Event-based visual servoing with features' prediction

G. J. Garcia, J. Pomares, F. Torres and P. Gil

Physics, Systems Engineering and Signal Theory Department, University of Alicante, San Vicente del Raspeig, Spain

{gjgg, jpomares, Fernando.Torres, Pablo.Gil}@ua.es

Abstract. Event-based visual servoing is a recently presented approach that performs the positioning of a robot using visual information only when it is required. From the basis of the classical image-based visual servoing control law, the scheme proposed in this paper can reduce the processing time at each loop iteration in some specific conditions. The proposed control method enters in action when an event deactivates the classical image-based controller (i.e. when there is no image available to perform the tracking of the visual features). A virtual camera is then moved through a straight line path towards the desired position. The virtual path used to guide the robot improves the behavior of the previous event-based visual servoing proposal.

Keywords: Visual Servoing, Event-based Control, Event-trigger, Visual Robot Control, Path planning.

1 Introduction

Currently, image-based control is a well-known approach to guide a robot using visual information [3]. These conventional approaches employ frame-based image acquisition and processing technologies in order to continuously obtain the image, extract features and apply the visual servoing controller. In general, the use of this kind of technologies is not computationally efficient because they do not take advantage of the dynamic characteristics of visual scenes. The constant image information processing does not stop even when nothing relevant occurs. In opposite with the previous approach, the event-based visual systems allows to increase the system performance [4]. In this paper, an event-based visual servoing approach is proposed which permits to reduce the image data stream using the event-based control theory [1]. In this case, an event is considered as something that occurs which requires some response. Therefore, it is only necessary to obtain and process image information when an event is generated. Event-based control has been applied to many fields, as in [12] where this strategy is used to control the level of a water tank. A method approached to the evaluation of optical flow using an asynchronous event-based acquisition is developed in [2]. From a pair of event-based cameras, in [11] is described an event-based stereo matching algorithm exploiting the asynchronous visual events. Recently, these cameras have been used in microrobotic applications [9]. In these works Ni et al. introduce an event-based

iterative closest point algorithm to track a microgripper's position at a high rate frequency. These dynamic vision sensors (eDVS) have also been used to track angular 2D coordinates frame to guide and operate in real-time autonomous mobile robots [8].

As it is previously indicated, the visual controller only is applied when an event is detected. Therefore, a behavior must be indicated when an event is not detected in order to continuously guide the robot. To do this, an approach based on virtual visual servoing [10] is proposed. Using virtual visual servoing systems the camera parameters can be estimated iteratively. This is done so that the extracted visual features correspond to the same features computed by the projection of the 3D model according to the current camera parameters. This approach is employed to determine the 3D position of the camera when an event is generated. Therefore, when an event is not obtained the last 3D camera position is known. This information is used to propose a new path planning algorithm which allows the robot guidance in the image virtual space when no real image information is obtained.

The main contribution of the paper is the use of an event-based approach to guide a robot using image information. Furthermore, a path planning algorithm is integrated in order to determine the desired behaviour in the virtual image space when no real image information is obtained. This virtual information is employed in order to guide the robot when events are not generated.

The paper is structured as follows: first, the basics of image-based visual servoing are detailed; the event-based controller using visual servoing is described in Section 3; Section 4 presents an improvement of this event-based visual servoing consisting on features' prediction, Section 5 presents different experiments to validate the proposal; and finally, in Section 6 the main conclusions are discussed.

2 Image-based visual servoing

Image-based visual servoing is based in minimizing the error between current and desired features on the image plane. The image acquired by the camera is the only information needed to obtain that error. The image function describing the task can be represented by $\mathbf{e}_t = \mathbf{s} - \mathbf{s}^*$, where $\mathbf{s}$ is an M x 1 vector containing M visual features corresponding to the current state, while $\mathbf{s}^*$ denotes the visual features values in the desired state.

In order to relate the variations in the image to the variations in the camera, the interaction matrix, $\mathbf{L}_s$, is employed, $\dot{\mathbf{s}} = \mathbf{L}_s\dot{\mathbf{r}}$ [6], where $\dot{\mathbf{r}}$ indicates the camera velocity.

The control law of classical image-based visual servoing is obtained by imposing an exponential decrease of $\mathbf{e}_t$ ($\dot{\mathbf{e}}_t = -\lambda\mathbf{e}_t$):

$$\mathbf{v}_c = -\lambda\hat{\mathbf{L}}_s^+\left(\mathbf{s}-\mathbf{s}^*\right) \tag{1}$$

where $\widehat{\mathbf{L}_s^+}$ is the pseudoinverse of an approximation of the interaction matrix and λ is the proportional control gain.

Image-based visual servoing can be schematized as in Fig. 1. This scheme shows how the image acquired by the camera located at the robot end-effector closes the control loop. This image provides the new position of the visual features, which are then

compared to the reference, consisting in the visual features position at the desired configuration. The error computed in this way is then used in the control law shown in (1). The output of the controller is a robot end-effector velocity in the Cartesian 3D space. This velocity is performed by the robot's internal controller, which computes the joint configuration to assure the desired end-effector velocity.

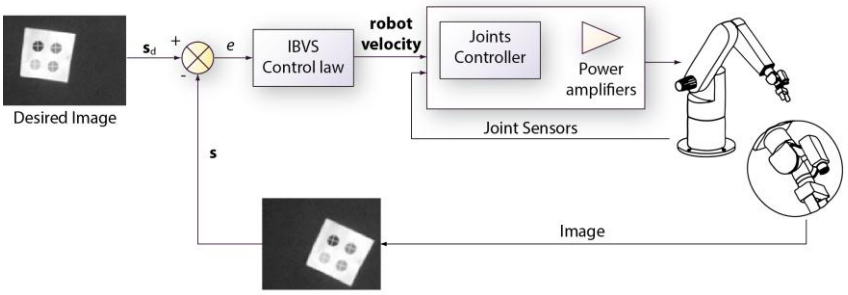

Fig. 1. Classical image-based visual servoing scheme.

3 Event-Based Visual servoing

There are several reasons to modify this scheme in order to improve the image-based visual servoing performance using the event-based control theory. As stated before, an event-based strategy can be employed in order to reduce the transfer of data between camera and controller. Thus, there will be an image transmission only when something relevant occurs (e.g. visual features are near the desired position or in a region where it can be loosen going out the image plane). In addition, large images from modern high-res and high-speed cameras increase considerably the image processing time necessary to segment the objects and obtain the position of the visual features. Finally, events can prevent a visual feature from going out of the field of vision, which may spoil the positioning task.

The event-based visual servoing proposed in [5] is shown in Fig. 2. The main modifications over the scheme of a classical image-based visual servoing are related to the event generator and the event detector blocks.

Event generator is a module that produces an event when any of the visual features enter into a specific region of the image. An event-based camera has been emulated in the researches described in this paper. Thus, image is acquired from a standard camera, but after processing the image in a computer (the cpu is not embedded into the sensor), the Event generator module produces an event every time a visual feature touches any of the predefined image regions. This emulation is required in order to validate the proposed scheme. The image regions are defined so that the visual features cannot leave the field of view. Fig. 3 shows the different regions in the image space and the lines that set the event trigger.

Once the event has been triggered, the Event detector module that can be seen in Fig. 2 must determine the way in which the system will compute the robot's velocity. Fig. 3 depicts the decision scheme of Event detector. If any of the pixel coordinates of the visual features is in the green region marked as ON in Figure 3, the event-based

controller activates the classical image-based visual sevoing to compute the robot end-effector velocity. This velocity is then stored in order to be used if the Event detector determines that all the visual features in the red region are marked as OFF in Figure 3. Image-based visual servoing must be activated during the first iteration of the positioning task wherever the visual features are in the image (i.e., even though an event is triggered in this first iteration).

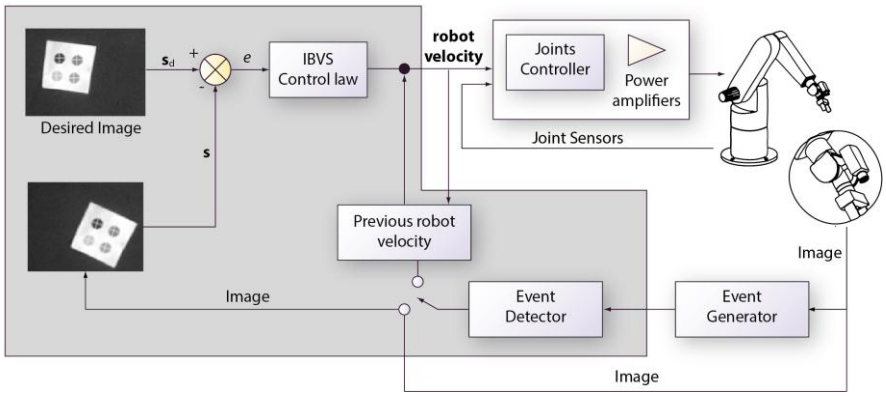

Fig. 2. Event-based visual servoing scheme without features' prediction.

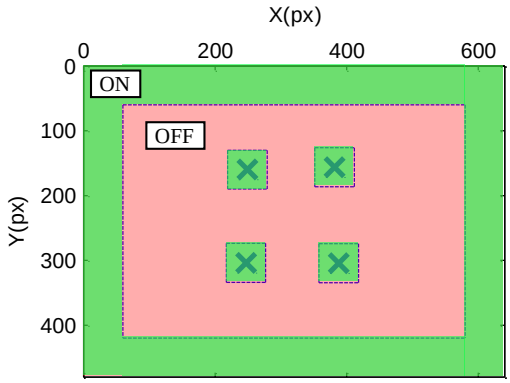

Fig. 3. Event generator regions and Event detector conditions into image space.

4 Features' Prediction with Virtual Visual Servoing

The scheme shown in Fig. 2 uses a very simple method to continue guiding the robot when the features enter into the OFF zone depicted in Fig. 3. This scheme proposes to deactivate the image-based visual servoing controller (as in this zone there is no image available) and maintain the last velocity computed in the previous loop iteration. In

order to improve the behavior of the controller at this point, in this Section a new path planning algorithm is proposed. This approach is based on virtual visual servoing.

4.1 Virtual Visual Servoing

Virtual visual servoing is based on the classical image-based visual servoing scheme described in Section 2. In virtual visual servoing, a virtual camera is positioned in the scene. Virtual visual servoing moves iteratively this virtual camera in order to obtain the location of the virtual camera from which the object is observed at the same position as in a reference image acquired by the real camera. In order to define the virtual visual servoing systems, the observed features in the reference image are denoted by $\mathbf{p}_d$, and $\mathbf{p}$ are the current positions of the image features projected using the camera intrinsic parameters, ξ, and the current extrinsic parameters, ${}^c\mathbf{M}_o$ (pose of the object frame with respect to the camera frame).

In order to determine the camera extrinsic parameters, it is necessary to minimize iteratively the error, $\mathbf{e}$, between the observed data, $\mathbf{p}_d$, and the position of the same features $\mathbf{p}$ computed using the following equation:

$$\mathbf{p}=pr_{\xi}\left({}^{c}\mathbf{M}_{o}\,{}^{o}\mathbf{P}\right) \tag{2}$$

where ${}^o\mathbf{P}$ are the 3D coordinates (in the object coordinate frame) of the points extracted by the camera and pr_ξ denotes the perspective projection model according to the intrinsic parameters. From the previous defined error, $\mathbf{e}$, it is possible to obtain (the camera intrinsic parameters do not vary. They can be obtained using an off-line calibration [14]):

$$\dot{\mathbf{e}} = \dot{\mathbf{p}} - \dot{\mathbf{p}}_d = \frac{\partial \mathbf{p}}{\partial \mathbf{r}} \frac{d\mathbf{r}}{dt} \tag{3}$$

where $\mathbf{r}$ is the camera pose. Equation (3) can be rewritten as:

$$\dot{\mathbf{e}} = \mathbf{L}_p \mathbf{v} \tag{4}$$

where $\mathbf{v}$ is the instantaneous virtual camera velocity and $\mathbf{L}_p$ the interaction matrix related to $\mathbf{p}$ [7].

In order to make $\mathbf{e}$ decrease exponentially to 0 ($\dot{\mathbf{e}} = -\lambda_1 \mathbf{e}$) the following control law is obtained:

$$\mathbf{v} = -\lambda_1 \mathbf{L}_p^{+} \mathbf{e} \tag{5}$$

When $\mathbf{e} = 0$, the extrinsic camera parameters are obtained.

4.2 Features' prediction

After performing a virtual visual servoing task, the pose of the camera in the Cartesian 3D space is obtained. Through this camera pose and the desired one, a straight line

path for the camera can be planned. The tracking of this path assures that the robot arrives at the final desired position. In order to track the path, a simple time-dependent method is proposed:

$$\mathbf{v}_c = -\lambda \hat{\mathbf{L}}_s^+ \left(\mathbf{s}(t) - \mathbf{s}^* \right) \tag{6}$$

The path of the camera is computed in the Cartesian 3D space because computing it in the image space is not possible (a path planned for a particular feature may not be coherent with the path obtained for another). The problem addressed in this section is to determine the 3D location ${}^c\mathbf{M}_{ok}$, which represents the extrinsic camera parameters at iteration k of the straight path planned. Fig. 4 summarizes the algorithm proposed to obtain this 3D location.

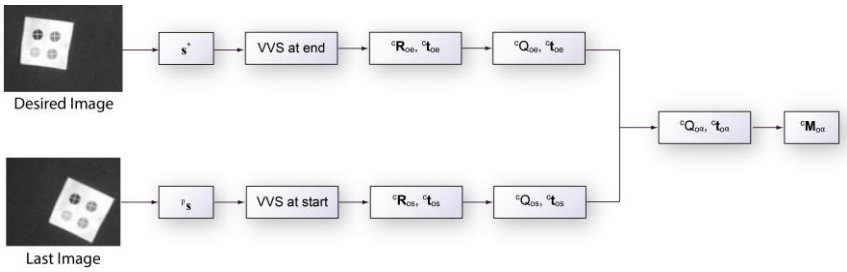

Fig. 4. Scheme of the algorithm to predict the camera pose in the blind zone

The camera extrinsic parameters ${}^c\mathbf{M}_{os}$ and ${}^c\mathbf{M}_{oe}$ are obtained from the set of features ${}^p\mathbf{s}$ and $\mathbf{s}^*$ respectively by using virtual visual servoing (see section 4.1). ${}^p\mathbf{s}$ is the set of visual features at the last image obtained from the camera. This last image is obtained before the camera launches the event where the features enter the OFF zone depicted in Fig. 3.

The translations ${}^c\mathbf{t}_{os}$ and ${}^c\mathbf{t}_{oe}$ can be easily extracted from ${}^c\mathbf{M}_{os}$ and ${}^c\mathbf{M}_{oe}$. A linear interpolation between both translations can be obtained using the following equation:

$${}^c\mathbf{t}_{o\alpha} = {}^c\mathbf{t}_{oe} + \alpha \cdot \left({}^c\mathbf{t}_{os} - {}^c\mathbf{t}_{oe} \right) \tag{7}$$

with $0 \le \alpha \le 1$. Furthermore, the rotations ${}^c\mathbf{R}_{os}$ and ${}^c\mathbf{R}_{oe}$ can also be extracted from ${}^c\mathbf{M}_{os}$ and ${}^c\mathbf{M}_{oe}$. In order to develop the linear interpolation for the orientation, it is necessary to represent the previous rotations by the quaternions ${}^c\mathbf{Q}_{os}$ and ${}^c\mathbf{Q}_{oe}$. Spherical linear interpolation [13] is used to interpolate the orientation, ${}^c\mathbf{Q}_{o\alpha}$, between each pair of quaternions:

$${}^c\mathbf{Q}_{o\alpha} = \frac{{}^c\mathbf{Q}_{oe} . \sin\left((1-\alpha)\theta \right) + {}^c\mathbf{Q}_{os} . \sin\left(\alpha\theta \right)}{\sin\theta} \tag{8}$$

where θ is the angle between the orientation of the camera in the end position and the orientation in the start pose of the camera obtained by virtual visual servoing through

${}^{p}\mathbf{s}$. After the computation of ${}^{c}\mathbf{Q}_{o\alpha}$, the rotation matrix ${}^{c}\mathbf{R}_{o\alpha}$ is obtained by transforming the quaternion.

When the linear interpolation ${}^{c}\mathbf{M}_{o}(\alpha) = [{}^{c}\mathbf{R}_{o\alpha}\ {}^{c}\mathbf{t}_{o\alpha}]$ is generated, the location ${}^{c}\mathbf{M}_{ok}$ is obtained by iterating over this linear interpolation. In order to achieve a natural behavior of the visual controller this location is achieved as a function of time using an exponential distribution:

$$\alpha = 1 - e^{-t/\beta} \tag{9}$$

where β is a constant that can be used to increase the slope of the exponential distribution. Exponential distribution represents better the natural performance of a classic image-based visual servoing system than a linear distribution. After computing ${}^{c}\mathbf{M}_{ok}$ from the current value of t, the visual features predicted can be computed from:

$$\mathbf{s}(t) = \mathrm{s}(\alpha) = pr_{\xi}\left({}^{c}\mathbf{M}_{o}(\alpha)\,{}^{o}\mathbf{P}\right) \tag{10}$$

where ${}^{o}\mathbf{P}$ is the set of object points considered and pr_{ξ} denotes the perspective projection model according to the intrinsic parameters, ξ.

The algorithm proposed in this Section is schematized in Fig. 5. The Event detector described in Section 3 activates the prediction module when there is no image available. The virtual information is then used to improve the controller performance.

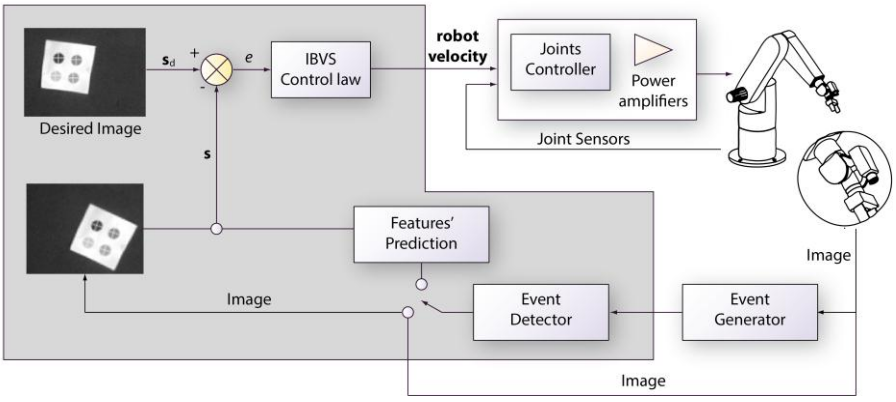

Fig. 5. Event-based visual servoing scheme with features' prediction.

5 Results

Two different experiments have been developed using both the event-based visual servoing described in Fig.2 and the new proposal depicted in Fig 3. The testing platform consists in a webcam placed at the end-effector of a Mitsubishi PA-10. The PA-10 is a robot manipulator of 7 degrees of freedom. The webcam employed can acquire images at an image resolution of 640x480 px.

Image processing issues are not the goal of this work, so, in order to guide the robot in these experiments, four visual features, **s,** (centroid points from very easily segmentable circular marks) have been used. The webcam does not allow performing a quick image acquisition. However, image acquisition is not a crucial parameter in the proposed controller validation.

Regions depicted in Fig. 3, can be adjusted to any experiment. For both the two experiments shown in this Section, the external ON region has been defined of 60 px width. In addition, the four internal ON regions are obtained from the image position of the points at the desired robot's pose. The square internal region is centered at the desired visual features location and are 60 px width.

5.1 Experiment without features' prediction

The first experiment starts with one of the four points in the external ON region. The set of desired visual features is $\mathbf{s}_d$=[249, 156, 382, 156, 388, 305, 247, 304]. With a gain of λ=0.1, Fig. 6 shows the evolution of the robot's end-effector during the positioning task. The blue cross sets the desired position. The event-based visual servoing proposed in [5] is a valid scheme to position the robot. However, this experiment demonstrates that deactivating visual controller when visual features enter the OFF zone cannot assure direct convergence of the positioning. In this experiment the Event generator triggers six events. The evolution of the end-effector presents three different changes of direction. These changes match the main event triggered during the positioning task.

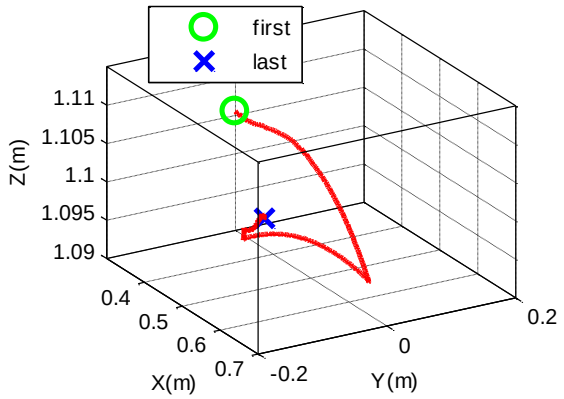

Fig. 6. Experiment without features' prediction. 3D trajectory of the robot end-effector.

Robot evolution in the Cartesian 3D space is not the best measure factor for the proposed controller. Image-based visual servoing systems' input is the image. Therefore, the best measure factor for these kinds of controllers is the evolution of the features in the image. Fig. 7 shows this evolution for this first experiment. The change in the evolution of the visual features corresponds to an event. The test begins with a visual feature in the external ON area. An event is launched when this feature leaves this external region and enters the OFF region (E1). The Event detector deactivates the visual

servoing controller and the last velocity is maintained until the next event occurs. Event 2 (E2) is triggered when the bottom right visual features enters the 60 px security bottom area. The Event detector activates the visual servoing controller and a new velocity is sent to the robot. This velocity minimizes the image error, pushing the visual feature away from the bottom edge towards its desired position. The event 3 (E3) is triggered once this point enters again into the OFF region. Although the last velocity computed by the visual servoing controller pushed the visual features towards the desired position, the lack of new images prevents the correction of the velocity to achieve the desired position. Thus, event 4 (E4) is triggered when the top right visual feature enters into the top ON region. A new velocity is computed and the event 5 (E5) deactivates the visual controller. This last velocity guides the features towards the internal square ON areas. The last event (E6) is triggered when one of the visual features enters into its square internal ON area, and after this, the Event detector activates the visual servoing to guide the robot towards the desired position.

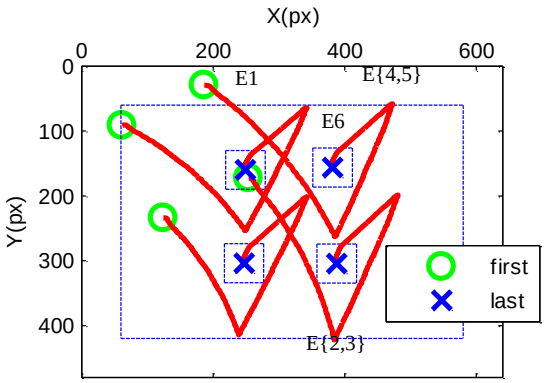

Fig. 7. Experiment without features' prediction. Evolution of the features in the image.

The six events triggered during the experiment deactivate the visual servoing controller three times. During the deactivation (the visual features are all inside the OFF region), the velocity is constant. Figure 8 illustrates the lineal and angular velocity of the end-effector during the task. Figure 9 shows a zoom over time regarding the lineal velocity depicted in Figure 8.a. Over Figure 9, different instants have been shaded in green or red.

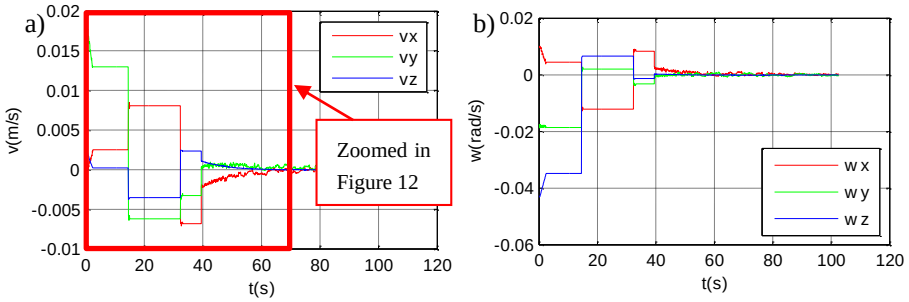

Fig. 8. Experiment without features' prediction. Velocity of the end-effector: a) lineal velocity, b) rotational velocity.

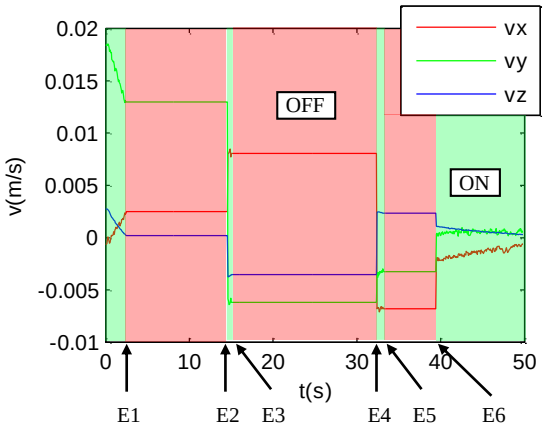

Fig. 9. Experiment without features' prediction. Zoom in lineal velocity of the end-effector. Event detector decisions.

Furthermore, the six events marked in Figure 7 over the evolution of the features in the image have been also marked in Figure 9. A green shaded area represents the lineal velocity sent to the robot when the Event detector activates the visual servoing controller. The velocity in this case presents an exponential decrease towards zero. This is the typical performance of the output of classic image-based visual servoing. Red shaded zones denote the velocity sent to the robot when the Event detector deactivates the visual servoing. When this occurs, the event-based visual servoing sends to the robot the last velocity calculated by the visual servoing. This is the reason why in the red shaded zones there is constant velocity.

5.2 Experiment with features' prediction

This test uses the proposed algorithm presented in Section 4 to adjust the velocity of the robot when the visual features enter the OFF zone and, thus, there is no image to compute the current features. The previous experiment has demonstrated the necessity of computing the velocity in this blind zone. The set of desired visual features is $\mathbf{s}_d$=[249, 162, 381, 156, 389, 303, 247, 310]. With a gain of λ=0.1, Fig. 10 shows the evolution of the robot's end-effector during the positioning task. The camera tracks a straight trajectory when there is no real image to compute real visual features. The experiment shows that the positioning is more robust than the obtained in the previous experiment as the robot converges towards the 60 px width zone of the desired features.

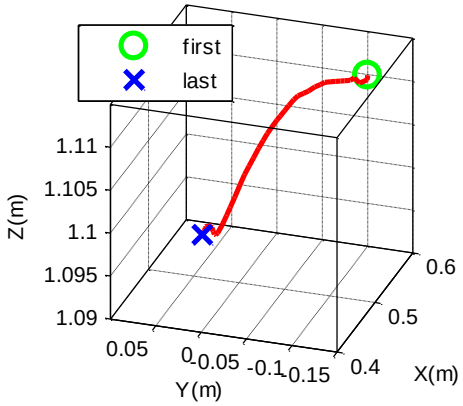

Fig. 10. Experiment with features' prediction. 3D trajectory of the robot end-effector.

Fig. 11 shows the evolution of the features in the image for this experiment. There is not a clear change in the evolution of the visual features. Only two events are triggered in this experiment. The initial position of the robot presents one of the visual features in the external ON area. Thus, the Event detector described in Section 3 activates the visual servoing controller in order to obtain a velocity that approaches the robot to the desired position. In addition, this velocity pushes this visual feature away from the image edges, avoiding thus losing it. The first event (E1) is triggered when the last visual feature leaves the external ON region. The Event detector activates the visual features' prediction, obtaining a straight path from the position of the last image acquired in the previous iteration and the desired camera position. Using this path, future iterations in the blind zone compute different camera poses obtained following the scheme proposed in Section 4. Once the camera pose is obtained the set of visual features is computed by using the intrinsic camera parameters. Finally, with this set of visual features the velocity sent to the robot is computed using the same image-based visual servoing used in the ON zone. This velocity is supplied to the robot. The second event (E2) is launched when one of the features enters its internal square ON area. This square region is centered at the desired feature position. The Event detector activates then the visual servoing controller and a new velocity obtained newly from a real image is sent to the robot. The new velocity sent to the robot attracts it towards the desired position.

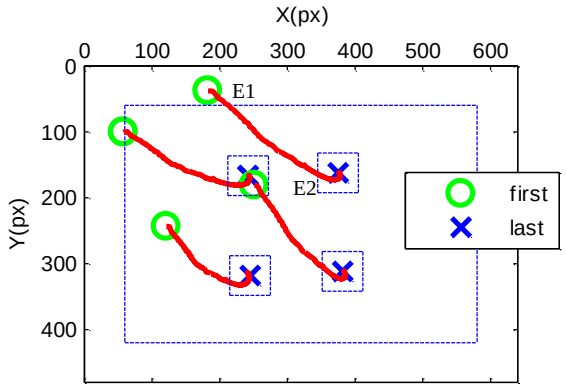

Fig. 11. Experiment with features' prediction. Evolution of the features in the image.

The behavior of the method proposed in this paper to obtain virtual features to guide the robot in the blind zone is very similar to a classical image-based visual servoing scheme. The main advantage of using the proposed scheme is that there is no image to process when visual features enter the blind zone. The time saved during this phase can be spent in any other task.

6 Conclusions

Event-based visual servoing is a control scheme to guide a robot from any initial pose to the desired one using images only when it is necessary. It not only reduces the transfer load between the camera and the processing unit but also reduces the processing time during the visual servoing task. Moreover, this controller prevents visual features from leaving the field of view, which would cause the failure of the task.

The system proposed in this paper improves the robustness of the event-based visual servoing. The Event detector of the previous proposal can only switch between visual servoing and the last visual servoing calculated velocity. In order to avoid situations like that of the first experiment, where visual features bounce more than one time over the external security area, an image position estimator based on virtual visual servoing has been used. Thus, the time saved during the deactivation of the visual servoing can be spent on estimating the set of visual features **s**. This estimation pushes the visual features towards the desired position as if the controller had a real image.

The works presented in this paper opened a new researching topic. After validating the proposed controller using a standard webcam, the next step is the use of an event-based camera in order to quantify the improvement in terms of data transfer load between camera and computer.

The controller proposed allows avoiding outliers using a simple strategy. Events can be used in the future to perform any change in the parameters of the visual servoing controller. An event can order the system to increase the image resolution to perform a

more precise feature tracking, or to decrease the image resolution to save the data transfer bandwidth. An event can also trigger a visual feature change, using points, lines or ellipses depending on the different events.

Acknowledgements

The research leading to these results has received funding from the Spanish Ministry of Education and Science and European FEDER funds, the Valencia Regional Government and the Research and Innovation Vice-president Office of the University of Alicante, through the research projects DPI2012-32390, GV2012/102 and PROMETEO/2013/085, GRE12-17, respectively.

References

1. Aström, K. J., 2008. Event Based Control. In *Analysis and Design of Nonlinear Control Systems*, 127-147. Springer Berlin Heidelberg.
2. Benosman, R., Ieng, S.-H., Clercq, C., Bartolozzi, C., Srinivasan, M., 2012. Asynchronous frameless event-based optical flow. *Neural Networks*, *27*(0), 32-37.
3. Chaumette, F., Hutchinson, S. A., 2006. Visual servo control. I. Basic approaches. *Robotics & Automation Magazine, IEEE*, *13*(4), 82-90.
4. Delbrück, T., Linares-Barranco, B., Culurciello, E., Posch, C., 2010. Activity-driven, event-based vision sensors. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2426-2429.
5. García, G.J., Pomares, J., Torres, F., Gil, P., 2013. Event-based visual servoing. In *International Conference on Informatics in Control (ICINCO)*.
6. Hutchinson, S., Hager, G. D., Corke, P. I., 1996. A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on*, *12*(5), 651-670.
7. Marchand, E., Chaumette, F. (2001). A new formulation for non-linear camera calibration using VVS. *Publication Interne 1366, IRISA*, Rennes, France.
8. Müller, G.R., Conradt, J., 2012. Self-Calibrating Marker Tracking in 3D with Event-Based Vision Sensors. In *Proceedings of Artificial Neural Networks and Machine Learning*, Lectures Notes in Computer Science, 7552, 313-321.
9. Ni, Z., Bolopion, A., Agnus, J., Benosman, R., Regnier, S., 2012. Asynchronous Event-Based Visual Shape Tracking for Stable Haptic Feedback in Microrobotics. *Robotics, IEEE Transactions on*, *28*(5), 1081-1089.
10. Pomares, J., Chaumette, F., Torres, F., 2007. Adaptative Visual Servoing by Simultaneous Camera Calibration. In *ICRA'07* 2811-2816.
11. Rogister, P., Benosman, R., Ieng, S.-H., Lichtsteiner, P., Delbrück, T., 2012. Asynchronous Event-Based Binocular Stereo Matching. *Neural Networks and Learning Systems, IEEE Transactions on*, *23*(2), 347-353.
12. Sánchez, J., Guarnes, M. A., Dormido, S., 2009. On the application of different event-based sampling strategies to the control of a zsimple industrial process, *Sensors*, *9*, 6795-6818.
13. Shoemake, K., 1985. Animating Rotation with Quaternion Curves. *Computer Graphics 19*(3), 245-254.
14. Zhang, Z., 1999. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the International Conference on Computer Vision*, 1, 666-673.