Event-based Visual Servoing

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

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

Abstract: Traditional visual servoing systems have been widely studied in the last years. These systems control the position of the camera attached to the robot end-effector guiding it from any position to the desired one. These controllers can be improved by using the event-based control paradigm. The system proposed in this paper is based on the idea of activating the visual controller only when something significant has occurred in the system (e.g. when any visual feature can be loosen because it is going outside the frame). Different event triggers have been defined in the image space in order to activate or deactivate the visual controller. The tests implemented to validate the proposal have proved that this new scheme avoids visual features to go out of the image whereas the system complexity is reduced considerably. Events can be used in the future to change different parameters of the visual servoing systems.

1 INTRODUCTION

Intuitively, the best human sense that makes a person change from any position to another is the sense of sight. The use of cameras as the only sensor to guide a robot has derived into a great amount of works encompassed under the term of visual servoing (Hutchinson, 1996). Visual servoing describes a technique that controls the camera spatial velocity through the on-line acquired images. Image-based visual servoing makes only use of the existing visual information in the image acquired by the camera at each iteration (Chaumette, 2006). Image resolution provided by modern cameras is increasingly growing. Nowadays, this increasing resolution offers a more accurate vision of the scene. Higher resolution means more detail. More detail means better chances of scene segmentation. However, higher resolution also implies greater bandwidth requirements, for transmission between sensor and cpu for processing. High speed cameras aggravate this problem, requiring a large amount of data transferred from the camera to the computer's input device. Off-line processing has been a solution for this high bandwidth requirements (Keen, 2007; Shu, 2010). Modern cameras permit to solve this issue by reducing the acquired area (also known as region of interest). Thus, the amount of data can be reduced allowing an on-line image processing, but reducing the field of view (Nasibov, 2012). If the bus is capable of transmitting the full frame at a high rate, another issue presented that can influence in the system performance is the entire frame on-line processing which remains time consuming.

To solve these problems, cameras with on-chip processing, such as FPGA or smart cameras, can be used as seen in (Li, 2012; Elouardi, 2009). However, this solution calls for extra expenses, development cycles and the requirement of specific hardware knowledge.

Image-based visual servoing requires a continuous image transfer between the camera and the computer in order to compute the robot velocity. This constant data streaming does not stop even when nothing relevant occurs. In this paper, this data stream is reduced by using the event-based control theory (Aström, 2008). An event is something that occurs requiring some response. The basic idea is to communicate, compute, or control only when something significant has occurred in the system. Event-based control has been applied to many fields, as in (Sanchez, 2009) where this strategy is used to control the level of a water tank.

Event-based cameras can provide with relevant information about the scene just when something important occurs (e.g. when the features used for tracking can be losen because they are going outside visual field of camera). Although the first work using a vision system with these features was carried about two decades ago (Mahowald, 1992), it is now when the performance of the address-event representation vision systems has motivated their use in the research community (Dellbrück, 2010). A method approached to the evaluation of optical flow using an asynchronous event-based acquisition is developed in (Benosman, 2012). From a pair of event-based cameras, in (Rogister, 2012) is described an event-based stereo matching algorithm exploiting the asynchronous visual events. Recently, these cameras have been used in microrobotic applications (Ni, 2012b). 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 (Müller, 2012).

The main contribution presented in this paper is the extension of event-based control to visual servoing systems. Other works like (Ni, 2012a) have used the event-based paradigm to track micro-particles simply by means of a see and move strategy to track and operate mobile robots as in (Müller, 2012). Control loop has not visual feedback. Our proposal is to use the events provided by the vision system to control the robot velocity. The robot is guided from one position to the desired one using only the relevant frames for such a task. The experiments developed validate this first approach of fusing event-based control and visual servoing systems.

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

2 IMAGE-BASED VISUAL SERVOING

This Section describes the fundamentals of visual servoing systems. Image-based visual servoing inputs are the images acquired by a camera. A visual servoing task can be described by an image function, $\mathbf{e}_t = \mathbf{s} - \mathbf{s}_d$, which must be regulated to 0, where $\mathbf{s}$ is an M x 1 vector containing M visual features corresponding to the current state, while $\mathbf{s}_d$ 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, $\dot{\mathbf{s}} = \mathbf{L}_s\dot{\mathbf{r}}$ (Hutchinson, 1996), 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 $\mathbf{e}_t$ ($\dot{\mathbf{e}}_t = -\lambda\mathbf{e}_t$):

$$\mathbf{v}_c = -\lambda\widehat{\mathbf{L}_s^+}(\mathbf{s} - \mathbf{s}_d) \qquad (1)$$

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

When the visual feature is a point in the image, the interaction matrix can be obtained from (Chaumette, 2006). This matrix depends on the intrinsic camera parameters. These intrinsic parameters can be obtained with high precision through a calibration process performed in an off-line step of the visual servoing task (Zhang, 1999). The different visual features are directly measured in pixels from the image acquired by the camera ((f_x, f_y) for a point). The interaction matrix also depends on the depth between the camera reference system and the 3D point represented in the image by (f_x, f_y). This is the only parameter that cannot be computed directly from the image. In (Malis, 2003) it is developed a study of the depth estimation influence in the robustness of the visual servoing scheme. The main conclusion drawn of this study is that the system is stable when using the depth from the camera to each visual feature measured at the desired robot position. However, using these depths, the evolution of the visual features in the image do not draw a straight line between the initial and final position. Parabolic trajectory of a visual feature in the image may lead the feature to leave the frame, avoiding the completion of the positioning task. The method proposed in the next Section avoids this situation by preventing any visual feature to pass through a security zone.

3 EVENT-BASED VISUAL SERVOING

Image-based visual servoing described in Section 2 can be schematized as in Figure 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.

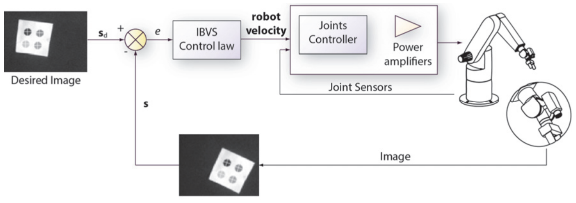

Figure 1: Classical image-based visual servoing scheme.

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 stop a visual feature from going out of the field of vision, which may spoil the positioning task.

The event-based visual servoing proposed is shown in Figure 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. Figure 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 Figure 2 must determine the way in which the system will compute the robot's velocity. Figure 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 proposed 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).

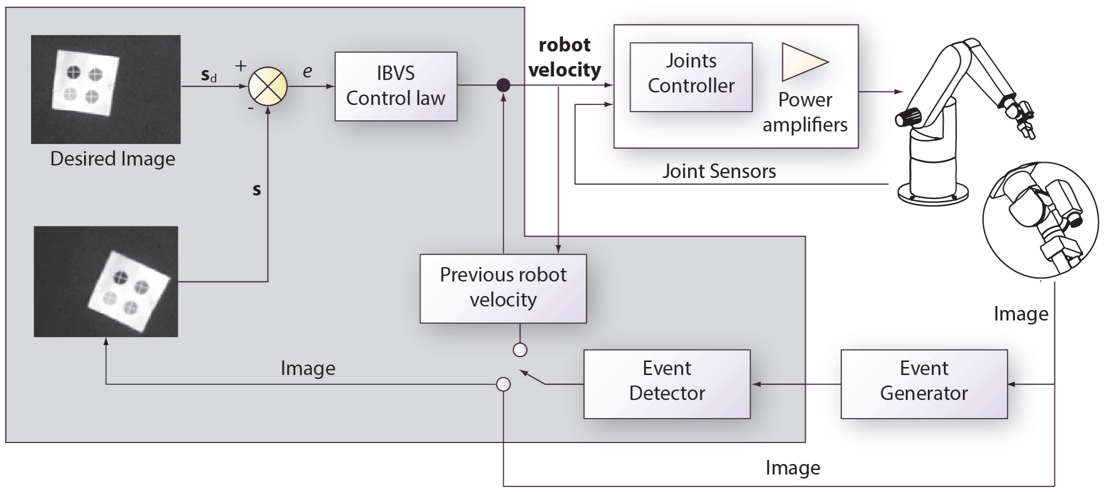

Figure 2: Event-based visual servoing scheme.

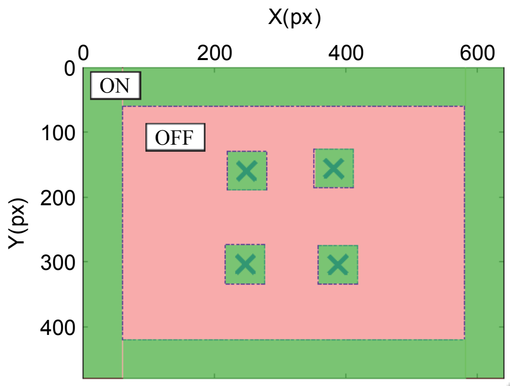

Figure 3: Event generator regions and Event detector conditions into image space.

4 RESULTS

In order to validate the new event-based visual servoing scheme proposed in this paper, two different positioning tasks are shown in this Section. 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. The experiments detailed in this Section prove that event-based visual servoing excludes the possibility of losing any visual feature due to the use of an estimation of the feature-camera distance. The tests also prove that the time elapsed in the computation of the robot's velocity can be considerably reduced during the task with the new proposed controller. This time saving reduces the processor load, and releases it so that it can attend other useful tasks.

Regions depicted in Figure 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.

4.1 Experiment 1

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, 160, 382, 156, 388, 305, 246, 311]. With a gain of λ=0.1, Figure 4 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 the paper is a valid scheme to position the robot.

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. Figure 5 shows this evolution for this first experiment. The change in the evolution of the visual features corresponds to an event. This event is triggered because one of the visual features enters in the 60 width area of its desired position.

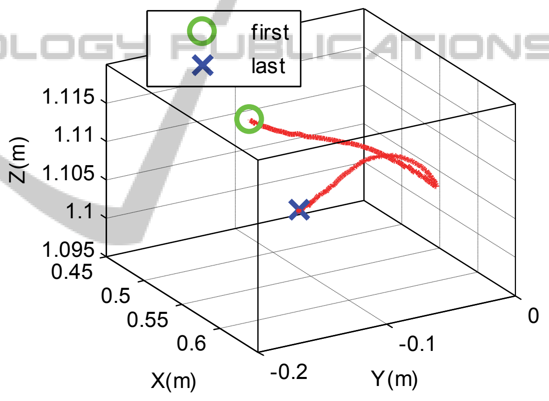

Figure 4: Experiment 1. 3D trajectory of the robot end-effector.

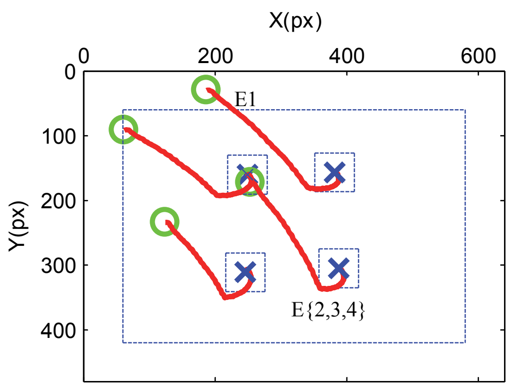

Figure 5: Experiment 1. Evolution of the features in the image.

Figure 6 illustrates the evolution of the controller output: the robot's end-effector velocity. In this first experiment only four events have been triggered. 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 deactivates the visual servoing controller, sending to the robot the last velocity computed by the visual servoing. This velocity is supplied to the robot until another event occurs. 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 is sent to the robot. The new velocity sent to the robot attracts it towards the desired position. The third event (E3) is triggered when visual features go for an instant out of the internal square regions. Finally, the visual features enter definitely into the ON area (E4) and the visual servoing guides the robot to the final pose.

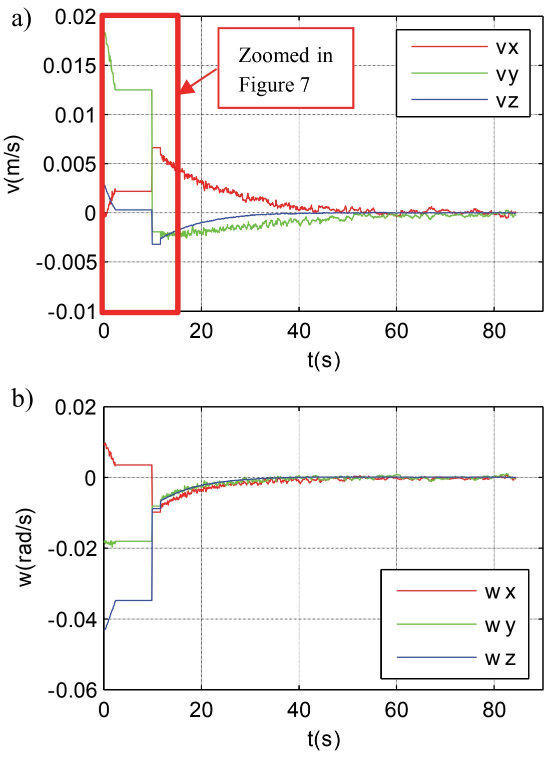

Figure 6: Experiment 1. Velocity of the end-effector: a) lineal velocity, b) rotational velocity.

The different events described over the Figure 5 can be better seen over a zoom in the evolution of the velocities sent to the robot. Figure 7 shows this zoom. The events are marked and the Event detector decisions shaded over the Figure.

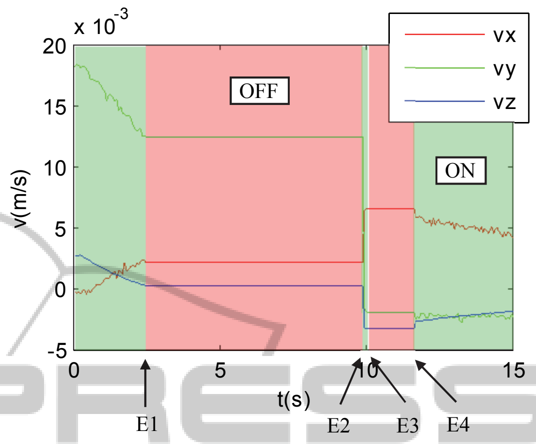

Figure 7: Experiment 1. Zoom in lineal velocity of the end-effector. Event detector decisions.

In order to prove the processor's time reduction, time processing has been plotted at each iteration of the event-based visual servoing controller (see Figure 8). The processing time is more than 10 times smaller when visual servoing is deactivated than time when it is activated.

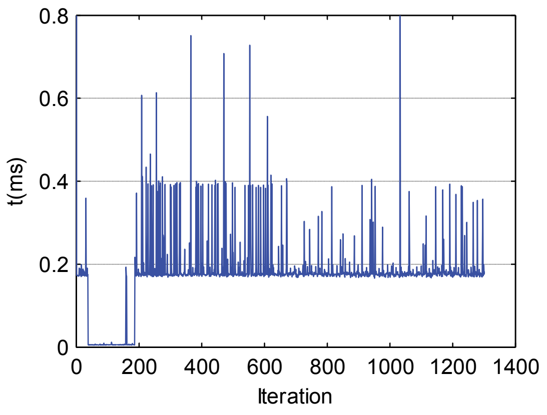

Figure 8: Experiment 1. Processing time.

4.2 Experiment 2

In the second experiment the Event generator triggers six events. The set of desired visual features is $\mathbf{s}_d$=[249, 156, 382, 156, 388, 305, 247, 304]. The visual servoing gain has been set to 0.1. Figure 9 shows that the robot reaches the final desired pose in the 3D space. The evolution of the end-effector presents three different changes of direction. These changes match the main event triggered during the positioning task.

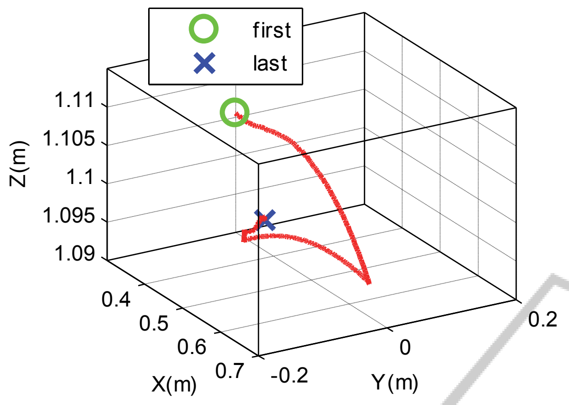

Figure 9: Experiment 2. 3D trajectory of the robot end-effector.

These changes of direction can be better seen in Figure 10. This Figure shows the evolution of the features in the image regarding this second experiment. As happened in the first experiment, 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.

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.

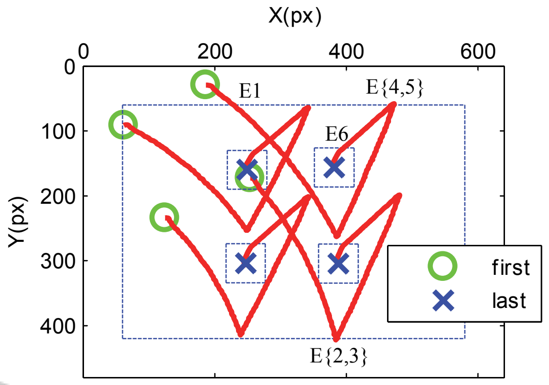

Figure 10: Experiment 2. Evolution of the features in the image.

Figure 11 illustrates the lineal and angular velocity of the end-effector during the task. Figure 12 shows a zoom over time regarding the lineal velocity depicted in Figure 11.a. Over Figure 12, different instants have been shaded in green or red.

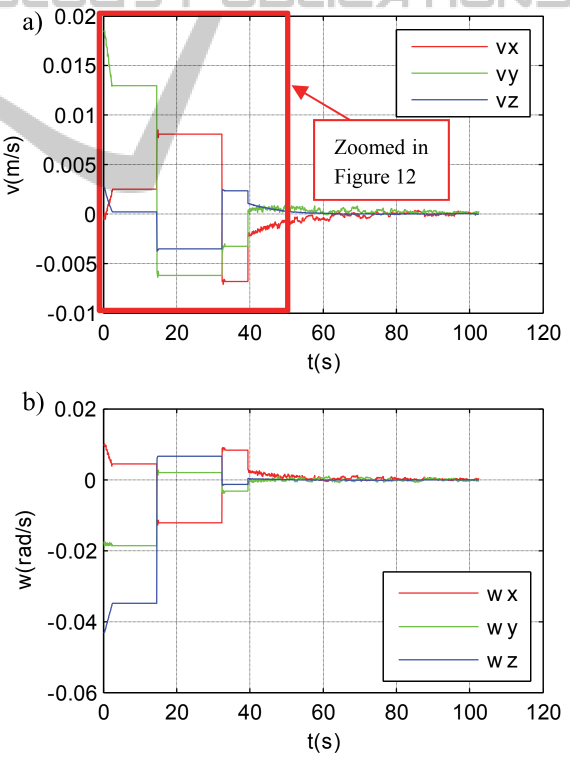

Figure 11: Experiment 2. Velocity of the end-effector: a) lineal velocity, b) rotational velocity.

Furthermore, the six events marked in Figure 10 over the evolution of the features in the image have been also marked in Figure 12. A green shaded area represents the lineal velocity sent to the robot when

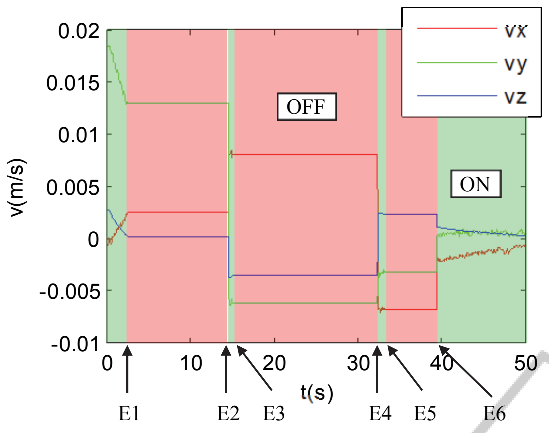

Figure 12: Experiment 2. Zoom in lineal velocity of the end-effector. Event detector decisions.

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 proposed 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.

Figure 13 shows the time processing elapsed at each iteration. The values are similar to the ones obtained in the first experiment (see Figure 8). The time saved during the deactivation of the visual servoing controller can be spent in any other task.

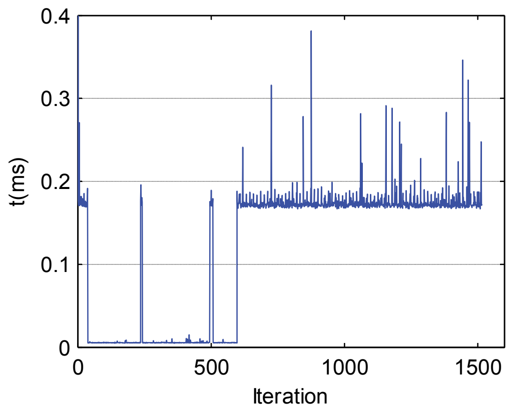

Figure 13: Experiment 2. Processing time.

5 CONCLUSIONS

The system proposed in this paper is able to guide a robot from any initial pose to the desired one using images only when it is necessary. Event-based visual servoing is a new control scheme that reduces the transfer load between the camera and the processing unit. It also reduces the processing time during the visual servoing task. In addition, the new controller described in this paper stops visual features from leaving the field of view, which would cause the failure of the task.

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.

Now, the authors are working in the robustness of the method. The Event detector presented can only switch between visual servoing and the last visual servoing calculated velocity. In order to avoid situations like that of experiment 2, where visual features bounce more than one time over the external security area, image position estimators based on the Kalman filter have 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.

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, GRE10-16, respectively.

REFERENCES

Aström, K. J., 2008. Event Based Control. In *Analysis and Design of Nonlinear Control Systems,* 127-147. Springer Berlin Heidelberg.

Benosman, R., Ieng, S.-H., Clercq, C., Bartolozzi, C., Srinivasan, M., 2012. Asynchronous frameless event-based optical flow. *Neural Networks*, *27*(0), 32-37.

Chaumette, F., Hutchinson, S. A., 2006. Visual servo control. I. Basic approaches. *Robotics & Automation Magazine, IEEE*, *13*(4), 82-90.

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.

Elouardi, A., Bouaziz, S., Dupret, A., Lacassagne, L., Klein, J. O., Reynaud, R., 2009. Image Processing: towards a System on Chip. *Image Processing*, InTech.

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.

Keen, S., Leach, J., Gibson, G., Padgett, M.J., 2007. Comparison of a high speed camera and a quadrant detector for measuring displacements in optical tweezers. *J. Opt. A: Pure Appl. Opt.* 9**,** S264–S266.

Li, J., Gao, H., Wang, Z., Geng, T., Zhao, Y., 2012. Embedded image grabber and processing system based on camera link. In *World Automation Congress*.

Mahowald, M. A., Mead, C., 1991. The silicon retina. *Sci. Amer*., 264(5), 76–82.

Malis, E., Rives, P., 2003. Robustness of image-based visual servoing with respect to depth distribution errors. In *Proceedings of the IEEE International Conference on Robotics and Automation,* 1, 1056-1061.

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.

Nasibov, H., Kholmatov, A., Akselli, B., Nasibov, A., 2012. A PIV dynamic velocity range enhancement approach using ROI option of imaging sensors, *Flow Measurement and Instrumentation*, 28, 35-44.

Ni, Z., Pacoret, C., Benosman, R., Ieng, S., Régnier, S., 2102. Asynchronous event-based high speed vision for microparticle tracking, *Journal of Microscopy*, 245, 236-244.

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.

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.

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.

Shu, Y., Guo, X., Liu, M., Buma, T., 2010. One-dimensional optoacoustic receive array employing parallel detection and video-rate acquisition, In *Proceedings of the IEEE Ultrasonics Symposium (IUS)*, 2396-2399.

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.