2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Deep metric learning for visual servoing: when pose and image meet in latent space

Samuel Felton, Élisa Fromont, Eric Marchand

Abstract— **We propose a new visual servoing method that controls a robot's motion in a latent space. We aim to extract the best properties of two previously proposed servoing methods: we seek to obtain the accuracy of photometric methods such as Direct Visual Servoing (DVS), as well as the behavior and convergence of pose-based visual servoing (PBVS). Photometric methods suffer from limited convergence area due to a highly non-linear cost function, while PBVS requires estimating the pose of the camera which may introduce some noise and incurs a loss of accuracy. Our approach relies on shaping (with metric learning) a latent space, in which the representations of camera poses and the embeddings of their respective images are tied together. By leveraging the multimodal aspect of this shared space, our control law minimizes the difference between latent image representations thanks to information obtained from a set of pose embeddings. Experiments in simulation and on a robot validate the strength of our approach, showing that the sought out benefits are effectively found.**

I. Introduction

Visual servoing (VS) is the task of controlling the motion of a robot in order to reach a desired goal or a desired pose using only visual information extracted from an image stream [7]. The camera can be mounted on the robot's end effector or directly observing the robot. Visual servoing usually requires the extraction and the tracking of visual information (usually geometric features) from the image in order to design the control law. The choice of features is a crucial aspect of VS, as it impacts the servoing behaviour (namely the convergence to the target pose and the trajectory in 3D space). Geometric features are usually split into two categories. The first one, Image-Based VS (IBVS), uses 2D primitives in the image space, such as points [28], lines [1] or moments [6], in order to create the control law. The second category, named Pose-Based VS (PBVS) [29], [6], uses image information to estimate the camera pose, which can be used to directly control the robot's motion. The PBVS control law is often seen as the most practical and optimal one: if the camera pose is perfectly estimated, then the scheme is globally convergent and the trajectory is the shortest path to the goal pose, both in translation and rotation.

In all cases, geometric features must be extracted, as well as matched between current and desired images. As this is an error-prone process, another way of performing VS has developed. By using photometric features, such as raw pixel intensities, feature extraction is avoided [9]. In Direct VS (DVS), the difference between pixel intensities is minimized, which leads to very accurate positioning, but with an unpredictable trajectory and a small convergence domain, since the cost function to minimize is highly non-linear. To alleviate the latter problem, a solution is to represent images by lower dimensional features that better correlate to the pose. Multiple representations have been studied: [2] expresses an image with photometric moments. In [22], servoing is performed in the frequency domain, where only the smoothly varying low-frequency information is preserved. A similar, learning-based approach was proposed in [21], where principal component analysis is used to project on a subspace, which maximally preserves the information of an image set. Inspired by this work, we previously proposed AEVS [11], projecting images in the latent space of an autoencoder.

Between all these VS methods, a trade-off becomes apparent: the easier feature extraction is, the harder servoing becomes. On one end of the spectrum lies DVS, with no extraction but a highly non-linear cost function. On the other end, PBVS requires estimating the pose (from a 3D model of the scene/object or directly from data) but the cost function is smooth. Pose estimation also has the drawback of introducing some noise into the robot's trajectory. However, as camera relocalization is a fundamental task of many computer vision applications, it has become a topic of interest for the deep learning community, that seeks to replace or support classical geometric approaches with a neural network. One of the first works was PoseNet [17], that uses a Convolutional Neural Network (CNN) to regress the camera pose in a scene (position + orientation) from a single RGB image. This process was repurposed for VS, with multiple works such as [4], [24], [31] which employ a CNN to estimate the pose difference between current and desired images. This difference is then fed to the PBVS control law in order to move the robot closer to the target pose. Servoing is then fully dependent on this estimation, and [33] notes that in the case of orientation regression, trained networks may still yield a large error, as the rotation space is not smooth and continuous. Furthermore, In some cases, the difference between translational and rotational motions may be nonobservable.

Our work proposes to replace the standard pose regression approach with **metric learning**. In metric learning, we seek to learn a similarity function between samples that is dependent on the task to be accomplished. These approaches can be used to group images of the same concept (high similarity), while pushing away images of different concepts (low similarity). This principle can be applied to many tasks, including classification [13], object tracking [10] or camera

Authors are with Univ Rennes, Inria, CNRS, Irisa, Rennes, France Email: {samuel.felton, elisa.fromont, eric.marchand}@irisa.fr

relocalization [3].

We propose to use metric learning to shape the latent space in which we perform VS. We aim to learn a space in which the similarity between two learned representations correlates with the distance between their respective camera poses. This representation space can be seen as an intermediate manifold between poses and images on which we project both modalities. Doing so, we propose a new servoing control law that combines the optimal behavior of PBVS with the accuracy and simplicity of photometric methods. In Section II, we give an overview of VS, detailing the generic minimization framework, as well as presenting how PBVS and visual servoing in an autoencoder latent space are achieved. With this knowledge, we introduce our method in Section III, that constrains the latent space to have good properties for VS via metric learning. By leveraging the multimodal aspect of this shared space, our control law minimizes image error thanks to information obtained from a set of pose projections. Finally, In Section IV, we present simulated and real-world experiments that validate our approach and illustrate its properties.

II. Related works

A. Visual servoing framework

Visual servoing aims to reach a desired pose $\mathbf{r}^*$, from its arbitrary, current pose $\mathbf{r}$. Many robotics tasks, such as navigation, tracking, object picking can be viewed through this prism, e.g. navigation is a succession of positioning tasks. VS thus seeks to solve an optimization problem, finding the pose closest to $\mathbf{r}^*$ that minimizes an error function $\mathbf{e}$:

$$\widehat{\mathbf{r}} = \arg\min_{\mathbf{r}} \mathbf{e}(\mathbf{r}). \tag{1}$$

If $\mathbf{e}$ is well designed and $\mathbf{e} = \mathbf{0}$, then $\widehat{\mathbf{r}} = \mathbf{r}^*$. Since the pose $\mathbf{r}$ may be unknown during servoing, $\mathbf{e}$ is defined as $\mathbf{e} = \mathbf{s}(\mathbf{r}) - \mathbf{s}^*$, which is the difference between what the camera sees at the current pose $\mathbf{s}(\mathbf{r}) = \mathbf{s}$, and what it should see at the desired pose $\mathbf{s}^*$. As stated before, the choice of features $\mathbf{s}$ is important as it conditions the 3D trajectory, the convergence domain — how far can $\mathbf{r}$ be from $\mathbf{r}^*$ before VS diverges — and the final accuracy — how close is $\widehat{\mathbf{r}}$ to $\mathbf{r}^*$. Features may lie in the image space (IBVS), in the 3D world (PBVS) or in photometric space (e.g. considering pixel intensities [9]). In all cases, the relationship between the variation (in time) of the features $\mathbf{s}$ and the camera velocity $\mathbf{v}$ must be established:

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

where $\mathbf{L}_{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}$ is called the interaction matrix. By inverting this equation, we can then define the control law that best minimizes the error $\mathbf{e}$:

$$\mathbf{v} = -\lambda \mathbf{L}_{\mathbf{s}}^{+}\mathbf{e} \tag{3}$$

where λ is a gain parameter $\mathbf{L}_{\mathbf{s}}^{+}$ the pseudo-inverse of $\mathbf{L}_{\mathbf{s}}$. The computed velocity $\mathbf{v}$ can be used to move the robot's end-effector closer to the desired pose $\mathbf{r}^*$. VS operates in a closed loop, with the minimization of $\mathbf{e}$ being performed in an iterative manner.

B. Deep learning for pose-based VS

A pose $\mathbf{r} = \begin{pmatrix}\mathbf{t} & \theta\mathbf{u}\end{pmatrix}^\top$ is an element of $\mathrm{SE}(3) = \mathbb{R}^3 \times \mathrm{SO}(3)$ the group that represents rigid transformations, combining translation $\mathbf{t}$ and rotation $\theta\mathbf{u}$, where $\mathbf{u}$ is the axis around which to rotate and θ is the angle of the rotation. It may also be represented as a homogeneous matrix ${}^{o}\mathbf{T}_{c}$, that expresses the pose of camera $\mathcal{F}_c$ (or any other frame) in a reference frame $\mathcal{F}_o$ (such as one given by the origin of a scene object o). The displacement between two poses $\Delta\mathbf{r} = {}^{c^*}\mathbf{T}_{c}$ can be computed as ${}^{c^*}\mathbf{T}_{c} = {}^{c^*}\mathbf{T}_{o}\,{}^{o}\mathbf{T}_{c}$.

Deep Learning (DL) has previously been used to estimate the camera pose $\mathbf{r}$, given a single image $\mathbf{I}$. The first major work to accomplish this was PoseNet [17]. Given a dataset of images and their associated poses (expressed in a common frame), a CNN is trained to minimize the following loss function:

$$\mathcal{L}_{pose}(\mathbf{t}, \mathbf{q}) = \|\hat{\mathbf{t}} - \mathbf{t}\|^2 + \beta\|\hat{\mathbf{q}} - \mathbf{q}\|^2 \tag{4}$$

where $\hat{\mathbf{t}}, \hat{\mathbf{q}}$ are the predicted position and orientation of the camera and $\mathbf{t}, \mathbf{q}$ is the ground truth. The orientation $\mathbf{q}$ can be expressed as a unit quaternion, or as an axis-angle $\theta\mathbf{u}$. β is an important hyperparameter, that balances the learning of both translation and orientation. This weighting is required as the two quantities are on different scales, and it must carefully be tweaked in order to get sensible results. In [16], the authors introduced a multi-task loss that automatically learns the weighting, improving upon the manual tuning of β. Given two images, it is also possible to regress the pose difference $\Delta\mathbf{r}$ [4], [24], [31]. $\Delta\mathbf{r}$ can then be plugged into the PBVS control law [29], [7] to compute $\mathbf{v}$. The interaction matrix of a pose expressed in a fixed frame $\mathcal{F}_{c^*}$ (also valid for $\mathcal{F}_o$) is defined as [7]:

$$\mathbf{L}_{\mathbf{r}} = \begin{pmatrix} {}^{c^*}\mathbf{R}_{c} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{\theta\mathbf{u}} \end{pmatrix} \tag{5}$$

with ${}^{c^*}\mathbf{R}_{c}$ a rotation matrix and $\mathbf{L}_{\theta\mathbf{u}}$ the interaction matrix defined in [8]. If $\Delta\mathbf{r}$ is perfectly estimated, then the control law $\mathbf{v} = -\lambda\mathbf{L}_{\mathbf{r}}^{-1}\Delta\mathbf{r}$ ensures that the 3D trajectory is a geodesic both in translation and rotation.

C. Servoing in latent space

In [11], we introduced AEVS, a method to perform VS in the latent space of an autoencoder (AE). This AE is a neural network that learns a projection from an image to a lower dimensional representation (encoder), as well as the inverse mapping (decoder). This approach is similar to PCA-based VS [21], except that AEs learn non-linear projections, which PCA cannot. The autoencoding objective is the minimization of the reconstruction error. Considering two embeddings $\mathbf{z}_{\mathbf{I}}, \mathbf{z}_{\mathbf{I}^*}$, the control law is of the form:

$$\mathbf{v} = -\lambda\mathbf{L}_{\mathbf{z}_{\mathbf{I}}}^{+}(\mathbf{z}_{\mathbf{I}} - \mathbf{z}_{\mathbf{I}^*}) \tag{6}$$

where $\mathbf{L}_{\mathbf{z}_{\mathbf{I}}}$ is computed analytically by applying the chain rule, finding $\mathbf{L}_{\mathbf{z}_{\mathbf{I}}}$ is the composition of the encoder Jacobian and the interaction matrix of the input image, detailed in [9], [20]: $\mathbf{L}_{\mathbf{z}_{\mathbf{I}}} = \frac{\partial \mathbf{z}_{\mathbf{I}}}{\partial \mathbf{I}}\frac{\partial \mathbf{I}}{\partial \mathbf{r}}$. Although this process is applied to

images, the same reasoning can be applied to an encoder for any type of inputs, as long as the interaction matrix of the input can be defined.

While this approach improves upon other photometric dimensionality reduction schemes, it has some drawbacks. First, the training objective is weakly correlated to pose estimation. This makes it hard to know whether the latent space of a trained AE will give good results when applied to VS. Second, the interaction matrix $\mathbf{L}_{\mathbf{z}_\mathbf{I}}$ depends on the interaction matrix of DVS, which is known to lead to unpredictable trajectories due to a highly non-linear cost function [9]. Finally, AEVS requires the camera's intrinsic calibration, as well as an estimate of the depth: a very coarse estimation works, but degrades the trajectory.

D. Metric learning

Metric learning aims to learn a similarity function between two compared inputs. The similarity measure is defined not in the input space (e.g. comparing pixel intensities) but rather from the factors that underlie the variations of the data. As an example, metric learning may learn to group images of the same subject (e.g. class, landmark or person) while enforcing a large margin between dissimilar concepts. Metric learning is often coupled to nearest neighbor search. A majority of the metric learning algorithms focuses on binary supervision to learn a meaningful metric: either two samples are similar or they are not. This is best seen in the common triplet loss [13] $\mathcal{L}_{triplet}(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n) = max(d(\mathbf{x}_a, \mathbf{x}_p) - d(\mathbf{x}_a, \mathbf{x}_n) + \epsilon, 0)$ that compares an anchor representation $\mathbf{x}_a$ with similar and dissimilar samples $\mathbf{x}_p, \mathbf{x}_n$, where ϵ is a margin parameter that gives the separation threshold between positive and negative data points. It is also possible to learn smoothly varying similarity functions with continuous metric learning, as studied in [18], [3]. An application of continuous metric learning that is closely tied to our work can be found in [3]. This approach learns a feature space where the Euclidean distance between two representations is directly correlated with the overlap between two images, i.e. how much of the scene is visible in both cameras. An additional pose difference regressor is then trained to obtain a better estimate for the camera relocalization task. This network compares the query with its nearest neighbors. Another work closely related to ours is [15] which learns a feature space that is equivariant to camera motion. This approach uses discrete metric learning and a discrete number of motion patterns (e.g. move forward, rotate left/right) is learned.

Using latent representations also allows projecting multiple modalities of the same data to compare them with a single metric, in a common space. In [27], authors transform text and images and map them to a shared space. Metric learning then allows for the retrieval of images that match a given textual query. Similarly, [25] embeds point clouds, textual tasks, and trajectories in the same space, so that the best fitting trajectory may be selected given a task. Multiple modalities can also be used in audio processing, by either creating a music sample-tag association [30], or an acoustic-linguistic relationship [26].

III. Metric learning for visual servoing

This section presents our novel approach to Visual Servoing, that leverages deep metric learning in order to frame VS in a learned space. We first detail the reasoning behind our method, then present our training procedure, that shapes the latent space. Finally, we describe how VS is performed in this new latent space.

A. Objective

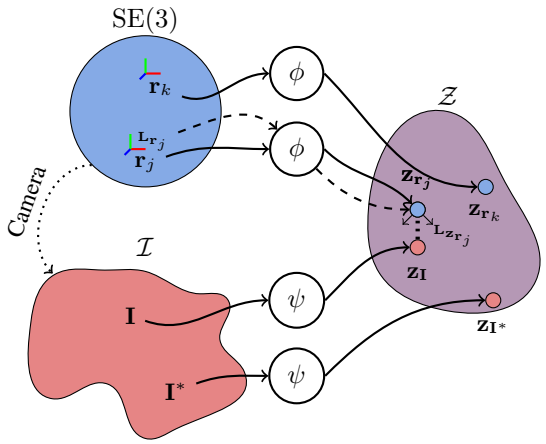


Fig. 1: The proposed latent space for visual servoing. Both images and poses are projected in $\mathcal{Z}$, where they can be compared.

In this paper, our goal is to propose a latent space servoing scheme with a behavior similar to PBVS, overcoming the limitations of other deep learning-based methods. To do so, we propose to create a multimodal latent space $\mathcal{Z}$, in which both pose and image representations are mapped. A pose $\mathbf{r} \in \mathrm{SE}(3)$ maps to an image $\mathbf{I} \in \mathcal{I}$ via a camera. A pose $\mathbf{r}_j$ maps to an embedding $\mathbf{z}_{\mathbf{r}_j}$, while the image acquired at $\mathbf{r}_j$ is noted $\mathbf{z}_{\mathbf{I}_j}$. The relationship between latent space and images/poses is illustrated in Figure (1) We argue that for the best VS behavior, the distance between two embeddings should be equal to the distance between their underlying poses: $d_\mathcal{Z}(\mathbf{z}_j, \mathbf{z}_k) = d_{\mathrm{SE}(3)}(\mathbf{r}_j, \mathbf{r}_k)$, where $d_\mathcal{Z}$ is the Euclidean distance:

$$d_\mathcal{Z}(\mathbf{z}_j, \mathbf{z}_k) = \|\mathbf{z}_j - \mathbf{z}_k\|_2 \tag{7}$$

The distance constraint holds true, whether $\mathbf{z}_j, \mathbf{z}_k$ are image or pose projections, i.e. $\mathbf{z}_j = \mathbf{z}_{\mathbf{I}_j}$ or $\mathbf{z}_j = \mathbf{z}_{\mathbf{r}_j}$. If this property is perfectly met, it follows that:

- for a given image $\mathbf{I}_j$, acquired at pose $\mathbf{r}_j$, $d_\mathcal{Z}(\mathbf{z}_{\mathbf{I}_j}, \mathbf{z}_{\mathbf{r}_j}) = 0$. This is similar to an absolute pose regression objective;
- for two images $\mathbf{I}_j, \mathbf{I}_k$ acquired at $\mathbf{r}_j, \mathbf{r}_k$, the constraint $d_\mathcal{Z}(\mathbf{z}_{\mathbf{I}_j}, \mathbf{z}_{\mathbf{I}_k}) = d_{\mathrm{SE}(3)}(\mathbf{r}_j, \mathbf{r}_k)$ is akin to estimating the relative pose difference between $\mathbf{r}_j$ and $\mathbf{r}_k$ from the images;
- finally, $d_\mathcal{Z}(\mathbf{z}_{\mathbf{I}_j}, \mathbf{z}_{\mathbf{I}_k}) = d_\mathcal{Z}(\mathbf{z}_{\mathbf{r}_j}, \mathbf{z}_{\mathbf{r}_k})$. As the two cost functions are the same, using the interaction matrix

linked to a pose representation in a servoing context is valid for the minimization of $d_{\mathcal{Z}}(\mathbf{z}_{\mathbf{I}_j}, \mathbf{z}_{\mathbf{I}_k})$.

We thus seek to learn a space that is equivariant to 3D motion. This is ideal for VS, as we wish for features that have a strong and straightforward relation to pose. Moreover, we seek to explicitly learn a space that is invariant to perturbations $\mathbf{P} \in \mathcal{P}$, such as lighting changes or occlusions (i.e. $\mathbf{z}_{\mathbf{I}+\mathbf{P}} = \mathbf{z}_{\mathbf{I}}$).

B. Learning an SE(3)-equivariant space

To learn the space $\mathcal{Z}$, we propose to use two distinct, parallel neural networks. The first is $\phi : \mathrm{SE}(3) \to \mathcal{Z}$, that maps a pose $\mathbf{r}$ to an embedding $\mathbf{z}_{\mathbf{r}} = \phi(\mathbf{r})$. The second model $\psi : \mathcal{I} \to \mathcal{Z}$, maps an image $\mathbf{I}$ to its latent representation $\mathbf{z}_{\mathbf{I}} = \psi(\mathbf{I})$.

In order to shape $\mathcal{Z}$, we devise our loss function that is based on the distances between latent representations. While most metric learning approaches focus on clustering problems, we require our distances to be continuously meaningful as in [3], [18]. Instead of comparing the embeddings of specific tuples $(\mathbf{r}_j, \mathbf{I}_j, \mathbf{r}_k, \mathbf{I}_k)$, we adopt a full-batch approach, comparing a representation with every other in the batch. To do so, we leverage the distance matrices in SE(3) and in the latent space, viewing an embedding batch as a fully connected graph. By using this dense approach, we encourage the latent representations to position themselves with respect to every neighbor, providing a more stable training signal for the models ϕ and ψ.

To train the pose encoder ϕ, Our first loss seeks to enforce the equivariance with SE(3) when considering pose projections. As SE(3) has no true distance metric, a weighting between translation $\mathbf{t}$ and rotation $\theta\mathbf{u}$ distances must be introduced to create a pseudo-metric $d_{\mathrm{SE}(3)}$, as in [16]. We propose to define the weightings from the data and normalize the translational and rotational distances by their average value in the dataset. To compute the translation/rotation distances between two poses $\mathbf{r}_j, \mathbf{r}_k$, we first compute the pose difference $\Delta\mathbf{r} = (\boldsymbol{\Delta}\mathbf{t}, \boldsymbol{\Delta}\theta\mathbf{u})$ we then define a single dataset-aware metric on SE(3), as

$$d_{\mathrm{SE}(3)}(\mathbf{r}_j, \mathbf{r}_k) = \frac{1}{w_{\mathbf{t}}}\|\boldsymbol{\Delta}\mathbf{t}\|_2 + \frac{1}{w_{\theta\mathbf{u}}}\Delta\theta \tag{8}$$

where $w_{\mathbf{t}}, w_{\theta\mathbf{u}}$ are averages of translational and rotational distances, computed on a subset of the dataset. With $d_{\mathrm{SE}(3)}$ defined, we introduce our first loss. Considering a batch of B samples, we seek to minimize

$$\mathcal{L}_{\phi,\mathrm{SE}(3)} = \frac{1}{B^2}\sum_{j=0}^{B}\sum_{k=0}^{B}(d_{\mathrm{SE}(3)}(\mathbf{r}_j, \mathbf{r}_k) - \|\mathbf{z}_{\mathbf{r}_j} - \mathbf{z}_{\mathbf{r}_k}\|_2)^2 \tag{9}$$

This loss is fairly straightforward to minimize, as ϕ has access to the full pose information, and its main task is to transform the combination of translational and rotational metrics into a single euclidean distance. Of course, since poses are not available during VS, we require ψ to project an image to the same embedding as its associated pose, as well as match the distances with other poses. This is modeled by:

$$\mathcal{L}_{\psi,\phi} = \frac{1}{B^2}\sum_{j=0}^{B}\sum_{k=0}^{B}(\|\mathbf{z}_{\mathbf{I}_j} - \mathbf{z}_{\mathbf{r}_k}\|_2 - \|\mathbf{z}_{\mathbf{r}_j} - \mathbf{z}_{\mathbf{r}_k}\|_2)^2 \tag{10}$$

By transitivity, $\mathcal{L}_{\psi,\phi}$ also minimizes $(d_{\mathrm{SE}(3)(\mathbf{r}_j,\mathbf{r}_k)} - \|\mathbf{z}_{\mathbf{I}_j} - \mathbf{z}_{\mathbf{I}_k}\|_2)^2$. To learn invariance to perturbations, we incorporate perturbed samples in the image batch. These noisy samples are exploited in $\mathcal{L}_{\psi,\phi}$. Moreover, the image representations associated to a single pose $\mathbf{r}_j$ (the original image and its P perturbed versions) are compared, and their distance to each other minimized:

$$\mathcal{L}_{\mathcal{P}_j} = \frac{1}{P^2}\sum_{m=0}^{P}\sum_{n=0}^{P}\|\mathbf{z}_{\mathbf{I}_j+\mathbf{P}_m} - \mathbf{z}_{\mathbf{I}_j+\mathbf{P}_n}\|_2 \tag{11}$$

To obtain our final training objective, we sum up the losses

$$\mathcal{L} = \mathcal{L}_{\phi,\mathrm{SE}(3)} + \mathcal{L}_{\psi,\phi} + \sum_j \mathcal{L}_{\mathcal{P}_j} \tag{12}$$

The impact of each loss is visualized in Figure (2). It can be seen that the objectives designed above act as push-pull forces, moving the embeddings to respect the distance constraints, as well as minimize the influence of perturbations. By comparing a representation with every neighbor, we ensure that a single iteration forces an embedding towards a more stable location.

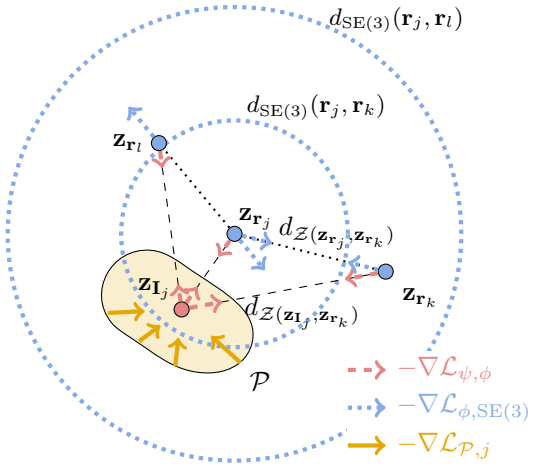


Fig. 2: Shaping the latent space with metric learning. We compare the representations of a sample j with their neighbors. While illustrating only the gradients for j, the loss is applied to every other sample.

C. Training policy

To constitute our batches of data, we randomly sample $N = 64$ poses from the dataset, and add their $M = 3$ closest neighbors found in a subset of the data to ensure that the constraints are met both globally and locally. For each image, we also generate $P = 2$ perturbations. Comparing to AEVS [11], not using a decoder (discarded for VS) network allows for larger batches. The perturbations consist in adding

Gaussian noise, changing the brightness of the image and performing random erasing [32]. The image/pose generation process is the same as the one presented in [11], leveraging simulation to create data cheaply and efficiently.

We set the dimensionality of the latent space so that $\mathcal{Z} = \mathbb{R}^{32}$. The image encoder ψ is a ResNet-34 [12], with the modifications of [11], i.e. replacing batch normalization [14] with weight normalization [23] and the end average pooling with a group convolution. The pose-embedding network ϕ is a 5-layer perceptron of dimensions $6 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 64 \rightarrow 32$, with ReLU activations after each hidden layer. The networks are jointly trained for 50 epochs, on a dataset of 100K samples. Training takes around 8h with a Quadro RTX 6000. Gradient descent is performed with the Adam optimizer [19] and a learning rate of 10^{-4}. With our networks trained, we can finally perform VS in the latent space.

D. Visual servoing in latent space

Our approach to VS is two-pronged, based on the multi-modal nature of $\mathcal{Z}$. We first embed the current and desired images $\mathbf{z}_\mathbf{I} = \psi(\mathbf{I})$, $\mathbf{z}_{\mathbf{I}^*} = \psi(\mathbf{I}^*)$ in $\mathcal{Z}$ to obtain $\mathbf{e} = \mathbf{z}_\mathbf{I} - \mathbf{z}_{\mathbf{I}^*}$. To minimize $\mathbf{e}$, we require an interaction matrix, that gives us the control directions. Before VS starts, we project a set of N poses $\{\mathbf{z}_{\mathbf{r}_1}, ...\mathbf{z}_{\mathbf{r}_N}\} = \{\phi(\mathbf{r}_1), ..., \phi(\mathbf{r}_N)\}$ in the latent space, along with their interaction matrix, thanks to the method described in Section II-C. The latent interaction matrix is then defined as $\mathbf{L}_{\mathbf{z}_{\mathbf{r}_j}} = \frac{\partial \mathbf{z}_{\mathbf{r}_j}}{\partial \mathbf{r}_j} \mathbf{L}_{\mathbf{r}_j}$, with $\frac{\partial \mathbf{z}_{\mathbf{r}_j}}{\partial \mathbf{r}_j}$ the Jacobian of ϕ with respect to $\mathbf{r}_j$ (computed via forward propagation) and $\mathbf{L}_{\mathbf{r}_j}$ given by Equation (5). Because we minimize $\mathcal{L}_{\psi,\phi}$ (Equation (10)), We can approximate the interaction matrix at $\mathbf{z}_\mathbf{I}$ as an interpolation of its neighbors' interaction matrices. We thus have a K-Nearest Neighbors (KNN) regression problem, defined as:

$$\mathbf{L}_{\mathbf{z}_\mathbf{r}} = \sum_{j=0}^{K} \alpha_j \mathbf{L}_{\mathbf{z}_{\mathbf{r}_j}} \text{with } \alpha_j = \frac{\|\mathbf{z}_{\mathbf{r}_j} - \mathbf{z}_\mathbf{I}\|_2}{\sum_{k=0}^{K} \|\mathbf{z}_{\mathbf{r}_k} - \mathbf{z}_\mathbf{I}\|_2} \tag{13}$$

To obtain the pose embeddings set, we generate poses on a 6D grid (displacements in translation and rotation), and oversample near $\mathbf{z}_{\mathbf{I}^*}$. In our experiments, we thus create a set of 1M pose representations that can be used for KNN. To be computationally tractable, we store them in a KD-tree [5]. Other, smarter, less memory demanding sampling strategies, based on the values of both $\mathbf{z}_\mathbf{I}$ and $\mathbf{z}_{\mathbf{I}^*}$, are possible. We however found that this straightforward approach works well in practice. Unlike AEVS, the interaction matrices of the pose representations (and thus the network Jacobian) are computed offline. Fitting the pieces together, the final control law is simply

$$\mathbf{v} = -\lambda \mathbf{L}^+_{\mathbf{z}_\mathbf{r}} (\mathbf{z}_\mathbf{I} - \mathbf{z}_{\mathbf{I}^*}) \tag{14}$$

In the next section, we explore the behavior of this VS scheme, both in simulation and on a 6DOF robot.

IV. Experiments and results

A. Simulation validation

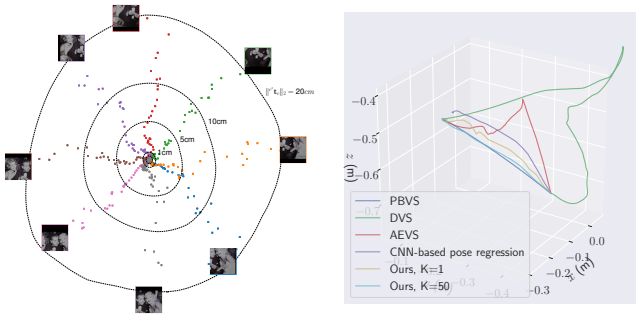
We start our experimental validation with a large scale experiment: we run 500 VS tasks with multiple methods. The initial positions are chosen with a "look-at/look-from" scenario: we sample camera positions in a volume of dimensions $1.2m \times 1.2m \times 0.3m$, centered on the desired position. we then sample the focal (look-at) points of the cameras on the planar scene, with a distance to the desired focal points between $8cm$ and $32cm$. The scene is a poster of dimensions $80cm \times 60cm$, and we set the desired camera elevation to $60cm$. From the camera position and focal point, we build the camera orientation, to which we add a rotation around the optical axis $\in [-120°, 120°]$. The average initial pose error is $47cm \pm 16cm, 74° \pm 28°$.

We experiment with multiple methods, the first one being a pure photometric scheme (DVS) [9]. We also compare with AEVS [11], as well as a PBVS visual servoing approach which uses a CNN-based pose regression approach, as developed in [24], [4] (referred as PBVS-CNN). To ease training and improve results, we adopt the automatic weighting loss of [16] that balances translation and rotation errors in Equation (4). For both AEVS and PBVS-CNN, we use the network and data described in Section III-C. For our method, we explore different numbers of neighbors K. We first report the percentage of samples that convergence for each method. A sample is defined as having converged if the VS method significantly reduces (by at least 90%) the initial positioning error and the final velocities are close to 0. For those samples, we measure the end positioning error, as well as the Absolute Pose Error (APE), averaged over all iterations of a trajectory, which describes how far the method strays from the geodesic of PBVS. The other included metrics are the mean length ratios: the length of trajectory of the observed method divided by the length of the PBVS trajectory. As can be seen in Table I, our method is able to converge reliably and accurately, with a positioning error that is comparable to photometric methods in the case of clean target images $\mathbf{I}^*$ (noted ✓in the table), while having a far larger domain of convergence. We can observe that a larger value for K leads to a more stable and accurate positioning. Looking at the results of the PBVS-CNN, it can be seen that the end positioning error is subpar ($3.3cm, 1.71°$ on average). The trajectory statistics (APE and length ratio) show that our method is far closer to the behavior of PBVS, compared to AEVS or DVS. This is made explicit in Figure (3b). The overall statistics when considering cases where $\mathbf{I}^*$ is perturbed (noted ✗) highlight that our method better handles variations in lighting and occlusions. While both convergence and accuracy degrade, they remain above that of the pose estimator, even on clean images.

Next, we explore the servoing behavior in the latent space. We perform 8 trajectories, where the initial errors are displacements on the x, y axes, with an initial error of $20cm$. We then visualize the trajectories in the latent space by projecting in a 2D-subspace with PCA (explained variance $\approx 96\%$). As can be seen in Figure (3a), the minimization of $\mathbf{e}$ in the latent space leads to nearly straight lines in the latent space. The error between pose embeddings also correlates well with the error from image representations.

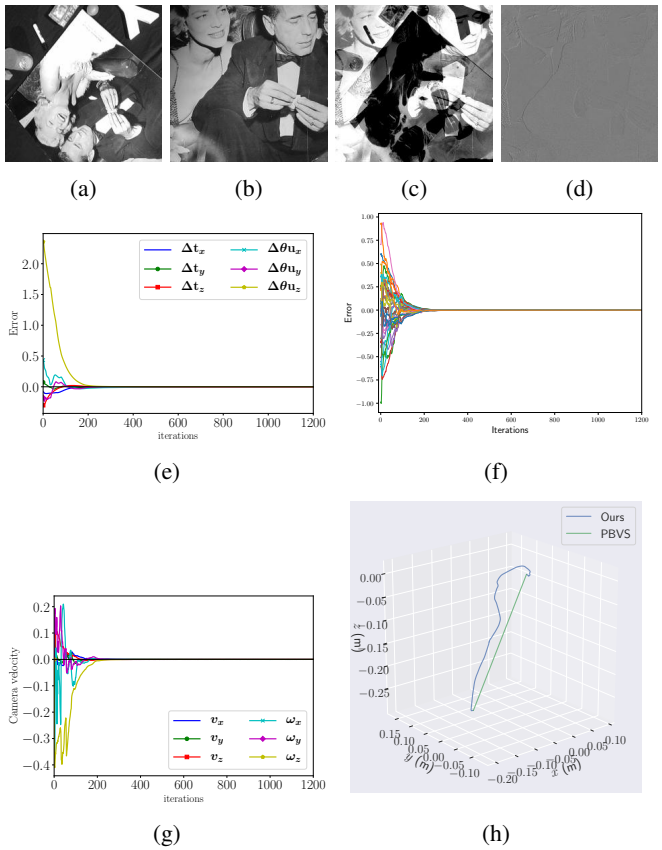
	Clean $\mathbf{I}^*$	Converged ↑	End error ↓		APE ↓		Length
		%	mm	∘	cm	∘	ratio ↓
DVS [9]	✓	9.8	0.0±0.0	0.0±0.0	17.32±12.48	31.0±14.35	2.6±4.9
AEVS [11]	✓	33.6	0.01±0.0	0.0±0.0	2.74±6.36	2.66±6.4	2.75±4.85
PBVS-CNN, e.g. [4]	✓	75.6	33.52±6.45	1.71±0.65	3.13±1.04	1.85±0.96	1.11±0.08
	✗	36.8	32.21±15.71	2.37±1.57	4.0±0.74	2.55±0.64	1.12±0.1
Ours, $K = 1$	✓	99.2	3.02±1.21	0.21±0.12	0.71±0.88	1.59±3.28	1.46±0.97
	✗	73.8	19.42±12.77	1.96±1.28	3.01±0.71	3.68±2.33	1.33±0.32
Ours, $K = 50$	✓	100.0	0.04±0.03	0.0±0.0	0.4±0.72	1.08±2.5	1.18±0.23
	✗	76.0	19.29±12.81	1.92±1.28	3.31±0.61	3.72±1.6	1.14±0.11

TABLE I: Comparative results of different VS methods on 500 trajectories. Error and trajectory statistics are reported for cases that converge. For metrics marked with ↑, higher is better. Lower is better when marked ↓.



(a) (b)

Fig. 3: (a) PCA projection of trajectories $\mathbf{z}_\mathbf{I} - \mathbf{z}_{\mathbf{I}^*}$ in the latent space, for 2D motions. Circles show the error for the pose embeddings $\mathbf{z}_\mathbf{r} - \mathbf{z}_{\mathbf{r}^*}$ for various distances. (b) 3D trajectories of different methods on a single example.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 4: First experiment: (a) Starting image $\mathbf{I}$. (b) Desired image $\mathbf{I}^*$. (c) Starting image difference $\mathbf{I} - \mathbf{I}^*$. (d) Final image difference. (e) Pose difference $\mathbf{r} - \mathbf{r}^*$. (f) Error in the latent space $\mathbf{z}_\mathbf{I} - \mathbf{z}_{\mathbf{I}^*}$. (g) Camera velocities $\mathbf{v}$. (h) 3D trajectory.

B. Robot experiment

We also deploy our method on a 6DoF gantry robot and study its behavior on a large motion. For the experiment, shown in Figure (4), the initial displacement is $\Delta\mathbf{r}_0 = (-8.16cm, 7.27cm, -29.98cm, 24.86°, -10.16°, 135.06°)$, with a large error in the image (Figure (4c)) and run our method for 1.2k iterations. Note that this motion (and thus the image) is far from what is seen during training. As servoing progresses, the error in the latent space (Figure (4f)) is quickly minimized. The final positioning error is $\Delta\mathbf{r}_{final} = (0.09cm, 0.08cm, -0.04cm, 0.08°, -0.12°, -0.01°)$, and the resulting error in image space is low (Figure (4d)). As shown in Figures (4e, 4g, 4h), the trajectory starts with some unwanted motion on the x, y axes (compensated by y/x rotations), probably due to the fact that $\mathbf{I}$ lies outside the training domain. However, our method recovers and exhibits a smooth decrease in positioning error.

V. Conclusion

In this paper, we proposed a method that allows visual servoing from both image and pose representations in a common latent space. This new visual servoing scheme combines the accuracy of photometric methods with the behavior of pose-based approaches. Our experiments show strong results, with a large convergence domain and accurate positioning. In future works, we plan to extend our method to deal with more than two modalities (i.e. adding depth or segmentation information), so that the visual features at the current and desired poses may be drawn from different domains. In addition, we believe that it is possible to reduce data requirements by including some form of self-supervised learning. For instance, small motions could be used to warp an image and generate new weakly labeled samples. Supervision would then be used to learn representations tied to large motions, while smaller displacements would be taken into account with self-supervision.

REFERENCES

[1] N. Andreff, B. Espiau, and R. Horaud. Visual servoing from lines. *Int. Journal of Robotics Research*, 21(8):669–699, August 2002.

[2] M. Bakthavatchalam, O. Tahri, and F. Chaumette. A Direct Dense Visual Servoing Approach using Photometric Moments. *IEEE Trans. on Robotics*, 34(5):1226–1239, October 2018.

[3] V. Balntas, S. Li, and V. Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 751–767, 2018.

[4] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke. Training deep neural networks for visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'18*, pages 3307–3314, Brisbane, Australia, May 2018.

[5] JL. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[6] F. Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Trans. on Robotics*, 20(4):713–723, August 2004.

[7] F. Chaumette and S. Hutchinson. Visual servo control, Part I: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[8] F. Chaumette and E. Malis. 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings. In *IEEE Int. Conf. on Robotics and Automation*, volume 1, pages 630–635, San Francisco, CA, April 2000.

[9] C. Collewet, E. Marchand, and F. Chaumette. Visual servoing set free from image processing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'08*, pages 81–86, Pasadena, CA, May 2008.

[10] X. Dong and J. Shen. Triplet loss in siamese network for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 459–474, 2018.

[11] S. Felton, P. Brault, E. Fromont, and E. Marchand. Visual Servoing in Autoencoder Latent Space. *IEEE Robotics and Automation Letters*, 7(2):3234–3241, April 2022.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[13] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015.

[14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML'15*, page 448–456, 2015.

[15] D. Jayaraman and K. Grauman. Learning image representations tied to ego-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1413–1421, 2015.

[16] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564, 2017.

[17] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. *IEEE International Conference on Computer Vision, ICCV*, pages 2938–2946, 2015.

[18] S. Kim, M. Seo, I. Laptev, M. Cho, and S. Kwak. Deep metric learning beyond binary supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2288–2297, 2019.

[19] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *2015 Int. Conf. on Learning Representations (ICLR)*, 2015.

[20] E. Marchand. Control camera and light source positions using image gradient information. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 417–422, Roma, Italia, April 2007.

[21] E. Marchand. Subspace-based visual servoing. *IEEE Robotics and Automation Letters*, 4(3):2699–2706, July 2019.

[22] E. Marchand. Direct visual servoing in the frequency domain. *IEEE Robotics and Automation Letters*, 5(2):620–627, 2020.

[23] T. Salimans and D. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Neural Information Processing Systems 2016*, pages 901–909, 2016.

[24] A. Saxena, H. Pandya, G. Kumar, A. Gaud, and K.M. Krishna. Exploring convolutional networks for end-to-end visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'17*, pages 3817–3823, May 2017.

[25] J. Sung, I. Lenz, and A. Saxena. Deep multimodal embedding: Manipulating novel objects with point-clouds, language and trajectories. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2794–2801. IEEE, 2017.

[26] V. Wan, Y. Agiomyrgiannakis, H. Silen, and J. Vit. Google's next-generation real-time unit-selection synthesizer using sequence-to-sequence lstm-based autoencoders. In *INTERSPEECH*, pages 1143–1147, 2017.

[27] L. Wang, Y. Li, and S. Lazebnik. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5005–5013, 2016.

[28] L.E. Weiss. Dynamic visual servo control of robots. an adaptive image based approach. Technical Report CMU-RI-TR-84-16, Carnegie-Mellon University, April 1984.

[29] W. Wilson, C. Hulls, and G. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Trans. on Robotics and Automation*, 12(5):684–696, October 1996.

[30] M. Won, S. Oramas, O. Nieto, F. Gouyon, and X. Serra. Multimodal metric learning for tag-based music retrieval. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 591–595. IEEE, 2021.

[31] C. Yu, Z. Cai, H. Pham, and Q. Pham. Siamese convolutional neural network for sub-millimeter-accurate camera pose estimation and visual servoing. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 935–941, 11 2019.

[32] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

[33] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.