Subsequent Keyframe Generation for Visual Servoing

Nathan Crombez, Jocelyn Buisson, Zhi Yan, Yassine Ruichek

Subsequent Keyframe Generation for Visual Servoing

Nathan Crombez, Jocelyn Buisson, Zhi Yan, Yassine Ruichek

Abstract— **In this paper, we study the problem of autonomous and reliable positioning of a camera w.r.t. an object when only this latter is known but not the rest of the scene. We propose to combine the advantages and efficiency of a visual servoing scheme and the generalization ability of a generative adversarial network. The paper describes how to efficiently create a synthetic dataset in order to train a network that predicts an intermediate visual keyframe between two images. Subsequent predictions are used as visual features to autonomously converge towards the desired pose even for large displacements. We show that the proposed method can be used without any prior knowledge on the scene appearance except for the object itself, while being robust to various lighting conditions and specular surfaces. We provide experimental results, both in simulation and using a real service robot platform to validate and evaluate the effectiveness, robustness, and accuracy of our approach.**

I. Introduction

Due to demographic and social issues, mobile manipulators capable of autonomously perform everyday home duty such as tidying rooms up, interacting with furniture or fetching and bringing of daily supplies are raising great interest. These challenging robot abilities are regularly evaluated throughout international competitions such as the "World Robot Summit"[1] or the "RoboCup@Home"[2]. In order to autonomously perform their tasks, an accurate positioning of their gripper w.r.t. a scene is a fundamental capacity.

Visual Servoing (VS) is an elegant and well-studied scheme that can be used to perform such precise positioning [3]. VS refers to closed-loop control methods of dynamic systems, such as robots, using data from vision sensors as feedback. The visual information is usually obtained by a camera that is set into motion by the system (eye-in-hand configuration) or by an external camera that captures the system displacements (eye-to-hand configuration). The classical VS scheme aims to regulate to zero an error between visual features extracted from a desired image and from images acquired by the robot. A control law is designed w.r.t. the chosen visual features to control the degrees of freedom of the camera to reach the desired pose. No matter the visual characteristics used, a set of features representing the desired camera pose is always required. Knowing in advance the configuration of the desired image features is often tricky or even impossible in many real use cases. In this paper, we address the following question: how to position a camera w.r.t. a known object that may lie in different places? Using VS terminology: how to control the camera displacement when only the projection of a single object is known in the desired image while the rest of the scene (e.g., background, other objects, etc.) remains unpredictable?

In this paper, we propose to exploit the image generation abilities of Generative Adversarial Network (GAN) to predict a subsequent desired keyframe in order to iteratively drive the camera towards a targeted pose. Even without any prior information regarding the scene that surrounds the object at the desired pose, our method is able to generate a consistent image in between the current acquired image and the partially unknown desired one. Our proposed method is flexible as it decouples the image prediction from the camera control. Therefore, any type of feature or control method in the literature can be used jointly with our keyframe prediction network. More precisely, the following contributions are described in this paper:

- a novel control pipeline called GANVS that combines both the advantages and efficiency of the VS framework and the image translation ability of a GAN,
- an automated synthetic dataset generation strategy that ensures the generalization of the proposed method to any kinds of backgrounds, surrounding scenes and diverse lighting conditions,
- a quantitative comparison with state-of-the-art methods,
- an accurate positioning performed on a mobile manipulator robot in various real environments.

The remainder of this paper is organized as follows. First, Section II discusses related work. Then, Section III introduces the general framework of the proposed method and explains the control scheme. After that, Section IV describes the GAN architecture and the process to create the dataset. Experimental results including comparative, qualitative and quantitative evaluations are presented in Section V. Finally, conclusion and future work based on preliminary results are explained in Section VI.

II. Related work

On the one hand, VS can be performed using geometric elements based approaches that exploit features such as image points or straight lines [4]. However, these approaches involve image processing techniques for the robust extraction of the features, their matching, and their real-time spatiotemporal tracking. The accuracy of these complex steps directly affects the performances of the geometric features based methods. In our study case, since no information is available in the desired image except the target object, which

Nathan Crombez, Jocelyn Buisson, Zhi Yan and Yassine Ruichek are with CIAD laboratory, Université Bourgogne Franche-Comté, UTBM, 90010 Belfort, France, e-mail: {firstname}.{name}@utbm.fr

This work has been supported by the MACPOLO research project. The MACPOLO project is a Toyota Partner Robot joint research project between Toyota Tsusho Corporation and University of Technology of Belfort-Montbéliard (UTBM).

may appear under different lighting conditions, thus building beforehand a set of robust and reliable desired geometric descriptors is a particularly non-trivial task. An approach has been recently proposed to autonomously predict a set of desired image features that represent the grasping pose of a targeted object [5]. This method has shown robustness to dynamic scenes but requires depth information and relies on image points detection and matching.

On the other hand, visual features that are directly based on the complete photometric information contained in the images such as pixel luminances [6], photometric moments [7] or Gaussian mixtures [8] do not require the image processing stages mentioned above. Moreover, it has been demonstrated that VS based on photometric information has several other advantages including very accurate convergence and robustness to unknown depth and partial occlusions. Interesting results have also been obtained using model-based VS even without any information regarding the rest of the scene [9]. However, since the considered visual features are the distances between 2D points in the image captured by the robot and the projection of the 3D lines that compose the object CAD model, the object pose in the initial image has to be computed or image processing has to be used to detect the edges of the object. Moreover, minimizing the point-line distance imposes a relatively small displacement between the initial and the desired camera poses. This approach is therefore more suitable and relevant for object tracking [10].

Exploiting Convolutional Neural Network (CNN) for VS has been recently investigated. Indeed, many different CNN architectures have been trained to learn the relative camera pose between two given images in order to build a VS task in an end-to-end paradigm. For example, the authors of [11] have trained a FlowNet network [12] on publicly available datasets to learn the camera ego-motion between two images in order to control the 6 degrees of freedom (dof) of a quadrotor. AlexNet [13] and VGG [14] networks have also been trained to estimate the relative camera pose between two input images, that is then used to build a 6 dof control law [15]. A similar approach has been proposed in order to solve 4 dof industrial grasp tasks using GoogleNet to output a motion command [16]. In contrast, authors of [17] have proposed to use an optical flow network [18] and a single-view depth network [19] to respectively fill the features error vector and the interaction matrix of a classical point-based VS framework [3].

Differently, Deep Reinforcement Learning (DRL) is also utilized to learn a policy of actions in the environment (i.e., robot/camera displacement) by maximizing visual rewards in order to build VS neural controllers. It has been shown in [20] that a combination of deep features selected using a Q-iteration algorithm and DRL can be used to control 4 dof in a synthetic car visual following context. Authors of [21] have proposed a DRL controller that was trained to learn a pertinent VS policy for a direct mapping between errors in the image and the linear velocity commands of a quadrotor. Since DRL approaches require self-exploration of the potential actions in the environment, the training stage

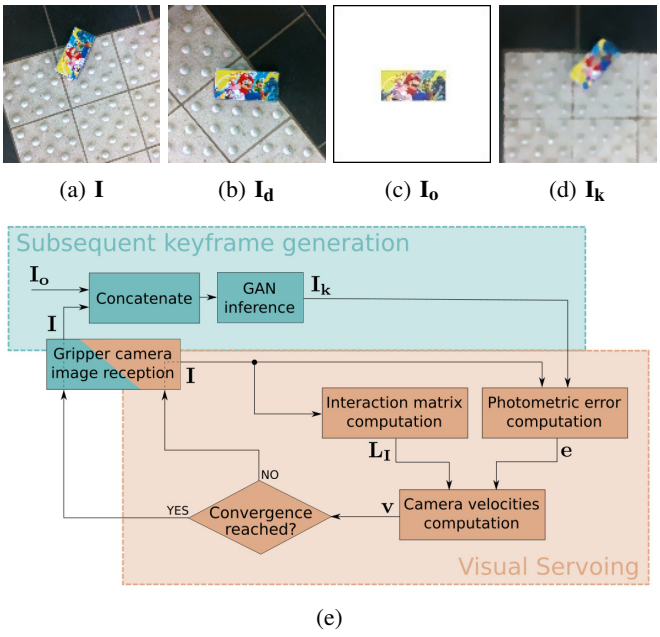


Fig. 1: Data and method: (a) Current captured image $\mathbf{I}$, (b) Unknown desired image $\mathbf{I_d}$, (c) Desired object-only image $\mathbf{I_o}$ that visually represents the targeted camera pose w.r.t. the object, (d) Generated subsequent keyframe $\mathbf{I_k}$, (e) Overview of the proposed method pipeline.

is almost impractical on a real robot and is then generally performed in simulation, leading inevitably to reality gap issues. To reduce this gap, it has been proposed to utilize a GAN (i.e., CycleGAN [22]) to translate the real images acquired during an experiment to the virtual domain of the synthetic training dataset [23]. All the aforementioned methods have shown interesting capabilities but fully known desired visual features are always required by the process.

III. Control scheme

We consider the positioning of a camera w.r.t. an object as an optimization problem that aims to regulate to zero the error:

$$\mathbf{e}(\mathbf{r}) = \mathbf{s}(\mathbf{r}) - \mathbf{s_d} \tag{1}$$

where $\mathbf{r}$ represents the current camera pose, and where $\mathbf{s}(\mathbf{r})$ and $\mathbf{s_d}$ are vectors of visual features extracted respectively from the current acquired image $\mathbf{I}$ (e.g., Fig.1a) and a desired image $\mathbf{I_d}$ (e.g., Fig.1b). As the targeted object may be located in different places, the surrounding background and potential other objects present in the camera field-of-view cannot be determined in advance. One can note that in other VS approaches, $\mathbf{I_d}$ (or a desired set of features) is assumed to be fully known. Actually, most of the time in real applications, if we replace every unknown pixel in $\mathbf{I_d}$ (Fig.1b) by white pixels, we obtain what we call the *desired object-only image* $\mathbf{I_o}$ (Fig.1c). This latter visually represents the targeted camera pose w.r.t. the object.

We propose an approach consisting of two independent stages depicted in Fig. 1e. The first stage named *Subsequent*

keyframe generation, predicts a visual image $\mathbf{I_k}$ (Fig.1d) of the scene when the camera is slightly moved to an intermediary pose $\mathbf{r_k}$ in between $\mathbf{r}$ and the desired pose $\mathbf{r_d}$. This predicted keyframe is obtained as follows:

$$\mathbf{I_k} = Gen(\mathbf{I_o}, \mathbf{I}) \tag{2}$$

where $Gen(\cdot)$ is the generator of our dedicated GAN that takes as input the object-only desired image $\mathbf{I_o}$ and the current image $\mathbf{I}$ (see Section IV).

In the second stage, the generated keyframe $\mathbf{I_k}$ is used as a temporary desired image to drive the camera towards the targeted pose. Since $\mathbf{I_k}$ is a visual prediction generated by the GAN, the resulting image (Fig. 1d) is often blurred and may contain artifacts [24]. Consequently, reliable geometric visual features such as points may be difficult to detect, match and track. Thus, we chose to use a photometric feature, i.e., the luminance of every image pixel of $\mathbf{I}$ and $\mathbf{I_k}$. Then, (1) becomes:

$$\mathbf{e}(\mathbf{r}) = \bar{\mathbf{I}}(\mathbf{r}) - \bar{\mathbf{I}}_\mathbf{k} \tag{3}$$

Note that the overbar denotes the vectorization of the image content.

The VS task is achieved by iteratively applying linear and angular velocities to the camera in order to regulate to zero the error expressed by (3). A classical Gauss-Newton control law is used to compute these velocities $\mathbf{v} = (\nu, \omega)$:

$$\mathbf{v} = -\lambda {\mathbf{L_I}}^{\top} \mathbf{e}(\mathbf{r}) \tag{4}$$

where λ is a positive scalar and $\mathbf{L_I}$ is the interaction matrix related to pixel luminance of the current image $\mathbf{I}(\mathbf{r})$, i.e., the matrix that associates the image features displacement with the motion of the camera pose $\mathbf{r}$ [6]. At convergence, the camera pose $\mathbf{r}$ tends to $\mathbf{r_k}$, i.e., the pose visually represented by the predicted keyframe $\mathbf{I_k}$. We consider that the convergence is reached when the residual photometric error, i.e., the Zero-mean Normalized Sum of Squared Differences (ZNSSD) between $\mathbf{I_k}$ and $\mathbf{I}$, becomes stable. Once the camera has reached the pose represented by the keyframe, a new one is subsequently generated and the process is repeated until the camera finally arrives at the desired pose represented by the object-only desired image $\mathbf{I_o}$.

IV. Subsequent Keyframe Generation

A. Training dataset generation

We developed a synthetic dataset generation strategy to ensure that our GAN generalizes to many kinds of backgrounds and lighting conditions, even in real scenes. We created a virtual environment consisting in a room with a textured and highly specular floor that does not contain strong patterns or visual repetitions. Each time a triplet of images $\{\mathbf{I}, \mathbf{I_o}, {\mathbf{I_k}}^*\}$ is rendered for the training set, the 3D model of the object is moved to a different place in the room, while the floor texture is changed and the intensity of each of the four ceiling lights used to lit the room is randomized (see Fig.2). We denote a keyframe in the dataset ${\mathbf{I_k}}^*$ as opposed to $\mathbf{I_k}$ in order to distinguish the one rendered for the purpose of training from the one resulting from the generator inference (2). We used two floor textures, a dark one and a light version derived by color inversion. The light intensity is randomly chosen in a range going from half to twice the reference light intensity for the room, which matches the lighting conditions of a normal well-lit room.

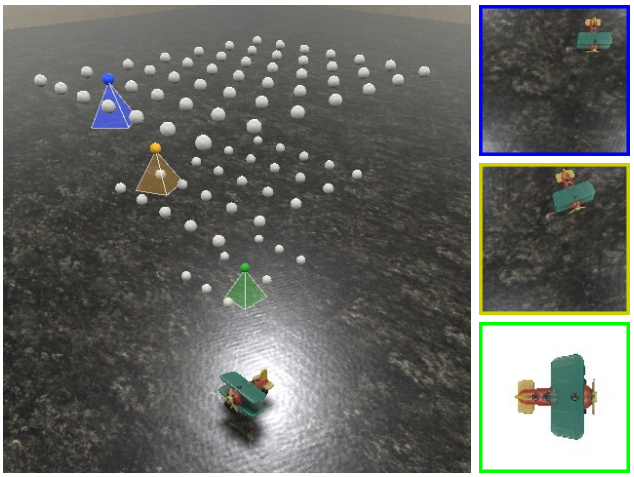


Fig. 2: Discretization of training example poses of the camera. The yellow sphere corresponds to the pose for ${\mathbf{I_k}}^*$ associated with the blue pose for $\mathbf{I}$ and green pose for $\mathbf{I_o}$. Multiple orientations are considered for each position in the pyramid.

In this work, we restrict the camera motion to 4 dof (i.e., the translations and the rotation around the optical axis) to simplify the creation of the dataset and the training of the GAN. Therefore, the set of all poses in which the object is visible can be approximated by an inverted cone. The current and the desired camera poses $\mathbf{r}$ and $\mathbf{r_d}$ could be anywhere within this volume. This latter is discretized into layered grids, with at least two levels corresponding to the minimum and maximum heights of the camera w.r.t. the object (see Fig. 2). As the dimensions of the object in image-space varies according to the camera height, we respectively set the grid dimensions at each level with a different number of fixed positions. For each position in the pyramid, a fixed number of rotations N_θ around the optical axis of the camera is considered. The discretization parameters must ensure that the photometric overlap as a percentage of the object area in the images rendered from two adjacent poses is near equal at each level.

Similar to [15], we also generate N_c additional poses $\mathbf{r}$ around each pose $\mathbf{r_d}$ in the pyramid to improve the accuracy of the final convergence. More precisely, the poses $\mathbf{r}$ are obtained using a Gaussian draw with standard deviations $\sigma_x = \sigma_y = \frac{g_s}{12}$, $\sigma_z = \frac{g_z}{12}$ and $\sigma_\theta = \frac{360^\circ}{12 \cdot N_\theta}$, where g_s is the distance between two consecutive poses in the grid at the level where $\mathbf{r_d}$ is located, and g_z is the mean distance between the current grid level and the two neighboring ones.

Each pose in the pyramid can be either an initial pose $\mathbf{r}$ or a desired one $\mathbf{r_d}$. Given a pair $\{\mathbf{r}, \mathbf{r_d}\}$, an intermediary pose $\mathbf{r_k}$ is calculated and the respective images triplet $\{\mathbf{I}, \mathbf{I_o}, {\mathbf{I_k}}^*\}$ is rendered. A straightforward approach to calculate $\mathbf{r_k}$ is to

use a linear interpolation between the two aforementioned poses. This approach, however, often leads the object to leave the field-of-view of the camera. We chose instead to calculate the pose $\mathbf{r_k}$ so that the trajectory of the object in image-space follows a straight line. Also, we ensure that the translation of the object centroid in $\mathbf{I}$ and $\mathbf{I_k}$ is not greater than a threshold δ_{uv} and the rotation of the object is not greater than δ_θ. These two thresholds must be adjusted according to the camera parameters and the dimensions of the object to ensure that the transformation between the two aforementioned images is consistent across all examples in the training set, and more importantly, to keep sufficient overlap between the two images allowing the VS to converge.

In order to render $\mathbf{I_o}$, we replace the floor texture with a pure white background and randomize the intensity of each of the four lights to vary the lighting conditions in $\mathbf{I}$ and $\mathbf{I_k}$ from the ones in $\mathbf{I_o}$. We iterate over the whole pyramid twice in order to create the complete training set, this ensures that the set contains examples of $\mathbf{I}$ with different backgrounds and lighting conditions but from the same camera pose $\mathbf{r}$.

B. GAN Architecture

The GAN architecture that we used is composed of a PixelGAN discriminator and a ResNet generator with 9 consecutive residual blocks, both of them are implemented in the official PyTorch Pix2Pix/CycleGAN framework[1]. Although the default UNet generator generally produces sharper and more detailed images than the ResNet one, our tests have showed that the latter can generalize better in terms of predicting the pose transformation between $\mathbf{I}$ and $\mathbf{I_o}$. We used Wasserstein GAN loss with gradient penalty [25] as it allows the model to learn on our very large and complex dataset for which the default loss led either to exploding or vanishing gradient. We therefore replaced every batch normalization layer in the generator and discriminator by instance ones, as the gradient penalty works with normalization schemes that do not introduce correlations between examples. Apart from that, the network is kept unchanged and all the other parameters are left to their default values.

V. Experimental results

We conducted experiments to validate, compare and evaluate our approach, both in simulation (Section V-A) and with a real robot (Section V-C). We used different target objects including a plane toy, a textured box and a giraffe toy. We generated a fully synthetic dataset for each of them, only replacing the 3D model of the object in the scene before launching the automated process that generates the images (Section IV-A). Indeed, since our objects have similar sizes, each synthetic training dataset has been generated using the same set of parameters. More precisely, the pose volume has been discretized into three levels: the highest level corresponds to the limit of the robot end-effector used in the real experiments, the lowest one corresponds to the closest distance between the camera and the object where the latter is fully in the field-of-view of the former and the third level corresponds to the median height. The grid dimensions of each level are respectively 7×7, 5×5 and 3×3, and we set the number of rotations N_θ to 10. In order to ensure a sufficient photometric overlap between $\mathbf{I}$ and $\mathbf{I_k}$, we chose a translation threshold $\delta_{uv}=16$ and a rotation threshold $\delta_\theta=20^\circ$. Regarding the final convergence, we set the number of poses N_c to 50. Considering this parameterization, each dataset contains around $600k$ triplets of images $\{\mathbf{I},\mathbf{I_o},\mathbf{I_k}^*\}$ and can be automatically generated in less than 3 hours on a computer with an Intel Core i7-8700K at 3.70GHz CPU and two NVIDIA GTX 1080 Ti GPUs. Each model has been trained using the Adam optimizer with a constant learning rate $lr=2.0\times10^{-4}$ and mini batches of 16 samples over 10 epochs. Training for one object took 27 hours on the same computer.

A. Validation

We performed many experiments to validate the effectiveness of our approach in a simulated environment consisting in a small room that contains one or several 3D textured known or unknown objects on the ground. The ground appearance was randomly selected from a set of 50 different textures including grass, concrete, wood, floor tiles, etc. Some of them contain strong patterns or visual repetitions, while some are more uniform.

Fig. 3 illustrates one of these experiments. It shows the initial image $\mathbf{I}$ with the object-only desired image $\mathbf{I_o}$ overlaid on its top left for ease of visualization (Fig. 3a). To converge to the desired pose, 17 subsequent keyframes $\mathbf{I_k}$ generated by the GAN have been used as photometric feature predictions. Some of them are displayed on the top row (Fig. 3b-3e). We insist on the fact that these images have been generated by our trained GANVS and are thus not rendered in our virtual scene. Even though the GAN was never trained with this kind of background, we can see that it actually recreated the pattern quite well. The final difference images $(\mathbf{I}(\mathbf{r})-\mathbf{I_k})$ at the end of each corresponding photometric VS are also shown on the bottom row (Fig. 3b-3e). The final image $\mathbf{I}$ when the desired pose has been reached can be seen in Fig. 3f, where the object-only desired image $\mathbf{I_o}$ is overlaid on the top left to visually evaluate the final convergence. Fig. 3g shows the residual photometric error, i.e the ZNSSD between $\mathbf{I}$ and $\mathbf{I_k}$ throughout the experiment. As our method can be seen as a series of individual photometric VS, each peak corresponds to the first iteration of a new $\mathbf{I_k}$, then the error decreases until reaching the convergence and so on. The computed linear and angular velocities of the camera are respectively shown in Fig. 3h and Fig. 3i. The camera trajectory in scene-space, as shown in Fig. 3j, does not follow a straight line since our GAN is trained to predict subsequential keyframes that keep the object in the camera field-of-view, i.e., the object follows a straight line in image-space (Section IV-A). The initial displacement between the initial and the desired poses was $[8.9cm, 21.0cm, 19.2cm, -96.3^\circ]$. Despite the large initial displacement, the high difference between the initial and the object-only desired images, in addition to important

[1] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

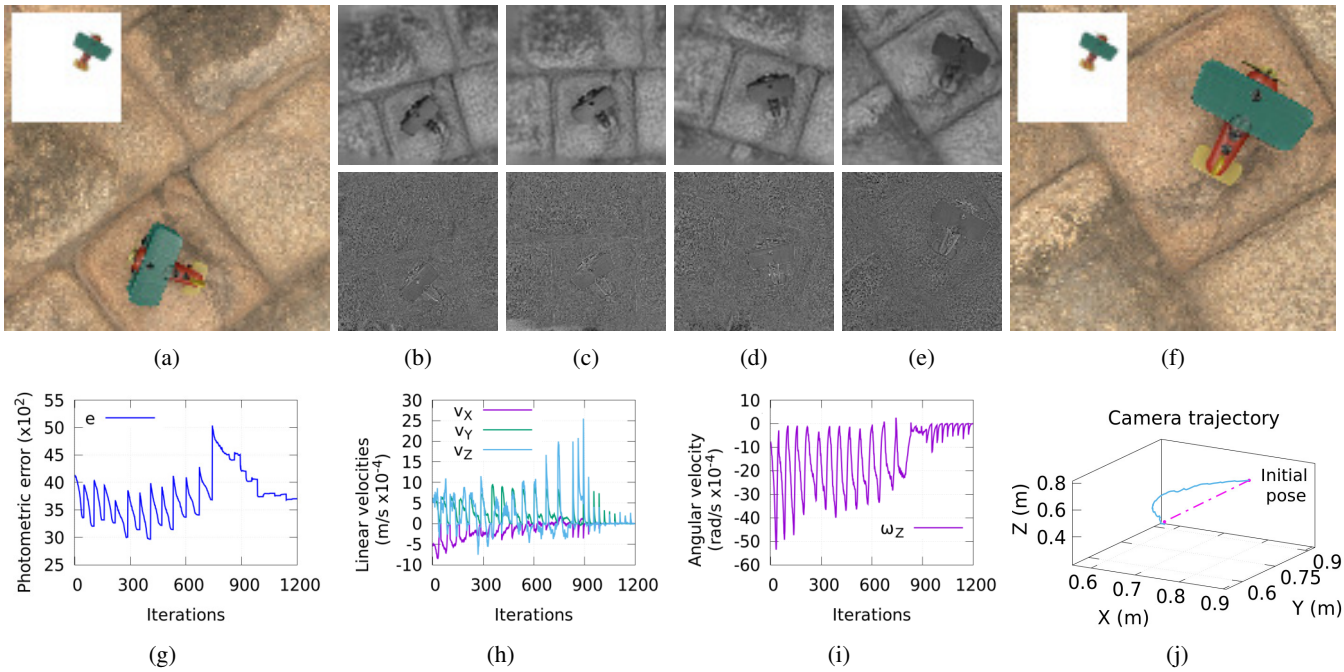


Fig. 3: Validation in simulation: (a) the initial $\mathbf{I}$ with $\mathbf{I_o}$ overlaid on its top left, (b-e) 4 among the 17 keyframes $\mathbf{I_k}$ generated by the GAN (top row) and their corresponding image differences after the camera convergence (bottom row), and (f) the final $\mathbf{I}$, (g) the residual photometric error (ZNSSD), (h) the computed linear camera velocities, (i) the computed angular camera velocity and (j) the camera trajectory.

part in the latter that is totally unknown (Fig. 3a), the camera converges to the desired pose with a low final pose error equal to $[-5.3mm, -1.0mm, -0.9mm, 1.6^\circ]$.

B. Comparisons with state-of-the-art methods

In this section, we compare our proposed method (GANVS) with the luminance based Direct VS (DVS) [6] and a deep neural network based VS (CNNVS) [15]. The CNNVS approach has been implemented and validated according to the original paper. For a fair comparison, both GANVS and CNNVS have been trained on the same dataset generated for one of our object (the plane toy). Especially, as recommended by the authors of [15], the CNNVS network (VGG) was pre-trained on ImageNet [26] then fine-tuned with our dataset.

The three approaches were compared on tasks of positioning w.r.t. one object over 10 batches of 50 individual runs. For each run, the object-only desired image $\mathbf{I_o}$ is rendered from a random camera pose, then the camera is randomly displaced to an initial pose and the texture of the ground is changed. The difficulty (i.e., the initial relative displacement between $\mathbf{r}$ and $\mathbf{r_d}$) is incremented over each batch. More precisely, displacements are drawn from Gaussian distributions on the 4 dof with standard deviations increasing linearly for each batch: from $1cm$ to $22cm$ for the translations and 1° to 60° for the rotation, respectively for batches 1 to 10.

Fig. 4 provides the results of this benchmark, i.e., the convergences rate of the three compared methods. It can be seen that the proposed approach has a larger convergence domain and a higher proportion of runs converging to the desired pose than the state-of-the-art ones. In order to guarantee that the camera converges to the desired pose, it is known that DVS requires a large photometric overlap between the initial and desired images. In our case, since most of the pixels in our desired image $\mathbf{I_o}$ are unknown, the only useful overlap is on the object itself, narrowing even further the convergence domain. It is interesting to observe that even if GANVS and CNNVS are two deep learning based methods that have been respectively trained and fine-tuned on the same dataset, our approach, which separates the features prediction and the camera control, provides significantly better results. CNNVS seems facing

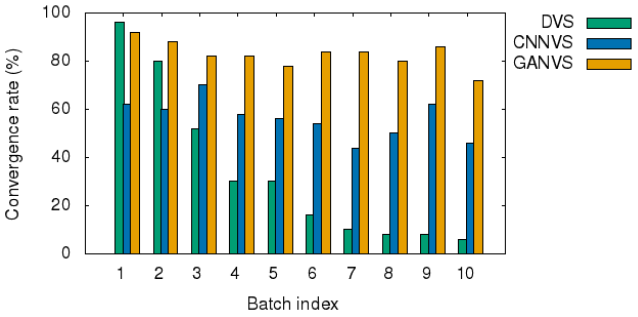


Fig. 4: Convergence rate comparison of DVS, CNNVS and GANVS methods for 10 batches of 50 individual runs. The difficulty of each batch is incremented linearly and each run is performed with a different ground texture.

more difficulties when the ground is particularly textured. Moreover, since CNNVS estimates the relative pose of the camera in scene space, the displacement follows a straight line in 3D space which corresponds to a curved trajectory of the object in image space. This sometimes leads the object to leave the field-of-view of the camera.

C. Real experiments

Experimental results were carried out using a Toyota HSR (Human Support Robot) [27]. The HSR has a 4 dof manipulator arm of revolute joints mounted on a prismatic torso and the whole is driven by an omnidirectional mobile base. A calibrated camera is embedded on the end-effector of the robot. We used our method to position the gripper of the HSR w.r.t. an object in different unknown and uncontrolled real environments (Fig. 5).

Fig. 5: The Toyota HSR performing precise positioning in different unknown environments.

The progress of two real experiments is illustrated in Fig. 6. These experiments have been conducted in a building hall next to large bay windows, thus the lighting conditions changed while GANVS was progressing. In the first experiment, the floor has two types of repetitive tiles, one dark and one light including different shapes. In the second experiment, the targeted object is a giraffe toy, surrounded by various unknown objects. Of course, the unknown objects do not appear in the object-only desired image.

Even with large initial displacements and high differences between the initial and the object-only desired images (Fig. 6a), the camera still converges to the desired pose with low final errors as shown by the final pixel-level alignments (Fig. 6f).

Due to paper length constraint, the curves of the photometric error, the camera velocities and the camera trajectory are not reported for these experiments. However, it can be seen in the complementary video clip that these curves and trajectories are similar to those obtained in simulation (Fig. 3g-3j), showing that GANVS has similar behaviours in real experiment and simulation. Moreover, while having been trained only on synthetic data, and even if no particular effort has been made on lighting conditions, GANVS reconstructed complex details from the background of the real images, including strong patterns, specular highlights, unknown objects and even the base of the robot that is partially visible in some of the experiments. Although parts of the generated keyframes may sometimes appear blurry, the images contain sufficient data for the photometric VS to converge to the desired pose.

VI. Conclusion and Future Work

We presented GANVS, a background-agnostic method for positioning a camera w.r.t. an object that combines the advantages and efficiency of photometric VS and the generalization ability of GAN. A dedicated GAN is used to predict intermediate visual keyframes between two images allowing to converge towards a desired pose following subsequent photometric VS. To the best of our knowledge, this is the first attempt at using a GAN to predict an image, visually representing a movement in space, to control a system. Our method can therefore be used even with no prior information about the scene around the targeted object and is also robust to varying and dynamic lighting conditions. Since several photometric VS are successively performed, the domain of convergence is very high, thus the displacement to position the camera can be very large. We tested and validated our method both in simulation and using a robotic platform, including qualitative and quantitative evaluations.

In the future, we intend to investigate the robustness of our method to partial occlusions and training an object-agnostic GANVS. Indeed, we actually conducted experiments in simulation placing 3D distractor models on top of the target object, covering large parts this latter. Even though GANVS has not been trained with occlusions, though the background and the distractors are completely unknown, and none of them are visible in the object-only image, it was able to correctly detect the target object, infer the translation and rotation and recreate the keyframe including the distractor objects allowing the VS scheme to converge to the desired pose. These encouraging results seem to indicate that the generalization ability of GANVS offers sufficient robustness to occlusions without being specifically trained for this purpose which simplifies the dataset creation.

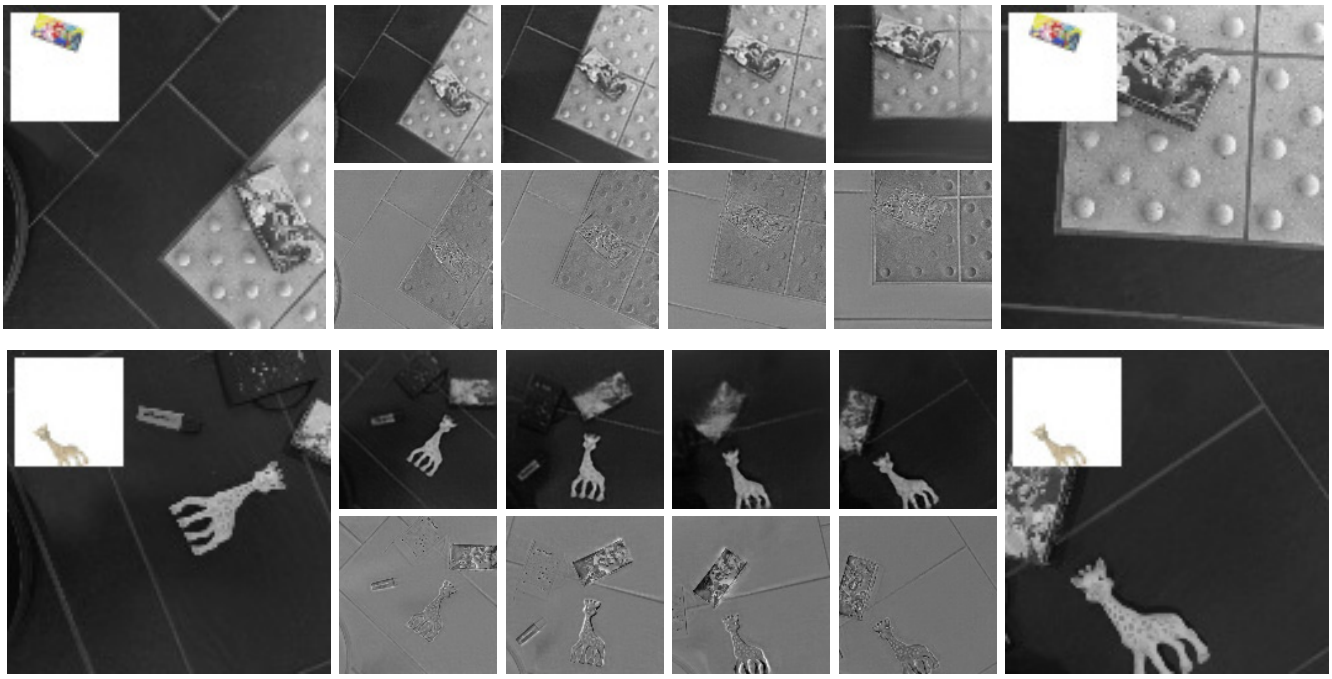
Additionally, even if GANVS is trained for a specific target object, it actually shows a moderate ability to generalize to target objects that the network has never seen before. Indeed, with both unknown background and unknown target object in simulation, GANVS was able to generate keyframes with correct reconstruction of the scene and correct displacement. However, the convergence accuracy is not satisfactory since the model oscillates around the desired pose. Preliminary results with fine-tuning on a small dataset for new object, containing less than 10k triplets of images, showed substantially improved convergence of GANVS and can be done rapidly (about 40 minutes). Both fine-tuning and modifying the dataset generation strategy to improve generalization to unknown objects should be investigated in future work.

References

[1] H. Okada, T. Inamura, and K. Wada, "What competitions were conducted in the service categories of the world robot summit?" *Advanced Robotics*, vol. 33, no. 17, pp. 900–910, 2019.

[2] L. Iocchi, D. Holz, J. Ruiz-del Solar, K. Sugiura, and T. van der Zant, "Robocup@home," *Artif. Intell.*, vol. 229, no. C, p. 258–281, 2015.

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



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

Fig. 6: Two real Experiments: (a) the initial $\mathbf{I}$ with $\mathbf{I_o}$ overlaid on its top left, (b-e) some keyframes $\mathbf{I_k}$ generated by GANVS (top row) and their corresponding image differences after the camera convergence (bottom row), and (f) the final $\mathbf{I}$.

[4] E. Marchand and F. Chaumette, "Feature tracking for visual servoing purposes," *Robotics and Autonomous Systems*, vol. 52, no. 1, pp. 53–70, 2005.

[5] J. Haviland, F. Dayoub, and P. Corke, "Predicting target feature configuration of non-stationary objects for grasping with image-based visual servoing," *CoRR*, vol. abs/2001.05650, 2020.

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

[7] M. Bakthavatchalam, O. Tahri, and F. Chaumette, "A Direct Dense Visual Servoing Approach using Photometric Moments," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1226–1239, Oct. 2018.

[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] E. Marchand, P. Bouthemy, and F. Chaumette, "A 2d–3d model-based approach to real-time visual tracking," *Image and Vision Computing*, vol. 19, no. 13, pp. 941–955, 2001.

[10] P. Han and G. Zhao, "A review of edge-based 3d tracking of rigid objects," *Virtual Reality & Intelligent Hardware*, vol. 1, no. 6, pp. 580 – 596, 2019.

[11] A. Saxena, H. Pandya, G. Kumar, A. Gaud, and K. M. Krishna, "Exploring convolutional networks for end-to-end visual servoing," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3817–3823.

[12] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2758–2766.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[15] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1–8.

[16] X. Fu, Y. Liu, and Z. Wang, "Active learning-based grasp for accurate industrial manipulation," *IEEE Trans. Automation Science and Engineering*, vol. 16, no. 4, pp. 1610–1618, 2019.

[17] Y. Harish, H. Pandya, A. Gaud, S. Terupally, S. Shankar, and K. M. Krishna, "Dfvs: Deep flow guided scene agnostic image based visual servoing," *arXiv preprint arXiv:2003.03766*, 2020.

[18] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1647–1655.

[19] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," *arXiv preprint arXiv:1812.11941*, 2018.

[20] A. X. Lee, S. Levine, and P. Abbeel, "Learning visual servoing with deep features and fitted q-iteration," *CoRR*, vol. abs/1703.11000, 2017. [Online]. Available: http://arxiv.org/abs/1703.11000

[21] C. Sampedro, A. Rodriguez-Ramos, I. Gil, L. Mejias, and P. Campoy, "Image-based visual servoing controller for multirotor aerial robots using deep reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 979–986.

[22] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2242–2251.

[23] O.-M. Pedersen, E. Misimi, and F. Chaumette, "Grasping Unknown Objects by Coupling Deep Reinforcement Learning, Generative Adversarial Networks, and Visual Servoing," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1–8.

[24] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 2017.

[25] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," in *International Conference on Neural Information Processing Systems (NIPS)*. Curran Associates Inc., 2017, p. 5769–5779.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[27] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, "Development of human support robot as the research platform of a domestic mobile manipulator," *ROBOMECH Journal*, vol. 6, no. 4, 2019.