A mathematical characterization of the convergence domain for Direct Visual Servoing

Meriem Belinda Naamani, Guillaume Caron, Mitsuharu Morisawa, El Mustapha Mouaddib

A mathematical characterization of the convergence domain for Direct Visual Servoing

Meriem Belinda Naamani[1,2], Guillaume Caron[1,3], Mitsuharu Morisawa[1], El Mustapha Mouaddib[3]

***Abstract*— Direct Visual Servoing (DVS) is a technique that controls the robot motion by using the pixel intensities captured by a camera. DVS demonstrates high accuracy at convergence, prompting the development of various methods aimed at expanding its convergence domain.**

In this paper, we propose a mathematical characterization of the DVS convergence domain with closed-form expressions for the controlled degrees of freedom. From these expressions, we concluded that the extent of the convergence domain is related to the presence of isotropic or defocus blur, a phenomenon that had only been observed previously as a trend in empirical experiments.

I. Introduction

A. Motivation

Image-based visual servoing is a fundamental technique that uses information extracted from the image as visual features to guide the motion of robotic systems for precise positioning, tracking and navigation tasks [1].

To date, the most accurate family of visual features is the direct use of the information captured in the image, which seminal approach is Photometric Visual Servoing (PVS). It uses image brightness directly to control the robot motion to reach a desired image captured at the desired robot pose [2]. Great accuracy at convergence can be obtained from a variety of initial poses within the local basin of attraction of the control law to the desired image. But for PVS and its successors, there is no analytical study of their convergence domain. The goal of this paper is to address this gap.

B. Related Works

In [3], the visual servoing task is defined as a nonlinear least squares minimization problem, where the goal is to move the robot to a reference position starting from an initial position. In [4], the photometric cost function is studied, and a number of optimization methods are presented, such as the Newton optimization technique, which involves the computation of the Hessian matrix (related to the curvature of the cost function). The Newton algorithm is used in [5], where it has been shown that by using the Hessian, the spatial trajectory of the camera is improved, and it overcomes the so-called advance/retreat problem of visual servoing. However, since the Hessian matrix is needed, the computational cost can become expensive, and the region of convergence can be small in practice. In [4], another optimization method is stated, which is the Gauss-Newton optimization law. This method is the most commonly used in visual servoing [1]. It approximates the Hessian matrix of the Newton algorithm using the Jacobian matrix.

One more optimization method mentioned in [4] is the Levenberg-Marquardt algorithm. This method is a modification of the Gauss-Newton algorithm, where a damping factor is introduced to adjust the contribution of the gradient and the Hessian matrices to the control law. A Levenberg-Marquardt-like control law is used in [6] and compared to the Gauss-Newton one. It resulted in much smoother trajectories with the former and a better time-to-convergence than the latter.

Despite the latter pros, PVS convergence domain could be improved such as in [6] by introducing Photometric Gaussian Mixtures (PGM) as dense features. This method consists of representing each pixel by a 2D Gaussian, and then combining them to produce the Gaussian mixture. Thus, the overlap between the current and desired images increases as the Gaussian extent increases, and it has been experimentally observed that there is a significant increase in the convergence domain. More recent works include the use of Discrete Orthogonal Moments (DOM) as visual features for the servoing. In [7], Tchebichef, Krawtchouk, and Hahn moments are used as examples of DOMs to extract visual features, allowing an efficient feature representation by eliminating redundant data between neighboring pixels. Experiments show that this method effectively enhances the region of convergence while preserving a high accuracy.

In addition, some learning-based methods have been recently introduced. In [8], a neural network is trained to map the camera images and the robot poses into a shared latent space. The points in this space correspond to the similarity of the images or the distance between robot positions. The goal is then to minimize the distance between the current image and desired image in the latent space.

While increasing the convergence domain, these methods are computationally expensive, and need some parameters tuning. The Defocus-Based Direct Visual Servoing (DDVS) [9] balances the convergence domain and the processing time by effectively using the defocus blur to optically smooth the image. Similarly to PGM, it was experimentally observed that the defocus blur increases the convergence domain.

C. Contributions

In order to explain how blurred images can lead to larger Direct Visual Servoing (DVS) convergence domains than sharp images, we make three contributions in this paper leveraging the thin lens camera model:

[1] CNRS-AIST JRL (Joint Robotics Laboratory), IRL, Tsukuba, Japan.
[2] CNRS-UM LIRMM, UMR 5506, Montpellier, France.
[3] University of Picardie Jules Verne, MIS laboratory, Amiens, France.
Corresponding author: meriem.naamani@lirmm.fr

- We formulate the first mathematical characterization of the convergence domain in DVS;
- We decline the latter mathematical result to prove both PVS narrow convergence domain and PGM VS large one;
- We introduce a variant of DDVS with defocused desired image highlighting an extended convergence domain even without any camera motion along its optical axis.

These contributions are validated with simulations and experiments in a simplified environment.

D. Outline

The remainder of the paper is organized as follows. First, Section II recalls the theoretical background on the Defocus Blur Model, and defocus-based direct visual servoing (DDVS). After that, in Section III, we derive the closed form analytical expression of the DDVS cost function and we extract its region of convergence. In Section V, the theory is put into practice with simulation and experimental results. Finally, the conclusion and future works are presented in Section VI.

II. Defocus-Based Direct Visual Servoing

A. Model of Defocus

Defocus blur is a loss of sharpness caused by the integration of light over the camera aperture area when the captured scene is not on the focal plane [10]. In that case, the scene is said to be out of focus (see the object plane versus the focal plane in Fig. 1).

When the scene is in focus, a camera is assumed similar to a pinhole, represented as the perspective projection model. The latter maps a 3D point $\mathbf{X} = \begin{bmatrix} X, & Y, & Z \end{bmatrix}^T \in \mathbb{R}^3$ into a single 2D point on the image plane at the intersection with the line joining the 3D point $\mathbf{X}$ to the center of projection, located at the pinhole [11].

Out of the focal plane, the pinhole assumption no longer holds; instead, the thin lens camera model applies (Fig.1). This model considers a range of apertures contrary to the infinitesimal pinhole. Thus, instead of mapping $\mathbf{X}$ into an image plane point, the thin lens model maps $\mathbf{X}$ to an image plane area known as the circle of confusion (CoC). The CoC is the intersection between the cone of rays emanating from $\mathbf{X}$ and the image plane. Its diameter, depicted by $d(Z) \in \mathbb{R}_+^*$ in Fig. 1, depends on the focus depth $Z_f \in \mathbb{R}_+^*$, the focal length $f \in \mathbb{R}_+^*$, and the aperture $D \in \mathbb{R}_+^*$ [9]:

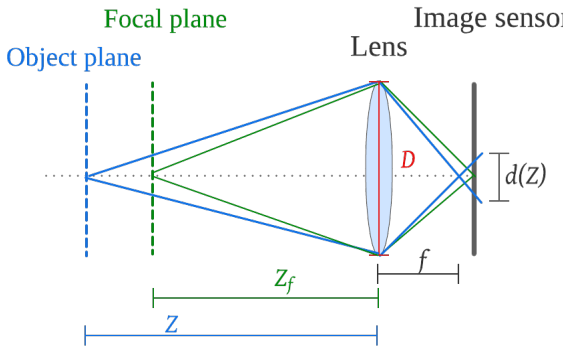


Fig. 1: Thin lens camera model: Z is the object depth; Z_f is the focus depth; D is the camera aperture; f is the focal length; $d(Z)$ is the diameter of the Circle of Confusion.

$$d(Z) = \frac{Df}{Z_f - f}\left(1 - \frac{Z_f}{Z}\right). \tag{1}$$

The defocus blur can be approximated by a point spread function that depends on the CoC diameter. As in [9] the defocus blur is approximated by a normal Gaussian kernel, which width (spread) depends on the CoC diameter, the camera parameters and the 3D point $\mathbf{X}$.

Denoting $\lambda(Z)$ the spread of the Gaussian in pixel units, and assuming 99.7% of the normal Gaussian spans the CoC, we have:

$$\lambda(Z) = \frac{d(Z)}{6K_u}, \tag{2}$$

with K_u the physical size of a pixel, used as a scale factor.

In this model of defocus, $\mathbf{X}$ is projected at the center of the CoC following the perspective camera model. This transformation is denoted by function pr: $\mathbf{X} \in \mathbb{R}^3 \rightarrow \mathbf{u} \in \mathbb{R}^2$ as:

$$pr(\mathbf{X}) = \begin{bmatrix} \frac{f}{K_u} & 0 & u_0 \\ 0 & \frac{f}{K_u} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{X}{Z} \\ \frac{Y}{Z} \\ 1 \end{bmatrix}, \tag{3}$$

where $\mathbf{X}$ is expressed in the camera frame and $(u_0, v_0) \in \mathbb{R}^2$ being the coordinates of the principal point.

Hence, the normal Gaussian kernel $\tilde{g}(\mathbf{u},\mathbf{X})$ is:

$$\tilde{g}(\mathbf{u},\mathbf{X}) = \frac{1}{2\pi\lambda(Z)^2} \exp\left(\frac{||\mathbf{u} - pr(\mathbf{X})||^2}{2\lambda(Z)^2}\right). \tag{4}$$

In sharp image regions, Z tends to Z_f, the CoC tends to a point as shown by the green lines in Fig. 1. In that case the Gaussian tends to a Dirac impulse. As the diameter of the CoC increases, i.e. $|Z - Z_f|$ increases as illustrated by the blue lines in Fig. 1, the spread of the Gaussian increases, resulting in a greater defocus blur.

Consider a continuous scene $\mathcal{X} \subset \mathbb{R}^3$ of 3D points, $\mathbf{X} \in \mathcal{X}$. Assuming that the scene radiance $L(\mathbf{X}) \in \mathbb{R}_+$ is equally mapped to brightness $I(pr(\mathbf{X})) \in \mathbb{R}_+$, then the defocus image I_d is expressed as:

$$I_d(\mathbf{u}) = \int_{\mathcal{X}} I(pr(\mathbf{X}))\tilde{g}(\mathbf{u},\mathbf{X})\, d\mathbf{X}. \tag{5}$$

B. Defocus-based direct visual servoing

With $(N,\ M)$ the width and height of the image, the goal of DDVS is to minimize the Sum of Squared Differences (SSD) of the vectorized current image $\mathbf{I}_d(\mathbf{p}) \in [0,255]^{NM\times 1}$ and the vectorized desired image $\mathbf{I}_d^* \in [0,255]^{NM\times 1}$:

$$C(\mathbf{p}) = \frac{1}{2}||\mathbf{I}_d(\mathbf{p}) - \mathbf{I}_d^*||^2, \tag{6}$$

with $\mathbf{p} \in \mathbb{R}^n$ the camera pose and n is the number of controlled Degrees-of-Freedom (DoF) (if $n = 6$, $\mathbf{p}$ is the stacking of the translation vector and the axis-angle rotation vector of the scene to camera transformation).

The control is based on the Gauss-Newton control law:

$$\mathbf{v} = -\mu \mathbf{L}_{I_d}^{+} \left(\mathbf{I}_d(\mathbf{p}) - \mathbf{I}_d^* \right), \tag{7}$$

where $\mathbf{v} \in \mathbb{R}^n$ is the camera velocity. $\mu \in \mathbb{R}_+$ is a gain. $\mathbf{L}_{I_d}^{+} \in \mathbb{R}^{NM \times n}$ is the pseudo-inverse of the interaction matrix that links the camera velocity to the change of brightness of the image $\mathbf{I}_d$ (see [9] for its detailed expression).

As with most DVS techniques, only local stability can be ensured, however the region of convergence revealed to be quite large in practice.

III. Convergence of Direct Visual Servoing using the thin lens camera model

Defining the set $U = [0, N-1] \times [0, M-1] \subset \mathbb{N}^2$, which consists of the integer coordinates of all image pixels, the DDVS control law can be rewritten as an optimization problem:

$$\hat{\mathbf{p}} = \underset{\mathbf{p}}{\operatorname{argmin}} \frac{1}{2} \sum_{\mathbf{u} \in U} \left(I_d(\mathbf{u}, \mathbf{p}) - I_d^*(\mathbf{u}) \right)^2. \tag{8}$$

For the theoretical convergence analysis of the above problem (8), the first step consists in expressing explicitly the cost function in terms of the controlled degrees of freedom.

A. An Analytical Expression of the Desired Image

Since there is no general analytical expression for an image, we consider an ideal scene $\mathcal{X}$ defined by a single bright 3D point at $\mathbf{X}^* = \begin{bmatrix} X^*, & Y^*, & Z^* \end{bmatrix}^T \in \mathbb{R}^3$ in the camera frame. Without loss of generality in the classical Lambertian scene assumption often made with DVS, we also assume an ideal camera that maps equally a sharp pixel intensity $I(pr(\mathbf{X}^*))$ to the scene luminance I_1 at $\mathbf{X}^*$. Thus, substituting $\tilde{g}(\mathbf{u}, \mathbf{X}^*)$ in (5) with (4), the desired image is expressed as:

$$I_d^*(\mathbf{u}) = \frac{I_1}{2\pi\lambda(Z^*)} \exp\left(-\frac{||\mathbf{u} - pr(\mathbf{X}^*)||^2}{2\lambda(Z^*)} \right), \tag{9}$$

with $pr(\mathbf{X}^*)$ given by (3), and $\lambda(Z^*)$ by (2). This is of course very simplified compared to a truly captured image. But in the rest of this section, the analytical expression of the desired image (9) allows to characterize analytically the convergence domain of (8). The experimental results of visual servoing using a real robot with a real camera (Sec. V-B.1) will also demonstrate its relevance.

B. Analytical Expression of the Current Image

Considering a bidimensional translational motion of the camera in the 3D space, the current 3D point in the camera frame is given by $\mathbf{X} = \begin{bmatrix} X^* + t_X, & Y^* + t_Y, & Z^* \end{bmatrix}^T \in \mathbb{R}^3$, where t_X and t_Y are the translations along the X-axis and Y-axis respectively. Thus, $n = 2$ (see Sec. II-B) with $\mathbf{p} = [t_X, t_Y]^T$, that we substitute with $\mathbf{t} = \mathbf{p}$ for more clarity on the DoF considered. Following the same steps as in Section III-A, the current image is then expressed by:

$$I_d(\mathbf{u}, \mathbf{t}) = \frac{I_1}{2\pi\lambda(Z^*)} \exp\left(-\frac{||\mathbf{u} - pr(\mathbf{X})||^2}{2\lambda(Z^*)} \right), \tag{10}$$

where $\lambda(Z^*)$ is given by (2), and $pr(\mathbf{X})$ is given by (3).

C. Analytical Expression of the Direct Cost Function

From the problem (8), the cost function is:

$$C_d(\mathbf{t}) = \frac{1}{2} \sum_{\mathbf{u} \in U} \left(I_d(\mathbf{u}, \mathbf{t}) - I_d^*(\mathbf{u}) \right)^2 \tag{11}$$

To ease the start of mathematical developments, let us assume an infinite camera plane, i.e. $\mathbf{u} \in \mathbb{R}^2$. Back to the continuous space, the expression of the cost becomes (12):

$$C(\mathbf{t}) = \frac{1}{2} \int_{\mathbf{u}} \left(I_d(\mathbf{u}, \mathbf{t}) - I_d^*(\mathbf{u}) \right)^2 d\mathbf{u}. \tag{12}$$

Evaluating the above integral (12) and substituting $pr(\mathbf{X})$ and $pr(\mathbf{X}^*)$ with (3), the closed-form expression of the direct cost function is then:

$$C(\mathbf{t}) = \frac{I_1^2}{4\pi\lambda(Z^*)^2} \left[1 - \exp\left(-\frac{f^2(t_X^2 + t_Y^2)}{(2K_u Z^* \lambda(Z^*))^2} \right) \right]. \tag{13}$$

Below, we begin by considering the Newton algorithm as an initial step to identify the region of convergence, before moving to the Gauss-Newton optimization algorithm.

D. Expression of the region of convergence for planar translation control with the Newton optimization algorithm

The Newton algorithm is a second-order optimization method based on a local approximation of the cost function by its second-order Taylor expansion [12]. Considering the minimization problem (8), one iteration of the Newton algorithm computes:

$$\mathbf{t}_{k+1} = \mathbf{t}_k - \left(\nabla^2 C(\mathbf{t}_k) \right)^{-1} \nabla C(\mathbf{t}_k), \tag{14}$$

where:

- $\mathbf{t}_k$: is the current pose;
- $\nabla C(\mathbf{t}_k)$: is the Gradient of the cost;
- $\nabla^2 C(\mathbf{t}_k)$: is the Hessian matrix of the cost.

For compactness, we omit the *k* subscript in the remainder of this section.

Newton's algorithm converges when two conditions are satisfied :

- **Condition 1**: the cost function must be twice differentiable;
- **Condition 2**: the Hessian matrix needs to be definite positive, i.e. : $\det(\nabla^2 C(\mathbf{t})) > 0$.

In (13), $C(\mathbf{t})$ is only made of operations twice differentiable so Condition 1 is met. Then, we leverage Condition 2 to find for which subset $\mathbf{t}$ is in the convergence domain of $C(\mathbf{t})$.

The Hessian of (13) expresses as:

$$\nabla^2 C(\mathbf{t}) = \begin{bmatrix} \dfrac{\partial^2 C}{\partial t_X^2}(\mathbf{t}) & \dfrac{\partial^2 C}{\partial t_X \partial t_Y}(\mathbf{t}) \\ \dfrac{\partial^2 C}{\partial t_Y \partial t_X}(\mathbf{t}) & \dfrac{\partial^2 C}{\partial t_Y^2}(\mathbf{t}) \end{bmatrix}, \tag{15}$$

which details as:

$$\nabla^2 C(\mathbf{t}) = \begin{bmatrix} 1 - \frac{f^2 t_X^2}{2(Z^* K_u \lambda(Z^*))^2} & -\frac{f^2 txty}{2(Z^* K_u \lambda(Z^*))^2} \\ -\frac{f^2 txty}{2(Z^* K_u \lambda(Z^*))^2} & 1 - \frac{f^2 t_Y^2}{2(Z^* K_u \lambda(Z^*))^2} \end{bmatrix} \frac{I_1^2 f^2}{8\pi(Z^* K_u^2 \lambda(Z^*)^2)} \exp\left(-\frac{f^2(t_X^2 + t_Y^2)}{(2K_u Z^* \lambda(Z^*))^2}\right). \tag{16}$$

The determinant of (16) is expressed as:

$$\det(\nabla^2 C(\mathbf{t})) = \frac{I_1^2 f^2}{8\pi(Z^* K_u^2 \lambda^2)} \left(1 - \frac{f^2(tx^2 + ty^2)}{2(K_u Z^* \lambda)^2}\right) \exp\left(-\frac{f^2(tx^2 + ty^2)}{(2K_u Z^* \lambda)^2}\right). \tag{17}$$

Then, we solve $\det(\nabla^2 C(\mathbf{t})) > 0$ for $\mathbf{t}$, leading to:

$$r^2 < \frac{D^2(Z^* - Z_f)^2}{18(Z_f - f)^2}, \tag{18}$$

where $r^2 = t_X^2 + t_Y^2$. In other words, the region of convergence is a disk centered at $\mathbf{t} = [0, 0]^T$ with a radius $r_{Ne} = \frac{\sqrt{2}}{6} D \left(\frac{Z^* - Z_f}{Z_f - f}\right)$. It is directly proportional to the camera aperture D and the focus depth $(Z - Z_f)$.

E. Expression of the region of convergence for planar translation control with Gauss-Newton optimization algorithm

The Gauss-Newton method is based on the Newton method, where the Hessian is approximated as:

$$\nabla^2 C(\mathbf{t}) \approx \mathbf{H} = \left(\frac{\partial \mathbf{I}}{\partial \mathbf{t}}(\mathbf{u}, \mathbf{t})\right)^T \left(\frac{\partial \mathbf{I}}{\partial \mathbf{t}}(\mathbf{u}, \mathbf{t})\right). \tag{19}$$

$\frac{\partial \mathbf{I}}{\partial \mathbf{t}}(\mathbf{u}, \mathbf{t})$ is the image brightness Jacobian. This ensures the approximated Hessian $\mathbf{H}$ is always positive.
One step of the Gauss-Newton optimization algorithm is given by [3]:

$$\mathbf{t}_{k+1} = \mathbf{t}_k - \mathbf{H}_k^{-1} \nabla C(\mathbf{t}_k), \tag{20}$$

where $\mathbf{H}_k$ is computed using (19) at iteration k.

Assuming $\mathbf{H}_k$ is not singular, the algorithm converges when $\nabla C(\mathbf{t}_k) = 0$. When $\nabla C(\mathbf{t}_0) = 0$, $\mathbf{t}_0$ corresponds either to a local minimum or to a saddle point surrounding a plateau area. A plateau refers to a region where the cost function stagnates [13].

If the Gauss-Newton algorithm is initialized near a saddle point then it will get stuck on said plateau. In order to find the region of convergence of the optimization using the Gauss-Newton method, we first express the Gradient of the direct cost (13):

$$\nabla C(\mathbf{t}) = \frac{I_1^2 f^2 \exp\left(-\frac{f^2(t_X^2 + t_Y^2)}{(2K_u Z^*)^2 \lambda(Z^*)^2}\right)}{8\pi(K_u Z^*)^2 \lambda(Z^*)^4} \begin{bmatrix} t_X \\ t_Y \end{bmatrix}. \tag{21}$$

Then, we can try solving $\nabla C(\mathbf{t}) = 0$ for $\mathbf{t}$, that leads to several solutions: either $\mathbf{t} = \begin{bmatrix} 0, & 0 \end{bmatrix}^T$ which is the global minimum; or $\mathbf{t} \to \pm\infty$. Out of the latter solutions, since $\nabla C(\mathbf{t}) > 0$, it means the algorithm is globally convergent. That is not surprising since both current (10) and desired (9) images are Gaussians defined to the infinity. But of course, a real camera does not perceive the slight variations of $C(\mathbf{t})$ around $\mathbf{t}$ of large norm due to its irradiance resolution and image quantification.

1) Thresholded analytical expression of images: To consider the latter physical limitations of the camera sensor, we introduce $l \in \mathbb{R}_+^*$ the lowest observable irradiance value leading to a non zero pixel integer intensity, hence:

$$I_{dl}(\mathbf{u}) = \begin{cases} I_d(\mathbf{u}) & \text{if} \quad I_d(\mathbf{u}) \geq l \\ l & \text{otherwise} \end{cases}. \tag{22}$$

Considering the desired image expression (9), and applying the condition in (22), the desired image can be rewritten as follows, posing $\rho_{\mathbf{u}}^* = ||\mathbf{u} - pr(\mathbf{X}^*)||$ for compactness:

$$I_{dl}^*(\mathbf{u}) = \begin{cases} \frac{I_1 \exp\left(-\frac{\rho_{\mathbf{u}}^{*2}}{2\lambda(Z^*)^2}\right)}{2\pi\lambda(Z^*)^2} & \text{if } \rho_{\mathbf{u}}^{*2} \leq -2\lambda(Z^*)^2 \ln\left(\frac{2\pi\lambda(Z^*)^2}{I_1}\right) \\ l \quad \text{otherwise.} \end{cases} \tag{23}$$

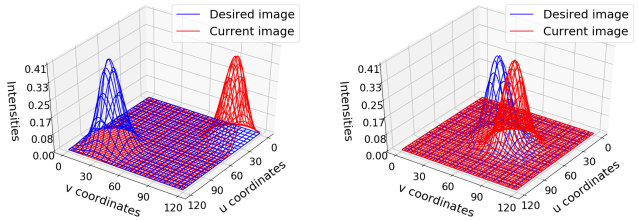
Similarly, posing $\rho_{\mathbf{u}} = ||\mathbf{u} - pr(\mathbf{X})||$ for compactness, the current image becomes:

$$I_{dl}(\mathbf{u}, \mathbf{t}) = \begin{cases} \frac{I_1 \exp\left(-\frac{\rho_{\mathbf{u}}^2}{2\lambda(Z^*)^2}\right)}{2\pi\lambda(Z^*)^2} & \text{if } \rho_{\mathbf{u}}^2 \leq -2\lambda(Z^*)^2 \ln\left(\frac{2\pi\lambda(Z^*)^2}{I_1}\right) \\ l \text{ otherwise.} \end{cases} \tag{24}$$

2) Direct Cost Function Expression with Thresholded Analytical Images: Since the expressions of $I_{dl}(\mathbf{u}, \mathbf{t})$ and $I_{dl}^*(\mathbf{u})$ are piecewise functions, we can consider two cases:

- **First case**: no overlap between $I_{dl}(\mathbf{u}, \mathbf{t})$ and $I_{dl}^*(\mathbf{u})$ (e.g. in Fig. 2a)
- **Second case**: overlapping between the two thresholded Gaussians as shown in Fig. 2b.

a) Cost function without overlapping: The range of definition of the Gaussian portion of the desired image is a disk centered at $\mathbf{o}_1 = pr(\mathbf{X}^*)$ with a radius of $r_d = \sqrt{-2\lambda(Z^*)^2 \ln(2\pi\lambda(Z^*)^2/I_1)}$, and for the current



(a) (b)

Fig. 2: Two cases for defining the piecewise direct cost function: (a) No overlap between the thresholded Gaussians of the current and desired images; (b) Overlap between the thresholded Gaussians of the current and desired images.

image a disk centered at $\mathbf{o}_2(\mathbf{t}) = pr(\mathbf{X})$ with a radius of $r_c = r_d$. The two latter disks do not overlap if the distance between their centers is larger than the sum of the two radii, i.e. $||\mathbf{o}_1 - \mathbf{o}_2(\mathbf{t})|| \geq r_c + r_d$. Substituting each variable of the latter expression with their detailed writing and rearranging leads to the condition for non-overlapping images:

$$t_X^2 + t_Y^2 \geq -8\lambda(Z^*)^2 \frac{(K_u Z^*)^2}{f^2} \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right). \quad (25)$$

Integrating (12) over the two radii r_c and r_d and taking its limit when $l \to 0$, the cost function in the case of no overlapping simplifies to the constant:

$$C_{no}(\mathbf{t}) = \frac{I_1^2}{4\pi\lambda(Z^*)^2}. \quad (26)$$

b) Cost function with overlapping: Following a similar reasoning than in Section III-E.2.a, the region where the two Gaussians overlaps is:

$$t_X^2 + t_Y^2 < -8\lambda(Z^*)^2 \frac{(K_u Z^*)^2}{f^2} \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right).$$

The domain of integration $U_i \subset \mathbb{R}^2$ is given by the region of intersection between the two disks, that are all $\mathbf{u}$ satisfying both equations (27a) and (27b) for the current and desired image respectively:

$$||\mathbf{u} - pr(\mathbf{X})||^2 < r_c^2, \quad (27a)$$

$$||\mathbf{u} - pr(\mathbf{X}^*)||^2 < r_d^2. \quad (27b)$$

Then, the cost function is evaluated by integrating (12) over U_i and taking its limit when $l \to 0$:

$$C_o(\mathbf{t}) = \frac{I_1^2}{4\pi\lambda(Z^*)^2}\left(1 - \exp\left(-\frac{f^2\left(t_X^2 + t_Y^2\right)}{(2K_u Z^*)^2\lambda(Z^*)^2}\right)\right). \quad (28)$$

c) Closed-form direct cost function with thresholded analytical images: Considering the results in Sections III-E.2.a and III-E.2.b, the direct cost function with thresholded analytical images is:

$$C_l(\mathbf{t}) = \begin{cases} \frac{I_1^2}{4\pi\lambda(Z^*)^2} & \text{if } t_X^2 + t_Y^2 \geq -\frac{(K_u Z^*)^2}{f^2} 8\lambda(Z^*)^2 \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right) \\ \frac{I_1^2}{4\pi\lambda(Z^*)^2}\left(1 - \exp\left(-\frac{f^2\left(t_X^2 + t_Y^2\right)}{(2K_u Z^*)^2\lambda(Z^*)^2}\right)\right) & \text{otherwise.} \end{cases} \quad (29)$$

3) Region of convergence for planar translation control with Gauss-Newton optimization algorithm: Using (29), we deduce the expression of $\nabla C(\mathbf{t})$ as:

$$\nabla C(\mathbf{t}) = \begin{cases} \begin{bmatrix} 0 \\ 0 \end{bmatrix} & \text{if } t_X^2 + t_Y^2 \geq -\frac{(K_u Z^*)^2}{f^2} 8\lambda(Z^*)^2 \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right) \\ \frac{I_1^2 f^2 \exp\left(-\frac{f^2\left(t_X^2 + t_Y^2\right)}{(2K_u Z^*)^2\lambda(Z^*)^2}\right)}{8\pi(K_u Z^*)^2\lambda(Z^*)^4} \begin{bmatrix} t_X \\ t_Y \end{bmatrix} & \text{otherwise.} \end{cases} \quad (30)$$

Substituting $\lambda(Z^*)$ by its expression with the thin lens model (2), and setting $r^2 = t_X^2 + t_Y^2$ the region of convergence expresses as function of the camera lens parameters:

$$r^2 < -2\frac{D^2(Z^* - Z_f)^2}{9(Z_f - f)^2} \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right). \quad (31)$$

The region of convergence is thus a disk centered at $\mathbf{t} = (0,0)^T$ with a radius of $r_{GN} = \sqrt{-2\frac{D^2(Z^* - Z_f)^2}{9(Z_f - f)^2} \ln\left(\frac{2\pi\lambda(Z^*)^2 l}{I_1}\right)}$.

IV. Region of convergence for PVS, PGM VS and DDVS for plan translational motion

A. Convergence of PVS

In [2], the photometric visual servoing is considered with a pinhole camera model. As mentioned in II-A the thin lens camera model simplifies to the pinhole model when considering a very small aperture, i.e. $D \to 0$. Thus, the limit of the general expression of the convergence domain (31) when $D \to 0$ leads to the PVS convergence domain:

$$r_{PVS} \to 0. \quad (32)$$

B. Convergence of PGM VS

PGM VS relies on mixing as much Gaussians as pixels in the image, the Gaussians sharing a common extension parameter but each having the magnitude of the image intensity, i.e.:

$$g(I, \mathbf{u}_g, \mathbf{u}, \lambda_g) = I(\mathbf{u}) \exp\left(-\frac{(u_g - u)^2 + (v_g - v)^2}{2\lambda_g}\right), \quad (33)$$

with:

- $\lambda_g \in \mathbb{R}_+^*$ is the extension parameter;
- $\mathbf{u}_g = [u_g, v_g]^T$ are the Gaussian function coordinates;
- $I(\mathbf{u})$ is the pixel intensity.

Such isotropic Gaussian expression is very close to the analytical expression of images (9) under the assumptions we made, except for the normalizing factor of the Gaussian related to defocus (4) that is absent in (33). However, our assumptions make $Z = Z^*$ always. Hence, $\lambda(Z) = \lambda(Z^*)$ and the PGM VS cost function with fixed λ_g is a constant factor times (29), thus not impacting the convergence domain boundaries.

Consequently, the boundary of PGM VS' region of convergence with fixed λ_g is:

$$r_{PGM}^2 = -\frac{(K_u Z^*)^2}{f^2} 8\lambda_g^2 \ln\left(\frac{l}{I_1}\right), \quad (34)$$

where it is clear that $r_{PGM} > 0$ as soon as $l < I_1$, which is a reasonable assumption. Indeed, the minimum observable intensity l by the camera should always be a tiny fraction of I_1, otherwise the image content will be at best a single pixel, meaning we fall back under the PVS case. Furthermore, the larger λ_g, the larger r_{PGM}.

As a partial conclusion, comparing (34) to (32) proves PGM VS' greater convergence domain than PVS.

C. Convergence of DDVS

In [9], DDVS considers a sharp desired image by setting the focus depth Z_f as close as possible to Z^*. For camera motions along its optical axis (at least), this setting is interesting to combine both the large convergence domain thanks to defocus blur and accuracy thanks to sharp images at convergence. However, when pure translational motion in the XY-plane is considered, the image stays sharp as in the PVS case.

Hence, to enhance the capabilities of DDVS, we leverage PGM VS' idea that even the desired image can be blurred to enable a large convergence domain even in the pure XY-plane translational motion case. This is done by setting Z_f not close to Z^*, i.e. making $(Z^* - Z_f)^2$ large in (31), where it is clear that the larger $(Z^* - Z_f)^2$, the farther the region of convergence boundary. But a key impact of the DDVS with defocused desired image over PGM VS with a large λ_g for the desired image is that defocus is the result of optical characteristics that does not need image processing, contrary to the computation of PGMs.

V. Results

A. Simulation results

Following the assumptions used in this paper, the simulation process considers a single 3D point at a desired position $\mathbf{X}^* = \begin{bmatrix} 0, & 0, & 1\ m \end{bmatrix}^T$, thus $Z^* = 1\ m$. The radiance at $\mathbf{X}^*$ is $L(\mathbf{X}^*) = 255$. The virtual thin lens camera has intrinsic parameters $\{f = 17\ mm, K_u = 5.3\ \mu m, u_0 = 180\ pixels, v_0 = 180\ pixels, D = f/0.95\}$. The controlled DoF are $\mathbf{t} = (t_X, t_Y)$. We apply Newton (14) then Gauss-Newton (20) methods. We start by considering a sharp desired image, i.e. $Z_f \approx Z^*$. Then, in order to highlight the effect of the defocused desired image on DDVS' region of convergence, we increase the value of $Z^* - Z_f$, leading to the results in Table I. Simulations confirm the theoretical result that the region of convergence with Gauss-Newton is larger than with Newton. Moreover, for both techniques, the region of convergence with the defocus blur is larger than with sharp images.

TABLE I: DDVS: theoretical and observed boundaries of regions of convergence (RoC) in simulation. Unit: mm.

	RoC for Newton		RoC for Gauss-Newton	
Focus depth (Z_f)	Theoretical	Simulation	Theoretical	Simulation
999.997	0.043	0.042	0.801	0.801
500	4.37	4.36	12.6	12.6
300	10.43	10.43	12.7	12.7

B. Experimental results

1) Experimental Setup: In this experiment, a Universal Robot 10 is used with a Flir Flea3 FL3-U3-13E4C camera and a Yakumo lens (focal length of 17 mm, and a maximum aperture F-0.95) on its end effector as shown in Fig. 3. The latter are connected to a laptop (Intel Core i9 CPU). Using (7), the velocities are computed, with

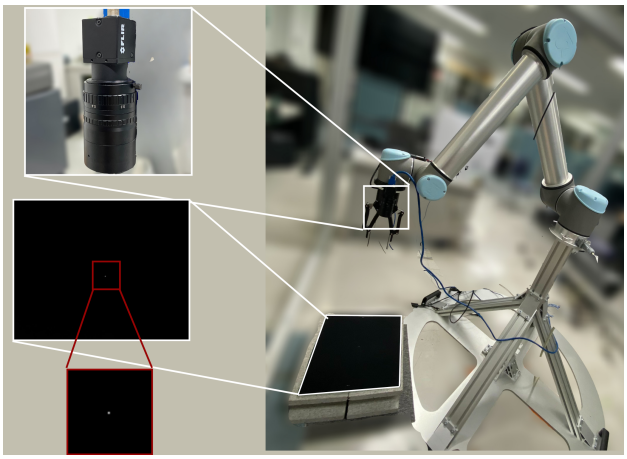


Fig. 3: Robot arm with a camera on the wrist pointed to a dark part of the floor where there is a tiny tin ball.

the camera parameters $\{f = 17\ mm, K_u = 5.3\ \mu m, u_0 = 160\ pixels, v_0 = 120\ pixels, D = f/0.95, Z_f\}$. The used images are of size 320×240 pixels.

The considered scene consists in a tin ball on a texture-less dark background, to mimic the single bright 3D point (Sec. III). Then we implemented[1] the DDVS with the C++ programming language using the *ViSP* library to implement the interaction matrix of [9]. In this experiment, the goal is to compare the theoretical regions of convergence with the practical ones, and to explicit their relationship with the defocus blur.

2) Experimental results with a sharp desired image: In this section, we control bidimensional translational degrees of freedom, t_X and t_Y. We report the experiment with an aperture diameter of $D = f/0.95$, a focus depth of $Z_f = 0.7\ m$, and $Z = Z^* = Z_f$. The image is acquired fronto-parallel to the flat scene. Using the camera parameters given in Section V-B.1 and the expression (31) the theoretical region of convergence is $|r| < 2.4\ mm$.

At first a translation of $|r| = 2.3\ mm$ is applied to the camera, which generates an error of $\delta u = 2\ pixels$ and $\delta v = -1\ pixel$ in the image. DDVS converges after 466 iterations, with a final error of $\delta r = 0.068\ mm$ (Fig. 4a- 4h).

Then, we increase progressively the initial error and observe the convergence of the DDVS. Once we reach an initial error of $|r| = 2.8\ mm$ equivalent to a pixel error of $\delta u = 2\ pixels$ and $\delta v = -2\ pixels$, we start noticing that the shape of the residual curve changes (Fig. 4o-4p), in this case taking 2449 iterations to converge with a final error of $\delta r = 0.15\ mm$.

We keep increasing the initial error, until we reach $|r| = 2.9\ mm$, applying an error of $\delta u = 2\ pixels$ and $\delta v = -3\ pixels$, we observe that the DDVS does not converge anymore (Fig. 4q-4x).

3) Experimental results with a Defocused Desired Image: As in Section V-B.2, we control two degrees of freedom, however the desired image is out of focus with $Z_f = 0.6\ m$

[1] https://github.com/jrl-umi3218/DirectVisualServoing

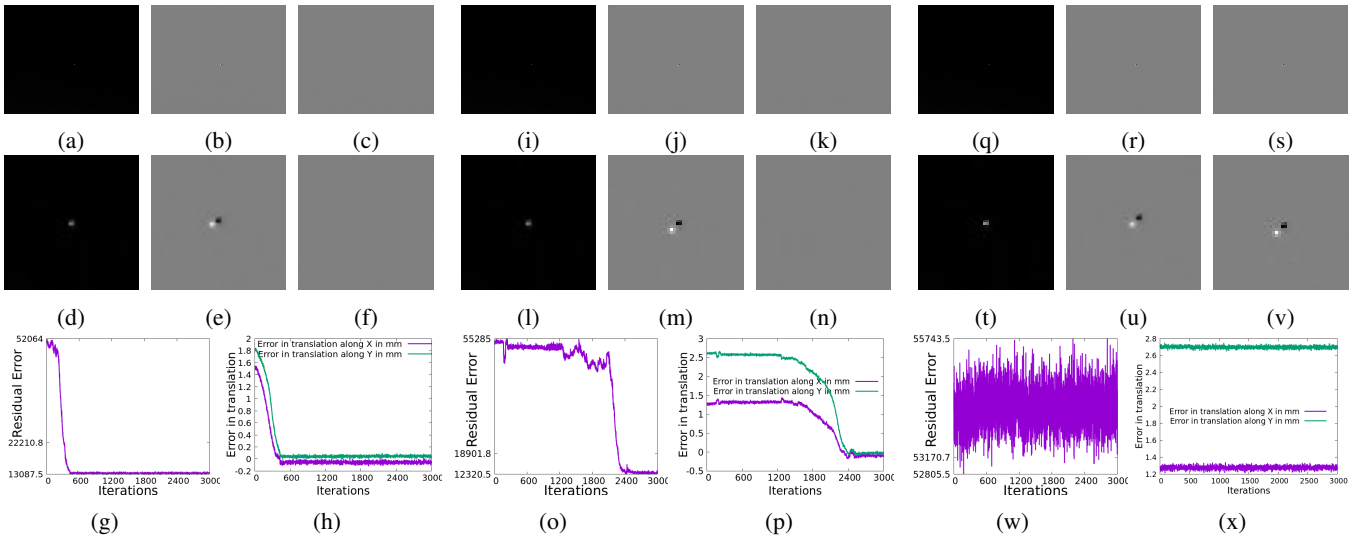


Fig. 4: DDVS with sharp images, (a-c): Desired, initial, and final difference images for an initial translation error of $\Delta r = 2.3\ mm$. (d-f): cropped version of (a-c) for visualization. (g): Evolution of the cost as function of iterations. (h): Translational error (t_X purple, t_Y green) as function of iterations. (i-p): Same as (a-h) for an initial translational error of $\Delta r = 2.8\ mm$. (q-x): same as (a-h) for an initial translational error of $\Delta r = 2.9\ mm$

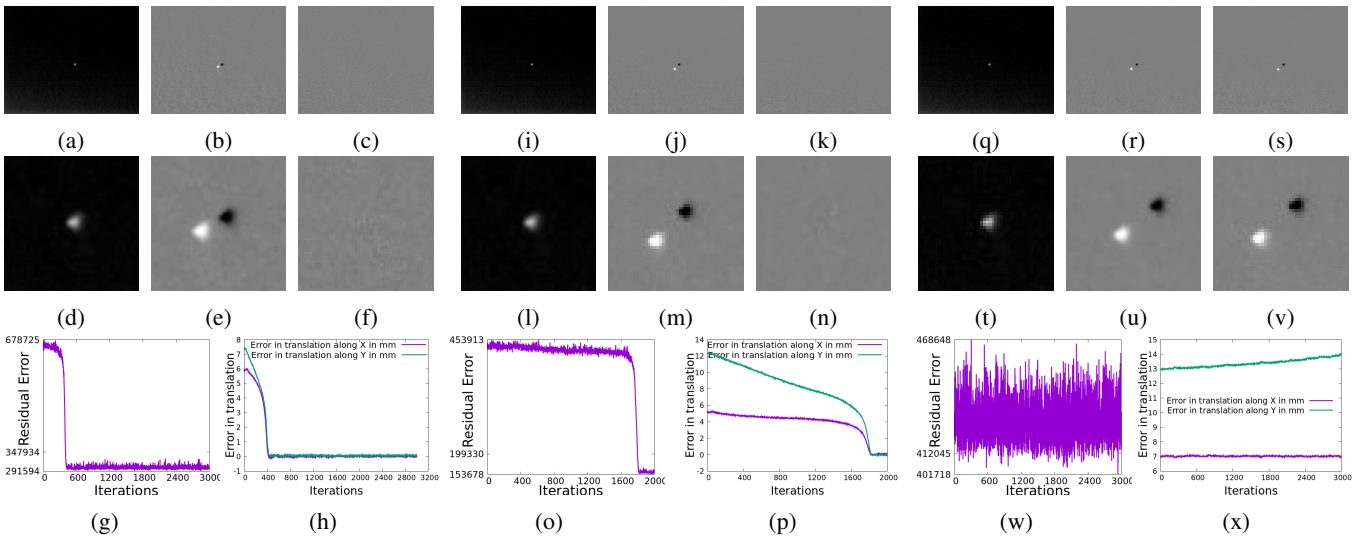


Fig. 5: DDVS with defocused desired images, (a-c): Desired, initial, and final difference images for an initial translation error of $\Delta r = 9.43\ mm$. (d-f): cropped version of (a-c) for visualization. (g): Evolution of the cost as function of iterations. (h): Translational error (t_X purple, t_Y green) as function of iterations. (i-p): Same as (a-h) for an initial translational error of $\Delta r = 13.6\ mm$. (q-x): same as (a-h) for an initial translational error of $\Delta r = 14.0\ mm$

and $Z = Z^* = 0.7\ m$. Using the same camera parameters, and the expression given by (31), the theoretical region of convergence is $|r| < 13.4\ mm$.

Starting with a translation of $|r| = 9.43\ mm$ applied to the camera, it generates an error of $\delta u = 9\ pixels$ and $\delta v = 5\ pixels$ in the image. DDVS converges after 466 iterations, with a final error of $\delta r = 0.08\ mm$ (Fig. 5a-5h).

As in Section V-B.2, we increase the initial error progressively to experimentally find the boundary of DDVS' convergence domain. Once we reach an initial error of $|r| = 13.6\ mm$, equivalent to a pixel error of $\delta u = 11\ pixels$ and $\delta v = -12\ pixels$, the shape of the residuals changes (Fig. 5i-5p), in this case taking 1800 iterations to converge with a final error of $\delta r = 0.034\ mm$.

We keep increasing the initial error until we reach $|r| = 14.0\ mm$, generating an error of $\delta u = 13\ pixels$ and $\delta v = 11\ pixels$ in the image. DDVS does not converge anymore (Fig. 5q-5x).

4) Summary: The results of Sections V-B.2 and V-B.3, are summarized in Table II. We observe that the theoretical region of convergence is on par with the practical one (a difference of around $0.4\ mm$). The difference is mainly due to the tin ball diameter not being small enough to be considered an infinitesimal point. We can also observe that

as we increase the defocus blur at the desired pose, the region of convergence increases significantly, by 485% in our experiments.

TABLE II: Theoretical and practical results for DDVS.

Focus depth (mm)	Theoretical RoC (mm)	Practical RoC (mm)	Initial error in pixels($\delta u, \delta v$)
$Z_f = 700$	2.4	2.8	(2,-2)
$Z_f = 600$	13.4	13.6	(13,11)

VI. Conclusion and Future Works

In this paper we expressed analytically the region of convergence for direct visual servoing by exploiting the thin lens camera model to express a closed-form cost function. This cost function is then used to express the convergence domain for the Newton and the Gauss-Newton algorithms. The latter could be achieved by introducing the thresholded defocused image model, that models physical limitations of a real camera. This analytical expression is general enough to allow finding the convergence domain expressions for the Photometric Visual Servoing and the Photometric Gaussian Mixtures-based Visual Servoing, in order to compare them analytically for the first time. As a consequence, the closed-form expression of the Defocus-based Direct Visual Servoing (DDVS) could also lead to the idea of considering a defocused desired image, highlighting a significant extension of DDVS' convergence domain, even without motion along the camera optical axis.

Simulation and practical experiments with a robot arm show that the theoretical convergence domains and the practical ones are very close, thus confirming the interest of our methodology.

Future works will tackle the expression of the region of convergence for more degrees of freedom.

References

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

[2] C. Collewet and E. Marchand, "Photometric Visual Servoing," *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 828–834, Aug. 2011.

[3] E. Malis, "Improving vision-based control using efficient second-order minimization techniques," in *IEEE Int. Conf on Robotics and Automation*, vol. 2, Jan. 2004, pp. 1843 – 1848.

[4] C. Collewet, E. Marchand, and F. Chaumette, "Visual servoing set free from image processing," in *IEEE Int. Conf. on Robotics and Automation*, 2008, pp. 81–86.

[5] J. Lapreste and Y. Mezouar, "A hessian approach to visual servoing," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 1, 2004, pp. 998–1003.

[6] N. Crombez, E. M. Mouaddib, G. Caron, and F. Chaumette, "Servoing With Photometric Gaussian Mixtures as Dense Features," *IEEE Trans. on Robotics*, vol. 35, no. 1, pp. 49–63, Feb. 2019.

[7] Y. Chen, M. Q.-H. Meng, and L. Liu, "Direct visual servoing based on discrete orthogonal moments," *IEEE Trans. on Robotics*, vol. 40, pp. 1795–1812, 2024.

[8] S. Felton, E. Fromont, and E. Marchand, "Deep metric learning for visual servoing: when pose and image meet in latent space," in *IEEE Int. Conf. on Robotics and Automation*, 2023, pp. 741–747.

[9] G. Caron, "Defocus-Based Direct Visual Servoing," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4056–4063, Apr. 2021.

[10] N. Joshi, "Defocus blur," in *Computer Vision: A Reference Guide*, K. Ikeuchi, Ed. Springer Int. Publishing, 2021, pp. 277–279.

[11] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[12] B. Polyak, "Newton's method and its use in optimization," *European J. of Operational Research*, vol. 181, no. 3, pp. 1086–1096, 2007.

[13] M. Ainsworth and Y. Shin, "Plateau phenomenon in gradient descent training of relu networks: Explanation, quantification, and avoidance," *SIAM J. on Scientific Computing*, vol. 43, no. 5, p. A3438, 2021.