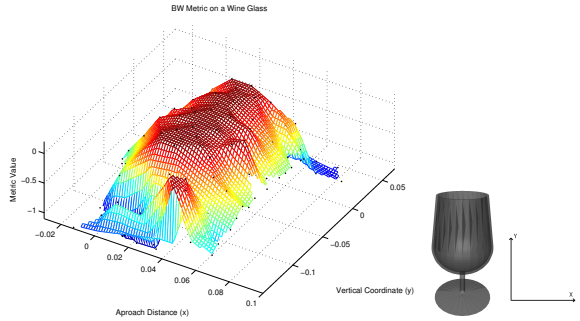

Fig. 7. *BW* Metric values of grasps in a wine glass, computed using the Bayesian Optimization method.

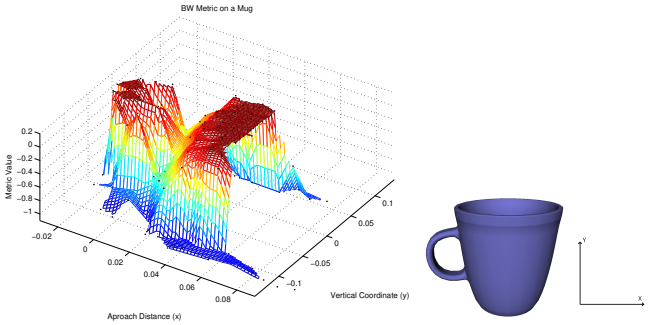

Fig. 8. *BW* Metric values of grasps in a mug, computed using the Bayesian Optimization method.

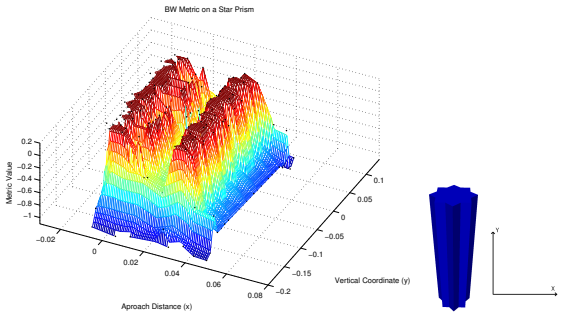

Fig. 9. *BW* Metric values of grasps in a star prism, computed using the Bayesian Optimization method.

TABLE I

NUMBER OF SAMPLES TAKEN FOR EACH OBJECT.

Object	**Bayesian Optimisation**			
-	Total	5%	10%	20%
Sphere	551	14	14	14
Wine Glass	306	31	19	19
Cylinder	485	39	20	20
Mug	331	55	55	1
Cuboid	564	1	1	1
Rotated Cuboid	388	59	59	59
Star Prism	402	105	105	31

on parameter space. This comparison is done by measuring the evolution of the best value found by each algorithm along a sequence with 60 iterations when performing 2D exploration.

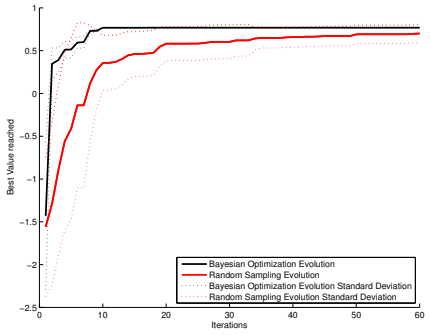

Fig. 10. Comparing the evolution of best value found when sampling a sphere with the Bayesian aproach versus the random sampling method.

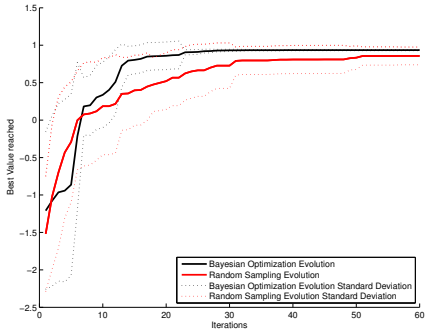

Fig. 11. Comparing the evolution of best value found when sampling a mug with the Bayesian aproach versus the random sampling method.

Figures 10 to 12 represent the results of some of these experiments. They show the evolution of the best value found by each method when sampling a sphere, a mug and a star prism respectively. The Bayesian method (black) clearly converges faster than the random sampling method (red) to the maximum function value even for a complicated objects such as the mug (non-convex). The same figures also show that the Bayesian method displays a very small variance when compared to the random search method.

IV. Conclusions

This work proposed a methodology to compute optimal or close-to-optimal grasps on a diversity of scenarios. We have shown that it is possible to create a robotic grasping system that "feels" the object and gradually searches for the optimal grasp, through the use of Bayesian Optimization methods. We showed that these methods can be more efficient than random search of the parameter space. We also showed that they provide a measure of the system's global uncertainty. This measure can be interpreted as how much do we actually understand the complete system.

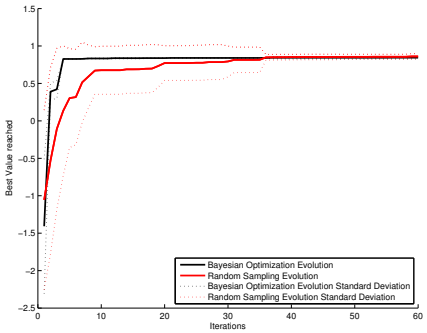

Fig. 12. Comparing the evolution of best value found when sampling a star prism with the Bayesian aproach versus the random sampling method.

Acknowledgments

This research was partially funded by the EU Commission within the Seventh Framework Programme FP7, under grant agreement 248258 (First-MM) and grant agreement 215843 (Poeticon++).

References

[1] A. Miller, S. Knoop, H. Christensen, and P. Allen, "Automatic grasp planning using shape primitives," in *Proc. of ICRA 2003*, 2003.

[2] A. Morales, T. Asfour, P. Azad, S. Knoop, and R. Dillmann, "Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands," in *Proc. of IROS 2006*, 2006.

[3] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *Proc. of IROS 2007*, 2007.

[4] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Simultaneous grasp and motion planning: humanoid robot ARMAR-III," *IEEE Robotics and Automation Magazine*, vol. 19, no. 2, pp. 43–57, 2012.

[5] O. B. Kroemer, R. Detry, J. Piater, and J. Peters, "Combining active learning and reactive control for robot grasping," *Robot. Auton. Syst.*, vol. 58, no. 9, pp. 1105–1116, 2010.

[6] E. Brochu, V. Cora, and N. D. Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions , with Application to Active User Modeling and Hierarchical Reinforcement Learning," *Arxiv preprint arXiv:1012.2599*, 2010. [Online]. Available: http://arxiv.org/abs/1012.2599

[7] D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans, "Automatic gait optimization with gaussian process regression," *Proc. of IJCAI*, 2007.

[8] M. Frean and P. Boyle, "Using Gaussian processes to optimize expensive functions," *AI 2008: Advances in Artificial Intelligence*, 2008.

[9] R. Martinez-cantin, D. F. Nando, 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 . . .*, no. 2005, 2009.

[10] Z. L. R. Murray and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC Press, 1994.

[11] M. a. Roa and R. Suárez, "Finding locally optimum force-closure grasps," *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 3, pp. 536–544, Jun. 2009.

[12] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[13] D. Lizotte, "Practical bayesian optimization," university of Alberta, Edmonton, Alberta, Canada, Tech. Rep. PhD Thesis, November 2008.

[14] D. E. Finkel, "DIRECT Optimization Algorithm User Guide," 2003.

[15] D. Jones, M. Schonlau, and W. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, pp. 455–492, 1998.

[16] R. Diankov, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep.*, no. July, 2008.