On the impact of the camera field-of-view to Direct Visual Servoing robot trajectories when using the Photometric Gaussian Mixtures as dense feature

Sinta Schulte, Antoine André, Nathan Crombez, Guillaume Caron

On the impact of the camera field-of-view to Direct Visual Servoing robot trajectories when using the Photometric Gaussian Mixtures as dense feature

Sinta Schulte[1], Antoine N. André [1], Nathan Crombez[2], Guillaume Caron[1,3]

***Abstract*— This paper studies the impact of cameras with different fields of view (FoV) on Direct Visual Servoing to control robot motions from pixel intensities. Focusing on the Photometric Gaussian Mixture Visual Servoing that showed great convergence domains, this paper investigates two types of FoV: the seminal perspective case and the novel full omnidirectional case. Implemented with our open-source generic software framework `libPeR` for a fair comparison, the Visual Servoing experiments on a 6 degrees-of-freedom robot arm provide an in-depth evaluation of the impact of each FoV on the convergence domain, straightness of the trajectory and time to reach convergence.**

I. INTRODUCTION

Visual Servoing (VS) leverages the information contained in the image to control the movement of a robot and reach a desired position by designing a control law aiming to minimize the error between the current and the desired image [1], [2]. For the past decades, numerous research studies have investigated how to increase the performances of VS schemes, exploring various approaches and camera models.

One approach is Indirect Visual Servoing (IVS), which relies on the extraction of geometric features to compute the control law [3]. While this method ensures a large convergence domain, allowing convergence as long as the detected geometric features are matched in both current and desired images, its major drawback is the feature extraction, thus limiting both accuracy and robustness.

Another approach is Direct Visual Servoing (DVS), which utilizes all the photometric information present in the image. DVS is used as a dense feature to compute the control law, such as the seminal Photometric Visual Servoing (PVS) [4]. Contrary to IVS approaches, DVS methods can perform VS tasks without relying on any geometrical feature present in the studied scene. The main advantage of using the whole image is its robustness and sub-millimetric accuracy at convergence. However, this accuracy comes at the cost of curvy trajectories and a narrower convergence domain.

The latter problem has been widely investigated in a first line of works focusing on the visual information to consider as input of the control law, including deeper image analysis through space changes [5], [6], [7] and by smoothing the images either digitally [8] or optically [9]. Image smoothing, especially when considering Photometric Gaussian Mixtures (PGM), proved to greatly increase the convergence domain without affecting the precision at convergence [8].

Another possibility to enlarge DVS' convergence domain is increasing the camera FoV. Indeed, thanks to the miniaturization of omnidirectional cameras, they have become easier to embed in robotic systems for various purposes including robot attitude estimation, place recognition and dense-scene geometry prediction [10], [11], [12], [13]. VS tasks also benefit from large FoV as demonstrated by gradually enhancing PVS' convergence domain, from using a single panoramic camera [14] to full 360° cameras [15].

Leveraging the above state-of-the-art works, the core contributions of this paper are:

1) combine the two approaches of enhancing DVS' convergence domain to introduce the PGM VS using a 360° camera.
2) propose a new VS quantitative evaluation methodology based on ideal and achieved trajectories.

The first contribution is achieved by considering the so-called equirectangular single image representation of the 360° FoV captured. Thus, we coin the term *equirectangular camera* for the rest of this paper. Furthermore, dedicated attention in both the mathematical and software developments enables switching seamlessly between perspective and equirectangular cameras within the same framework (see Sec. II) for a fair assessment of the impact of the camera FoV to the DVS scheme using the same PGM visual information. The latter comparison follows the methodology of the second contribution to conclude based on a quantitative analysis comparing the achieved robot trajectories to the ideal ones, as defined in Section III. Section IV presents the VS results using perspective and equirectangular cameras on the same 6 Degrees-of-Freedom (DoF) robot arm and Section V concludes the paper.

II. EQUIRECTANGULAR PHOTOMETRIC GAUSSIAN MIXTURES-BASED VISUAL SERVOING

This section first recalls the PGM formulation and the control law regulating to zero the error defined between the PGM of the current image and the PGM of the desired image, including the generic expression of the interaction matrix linking the variation of the error to the one of the camera

*This work was supported by the AIST ITH DVS-Straight project.

[1]CNRS-AIST Joint Robotics Laboratory (JRL), IRL, Tsukuba, Japan. `sinta.schulte@gmail.com`, `antoine.andre@aist.go.jp`

[2]University of Technology Belfort-Montbeliard, CIAD laboratory, Montbeliard, France. `nathan.crombez@utbm.fr`

[3]University of Picardie Jules Verne, MIS laboratory, Amiens, France. `guillaume.caron@u-picardie.fr`

DoFs. After that, the new interaction matrix expressed considering the equirectangular camera is developed.

A. Photometric Gaussian Mixture

Instead of considering each image pixel $I(\mathbf{u}) : \mathcal{U} \subset \mathbb{N}^2 \longmapsto [0, 255] \subset \mathbb{N}$, as a pulse whose magnitude equals the captured intensity, a PGM transforms every pixel pulse as Photometric Gaussians (PG) that spread each pixel's attractive potential to the rest of the image plane. Thus, stacking every pixel of the image I as a vector $\mathbf{I}$ and considering the same Gaussian extent $\lambda_g \in \mathbb{R}_+ \backslash \{0\}$ for every PG, the PG g evaluated at location $\mathbf{u_g} = (u_g, v_g)^T \in \mathcal{U}$ is expressed as:

$$g(\mathbf{I}, \mathbf{u_g}, \mathbf{u}, \lambda_g) = I(\mathbf{u}) E(\mathbf{u_g} - \mathbf{u}), \tag{1}$$

where $E(\mathbf{u_g} - \mathbf{u})$ refers to the Gaussian function:

$$E(\mathbf{u_g} - \mathbf{u}) = \exp\left(-\frac{(u_g - u)^2 + (v_g - v)^2}{2\lambda_g^2}\right). \tag{2}$$

Then, the PGM G of an image I is nothing but the sum of all g:

$$G(\mathbf{I}, \mathbf{u_g}, \lambda_g) = \sum_{\mathbf{u} \in \mathcal{U}} g(\mathbf{I}, \mathbf{u_g}, \mathbf{u}, \lambda_g). \tag{3}$$

The extension parameter λ_g and its effects can be observed as an image *smoothing* factor about which Figure 1 reports example images for two types of cameras, one perspective and one equirectangular.

B. PGM-based Visual Servoing

Following the general Visual Servoing methodology [2], PGM VS (noted short for compactness) aims at regulating to zero the difference between the PGM G of the current image I captured at the camera pose $\mathbf{p} \in \mathbb{R}^6$ with respect to the PGM G^* of the desired image I^* defined at the desired camera pose $\mathbf{p}^* \in \mathbb{R}^6$. In each pose vector, the first three elements are the tridimensional translation and the three last are the tridimensional rotation represented as axis-angle.

Stacking for all $\mathbf{u_g} \in \mathcal{U}$, all current Gaussian mixture elements $G(\mathbf{u_g})$ in a vector $\mathbf{G}(\mathbf{p}) \in \mathbb{R}^{|\mathcal{U}|}$, and desired ones $G^*(\mathbf{u_g}^*)$ in a vector $\mathbf{G}^*(\mathbf{p}^*) \in \mathbb{R}^{|\mathcal{U}|}$, we express the cost function between desired and current dense features [8]:

$$C(\mathbf{p}) = \frac{1}{2} ||\mathbf{G}(\mathbf{p}) - \mathbf{G}^*(\mathbf{p}^*)||^2. \tag{4}$$

To minimize (4) with PGM VS, we highlight the so-called interaction matrix $\mathbf{L_G} \in \mathbb{R}^{|\mathcal{U}| \times 6}$ relating the time derivative of the dense features $\mathbf{G}(\mathbf{p})$ to the camera velocity $\mathbf{v} \in \mathbb{R}^6$:

$$\dot{\mathbf{G}} = \mathbf{L_G} \mathbf{v}, \tag{5}$$

where $\mathbf{p}$ has been dropped from $\mathbf{G}(\mathbf{p})$ for conciseness.

Finally, considering a gain $\mu \in \mathbb{R}_+$ and the matrix pseudo-inverse operator $^+$, we express the control law to compute the velocity that the camera should follow in order to minimize (4) with an exponential decay:

$$\mathbf{v} = -\mu \, {\mathbf{L_G}}^+ (\mathbf{G}(\mathbf{p}) - \mathbf{G}^*), \tag{6}$$

where $\mathbf{p}^*$ has been dropped from $\mathbf{G}^*(\mathbf{p}^*)$ for conciseness.

In order to highlight the role of the camera projection model, let us remind that the interaction $\mathbf{L_G}$ is the stacking of all interaction matrices $\mathbf{L}_G(\mathbf{u_g})$ for all $\mathbf{u_g} \in \mathcal{U}$, each of them being analytically expressed thanks to the derivatives chain rule as a product Jacobian matrices [8]:

$$\mathbf{L}_G(\mathbf{u_g}) = \frac{\partial G}{\partial \mathbf{u_g}} \frac{\partial {\mathbf{u_g}}^{(\{p,e\})}}{\partial \mathbf{X}} \mathbf{L_X}, \tag{7}$$

where $\mathbf{X} \in \mathbb{R}^3$ is the tridimensional point expressed in the camera coordinate system projecting in $\mathbf{u_g}$ according to the camera projection model (denoted in (7), noted with the exponent $^{(p)}$ for perspective camera and $^{(e)}$ for equirectangular camera, as described in Sec. II-C) and $\mathbf{L_X}$ is the interaction matrix related to $\mathbf{X}$. We refer to [2] for the details to obtain the expression of $\mathbf{L_X}$ as:

$$\mathbf{L_X} = \begin{bmatrix} -1 & 0 & 0 & 0 & -Z & Y \\ 0 & -1 & 0 & Z & 0 & -X \\ 0 & 0 & -1 & -Y & X & 0 \end{bmatrix}, \tag{8}$$

and to [8] for the expression of $\dfrac{\partial G}{\partial \mathbf{u_g}}$:

$$\frac{\partial G}{\partial \mathbf{u_g}} = -\sum_{\mathbf{u} \in \mathcal{U}} I(\mathbf{u}) \begin{bmatrix} -(u_g - u)/\lambda_g^2 \\ -(v_g - v)/\lambda_g^2 \end{bmatrix} E(\mathbf{u_g} - \mathbf{u}). \tag{9}$$

The next subsection details the expressions for $\dfrac{\partial {\mathbf{u_g}}^{(\{p,e\})}}{\partial \mathbf{X}}$.

C. Partial derivatives of camera models

As this article mainly focuses on the impact of the FoV on DVS, let us consider the extreme cases of the conventional perspective camera and of the widest field-of-view that is the equirectangular camera of $360°$ field-of-view. For the classical perspective case, a tridimensional point $\mathbf{X} = (X, Y, Z)^T$ is projected on the normalized image plane as $\mathbf{x} = (x, y)^T \in \mathbb{R}^2$ with:

$$\begin{cases} x = X/Z \\ y = Y/Z \end{cases}. \tag{10}$$

Then, considering the perspective camera projection model p, ${\mathbf{u_g}}^{(p)}$ is obtained from $\mathbf{x}$ using the camera principal point $u_0, v_0 \in \mathbb{R}^2$, focal length $f \in \mathbb{R}_+$ and photodiode sides $k_u \in \mathbb{R}_+$ and $k_v \in \mathbb{R}_+$:

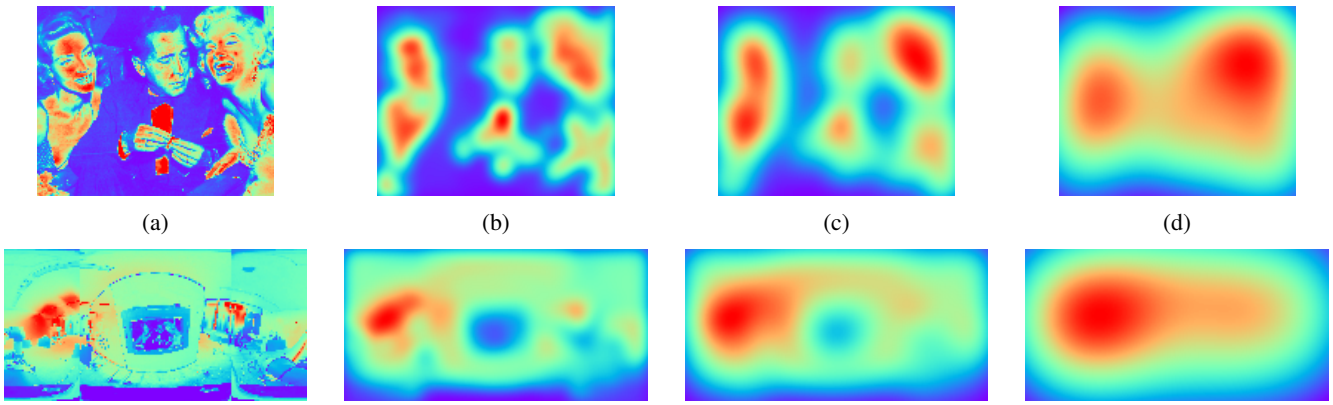
$$\begin{cases} u_g^{(p)} = \frac{f}{k_u} x + u_0 = \alpha_u x + u_0 \\ v_g^{(p)} = \frac{f}{k_v} y + v_0 = \alpha_v y + v_0 \end{cases}, \tag{11}$$

where intrinsic parameters α_u, α_v, u_0, v_0 are generally obtained with camera calibration.

Thus, we develop the Jacobian $\dfrac{\partial {\mathbf{u_g}}^{(p)}}{\partial \mathbf{X}}$ in (7) to:

$$\frac{\partial {\mathbf{u_g}}^{(p)}}{\partial \mathbf{X}} = \frac{\partial {\mathbf{u_g}}^{(p)}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{X}}, \tag{12}$$

where it is trivial to express $\dfrac{\partial \mathbf{x}}{\partial \mathbf{X}}$, resp. $\dfrac{\partial {\mathbf{u_g}}^{(p)}}{\partial \mathbf{x}}$, from (10), resp. (11) (see [2]).



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 1: Impact of λ_g = [0.1, 5, 10, 25] on gray scale picture acquired by (a-d) perspective and (e-h) panoramic camera

Then, for the equirectangular camera, since its image plane represents the azimuth $\theta \in [-\pi, \pi]$ and elevation $\phi \in [-\pi/2, \pi/2]$ angles of $\mathbf{X}$, let us express:

$$\begin{cases} \theta = \arctan \frac{X}{Z} \\ \phi = \arctan \frac{Y}{\sqrt{X^2+Y^2}} \end{cases} . \tag{13}$$

We transform θ and ϕ to the digital image of width $W \in \mathbb{N}$ and height $H \in \mathbb{N}$ to finalize the equirectangular projection model expressing $\mathbf{u_g}^{(e)}$ as:

$$\begin{cases} u_g^{(e)} = \frac{W}{2\pi}\theta + \frac{W}{2} = \epsilon_u \theta + u_e \\ v_g^{(e)} = \frac{H}{\pi}\phi + \frac{H}{2} = \epsilon_v \phi + v_e \end{cases} , \tag{14}$$

where u_e and v_e are used for convenience.

Finally, noting $\boldsymbol{\theta} = [\theta, \phi]^T$, we develop the Jacobian $\frac{\partial \mathbf{u_g}^{(e)}}{\partial \mathbf{X}}$ in (7) to:

$$\frac{\partial \mathbf{u_g}^{(e)}}{\partial \mathbf{X}} = \frac{\partial \mathbf{u_g}^{(e)}}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial \mathbf{X}}, \tag{15}$$

which details, thanks to the partial derivatives of (13), resp. (14), as:

$$\frac{\partial \boldsymbol{\theta}}{\partial \mathbf{X}} = \begin{bmatrix} \frac{Z}{X^2+Z^2} & 0 & -\frac{X}{X^2+Z^2} \\ -\frac{XY}{\sqrt{X^2+Z^2}\rho^2} & \frac{\sqrt{X^2+Z^2}}{\rho^2} & -\frac{YZ}{\sqrt{X^2+Z^2}\rho^2} \end{bmatrix}, \tag{16}$$

with $\rho^2 = X^2 + Y^2 + Z^2$, resp.:

$$\frac{\partial \mathbf{u_g}^{(e)}}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \epsilon_u & 0 \\ 0 & \epsilon_v \end{bmatrix}. \tag{17}$$

III. Visual Servoing Trajectory Evaluation

Contrary to Visual Odometry (VO) and Simultaneous Localization And Mapping (SLAM) that have adopted standard evaluation routines of estimated camera trajectories [16], DVS lacks standard metrics and tools to compare robot trajectories achieved by control laws beyond the initial pose error, the accuracy at convergence and time to reach the desired pose. Evaluations of VO and SLAM use metrics such as differences in trajectory lengths, Absolute Pose Error (APE), Relative Pose Error (RPE), and variants. But their implementation within tools such as the `evo` package [17], which has become a *de facto* standard, need a reference trajectory, *i.e.* an ordered set of timestamped ground truth poses. However, in the context of a DVS experiment, there is no ground truth reference camera trajectory. We therefore introduce a procedure for generating a sequence of reference poses based on the desired pose $\mathbf{p}^*$, the initial pose $\mathbf{p}_{init}$, and the total number of iterations N of a completed VS experiment. By doing so, VO and SLAM evaluation metrics and tools become available for fair quantitative and qualitative evaluations of VS.

Thus, we consider the straight line between the initial and the desired positions as the reference trajectory, *i.e.* the optimal Cartesian trajectory that would be nice to achieve even if the visual error in DVS is not explicitly defined to achieve a straight line in the tridimensional space. This trajectory is discretized into N reference poses $\mathbf{p}_{ref}$ as follows:

$$\mathbf{p}_{ref_i} = \mathbf{p}_{init} + t_i(\mathbf{p}^* - \mathbf{p}_{init}) \ , i \in \{0, ..., N-1\}, \tag{18}$$

where the poses' orientation are quaternions computed from the axis-angle representation, and where vector $\mathbf{t} = [..., t_i, ...]^T \in [0, 1]^N$ contains N interpolation factors evenly spaced from 0 to 1 on a logarithmic scale in order to match the exponential decay that the VS control law (6) aims at achieving. The resulting set of poses is considered as the reference trajectory of the studied VS experiment, allowing its evaluation and analysis using the various tools included, for instance, in the `evo` package (*e.g.* see Fig. 2).

This proposed protocol is shared open-source as `evs` software[1].

IV. Comparison Structure

A. Experimental Setup

The experiments are performed with an UR5e industrial collaborative robot, with the cameras attached one-by-one to its end-effector (Fig. 3). The C-mount Flir Flea3 USB3 (FL3-U3-13E4C) camera with a 1/1.8" sensor and 8 mm focal length Computar lens M0814MP2, as well as the dual-fisheye

[1] `https://github.com/NathanCrombez/evs`

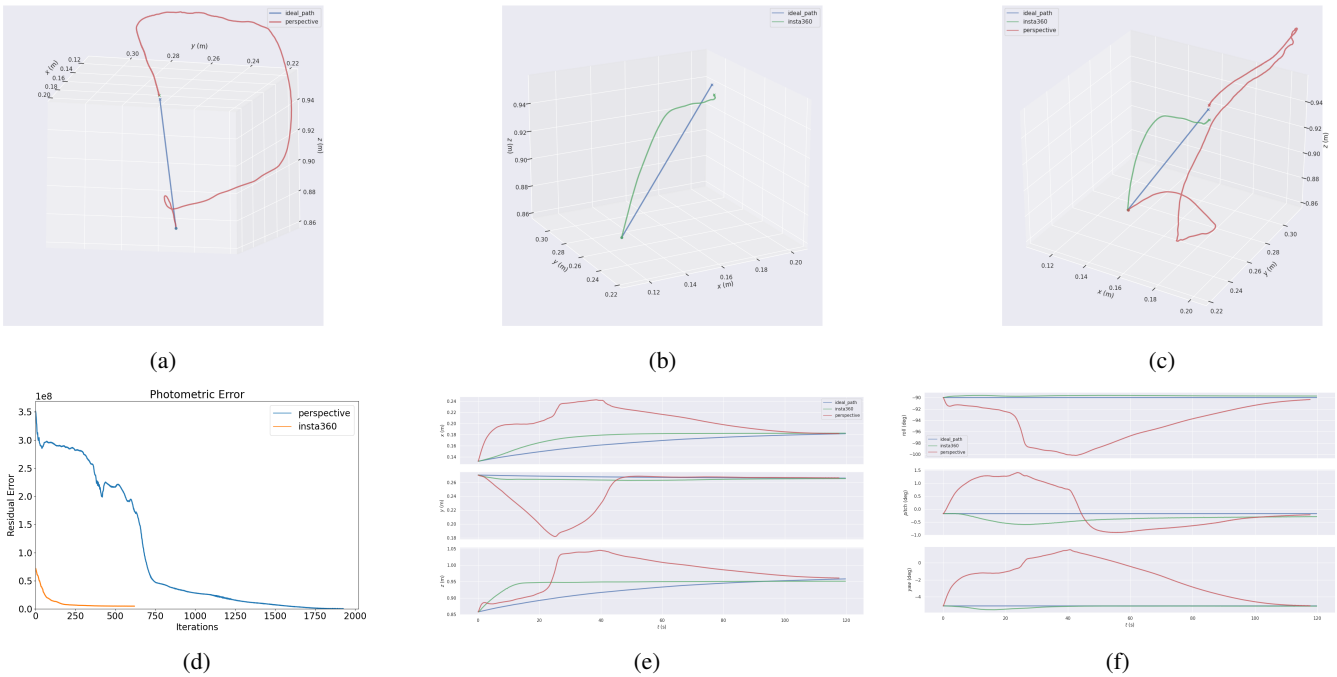


Fig. 2: Comparison results for PGMVS tasks using perspective and 360°cameras. Subfigures (a) and (b) show PGM VS trajectories with identical initial and desired positions for the Flea3 and Insta360 (blue lines are the ideal reference trajectory). Subfigure (c) plots together trajectories with the perspective (red) and the 360°(green) camera and (d) shows the corresponding PGM cost over iterations. Corresponding xyz (e) and rpy (f) elements of robot end-effector poses throughout the trajectories.

Insta360 ONE X2 camera are utilized. Thus, the Flea3 camera captures images with a FoV of 51.9° (diagonal), 42.5° (horizontal) and 32.4° (vertical). The Insta360 camera offers a 360° FoV.

On the software side, the C++ open-source robot perception library `libPeR`[2] is used to compute the visual features and the control laws, in combination with `ROS` for integration.

B. Experimental results

For the first series of experiments, PGM VS with each camera is using identical parameters, with λ_g set to 0.805 (except differently stated), the gain μ to 0.8, and a scene depth Z of 0.5 m corresponding approximately to the distance between the camera and the printed picture (Fig. 3) very common in many DVS articles. The experimental setup is designed such that the capturing of the desired image for the perspective camera has said picture in the frame. The initial displacements in Table I are defined with gradual difficulty such that the first 10 lead to an overlap between the desired and initial images for the Flea3 camera, while the last two have none. To provide the robot with velocity vectors at the same rate as the camera capture frequency and with minimum latency, the images are resized to a resolution of 80×64 pixels for the Flea3 and 184×92 pixels for the Insta360. Such low image resolutions are known to not prevent accuracy at convergence with the PGM feature [8].

[2]https://github.com/PerceptionRobotique/libPeR_base

#	Initial displacement	Ideal	Actual (persp.)	Actual (360)
1	(0cm, 0.5cm, 0.5cm, 0°, 0°, 0°)	0.7cm	+1.1	+0.4
2	(5cm, 5cm, 0cm, 0°, 0°, 0°)	7.1cm	+16.8	+5.8
3	(2cm, 6cm, 5cm, 0°, 0°, 0°)	8.1cm	+24.4	+6.2
4	(-6cm, 5cm, 2cm, 0°, 0°, 0°)	8.1cm	+8.1	+2.1
5	(0cm, 5cm, -10cm, 0°, 0°, 0°)	11.2cm	+34.2	+1.5
6	(0cm, 0cm, 0cm, 15°, 0°, 0°)	0cm	+6.1	+14.7
7	(0cm, 0cm, 1cm, 10°, 10°, 0°)	1cm	x	+5.8
8	(0cm, -5cm, -5cm, 0°, 10°, 10°)	7.1cm	x	+8.8
9	(-4cm, -5cm, 2cm, 0°, 0°, -20°)	6.7cm	x	+16.2
10	(3cm, 2cm, 5cm, 0°, 5°, 10°)	6.2cm	x	+2.9
11	(5cm, 10cm, 20cm, 0°, 0°, 0°)	22.9cm	x	+10.9
12	(0cm, -20cm, 50cm, 0°, 20°, 0°)	53.4cm	x	+91.1

TABLE I: Initial displacement in robot end-effector frame, ideal path length to convergence, additional path length in cm (diverging trajectory noted x).

For the perspective camera, the visual servoing relies exclusively on the content of the desired image. This fact can cause divergence if the initial pose leads to insufficient

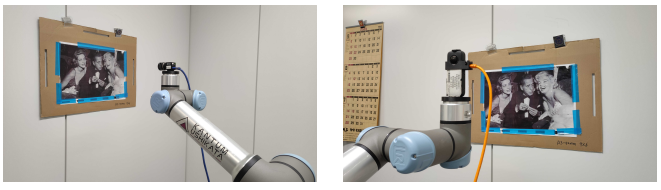


Fig. 3: UR5e experimental setup with Flea3 and Insta360 camera mounted to its endeffector

overlap with the desired image content. The experiments demonstrate that this camera type generates curvy trajectories, longer (third column of Tab. I) than the ideal reference trajectories (second column of Tab. I) computed as described in Section III.

In contrast, the equirectangular camera captures a comprehensive view of the entire scene, thus utilizing a broader range of visual information. Hence, this type of camera shows a larger convergence domain than the perspective one, successfully converging for all initial poses considered. The perspective camera successfully converges only in six out of twelve trials, with the last two experiments diverging as predicted due to a lack of visual overlap. Notably, for the experiment involving a pure rotational initial error, the trajectory path for the perspective camera is shorter than that of the 360-degree camera. However, this is the only instance in which the perspective camera reaches convergence under rotational conditions, highlighting its limited robustness in handling rotations.

For the fifth experiment in Tab. I, Fig. 2 visualizes the trajectories, residuals and velocities. The trajectory length to convergence for the perspective camera case is about 3.5 times longer than for the equirectangular camera and it can be observed that the latter follows a straighter path. Fig. 2d displays the PGM cost (4) with respect to iterations. Comparing the desired and initial view of the two cameras in Fig. 4 and 5 explains the higher residuals of the Flea3 at the beginning of the VS task. The residuals for the Insta360 demonstrate a smooth exponential decay, whereas the perspective camera, even overall showing a decrease of the cost, is not monotonous and requires a higher amount of iterations to converge.

Extending a bit the fifth experiment of Tab. I, instead of solely setting $\lambda_g = 0.805$, we consider a recommendation of [8] that introduced a second run of PGM VS with smaller λ_g to ensure accuracy at convergence. When finishing the PGM VS with $\lambda_g = 0.5$, the final error changes from a Euclidean position and angular distance of error $e = (2.69mm, 6.1mrad)$ to $e = (0.28mm, 0.6mrad)$ for the perspective camera, respectively from $e = (7.12mm, 5mrad)$ to $e = (1.7mm, 1.9mrad)$ for the 360-degree camera.

Fig. 4: Flea3: desired and initial view

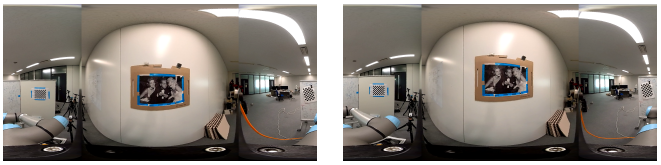


Fig. 5: Insta360: desired and initial view

V. Conclusions

In this paper, we investigated the impact of the camera field-of-view on Direct Visual Servoing, leveraging the Photometric Gaussian Mixtures image representation. A fair comparison between perspective and omnidirectional cameras to control the motion of a robot arm has been conducted thanks to the novel evaluation approach of achieved trajectories with respect to computed ideal ones. After recalling the design of VS control laws, we presented an experimental setup and protocol to quantify the performance differences between both small ($51°$) and wide ($360°$) field-of-view.
In particular, this study demonstrated straighter trajectories with an omnidirectional camera than with a perspective one. Further studies will investigate the impact of other field-of-view and of other types of dense visual features, such as Mixtures of Photometric Potentials which already proved to be a strong candidate to widen the convergence domain of visual alignment tasks.

References

[1] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, 1992.

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

[3] N. Andreff, B. Espiau, and R. Horaud, "Visual servoing from lines," *The International Journal of Robotics Research*, vol. 21, no. 8, pp. 679–699, 2002.

[4] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 828–834, 2011.

[5] E. Marchand, "Subspace-based direct visual servoing," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2699–2706, 2019.

[6] Q. Bateux and E. Marchand, "Histograms-based visual servoing," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 80–87, 2016.

[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] N. Crombez, E. M. Mouaddib, G. Caron, and F. Chaumette, "Visual servoing with photometric gaussian mixtures as dense features," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 49–63, 2018.

[9] G. Caron, "Defocus-based direct visual servoing," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4056–4063, 2021.

[10] J. D. Adarve and R. Mahony, "Spherepix: A data structure for spherical image processing," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 483–490, 2016.

[11] G. Caron and F. Morbidi, "Spherical visual gyroscope for autonomous robots using the mixture of photometric potentials," in *IEEE Int. Conference on Robotics and Automation*, 2018, pp. 820–827.

[12] J. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *IEEE conference on computer vision and pattern recognition*, 2012, pp. 2695–2702.

[13] N. Zioulis, A. Karakottas, D. Zarpalas, and P. Daras, "Omnidepth: Dense depth estimation for indoors spherical panoramas," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 448–465.

[14] G. Caron, E. Marchand, and E. M. Mouaddib, "Photometric visual servoing for omnidirectional cameras," *Autonomous Robots*, vol. 35, pp. 177–193, 2013.

[15] N. Crombez, J. Buisson, A. N. André, and G. Caron, "Dual-hemispherical photometric visual servoing," *IEEE Robotics and Automation Letters*, 2024.
[16] D. Sharafutdinov, M. Griguletskii, P. Kopanev, M. Kurenkov, G. Ferrer, A. Burkov, A. Gonnochenko, and D. Tsetserukou, "Comparison of modern open-source visual slam approaches," *J/ Intell Robot Syst*, vol. 107, 2023.
[17] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.