IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

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

Fuyuki Tokuda, *Member, IEEE*, Akira Seino, Akinari Kobayashi, Kai Tang, *Graduate Student Member, IEEE*, and Kazuhiro Kosuge, *Life Fellow, IEEE*

***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.**

***Index Terms*—Visual servoing, dual arm manipulation, deep learning for visual perception, compliance and impedance control, perception for grasping and manipulation.**

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 assume that Fabric A and Fabric B have the same texture.

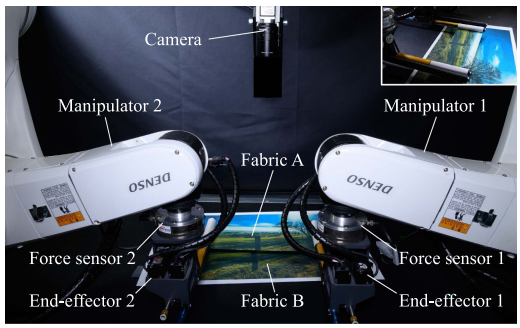


Fig. 1. Overview of the dual-arm manipulator system used for fabric alignment and placement.

In this paper, 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 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

Received 1 August 2025; accepted 19 November 2025. Date of publication 11 December 2025; date of current version 18 December 2025. This article was recommended for publication by Associate Editor N. Rojas and Editor J. Borras Sol upon evaluation of the reviewers' comments. This work was supported in part by the Innovation and Technology Commission of the HKSAR Government through InnoHK initiative, in part by the JC STEM Lab of Robotics for Soft Materials through The Hong Kong Jockey Club Charities Trust, and in part by Tohoku University through the Joint Research Program. *(Corresponding author: Fuyuki Tokuda.)*

The authors are with the JC STEM Lab of Robotics for Soft Materials, Department of Electrical and Electronic Engineering, Faculty of Engineering, The University of Hong Kong, Hong Kong SAR, China, and also with the Centre for Transformative Garment Production, Hong Kong SAR, China (e-mail: fuyuki.tokuda@transgp.hk; akira.seino@transgp.hk; akinari.kobayashi@transgp.hk; tangkai@eee.hku.hk; kosuge@hku.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3643335, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3643335

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

experiments that the system is capable of aligning fabric pieces under various unseen textures and unseen lighting conditions.

II. Related Work

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], [4], [5], [6], [7], [8] have expanded its applications by eliminating the need for manual feature engineering and precise camera calibration. These approaches have demonstrated robust performance in tasks such as connector insertion [5] and assembly [6], [8], even under challenging conditions such as varying lighting [4], [8] and occlusion [4], [8]. More recently, CNN-based visual servoing has been extended to deformable object manipulation. Tokuda et al. [9] achieved fabric positioning and flattening using a dual-arm manipulator and a projected pattern to highlight wrinkles. However, the network in their work was trained on real images collected from physical experiments, which limits its generalization to unseen textures and lighting conditions.

In contrast to CNN-based frameworks, several studies have explored non-learning-based visual servoing for deformable object control. Shetab-Bushehri et al. [10] proposed a lattice-based tracking and servoing method for controlling elastic object shapes, while Qi et al. [11] introduced a contour-moment-based control strategy with finite-time model estimation for composite rigid–deformable objects. Despite methodological differences, most existing visual servoing approaches for deformable objects share several limitations: (a) reliance on visible edges or contours, restricting applicability under occlusion; (b) limited adaptability to textured deformable objects; and (c) dependence on large-scale real-world datasets.

Beyond visual servoing, recent policy/trajectory learning has demonstrated impressive generalization capabilities in robotic manipulation. Diffusion Policy [12] learns visuomotor control through action diffusion, enabling robust imitation learning from demonstration data. Similarly, TraKDis [13] adopts a transformer-based knowledge distillation framework for reinforcement learning, achieving effective policy transfer to deformable object tasks such as cloth manipulation. From a planning perspective, Wang et al. [14] proposed Deformation-Aware RRT*, which incorporates deformation modeling into sampling-based motion planning for planar fabric repositioning. These methods focus primarily on policy learning or trajectory optimization, whereas visual servoing provides real-time, closed-loop control for precise alignment and positioning.

In summary, previous methods have advanced visual servoing and policy/trajectory learning methods for both rigid and deformable objects. However, CNN-based and non-learning visual servoing frameworks still struggle to adapt to different textures and lighting, and rely heavily on real-world data. Inspired by recent progress in policy and trajectory learning that utilizes Transformer-based attention architectures, our approach introduces these capabilities into visual servoing to achieve texture-based alignment using only synthetic data, unifying visual perception and dual-arm impedance control for zero-shot real-world deployment.

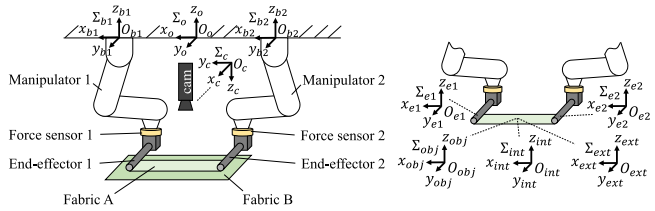


Fig. 2. Definition of coordinate systems.

III. Coordinate Systems

The coordinate systems are defined, as shown in Fig. 2. The world coordinate system ($\Sigma_o = O_o{-}x_o y_o z_o$), the bases of the manipulators (Manipulator 1 and Manipulator 2) Σ_{b1} and Σ_{b2}, and the camera coordinate system Σ_c are related through homogeneous transformations obtained by camera-robot calibration. Each end-effector coordinate system, Σ_{e1} and Σ_{e2}, is obtained from the corresponding base frame via forward kinematics.

The object coordinate system Σ_{obj} represents the pose of Fabric A and is attached to the midpoint between Σ_{e1} and Σ_{e2}. Its z-axis is aligned with that of Σ_o, and its x-axis points from Σ_{e1} to Σ_{e2}. The internal and external force coordinate systems, Σ_{int} and Σ_{ext}, used for impedance control, are identical to Σ_{obj}.

IV. 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.

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

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

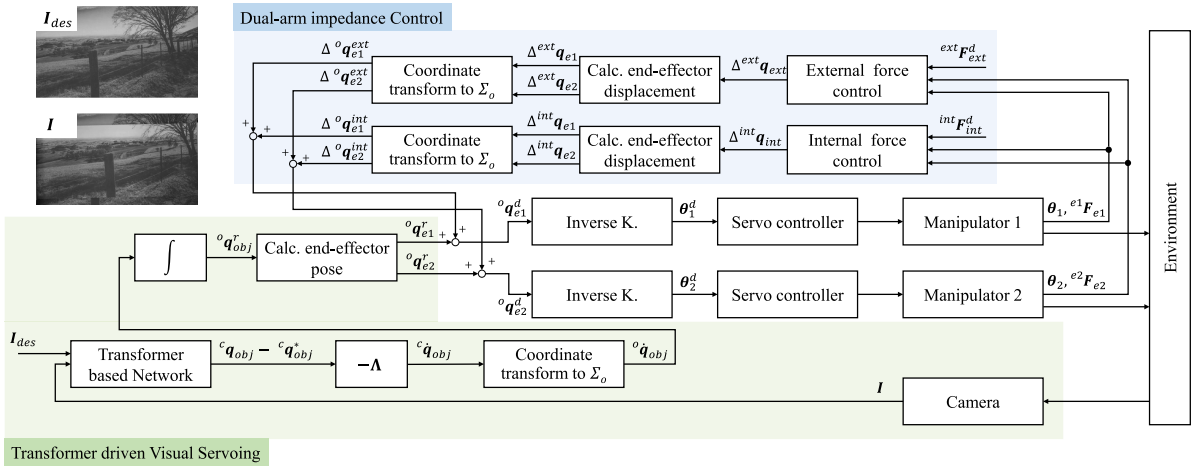


Fig. 3. Block diagram of the control system. The proposed control system consists of Transformer-driven visual servoing (in green) and dual-arm impedance control (in blue).

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 $\boldsymbol{\Lambda} \in \mathbb{R}^{6\times6}$ 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 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: Conventional CNN-based visual servoing methods typically concatenate or subtract features from Siamese backbones before feeding them into fully connected layers [5], [8]. This simple concatenation or subtraction-based feature fusion has been widely used for learning pose differences between image pairs due to its simplicity. In contrast, our network adopts a pre-trained Transformer backbone with Difference Extraction Attention Module (DEAM) to handle the fabric alignment task, as shown in Fig. 4. Because alignment depends on subtle, spatially distributed texture cues rather than distinct geometric landmarks, global feature modeling is essential. The Transformer captures long-range dependencies, while DEAM extracts inter-image feature differences at multiple stages to enhance sensitivity to small texture shifts.

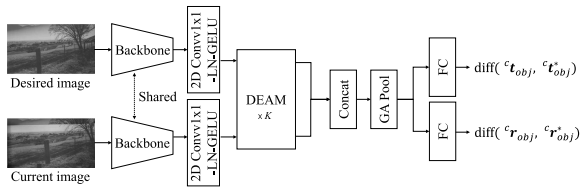


Fig. 4. Overall architecture of the Transformer-driven visual servoing network.

Unlike CNN-based Siamese models that compute differences through simple concatenation or subtraction, our design embeds this comparison process directly into the attention mechanism, enabling adaptive learning of cross-image differences. Compared with attention mechanisms such as SE [15], ECA [16], and GRN [17], DEAM explicitly models cross-image feature interactions rather than applying attention independently within each map.

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 (kernel size of one), 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

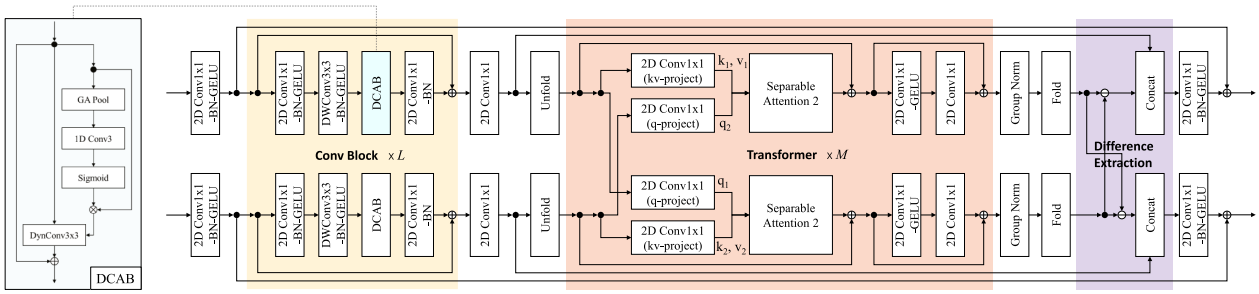


Fig. 5. Architecture of the Difference Extraction Attention Module (DEAM), which combines convolutional blocks, Transformer blocks, and a difference extraction blocks to enhance prediction accuracy. The convolutional block includes the proposed Dynamic Convolution by Attention Blocks (DCAB).

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. 5, 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 [18]. 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. 5.

DCAB is built upon a dynamic convolutional layer [19] with a 3×3 kernel (DynConv3×3), whose convolution kernels are dynamically generated from input-dependent attention scores computed using Efficient Channel Attention (ECA) [16]. 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 [20], and then passed into Transformer Blocks.

The Transformer Block is a cross-attention mechanism [21] that applies a separable attention mechanism [22]. 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 features are then folded back to their original spatial dimensions. The Transformer Block is repeated M layers. In our design, the cross-attention in the desired image branch uses query features from the current image and key/value features from the desired image. The query features indicate the regions to be aligned, while the key/value features provide the reference appearance and context. This allows the network to find correspondences between the two feature spaces. Conversely, the cross-attention in the current image branch learns complementary relationships in the opposite direction.

Fig. 6. Examples of synthetic image pairs used in training. Top row: desired images. Bottom row: corresponding current images.

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. 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. 6, where the first row contains the desired images and the second row shows the current images.

B. Dual-Arm Impedance Control

To flatten the fabric piece during the visual servoing, dual-arm impedance control is utilized based on coordinated motion control proposed by Kosuge et al. [23], [24] 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

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

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 [24].

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

$$\begin{aligned}\boldsymbol{M}_{ext}\Delta^{ext}\ddot{\boldsymbol{q}}_{ext} + \boldsymbol{D}_{ext}\Delta^{ext}\dot{\boldsymbol{q}}_{ext} + \boldsymbol{K}_{ext}\Delta^{ext}\boldsymbol{q}_{ext} \\ = \boldsymbol{S}_{ext}({}^{ext}\boldsymbol{F}_{ext} - {}^{ext}\boldsymbol{F}^{d}_{ext}),\end{aligned} \tag{5}$$

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 ${}^{int}\boldsymbol{F}_{int} = [{}^{int}\boldsymbol{f}^{T}_{int}, {}^{int}\boldsymbol{\eta}^{T}_{int}]^{T} \in \mathbb{R}^6$ 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 [24].

The internal force is controlled based on the model as

$$\begin{aligned}\boldsymbol{M}_{int}\Delta^{int}\ddot{\boldsymbol{q}}_{int} + \boldsymbol{D}_{int}\Delta^{int}\dot{\boldsymbol{q}}_{int} \\ = \boldsymbol{S}_{int}({}^{int}\boldsymbol{F}_{int} - {}^{int}\boldsymbol{F}^{d}_{int}),\end{aligned} \tag{6}$$

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

$${}^{o}\boldsymbol{q}^{d}_{e1} = {}^{o}\boldsymbol{q}^{r}_{e1} + \Delta^{o}\boldsymbol{q}^{ext}_{e1} + \Delta^{o}\boldsymbol{q}^{int}_{e1}, \tag{7}$$

$${}^{o}\boldsymbol{q}^{d}_{e2} = {}^{o}\boldsymbol{q}^{r}_{e2} + \Delta^{o}\boldsymbol{q}^{ext}_{e2} + \Delta^{o}\boldsymbol{q}^{int}_{e2}. \tag{8}$$

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.

V. Experimental Results

A. Network Comparison

For the backbone network, we adopt the Vision Transformer (ViT) [25] 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 [26] optimizer, with a batch size of 64 (8 samples per GPU $\times$ 8 GPUs).

We use a total of 100,000 samples for training (with 10% held out for validation) and 58,000 samples for testing. 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 [27]. The loss function is defined as

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

where $({}^{c}\widehat{\boldsymbol{t}_{obj} - {}^{c}\boldsymbol{t}^{*}_{obj}})$ and $({}^{c}\widehat{\boldsymbol{r}_{obj} - {}^{c}\boldsymbol{r}^{*}_{obj}})$ 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 $\pm 10^{\circ}$ around each axis.

Table I shows a comparison between the proposed network and several other architectures. The Concat [5] method employs a Siamese architecture with two shared-weight ViT [25] backbones whose extracted features are concatenated and fed into fully connected layers. The proposed method, DEAM (DCAB), achieves the highest accuracy, significantly reducing the loss E (defined in (9)) compared to Concat. A Wilcoxon signed-rank test on the paired errors from the same 58,000 test samples confirmed a significant difference between the two models ($W = 3.04 \times 10^8, p < 0.001, r = 0.55$), indicating that DEAM (DCAB) outperformed Concat with a large effect size. 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 [15]), DEAM (ECA [16]), and DEAM (GRN [17]). 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 [15], ECA [16], and GRN [17] independently to each of the two feature maps extracted from the shared backbone, before concatenation and fully connected layers. While these attention

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT NETWORK ARCHITECTURES

Method	Loss E $[\times 10^{-3}]\downarrow$	Trans / Rot RMSE ↓	Std $[\times 10^{-3}]\downarrow$	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 [15]	7.968	6.658 / 1.843	2.382	78.273	0.0158
ECA [16]	8.006	6.544 / 1.880	2.407	78.273	0.0158
GRN [17]	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), evaluated on 58,000 test samples. 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.

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 $0.0249\,\mathrm{mm/pixel}$. FabricA is a rectangular piece measuring 100×400 mm, and FabricB measures 297×420 mm. Each end-effector is a custom-built "Roll-Up" [1], which rolls up the fabric edge around a cylindrical surface to securely grasp it without slippage. The end-effectors were designed and fabricated in-house based on our previous work.

Five different textures (Texture 1 to 5), which are not included in the training dataset, are used in the experiments. To ensure the alignment 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 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. 7 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

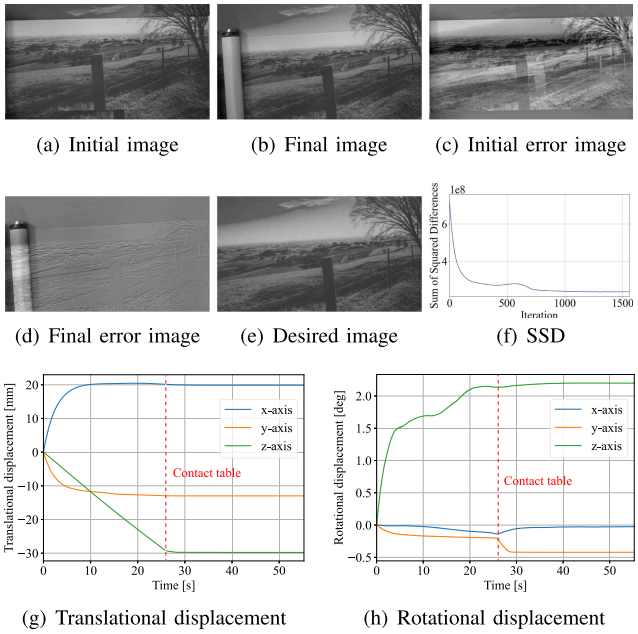
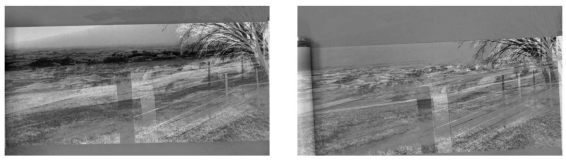


Fig. 7. Visual servoing result for Texture 1 using the proposed scheme. (a) Initial image. (b) Final image after alignment. (c,d) Error image ($\boldsymbol{I} - \boldsymbol{I}_{des}$) before and after visual servoing. (e) Desired image. (f) Sum of squared differences (SSD) between $\boldsymbol{I}$ and $\boldsymbol{I}_{des}$ over time. (g,h) Translational and rotational displacement of the object coordinate system from the initial pose.

reduced compared to the initial state, as shown in Fig. 7(c) and (d). The sum of squared differences (SSD) between the desired and current images decreases through visual servoing as shown in Fig. 7(f). Despite occlusion caused by the end-effectors, the proposed method maintains accurate alignment, demonstrating robustness to occlusion. As shown in Fig. 7(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

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



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

Fig. 8. Result of direct visual servoing for Texture 1. (a, b) Error images before and after visual servoing.

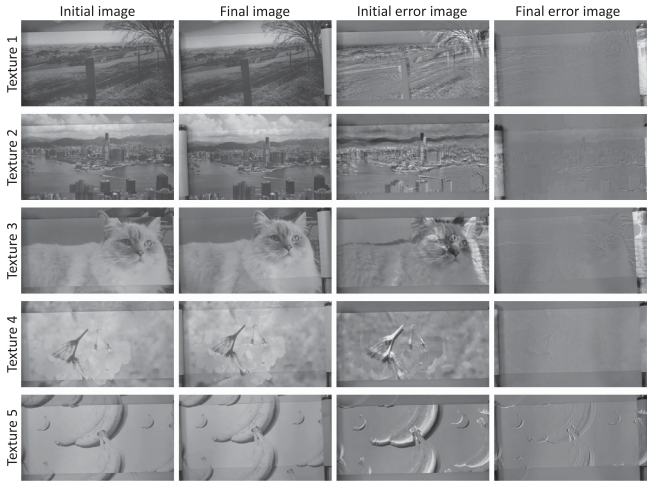


Fig. 9. Qualitative results of the proposed visual servoing method across all five textures (Texture 1 to 5).

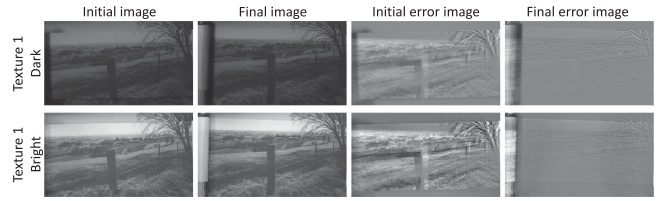


Fig. 10. Results of visual servoing for Texture 1 under varying lighting conditions (dark and bright).

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. 8 presents the result of direct visual servoing [28] 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.

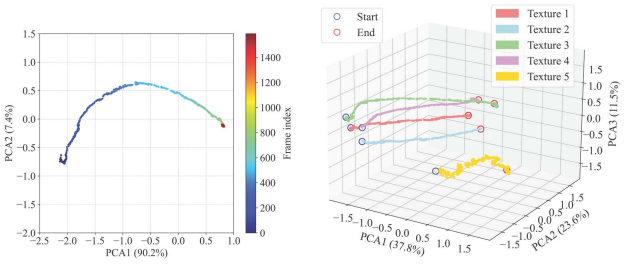
Fig. 9 summarizes the qualitative results of the proposed visual servoing using Texture 1 to 5. The proposed control scheme successfully aligns Fabric A with Fabric B across all textures, demonstrating strong generalization to previously unseen textures. Each texture presents unique challenges. Texture 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, 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.

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. 10. 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.

3) Visual Servoing of Repetitive Texture With Wrinkles: To further evaluate the robustness of the proposed method, we conducted an additional experiment using a fabric that simultaneously exhibits a repetitive texture pattern and wrinkles (i.e., crumpled-like deformations) under a sagging configuration. In this experiment, the network was trained with RGB inputs to verify that the proposed framework can also operate with color information. Note that the RGB-trained network achieved approximately 14% lower test loss compared to the grayscale version. As shown in the supplementary video, the proposed method achieved stable convergence while aligning the repetitive texture, even in the presence of deformations and wrinkles. These results further show that the proposed approach generalizes well to realistic garment manipulation scenarios involving non-flat surfaces and repetitive textures.

4) 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. 11(a) shows the PCA results of features extracted during visual servoing with Texture 1, based on the experiments in Fig. 7. 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.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



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

Fig. 11. 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.

Fig. 11(b) shows the PCA results of features extracted during visual servoing with all five textures, based on the experiments in Fig. 9. 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 low-saliency and low-contrast textures remain more challenging, performance may be further improved by incorporating additional visual cues, such as structured light.

VI. 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 handling," *IEEE Robot. Autom. Lett.*, vol. 10, no. 11, pp. 11204–11211, Nov. 2025.

[2] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 935–941.

[6] F. Tokuda, S. Arai, and K. Kosuge, "Object positioning by visual servoing based on deep learning," *Trans. Soc. Instrum. Control Eng.*, vol. 55, pp. 717–725, Jan. 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 *Proc. IEEE Int. Conf. Robot. Autom.*, 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. 91820–91835, 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 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 Trans. Robot.*, vol. 40, pp. 364–381, 2024.

[11] J. Qi et al., "Contour moments based manipulation of composite rigid-deformable objects with finite time model estimation and shape/position control," *IEEE/ASME Trans. Mechatron.*, vol. 27, no. 5, pp. 2985–2996, Oct. 2022.

[12] C. Chi et al., "Diffusion policy: Visuomotor policy learning via action diffusion," *Int. J. Robot. Res.*, vol. 44, no. 10–11, pp. 1684–1704, 2025.

[13] W. Chen and N. Rojas, "TrakDis: A transformer-based knowledge distillation approach for visual reinforcement learning with application to cloth manipulation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2455–2462, Mar. 2024.

[14] Y. Wang et al., "Efficient planar fabric repositioning: Deformation-aware RRT* for non-prehensile fabric manipulation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 12, pp. 11258–11265, Dec. 2024.

[15] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[16] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient channel attention for deep convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11531–11539.

[17] S. Woo et al., "Convnext v2: Co-designing and scaling convnets with masked autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16133–16142.

[18] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 6105–6114.

[19] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic convolution: Attention over convolution kernels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11027–11036.

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

[21] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 5998–6008.

[22] S. Mehta and M. Rastegari, "Separable self-attention for mobile vision transformers," 2022, *arXiv:2206.02680*.

[23] K. Kosuge, S. Hashimoto, and K. Takeo, "Coordinated motion control of multiple robots manipulating a large object," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1997, vol. 1, pp. 208–213.

[24] K. Kosuge, K. Kamei, and T. Nammoto, "Coordinated motion control of dual manipulators for handling a rigid object with non-negligible deformation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 5145–5151.

[25] A. Dosovitskiy et al., "An image is worth 16 × 16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

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

[27] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM J. Control Optim.*, vol. 30, no. 4, pp. 838–855, 1992.

[28] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 828–834, Aug. 2011.