2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

KOVIS: Keypoint-based Visual Servoing with Zero-Shot Sim-to-Real Transfer for Robotics Manipulation

En Yen Puang[1,2], Keng Peng Tee[1] and Wei Jing[1,2,*]

Abstract*— We present KOVIS, a novel learning-based, calibration-free visual servoing method for fine robotic manipulation tasks with eye-in-hand stereo camera system. We train the deep neural network only in the simulated environment; and the trained model could be directly used for real-world visual servoing tasks. KOVIS consists of two networks. The first keypoint network learns the keypoint representation from the image using with an autoencoder. Then the visual servoing network learns the motion based on keypoints extracted from the camera image. The two networks are trained end-to-end in the simulated environment by self-supervised learning without manual data labeling. After training with data augmentation, domain randomization, and adversarial examples, we are able to achieve zero-shot sim-to-real transfer to real-world robotic manipulation tasks. We demonstrate the effectiveness of the proposed method in both simulated environment and real-world experiment with different robotic manipulation tasks, including grasping, peg-in-hole insertion with 4mm clearance, and M13 screw insertion. The demo video is available at: *http://youtu.be/gfBJBR2tDzA

I. INTRODUCTION

Visual Servoing (VS) is a framework to control the motion based on the visual image input from the camera [1], [2]. VS could be used to provide flexible vision-guided motion for many robotic manipulation tasks such as grasping, insertion, pushing. Therefore, for intelligent robotic applications in unstructured environment, VS is highly desired for the robot to complete the manipulation tasks, by controlling the end-effector motion based on the visual feedback.

Traditionally, VS usually requires extracting and tracking of visual features, in order to estimate the pose differences between the current pose and the target pose [1], [2], [3]. The pose difference will then be used as feedback for the VS controller to move the robotic end-effector towards the target pose. The approach usually requires manually hand-crafted features for different applications, as well as manually labeling the target poses, which limits the task generalization of VS in robotic applications. As a result of recent advancement in deep learning, making use of deep Convolutional Neural Networks (CNN) allows direct learning of the controller for VS, instead of relying on manually handcraft features for the detection. Recent work on combining deep learning method with VS has been explored by researchers [4], [5]. However, these methods require collecting large amounts of training data from real experiments, and estimating camera pose differences based on input images. Thus, the target pose of the camera has to be identified for robotic manipulation.

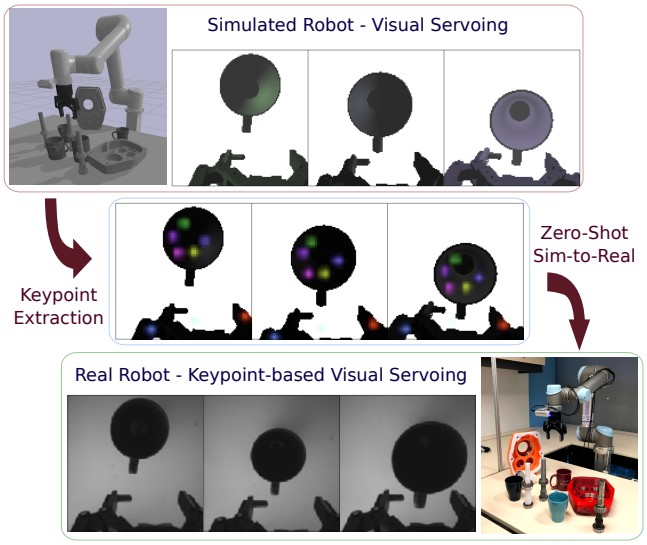

Fig. 1: KOVIS is a learning-based visual servoing framework for robotics manipulation tasks. It is trained entirely with synthetic data, and works on real-world scenarios with zero-shot transfer, using robust keypoints latent representation.

To achieve end-to-end, data-efficient deep learning based VS method for fine robotic manipulation tasks, we propose KOVIS, a **K**eyp**o**int based **Vi**sual **S**ervoing framework. KOVIS learns the VS controller that moves the robot end-effector to the target pose (e.g. pre-insertion pose) for manipulation tasks. The proposed KOVIS learns the VS only with synthesis data in simulated environment, and directly applies the learnt model in real-world robotic manipulator tasks.

The main contributions of this paper are:

- A general and efficient learning framework of VS for robotic manipulation tasks;
- A self-supervised keypoint representation learning with autoencoder to identify the "peg-and-hole" relationship between the tool and target, thus achieves calibration-free in hand-in-eye system for the manipulation;
- An end-to-end, self-supervised learning method for VS controller to move the robot end-effector to the target pose for manipulation without estimating the pose differences; and
- Combination of different training schemes to achieve zero-shot sim-to-real transfer for real-world fine manipulation tasks such as grasping and insertion.

[1] Institute for Infocomm Research (I²R), A*STAR, Singapore
[2] Institute for High Performance Computing (IHPC), A*STAR, Singapore
* Corresponding author, email addresses: {puangey, kptee, jingw}@i2r.a-star.edu.sg

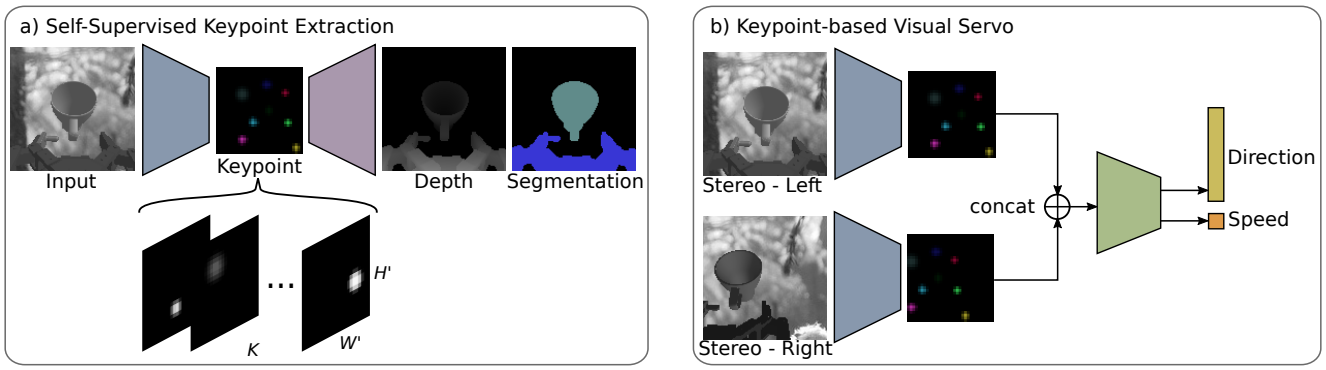

Fig. 2: Architecture overview: KOVIS consists of two major network modules that are trained together end-to-end: (a) Keypoint learning involves an autoencoder for self-supervision and a keypointer bottle-neck for robust latent representation; (b) Keypoint-based visual servo then predicts motion commands in terms of direction (unit-vector) and normalized speed (scalar) based-on the extracted keypoints from stereo inputs.

II. RELEVANT WORK

A. Visual Perception and Representation for Manipulation

Keypoints provide a compact representation for perception of images, and has been used in many computer vision applications such as image generation [6], face recognition [7], tracking [8] and pose estimation [9]. Recently, keypoints have also been used as a representation for robotic manipulation tasks. For example, [10], [11] used keypoints for manipulation planning of category tasks with supervised learning, and [12] focused on learning a task-specific keypoint representation for robotic manipulation via self-supervised learning.

Autoencoders and their variations have also been applied in many robotic applications to deal with perception in manipulation [13], [14]. An autoencoder maps the input image to a latent space, where the encoded latent value is usually used for planning [15] or as a feature extraction component for further learning process [14].

Unlike supervised keypoint learning methods for robotic manipulation, which mostly focused on representation learning, our work utilizes *self-supervised* learning with an autoencoder to extract the keypoints without any human labelling or annotations. In addition, our work includes learning VS controller based on the learnt keypoint representation.

B. Visual Servoing

Many research works have been conducted on VS in past years. Traditionally, VS algorithms usually rely on explicit geometric feature extraction [16], [17]. Direct VS methods have been explored, but they still require global feature based image processing methods [18], [19]. Recently, deep learning based methods have been used for VS, including CNN to learn the pose difference directly from the image input for the visual servoing [5], [20] and Siamese CNN to learn the pose differences by comparing the difference of images of the target and current poses [4].

The proposed KOVIS directly learns the end-to-end robotic motion based on automatically-extracted keypoints of the image input, without estimating the pose difference. Our task and environment setting is similar to [4]. However, our model is trained entirely in a simulated environment, without any data collected from real experiments.

C. Self-supervised learning and sim-to-real transfer for robotic manipulation

For robotic manipulation applications, it is expensive to get real-world data for learning-based methods. In order to overcome the data shortage problem in robotic learning applications, self-supervised learning in simulation environment [21], and efficient sim-to-real transfer, are important [22]. Several approaches have been proposed to address the sim-to-real problem, such as domain randomization [23], simulation randomization [24], adversarial training [25]. KOVIS is data-efficient by adopting domain randomization and adversarial example techniques to achieve zero-shot sim-to-real transfer without requiring any real-world data.

III. METHOD

The proposed framework KOVIS, as shown in Fig. 2, consists of two major modules: an autoencoder for learning a keypoint representation from input images; and a feed-forward network for learning VS motion commands. Both networks are trained end-to-end using ground-truth data gathered from simulations. To achieve zero-shot sim-to-real transfer, we also adapt several methods in the training to overcome the "reality-gap" and improve robustness in real-world manipulation scenarios.

A. Self-Supervised Keypoint Extraction

We first learn the latent keypoint representation of objects in a VS scene with an CNN-based autoencoder. For an input image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times C}$, we formulate the keypoint $\boldsymbol{k} \in \mathbb{R}^{H'\times W'\times K}$ as a latent representation in the autoencoder architecture where the encoder $f : \boldsymbol{x} \mapsto \boldsymbol{z}$, keypointer $\Phi : \boldsymbol{z} \mapsto \boldsymbol{k}$ and decoder $g : \boldsymbol{k} \mapsto \boldsymbol{x}'$ are optimized to minimize reconstruction loss in self-supervised approach:

$$\min_{\theta_f,\theta_g} \|\boldsymbol{x} - (g * \Phi * f)(\boldsymbol{x})\|_2 \tag{1}$$

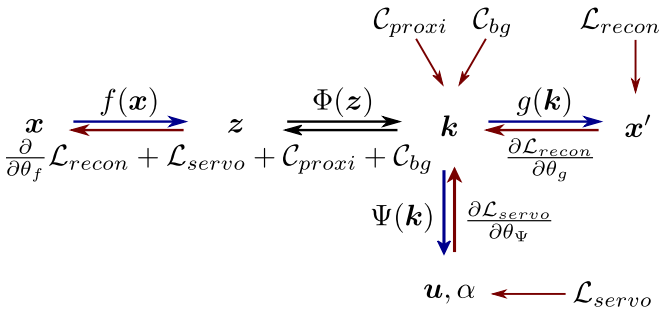

Fig. 3: Training of KOVIS: All components are trained end-to-end together with the differentiable keypointer Φ which contain no learnable parameters.

Succeeding the encoder, the keypointer Φ transforms the feature map $\boldsymbol{z}$ into K individual keypoints $\boldsymbol{k}_i$ on the 2D feature maps $\Omega = \mathbb{Z}^{H'\times W'}$ in two steps. First, the softmax of $\boldsymbol{z}$ is used for computing the channel-wise 2D centroid for each channel. Then a 2D Gaussian distribution with fixed covariance ρ^{-1} is used to model the unimodal Gaussian keypoint with mean the centroid of the channel's softmax:

$$\boldsymbol{j}_i^* = \sum_{\boldsymbol{j}\in\Omega} \boldsymbol{j} \frac{\exp(\boldsymbol{z}_i)}{\sum_\Omega \exp(\boldsymbol{z}_i)} \tag{2}$$

$$\alpha_i = \sigma\left(\max_\Omega(\boldsymbol{z}_i)\right) \tag{3}$$

where $\boldsymbol{j} = (j_1, j_2)$ represent the indices in vertical and horizontal axes in Ω, and $\boldsymbol{j}^*$ the centroid of the 2D feature map. This keypoint formulation is similar to [6] but with additional keypoint confidence α from the sigmoid $\sigma(\cdot)$ of the channel's maximum activation:

$$\boldsymbol{k}_i = \alpha_i \prod_{j=(j_1,j_2)} \exp\left(-\rho\,(j - j_i^*)^2\right) \quad \forall \boldsymbol{j}\in\Omega \tag{4}$$

Additionally, we enforce two soft constraints in the training of keypoints extraction to achieve better feature localization and representation. Since widely-spread keypoints are more distinct compared to concentrated ones, the first constraint $\mathcal{C}_{proxi}$ encourages better representation by pushing keypoints away from each other through penalizing the L2-norm among extracted keypoints centroid as in [26] with a hyper-parameter γ:

$$\mathcal{C}_{proxi} = \sum_{i,i'|i'>i}^{K} \alpha_i\alpha_{i'} \exp\left(-\gamma\|\boldsymbol{j}_i^* - \boldsymbol{j}_{i'}^*\|_2\right) \tag{5}$$

To prevent arbitrary keypoint formation which is bad for interpretability, the second constraint $\mathcal{C}_{bg}$ encourages better features localization by penalising any keypoints that fall into the background segmentation mask of the input image:

$$\mathcal{C}_{bg} = \sum_{i}^{K}\sum_{\boldsymbol{j}}^{\Omega} \mathbb{I}\left[\boldsymbol{j}\in B\right]\boldsymbol{k}_{i\boldsymbol{j}} \tag{6}$$

where $\mathbb{I}[\boldsymbol{j}\in B]$ is a binary logic that returns true when $\boldsymbol{j}$ is in the background set B of ground-truth segmentation mask (assumed to be unavailable during inference). Both

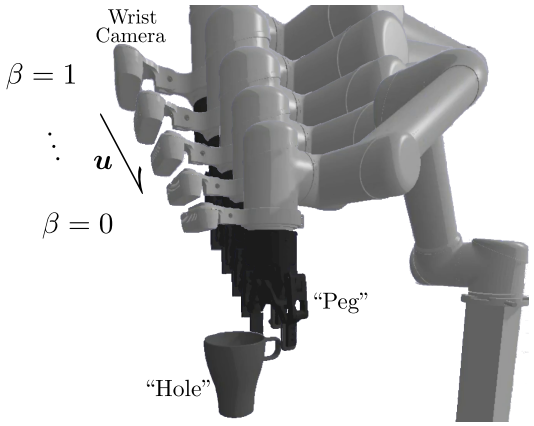

Fig. 4: Simulating visual servoing task with UR5 robot, 2-finger gripper and a wrist camera. For mug picking task, gripper is the "peg" and mug handle is the target "hole".

constraints are scaled by a keypoint confidence α to reduce contributions from low-confidence keypoints. The combination of these 2 constraints ensure that the extracted keypoints are localized within object of interest, and hence perform better at capturing essential information regarding object geometry for the downstream VS task.

B. Self-Supervised Keypoint-based Visual Servoing

VS is a type of reactive controller whose objective is to generate motion that minimizes the differences between the goal and current visual observation/feedback. In contrast to conventional VS in which the servo takes in both current and goal states, KOVIS trains a servo for a single task and hence only requires the current state as input during inference. Given the extracted keypoints $\boldsymbol{k}$ from input image, the servo $\Psi : \boldsymbol{k} \mapsto (\boldsymbol{u}, \beta)$ is a multi-layer Fully-Connected network trained with supervised learning approach to predict the motion of the robot end-effector. The predicted motion consists of the direction $\boldsymbol{u} \in \mathbb{R}^d \mid |\boldsymbol{u}| := 1$ of the end-effector, as well as its normalized speed $\beta \in [0, 1]$ which is 0 at the desired pose and saturates to 1 when far away.

This setup is not to align keypoints between current and goal states but to directly minimize the differences between the predicted and ground-truth motion. In other words, the servoing task is determined by the training data as depicted in Fig. 4. We define the loss function of the servo as:

$$\mathcal{L}_{servo} = \beta^*\left(1 - \frac{\boldsymbol{u}^\intercal\boldsymbol{u}^*}{|\boldsymbol{u}|}\right) + \text{BCE}(\beta, \beta^*) \tag{7}$$

which consists of a scaled inverted cosine similarity for $\boldsymbol{u}$ and binary-cross-entropy loss BCE$(\cdot)$ for β with $\boldsymbol{u}^*$ and β^* as the ground-truth. Note that the former is scaled by β^* to reduces the loss from direction when the speed is low.

C. End-to-End Training and Zero-Shot Transfer

KOVIS is trained end-to-end using the input images from eye-in-hand stereo camera to the motion commands in robot end-effector frame. The training is done entirely on synthetic data gathered from a simulated environment.

Training Data Generation. The first step for this framework therefore is to setup the simulated VS task for generating training data. This involves 1.) getting the CAD models of the involved objects, 2.) defining their target pose and motion for each of the components, and lastly 3.) the robotic and camera system according to the requirement of the intended manipulation task. Once the VS task is established, synthetic images paired with motion ground-truth are generated and then used for the training in Sections III-A and III-B. Data are recorded in the form of roll-outs. Each roll-out starts at the same target pose and then propagates out in the opposite of $\boldsymbol{u}$ with a speed factor, both randomized among roll-outs. Roll-out terminates early when collision occurs.

Object Anchoring. Motion commands are generated in end-effector frame based on images from a camera whose transformation relative to the end-effector is not calibrated. In other words, KOVIS realizes camera calibration-free configuration for manipulative VS by identifying "peg-AND-hole" relationship from the input image during training. This relationship requires the "hole" (i.e. normally the target object) to act as the pose anchor while the servo generates the motion command for the "peg" (e.g. gripper or tool) based on the relative pose between them, instead of the absolute camera-object pose that rely on deliberate camera calibrations. Moreover, the dimension of motion direction $\boldsymbol{u}$ is adapted accordingly to disambiguate symmetries in object geometry, as well as the requirement of the VS task itself.

Data Domain and Losses. "Reality-gap" [22] is a problem for models trained entirely in simulation without any real world adaptation. In staying agnostic towards object texture, the input image $\boldsymbol{x}$ is transformed into grayscale domain, and the decoder's outputs $\boldsymbol{x}' := \left(\boldsymbol{x}^d, \boldsymbol{x}^s\right)$ reconstructs the depth buffer $\boldsymbol{x}^d$ and semantic segmentation $\boldsymbol{x}^s$ only using the keypoint. This is to ensure that the keypoint encodes only the essential geometric information and not overfit subtle details or artifacts in the simulation. In staying agnostic towards the quality of real depth images, keypoints extracted from stereo pair images are used instead in training the servo by concatenating 2 sets of keypoints extracted individually from the left and right stereo images. As the result, without loss of generality, this setup assumed the use of stereo camera in the intended VS task.

The total loss function $\mathcal{L}$ for training KOVIS is the sum of all losses and constraints:

$$\mathcal{L} = \mathcal{L}_{recon} + \mathcal{L}_{servo} + \mathcal{C}_{proxi} + \mathcal{C}_{bg} \tag{8}$$

where $\mathcal{L}_{recon}$ is the reconstruction loss which involve mean-squared-error for depth buffer and multi-class cross-entropy for semantic segmentation for both left and right stereo input images. Fig. 3 depicts the semantics of end-to-end training of all components in KOVIS.

Adversarial Examples and Training. We adopt image augmentation (e.g. lighting, blurring and shifting) and domain randomization (background, texture, camera pose) [22] methods for the training in simulated environment. In addition, adversarial examples are used to effectively augment training data which the hand-crafted methods could not achieve [27].

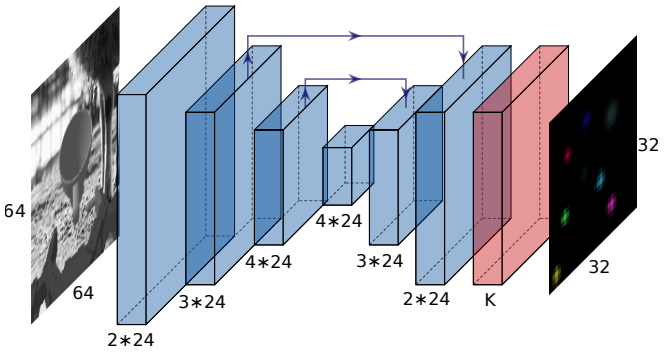

Fig. 5: The encoder (blue) is a U-Net with DenseNet backbone which has total 0.5M parameters. It consists of several DenseBlocks each with 2 to 4 DenseLayers and a growth-rate of 24. Output of keypointer (red) is a 32×32 ($H' \times W'$) feature map with K channels representing K keypoints.

Fig. 6: Visual servoing for pick-and-place alignment tasks: Pick-Mug (left), Insert-Shaft (mid) and Insert-Plug (right).

A randomized combination of multiple generation methods including Fast Gradient Sign Method (FGSM), iterative FGSM and least-likely FGSM are used together with random augmentation strength and number of iterations as in [14]. These augmentations are applied to the entire mini-batch before being used for training. This method, hereinafter referred to as Adex, aims to widen the domain that the encoder is able to operate in, thereby becoming more robust when handling real-world data, despite being trained only on synthetic ones.

IV. IMPLEMENTATIONS AND EXPERIMENTS

We implement KOVIS as the "last-mile" solution for robotics manipulation tasks. Here we demonstrate 3 applications on VS alignment for pick-and-place tasks with an eye-in-hand stereo camera mounted near robot end-effector as depicted in Fig. 4 and Fig. 6.

The encoder is a Fully-Convolutional U-Net [28] based on DenseNet [29] architecture with skip-connections (by concatenation) linking downward and upward streams, as

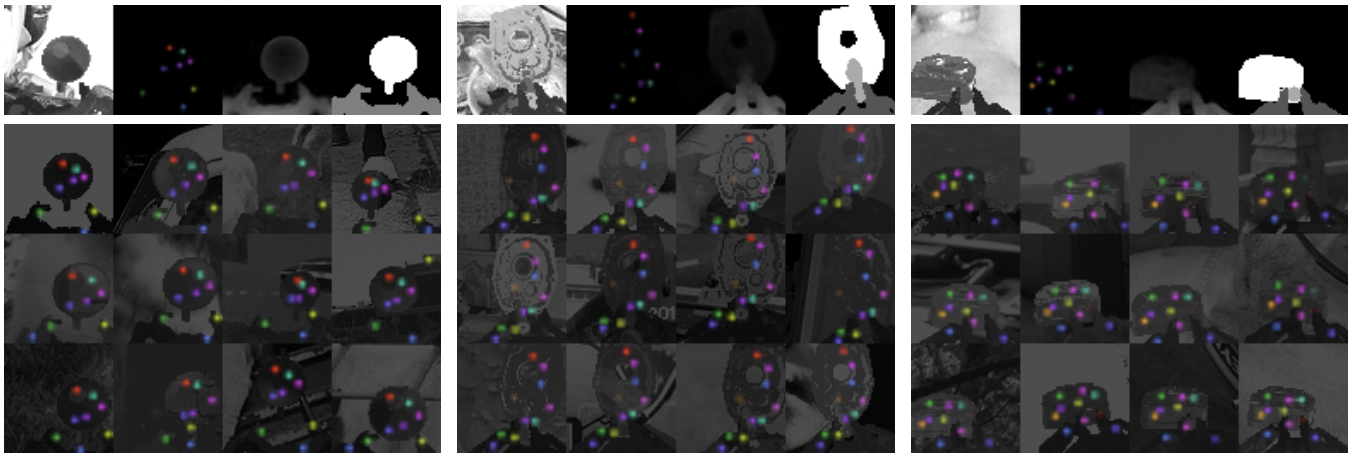

Fig. 7: Top row: Synthetic input image, extracted keypoint, predicted depth buffer and semantic segmentation for 3 VS tasks. Bottom row: Overlaid images between inputs and their respective extracted keypoints for the respectively VS tasks.

TABLE I: Ablation study on the effect of input image perturbations on keypoint localization. Average difference of keypoint location and confidence $(\delta \boldsymbol{j}^*, \delta\alpha)$ (the lower the better) between none and various perturbations are shown here. Keypoint location are normalized from $[0, H']$ and $[0, W']$ to $[0, 1]$ while keypoint confidence remains between $[0, 1]$ in percentage $\%$.

Background	Texture	Lighting	Pick-Mug		Insert-Shaft		Insert-Plug	
			w/o. Adex	with Adex	w/o. Adex	with Adex	w/o. Adex	with Adex
✓			(0.5, 0.7)	(0.5, 0.6)	(0.2, 0.1)	(0.2, 0.0)	(0.8, 1.2)	(1.1, 0.5)
	✓		(0.4, 0.5)	(0.4, 0.5)	(0.2, 0.0)	(0.2, 0.0)	(0.5, 1.0)	(0.5, 0.3)
		✓	(0.3, 0.4)	(0.3, 0.4)	(0.3, 0.1)	(0.4, 0.0)	(0.7, 1.1)	(0.8, 0.4)
✓	✓	✓	(0.6, 0.7)	(0.5, 0.6)	(0.3, 0.1)	(0.4, 0.0)	(0.9, 1.2)	(1.2, 0.5)
	Average		0.51	**0.48**	0.16	**0.15**	0.93	**0.66**

shown in Fig. 5. The decoder on the other hand is a feed-forward expanding DenseNets while the servo is made of multiple Fully-Connected layers. All DenseNet used here are of *Weight-BatchNorm-Activation* form and all convolutional layers has increasing dilation as in [14]. The total learnable parameters of KOVIS is around 1M.

The settings of hyper-parameters highly depend on the application scenarios. For applications that require high precision, higher ρ (Eq. 4), H and W (Eq. 1) are preferred. For applications that involve complex or large object geometry more keypoint K (Eq. 1) and smaller γ (Eq. 5) are preferred. In all of our experiments we use a fixed combination of 64×64 ($H \times W$) input image size, 32×32 ($H' \times W'$) keypoint spatial size, 2.5 for ρ in keypoint representation and 20 for γ in proximity constraint.

For the experiment setup, we use a UR5 robotic arm, a Robotiq 2F-85 gripper, and a Intel Realsense D435 camera (which we only use its stereo images) mounted on the robot arm. In the simulation environment [30], we create a virtual robot workspace by duplicating our real-world robotics system. For each of the VS task, we simulate 7000 servoing roll-outs and collected around 35000 set of images (i.e. color, depth and segmentation mask) paired with the ground-truth motion commands in the end-effector frame.

A. Keypoint Localization

In this experiment we evaluate the localization capability of the learnt keypoint and its robustness under input perturbations. First, we show the extracted keypoints from various viewing angles of the three VS tasks. Fig. 7 shows that keypoints consistently locate object of interest ("peg" and "'hole') in each of the tasks. Note that in the absence of severely challenging background, none of the keypoints are located outside of object of interest, and their formation change accordingly to its input. These show that the keypoint extraction encodes the correct object geometric information instead of just any arbitrary latent representation. Moreover, from Insert-Plug task ($K = 10$) we observe that excessive keypoints are automatically trimmed (low α_i) thanks to the soft constraints and keypoint confidence used in KOVIS.

Next, we evaluate keypoint stability under input image perturbations. Table I shows the ablation study on the effect of keypoint localization stability over input image perturbations. The differences of keypoint location $\delta \boldsymbol{j}^*$ and confidence $\delta\alpha$ are compared when background, texture and lighting perturbations are introduced , while camera pose perturbation is disabled. More than 400 synthetic input images from each tasks are sampled and the resulted keypoints after the perturbations are compared to itself before any perturbations. As shown in Table I, Adex generally reduces the changes in keypoint confidence but introduces noise in keypoint

TABLE II: Servo consistency in term of scaled inverted cosine similarity loss for direction and BCE loss for speed ($\mathcal{L}_{\boldsymbol{u}}, \mathcal{L}_{\beta}$) (the lower the better) over camera perturbations at different magnitudes (3D translations, 3D rotations).

Camera Perturbation	Pick-Mug		Insert-Shaft		Insert-Plug	
	w/o. Adex	with Adex	w/o. Adex	with Adex	w/o. Adex	with Adex
($\pm$0.5cm, $\pm2.5^\circ$)	(0.022, 0.153)	(0.021, 0.161)	(0.018, 0.165)	(0.014, 0.161)	(0.039, 0.173)	(0.031, 0.174)
($\pm$1.0cm, $\pm5.0^\circ$)	(0.024, 0.167)	(0.028, 0.167)	(0.024, 0.178)	(0.018, 0.172)	(0.043, 0.183)	(0.035, 0.183)
($\pm$1.5cm, $\pm7.5^\circ$)	(0.039, 0.177)	(0.042, 0.182)	(0.041, 0.215)	(0.039, 0.213)	(0.047, 0.214)	(0.043, 0.213)
Average	(**0.028, 0.166**)	(0.033, 0.170)	(0.028, 0.186)	(**0.024**, **0.182**)	(0.043, 0.190)	(**0.036**, **0.190**)

location. Moreover, random background changes has the highest influences in keypoint localization than the others. Stability of keypoint localization of objects, nonetheless, is robust against input perturbations as the fluctuations only appeared to be at the scale of 0.1%.

Lastly we evaluate the performance of object anchoring through camera pose perturbations. During synthetic training data generation the 6D camera pose is perturbed ($\pm$1cm, $\pm5^\circ$) for every image captured to encourage object anchoring during training, described in Section III-C. Here we perturb the camera pose with different magnitude and observe the error (with respect to its ground-truth) of the predicted motion commands in term of scaled inverted cosine similarity loss for direction $\mathcal{L}_{\boldsymbol{u}}$ and BCE loss for speed $\mathcal{L}_{\beta}$ shown in Eq. 7. As shown in Table II quality of motion commands deteriorates as the perturbation increases. On the other hand, performance is improved by Adex and KOVIS is able to maintain robustness even when magnitude of camera pose perturbations is higher than how it was trained.

B. Alignment with Eye-in-hand Camera

In this experiment we evaluate KOVIS in real world VS manipulation tasks. Tasks depicted in Fig. 6 are different in their object size and margin of error. The first task is a mug picking task in which the robot is required to place its hook into the mug's handle as the gripper closes. The second and third task are insert-task which involve a shaft on tapered roller bearing and a plug on M13 screw respectively. The robot is required to orient the "peg" with the "hole" before a downward push to complete the insertion. All tasks are being trained with 4 dimensional motion ($d = 4$ in Eq. 7) namely xyz translation and xy in-plane rotation, and with 5, 10 and 16 number of keypoint. Table III shows the success rate under complex background over 10 runs each. Success rate is hindered by sub-optimal number of keypoint that causes under or over-fitting. Besides, Adex improves the performance only when the number of keypoint is appropriate. In general, KOVIS is able to complete these 3 servo tasks with more than 90% success rate using Adex and 10 keypoints.

Besides accuracy, we show the servoing smoothness by tracking the robot end-effector trajectory from multiple initial poses. Fig 8 depicts 10 trajectories for each of the tasks. Error of end-effector pose in 4-dimensional from the goal pose are plotted against normalized time (stretched to accommodate trajectories with various length). Trajectories are generally smooth while following the predicted direction $\boldsymbol{u}$ with small magnitude linearly proportional to β, partly due to the nature of speed control at robot end-effector. As the result each trajectory takes about 6 seconds to complete depending on the initial pose.

TABLE III: Success rate of KOVIS on three real world VS tasks under complex background with multiple number of keypoint and use of Adex.

Servo Task	$K = 5$	K = 10	K = 16
Pick-Mug			
w/o. Adex	1.0	1.0	1.0
with Adex	1.0	1.0	1.0
Insert-Shaft			
w/o. Adex	0.2	0.6	0.6
with Adex	0.0	**1.0**	0.6
Insert-Plug			
w/o. Adex	0.8	0.8	0.9
with Adex	0.6	**0.9**	0.2

V. CONCLUSION

In this paper, we presented KOVIS, a **K**eyp**o**int based **Vi**sual **S**ervoing framework for the "last-mile" robotic manipulation task. KOVIS learns an efficient and effective keypoint representation for identifying object geometric information in robotic VS settings. It consists two major modules, an autoencoder for keypoint extraction, and a VS network for learning the robotic motion. Both networks are trained end-to-end and entirely on synthetic data. KOVIS does not require any real-world data or adaptation and achieves zero-shot sim-to-real transfer by having multiple data augmentations strategies. In addition, external calibration of hand-in-eye camera is not required for manipulation tasks due to KOVIS's end-to-end nature (input image to motion command) and its ability to identify the geometric "peg-and-hole" relationships between the tool and the target. The effectiveness of the proposed KOVIS has been demonstrated in several fine robotic manipulation tasks with high success rate. Through experiments we also demonstrate the stability the predicted keypoints and motion commands over input image and camera pose perturbations. The future work will be on extending the current methods to category-level generalization for robotic manipulation tasks, as well as using multi-modality sensory information to achieve better performance on fine manipulation tasks.

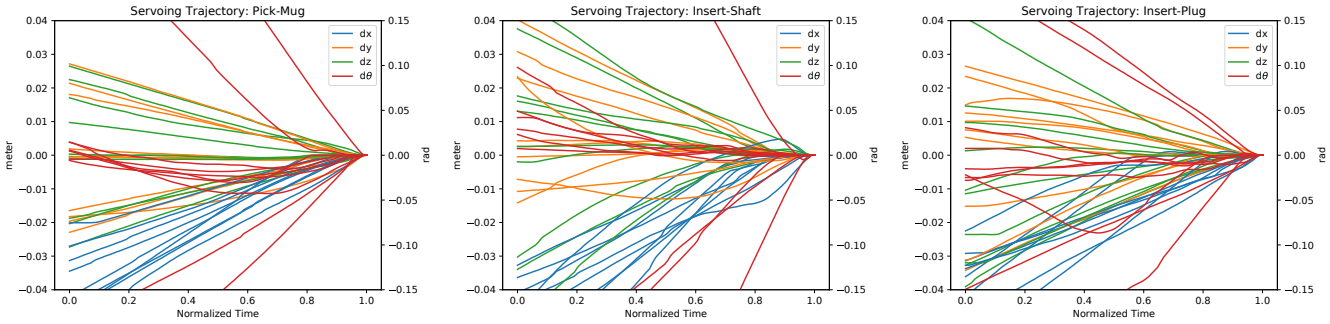

Fig. 8: Trajectories of robot end-effector in 4 dimensional pose against normalized time for each of the three VS tasks.

ACKNOWLEDGEMENT

This research is supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under its AME Programmatic Funding Scheme (Project #A18A2b0046).

REFERENCES

[1] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE transactions on robotics and automation*, vol. 12, no. 5, pp. 651–670, 1996.

[2] D. Kragic, H. I. Christensen, *et al.*, "Survey on visual servoing for manipulation," *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, vol. 15, p. 2002, 2002.

[3] F. Chaumette, S. Hutchinson, and P. Corke, "Visual servoing," in *Springer Handbook of Robotics*. Springer, 2016, pp. 841–866.

[4] C. Yu, Z. Cai, H. Pham, and Q.-C. Pham, "Siamese convolutional neural network for sub-millimeter-accurate camera pose estimation and visual servoing," *arXiv preprint arXiv:1903.04713*, 2019.

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

[6] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, "Unsupervised learning of object landmarks through conditional image generation," in *Advances in Neural Information Processing Systems*, 2018, pp. 4016–4027.

[7] S. Berretti, B. B. Amor, M. Daoudi, and A. Del Bimbo, "3d facial expression recognition using sift descriptors of automatically detected keypoints," *The Visual Computer*, vol. 27, no. 11, p. 1021, 2011.

[8] S. Hare, A. Saffari, and P. H. Torr, "Efficient online structured output learning for keypoint-based object tracking," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1894–1901.

[9] S. Tulsiani and J. Malik, "Viewpoints and keypoints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1510–1519.

[10] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kpam: Keypoint affordances for category-level robotic manipulation," *arXiv preprint arXiv:1903.06684*, 2019.

[11] W. Gao and R. Tedrake, "kpam-sc: Generalizable manipulation planning using keypoint affordance and shape completion," *arXiv preprint arXiv:1909.06980*, 2019.

[12] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," *arXiv preprint arXiv:1910.11977*, 2019.

[13] A. Byravan, F. Lceb, F. Meier, and D. Fox, "Se3-pose-nets: Structured deep dynamics models for visuomotor control," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–8.

[14] E. Y. Puang, P. Lehner, Z.-C. Marton, M. Durner, R. Triebel, and A. Albu-Schäffer, "Visual repetition sampling for robot manipulation planning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9236–9242.

[15] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2118–2124.

[16] C. Cai, E. Dean-León, D. Mendoza, N. Somani, and A. Knoll, "Uncalibrated 3d stereo image-based dynamic visual servoing for robot manipulators," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 63–70.

[17] C. Cai, N. Somani, and A. Knoll, "Orthogonal image features for visual servoing of a 6-dof manipulator with uncalibrated stereo cameras," *IEEE transactions on Robotics*, vol. 32, no. 2, pp. 452–461, 2016.

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

[19] ——, "Particle filter-based direct visual servoing," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4180–4186.

[20] 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.

[21] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *The International Journal of Robotics Research*, p. 0278364919872545, 2019.

[22] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[23] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 1–8.

[24] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[25] F. Zhang, J. Leitner, Z. Ge, M. Milford, and P. Corke, "Adversarial discriminative sim-to-real transfer of visuo-motor policies," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1229–1245, 2019.

[26] Y. Zhang, Y. Guo, Y. Jin, Y. Luo, Z. He, and H. Lee, "Unsupervised discovery of object landmarks as structural representations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2694–2703.

[27] C. Xie, M. Tan, B. Gong, J. Wang, A. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," *arXiv preprint arXiv:1911.09665*, 2019.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[29] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[30] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.