Transformer Driven Visual Servoing for Fabric Texture Matching Using Dual-Arm Manipulator

***Abstract*— In this paper, we propose a method to align and place a fabric piece on top of another using a dual-arm manipulator and a grayscale camera, so that their surface textures are accurately matched. We propose a novel control scheme that combines Transformer-driven visual servoing with dual-arm impedance control. This approach enables the system to simultaneously control the pose of the fabric piece and place it onto the underlying one while applying tension to keep the fabric piece flat. Our transformer-based network incorporates pre-trained backbones and a newly introduced Difference Extraction Attention Module (DEAM), which significantly enhances pose difference prediction accuracy. Trained entirely on synthetic images generated using rendering software, the network enables zero-shot deployment in real-world scenarios without requiring prior training on specific fabric textures. Real-world experiments demonstrate that the proposed system accurately aligns fabric pieces with different textures.**

I. INTRODUCTION

In the garment manufacturing process, robotic handling of fabric pieces is one of the technologies being explored to replace human labor. As shown in Fig.1, we propose a method for aligning and placing a fabric piece (Fabric A) on top of another (Fabric B) so that their surface textures match. This is a common task in garment production, such as aligning the texture of a pocket with that of a shirt.

In our proposed system, both edges of Fabric A are rolled up and constrained by end-effectors [1] attached to a dual-arm manipulator. We then introduce a novel motion control scheme based on visual servoing, which enables precise pose control of Fabric A while maintaining its flatness using the dual-arm manipulator. We make the following assumptions:

- Fabric A and Fabric B have the same texture.
- A unique solution is assumed for texture matching; thus, repetitive patterns such as checkerboards are excluded from consideration.

Over the years, visual servoing [2] has evolved significantly, particularly with the advent of deep learning techniques such as convolutional neural networks (CNNs). Recent advances in CNN-based visual servoing [3]–[8] have greatly expanded its applications by eliminating the need for manual feature engineering and precise camera calibration. These advances have enabled more robust performance in tasks such as connector insertion [5] and assembly task [6], [8], even under challenging conditions such as varying lighting [4], [8] and occlusion [4], [8] compared to conventional visual servoing. Recent studies have achieved fabric positioning and flattening with dual-arm manipulators using CNN-based visual servoing [9], where a projected pattern highlights wrinkles for fabric shape control. The network is trained on real images from physical experiments.

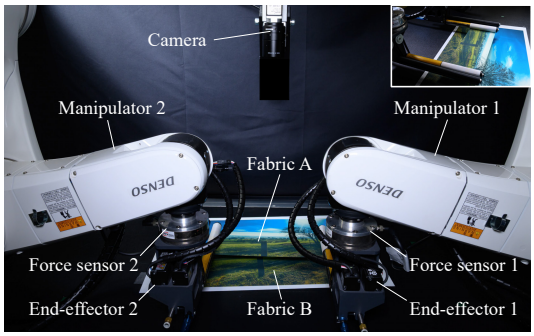


Fig. 1. Overview of the dual-arm manipulator system used for fabric alignment and placement.

In contrast to CNN-based visual servoing, other studies have applied non-learning-based visual servoing frameworks to manipulate deformable objects. For example, Shetab-Bushehri et al. [10] proposed a tracking and servoing method that controls the shape of elastic objects by defining a lattice around the object and servoing the lattice. Qi et al. [11] proposed a control strategy that uses contour moments and finite-time model estimation to enable the manipulation of composite rigid-deformable objects. Despite differences in methodology, most existing visual servoing approaches for deformable object positioning share several fundamental limitations: (a) reliance on visible object edges or contours, which limits applicability when these features are occluded; (b) limited adaptability to textured deformable objects; (c) dependence on large-scale real-world training data; and (d) reliance on simplified deformation models, which often fail to generalize to different object types.

To address prior limitations, we propose a novel method that combines Transformer-based visual servoing with dual-arm impedance control, enabling precise alignment and placement of fabric pieces using solely their texture patterns. The Transformer network, trained entirely on synthetic data, achieves zero-shot sim-to-real generalization and robustly aligns a wide range of previously unseen fabric textures. Simultaneously, the dual-arm impedance controller applies internal forces to keep the fabric flat, ensuring that the texture patterns of both fabric pieces are consistent, which is essential for accurate visual servoing.

The contributions of this paper can be summarized as follows:

1) We propose a new motion control scheme for a dual-arm manipulator that employs Transformer-driven visual servoing and dual-arm impedance control. This

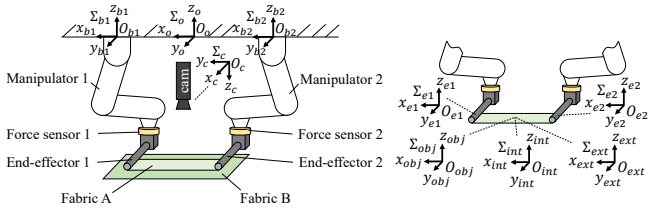


Fig. 2. Definition of coordinate systems.

scheme enables precise alignment and placement of a fabric piece on top of another, so that their textures accurately match.

2) We propose a new network architecture for visual servoing based on a pre-trained backbone and a newly proposed Difference Extraction Attention Module (DEAM), which significantly improves the pose difference prediction accuracy.
3) The proposed network is trained entirely on images generated using rendering software, enabling zero-shot application in real-world environments without prior training on specific fabric textures.
4) The experiments using unseen textures demonstrate that our system can align the fabric piece with an average position error of 0.1 mm. We further show through experiments that the system is capable of aligning fabric pieces under various unseen textures and unseen lighting conditions.

The remainder of this paper is organized as follows. Section II defines the coordinate systems used for the control system. Section III describes the proposed dual-arm motion control scheme for fabric texture matching. Section IV presents the experimental results. Section V concludes the paper.

II. COORDINATE SYSTEMS

The coordinate systems are defined as shown in Fig. 2. Let Σ_o denote the world coordinate system, defined as $O_o{-}x_o y_o z_o$. Let Σ_{b1} and Σ_{b2} denote the base coordinate systems $O_{b1}{-}x_{b1}y_{b1}z_{b1}$ and $O_{b2}{-}x_{b2}y_{b2}z_{b2}$, which are attached to the bases of Manipulator 1 and Manipulator 2, respectively. Let Σ_c denote the camera coordinate system $O_c{-}x_c y_c z_c$, which is attached to the camera. The homogeneous transformation matrices between the world coordinate system Σ_o and each of the coordinate systems Σ_{b1}, Σ_{b2}, and Σ_c are assumed to be obtained through camera calibration.

Let Σ_{e1} and Σ_{e2} denote the end-effector coordinate systems $O_{e1}{-}x_{e1}y_{e1}z_{e1}$ and $O_{e2}{-}x_{e2}y_{e2}z_{e2}$, which are attached to the end-effectors of Manipulator 1 and Manipulator 2, respectively. The homogeneous transformation matrix between Σ_{b1} and Σ_{e1} can be calculated using the forward kinematics of the manipulator. Similarly, the transformation between Σ_{b2} and Σ_{e2} can also be obtained from the forward kinematics calculation.

We define Σ_{obj} as the object coordinate system $O_{obj}{-}x_{obj}y_{obj}z_{obj}$, which represents the pose of Fabric A. Σ_{obj} is attached to the midpoint between Σ_{e1} and Σ_{e2}. The z-axis of Σ_{obj} is aligned with that of the world coordinate system Σ_o, and the x-axis is defined as the unit vector pointing from the origin of Σ_{e1} to the origin of Σ_{e2}. We also define the internal and external force coordinate systems as $\Sigma_{int} = O_{int}{-}x_{int}y_{int}z_{int}$ and $\Sigma_{ext} = O_{ext}{-}x_{ext}y_{ext}z_{ext}$ used for impedance control. Note that Σ_{obj}, Σ_{int}, and Σ_{ext} are identical.

III. PROPOSED METHOD

Fig.3 shows the block diagram of the proposed control scheme. The Transformer-driven visual servoing (in green) controls the pose of Fabric A to align its texture with that of Fabric B, using grayscale images captured by a monocular camera. The dual-arm impedance control (in blue) maintains texture consistency between Fabric A and Fabric B by flattening the fabric by controlling the internal force. The details of both the Transformer-driven visual servoing and the dual-arm impedance control are explained in the following subsections.

A. Visual Servoing Control

1) Control Law: As shown in Fig. 3, the network takes two images as input: the desired image $\boldsymbol{I}_{des}$ and the current image $\boldsymbol{I}(t)$. The desired image $\boldsymbol{I}_{des}$ is an image of Fabric B captured before each visual servoing execution, while $\boldsymbol{I}(t)$ is a grayscale image of Fabric A and part of Fabric B captured at time t during visual servoing. Example input image pairs, $\boldsymbol{I}_{des}$ and $\boldsymbol{I}$, are shown in Fig. 4.

Let ${}^c\boldsymbol{q}_{obj} = [{}^c\boldsymbol{t}_{obj}^T, {}^c\boldsymbol{r}_{obj}^T]^T$ be the stack of the three-dimensional position vector and the XYZ Euler rotation vector representing the current Fabric A pose in the camera coordinate system. Similarly, let ${}^c\boldsymbol{q}^*_{obj} = [{}^c\boldsymbol{t}^{*T}_{obj}, {}^c\boldsymbol{r}^{*T}_{obj}]^T$ denote the desired Fabric A pose in the camera coordinate system. The output of the network is defined as

$$ {}^c\boldsymbol{q}_{obj}(t) - {}^c\boldsymbol{q}^*_{obj} = f(\boldsymbol{I}_{des}, \boldsymbol{I}(t)), \tag{1} $$

where f is a neural network that maps the image pair $(\boldsymbol{I}_{des}, \boldsymbol{I}(t))$ to the pose difference. Although the subtraction operator is used for notational simplicity, the rotational difference is calculated by converting the poses to rotation matrices or quaternions, instead of subtracting Euler angles directly.

The desired velocity command for visual servoing, ${}^c\dot{\boldsymbol{q}}_{obj}$, represented in the camera coordinate system are computed as

$$ {}^c\dot{\boldsymbol{q}}_{obj}(t) = -\boldsymbol{\Lambda} \cdot ({}^c\boldsymbol{q}_{obj}(t) - {}^c\boldsymbol{q}^*_{obj}), \tag{2} $$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{6\times 6}$ is a diagonal, positive definite gain matrix defined as

$$ \boldsymbol{\Lambda} = \mathrm{diag}(\lambda_{tx}, \lambda_{ty}, \lambda_{tz}, \lambda_{rx}, \lambda_{ry}, \lambda_{rz}), \tag{3} $$

where λ_{tx}, λ_{ty}, and λ_{tz} are the proportional gains for translational errors along the x-, y-, and z-axes, respectively, and λ_{rx}, λ_{ry}, and λ_{rz} are the gains for rotational errors. In our experiments, λ_{tz}, λ_{rx}, and λ_{ry} are defined as zero. x-axis translation, y-axis translation, and z-axis rotation are

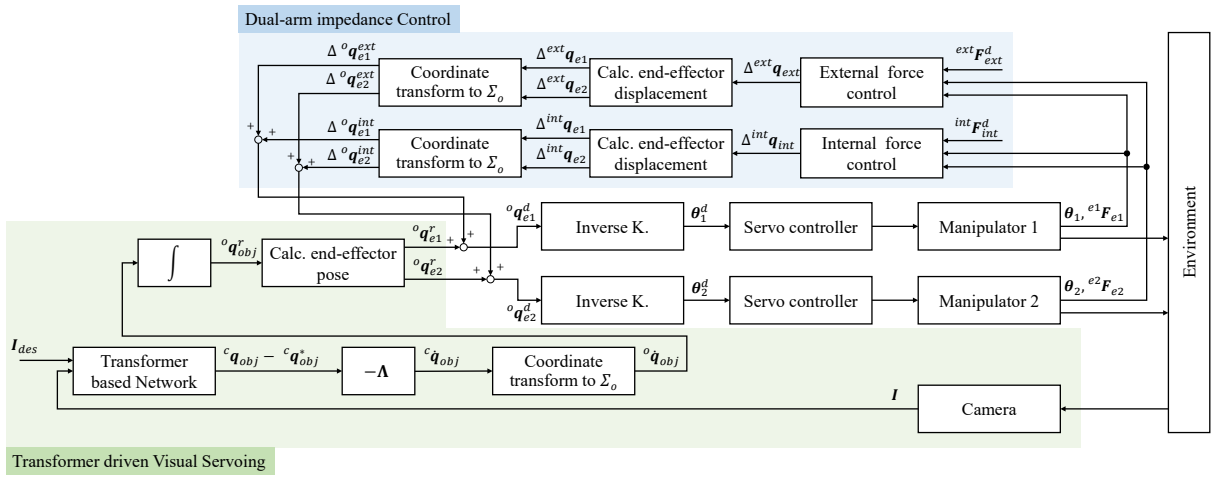


Fig. 3. Block diagram of the control system. The proposed control system consists of Transformer-driven visual servoing (in green) and dual-arm impedance control (in blue).

Fig. 4. Example of input image pairs fed to the network: $\boldsymbol{I}_{des}$ and $\boldsymbol{I}$. $\boldsymbol{I}_{des}$ is an image of Fabric B and $\boldsymbol{I}$ is an image of Fabric B occluded by Fabric A.

controlled by visual servoing, while z-axis translation, x-axis rotation, and y-axis rotation are controlled by impedance control.

Then, the desired velocity command ${}^o\dot{\boldsymbol{q}}_{obj}$, represented in Σ_o, is computed by applying a coordinate transformation to ${}^c\dot{\boldsymbol{q}}_{obj}$ expressed in Σ_c. The reference pose ${}^o\boldsymbol{q}^r_{obj}$, which is the desired Fabric A pose in visual servoing, is calculated as

$$ {}^o\boldsymbol{q}^r_{obj} = {}^o\boldsymbol{q}_{obj}(0) + \int_0^{t+T} {}^o\dot{\boldsymbol{q}}_{obj}(\tau)\, d\tau, \tag{4} $$

where T is the control cycle time of the visual servoing loop. Finally, the reference poses of the end-effectors, ${}^o\boldsymbol{q}^r_{e1}$ and ${}^o\boldsymbol{q}^r_{e2}$, are computed from ${}^o\boldsymbol{q}^r_{obj}$ using the transformations between the object coordinate system and the each end-effector coordinate system.

2) Network Architecture: The conventional CNN-based visual servoing methods typically concatenate or subtract features from Siamese backbones before passing them into fully connected layers [5], [8]. The proposed network architecture consists of a pre-trained backbone and newly proposed Difference Extraction Attention Modules (DEAMs), as illustrated in Fig. 5.

First, the network processes two input images (current and desired) independently using shared-weight backbones. The features from each backbone are first projected into a lower-dimensional embedding space via a 1×1 convolutional layer, followed by Layer Normalization and a GELU activation function (2D Conv1×1–LN–GELU). The features are then passed into the K layers of Difference Extraction Attention Module (DEAM) to progressively extract feature differences. The two feature maps output from the final DEAM layer are concatenated, globally average pooled (GA Pool), and then passed into fully connected layers (FC) to predict the 6-DoF pose difference (translation and rotation) of Fabric A.

As shown in Fig.6, each DEAM comprises three components: (1) Convolutional Blocks, (2) Transformer Blocks, and (3) a Difference Extraction Block. DEAM begins with a stack of Convolutional Blocks that employ the channel expansion–compression strategy from EfficientNet [12], [13]. Each block consists of a 1×1 convolutional layer, followed by Batch Normalization and a GELU activation function (2D Conv1×1–BN–GELU), and then a depthwise 3×3 convolutional layer with Batch Normalization and GELU activation (DW Conv3×3–BN–GELU). The resulting features are subsequently fed into a proposed module named Dynamic Convolution by Attention Blocks (DCAB), as illustrated in Fig. 7.

DCAB is built upon a dynamic convolutional layer [14] with a 3×3 kernel (DynConv3×3), whose convolution kernels are dynamically generated from input-dependent attention scores computed using Efficient Channel Attention (ECA) [15]. The output of DCAB is fed into the final layer of the Convolutional Block: a 2D Conv1×1–BN layer that projects the features back to their original channel dimension. After L layers of the Convolutional Block, the extracted features are projected into a lower-dimensional space by a 2D Conv1×1 layer, unfolded [16], [17], and then passed into Transformer Blocks.

The Transformer Block is a cross-attention mechanism [18] that applies a separable attention mechanism [17]. Separable attention improves computational efficiency by decoupling the attention operation into spatial and channel-wise components. The Transformer mechanism takes ”key” and ”value” features from one image and ”query” features from the other, processes them through separable attention, and then through a feed-forward network consisting of 2D Conv1×1–GELU and 2D Conv1×1 layers. The output fea-

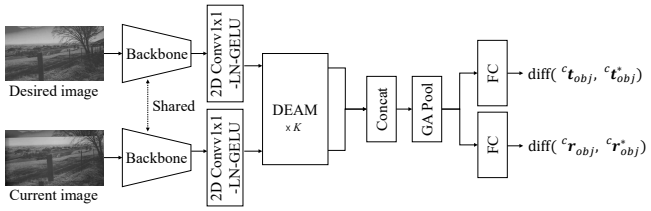


Fig. 5. Overall architecture of the Transformer-driven visual servoing network.

tures are then folded back to their original spatial dimensions. The Transformer Block is repeated M layers.

To enhance sensitivity to feature-level differences, the element-wise difference between the two streams is concatenated with each of the original feature maps. The resulting features are passed through a 2D Conv1×1–BN–GELU layer to project them back to their original channel dimension.

3) Training Dataset: The training dataset is generated using the rendering software Blender [19]. The end-effectors, Fabric A, Fabric B, the camera, and LED light sources are all simulated within the environment. The physics engine in Blender is used to simulate fabric deformation, and image rendering is performed using the Cycles renderer. During each rendering session, the poses of the fabric pieces and the camera, as well as the intensity and positions of the light sources, are randomly varied within predefined ranges.

To generate diverse fabric deformations, the relative position between the two end-effectors grasping the fabric is also randomly changed within a specified range. In addition, the number of light sources is randomly varied. The textures of the fabric pieces are randomly assigned using images obtained via the Pexels API. Examples of the rendered images are shown in Fig. 8, where the first row contains the desired images and the second row shows the current images.

B. Dual-Arm Impedance Control

To make the fabric piece flattened during the visual servoing, dual-arm impedance control is utilized. The dual-arm impedance control in this paper employs coordinated motion control based on the concept of a virtual mechanism [20]–[23] to control both the external and internal forces applied to the end-effectors and Fabric A.

1) External Force Control: Let ${}^{ext}\boldsymbol{F}_{ext} = [{}^{ext}\boldsymbol{f}_{ext}^T, {}^{ext}\boldsymbol{\eta}_{ext}^T]^T \in \mathbb{R}^6$ denote the equivalent force and torque at the origin of the external force coordinate system, applied by the two end-effectors. ${}^{ext}\boldsymbol{F}_{ext}$ is computed from the forces and torques applied to the two end-effectors, ${}^{e1}\boldsymbol{F}_{e1} \in \mathbb{R}^6$ and ${}^{e2}\boldsymbol{F}_{e2} \in \mathbb{R}^6$. For details, please refer to [23].

The external force is controlled according to the dynamics of a virtual mechanism based on the impedance model, as

$$\boldsymbol{M}_{ext}\Delta^{ext}\ddot{\boldsymbol{q}}_{ext} + \boldsymbol{D}_{ext}\Delta^{ext}\dot{\boldsymbol{q}}_{ext} + \boldsymbol{K}_{ext}\Delta^{ext}\boldsymbol{q}_{ext} = \boldsymbol{S}_{ext}({}^{ext}\boldsymbol{F}_{ext} - {}^{ext}\boldsymbol{F}^d_{ext}), \quad (5)$$

where $\boldsymbol{M}_{ext} \in \mathbb{R}^{6\times6}$, $\boldsymbol{D}_{ext} \in \mathbb{R}^{6\times6}$, and $\boldsymbol{K}_{ext} \in \mathbb{R}^{6\times6}$ are the inertia, damping, and stiffness matrices for external force control, respectively. $\Delta^{ext}\boldsymbol{q}_{ext}$ denotes the displacement from the reference pose of Fabric A, ${}^o\boldsymbol{q}^r_{obj}$, which is computed by the visual servoing control described in the previous subsection. ${}^{ext}\boldsymbol{F}^d_{ext} \in \mathbb{R}^6$ is the desired external force expressed in the external force coordinate system. The selection matrix $\boldsymbol{S}_{ext} = \mathrm{diag}(0,0,1,1,1,0) \in \mathbb{R}^{6\times6}$ determines which axes are controlled by external force control.

The displacement of each end-effector, $\Delta^o\boldsymbol{q}_{e1}$ and $\Delta^o\boldsymbol{q}_{e2}$, is calculated from $\Delta^{ext}\boldsymbol{q}_{ext}$ using the transformation matrices between the external force coordinate system and the two end-effector coordinate systems.

2) Internal Force Control: Let us define ${}^{int}\boldsymbol{F}_{int} = [{}^{int}\boldsymbol{f}_{int}^T, {}^{int}\boldsymbol{\eta}_{int}^T]^T \in \mathbb{R}^6$ as the internal force applied by the two end-effectors of Manipulator 1 and Manipulator 2. ${}^{int}\boldsymbol{F}_{int}$ is computed based on the method described in [23].

The internal force is controlled based on the model as

$$\boldsymbol{M}_{int}\Delta^{int}\ddot{\boldsymbol{q}}_{int} + \boldsymbol{D}_{int}\Delta^{int}\dot{\boldsymbol{q}}_{int} = \boldsymbol{S}_{int}({}^{int}\boldsymbol{F}_{int} - {}^{int}\boldsymbol{F}^d_{int}), \quad (6)$$

where $\boldsymbol{M}_{int} \in \mathbb{R}^{6\times6}$ and $\boldsymbol{D}_{int} \in \mathbb{R}^{6\times6}$ are the inertia and damping matrices for internal force control, respectively. $\Delta^{int}\boldsymbol{q}_{int}$ denotes the displacement from the initial relative pose of the two end-effectors with respect to the internal force coordinate system. ${}^{int}\boldsymbol{F}^d_{int} \in \mathbb{R}^6$ is the desired internal force expressed in the internal force coordinate system. The selection matrix $\boldsymbol{S}_{int} = \mathrm{diag}(1,0,0,0,0,0) \in \mathbb{R}^{6\times6}$ determines which axes are controlled by internal force control.

The displacements of each end-effector, $\Delta^o\boldsymbol{q}_{e1}$ and $\Delta^o\boldsymbol{q}_{e2}$ are computed from $\Delta^{int}\boldsymbol{q}_{int}$ using the transformation matrices between the internal force coordinate system and the end-effector coordinate systems. Finally, the desired pose of the end-effectors of Manipulator 1 and Manipulator 2 are computed as

$${}^o\boldsymbol{q}^d_{e1} = {}^o\boldsymbol{q}^r_{e1} + \Delta^o\boldsymbol{q}^{ext}_{e1} + \Delta^o\boldsymbol{q}^{int}_{e1}, \quad (7)$$

$${}^o\boldsymbol{q}^d_{e2} = {}^o\boldsymbol{q}^r_{e2} + \Delta^o\boldsymbol{q}^{ext}_{e2} + \Delta^o\boldsymbol{q}^{int}_{e2}. \quad (8)$$

The desired joint angles of each manipulator commanded to each robot servo controller, $\boldsymbol{\theta}^d_1 \in \mathbb{R}^6$ and $\boldsymbol{\theta}^d_2 \in \mathbb{R}^6$, are obtained by solving the inverse kinematics based on the desired end-effector poses.

IV. EXPERIMENTAL RESULTS

A. Network Comparison

For the backbone network, we adopt the Vision Transformer (ViT) [24] with a patch size of 32, considering both prediction accuracy and computational cost. The backbone networks are pre-trained on ImageNet. The entire network, including the two backbones and the DEAM ($K = 1$, $L = 1$, $M = 1$), is trained for 100 epochs using the AdamW [25] optimizer, with a batch size of 64 (8 samples per GPU × 8 GPUs).

We use a total of 100,000 samples for training (with 10% held out for validation) and 58,000 samples for testing.

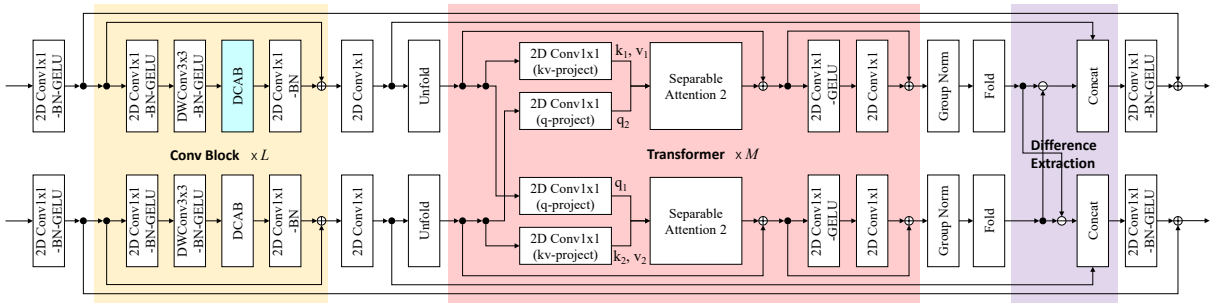


Fig. 6. Architecture of the Difference Extraction Attention Module (DEAM), which combines convolutional blocks, Transformer blocks, and a difference extraction blocks to enhance prediction accuracy.

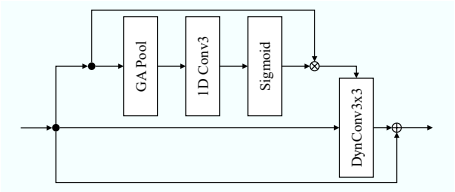


Fig. 7. Architecture of the Dynamic Convolution by Attention Blocks (DCAB).

Fig. 8. Examples of synthetic image pairs used in training. Top row: desired images. Bottom row: corresponding current images.

The input size of both the current and desired images is 960×540 pixels (width $\times$ height). The learning rate is linearly increased from 1×10^{-6} to 1×10^{-4} over the first 5 epochs, and then decayed to 1×10^{-5} using a cosine annealing schedule [26]. The loss function is defined as

$$E = \alpha \left\| (\widehat{{}^c\boldsymbol{t}_{obj} - {}^c\boldsymbol{t}^*_{obj}}) - ({}^c\boldsymbol{t}_{obj} - {}^c\boldsymbol{t}^*_{obj}) \right\|_2 + \beta \left\| (\widehat{{}^c\boldsymbol{r}_{obj} - {}^c\boldsymbol{r}^*_{obj}}) - ({}^c\boldsymbol{r}_{obj} - {}^c\boldsymbol{r}^*_{obj}) \right\|_2, \quad (9)$$

where $(\widehat{{}^c\boldsymbol{t}_{obj} - {}^c\boldsymbol{t}^*_{obj}})$ and $(\widehat{{}^c\boldsymbol{r}_{obj} - {}^c\boldsymbol{r}^*_{obj}})$ denote the predicted translational and rotational difference, respectively. $({}^c\boldsymbol{t}_{obj} - {}^c\boldsymbol{t}^*_{obj})$ and $({}^c\boldsymbol{r}_{obj} - {}^c\boldsymbol{r}^*_{obj})$ denote the ground-truth of translational and rotational difference, respectively. The coefficients $\alpha = 1.0$ and $\beta = 1.0$ are used to weight the translation and rotation losses. Note that both the input images and output vectors are normalized to the range $[0.0, 1.0]$ using min–max normalization based on their respective minimum and maximum values. In our dataset, the fabric piece is randomly moved within a translation range of ± 20 mm and a rotation range of $\pm 10°$ around each axis.

Table I shows a comparison between the proposed network and several other architectures. The CONCAT methods follow a Siamese architecture, in which two shared-weight ViT [24] backbones extract features from the input image pair. In CONCAT, the extracted features from backbone are directly concatenated and passed to fully connected layers [5].

The proposed method, DEAM (DCAB), achieves the highest accuracy, significantly reducing the loss E (defined in eq. (9)) compared to CONCAT. This demonstrates the effectiveness of DEAM with DCAB, which enhances pose difference prediction accuracy by explicitly capturing feature differences through a combination of convolutional, transformer, and difference extraction blocks. When the difference extraction block is removed, as in DEAM (DCAB, W/O DIFF. EXT. BLOCK), performance degrades notably, highlighting its essential role.

We also evaluated DEAM variants in which the DCAB module is replaced with existing attention mechanisms: DEAM (SE [12], [13], [27]), DEAM (ECA [15]), and DEAM (GRN [28]). Among them, DEAM (SE) and DEAM (GRN) show relatively strong performance, but none outperform DEAM (DCAB). Although all DEAM variants incur slightly higher GFLOPs than CONCAT, their inference latency remains below 0.017 seconds per image with a batch size of 1.

Finally, as a supplementary comparison, we applied SE [12], [13], [27], ECA [15], and GRN [28] independently to each of the two feature maps extracted from the shared backbone, before concatenation and fully connected layers. While these attention modules are typically designed to be embedded within backbone networks, we include them here to provide additional points of reference. Their performance is notably lower than that of the DEAM-based methods, highlighting the advantage of DEAM's design, which explicitly extracts image feature differences for improved prediction accuracy.

B. Experimental Results of Visual Servoing

The system used for the experiments is shown in Fig. 1. It consists of two 6-DOF manipulators (Denso VS-068), each equipped with a force sensor (ATI Axia80) mounted on the wrist, and a camera (Basler acA1920-155um) that captures grayscale images at a resolution of 960×540 pixels and a frame rate of 60 fps. The spatial resolution of the camera is 0.0249 mm/pixel. FabricA is a rectangular piece measuring

TABLE I

PERFORMANCE COMPARISON OF DIFFERENT NETWORK ARCHITECTURES

Method	Loss E $[\times 10^{-3}]$ ↓	Trans / Rot RMSE ↓	Std $[\times 10^{-3}]$ ↓	GFLOPs ↓	Latency [sec/image] ↓
CONCAT [5]	8.067	6.650 / 1.879	2.411	**78.272**	**0.0157**
DEAM (DCAB)	**3.831**	**1.654 / 1.341**	**1.410**	81.865	0.0167
DEAM (DCAB, W/O DIFF. EXT. BLOCK)	4.720	1.975 / 1.664	1.677	81.769	0.0167
DEAM (SE)	4.073	1.786 / 1.419	1.482	81.850	0.0165
DEAM (ECA)	4.460	1.923 / 1.560	1.601	81.845	0.0165
DEAM (GRN)	4.049	1.741 / 1.417	1.469	81.843	0.0165
SE [12], [13], [27]	7.968	6.658 / 1.843	2.382	78.273	0.0158
ECA [15]	8.006	6.544 / 1.880	2.407	78.273	0.0158
GRN [28]	8.062	6.632 / 1.881	2.411	78.272	0.0158

† E represents the loss computed using normalized network outputs, defined in eq. (9). Translation and Rotation RMSE values are denormalized and reported in millimeters [mm] and degrees [deg], respectively. Std denotes the standard deviation of E over the test set. Inference latency is measured per image with batch size = 1.

100×400 mm, and FabricB measures 297×420 mm.

Five different textures (Texture 1 to 5), which are not included in the training dataset, are used in the experiments. To ensure the aligning accuracy in real-world experiments, the proposed DEAM (DCAB) ($K = 5$, $L = \{2, 2, 3, 3, 3\}$, $M = \{2, 2, 3, 3, 4\}$) is trained on an extended dataset comprising 200,000 samples for 300 epochs. When running on the robot-control PC equipped with an NVIDIA RTX 4090 GPU, the network exhibited a latency of approximately 34.5 ms during visual servoing. The visual servoing gains are tuned empirically.

The experiments are carried out according to the following procedure:

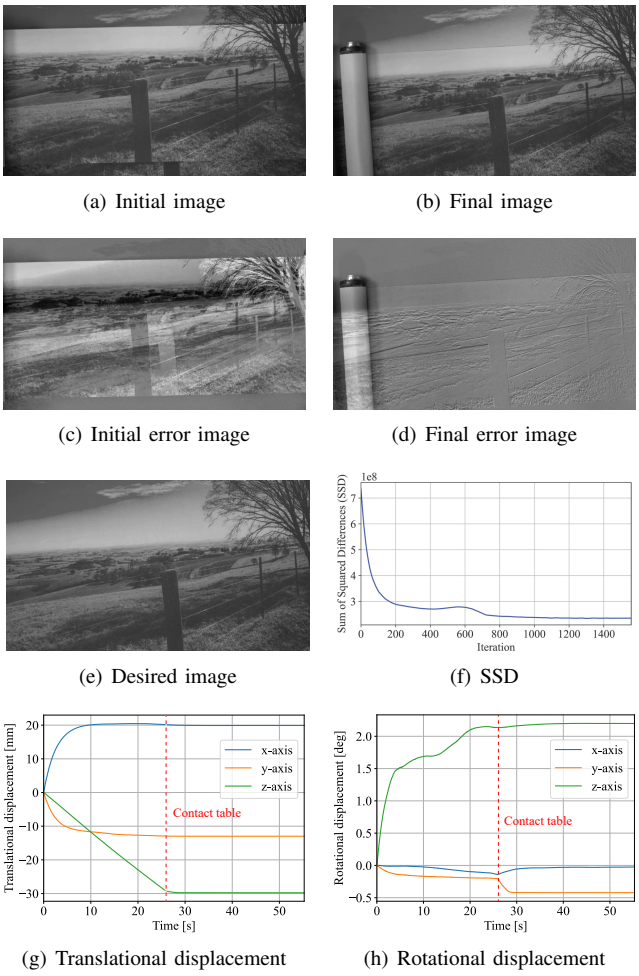
1) Fabric A and Fabric B are randomly placed on a flat table.
2) An image of Fabric B is captured and stored as the desired image.
3) Fabric A is detected (the detection algorithm is omitted due to the page limitations) and grasped by the robots' end-effectors.
4) The grasped Fabric A is moved to a fixed pose above Fabric B.
5) The proposed control scheme using DEAM (DCAB) is initiated.

1) Visual Servoing of Unseen Textures: Fig. 9 shows an example result using Texture 1. Texture 1 depicts an open field with a distinctive fence at the center and a tree in the upper right, providing strong visual cues. The proposed control scheme successfully aligns and places Fabric A onto Fabric B in all 30 trials. After visual servoing, the image error ($\boldsymbol{I} - \boldsymbol{I}_{des}$) is significantly reduced compared to the initial state, as shown in Fig. 9 (c) and (d). The sum of squared differences (SSD) between the desired and current images decreases through visual servoing as shown in Fig. 9 (f). Despite occlusion caused by the end-effectors, the proposed method maintains accurate alignment, demonstrating robustness to occlusion. As shown in Fig. 9 (g) and (h), the Fabric A pose converges through visual servoing. The figures also show that the end-effectors make contact with the table at around 23 seconds.

To quantitatively evaluate the positioning accuracy, we compute the average distance between corresponding feature points matched between the final and desired images over 30 visual servoing trials. Note that, due to the difficulty of measuring the actual poses of Fabric A and B, we instead evaluate the average distance between corresponding feature points. Feature points are extracted using SIFT, and mismatches or irrelevant correspondences are manually removed. Using the remaining 7,478 matched points, we calculate the average Euclidean distance between the corresponding points based on camera parameters. The overall average positioning error is 0.106 mm, with a standard deviation of 0.043 mm and a maximum error of 0.450 mm.

Fig. 10 presents the result of direct visual servoing [29] under the same conditions. Unlike the proposed method, direct visual servoing often fails to converge and sometimes even diverges, resulting in low positioning accuracy. However, it is worth noting that when the initial pose error is small (within approximately 3 mm and $0.5°$), direct visual servoing can achieve remarkably good performance. The failure under larger errors is mainly due to two factors. First, the image-based cost function is highly nonlinear with respect to the pose. When the initial error is large, the optimization can easily get trapped in local minima. Second, as the robot approaches the desired pose, the end-effectors occlude the fabric piece in the camera view, causing an appearance mismatch between the current and desired images. This occlusion severely affects the convergence of direct visual servoing.

Fig. 11 summarizes the qualitative results of the proposed visual servoing using Texture 1 to 5. The proposed control scheme successfully aligns Fabric A with Fabric B across all textures, demonstrating strong generalization to previously unseen textures. Each texture presents unique challenges. Texture 2 is a landscape image captured by the author in Hong Kong, characterized by dense, detailed patterns that provide rich visual features. Texture 3 contains a distinctive object, a cat, located near the center of the image. Texture 4 features a sharply focused cherry blossom at the center, while the surrounding blurred blossoms result in a low-contrast appearance. Texture 5 is the most challenging, as it contains sparsely distributed patterns of bananas with low saliency and exhibits a low-contrast appearance, providing only weak visual cues for feature extraction. Despite these limitations,



(a) Initial image (b) Final image

(c) Initial error image (d) Final error image

(e) Desired image (f) SSD

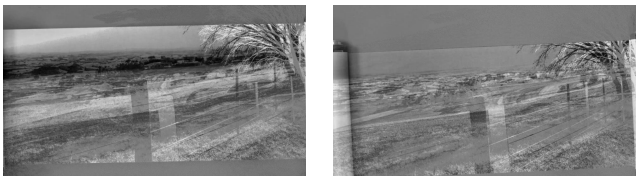
(g) Translational displacement (h) Rotational displacement

Fig. 9. Visual servoing result for Texture 1 using the proposed scheme. (a) Initial image. (b) Final image after alignment. (c,d) Error image ($\boldsymbol{I} - \boldsymbol{I}_{des}$) before and after visual servoing. (e) Desired image. (f) Sum of squared differences (SSD) between $\boldsymbol{I}$ and $\boldsymbol{I}_{des}$ over time. (g,h) Translational and rotational displacement of the object coordinate system from the initial pose.

the proposed system achieves a reasonable level of alignment even for Texture 5. These results highlight the robustness and generalization capability of the proposed network, demonstrating successful alignment even with previously unseen and visually challenging textures.

Furthermore, we conduct additional experiments in which the grasped fabric piece is in a naturally sagging configuration at the start of visual servoing. The proposed method still succeeds in achieving alignment. The results are provided in the supplementary video.

2) Visual Servoing Under Various Lightning Conditions: To evaluate the generalization capability of the network under varying lighting conditions, experiments using Texture 1 are conducted under both dark and bright lighting conditions. The results are shown in Fig. 12. Despite the presence of shadows and partial occlusions caused by the end-effectors, the proposed control scheme maintains stable convergence behavior. These results demonstrate the robustness of the proposed method against environmental disturbances such as lighting variations and occlusions.



(a) Initial error image (b) Final error image

Fig. 10. Result of direct visual servoing for Texture 1. (a, b) Error images before and after visual servoing.

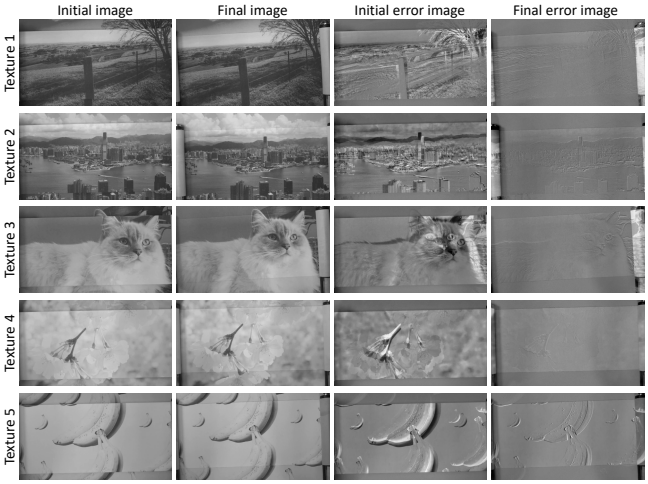


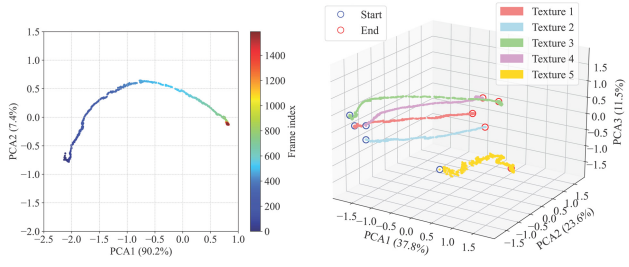
Fig. 11. Qualitative results of the proposed visual servoing method across all five textures (Texture 1 to 5).

3) Discussion: To gain insight into the feature representations of the network, we perform principal component analysis (PCA) on the intermediate features extracted from the global average pooling (GAP) layer. Fig. 13 (a) shows the PCA results of features extracted during visual servoing with Texture 1, based on the experiments in Fig. 9. The extracted features form a continuous trajectory from the start to the end of visual servoing in the PCA space, suggesting that the network has learned a well-structured feature representation.

Fig. 13 (b) shows the PCA results of features extracted during visual servoing with all five textures, based on the experiments in Fig. 11. The features corresponding to each texture form a continuous trajectory in the feature space, demonstrating the generalization capability of the proposed network to textures not included in the training dataset. The clear separability between features of different textures further indicates that the network effectively captures texture-specific differences, which is essential for achieving robust visual servoing across previously unseen textures.

A notable observation is the less consistent trajectory of Texture 5, which contains sparse patterns and a low-contrast appearance, making feature extraction more challenging. In contrast, textures with rich visual details and strong contrast, such as Texture 1 to Texture 4, exhibit more consistent trajectories. These results indicate that the network learns a structured and discriminative feature space that enables robust visual servoing across diverse texture types. While

Fig. 12. Results of visual servoing for Texture 1 under varying lighting conditions (dark and bright).



(a) Texture 1 (b) Texture 1 to 5

Fig. 13. Principal component analysis (PCA) of image features during visual servoing. (a) Features for Texture 1 in PCA space. (b) Features for Textures 1 to 5 in PCA space.

low-saliency and low-contrast textures remain more challenging, performance may be further improved by incorporating additional visual cues, such as structured light.

V. CONCLUSION

This paper proposes a method to align and place a fabric piece on another using a dual-arm manipulator and a grayscale camera, enabling accurate texture matching. The system combines Transformer-based visual servoing and dual-arm impedance control to control pose while keeping the fabric piece flat. Trained solely on synthetic images, the network enables zero-shot generalization to real fabrics. Experiments confirm accurate alignment across diverse textures. Code is available at: `https://github.com/tfductft/paper-code.git`.

REFERENCES

[1] A. Kobayashi, W. Dong, A. Seino, F. Tokuda, and K. Kosuge, "Rollup: Rolling-up end-effector for fabric handing," *Authorea Preprints*, 2025.

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

[3] A. Saxena, H. Pandya, G. Kumar, A. Gaud, and K. M. Krishna, "Exploring convolutional networks for end-to-end visual servoing," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3817–3823.

[4] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 3307–3314.

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

[6] F. Tokuda, S. Arai, and K. Kosuge, "Object positioning by visual servoing based on deep learning," *Transactions of the Society of Instrument and Control Engineers*, vol. 55, pp. 717–725, 01 2019.

[7] Y. Harish, H. Pandya, A. Gaud, S. Terupally, S. Shankar, and K. M. Krishna, "Dfvs: Deep flow guided scene agnostic image based visual servoing," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9000–9006.

[8] F. Tokuda, S. Arai, and K. Kosuge, "Convolutional neural network-based visual servoing for eye-to-hand manipulator," *IEEE Access*, vol. 9, pp. 91 820–91 835, 2021.

[9] F. Tokuda, A. Seino, A. Kobayashi, and K. Kosuge, "CNN-based visual servoing for simultaneous positioning and flattening of soft fabric parts," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 748–754.

[10] M. Shetab-Bushehri, M. Aranda, Y. Mezouar, and E. Özgür, "Lattice-based shape tracking and servoing of elastic objects," *IEEE Transactions on Robotics*, vol. 40, pp. 364–381, 2023.

[11] J. Qi, G. Ma, J. Zhu, P. Zhou, Y. Lyu, H. Zhang, and D. Navarro-Alarcon, "Contour moments based manipulation of composite rigid-deformable objects with finite time model estimation and shape/position control," *IEEE/ASME transactions on mechatronics*, vol. 27, no. 5, pp. 2985–2996, 2021.

[12] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[13] ——, "Efficientnetv2: Smaller models and faster training," in *International conference on machine learning*. PMLR, 2021, pp. 10 096–10 106.

[14] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic convolution: Attention over convolution kernels," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 030–11 039.

[15] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 534–11 542.

[16] S. Mehta and M. Rastegari, "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer," *arXiv preprint arXiv:2110.02178*, 2021.

[17] ——, "Separable self-attention for mobile vision transformers," *arXiv preprint arXiv:2206.02680*, 2022.

[18] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[19] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[20] K. Kosuge, K. Furuta, and T. Yokoyama, "Virtual internal model following control of robot arms," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, 1987, pp. 1549–1554.

[21] K. Kosuge, H. Yoshida, T. Fukuda, M. Sakai, K. Kanitani, and K. Hariki, "Unified control for dynamic cooperative manipulation," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94)*, vol. 2, 1994, pp. 1042–1047 vol.2.

[22] K. Kosuge, S. Hashimoto, and K. Takeo, "Coordinated motion control of multiple robots manipulating a large object," in *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97*, vol. 1, 1997, pp. 208–213 vol.1.

[23] K. Kosuge, K. Kamei, and T. Nammoto, "Coordinated motion control of dual manipulators for handling a rigid object with non-negligible deformation," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 5145–5151.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[25] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[26] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM journal on control and optimization*, vol. 30, no. 4, pp. 838–855, 1992.

[27] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[28] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie, "Convnext v2: Co-designing and scaling convnets with masked autoencoders," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 16 133–16 142.

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