Comparison of Extremum Seeking Control Algorithms for Robotic Applications

Berk Calli, Wouter Caarls, Pieter Jonker, Martijn Wisse

Abstract— **The purpose of this paper is to help engineers and researches to choose among the extremum seeking control (ESC) techniques for robotic applications such as object grasping, active object recognition and viewpoint optimization. These techniques are categorized into five main groups: Sliding mode ESC, neural network ESC, approximation based ESC, perturbation based ESC and adaptive ESC. These groups are explained briefly by stressing their working principles and the effect of the parameters. Then, the techniques are compared with respect to their robustness to noise and system dynamics by simulations. In conclusion, we propose the usage of the approximation based methods when the noise level is negligible. When noise is present, the neural network based optimizers are a better choice thanks to their hysteresis functions. However, if the system has both high noise and dynamic effects, then the perturbation based method is preferable since large motions provide robustness to noise and smooth references generated by the algorithm are less likely to cause instability. An application example is also given on texture density maximization.**

I. Introduction

Extremum seeking control (ESC) is a well developed field which addresses the problem of objective value optimization when the objective function, its gradient and optimum value are unknown. Some widely known applications of ESC are ignition time selection for combustion engines [1], bioreactor optimization [2] and anti-lock braking system control [3]. A large list of applications can be found in [4].

Besides these applications, ESC algorithms have great potential in robotics. Especially for robots that operate in unknown/dynamic environments (i.e. homes, offices, elderly care centers), the optimum values of the objective variables of a task are unknown because of the unknown state/effect of the environment. In these cases, online optimization tools like ESC can be utilized in order to optimize these task variables.

A very recent example of ESC in the robotics field is proposed in [5]. In this work, an eye in hand system is used and the saliency (a measure of interest) of the view of the camera is maximized by providing references to the robot arm. Here, since the saliency map of the environment is unknown, the objective function cannot be used for the optimization purposes. However, the objective value can be calculated for each view. The ESC framework enables online optimization for such a task. Another example is presented in [6] and [7] where ESC is implemented for non-holonomic systems in order to seek the source of a signal.

Our interest in ESC algorithms stems from our vision based grasping research [8]. One challenge of this field is to develop algorithms that can calculate the positions of stable grasping points on the object. This can be achieved by constructing an objective function based on the object model and running an optimization algorithm which searches for the optimum grasping points. However, we are specifically working on grasping unknown objects, which automatically implies that the model of the object is unknown. On the other hand, using an eye in hand system, an objective value can be calculated for each image of the object. In this case, an ESC algorithm can be utilized in order to run an online optimization for grasping purposes, so that the object modeling process can be skipped and a faster grasping can be achieved. With a very similar formulation, these algorithms can also be used to increase the performance of the object recognition systems by altering the viewpoint of the object in order to see surfaces with high texture density.

In this paper, we analyze the ESC algorithms for robotics applications. The robotic systems can be characterized by their nonlinear dynamics and sensor noise. The algorithm should be able to perform well under these circumstances. The performance of the algorithms are analyzed by their transient response using the rise time and settling time measures. The smoothness of the velocity references generated by the algorithms is also another criterion. Moreover, since most robots can operate in multiple dimensions, the performance of the algorithm in the multivariate case is also crucial. For this case, the trajectories generated by the algorithms are presented in order to analyze the motion in the search space, and a numerical comparison is given by presenting the total distance traveled until 95% of the optimum value is reached. In most robotic applications, the objective value can be measured continuously. For example, in the implementation of [5], the saliency value can be computed for each image acquired from the camera. Therefore, in this paper we specifically concentrate on *analog* ESC algorithms, which utilize the objective value continuously, and exclude algorithms like the simplex method [9].

Two other comparison papers on ESC can be found in [10] and [11]. These papers present a detailed comparisons of two ESC methods whereas we focus on robotic applications and present a larger set. Also, a good history study of the ESC can be found in [12].

The analog extremum seeking control algorithms can be categorized into five groups: Sliding mode ESC, neural network ESC, approximation based ESC, perturbation based ESC, adaptive ESC. The next section summarizes these ESC methods. The third section gives comparative simulation results. The fourth section presents an analysis based on the simulation results. In the fifth section, experimental results on texture density maximization is presented. Finally, in section six the paper is concluded with a summary.

II. Analog Extremum Seeking Control Methods

The problem of ESC is formalized as follows: For a given system

$$y = f(x) \tag{1}$$

where $f(x)$ is the unknown objective function, $x \in R^n$ is the state vector with dimension n, and y is the objective value that can be measured continuously, an analog extremum seeking control algorithm searches the state space for the state vector x that corresponds to the optimum (most of the time local optimum) of the objective value. To achieve this goal, the following algorithms are presented in the literature (all the derivations below are for the minimization of the objective value):

A. Sliding mode ESC

In early 70's, Korovin and Utkin proposed sliding mode ESC [13], [14]. This method is based on a driving signal. Unlike conventional control problems, the reference value is unknown in the ESC framework since the optimum value of the objective function is unknown. The driving signal in sliding mode ESC functions as a reference generator for the system, and it is designed to be monotonically decreasing when searching for the local minimum. The ESC rule is then formulated such that the system tracks this driving signal.

As in the most sliding mode systems, the tracking of the signal relies on the high frequency chattering around the signal's value. This behavior makes this ESC algorithm unsuitable for robotic applications. Preserving the main idea, Yu and Ozguner [15] altered this method in order to prevent the high frequency chattering. This type of sliding mode ESC is analyzed in this paper.

Define a monotonically decreasing driving signal g(t) whose derivative is given as

$$\dot{g}(t) = -\rho \tag{2}$$

where ρ is a positive constant. The error between objective value and the driving signal is

$$e = y - g(t) \tag{3}$$

The output of the ESC algorithm is designed as

$$u = sign(sin(\pi e/\alpha)) \tag{4}$$

where α is a positive constant. This output is fed to the system as velocity reference with a gain k which determines the convergence rate:

$$\dot{x}_r = ku \tag{5}$$

The reason of using a sine inside signum function is the following: since the objective function is unknown, the ESC algorithm does not know the direction in which it should lead the system in order to minimize the objective value. If the system goes in a wrong direction, the error will increase. This rise will continue until the value of the sine function changes sign. Then, the velocity reference will also change direction, and the system will start following the driving input. The bound on the error that is allowed by the system can be tuned by the parameter α. If the system is slower or faster than the driving signal, then this will cause chattering and the frequency of this chattering can be remarkably decreased by increasing the α parameter. When the optimum value is reached, the system oscillates within the allowed error bound. This method is implemented for an anti-lock braking system in [15].

Multivariate Extension: Although it is quite straightforward to implement sliding mode ESC in the unidimensional case, the multidimensional implementation is problematic. For the algorithm in [13], there are some propositions on multivariate extensions, but to the best of our knowledge there is no successful implementation of sliding mode ESC in the multivariate case. Thus, in Section III we will only analyze the performance of the sliding mode algorithm for a single dimension.

B. Neural network ESC

Neural network ESC [16], [17] is based on a minimum peak detector and two switching functions with hysteresis. Just like the driving signal concept in sliding mode ESC, neural network ESC also has a reference generator. One component of this reference is the minimum peak detector, and it is designed to be monotonically decreasing as follows:

$$\dot{y}_p = \begin{cases} 0 & (y_p \le y) \\ -M & (y_p > y) \end{cases} \tag{6}$$

where y_p is the value of the minimum peak detector, and M is a parameter that defines the speed of convergence of y_p to y, when y is smaller. This speed should be higher than the rate of change of the objective value in order to maintain convergence:

$$|\dot{y}| < M \tag{7}$$

The second component of the reference is provided by the switching function W. This switch starts functioning after the optimum value is reached, and it will be explained shortly. The reference is formed by the addition of the two components:

$$y_r = y_p + y_w \tag{8}$$

The error function is defined as the difference between the objective value and the reference value:

$$e = y_r - y \tag{9}$$

The control law u is another switching function with hysteresis and is designed as:

$$u = \begin{cases} -A & e < -\delta \\ A & e > \delta \\ [previous\_state] & \text{otherwise} \end{cases} \tag{10}$$

Here, A is a constant that specifies the magnitude of the velocity reference, and δ is the hysteresis width. If the direction of the system makes the objective value converge to the optimum value, then this direction is kept. Otherwise, the error will increase, and when it is greater than the hysteresis, the direction will be reversed. However, this function can not

stop the system when the optimum value is reached. The role of the y_w signal begins here and it alters the reference value. The derivative of the y_w signal is defined as

$$\dot{y}_w = \begin{cases} 0 & e > \Delta \\ B & e < -\Delta \\ [previous\_state] & \text{otherwise} \end{cases} \tag{11}$$

where B is a positive constant. The value of the second hysteresis width Δ should be greater than δ, since y_w should start functioning after the optimum value is reached:

$$\Delta > \delta \tag{12}$$

The control signal u is fed to the systems as velocity input:

$$\dot{x}_r = u \tag{13}$$

When the optimum value is reached, the system oscillates with an error under Δ.

Multivariate Extension: In order to extend the neural network based algorithm to the multivariate case, the usage of a coordinate descent algorithm is proposed [17]. The main logic of this algorithm is to descritize the search directions and to minimize in one direction at a time. To achieve this, the control law u is implemented with $2n-1$ switching functions with different hysteresis values where n is the dimensionality of the state. The direction of the search is changed whenever a hysteresis threshold is reached in that direction and this brings the following constraint to the hysteresis widths:

$$\delta_1 < \delta_2 < ... < \delta_{2n-1} < \Delta \tag{14}$$

C. Approximation based ESC

When the objective function to be optimized is unknown, one natural choice is to derive a local representation of this function based on the past data. This representation can then be used with either a gradient or a non-gradient based approach. For feedback linearizable systems, a trust region based algorithm is designed in [18]. The main idea of the trust region algorithm is the following: First a well-poised approximation set is formed, and a local approximation of the objective function $\hat{f}(x)$ is obtained. Since this is only a local approximation, it can only be trusted in a local region. An initial value Δ_0 is set for this trust region, and the approximated objective function is optimized within this region. Let's say the system is at x_k, and the optimum value in this local region is found at $x_k + p_k$. The system is led to this new set point and whenever the set point is reached the following ratio is calculated in order to measure how good the approximation of the objective function was:

$$\rho_k = \frac{f(x_k) - f(x_k + p)}{\hat{f}(x_k) - \hat{f}(x_k + p)} \tag{15}$$

If ρ_k is close to 1, this signifies a good approximation, and trust region radius can be increased, otherwise it should be decreased. This update rule is as follows:

$$\Delta_{k+1} = \begin{cases} [\Delta_k, \infty], & \rho_k \geq \eta_2 \\ [\gamma_2 \Delta_k, \Delta_k] & \rho_k \in [\eta_1, \eta_2) \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k], & \rho_k < \eta_1 \end{cases} \tag{16}$$

where $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \gamma_1 \leq \gamma_2 \leq 1$. For more detailed information about the trust region methods, see [19].

Considering [18], the advantage of using feedback linearizable systems is to be able to ensure the convergence of the system to the set points and be able to estimate the regulation time, so that the time when $x_k + p_k$ is reached is known approximately. Since we don't want to impose any constraints on the system in this paper, we used the error between $x_k + p_k$ and x in order to detect when the reference is reached.

Alternatively, since the gradient of the objective function can be estimated locally by the derivative of the approximated objective function, a line search algorithm such as gradient descent can also be utilized as in [4]. In this case, the velocity reference can be generated as follows:

$$\dot{x}_r = -k \frac{d\hat{f}(x(k))}{dx} \tag{17}$$

where k is a positive gain. The simulation results are presented for both methods in Section III.

Multivariate Extension: For the multivariate case, multivariate approximation techniques can be used to approximate the objective function locally. We used bilinear approximation in this paper for two dimensions. The rest of the derivation is the same as the unidimensional case.

D. Perturbation based ESC

It would be safe to say that perturbation based ESC framework is the most popular method in the literature. These types of methods [20], [21], [22], [23], [24] use an external perturbation signal and modulation theory to find the optimum value. The most commonly used perturbation signal is the sine wave. The ESC algorithm sends the sine wave to the system as position reference together with an adaptation input:

$$x = a \cdot sin(\omega t) + \hat{x} \tag{18}$$

with some amplitude a and modulation frequency ω. The adaptation signal $\hat{x}$ shifts the sine wave towards the gradient direction. A way of calculating this signal is the following procedure: Combining the adaptation signal with the sine signal as in (18) is the modulation phase of the algorithm. The response of the system to this signal is measured in the objective value y. This output is filtered by a high pass filter to eliminate the DC component and demodulated by the same sine signal to extract the gradient direction.

$$\xi = y \left(\frac{s}{s+h} \right) (a \cdot sin(\omega t)) \tag{19}$$

This information is utilized in order to calculate the shift in the sine signal towards the gradient. This part of the algorithm is called the adaptation law.

$$\hat{x} = -\xi \frac{k}{s} \tag{20}$$

Here, k is a positive constant that specifies the adaptation speed. The algorithm has several variations. Some of them include applying a low pass filter to ξ before using it in the update rule since only the DC component of the demodulated signal is needed for gradient calculation. In some other variations, the adaptation law is combined with a compensator which relaxes the design constraints on the parameter k [25].

The amplitude and the frequency of the sine wave signal and the cutoff frequencies of the filters are important design parameters. A detailed study of how the design of the perturbation signal affects the performance of the ESC system is presented in [26]. Further analysis with variable gains and with the presence of noise in the measurement channel is presented in [27]. The applications of perturbation based ESC on a bioprocess, fluid flow control, magnetically suspended flywheel and online parameter tuning for combustion timing can be found in [28], [29], [30] and [1] respectively. Also, an application for non-holonomic systems is given in [31] where the algorithm is enhanced for being able to alter the trajectory of the robot.

Multidimensional Extension: Extending this method to the multidimensional case is done by assigning a sine wave to every input channel with some phase shifts. For example, in two dimensional case, if the phase shift between the sine waves of the first channel and the second channel is $\pi/2$, then the system moves with circles in the task space, and the center of the circle is shifted by the adaptation input which is calculated by the same modulation and demodulation steps.

E. Adaptive ESC

Adaptive ESC [32], [33] uses adaptive control schemes for the online optimization. Unlike the approximation based methods, the adaptive ESC approximates the objective function globally. This algorithm necessitates the type of the objective function to be known. The adaptive ESC implementations for Hammerstein and Wiener type nonlinear objective functions are given in [33] and [34]. In this paper, we do not want to impose any constraints on the objective function, so we will not analyze this method. However, we find it important to note that, if the form of the objective function is known, then this method provides efficient results by quickly identifying the extremum, and driving the system towards it.

III. Simulations

The performance of the above mentioned ESC algorithms are analyzed by simulations in both unidimensional and multidimensional case. All the Matlab files of these simulations can be found at [1].

[1]http://www.dbl.tudelft.nl/over-de-faculteit/afdelingen/biomechanical-engineering/onderzoek/dbl-delft-biorobotics-lab/software/

A. Unidimensional Simulations:

In this subsection, the robustness and general characteristics of the ESC algorithms are examined for the unidimensional case. The simulations are conducted for the following cases:

1) System without dynamics and noise
2) System with noise without dynamics
3) System with dynamics without noise

For the third case, the algorithm is analyzed by using a model with the dynamics of a six DOF robot manipulator provided by a Matlab robotics toolbox [35]. The optimization problem is defined in the task space and the velocity references that are supplied by the ESC algorithms are sent to an inner velocity control loop.

The objective function is specified as a Gaussian based function where μ and σ signify the mean and the variance respectively:

$$f(x) = 2 - e^{\frac{-1}{2}\frac{(x-\mu)^2}{\sigma^2}} \tag{21}$$

Here, the parameters μ and σ are set to -0.4 and 0.27. The system starts at $x = 0$, and the objective value measured at that state is $y = 2$. The ESC algorithms search for the minimum value of this function which is $y = 1$.

The maximum velocity of the systems are limited to $1m/s$. The algorithms are tuned for each separate case considering fast convergence, low chattering, and the steady state error is aimed to be kept below %5. For all the algorithms, objective value is plotted with respect to time. These plots are evaluated using rise time and settling time measures. Also, the references generated by these algorithms are analyzed considering smoothness.

1) Simulations without noise and dynamics: Results for the simulations without noise and dynamics are presented in Figure 1. It can be seen that the *sliding mode ESC* and the *neural network ESC* performs very similarly. They both lead the system to the wrong direction for a brief amount of time. Then, due to the increasing error between the driving signal and the measurement, both algorithms change the sign of the velocity reference by the mechanisms explained in Section II, and the system starts to approach to the objective value with a constant velocity. The rise time and the settling time of the sliding mode ESC are ≈ 0.31 and ≈ 0.44 seconds. These values are ≈ 0.31 and ≈ 0.37 seconds for the neural network controller. When the optimum value is reached, the systems oscillate around the optimum value.

Among the *approximation based ESC* methods, the results of the trust region and gradient based methods are given with the dashed red and blue lines in Figure 1 respectively. For the trust region algorithm, each peak at the generated velocity corresponds to a new set point. It can be seen that the thrust region width first increases at the almost linear region of the Gaussian function since the local approximations hold for larger regions. However, near the optimum value, the shape of the Gaussian function changes more rapidly which causes a decrease of the width. The objective value is reached with

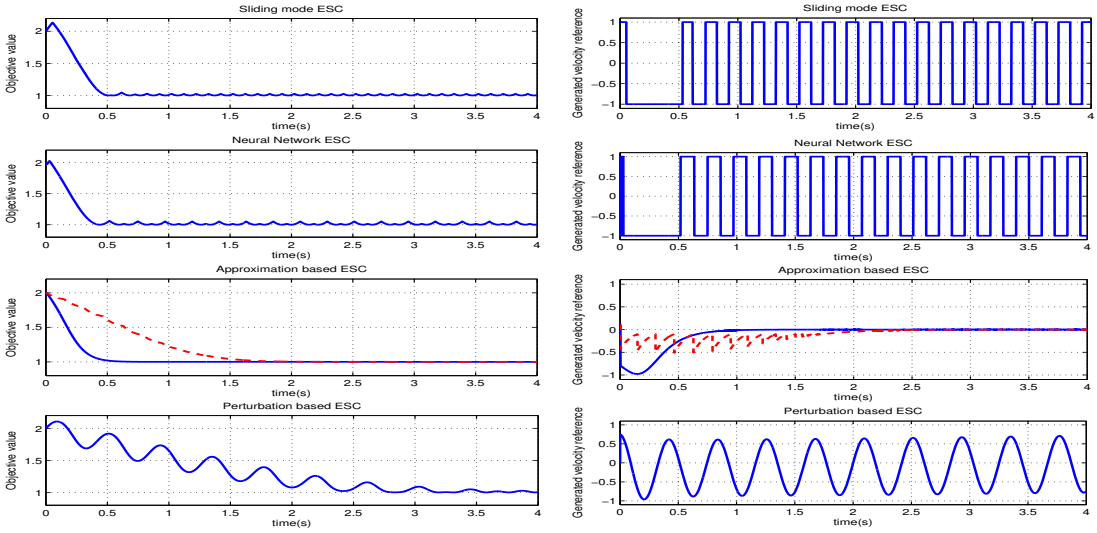

Fig. 1. Simulation results of the analog ESC methods without noise and dynamics in unidimensional case.

a rise time of ≈ 1.39 seconds, a settling time of ≈ 1.46 seconds.

On the other hand, the gradient based method generates a very smooth velocity reference. This feature is clearly better than the previous algorithms. The rise time is measured as ≈ 0.4 seconds, and the settling time is ≈ 0.42 seconds. It is clear that the performance of the gradient based method is better than the trust region algorithm considering both the speed of convergence and reference smoothness. Thus, for the rest of this section, only the results with gradient based method will be presented.

The *perturbation based ESC* algorithm provides sinusoidal velocity references to the system. The system converges to the optimum value while oscillating at the same time with smooth references. When the optimum value is reached, the amplitude of the oscillations decreases gradually. This algorithm necessitates a larger motion and this makes rise time and settling time increase drastically. These values are measured as 2.15 and 2.35 seconds respectively.

2) Simulations with noise without dynamics: The simulations are run for three different measurement noise amplitudes: 0.05, 0.1 and 0.2. The noise is uniformly distributed. These results are presented in Figure 2. For the *sliding mode ESC*, an oscillation is observed in the transient response. The rise time and settling time are increased to ≈ 0.58 and ≈ 0.88 respectively. However, more importantly, high frequency oscillations are observed at the direction change points. The amount of oscillations is increased as the noise amplitude gets higher. These oscillations are prevented by the hysteresis mechanisms of the *neural network ESC*. For this method only small irregularities are observed at the steady state oscillations. Other than that, its performance is very close to the case without noise.

For the *approximation based ESC*, more data points are used for interpolation than in the noise-free case in order to cope with noise. The motion until $t = 0.5$ is for collecting these necessary data. This motion caused an increase in the settling time which is measured as ≈ 1.08. Additionally, although the objective value converges smoothly to the optimum value, the smoothness of the velocity reference is effected negatively. This causes the system to diverge from the optimum value, and it gets worse when the noise is increased.

The *perturbation based ESC* stays unaffected for all three noise levels. It provides the same performance as the noise-free case.

3) Simulations with dynamics without noise: The results with dynamics are given in Figure 3. For the *sliding mode ESC*, the convergence to the optimum value is maintained with an oscillation in the transient. The rise time and settling time values are measured as ≈ 0.65 and ≈ 0.88 seconds respectively. There is also a risk of drift in the steady state which is discussed in Section IV.

For the *neural network ESC* method, higher amplitude oscillations are observed in the steady state. The reason is that, because of the dynamics of the system, the system does not stop immediately after the first threshold is exceeded which causes the system to exceed the second threshold as well. The rise time and settling time values are again very close to the sliding mode ESC; the values are ≈ 0.57 and ≈ 0.8 seconds respectively

Approximation based ESC is quite robust to the robot dynamics. Since the approximation phase uses the current state values of the robot, it is not affected by the dynamics. The rise time and settling time values are smaller than the previous methods with values ≈ 0.52 and ≈ 0.58 seconds. The *perturbation based method* is again very robust to the robot dynamics and performs very close to the dynamics free cases.

B. Two dimensional Simulations:

In this section, the motion generated by the algorithms in the two dimensional case is analyzed. No noise or system

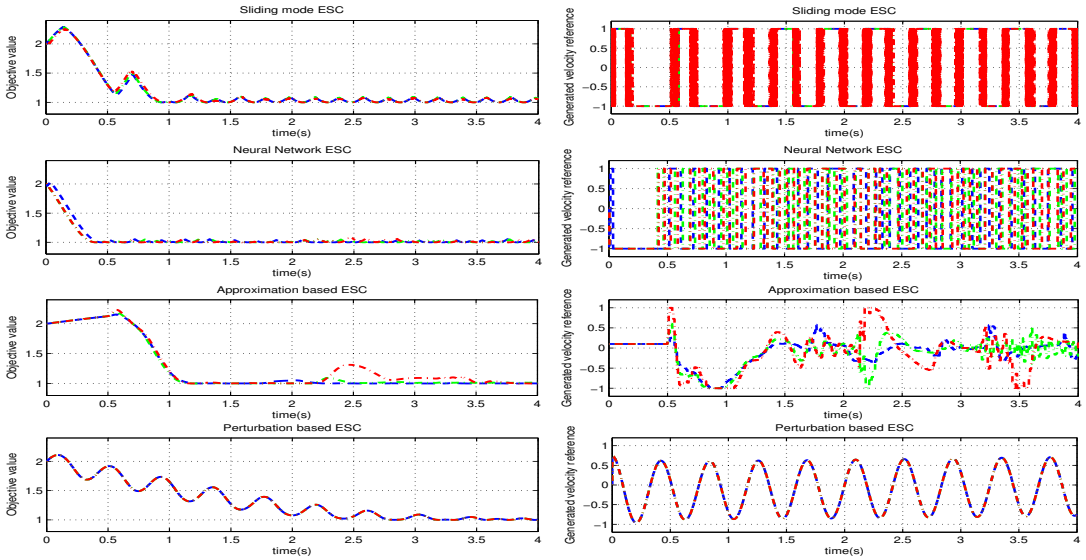

Fig. 2. The simulation results with three different noise levels. The blue, green are red dashed lines are the results with noise amplitudes 0.05, 0.1, and 0.2 respectively.

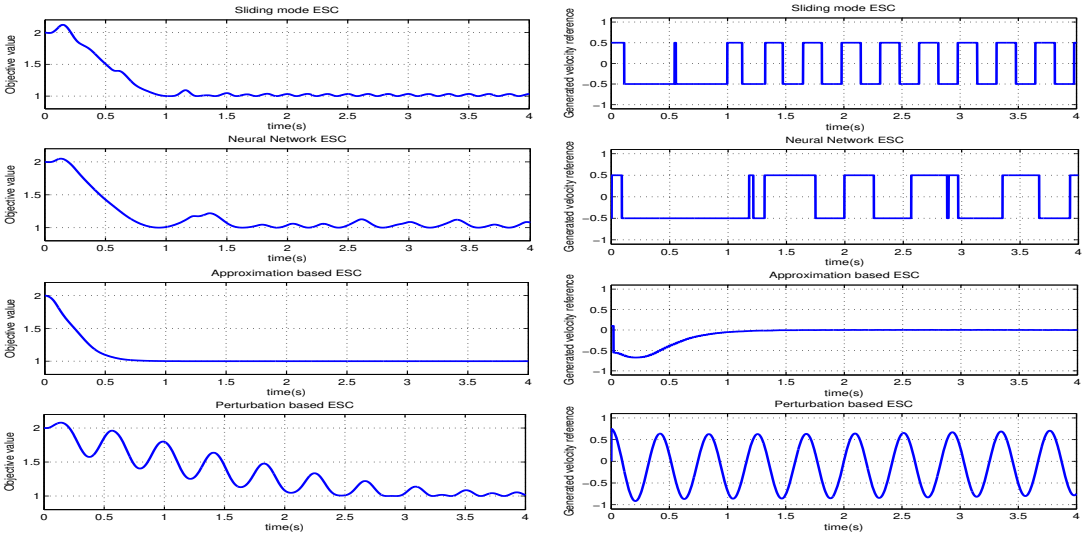

Fig. 3. The simulation results with a dynamic model of a six DOF robot.

dynamics are used (like in Subsection III-A.1) in order to examine the motion characteristics separately. The total distance traveled until 95% of the optimum value is reached is also given for numerical comparison. Since there is no extension of sliding mode ESC to the multivariate case, only the other algorithms are considered. The objective function is defined as a multivariate Gaussian based function:

$$y = 2 - e^{-0.5(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{22}$$

Here, the mean vector μ and the covariance matrix Σ are selected as $\begin{pmatrix} 0.5 \\ 0.2 \end{pmatrix}$ and $\begin{pmatrix} 0.15 & 0 \\ 0 & 0.15 \end{pmatrix}$ respectively.

The results are presented in Figure 4. For *neural network based ESC*, the descritization of the search space can be seen clearly. The total distance traveled is ≈ 0.81 meters. For *approximation based ESC*, it can be seen that the objective value is reached following an almost optimal route with a total distance of ≈ 0.41. The result for *perturbation based ESC* clearly shows the circular motion in task space with a traveled distance of ≈ 6.73 meters.

IV. Analysis

Effectively, sliding mode ESC and neural network ESC functions almost the same when there is no noise and dynamics effects. However, for sliding mode ESC, there should be a balance between $\dot{g}(t)$ and $\frac{dy}{dx}$ in order to avoid chattering. In neural network ESC, the driving input moves together with the objective value. This makes the parameter tuning of the neural network method easier. The approximation based method generates very smooth references for the system, and

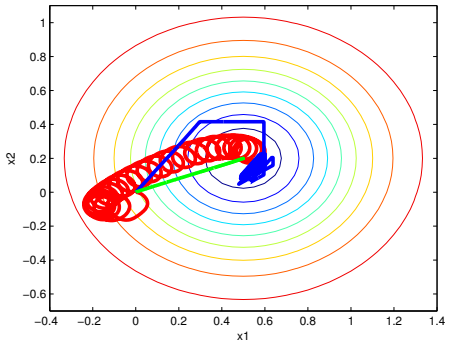

Fig. 4. The simulation results of neural network ESC (blue) the approximation based ESC (green), and the perturbation based ESC (red) for the two dimensional case: The algorithms lead the system to the minimum value of the Gaussian like function.

looks like the best choice for the systems without noise. It does not oscillate around the optimum value and has a very small steady state error. The perturbation based method looks like overkill for systems with negligible noise and dynamics. The rise time, settling time and the total distance results are all significantly larger than the other methods.

It is seen that the noise in the measurement channel affects all the methods except the perturbation based one, however the settling time of this method is still larger. The neural network ESC also shows robustness by its hysteresis mechanism except for the small steady state irregularity, and it shows the best performance in settling time. In the sliding mode based approach, the transient response is also affected, but high frequency oscillations when the velocity reference is changing are a greater problem. The performance of the approximation based method also drops due to the degradation of smoothness of the generated velocity reference, and higher settling time is observed.

It is seen that sliding mode ESC and neural network ESC show a performance decline due to the dynamics effects. However, for sliding mode ESC there is a potential risk of drift of the system in steady state. The reason of this is the growth of the driving signal. Actually, due to the design of the driving signal in (2), it increases all the time, even after the optimum value is reached. Therefore, the error between the driving signal and the objective value grows, and the error signal looses its meaning for the control law in (4); it only causes oscillations as the sine function changes sign with the growing error. In order to solve this problem, the driving input can be designed similar to the one in [14] so that if the objective value is unable to track the driving signal, the driving signal takes some steps back. Another option may be to use some heuristics. If the dynamic effects are not too large, the neural network ESC method is still usable. However, as a result of the smooth velocity reference generation, the approximation based method shows the best performance. Perturbation based ESC again presents a very robust result under the dynamic effects too. Considering its performance in for all of the cases, this method is preferable due to the smooth references and when consistency is an issue.

Fig. 5. Snapshots from the experiment. The robot starts with the initial image on the left. The neural network ESC algorithm provides references in order to maximize the texture density and the snapshot on the right is achieved at the final position.

Among the algorithms that have been extended to the multivariate case in the literature, the approximation based method looks the best. However, it should be noted that, in the presence of noise, the performance of the algorithm will be degraded as in Section III-A. The neural network ESC presents a reasonable result with a good convergence rate. On the other hand, the perturbation based algorithm makes the system travel long distances. All these results are presented in Table I.

V. Experimental Results

In this section, we show that ESC works for our application area by giving an indicative example. A six DOF UR5 type Universal Robots arm is used with a camera mounted on the tooltip. In this experiment, the goal is to change the viewpoint of an object in order to maximize textured surface. This kind of system can be used to increase the success rate of an object recognition system. The texture density value is calculated by counting the edge density over a region, and it is subtracted from the maximum possible value.

The velocity references of the ESC algorithm are fed to the translation motions in the x and y directions of the task space. The orientations around the x and y axes of the tooltip are controlled in order to keep the object always at the center of the image. The translation in the z direction and the orientation around the z axis are not controlled. Based on our analysis in Section III, we chose to use the neural network ESC algorithm since we have noise on the texture density calculation and low dynamics effects.

The initial and final views of the object are given in Figure 5. The objective value with respect to time is given in Figure 6. The ESC algorithm successfully minimizes the objective value and leads the robot to a much better view point for recognition purposes, indicating that the algorithm is indeed a good fit to our experimental conditions. In these experiments, since the workspace of the robot is limited, we needed to introduce the workspace constraints to the ESC problem. We have noticed that, this can be easily achieved in the neural network ESC by forcing a direction change (just like it happens when the threshold of that direction is exceeded) whenever the workspace borders are reached.

	Effect of noise	Effect of system dynamics	Smooth References	Multivariate Extension / Traveled distance
Sliding mode ESC	Distortion in transient, high frequency oscilation in steady state	Minor distortion at transient, risk of drift	No	No
Neural Network ESC	Minor irregularities in steady state	Minor distortion in the steady state	No	Yes / medium
Approximation based ESC	Distortion at the generated velocity, Performance fall at the steady state	Robust	Yes, if noise is negligible	Yes / low, very close to optimum
Perturbation based ESC	Robust	Robust	Yes	Yes / large

TABLE I

SUMMARY OF THE ANALYSIS OF THE ANALOG ESC ALGORITHMS.

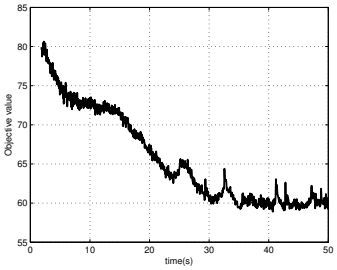

Fig. 6. Objective value plot for the experiment.

VI. CONCLUSION

In this paper, we categorized and summarized a number of analog extremum seeking controllers. They were analyzed for their usability in robotic applications. For the systems with negligible noise and dynamic effects, approximation based methods look like the best option. When noise is present, neural network ESC gives robust and efficient results compared to the other algorithms. Under dynamic effects, the approximation based method presents efficient and robust results. On the other hand, it is observed that, the perturbation based method gives very consistent results with smooth references under both high noise and dynamic effects.

REFERENCES

[1] N. Killingsworth, S. Aceves, D. Flowers, and M. Krstic, "Extremum seeking tuning of an experimental hcci engine combustion timing controller," in *ACC*, 2007.

[2] H.-H. Wang, M. Krstic, and G. Bastin, "Optimizing bioreactors by extremum seeking," *Int. J, of Adap. Control Signal Proc.*, vol. 13, pp. 651–669, 1999.

[3] S. Drakunov, U. Ozguner, P. Dix, and B. Ashrafi, "Abs control using optimum search via sliding modes," *Control Systems Technology*, vol. 3, pp. 79 –85, 1995.

[4] C. Zhang and R. Ordonez, "Robust and adaptive design of numerical optimization-based extremum seeking control," *Automatica*, vol. 45, pp. 634–646, 2009.

[5] Y. Zhang, J. Shen, M. Rotea, and N. Gans, "Robots looking for interesting things: Extremum seeking control on saliency maps," in *IROS*, 2011.

[6] J. Cochran, E. Kanso, S. D. Kelly, H. Xiong, and M. Krstic, "Source seeking for two nonholonomic models of fish locomotion," *Trans. Rob.*, vol. 25, pp. 1166–1176, 2009.

[7] C. Zhang, D. Arnold, N. Ghods, A. Siranosian, and M. Krstic, "Source seeking with non-holonomic unicycle without position measurement and with tuning of forward velocity," *Systems & Control Letters*, vol. 56, pp. 245 – 252, 2007.

[8] B. Calli, M. Wisse, and P. Jonker, "Grasping of unknown objects via curvature maximization using active vision," in *IROS*, 2011.

[9] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, pp. 308–313, 1965.

[10] C. Olalla, M. Arteaga, R. Leyva, and A. El Aroudi, "Analysis and comparison of extremum seeking control techniques," in *ISIE*, 2007.

[11] Nusawardhana and S. Zak, "Extremum seeking using analog non-derivative optimizers," in *ACC*, 2003.

[12] Y. Tan, W. Moase, C. Manzie, D. Nesic, and I. Mareels, "Extremum seeking from 1922 to 2010," in *Chinese Control Conference*, 2010.

[13] S. K. Korovin and V. I. Utkin, "Use of the slip mode in problems of static optimization," *Automatic and Remote Control*, pp. 50–60, 1972.

[14] S. Korovin and V. Utkin, "Using sliding modes in static optimization and nonlinear programming," *Automatica*, vol. 10, pp. 525 – 532, 1974.

[15] H. Yu and U. Ozguner, "Extremum-seeking control strategy for abs system with time delay," in *ACC*, 2002.

[16] M. Teixeira and S. Zak, "Analog nonderivative optimizers," in *ACC*, 1997.

[17] ——, "Analog neural nonderivative optimizers," *Neural Networks, IEEE Transactions on*, vol. 9, pp. 629 –638, 1998.

[18] C. Zhang and R. Ordonez, "Non-gradient extremum seeking control of feedback linearizable systems with application to abs design," in *CDC*, 2006.

[19] K. S. A.R. Conn and P. L. Toint, "On the convergence of derivative-free methods for unconstrainted optimization," IBM T.J. Watson Research C. Ind. Eng. and Operations Res. Dep., Columbia U. Dep. of Math., Facultes Universitaires ND de la Paix, Tech. Rep., 1996.

[20] M. Krstic and H.-H. Wang, "Stability of extremum seeking feedback for general nonlinear dynamic systems," *Automatica*, vol. 36, pp. 595–601, 2000.

[21] K. Ariyur and M. Krstic, "Analysis and design of multivariable extremum seeking," in *ACC*, 2002.

[22] K. B. Ariyur and M. Krstic, *Real-Time Optimization by Extremum-Seeking Control*. Wiley, 2003.

[23] Nusawardhana and S. Zak, "Simultaneous perturbation extremum seeking method for dynamic optimization problems," in *ACC*, 2004.

[24] M. Rotea, "Analysis of multivariable extremum seeking algorithms," in *ACC*, 2000.

[25] M. Krstic, "Performance improvement and limitations in extremum seeking control," *Sys, & Cont. Letters*, vol. 39, pp. 313–326, 2000.

[26] D. Nesic, Y. Tan, and I. Mareels, "On the choice of dither in extremum seeking systems: a case study," in *CDC*, 2006.

[27] M. S. Stankovic and D. M. Stipanovic, "Extremum seeking under stochastic noise and applications to mobile sensors," *Automatica*, vol. 46, pp. 1243 – 1251, 2010.

[28] N. Dragan, "Extremum seeking control: Convergence analysis," *European J. of Cont.*, vol. 15, pp. 331–347, 2009.

[29] R. King, R. Becker, G. Feuerbach, L. Henning, R. Petz, W. Nitsche, O. Lemke, and W. Neise, "Adaptive flow control using slope seeking," in *Medi. Conf. on Cont. and Automation*, 2006.

[30] G. Walsh, "On the application of multi-parameter extremum seeking control," in *ACC*, 2000.

[31] J. Cochran and M. Krstic, "Nonholonomic source seeking with tuning of angular velocity," *Automatic Control, IEEE Transactions on*, vol. 54, pp. 717 –731, 2009.

[32] J. Sternby, "Adaptive control of extremum systems," in *Methods and Applications in Adaptive Control*. Springer Berlin / Heidelberg, 1980.

[33] B. Wittenmark and A. Urquhart, "Adaptive extremal control," in *CDC*, 1995.

[34] B. Wittenmark and R. Evans, "Extremal control of wiener model processes," in *CDC*, 2002.

[35] P. Corke, "A robotics toolbox for matlab," *Robotics Automation Magazine, IEEE*, vol. 3, pp. 24 –32, 1996.