# Combining IBVS and PBVS to ensure the visibility constraint

Olivier Kermorgant, F. Chaumette

▶ **To cite this version:**

Olivier Kermorgant, F. Chaumette. Combining IBVS and PBVS to ensure the visibility constraint. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'11, 2011, San Francisco, USA, United States. pp.2849-2854. hal-00639683

**HAL Id: hal-00639683**
**https://inria.hal.science/hal-00639683**

Submitted on 9 Nov 2011

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Combining IBVS and PBVS to ensure the visibility constraint

Olivier Kermorgant and François Chaumette

***Abstract*— In this paper we address the issue of hybrid 2D/3D visual servoing. Contrary to popular approaches, we consider the position-based visual servo as the core of our scheme. 2D information is only added when necessary, that is when the observed object is about to leave from the field of view, even partially. The injection of 2D information is tuned by only one parameter, that simply defines the area where a point is considered to be near to the image border. Simulations allow the comparison of the proposed scheme with both position-based and hybrid schemes, showing nice properties. Finally, experiments are performed on a real object that is tracked by a pose estimation algorithm. Results validate the approach by showing the object stays entirely in the image field of view.**

***Index Terms*— Visual servoing, hybrid control**

## I. Introduction

A main issue in visual servoing is the choice of the set of visual features. Indeed, contrary to classical sensors such as laser range sensors or robot encoders, cameras deliver high-dimensional data that need to be processed in order to obtain information about the robot environment. Visual servo are classically divided in two main approaches [2]. Image-based visual servoing (IBVS) focus on using geometric features that can be directly obtained from the image. When used with a teaching-by-showing to specify the desired robot pose, these schemes are known to be robust to calibration and model errors. The main drawback is that the induced 3D trajectory is not predictable and may be unsatisfactory. Also, jacobian singularities or local minima may exist [1]. On the opposite, position-based visual servoing (PBVS) uses the 3D pose of the camera as visual features and is globally asymptotically stable when the pose is assumed to be perfectly estimated. Additional information such as true camera parameters and a 3D model of the observed object are necessary to estimate the camera pose from the acquired images. For some PBVS schemes, the induced 3D trajectory is a 3D straight line, but there is no control in the image space and the object may get out of the field of view (FoV). Another design of PBVS does not lead to a straight line but implicitly ensures that the object reference point stays in the image [15]. A similar approach is called 2 1/2D VS, where some of the 3D features are replaced by 2D information, leading to a set of 6 features that allow analytical proof of convergence and study the sensibility to calibration errors. Popular choices are to use 2D coordinates of an image point together with 3D translation (resp. rotation) along the $z$-axis and the whole vector for 3D rotation (resp. translation) [11]. This strategy ensures the reference point stays in the image, yet the object may partially leave the FoV. In [14] another 6-feature-set is designed to cope the visibility issue: 2D coordinates and 3D rotation together with the radius of the circumcircle of the object projection. Yet, this shape may not be suited for all 3D objects, and a planning strategy must be done in the general case. When using more than 6 features, merging 2D and 3D information leads to various hybrid approaches. In [3], PBVS translational and rotational motion are switched when the image points are near to the border. The goal is to perform only the 3D motion that will keep the points in the image. Isolating the $z$-axis is also considered in [5] with a partitioned strategy. In practice, this leads to backward translational motions which is not an optimal 3D trajectory. In [7], a switching between IBVS and PBVS is proposed. A maximum error is defined for each scheme, that makes the system switch to IBVS (resp. PBVS) if the 2D (resp. 3D) error norm is too high. However the maximum acceptable 2D error may be difficult to define: if too high, a point may be able to reach the image border. If too small the scheme may stay in IBVS mode. Finally, a hybrid control law has been proposed in [10] with a 5D-objective function that allows determining the best weighting between IBVS and PBVS. Once again, the tuning may be difficult in practice and does not ensure the visibility because the 2D weights are bounded. To cope with partial visibility lost, a scheme has been proposed in [8] where the features can be continuously added and removed from the task when they enter or get out of the field of view. As a point is approaching the image border, its participation in the control law is decreasing and becomes null before it is no more visible. Still, the number of observed points must be enough to perform the pose estimation. Our work is part of the hybrid approaches that let PBVS keep the points in sight. The main idea is to modify the PBVS as little as possible since this scheme provides nice 3D trajectories. To do so, we choose the opposite approach of [8]: instead of not taking into account the points that are about to leave the field of view, we inject them into the control law so that they stay in the image. Contrary to previous hybrid schemes, all point coordinates are treated separately depending on their distance to the image border. First, visual servoing classical control laws are recalled. In Section III we expose the proposed hybrid scheme and we proof the local stability. We also consider particular issues due to discretization. Finally, we compare our approach to other popular schemes in both simulations and experiments.

Olivier Kermorgant and François Chaumette are with INRIA Rennes-Bretagne Atlantique, Campus de Beaulieu, Rennes, France.
E-mail: Firstname.Name@inria.fr

## II. Visual servoing schemes

In this section we recall the modeling of classical visual servoing control laws. VS schemes consist in defining a robot task by an error function to be minimized:

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^* \tag{1}$$

where $\mathbf{s}$ is a vector of $m$ visual features with $\mathbf{s}^*$ being their desired values. In the following, we assume $m \geq 6$. Once the visual features have been chosen, the time variation of $\mathbf{s}$ is directly related to the camera velocity screw $\mathbf{v}_s$ by:

$$\dot{\mathbf{s}} = \dot{\mathbf{e}} = \mathbf{L}_s \mathbf{v}_s \tag{2}$$

where $\mathbf{L}_s$ is the interaction matrix related to $\mathbf{s}$ and can usually be computed analytically [2]. Assuming the robot can be controlled with the camera velocity, (2) leads to the following control law:

$$\mathbf{v}_s = -\lambda \widehat{\mathbf{L}_s^+} \mathbf{e} \tag{3}$$

where $\widehat{\mathbf{L}_s^+}$ is an estimation of the Moore-Penrose pseudo-inverse of $\mathbf{L}_s$, that ensures at best that the error $\mathbf{e}$ is exponentially minimized in terms of euclidean norm. We now recall the VS schemes that are considered in our hybrid approach.

### A. Position-based visual servoing

In PBVS [16], a computer vision method retrieves the 3D pose (position and orientation) of the camera, which is then used as the visual features. If a CAD model of the object is known, tracking the edges of the object [4], [6] is a popular pose estimation approach. On the other hand, a model-free method has been presented in [11], allowing for the homography estimation from a set of corresponding points. The pose matrix that transforms points from object frame to camera frame is an element of the group of rigid body transformations $SE(3)$ and can be written:

$${}^c\mathbf{M}_o = \begin{bmatrix} {}^c\mathbf{R}_o & {}^c\mathbf{t}_o \\ 0 & 1 \end{bmatrix} \tag{4}$$

where ${}^c\mathbf{R}_o \in SO(3)$ is a rotation matrix and ${}^c\mathbf{t}_o \in \mathbb{R}^3$ is a translation vector. Classically, 3D features are $\mathbf{s} = (\mathbf{t}, {}^{c*}\theta\mathbf{u}_c)$ where ${}^{c*}\theta\mathbf{u}_c$ expresses the angle and the axis of the 3D rotation that the robot has to achieve. Two popular choices exist for the translation, determining the behavior of the robot [2].

*1) 3D straight line:* $\mathbf{t}$ can be chosen to express the transformation between the current camera frame $\mathcal{F}_c$ and the desired one $\mathcal{F}_{c*}$, that is $\mathbf{s} = ({}^{c*}\mathbf{t}_c, {}^{c*}\theta\mathbf{u}_c)$. In this case the corresponding desired feature is a null vector, and the interaction matrix is known to be bloc-diagonal, inducing decoupled translational and rotational motions. The corresponding camera trajectory is a straight 3D line. Although this scheme, that is denoted in this paper as ${}^{c*}\mathbf{t}_c$-PBVS, is popular for its very nice behavior in the 3D space, no control at all is done in the image and the visual features that allow for the pose estimation may be lost.

*2) 2D straight line:* Another choice is to use the translational vector relative to the observed object frame, that can be retrieved from the pose estimation algorithm. In this case, $\mathbf{s} = ({}^c\mathbf{t}_o, {}^{c*}\theta\mathbf{u}_c)$ and the interaction matrix $\mathbf{L}_s$ is bloc-triangular. The corresponding control law, denoted ${}^c\mathbf{t}_o$-PBVS, implicitly ensures that the reference point of the object frame $\mathcal{F}_o$ draws a straight line trajectory in the image. On the other hand, the camera trajectory does not follow a 3D straight line as soon as a rotational motion has to be performed. When the reference point is in the center of the object this scheme is very similar to 2 1/2D VS [11]. However, even if one point is ensured to stay in the image, nothing can be said for the other points or visual elements used in the pose estimation. Visibility is improved in [14] but problems still remain depending on the object shape and the camera pose. In the following the interaction matrix of the PBVS scheme is denoted $\mathbf{L}_{3d}$.

### B. Image-based visual servoing

IBVS uses features that can be measured in the image. Although many choices exist for 2D visual features, most hybrid strategies use the cartesian coordinates of image points. These features are known to induce large 3D motion in particular cases [1], however they are well-suited for the problem of visibility lost: indeed, as images are rectangular the distance to the border expresses naturally. The analytical expression of the interaction matrix of an image point, denoted $\mathbf{L}_{2d}$, depends both on its image coordinates $(x, y)$ and on its depth $Z$. While $x$ and $y$ can be retrieved from the image, a common issue in IBVS is that the depth has to be estimated. However in hybrid control the pose is assumed to be estimated hence point depths can be easily retrieved. The IBVS control law induces straight line trajectories of the selected points in the image. The main drawback is a lack of control in 3D space. Actually, in the case of a task controlling all the 6-DOFs, at least 4 points have to be used in order to avoid potential singularities of $\mathbf{L}_{2d}$ [1]. As $\mathbf{L}_{2d}$ is of rank 6 at most, it is overdetermined, which makes it possible to reach a local minimum, and prevents from ensuring perfect straight line trajectories for each point in some cases.

Finally, 2D virtual points can be computed from any 3D point when the camera pose ${}^c\mathbf{M}_o$ is known. Denoting ${}^o\mathbf{x}$ the coordinates of a 3D point in the object frame, the corresponding coordinates ${}^c\mathbf{x} = (X, Y, Z)$ in the camera frame yield:

$$\begin{bmatrix} {}^c\mathbf{x} \\ 1 \end{bmatrix} = {}^c\mathbf{M}_o \begin{bmatrix} {}^o\mathbf{x} \\ 1 \end{bmatrix} \tag{5}$$

2D coordinates can then be expressed with:

$$\begin{cases} x &= X/Z \\ y &= Y/Z \end{cases} \tag{6}$$

Hence, the interaction matrix of the image projection of a 3D point can be computed even if the point is not actually extracted from the image. Similarly, the corresponding desired features $\mathbf{s}^* = (x^*, y^*)$ can be computed from the desired camera position ${}^{c*}\mathbf{M}_o$ if this matrix is known. We now show how a set of 2D points can improve PBVS.

## III. 2D-AUGMENTED PBVS

### A. Hybrid control law

We assume a PBVS scheme is performed with the corresponding task denoted $\mathbf{e}_{3d} = \mathbf{s}_{3d} - \mathbf{s}^*_{3d}$. Considering a set of $p$ 3D points $({}^o\mathbf{x}_1, \ldots, {}^o\mathbf{x}_p)$ attached to the observed object frame, a 2D task $\mathbf{e}_{2d} = \mathbf{s}_{2d} - \mathbf{s}^*_{2d}$ can be defined as described in Section II-B. We define the global weighted task of dimension $(6 + 2p)$ by:

$$\mathbf{e_H} = \mathbf{H} \begin{bmatrix} \mathbf{e}_{3d} \\ \mathbf{e}_{2d} \end{bmatrix} = \mathbf{H}\mathbf{e} \tag{7}$$

where $\mathbf{H}$ is a positive semi-definite diagonal activation matrix that allows continuous adding and removing of some features [12]. The global minimum of $\mathbf{e}$ corresponds to the desired pose of the sole PBVS scheme, that is $\mathbf{s}_{3d} = \mathbf{s}^*_{3d}$. Indeed, the latter is equivalent to having ${}^c\mathbf{M}_o = {}^{c*}\mathbf{M}_o$, hence $\mathbf{s}_{2d} = \mathbf{s}^*_{2d}$ whatever the value of $\mathbf{H}$. The derivative of the task is $\dot{\mathbf{e}}_\mathbf{H} = \mathbf{H}\dot{\mathbf{e}} + \dot{\mathbf{H}}\mathbf{e}$. The second term can be neglected in a neighborhood of the desired position [2], leading to the following error time derivative:

$$\dot{\mathbf{e}}_\mathbf{H} = \mathbf{H} \begin{bmatrix} \mathbf{L}_{3d} \\ \mathbf{L}_{2d} \end{bmatrix} \mathbf{v}_s = \mathbf{H}\mathbf{L}_s\mathbf{v}_s \tag{8}$$

which, by analogy with (3), leads to the control law:

$$\mathbf{v}_s = -\lambda(\widehat{\mathbf{H}\mathbf{L}_s})^+\mathbf{H}\mathbf{e} \tag{9}$$

In the considered approach, $\mathbf{H}$ allows for the continuous injection of 2D features into the PBVS scheme. The weighting matrix is written under a particular form:

$$\mathbf{H} = \begin{bmatrix} \mathbb{I}_6 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{2d} \end{bmatrix} \tag{10}$$

where the weights are always equal to 1 for the 3D part, while the $2p$ weights corresponding to the 2D features are varying. In the following, the resulting control law is denoted 2D-augmented PBVS. Such a control law with varying weights is known to be continuous under three conditions. $\mathbf{H}\mathbf{L}_s$ and $\mathbf{H}\mathbf{e}$ have to be continuous, which is ensured by the computation of the weights that is exposed in Section III-C. The third condition is the pseudo-inverse being continuous, that is also ensured in our case since $\mathbf{H}\mathbf{L}_s$ is always of full rank 6 thanks to the PBVS scheme that has non-null weights.

The nearest approach to the proposed one is found in [10] and called 5D VS. In this case, the control law yields:

$$\mathbf{v}_s = -\lambda\widehat{\mathbf{L}_s}^+ \begin{bmatrix} h_{3d}\mathbb{I}_6 & \mathbf{0} \\ \mathbf{0} & h_{2d}\mathbb{I}_{2p} \end{bmatrix} \mathbf{e} \tag{11}$$

where scalars $h_{3d}$ and $h_{2d}$ allow for balancing the PBVS and the IBVS schemes. However, all 2D features are treated as a whole IBVS. Also, not taking into account the weighting matrix in the pseudo-inverse induces a conservative behavior in the case of small weights. Indeed, the zeroed error components are still taken into account, which is not the case with our approach as we will show in the next section. Finally, the weighting strategy that is proposed for 5D VS does not ensure that the points stay in the image FoV.

### B. Stability analysis

In the case of null 2D feature weights $\mathbf{H}_{2d} = \mathbf{0}$, the proposed control law (9) is equivalent to the PBVS part:

$$\begin{aligned}\mathbf{v}_s &= -\lambda \begin{bmatrix} \widehat{\mathbf{L}_{3d}} \\ \mathbf{H}_{2d}\widehat{\mathbf{L}_{2d}} \end{bmatrix}^+ \begin{bmatrix} \mathbf{e}_{3d} \\ \mathbf{H}_{2d}\mathbf{e}_{2d} \end{bmatrix} = -\lambda \begin{bmatrix} \widehat{\mathbf{L}_{3d}} \\ \mathbf{0} \end{bmatrix}^+ \begin{bmatrix} \mathbf{e}_{3d} \\ \mathbf{0} \end{bmatrix} \\ &= -\lambda \begin{bmatrix} \widehat{\mathbf{L}_{3d}}^+ & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{e}_{3d} \\ \mathbf{0} \end{bmatrix} = -\lambda\mathbf{L}^+_{3d}\mathbf{e}_{3d}\end{aligned} \tag{12}$$

Our scheme is thus different from (11). In the following of this section we assume at least one weight is non-null.
From (8) and (10), $\mathbf{H}\mathbf{L}_s$ is of dimension $((6+2p)\times 6)$ and is of full rank 6. Following the classical stability analysis of overdetermined VS schemes [2], local stability can be ensured as long as $\mathbf{L}_s$ is sufficiently well estimated. However, potential local minima may exist in the configurations where $\mathbf{H}^2\mathbf{e} \in \text{Ker } \mathbf{L}_s^\top$. In practice, we have not encountered any. We modify as little as possible the PBVS scheme and results will show that the domain of stability is very large.

### C. Computation of the weights

In this section we first show that a sufficiently high weight ensures that the corresponding feature stays in the image. We then expose our weighting strategy for the 2D features.

*1) 2D features convergence:* A classical Lyapunov function associated with the task (7) is $V(\mathbf{e}_\mathbf{H}) = \frac{1}{2}\mathbf{e}_\mathbf{H}^\top\mathbf{e}_\mathbf{H}$. Assuming we are in the domain of local stability, the time derivative of $V$ yields:

$$\dot{V} = \frac{\partial V}{\partial \mathbf{e}}\dot{\mathbf{e}} = \sum_{i=1}^{6+2p} h_i^2 e_i\dot{e}_i < 0 \tag{13}$$

Let $i > 6$ be a 2D feature index. We consider the desired feature $\mathbf{s}^*_i$ is strictly inside the image FoV. A sufficient condition for that feature to stay in the image is $e_i\dot{e}_i \leq 0$, which is ensured in three cases:

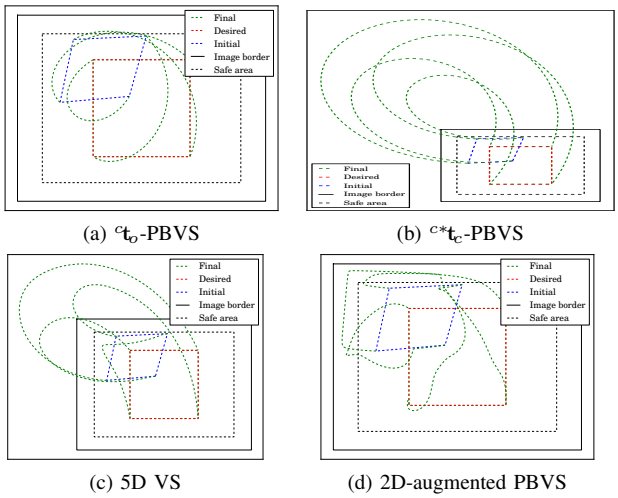
1) $e_i = 0$: thanks to the decoupled decrease, $\dot{e}_i$ is very small when $e_i = 0$. Since the feature $\mathbf{s}_i = \mathbf{s}^*_i$ is strictly inside the image, it does not leave the FoV at the next iteration.
2) $\dot{e}_i = 0$: the feature has no motion in the image, hence it stays in the FoV at the next iteration.
3) $e_i\dot{e}_i < 0$: in that case, (13) is equivalent to:
$$h_i^2 > -\frac{1}{e_i\dot{e}_i}\sum_{j\neq i} h_j^2 e_j\dot{e}_j \tag{14}$$
where $j$ is the index for the other features.

At any position there exists a sufficiently high weight that prevents an image point from leaving the field of view. However, when several feature points have to be considered simultaneously, the exact set of sufficient values may not be possible to determine: indeed, in (14) the minimum value of $h_i$ depends on the other weights $(h_j)_{j\neq i}$. In the next section, the weights are computed independently from one another. This prevents from computing the smallest weights, but we will see in Section V that the obtained values are still small in practice.

*2) Weighting strategy:* Classically when dealing with control in the image space, the smaller distance from the 2D points to the image border is computed. This value is then used to weight the IBVS scheme [3], [9]. In our case, each 2D feature is treated separately . Let $(x^-, x^+, y^-, y^+)$ be the image borders. We define the image safe region $\mathcal{S} = [x^{s-}, x^{s+}] \times [y^{s-}, y^{s+}]$ with:

$$\begin{cases} x^{s-} = x^- + \rho(x^+ - x^-) \\ x^{s+} = x^+ - \rho(x^+ - x^-) \end{cases}, \quad \begin{cases} y^{s-} = y^- + \rho(y^+ - y^-) \\ y^{s+} = y^+ - \rho(y^+ - y^-) \end{cases}$$

where $\rho \in [0, 0.5]$ is a tuning parameter. We have shown that a sufficiently high weight ensures that the feature stays in the




(a) ${}^{c}\mathbf{t}_{o}$-PBVS (b) ${}^{c*}\mathbf{t}_{c}$-PBVS

(c) 5D VS (d) 2D-augmented PBVS

Fig. 1. Image behavior for the 4 schemes. ${}^{c}\mathbf{t}_{o}$-PBVS (a) and 2D-augmented (d) keep the points in the image while ${}^{c*}\mathbf{t}_{c}$-PBVS (b) and 5D VS (c) lose visibility.

FoV. Hence, we propose the following weighting function:

$$h(x) = \begin{cases} \frac{x-x^{s+}}{x^{+}-x} & \text{if } x > x^{s+} \\ \frac{x-x^{s-}}{x^{-}-x} & \text{if } x < x^{s-} \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

$$h(y) = \begin{cases} \frac{y-y^{s+}}{y^{+}-y} & \text{if } y > y^{s+} \\ \frac{y-y^{s-}}{y^{-}-y} & \text{if } y < y^{s-} \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

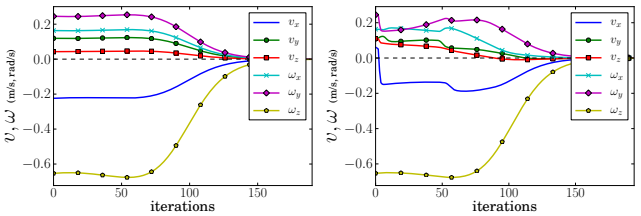
$h$ is null in the safe region and increases to $\infty$ as the feature approaches the image border. The condition (14) is thus ensured when any point is close to the border. All weights being continuous, the corresponding control law is continuous. Since very high values are never encountered in practice, they do not endanger the conditioning of $\mathbf{HL}_s$ for computing the pseudo-inverse. The only parameter for the tuning is $\rho$, that defines the image safe area.

## IV. Simulation Results

The considered schemes are evaluated in a simulation environment with ViSP software [13]. The object simply consists in four coplanar 3D points forming a square. First, we present the results of exhaustive runs, then the comparison with other schemes is exposed.

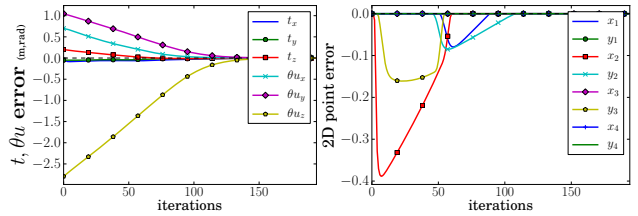
### A. Exhaustive tests

We have generated a set of 100 random poses such that the 4 points are in the image safe area defined by $\rho = 0.05$. All combinations of initial and desired poses, that is 9900 runs, are performed with our control law based on ${}^{c*}\mathbf{t}_{c}$-PBVS. All runs converge without leaving the image FoV with the control gain of $\lambda = 0.1$. Furthermore, even if only local stability has been proven, all runs converge to the global minimum. Over all the runs, the maximum weight is less than 20, and the mean of the maximum weight of each run is about 2.4. This reveals 2D features can very easily cope with the visibility constraint without impacting the control law conditioning. Still, as exhibited in [7] potential issues




(a) $\mathbf{v}_s$ in ${}^{c}\mathbf{t}_{o}$-PBVS (b) $\mathbf{v}_s$ in 2D-augmented PBVS

Fig. 2. Camera velocity in ${}^{c}\mathbf{t}_{o}$-PBVS and 2D-augmented PBVS. The velocities are similar although the schemes are different.




(a) $\mathbf{e}_{3d}$ in 2D-augmented (b) $\mathbf{H}_{2d}\mathbf{e}_{2d}$ in 2D-augmented

Fig. 3. Global error of 2D-augmented PBVS. 3D error (a) shows nice convergence of the pose, while the weighted 2D errors (b) illustrates the injection of 2D points during the task.

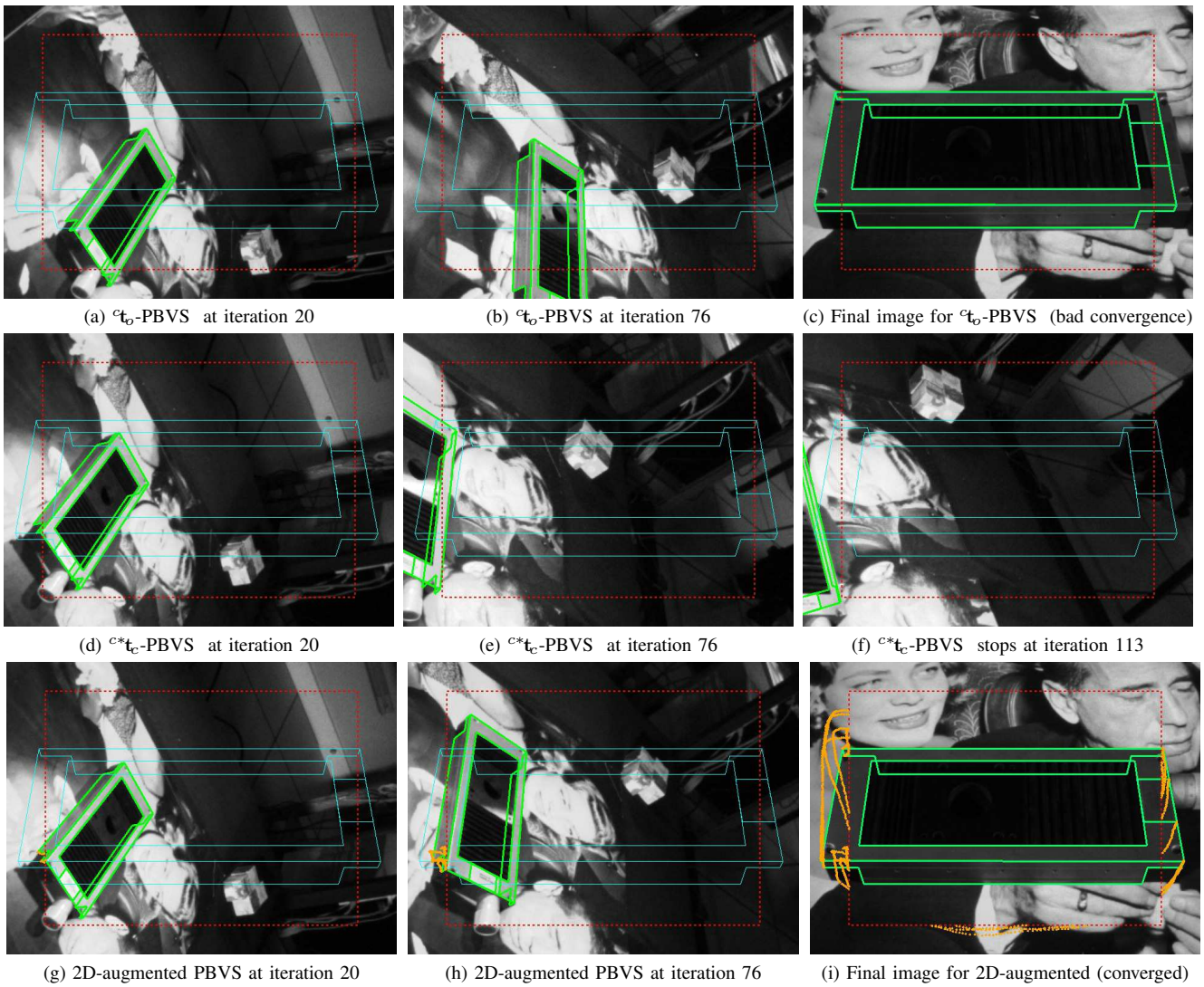
due to discretization may be encountered in the case of high control gain when some points are very near to the border. An easy way to cope with such problems is to use an adaptative gain that decreases as a point approaches the image border.

### B. Comparison with PBVS

Four schemes are compared in this section: ${}^{c}\mathbf{t}_{o}$-PBVS, ${}^{c*}\mathbf{t}_{c}$-PBVS, 5D VS [10] and 2D-augmented PBVS, with an initial pose making it necessary to perform both translational and rotational motions. The two hybrid schemes are performed together with the ${}^{c*}\mathbf{t}_{c}$-PBVS, that is the one inducing a straight 3D trajectory and the worst image behavior.

${}^{c}\mathbf{t}_{o}$-PBVS performs well as seen in Fig. 1a. In the considered case, the object reference point is in the middle of the square and ${}^{c}\mathbf{t}_{o}$-PBVS is similar to 2 1/2D VS. No control is done in the image except for the straight line trajectory of the object center, yet none of the 4 points leave the FoV. We will see in the next section that it can happen for other configurations. On the opposite, Fig. 1b shows that ${}^{c*}\mathbf{t}_{c}$-PBVS induces a large motion out of the image. The 5D VS is able to reduce the motion, but Fig. 1c shows the points still leave the FoV because of the bounded weights induced by this strategy. Finally, 2D-augmented PBVS is represented in Fig. 1d and is able to keep the points in sight. The trajectory change for the 2D points is particularly visible in the unsafe area, where the image border avoidance is performed. The two successful schemes are compared in terms of velocity in Fig. 2. It is interesting to notice that the camera velocities are very similar, although the control law are based on different schemes. Actually, the behaviors differ at the very beginning in Fig. 2b. Important variations are about $v_x$ and $\omega_y$, which are the main velocities that control the $x$-coordinate of the 2D points, that matches to Fig. 1b showing that in ${}^{c*}\mathbf{t}_{c}$-PBVS the points leave the FoV along the image $x$-axis.

The global error of 2D-augmented PBVS is represented in



(a) ${}^{c}\mathbf{t}_{o}$-PBVS at iteration 20 (b) ${}^{c}\mathbf{t}_{o}$-PBVS at iteration 76 (c) Final image for ${}^{c}\mathbf{t}_{o}$-PBVS (bad convergence)

(d) ${}^{c*}\mathbf{t}_{c}$-PBVS at iteration 20 (e) ${}^{c*}\mathbf{t}_{c}$-PBVS at iteration 76 (f) ${}^{c*}\mathbf{t}_{c}$-PBVS stops at iteration 113

(g) 2D-augmented PBVS at iteration 20 (h) 2D-augmented PBVS at iteration 76 (i) Final image for 2D-augmented (converged)

Fig. 4. Evolution of the acquired image for ${}^{c}\mathbf{t}_{o}$-PBVS (top), ${}^{c*}\mathbf{t}_{c}$-PBVS (middle) and 2D-augmented PBVS (bottom). In ${}^{c}\mathbf{t}_{o}$-PBVS the object partially leaves the FoV, inducing bad pose estimation. In ${}^{c*}\mathbf{t}_{c}$-PBVS the object quickly leave the FoV and the task fails. Our scheme converges with the desired position being unsafe for some points

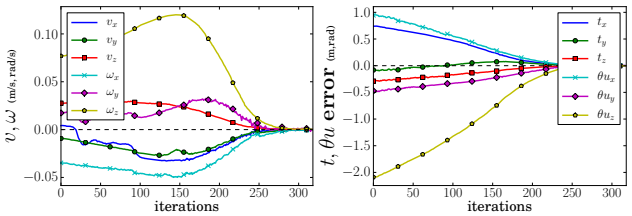
Fig. 3. 3D convergence to the desired pose is clearly visible in Fig. 3a. The weighted 2D error, that is $\mathbf{H}_{2d}\mathbf{e}_{2d}$, is shown in Fig. 3b. As can be seen, only 4 features out of 8 are used in the servo. As expected the first one is a $x$-coordinate (red curve), inducing the velocity change that can be seen in Fig. 2b. Basic correspondences can be done between Fig. 3b and Fig. 1d:

- During the task, 2 $x$-coordinates and 2 $y$-coordinates have been activated. They correspond to the unsafe trajectories that appear on the left or above the safe area in Fig. 1d.
- Only point 2 has both coordinates activated. Hence, it is the only point approaching the top left hand corner in Fig. 1d.

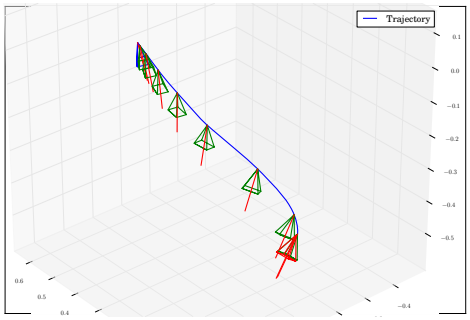
## V. Experimental results

Experiments are carried on a 6 DOFs gantry robot (see the video accompanying this paper). The camera and the robot are calibrated. The camera observes a mechanical object, the CAD model of which being known. During the task, the object lies on a textured surface. The edges are tracked to allow for the pose estimation [4] at camera rate (30Hz). Three schemes are compared: ${}^{c}\mathbf{t}_{o}$-PBVS, ${}^{c*}\mathbf{t}_{c}$-PBVS, and 2D-augmented PBVS. As in Section IV, the latter is performed with ${}^{c*}\mathbf{t}_{c}$-PBVS. This time, the 3D points are the ones defining the 24 nodes of the CAD model. In order to reduce the number of considered 3D points, one could use the set that defines the model envelope, since the inner points can never leave the FoV if the outer points are in the image.

The image behavior of the 3 schemes are represented in Fig. 4. The safe area is represented in red dotted line. The current projected CAD model is in green, while the desired one is in cyan. This time, ${}^{c}\mathbf{t}_{o}$-PBVS makes some points leave the image as can be seen in Fig. 4b. This occlusion prevents the successful tracking of the object, which leads to a convergence that does not correspond to the desired camera pose: the edges that are displayed in Fig. 4c are not the actual object edges. ${}^{c*}\mathbf{t}_{c}$-PBVS induces a trajectory that lacks of control in the image and quickly loses the object (see Fig. 4f). Our control law allows for the convergence of




(a) Camera velocity $\mathbf{v}_s$ (b) 3D pose error $\mathbf{e}_{3d}$




(c) 3D trajectory

Fig. 5. 3D behavior of our scheme. Camera velocity (a), pose error (b) and 3D trajectory (c). 2D features slightly modify the trajectory (c).

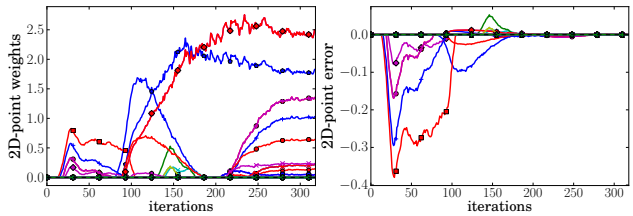
the scheme. The 2D features that are used are represented in orange. They are in the left side in the first images, that is the direction the ${}^{c*}\mathbf{t}_c$-PBVS leaves the FoV. As can be seen in Fig. 4i, during the task several 2D features have been activated and deactivated. Contrary to the simulation case, the final pose corresponds to the unsafe area for some points. This does not prevent the scheme from converging.

The 3D behavior of the camera is represented in Fig. 5. A large rotational motion around the optical axis can be seen in Fig. 5a. Although our scheme is based on the ${}^{c*}\mathbf{t}_c$-PBVS where the 3D trajectory is a straight line, Fig. 5c shows that here this is not the case at all. However, the nice convergence to the desired pose is observed in Fig. 5b.

Fig. 6a shows that several 2D features have non-null weights at the end of the task. They correspond to the final points being in the unsafe area in Fig. 4i. More particularly, 4 features have non-null weights at iterations 20-80. They correspond to the object corner being in the unsafe area in terms of $x$-coordinates in Fig. 4h. In Fig. 6b we can see these 4 features induce a large 2D error, that prevents from leaving the FoV. After iteration 100, very small 2D error is sufficient for keeping the object in the image. As announced in Section III-C, the corresponding weights are not high: the maximum value is about 2.6 and corresponds to a feature that is about in the middle of the unsafe area.

## VI. Conclusion

A new approach has been proposed to address the classical balance between PBVS and IBVS scheme. Starting with a PBVS that is known to have nice 3D properties, we use only the 2D features that are necessary for keeping the object in the field of view. The corresponding control law is locally stable and has its global minimum at the PBVS desired pose. The parameter for the injection of the 2D features sets the safe distance to the image border. Comparison with classical




(a) 2D weights $\mathbf{H}_{2d}$ (b) 2D weighted error $\mathbf{H}_{2d}\mathbf{e}_{2d}$

Fig. 6. 2D behavior of our scheme. Several weights are non-null at the end of the task, yet the 2D error is null as the points have converged to their desired position in the image.

PBVS schemes show that a few 2D features allow to ensure the visibility constraint.

## References


[1] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The Confluence of Vision and Control*. LNCIS Series, No 237, 1998.

[2] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, 2006.

[3] G. Chesi, K. Hashimoto, D. Prattichizzo, and A. Vicino, "Keeping features in the field of view in eye-in-hand visual servoing: A switching approach," *IEEE Trans. Robot.*, vol. 20, no. 5, 2004.

[4] A. Comport, E. Marchand, and F. Chaumette, "Efficient model-based tracking for robot vision," *Advanced Robotics*, vol. 19, no. 10, October 2005.

[5] P. Corke and S. Hutchinson, "A new partitioned approach to image-based visual servo control," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, 2001.

[6] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, 2002.

[7] N. Gans and S. Hutchinson, "Stable visual servoing through hybrid switched-system control," *IEEE Trans. Robot.*, vol. 23, no. 3, 2007.

[8] N. García-Aracil, E. Malis, R. Aracil-Santonja, and C. Pérez-Vidal, "Continuous visual servoing despite the changes of visibility in image features," *IEEE Trans. Robot.*, vol. 21, no. 6, 2005.

[9] A. Hafez and C. Jawahar, "Probabilistic Integration of 2D and 3D Cues for Visual Servoing," in *9th IEEE Int. Conf. on Control, Automation, Robotics and Vision*, 2007.

[10] ——, "Visual servoing by optimization of a 2D/3D hybrid objective function," in *IEEE Int. Conf. on Robotics and Automation*, Roma, Italy, 2007.

[11] E. Malis, F. Chaumette, and S. Boudet, "2-1/2d visual servoing," *IEEE Trans. Robot. Autom.*, vol. 15, no. 2, Apr. 1999.

[12] N. Mansard, A. Remazeilles, and F. Chaumette, "Continuity of varying-feature-set control laws," *IEEE Trans. Autom. Control*, vol. 54, no. 11, November 2009.

[13] E. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, December 2005.

[14] G. Morel, T. Liebezeit, J. Szewczyk, S. Boudet, and J. Pot, "Explicit incorporation of 2d constraints in vision based control of robot manipulators," *Experimental Robotics VI*, 2000.

[15] B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice, "Position based visual servoing: keeping the object in the field of vision," in *IEEE Int. Conf. on Robotics and Automation*, 2002.

[16] W. Wilson, W. Hulls, and G. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 5, 2002.