Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 590138, 11 pages
http://dx.doi.org/10.1155/2015/590138

*Research Article*

# Applying BAT Evolutionary Optimization to Image-Based Visual Servoing

**Marco Perez-Cisneros,[1] Gerardo Garcia-Gil,[1] Sabrina Vega-Maldonado,[2] J. Arámburo-Lizárraga,[2] Erik Cuevas,[1] and Daniel Zaldivar[1]**

[1]*Departamento de Ingenierías CUTONALA, Universidad de Guadalajara, Avenida N. Periférico No. 555 Ejido San José Tateposco, 48525 Tonalá, JAL, Mexico*
[2]*Departamento de Sistemas de Información CUCEA, Universidad de Guadalajara, Periférico Norte 799, Los Belenes, 45100 Zapopan, JAL, Mexico*

Correspondence should be addressed to Marco Perez-Cisneros; marco.perez@cutonala.udg.mx

Received 11 October 2014; Accepted 14 May 2015

Academic Editor: Sabri Arik



This paper presents a predictive control strategy for an image-based visual servoing scheme that employs evolutionary optimization. The visual control task is approached as a nonlinear optimization problem that naturally handles relevant visual servoing constraints such as workspace limitations and visibility restrictions. As the predictive scheme requires a reliable model, this paper uses a local model that is based on the visual interaction matrix and a global model that employs 3D trajectory data extracted from a quaternion-based interpolator. The work assumes a free-flying camera with 6-DOF simulation whose results support the discussion on the constraint handling and the image prediction scheme.

## 1. Introduction

Past decades have witnessed the extensive development of the visual servoing (VS) control. Three fundamental schemes have practically represented most of VS implementations [1, 2]. First, the image-based visual control (IBVS), also known as 2DVS, employs an error computation between the visual features belonging to a target object in a given image and its corresponding features in a target image. Such error is subsequently employed as guidance for the visual control algorithm just as it is carefully detailed in the following. Second, the position-based visual servoing (PBVS), also named as 3DVS, works entirely on the visual computation of geometric poses, whose values are subsequently used to regulate the camera movement. Likewise, a third group is represented by a wide number of hybrid VS approaches that generally profit over a mindful combination between 2DVS and 3DVS advantages.

In particular, the classic IBVS control problem is defined as an exponential minimization of the aforementioned image plane error between the current and target image $\mathbf{e}(t) = \mathbf{s}(t) - \mathbf{s}^*$. In turn, such error can be subjected to a classic minimization procedure assuming a gradient-like approach such that $\dot{\mathbf{e}} = -\lambda \mathbf{e}(t)$. A well-known relationship between the object's velocity and its corresponding image plane velocity can thus be defined by stocking each point velocity relationship into a single matrix $\mathbf{L}_s$ known as the interaction matrix or the visual Jacobian. Mathematically, the overall velocity relationship can thus be defined (see [1]) as $\dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v}(t)$ with $\mathbf{L}_s$ being the interaction matrix and $\mathbf{v}(t)$ representing the velocity screw vector over time. A classical feedback control law can thus be defined as

$$\boldsymbol{\tau}(t) = -\lambda \widehat{\mathbf{L}}_s^{+} \mathbf{e}(t). \tag{1}$$

In this case, $\widehat{\mathbf{L}}_s^{+}$ is the pseudoinverse of the image Jacobian matrix and $\lambda$ a negative constant, with $\boldsymbol{\tau}(t)$ being the resulting control signal. Despite the implementation of such VS scheme being fairly simple, some important drawbacks have been highlighted by Chaumette in [3], with unstable behavior arising from the tracking of large displacements and complex rotations, or from the generation of nonfeasible motions.

Therefore, the handling of either the 2D constraints or the 3D limitations, as well as the generation of feasible trajectories for a given visual task, must be all appropriately addressed.

Two constraints must be appropriately handled in order to assure an appropriate visual control behavior: first, the well-known visibility constraint that refers to the adequate handling of the control problem in order to assure that visual features always remain within the camera field of view and second the 3D constraint that challenges the generation of convenient visual servoing schemes that yield admissible camera motions within a valid workspace.

The use of optimal control fundamentals for visual servoing has been defined as an appropriate and convenient tool to build visual servoing schemes that carefully considered the aforementioned visual constraints. Actually, several applications have been reported in the literature over the last two decades. First, the seminal works of Hashimoto and Kimura in [4] and Schramm and Morel in [5] that incorporated an LQ-based optimal control scheme and a Kalman filter-based algorithm, respectively, in order to guide the movements of a robotic manipulator.

Other approaches have capitalized the advantages of the LMI approach to build predictive control schemes for visual servoing [6, 7]. Despite the fact that such works have focused over the designing of an appropriate control law for the visual servoing scheme, other proposals have also included optimal schemes for the combination of path planning and trajectory tracking in order to assure the fulfilment of the visibility constraint and the generation of an optimal trajectory for the camera. Excellent examples of such combination can be found in the works of Schramm and Morel in [8] and the use of LMI structures made by Chesi in [9]. In the particular case of path planning, it is important to consider the work of Mezouar and Chaumette [10] and the robust approach proposed later by Kazemi et al. in [11]. In this case, an LMI based algorithm is used to define an optimal path planning solution assuming that not a unique solution for the problem may exist and also that it may not be unique, while the required camera tracking is supplied through a classic image-based visual controller [12].

Other optimal VS control implementations include the use of predictive control to compensate for errors in the tracking task of a visual feedback scheme in case of no prior information about the 3D model being supplied to the visual controller [13] or in the case of using active filtering through predictive control for biomedical applications that support robotized surgery [14].

Recently, the strategy to incorporate the handling of both visual constraints, that is, the visibility and the feasible motion constraint, within the visual control structure has been focused on expressing the overall visual task from a nonlinear optimization perspective. Therefore, this paper presents a novel optimization scheme that employs an evolutionary optimization method to handle both constraints through a visual predictive control scheme. Under such circumstances, 3D constraints can be considered as state variables while the visibility constraint can be assumed just as a constraint within the output space, just as it has been done in [12]. In order to provide an appropriate model prediction agent, two options are to be considered following the proposal of Allibert and Courtial in [15]. First, a local model uses the classic image Jacobian matrix while a second test uses a quaternion-based 3D trajectory generator. As it will be carefully discussed, the optimization algorithm uses prediction to improve the overall visual servoing performance by means of a predictive control structure that has been specifically designed to fit within the visual control scheme.

Just as it has been widely demonstrated, the use of optimization within the visual servoing control scheme has delivered some relevant contributions in particular for the image-based schemes that naturally handle the most important visual constraints at the same time control signals are generated with remarkable examples being found in [12, 15–17]. However all these solutions use classic optimization methods in order to minimize an objective function since the goal of an optimization scheme is to find an acceptable solution of a given objective function that is defined over a given search space; novel methods that are known as Evolutionary Methods have been proposed as a handy alternative.

In particular, evolutionary algorithms (EA), which are considered as stochastic optimization methods, have been developed by a combination of rules and randomness that mimics several natural phenomena that include some evolutionary processes such as the evolutionary algorithm (EA) proposed by Fogel et al. [18], De Jong [19], and Koza [20], the Genetic Algorithm (GA) proposed by Holland [21] and Goldberg [22], the Artificial Immune System proposed by de Castro and bon Zuben [23], and the Differential Evolution Algorithm (DE) proposed by Storn and Price [24]. Some other methods which are based on physical processes include the Simulated Annealing proposed by Kirkpatrick et al. [25], the Electromagnetism-Like Algorithm proposed by Birbil and Fang [26], and the Gravitational Search Algorithm proposed by Rashedi et al. [27]. Also, there are other methods based on the animal-behavior phenomena such as the Particle Swarm Optimization (PSO) algorithm proposed by Kennedy & Eberhart [28], the Ant Colony Optimization (ACO) algorithm proposed by Dorigo et al. [29], and the BAT algorithm proposed by Yang [30], which is of special importance for this paper.

In particular, this paper approaches the IBVS from an optimization-like perspective that naturally supports the inclusion of visual constraints in the implementation of the vision-based control scheme. As a result, the overall performance of the visual servoing scheme is improved at the same time that the aforementioned constraints are carefully taken into consideration.

The paper has been developed as follows. Section 2 presents an overview of the overall optimization strategy, the control scheme, and its mathematical formulation, as well as the management of image-based constraints that support the optimal IBVS approach. Section 3 focuses on the principles of the BAT optimization algorithm and its basic operational principles. Section 4 discusses the local and global models that are required in the image-prediction scheme, which in turn are represented by the classic IBVS control algorithm and the quaternion-based guidance. Section 5 presents some simulation of the free flying 6-DOF camera in order to

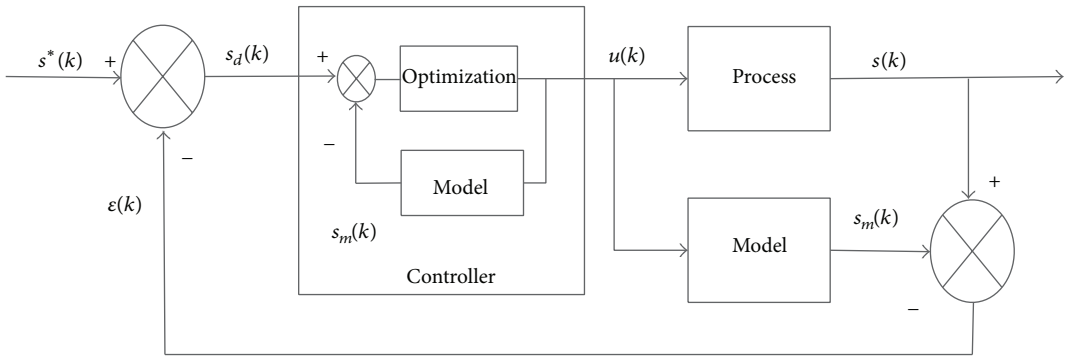



Figure 1: Predictive visual control scheme.

demonstrate the active contribution of the the algorithm's tracking performance and discuss the differences between using the local or the global model for prediction. The last section draws some final conclusions.

## 2. An Optimization Approach to IBVS

*2.1. Structure of the Control Scheme.* One of the most successful strategies to incorporate optimization into a feedback control scheme is beyond any doubt of the predictive control. In turn, one of the most well-known structures for predictive control is the internal model control approach [31], whose basic structure has been customized for the image-based visual servoing in the work of Allibert [12]. The basic structure is reproduced in Figure 1 where the robot and its attached camera are modelled inside the plant block. The control input to the system is represented by $u$ while the output has been marked as $s$ which represents the image plane coordinates of four selected features to track in the image of the object of interest. However, as it is typical in IBVS, the scheme requires the definition of desired (target) locations for the object features in the image, typically represented by $s^*$. By making use of the error model for IBVS from (1), the predictive control is based upon a generalized error that is defined by the difference between the current plant output at time $k$ and the corresponding model output. Define such generalized error $\varepsilon(k)$ as the difference of the system's output $s(k)$ and the predicted model output $s_m(k)$, yielding $\varepsilon(k) = s(k) - s_m(k)$, at time $k$. The algorithm should assure that a desired trajectory of visual features on the image plane follows an adequate sequence of points in order to guarantee the fulfillment of both visual constraint that have been mentioned earlier. Therefore, an easy definition for the required trajectory $s_d(k)$ can be defined as the difference between the target feature locations $s^*(k)$ and the preregistered plant-model error $\varepsilon(k)$, which in turn generates the following expression:

$$s_d(k) = s^*(k) - \varepsilon(k). \tag{2}$$

The overall error $\varepsilon(k)$ that includes the plant-model difference at time $k$ can be included yielding:

$$s_d(k) = s^*(k) - [s(k) - s_m(k)]. \tag{3}$$

A very interesting fact emerges as the overall equation is rewritten as follows:

$$s_d(k) - s_m(k) = s^*(k) - s(k). \tag{4}$$

This last expression holds a key issue for the optimization approach of IBVS schemes. The minimization of the difference between the desired visual features location $s^*(k)$ and the system's output $s(k)$ corresponds to minimizing the difference between the required visual trajectory $s_d(k)$ and the model output $s_m(k)$. Actually, the last fact supports the operation of the optimization algorithm that is to be completed if an objective function and some operative rules are defined as it is discussed below.

*2.2. Building the Mathematical Framework.* As explained above, the definition for the predictive control structure depends on drawing an appropriate objective function of the form: $\min_{\tilde{u}\in K} J(u)$, which will yield a control sequence of the form:

$$\tilde{u} = \{u(k), u(k+1), \ldots, u(k+N_c), \ldots, u(k+N_p-1)\} \tag{5}$$

with $N_p$ and $N_c$ representing the prediction and control horizon, respectively. The prediction horizon represents the amount of forecast terms to be calculated in advance from the model while the control horizon holds the number of calculated terms that are actually applied to control the plant [32]. In the particular IBVS implementation, only the first term of the control horizon is actually applied to the system [12].

Considering that the overall problem is managed over the image plane and that the visibility constraint is referred to the image plane, the objective function can be initially defined as follows:

$$J(u) = \sum_{j=k+1}^{k+N_p} [s_d(j) - s_m(j)]^T Q(j) [s_d(j) - s_m(j)] \tag{6}$$

with $Q(j)$ being a weighting symmetric definite-positive matrix with the dynamics of the system being described by the nonlinear system: $x(j) = f[x(j-1), u(j-1)]$ and

$s_m = h(x(j))$ with $s_d(j) = s^*(j) - \varepsilon(j)$ and $x(j)$ representing the predicted state at time $j$ with $\forall j \in [k+1, k+N_p]$. The variables for the state and the control signal are defined according to $x, u \in \mathbb{R}^n$ and the model output as $s_m \in \mathbb{R}^p$. It is important to note that the state computation can vary depending on the particular prediction model that is employed. This issue is carefully addressed in the following.

*2.3. Constraints of the Predictive IBVS.* Since one of the immediate advantages of the optimization-like approach of the IBVS control is the natural handling of inner constraints of the visual challenge, it is important to denote how such constraints are to be managed by the proposed structure.

The most important constraints have been previously identified as the visibility constraint and the 3D motion constraint. The case for the visibility constraint is also known as the 2D condition. It aims to assure that the location of object's features of interest for the visual algorithm always remains within a valid location in the image plane. On the contrary, this property can be used to denote inconvenient areas within the image. In terms of the optimization algorithm, the constraint is simply introduced under the limit $s_{\text{low}} \leq s_m(k) \leq s_{\text{up}}$ that includes both the lowest and the highest accepted location within the image space.

On the other hand, the generation of valid 3D trajectories also can be easily included in the optimization process. Since each robotic device must comply with mechanical and dynamic limitation due to workspace limits or actuator saturation, each kinematic pose can be defined in terms of the corresponding instantaneous generalized coordinates $\mathbf{q}(k)$. Likewise, the overall pose $\mathbf{T}(q)$ can also be geometrically constraint under well-known properties such as the full-rank in the instantaneous Jacobian matrix [33]. Therefore, again both considerations can be introduced under the following expressions:

$$\begin{aligned} {}^0\mathbf{T}_{n_{\text{low}}} &\leq {}^0\mathbf{T}_n(k) \leq {}^0\mathbf{T}_{n_{\text{up}}}, \\ \mathbf{q}_{\text{low}} &\leq \mathbf{q}(k) \leq \mathbf{q}_{\text{up}} \end{aligned} \tag{7}$$

with ${}^0\mathbf{T}_{n_{\text{low}}}$ and ${}^0\mathbf{T}_{n_{\text{up}}}$ being minimum and maximum allowed pose while $\mathbf{q}_{\text{low}}$ and $\mathbf{q}_{\text{up}}$ represent the lowest and the highest generalized coordinate limits. In a similar fashion, it is even feasible to include other mechanical constraints such as actuator limits or torque or force constraints [12]. The optimization procedure commonly includes all aforementioned constraints in the form of nonlinear expression that can be evaluated as the overall predictive algorithm evolves.

*2.4. Optimal Approach to IBVS.* This section discusses the step by step implementation of the control structure presented by Figure 1. Two parameters are of vital importance in the implementation of the prediction horizon and the control horizon value. Both will in turn coordinate the extent of the optimization influence inside the predictive control scheme. The visual predictive controller and its corresponding optimization cycle are initially computed starting from the error calculation. Such error value is subsequently used to draw the desired trajectory over the number of steps that are defined by the prediction horizon $N_p$. A step-by-step description is presented below.

(1) Current location of visual features $s(k)$ is registered.

(2) Calculate the value of error $\varepsilon(k) = s(k) - s_m(k)$, assuming it is kept constant during the number of steps included in the prediction horizon $N_p$; that is, $\forall j \in [k+1, k+N_p]$.

(3) Compute the desired trajectory according to $s_d(j) = s^*(j) - \varepsilon(j)$, also $\forall j \in [k+1, k+N_p]$.

(4) The measured current feature location in the image plane $s(k)$ is employed to initialize the model output $s_m(k)$, which in turn constitutes the feedback loop that is required by the internal model control structure [31].

(5) The optimal control signal $u(k)$ is defined by the optimization algorithm according to (5); its value is kept constant over the interval $u(k+N_c+1)$ to $u(k+N_p-1)$, with $N_c$ and $N_p$ being the control and the prediction horizon, respectively.

Evidently, the two most important parameters in the optimization process are the $N_c$ and $N_p$. The value of $N_p$ is vital to guarantee an adequate equilibrium between the system stability and the computational feasibility of the overall implementation. A high value of $N_p$ implies the generation of softer control signals while a small value allows a wider exploration of novel control values at the cost of weakening the overall system stability. On the other hand, the control horizon value $N_c$ regulates how many steps forward are required to reach the objective. A high value of $N_c$ accounts for a slower control behavior that is not feasible for visual control implementations. In practical terms, the value of $N_c$ is commonly assigned to 1, which corresponds to keeping the control signal constant over the number of steps previously defined by the prediction horizon $N_p$.

Finally, it is important to discuss about weighting matrix $Q(j)$ that is a third participant of the optimization configuration set. Its value is commonly assigned to an identity matrix of dimension $(p, p)$, despite some successful examples of using a time-varying matrix values in order to increase the sensibility to the error value when the steps are close to reach the horizon value $N_p$ [15].

Since the overall mathematical design of the visual predictive control has been envisioned, the study should turn to discuss over the feasibility of employing an evolutionary optimization algorithm to increase the performance of a predictive visual control structure.

## 3. The BAT Evolutionary Algorithm

Approaching the IBVS from an optimization-like perspective naturally supports the inclusion of visual constraints in the implementation of the vision-based control scheme. In particular, the use of the BAT evolutionary algorithm as the main optimization procedure provides an easy implementation while still delivering an acceptable performance.

The analogy supporting the BAT algorithm is based on the echolocation ability that is exhibited by microbats in their quest for food. Bats generate an ultrasonic beam that can vary its pulse frequency or its intensity which is commonly known as the loudness in the algorithm. The ultrasonic signal is delivered in advance to their movements. By using loudness variations, intensity variations between both ears and the time delay in receiving the signals back, bats are able to reconstruct an overall scenario despite the fact that they may move through a varying context.

The BAT algorithm is therefore built over the assumption that they fly at random while looking for food. Such movement is registered at position $\mathbf{x}_i$ with velocity $\mathbf{v}_i$ assuming the bat emits a fixed frequency $f_{\min}$ and a variable with loudness $A_0$. During the search, the pulse emission rate $r \in [0, 1]$ can vary in accordance to the proximity of the target. For simplicity the analogy considers frequency to fall in the interval $f \in [0, f_{\max}]$ assuming that higher frequencies imply a shorter travelling distance. Under the same simple assumption, the rate pulse is computed as $r \in [0, 1]$, with 1 representing the maximum rate of pulse emission [30].

The heart of the BAT algorithm is centered over the location computation for virtual bats. The movement is administered over the $d$-dimensional search space by updating positions $\mathbf{x}_i$ and velocities $\mathbf{v}_i$ as follows:

$$\mathbf{x}_i(k) = \mathbf{x}_i(k-1) + \mathbf{v}_i(k), \tag{8}$$

$$\mathbf{v}_i(k) = \mathbf{v}_i(k-1) + \left[\mathbf{x}_i(k) - \mathbf{x}^*\right] \cdot f_i, \tag{9}$$

$$f_i = f_{\min} + \left(f_{\max} - f_{\min}\right) \cdot \beta. \tag{10}$$

In this case, $\mathbf{x}^*$ represents the current global best solution after assessing all current available solutions. Frequency $f_i$ is computed through the difference between $f_{\max}$ and $f_i$, with $\beta$ being a uniformly distributed random value. In practice, the value of $f_i$ is of vital importance because it controls the movement scope for each particle.

In a similar fashion, loudness and pulse rate are defined with regard of the bat's analogy. Loudness $A$ should decrease as the bat is approaching its prey. A range between 0 and 1 is used to implement this feature, with $A = 0$ being used when one bat (searching particle) has found a prey (minimum) and therefore is not emitting a sound signal; otherwise $A = 1$ signals when the bat is searching through the space and therefore producing its maximum sound. An easy implementation considers a variable $0 < \alpha < 1$, yielding

$$A_i(k+1) = \alpha \cdot A_i(k). \tag{11}$$

The pulse emission rate $r$ is defined under a similar scheme but assuring an exponential decay on its influence as time is evolving as follows:

$$r_i(k+1) = r_i(0)\left[1 - e^{-\gamma t}\right]. \tag{12}$$

With $t$ representing sampling time, $t \to \infty$, and $\gamma$ being a tuning constant. Avoiding any loss of generality, $r_i(0)$ accounts for the last updated value of pulse emission rate. It must be noticed that according to the original BAT implementation [30], loudness and pulse emission rate will only be updated if new solutions are improved which means that searching particles are moving closer to an optimal solution.

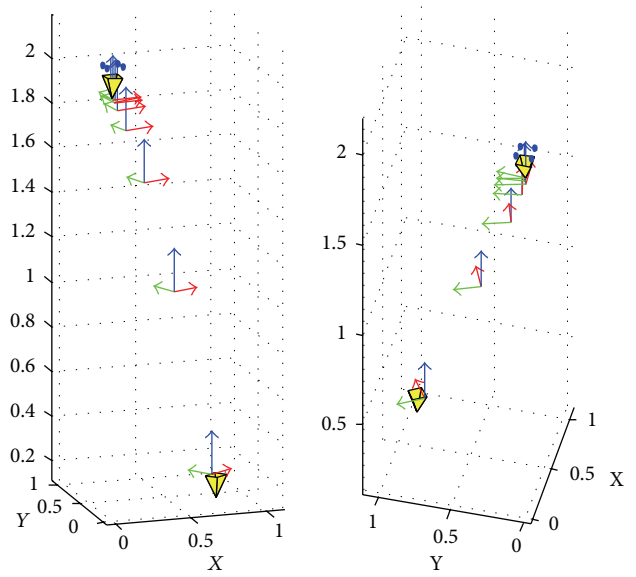
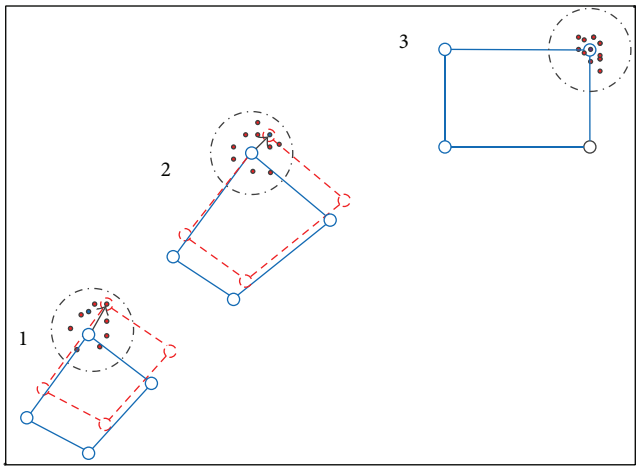



Figure 2: Classic visual servoing scheme.




- Particles generated by the heuristic algorithm BAT
- Particle generated by the quaternion spherical linear terpolation.
- Coordinates of the desired position $f(x, y)$

Figure 3: Contribution of the slerp quaternion-based interpolation within the optimal visual servoing scheme.

The overall BAT algorithm can be summarized over the following pseudocode:

(1) Define the initial population $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$ and initial velocities in vector $\mathbf{v}_i$.

(2) Select a pulse frequency $f_i$ at $\mathbf{x}_i$ and pulse rates $r_i$ and loudness values $A_i$.

(3) Do until get to $k$ number of iterations:

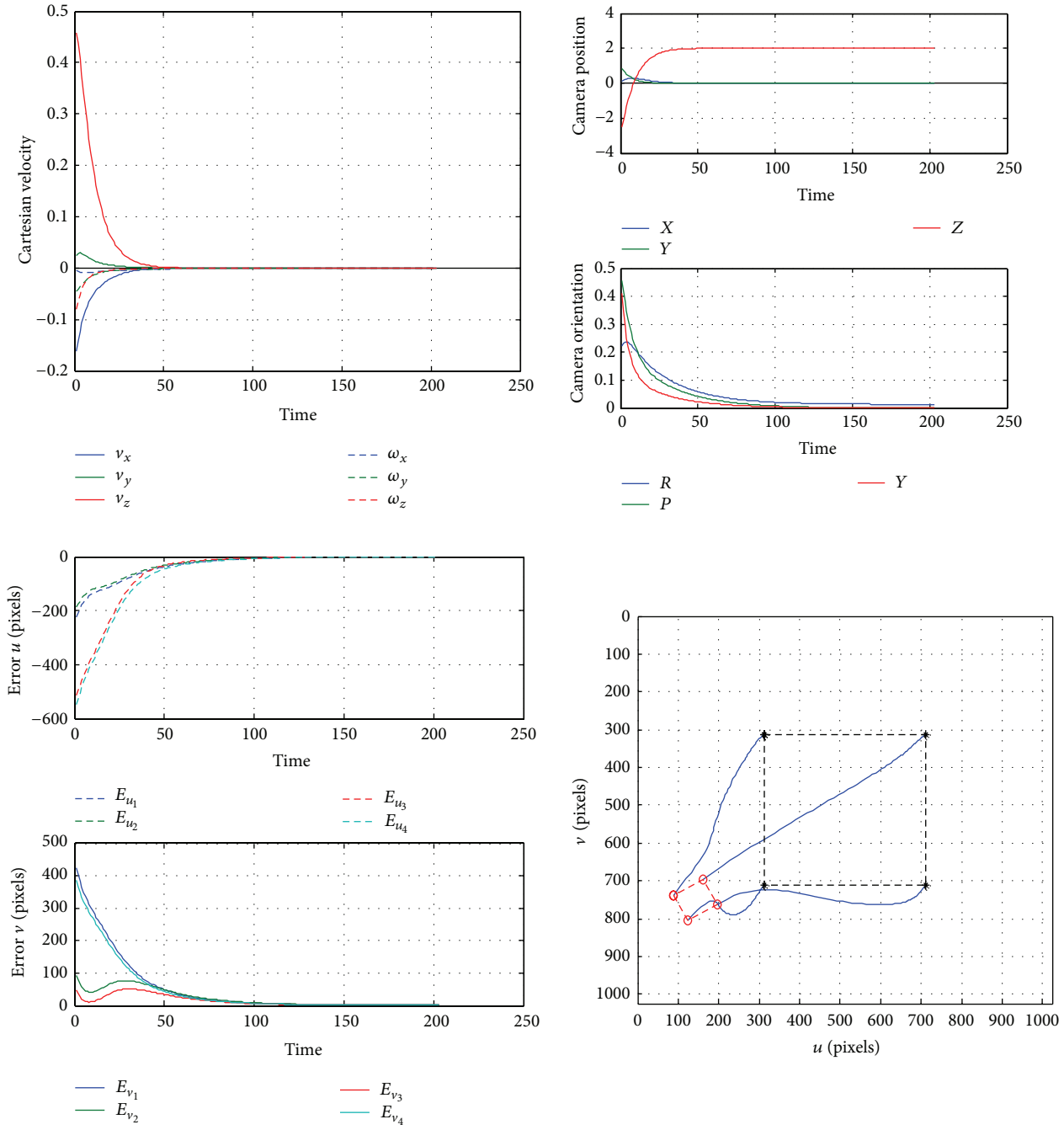



FIGURE 4: Evolution of the visual servoing control scheme, showing (clockwise from upper left) the screw vector evolution, pose errors, feature trajectories in the image plane, and feature trajectories in the horizontal and vertical directions.

(3.1) Calculate new solutions through frequency $f_i$.

(3.2) Update velocity and location for each bat (particle), using (8) and (9).

(3.3) Generate a random value $r$ and compare to $r_i$.

(3.4) If $r > r_i$, one solution among best solutions must be chosen. A new local solution must be generated around that selected best solution.

(3.5) Using a random bat's fly (particle movement), generate a new solution.

(3.6) If $f(x_i) < f(x^*)$ and $r < A_i$, then accept new solutions, increase $r_i$, and decrease $A_i$.

(3.7) Reevaluate all particles to find the current new best $x^*$.

(4) Publish results.

In practical grounds, the use of a similar value for $\alpha$ and $\gamma$ yields a similar treatment for decreasing the loudness and for increasing the emitted pulses rate at the time

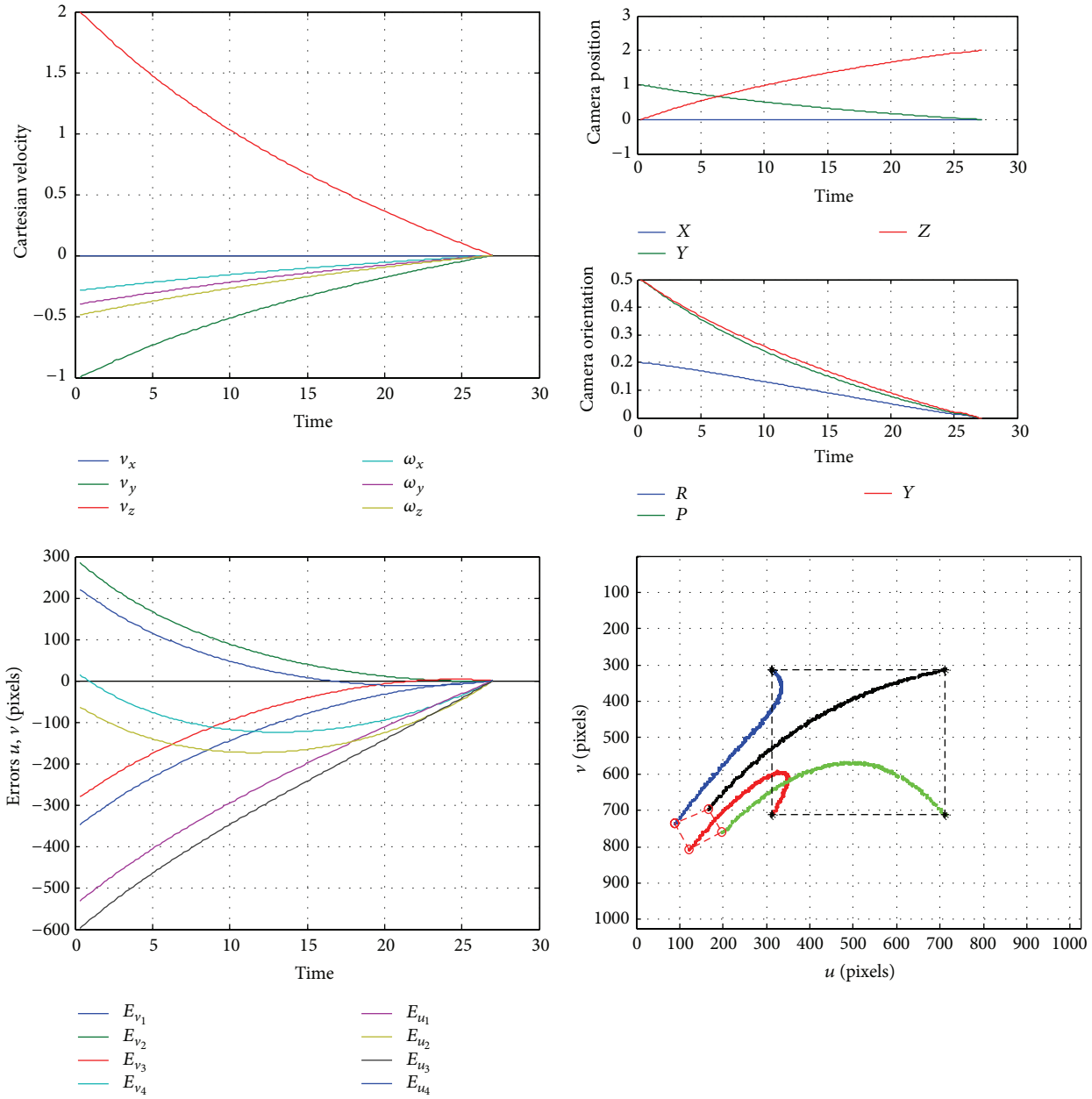



FIGURE 5: Predictive visual servoing control scheme using the BAT evolutionary algorithm with (clockwise from upper left) the screw vector evolution, pose errors, feature trajectories in the image plane, and feature trajectories in the horizontal and vertical directions.

a prey (minimum) is being located. Appropriate values for both values must be experimentally determined. In our implementation a simple selection of $\alpha = \gamma = 0.9$ has been used with good results. Several other BAT implementations with different parameters settings are reported in [32].

## 4. The Local and Global Mathematical Models for Optimization

Once the BAT optimization algorithm has been carefully developed, the discussion should turn into the selection of the plant model that is to guide the predictive control strategy. The model is used to predict the movement of visual features with respect to the camera velocity over a finite prediction horizon [12]. The classic visual servoing scheme can be easily implemented to provide a simple model, denoted as SM in the following. Such model uses fundamental equations to define the movements on the image plane as a result of the camera movement. This paper has also explored the use of a quaternion-based interpolation to generate the required location of the visual features over the image plane. Fundamentals of the classic visual servoing model, denoted as SM, are sketched below.

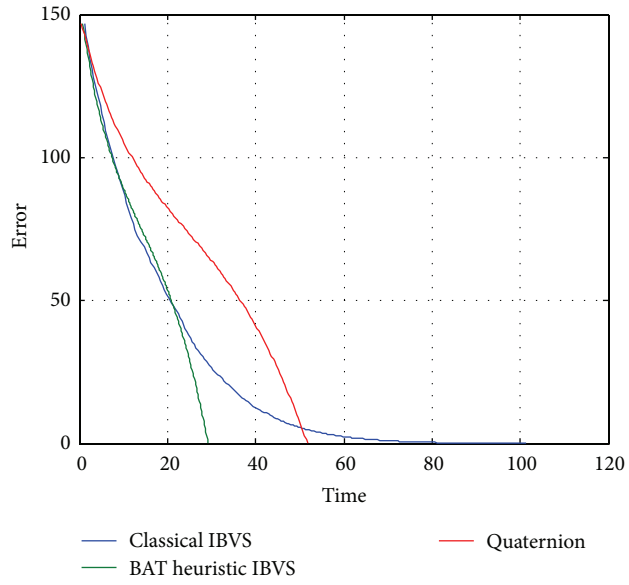



FIGURE 6: Time performance of the three algorithms: the predictive visual servoing control scheme, the classic IBVS scheme, and a pure-quaternion guidance of the visual servoing task.

*4.1. Local Model: The Classic Visual Servoing Scheme.* The most well-known visual servoing scheme employs point-like image features as main reference. The location of such features in the image plane is denoted as $s_m = (u, v)$, with $m$ referring to the feature number and $u$ and $v$ representing the vertical and horizontal image plane coordinates, respectively. For a given 3D point in the space, that is, $p = [x, y, z]^T$, that is defined with respect to the camera frame, its projection into the image plane is easily defined in normalized coordinates assuming $u = x/z$ and $v = y/z$. Likewise, the camera velocity is defined through the classic screw vector $w = [d_x, d_y, d_z, w_x, w_y, w_z]^T$. A careful review of the fundamentals of visual servoing in [1] explains the velocity relationship from the camera velocity $w$ to the image plane feature's velocity through the following relationship:

$$\dot{s}_m(t) = L_s(t)\, w(t), \tag{13}$$

where $L_s(t)$ represents the image Jacobian or interaction matrix which holds the required velocity relationship. The overall matrix is built by literately piling up the following matrix for each feature that is being characterized, as follows:

$$L_s = \begin{bmatrix} -\frac{1}{z} & 0 & \frac{u}{z} & uv & -\left(1+u^2\right) & v \\ 0 & -\frac{1}{z} & \frac{v}{z} & 1+v^2 & -uv & -u \end{bmatrix}. \tag{14}$$

The location of each feature can thus be defined through a simple integration method as follows:

$$s_m(k+1) = s_m(k) + \widehat{L}_s(t) \cdot T_e \cdot w(t) \tag{15}$$

with $T_e$ representing the sample time, $w$ defining the screw vector, and $\widehat{L}_s(t)$ accounting for the best estimation of $L_s(t)$, considering that depth information is required and must be either calculated through a 3D model of the object of interest or through a careful assessment of its approximated value. In this paper, we use the advantage of holding a full 3D model of the image formation and therefore its computation can be exactly defined. Following the procedure in [12], the optimization process can be acutely described by considering each feature $s_m$ as a state $x$ which allows expressing the overall optimization function as follows:

$$\begin{aligned} x(k+1) &= x(k) + \widehat{L}_s(t) \cdot T_e \cdot w(t) \\ &\longrightarrow f(x(k), w(t)), \\ s_m(k) &= x(k) \longrightarrow h(x(k)). \end{aligned} \tag{16}$$

By using such optimization function, 2D constraints are naturally handled within the SM visual servoing model.

*4.2. Global Model: Spherical Interpolation.* A global model is required to generate an alternative option in order to support the optimization contribution within the visual servoing scheme. In this case, the quaternion-based interpolation, also known as slerp [34], is a very useful tool considering its intrinsic advantages such as the smooth interpolation, fast concatenation and simple inversion of angular displacements, and a quick conversion to homogeneous transforms.

The classic visual servoing problem considers both a start and a target pose which, in turn, can be easily expressed in quaternion grounds. Once both are converted, the interpolation is easily computed by the following expression:

$$\begin{aligned} \text{slerp}(q_0, q_1, t) = \left(q_1 q_0^{-1}\right)^t q_0 \longrightarrow q_0 = T(0); \\ q_1 = T_d. \end{aligned} \tag{17}$$

With $t$ being the step index, $0 < t < 1$ whose value defines the interpolation step, with $t = 1$ signaling for the last step in the sequence. The spherical interpolation is incorporated into the predictive control strategy as the model guiding the BAT optimization algorithm. It compares the fitness of the proposed interpolated pose through its corresponding features and those features that are generated by each proposed particle. The comparison variables can be easily explained by the following three cases that are illustrated by Figure 3, as follows:

(1) Particles generated by the BAT evolutionary algorithm are depicted in bold circles while the particle generated by the spherical linear interpolation is represented in a void circle. The algorithm compares the fitness value for each candidate solution. In this case, one of the BAT generated particles is selected.

(2) In the second case, the particle generated by the slerp interpolator is chosen as it has obtained the best fitness value.

(3) The third case shows when the algorithm has reached the desired location and the visual servoing task is finished.

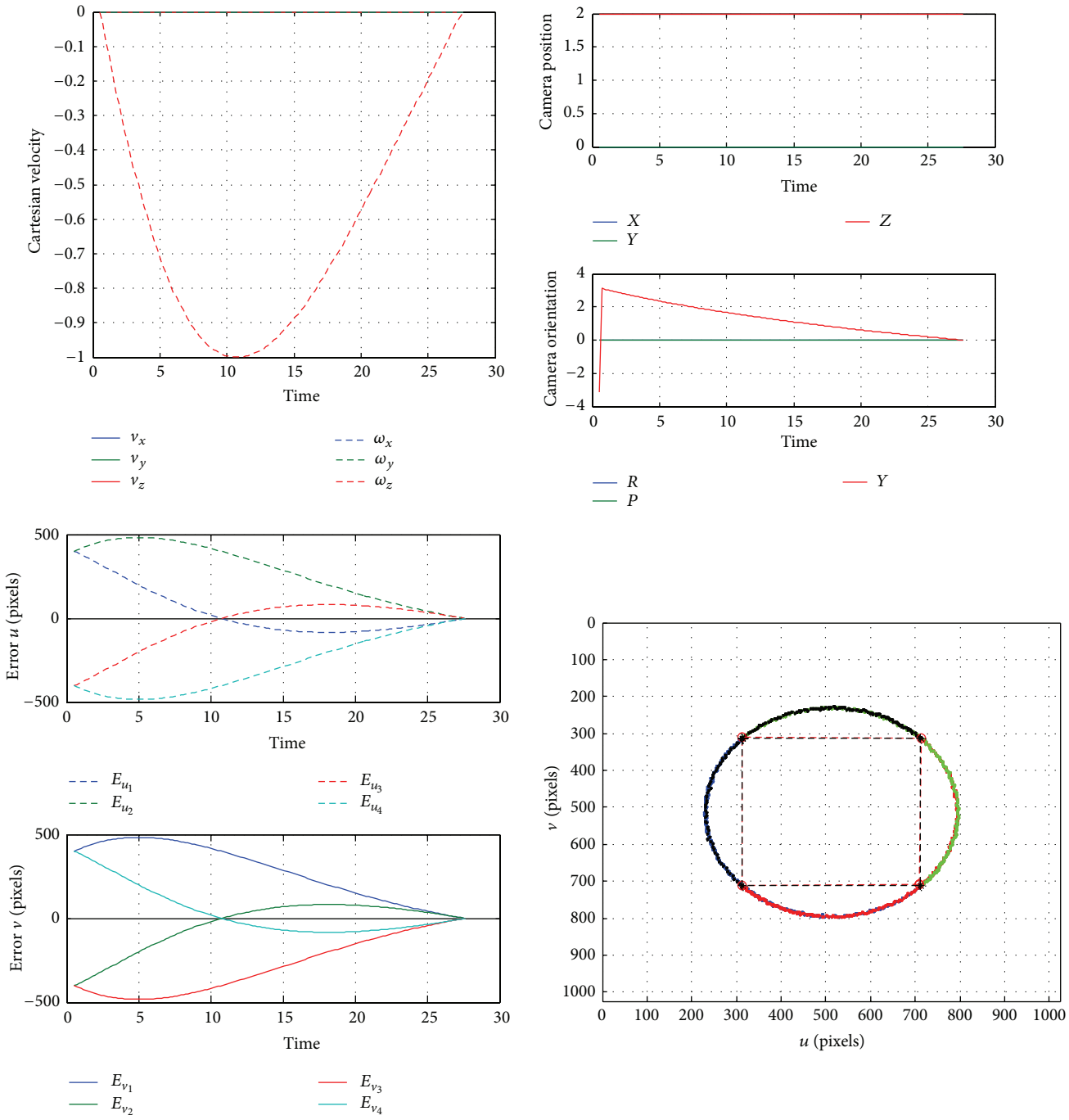



FIGURE 7: Predictive visual servoing control scheme using the BAT evolutionary algorithm to solve the half turn problem, with (clockwise from upper left) the screw vector evolution, pose errors, feature trajectories in the image plane, and each feature trajectory in the horizontal and vertical directions.

## 5. Simulations

For all the simulations in this paper, the sample time is constraint by the number of frames per second that is provided by the camera. Since a 30 frames per second device is used, the sampling period is $t = 33$ ms. A free-flying camera is located in an initial position defined by vector $\mathbf{x}(0) = [0, 1, -3, 0.2, 0.5, 0.5]^T$. First three values are defined in meters while last three components, the roll, pitch, and yaw angles, are referred in radians. Figure 2 shows the initial and final pose for the camera. It also illustrates the four features that represent the object of interest. The simulation uses the classic SM model with the classic image Jacobian matrix.

Figure 4 shows the evolution of the visual servoing control, with the graphs showing, in clockwise order from the upper left corner, the resulting screw vector evolution, the

camera pose error, considering location and orientation, and the image plane feature trajectories and the image errors in the horizontal and vertical direction, respectively.

The predictive visual servoing scheme is tested over the same initial conditions. 20 bat particles are initialized to random positions; the pulse emission rate and the loudness value are defined to $\alpha = \gamma = 0.7$, with the decay variable $\alpha$ falling within the range of 0 to 1.

Figure 5 presents the results of the predictive visual servoing scheme with a BAT optimization algorithm in the control feedback. The graphs show the evolution of the visual servoing control featuring, in clockwise order from the upper left corner, the resulting screw vector evolution, the camera pose error, considering location and orientation, and the image plane feature trajectories and the image errors in the horizontal and vertical direction, respectively.

In practical grounds, the classic visual servoing scheme required 39.62 seconds to complete the full servoing task. On the other hand, the BAT-based predictive visual controller required only 13.28 seconds in order to accomplish the same assignment.

A very handy comparison of the BAT visual predictive controller is related to the execution time of such algorithms and their contrast to the use of a quaternion-based simple interpolator. Figure 6 shows the time improvement of the BAT-based predictive visual control scheme.

Figure 6 shows a time comparison between the classic IBVS method, the pure-quaternion solution, and the BAT-based predictive visual control scheme. The BAT-based scheme shows a sharper trajectory in the image plane as a result of the improvement related to the optimal use of the solution holding the best fitness value. It is important to remind that the scheme is also capable of naturally handling both of the required visual constraints.

A challenging second experiment is performed in order to demonstrate the handling of both aforementioned visual constraints. A typical problem in the classic visual servoing scheme emerges when the required movement is a rotation of $\pi$ radians over the vertical axis of a given object; its visual feature tracking typically tends to fail in the control law, because such movement implies a sudden backwards movement of the camera, which in turn yields a tracking failure for the object of interest features.

The use of the predictive visual control scheme with the BAT optimization algorithm in the feedback loop yields a solution for the aforementioned problem. Thanks to the particle delivery and the fitness evaluation for each particle during the control phase, the controller manages to generate appropriate spherical trajectories that avoid any instability due to the feature trajectory crossing in the image plane [3]. Figure 7 shows an illustration of the optimal controller response to such circumstance.

Notice in Figure 7 that the generated screw vector requires velocity only in the direction of $\omega_x$ as the controller aims to turn the overall problem 180 degrees. A schematic view is illustrated by Figure 8. It is evident how the required movement is simply a rotation around the vertical axis. The required time to complete the turning of image features reaches the 27 seconds. The feature trajectories in the image plane seem to describe a circular movement as a result of the contribution from the quaternion-based interpolator.

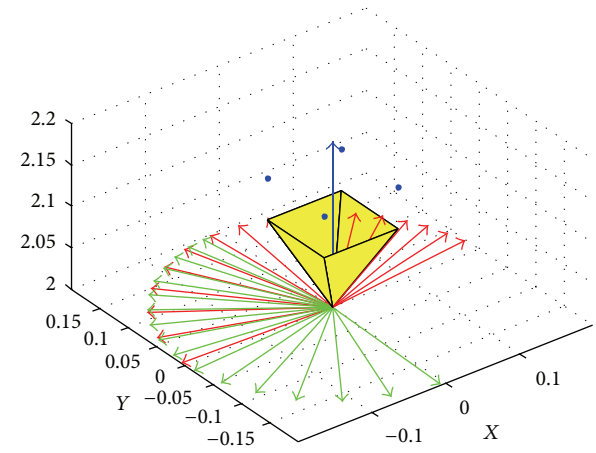



Figure 8: 3D view of the visual servoing control for the half turn problem.

## 6. Conclusions

This research has demonstrated the usefulness of a predictive control strategy for an image-based visual servoing scheme that employs an evolutionary optimization algorithm to improve the performance of the servoing task. The visual control task is approached as an optimization problem that naturally handles relevant visual servoing constraints such as workspace limitations and visibility restrictions. Two models have been used in the implementation: a simple model based on the classic visual servoing scheme and a quaternion-based slerp interpolator. The mindful contribution of both models is controlled through a particular objective function that evaluates the fitness of the proposed interpolated pose through its corresponding features and those features that are generated by each proposed particle. In practical grounds, the spherical interpolation is incorporated into the predictive control strategy as the model guiding the BAT optimization algorithm. The simulation results support the contribution of the proposed scheme regarding the handling of required visual constraints.

## Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

## References

[1] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[2] F. Chaumette and S. Hutchinson, "Visual servo control. II. Advanced approaches," *IEEE Robotics and Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.

[3] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The Confluence of Vision and Control*, vol. 237 of *Lecture Notes in Control and Information Sciences*, pp. 66–78, Springer, London, UK, 1998.

[4] K. Hashimoto and H. Kimura, "LQ optimal and nonlinear approaches to visual servoing," in *Visual Servoing*, K. Hashimoto, Ed., vol. 7 of *World Scientific Series in Robotics and Intelligent Systems*, pp. 165–198, World Scientific, Singapore, 1993.

[5] F. Schramm and G. Morel, "Ensuring visibility in calibration-free path planning for image-based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 22, no. 4, pp. 848–854, 2006.

[6] S. Durola, P. Dan, D. F. Coutinho, and M. Courdesses, "Rational systems and matrix inequalities to the multicriteria analysis of visual servos," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '09)*, pp. 1504–1509, IEEE, Kobe, Japan, May 2009.

[7] P. Danés and D. Bellot, "Towards an LMI approach to multicriteria visual servoing in robotics," *European Journal of Control*, vol. 12, no. 1, pp. 86–110, 2006.

[8] F. Schramm and G. Morel, "Ensuring visibility in calibration-free path planning for image-based visual servoing," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 848–854, 2006.

[9] G. Chesi, "Visual servoing path planning via homogeneous forms and LMI optimizations," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 281–291, 2009.

[10] Y. Mezouar and F. Chaumette, "Optimal camera trajectory with image-based control," *International Journal of Robotics Research*, vol. 22, no. 10-11, pp. 781–803, 2003.

[11] M. Kazemi, K. Gupta, and M. Mehrandezh, "Global path planning for robust visual servoing in complex environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '09)*, pp. 326–332, Kobe, Japan, May 2009.

[12] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 933–939, 2010.

[13] J. A. Gangloff and M. F. De Mathelin, "Visual servoing of a 6-DOF manipulator for unknown 3-D profile following," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 511–520, 2002.

[14] R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M. M. A. Sanchez, and J. Marescaux, "Active filtering of physiological motion in robotized surgery using predictive control," *IEEE Transactions on Robotics*, vol. 21, no. 1, pp. 67–79, 2005.

[15] G. Allibert and E. Courtial, "What can prediction bring to image-based visual servoing?" in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, pp. 5210–5215, St. Louis, Mo, USA, October 2009.

[16] G. Allibert, E. Courtial, and Y. Touré, "Visual predictive control for manipulators with catadioptric camera," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA' 08)*, pp. 510–515, Pasadena, Calif, USA, May 2008.

[17] G. Allibert, E. Courtial, and Y. Touré, "Real-time visual predictive controller for image-based trajectory tracking of mobile robot," in *Proceedings of the 17th IFAC World Congress*, Seoul, Republic of Korea, July 2008.

[18] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley, Chichester, UK, 1966.

[19] K. De Jong, *Analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, University of Michigan, Ann Arbor, Mich, USA, 1975.

[20] J. R. Koza, "Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems," Report No. STAN-CS-90-1314, Stanford University, Stanford, Calif, USA, 1990.

[21] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[22] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, Boston, Mass, USA, 1989.

[23] L. N. de Castro and F. J. bon Zuben, "Artificial immune systems: part I—basic theory and applications," Tech. Rep. TR-DCA 01/99, 1999.

[24] R. Storn and K. Price, "Differential evolution—a simple and efficient adaptive scheme for global optimisation over continuous spaces," Tech. Rep. TR-95–012, ICSI, Berkeley, Calif, USA, 1995.

[25] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[26] Ş. İ. Birbil and S.-C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.

[27] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Filter modeling using gravitational search algorithm," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 117–122, 2011.

[28] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.

[29] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive feedback as a search strategy," Tech. Rep. 91-016, Politecnico di Milano, 1991.

[30] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, Berlin, Germany, 2010.

[31] M. Morari and E. Zafiriou, *Robust Process Control*, Prentice-Hall, 1989.

[32] L. Wang, *Model Predictive Control System Design and Implementation using MATLAB*, Springer, 2009.

[33] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, Springer Tracts in Advanced Robotics, 1st edition, 2011.

[34] F. Dunn and I. Parberry, *3D Math Primer for Graphics and Game Development*, A K Peters, CRC Press, 2nd edition, 2011.