ViComp: Video Compensation for Projector-Camera Systems

Yuxi Wang, Haibin Ling, and Bingyao Huang

Abstract— Projector video compensation aims to cancel the geometric and photometric distortions caused by non-ideal projection surfaces and environments when projecting videos. Most existing projector compensation methods start by projecting and capturing a set of sampling images, followed by an offline compensation model training step. Thus, abundant user effort is required before the users can watch the video. Moreover, the sampling images have little prior knowledge of the video content and may lead to suboptimal results. To address these issues, this paper builds a video compensation system that can online adapt the compensation parameters. Our approach consists of five threads and can perform compensation, projection, capturing, and short-term and long-term model updates in parallel. Due to the parallel mechanism, rather than projecting and capturing hundreds of sampling images and training the model offline, we can directly use the projected and captured video frames for model updates on the fly. To quickly apply to the new environment, we introduce a deep learning-based compensation model that integrates a fixed transformer-based method and a novel CNN-based network. Moreover, for fast convergence and to reduce error accumulation during fine-tuning, we present a strategy that cooperates with short-term and long-term memory model updates. Experiments show that it significantly outperforms state-of-the-art baselines.

Index Terms—Projector compensation, Video compensation, Spatial augmented reality, Projector-camera system, Continuous projection mapping

1 Introduction

Projectors are widely used in many applications, such as interactive entertainment [2, 3, 27, 39], immersive display [32, 33, 41, 57, 60], and spatial augmented reality [16, 34, 35, 53]. For immersive and accurate visual experiences in a complex environment, projector compensation is usually applied to correct geometric and photometric distortions due to the environment or projection surfaces. Early approaches project specifically designed sampling images onto the projection surface, and independently estimate geometric and photometric distortions using pixel correspondences [12, 36, 47, 50]. Recently, deep learning-based methods have been proposed to jointly estimate geometric and photometric distortions with a unified network [21, 31, 56]. These methods show promising performance when the sampling image collection and model training can be performed offline with some user effort.

Despite the progress achieved so far, current methods have three major challenges for video compensation: (1) Existing methods mostly require capturing many sampling images, and learning geometric and photometric transformations offline. This process remains complex and time-consuming for non-expert users. Moreover, users must redo this process when the environment changes to some degree. (2) The sampling images usually consist of handcrafted patterns or natural images and have different data distributions compared to the actual video frames users play. Using these samples to learn compensation parameters may result in suboptimal performance. (3) Unlike single images, videos, especially in lengthy formats like movies, present a dynamic scenario where content evolves continuously within each cut. There are noticeable variations in content between consecutive cuts and the repetition of similar content across multiple cuts. However, existing compensation methods treat videos as independent frames without considering the characteristics of cuts and frames.

To address these video compensation challenges, we present a multi-thread system called **ViComp** (**Vi**deo **Comp**ensation), which comprises five threads and concurrently achieves online video compensation, projection, data collection, and model update. An example of this system's application is shown in Fig. 1. We first initialize the base compensation model using pre-trained parameters. Subsequently, we compensate video frames using this base model and simultaneously perform online model updates using synchronized frame pairs.

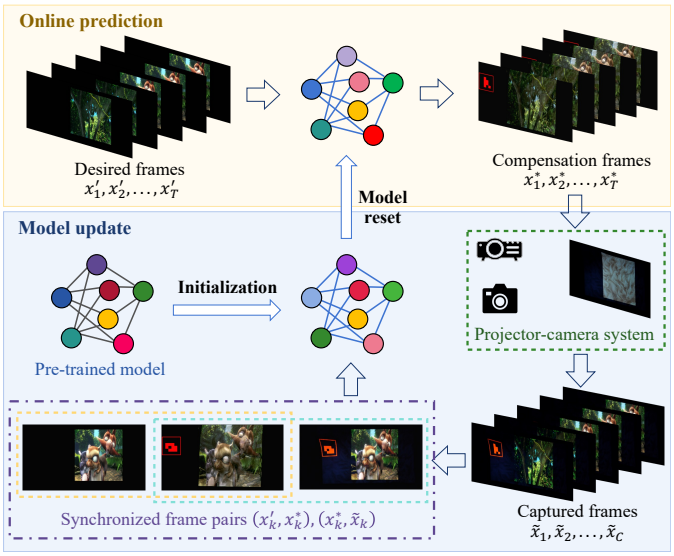


Fig. 1: We consider video compensation and learn/update the compensation model on the fly. Our system receives desired frames from a video source that maintains continuity within a single cut but exhibits significant differences across different cuts. To start, we use a base pre-trained image compensation model to initialize the network. Afterward, compensation frames are generated online and then projected onto a textured non-planar surface via a projector-camera system. Meanwhile, we adjust model parameters to the target environment using synchronized frame pairs, which are formed by sampling from the desired frames, the corresponding compensation frames, and the captured frames.

In ViComp, multiple threads are coordinated: the compensation thread generates compensation frames and sends them to the projection thread; the projection thread then projects the received frames onto the surface to achieve high-quality visual effects; the capture thread captures frames using a static camera at scheduled intervals; meanwhile,

- *Yuxi Wang is with Hangzhou Dianzi University. E-mail: yxwang@hdu.edu.cn*
- *Haibin Ling is with Stony Brook University. E-mail: haibin.ling@stonybrook.edu.*
- *Bingyao Huang is with Southwest University. E-mail: bhuang@swu.edu.cn. Corresponding author.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

two additional threads collaborate to update the model parameters using the desired frame, the projector input and the frame captured from the observer's viewpoint. Thanks to this collaboration, our system can achieve video compensation and model updates in parallel.

Specifically, to rapidly generate high-quality compensation frames, we employ an efficient network that consists of two subnets as the base compensation model. The first subnet is a transformer-based network that estimates the displacement field using a pair of images. Note that we use the pre-trained model for this subnet without offline or online fine-tuning. The second subnet is a CNN-based network that models the photometric transformation function and generates high-resolution compensation images from the low-resolution frontal view images with color distortion. This network is efficient and can quickly converge to the optimal solution even with minimal samples in new settings.

However, naive online model update strategies use every frame to update the compensation model and may lead to overfitting and error accumulation. To address these issues, we propose a novel model update strategy that can leverage the knowledge of both the environment and the video content. In particular, by considering the characteristics that videos have a strong correlation between adjacent frames within a single cut and difference across cuts, we employ two memory threads: a short-term memory thread to update the model using every other frame within a cut; and a long-term memory thread to update the model using the first synchronized frame pair of each cut.

Our contributions are summarized as follows:

- We build an online video compensation system that compensates frames and adjusts model parameters in parallel.
- The system employs an efficient base model that can generate high-quality compensation frames and be optimized using a few samples in new environments.
- we propose a model update strategy combining long-term and short-term memory mechanisms that enable the compensation model to adapt to the target configuration and video content rapidly while reducing error accumulation.
- Experiments demonstrate the effectiveness of our system on video compensation. It outperforms the state-of-the-art baselines.

Our source code is available at https://github.com/cyxwang/ViComp.

2 Related Work

2.1 Projector compensation

Projector compensation involves two key tasks: geometric correction which calculates the corresponding positions of pixels between the projected input and the captured frame, and photometric compensation which estimates the reflectance properties and the color transformation of each pixel. Detailed reviews can be found in [4, 15]. Traditionally, to correct geometry, the geometric mapping function can be estimated by obtaining pixel correspondences between the projector input and the frames captured by a camera placed at the viewpoint. Most existing methods project well-defined patterns or markers onto the surface, such as structured light [14, 50]. Due to their high precision, these methods have been widely used in many applications. Subsequently, some works are devoted to simplifying this process by designing specific patterns. Narita *et al*. [36] designed fiducial markers consisting of four types of dot clusters, which can be detected in real-time and applied to dynamic projection mapping. Then, Watanabe *et al*. [59] extended this work to design the marker automatically. To make the marker imperceptibility, Kagami *et al*. [25] inserted the 3×3 chessboard with a larger margin into every sixth binary frame using a white light source. Li *et al*. [30] obtained sparse correspondences by extracting and matching natural features. Furthermore, some methods incorporate additional hardware, such as infrared (IR) camera [17], depth camera [23, 48] for tracking surfaces without visible patterns. They can avoid interference from markers in dynamic projection mapping.

Photometric compensation generates a projector input that compensates for the distortion of color and texture of the projection surface and the photometric environment. Prior algorithms estimate the color transfer function by projecting additional patterns. Akiyama *et al*. [1] applied a perceptually-based optimization to generate compensation images. Moreover, for dynamic mapping, Fujii *et al*. [12] adapted the model according to the difference between the captured image and the desired image. Bokaris *et al*. [5, 6] generated images using a linear transformation matrix for dynamic surfaces with one-frame delay. Kurth [29] corrected image color continuously by solving the projector's response curve, the environment lighting in the scene, and the target object's surface color. Hashimoto *et al*. [18] utilized the offset of the adjacent moment to update the inter-pixel correspondence and optimized the current reflectance using the present correspondence and the sum of past correspondence. Considering the effect of inter-pixel coupling, Shih *et al*. [47] estimated the dynamic reflectance using the projected images as calibration patterns. As the successful application of deep learning in computer vision, Huang *et al*. [19] proposed an end-to-end trainable method that learns the photometric transformation function from the projected and captured image pairs. Kageyama *et al*. [26] presented an online deblurring method for video display. This method extracts information from the current frame and the previous as well as its projecting results using two U-Net-like subnets and then generates a compensation image by incorporating this information.

Additionally, some methods estimate geometric and photometric distortions jointly using well-designed patterns [40, 45]. Recently, the latest works trended to design deep neural networks to model both geometric and photometric transformation functions within a unified framework. Huang *et al*. [20, 21] proposed an end-to-end method that consists of a coarse-to-fine process for calculating warping field and a U-Net [42] like network for modeling the photometric transformation function. Then, Wang *et al*. [56] improved its efficiency for high-resolution applications by introducing the pixel shuffle operation. Li *et al*. [31] proposed a physics-based framework that compensates for geometric distortion and color deviation by considering the response function, geometric displacement, and the reflectivity of textured surfaces using natural images. Erel *et al*. [11] presented a solution for geometric and color compensation by training a neural reflectance field using black, white, and lollipop-illuminated frames. These trainable methods can generate high-quality compensation images with a small amount of training data. However, they require offline sampling image collection and model training and thus are less user-friendly to video compensation. Moreover, due to the lack of prior knowledge of video content, the model optimized using offline samples cannot achieve the best compensation effect for target videos. Thus, online methods, e.g., test-time adaptation ones are preferable for video compensation.

2.2 Test-time adaptation methods

For models that are pre-trained in the source domain, test-time adaptation improves their performance on the target domain in the absence of source data and the label of test input during inference. Most of the existing methods [24, 51, 54] leverage pseudo-labeling or entropy minimization techniques to address this problem. Wang *et al*. [54] adapted the pre-trained model and update the parameters in BatchNorm layers by minimizing the entropy. Iwasawa and Matsuo [24] focused on adjusting only the linear layer at test time to reduce their computation. Nguyen *et al*. [37] employed an invariance regularization technique and updated affine parameters using an unsupervised surrogate loss function. Instead of revising the parameters directly, some approaches [52, 55] use teacher-student strategies to average model weights. These methods exhibit significant improvements when the test data originates from the same domain. However, as the environment continually changes in the real world, models tend to forget the knowledge acquired during pre-training, and the noise can lead to error accumulation in lifelong adaptation. To prevent forgetting, Wang *et al*. [55] restored a small subset of model parameters from the source model randomly. Brahma and Rai [7] took into account the model uncertainty and restored only irrelevant parameters based on the Fisher Information Matrix. Döbler *et al*. [10] introduced a mean teacher with symmetric cross-entropy loss.

Different from the applications of test-time adaptation methods, the training samples with ground truth can be obtained with a one-

frame delay in the projector-camera system. Thus, in our task, fine-tuning using these samples can improve the compensation performance. However, due to the similarity of video content, the continuous updating process may lead to overfitting and error accumulation. This is a key concern that needs to be addressed in our online update method.

2.3 Our method

Unlike existing image compensation methods [1, 11, 12, 14, 25], we develop ViComp, an online video compensation system using deep neural networks under the assumption that users play videos in new environments where lighting, projection surface, and reflections are unknown and remain constant.

Drawing inspiration from the deep learning-based compensation methods [21, 31], we introduce two subnets to model both the geometric and photometric transformation functions. Unlike these methods requiring training samples from the target environment, our base model performs well using pre-trained parameters, so it can be easily configured to unknown environments.

To enhance display quality, motivated by the adaptive compensation method [12], we update our compensation model during video playback without manual intervention. But different from [12], our model update process not only considers the difference between captured frames and desired effects but also uses synchronized captured and projected frame pairs as training data. Notably, for the specific video playback task, our method employs an online update strategy that combines long-term and short-term memory models based on video characteristics, where the contents of long videos exhibit strong consistency and distinctiveness. Moreover, to prevent overfitting and reduce accumulated errors during model update, inspired by the novel test-time adaptation method [55], we introduce an averaging strategy to the long-term model update process. This online update strategy allows our base model to quickly adapt to the target configuration and video contents.

Furthermore, considering the requirement for real-time projection in the video compensation task, our system integrates projection, capturing, video compensation, and model update into a parallel framework to improve its efficiency.

In summary, to the best of our knowledge, our ViComp is the first online video compensation system that compensates frames and adapts model parameters in parallel. It is user-friendly and can be rapidly configured to unknown environments with good performance.

3 Projector Compensation Problem

In this section, we first provide a brief description of the compensation problem and then introduce the architecture of our base video compensation model.

3.1 Problem formulation

Our compensation model is designed to compensate for geometric and photometric disturbances online. Let the projector input be $\boldsymbol{x}$, the global lighting be $\boldsymbol{g}$, and the surface reflection parameters be $\boldsymbol{s}$. Denote $\mathcal{G}$ as the function that maps the projector input to the camera view, and $\mathcal{P}$ be the function for mapping the input frame pixel color to the uncompensated camera-captured ones. Then the entire projection and capturing process can be written as:

$$\tilde{\boldsymbol{x}} = \mathcal{G}(\mathcal{P}(\boldsymbol{x}; \boldsymbol{g}, \boldsymbol{s})) \tag{1}$$

where $\tilde{\boldsymbol{x}}$ is the captured frame. The goal of projector compensation is to find the projector input image $\boldsymbol{x}^*$ by modeling the pseudo-inverse of $\mathcal{P}$ and the inverse of $\mathcal{G}$, such that when projected, the camera-captured frame matches the ideal viewer-perceived frame $\boldsymbol{x'}$:

$$\boldsymbol{x}^* = \mathcal{P}^\dagger(\mathcal{G}^{-1}(\boldsymbol{x'}; \boldsymbol{g}, \boldsymbol{s})) \tag{2}$$

Inspired by [21], $\boldsymbol{s}$ and $\boldsymbol{g}$ can be implicitly captured by the camera-captured surface image $\tilde{\boldsymbol{s}}_0$:

$$\tilde{\boldsymbol{s}}_0 = \mathcal{G}(\mathcal{P}(\boldsymbol{x}_0; \boldsymbol{g}, \boldsymbol{s})) \tag{3}$$

where $\boldsymbol{x}_0$ is an input image with plain gray illumination. Thus, Equ. (2) can be rewritten as:

$$\boldsymbol{x}^* = \mathcal{P}^\dagger(\mathcal{G}^{-1}(\boldsymbol{x'}), \mathcal{G}^{-1}(\tilde{\boldsymbol{s}}_0)) \tag{4}$$

Deep neural networks are employed in [21] to model $\mathcal{G}^{-1}$ and $\mathcal{P}^\dagger$ in Equ. (4). The parameters are trainable through the use of frame pairs $(\boldsymbol{x'}, \boldsymbol{x}^*)$. However, a significant challenge arises because $\mathcal{G}^{-1}$ in [21] cannot be estimated through pre-training. Consequently, additional training samples are required when applying it with different settings. In this paper, inspired by [31], considering that the geometric correction process can be viewed as the estimation of the displacement field, we reformulate Equ. (4) as:

$$\boldsymbol{x}^* = \mathcal{P}^\dagger(\mathcal{W}(\boldsymbol{x'}, \boldsymbol{u}), \mathcal{W}(\tilde{\boldsymbol{s}}_0, \boldsymbol{u})) \tag{5}$$

where $\boldsymbol{u}$ represents the calculated displacement field from the camera view to the projector input, while $\mathcal{W}$ denotes the warping operation that transforms the captured frame into the projector's canonical frontal view.

According to Equ. (5), we transform the problem of fine-tuning the parameters in both geometric transformation function $\mathcal{G}^{-1}$ and the photometric transformation function $\mathcal{P}^\dagger$ using pre-collected training data into the task of estimating the displacement field $\boldsymbol{u}$ during initialization and gradually updating the parameters of photometric transformation function $\mathcal{P}^\dagger$ during video playback.

3.2 Base compensation network

Our base compensation network consists of two subnets that estimate the displacement field $\boldsymbol{u}$ and model the photometric transformation function $\mathcal{P}^\dagger$ respectively. The architecture is shown in Fig. 2. To improve the display quality, we input high-resolution frames into the base model and subsequently generate high-resolution compensation frames for projection.

3.2.1 Displacement field estimation

Denote the displacement field estimation function as $\mathcal{F}$, then the displacement field $\boldsymbol{u}$ in Equ. (5) can be estimated by:

$$\boldsymbol{u} = \mathcal{F}(\tilde{\boldsymbol{x}}, \boldsymbol{x}) \tag{6}$$

Inspired by [31], we employ FlowFormer [22], a novel transformer-based network to model $\mathcal{F}$. In this network, a Siamese encoder is used to extract features from both the source and target images, and an encoder and a recurrent decoder are leveraged to encode the 4D cost volume to a cost memory and estimate the displacement of each pixel from the cost memory. The detailed architecture can be found in [22]. Particularly, in our system, we use the parameters pre-trained on the Sintel dataset [8] provided by [22] directly. Benefiting from it, the displacement field $\boldsymbol{u}$ can be pre-calculated using a pair of texture images during initialization.

Furthermore, as shown in Fig. 2, to estimate the displacement field more accurately, we preprocess the captured image before estimating $\boldsymbol{u}$. During initialization, we first project a gray image $\boldsymbol{s_0}$ and capture the surface image $\tilde{\boldsymbol{s}}_\boldsymbol{0}$. Then we find the field of view (FOV) from $\tilde{\boldsymbol{s}}_\boldsymbol{0}$ using the Ostu's method [38] and extract the region of interest (ROI) from $\tilde{\boldsymbol{s}}_\boldsymbol{0}$. After that, we project an additional texture image $\boldsymbol{x_a}$ onto the surface and obtain the camera-captured image $\tilde{\boldsymbol{x}}_\boldsymbol{a}$, then extract its ROI and rescale the cropped patch to match the resolution of $\boldsymbol{x_a}$. Finally, we estimate the displacement field $\boldsymbol{u}$ between the cropped patch of $\tilde{\boldsymbol{x}}_\boldsymbol{a}$ and $\boldsymbol{x_a}$ using $\mathcal{F}$.

3.2.2 Photometric compensation

To adapt parameters of the base compensation model to unknown environments, we utilize the photometric compensation subnet introduced in CompenNeSt++ [21] to model the photometric transformation function $\mathcal{P}^\dagger$. This network consists of a Siamese encoder and a decoder with six convolutional layers, two transposed convolutional layers, and five skip convolutional layers. Additional details can be found in [21].

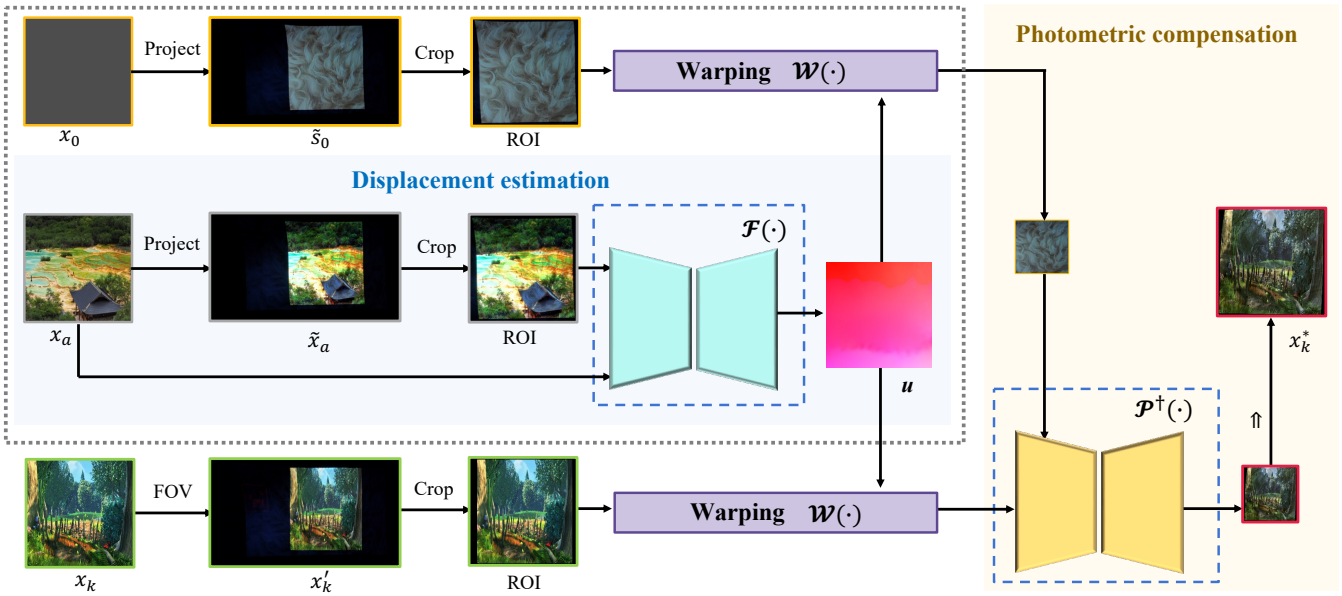


Fig. 2: **The base model of video compensation**. During initialization (in the gray dash bounding box), we project a gray image $\boldsymbol{x_0}$ and a texture image $\boldsymbol{x_a}$ onto the surface and capture the images $\tilde{\boldsymbol{s}}_\mathbf{0}$ and $\tilde{\boldsymbol{x}}_\boldsymbol{a}$. We extract the region of interest (ROI) from captured images and estimate the displacement field $\boldsymbol{u}$ from the cropped ROI of $\tilde{\boldsymbol{x}}_\boldsymbol{a}$ to $\boldsymbol{x_a}$ using a displacement field estimation function $\mathcal{F}$. Then, we get the warped surface $\tilde{\boldsymbol{s}}_\boldsymbol{w}$ from $\tilde{\boldsymbol{s}}_\mathbf{0}$ by warping operation. During online testing, according to the field of view (FOV), the video frame $\boldsymbol{x}_k$ is converted to the desired frame $\boldsymbol{x}'_k$, and then its ROI is sent to the base compensation model, which consists of a warping operation and a photometric transformation function $\mathcal{P}^{\dagger}$. After that, the high-resolution compensation frame $\boldsymbol{x}^*_k$ is generated.

This architecture demonstrates good performance on photometric compensation and can be efficiently trained with limited training samples. However, it is essential to note that the computation of this network increases significantly with the resolution of the training samples. To address this issue without compromising the performance of compensation, we use it to obtain low-resolution compensation frames, and then acquire the high-resolution projector input via an upsampling operation. Thus, Equ. (5) can be reformulated as:

$$\boldsymbol{x}^* = \Uparrow (\mathcal{P}^{\dagger}(\mathcal{W}(\boldsymbol{x'}, \boldsymbol{u}), \mathcal{W}(\tilde{\boldsymbol{s}}_\mathbf{0}, \boldsymbol{u}))) \tag{7}$$

where $\Uparrow(\cdot)$ denotes the upsampling operation, which is realized by the bicubic interpolation algorithm.

4 Parallel Video Compensation System

In this section, we will introduce the proposed ViComp system and its key components.

4.1 Multithread system

The proposed ViComp system consists of five threads: the primary thread $\mathcal{T}_p$ for projection, and four auxiliary threads $\mathcal{T}_c$, $\mathcal{T}_m$, $\mathcal{T}_s$, $\mathcal{T}_l$ for capturing, compensation, short-term memory model update, and long-term memory model update, respectively. Their collaboration is described in Fig. 3.

During initialization, we first project a gray image $\boldsymbol{x_0}$ and a texture image $\boldsymbol{x_a}$ with the resolution of 1024×1024 onto the projection surface, and capture the corresponding images $\tilde{\boldsymbol{s}}_\mathbf{0}$ and $\tilde{\boldsymbol{x}}_\boldsymbol{a}$ from the observer's viewpoint. Afterward, we estimate the displacement field $\boldsymbol{u}$ from $\tilde{\boldsymbol{x}}_\boldsymbol{a}$ to $\boldsymbol{x_a}$ according to Section 3.2.1, and generate the warped surface image $\tilde{\boldsymbol{s}}_\boldsymbol{w}$ from $\tilde{\boldsymbol{s}}_\mathbf{0}$ using $\boldsymbol{u}$ with bilinear interpolation. Then we initialize parameters of the short-term memory model f_{θ_0} and the long-term memory model g_{θ_0} using the pre-trained model provided by [21]. After that, all auxiliary threads start and wait for data exchange.

For online compensation and model updates, as summarized in Algorithm 1, the compensation thread $\mathcal{T}_m$ reads a video frame $\boldsymbol{x}_k$ from memory and converts it to the desired frame $\boldsymbol{x}'_k$, then generates the compensation frame $\boldsymbol{x}^*_k$ using the short-term memory model $f_{\theta_{n-1}}$. The details of the online compensation process are described in Fig. 2. Additionally, due to the challenge of achieving hardware synchronization in projector-camera systems, we synthesize the projector input using $\boldsymbol{x}^*_k$ and print an ArUCo marker [13] next to $\boldsymbol{x}^*_k$ at every other frame to implement a soft synchronization. After receiving the synthetic frame, the projection thread $\mathcal{T}_p$ projects it onto the surface. Meanwhile, the capturing thread $\mathcal{T}_c$ collects images every 0.01s and recognizes the marker in the captured frame $\tilde{\boldsymbol{x}}_k$. Subsequently, the synchronized frame pairs $(\boldsymbol{x}'_k, \boldsymbol{x}^*_k)$ and $(\boldsymbol{x}^*_k, \tilde{\boldsymbol{x}}_k)$ are sent to model updating threads. By learning knowledge from these synchronized frame pairs, the short-term memory model update thread $\mathcal{T}_s$ and long-term memory model update thread $\mathcal{T}_l$ cooperate to adapt the base model to the target configuration. Thanks to the parallel processing of these multiple threads, our system can perform online compensation, projection, and model updates automatically. Therefore, it can be easily and quickly applied to different configurations with high display quality.

4.2 Online model updates

In ViComp, we present an online model update strategy to adapt the base compensation model to the target environment during video playback.

As shown in Fig. 3, the video used for projection generally consists of multiple cuts, with the content between each cut having continuity, while the content across adjacent cuts typically varies drastically. Based on this observation, we introduce a short-term memory model to learn the local information (e.g., video cut content) within the single cut and a long-term memory model to learn the global information, e.g., environmental knowledge of the scene and the entire video (e.g., color tunes, styles).

In the short-term memory model update thread $\mathcal{T}_s$, we fine-tune the model f_{θ_n} using each synchronized frame pairs $(\tilde{\boldsymbol{x}}_k, \boldsymbol{x}^*_k)$ and $(\boldsymbol{x}^*_k, \boldsymbol{x}'_k)$ within a cut according to the loss function described in Equ. (8):

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{\text{cmp}} + \mathcal{L}_{\text{vis}} \\ &= \ell_{\text{mix}}\left(\mathcal{P}^{\dagger}(\mathcal{W}(\boldsymbol{x}'_k, \boldsymbol{u}), \tilde{\boldsymbol{s}}_\boldsymbol{w}), \mathcal{P}^{\dagger}(\mathcal{W}(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}), \tilde{\boldsymbol{s}}_\boldsymbol{w})\right) + \ell_{\text{mix}}\left(\boldsymbol{x}'_k, \tilde{\boldsymbol{x}}_\boldsymbol{k}\right) \\ &= \ell_{\text{mix}}\left(\boldsymbol{x}^*_k, f_{\theta_n}(\tilde{\boldsymbol{x}}_k)\right) + \ell_{\text{mix}}\left(\boldsymbol{x}'_k, \tilde{\boldsymbol{x}}_\boldsymbol{k}\right)\end{aligned} \tag{8}$$

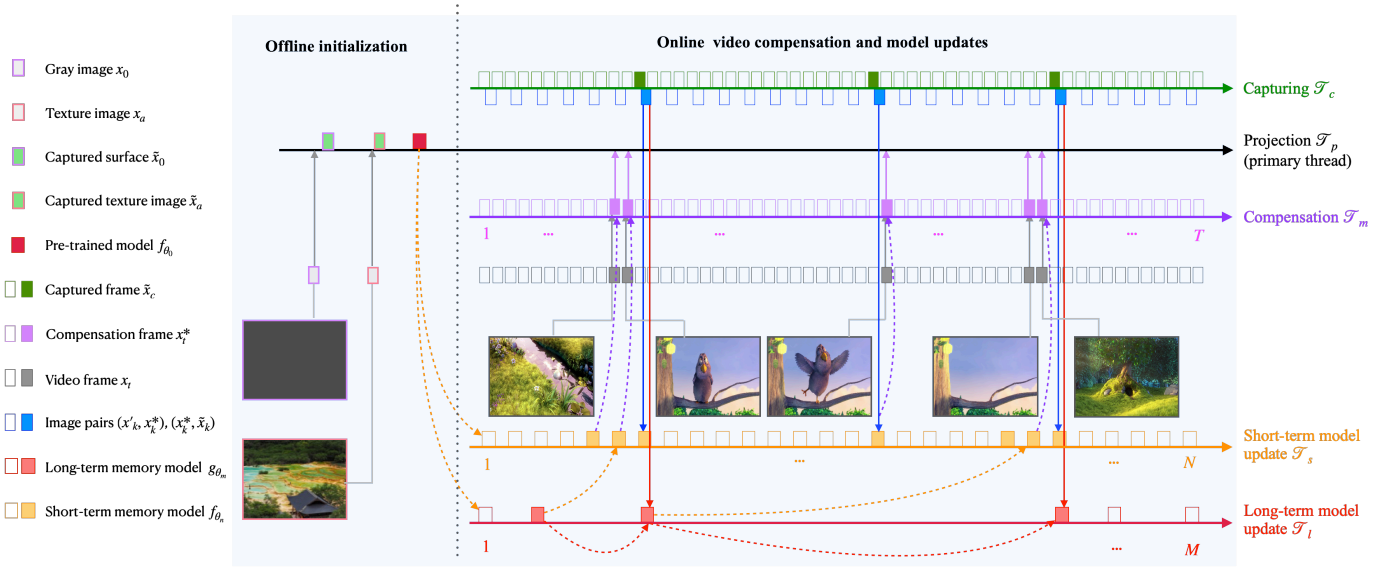


Fig. 3: **Pipeline of ViComp.** The system consists of five threads. The first two threads are timed to start: $\mathcal{T}_p$ for initialization and projection, and $\mathcal{T}_c$ for collecting synchronized frame pairs. The third thread $\mathcal{T}_m$ reads the source video and generates the compensation frame. The last two threads $\mathcal{T}_s$ and $\mathcal{T}_l$ work in coordination, updating model parameters using the synchronized image pairs to adapt to the target environment. The rectangular blocks indicate different image data, while square blocks indicate different models. Solid blocks are examples of empty blocks. The solid lines with arrows represent the transmission of image data between threads, while the dashed lines with arrows represent the transmission of model parameters.

Algorithm 1 Online compensation and model update algorithm

Require: Video frames $\boldsymbol{X'} = \{\boldsymbol{x}'_t\}_{t=0}^{T}$, pre-trained model of displacement field estimation function $\mathcal{F}$, pre-trained model of photometric transformation function $\mathcal{P}^{\dagger}$, cut index $\boldsymbol{H} = \{h_l\}_{l=0}^{L}$.

Ensure: Captured frames $\tilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}\}_{c=0}^{C}$

Initialization:
Project a gray image $\boldsymbol{s_0}$ and capture the surface $\tilde{\boldsymbol{s}}_{\boldsymbol{0}}$, then project a texture image $\boldsymbol{x_a}$ and capture the image $\tilde{\boldsymbol{x}}_{\boldsymbol{a}}$.
Calculate FOV based on $\tilde{\boldsymbol{s}}_{\boldsymbol{0}}$.
Estimate displacement $\boldsymbol{u}$ from $\tilde{\boldsymbol{x}}_{\boldsymbol{a}}$ to $\boldsymbol{x_a}$ using the pretrained model of $\mathcal{F}$, then obtain the warped surface $\tilde{\boldsymbol{s}}_{\boldsymbol{w}}$.
Initialize the short-term memory model f_{θ_0}, and the long-term memory model g_{θ_0} using $\boldsymbol{u}$ and the pre-trained model of $\mathcal{P}^{\dagger}$.
Denote the short-term memory model index as $n = 0, 1, ..., N$ as and the long-term memory model index as $m = 0, 1, ..., M$

while $t <= T$ **do**
  Convert video frame $\boldsymbol{x}_t$ to the desired frame $\boldsymbol{x}'_t$
  if $t >= h_l$ **then**
    Reset f_{θ_n} using long-term memory model g_{θ_m}
  Compensation using short-term memory model: $\boldsymbol{x}_t^* = f_{\theta_n}(\boldsymbol{x}'_t, \tilde{\boldsymbol{s}}_{\boldsymbol{w}}) = \mathcal{P}^{\dagger}(\mathcal{W}(\boldsymbol{x}'_t, \boldsymbol{u}), \tilde{\boldsymbol{s}}_{\boldsymbol{w}})$
  Project $\boldsymbol{x}_t^*$
  Capture the frame $\tilde{\boldsymbol{x}}_c$
  if the k-th frame can be recognized **then**
    if $k > h_l$ **then**
      $l \leftarrow l+1, m \leftarrow m+1$
      Update long-term memory model g_{θ_m} using frame pairs $(\boldsymbol{x}'_k, \boldsymbol{x}_k^*)$ and $(\boldsymbol{x}_k^*, \tilde{\boldsymbol{x}}_k)$
    $n \leftarrow n+1$
    Update short-term memory model f_{θ_n} using frame pairs $(\boldsymbol{x}'_k, \boldsymbol{x}_k^*)$ and $(\boldsymbol{x}_k^*, \tilde{\boldsymbol{x}}_k)$
  $t \leftarrow t+1$
return $\tilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}\}_{c=0}^{C}$

where $\mathcal{L}_{\text{cmp}}$ denotes the compensation loss calculated between the projector input $\boldsymbol{x}_k^*$ and the compensation frame generated from the captured image $\tilde{\boldsymbol{x}}_k$, $\mathcal{L}_{\text{vis}}$ denotes the visual loss computed between the desired effect and the captured frame, and ℓ_{mix} represents a combination of l_1 norm, l_2 norm and the structure similarity [58].

Using this naive fine-tuning method, the short-term memory model can be quickly optimized to a local optimum with a few frame pairs. However, overfitting and error accumulation may appear in this model update process after multiple training iterations with similar content. As a result, it leads to poor display quality. To handle this issue, we leverage a long-term memory model update thread that keeps track of global information using frame pairs with different content to reset the short-term memory model after each cut transition.

In the long-term memory model update thread $\mathcal{T}_l$, inspired by [55], we preserve the knowledge from two models: the long-term student model $g_{\theta'_m}$ and the long-term teacher model g_{θ_m}. The long-term student model $g_{\theta'_m}$ extracts features from the long video, and the parameters are updated according to the loss function:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{\text{lt}} + \mathcal{L}_{\text{ls}} + \mathcal{L}_{\text{vis}} \\ &= \ell_{\text{mix}}\left(\boldsymbol{x}_k^*, g_{\theta'_m}(\tilde{\boldsymbol{x}}_k)\right) + \ell_{\text{mix}}\left(\boldsymbol{x}_K^*, g_{\theta_m}(\tilde{\boldsymbol{x}}_k)\right) + \ell_{\text{mix}}\left(\boldsymbol{x}'_k, \tilde{\boldsymbol{x}}_{\boldsymbol{k}}\right)\end{aligned} \tag{9}$$

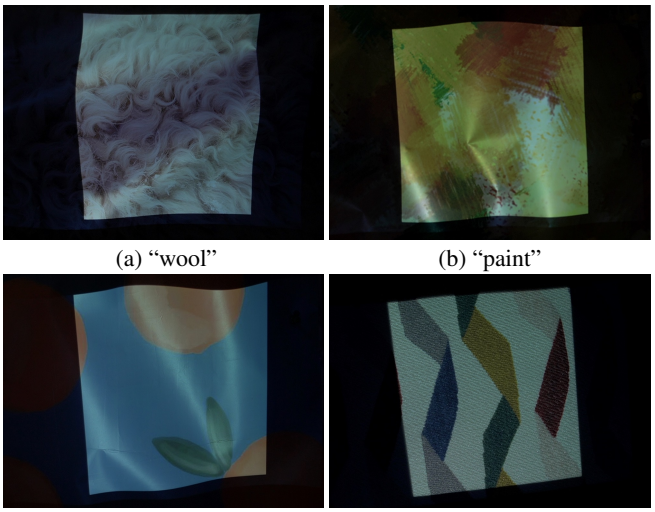
where $\mathcal{L}_{\text{lt}}$ and $\mathcal{L}_{\text{ls}}$ represent the compensation loss calculated using the teacher and student models respectively.

After updating the student model, the long-term teacher model g_{θ_m} can be adjusted by:

$$g_{\theta_m} = \alpha g_{\theta_m} + (1-\alpha) g_{\theta'_m} \tag{10}$$

where α is a smoothing factor, and is set to 0.6 in our experiments. In the transition of different cuts, the short-term memory model f_{θ_n} needs to be reset by the long-term memory model g_{θ_m}.

The collaboration of long-term and short-term memory models enables our compensation model to not only acquire knowledge from the environment but also extract information about the video content. Consequently, it effectively prevents overfitting and reduces error accumulation.



(a) "wool" (b) "paint"

(c) "orange" (d) "pillow"

Fig. 4: Four examples of surfaces used in our experiments. (a) (b) (c) are poster papers with different textures, and (d) is a fabric pillow.

5 Experiments

5.1 System configuration

Our projector-camera system comprises a Sony α6400 camera and an EPSON projector with a resolution of 1920×1080. An Elgato Cam Link 4K video capture card is used to collect real-time data from the camera and then transfer it to the processor. The projector is placed around 1.0 meters in front of the projection surface, while the camera is placed in the range of 0.2 to 0.5 meters away from the projector.

We use four different texture surfaces for projection, including three poster papers with different patterns and one striped fabric pillow (Fig. 4), each paper is covered with plastic film. In each experiment, we arrange them with different deformation and adjust camera settings such as exposure, focus, and white balance manually based on the ambient light and surface material. These settings remained fixed during data capture.

Our system is built on an Intel Core i9 13900K CPU, and the online compensation and model update processes are implemented using Pytorch and executed on an NVIDIA GeForce RTX 4090 GPU. We utilize the Adam optimization algorithm [28] with the learning rates of 0.0001 for updating the short-term memory model and 0.001 for updating the long-term memory model one step at each iteration.

5.2 Experiment settings

The testing video sequences are extracted from two open-source films provided by Blender [9]. We take all the frames in each video except for the opening credits with a large amount of black background and then resize them to 1024×1024. The cut indexes are initially detected by [49] and then revised manually. The details of these sequences are listed in Tab. 1.

Table 1: Video sequences used for compensation. (BBB: Big Buck Bunny.)

Data	Duration	Resolution	#Frame	#Cut
BBB [43]	10m34s	1920×1080	18930	137
Spring [44]	7m44s	2048×858	11040	104

For evaluation, we start a new thread for saving ROI of all synchronized frame pairs $(\tilde{\boldsymbol{x}}, \boldsymbol{x'})$ with the resolution of 256×256, and then measure the similarity between them using PSNR, RMSE, SSIM [58] and ΔE (the CIE standard for perceptual color differences) [46].

Table 2: Quantitative comparison of different methods. We sequentially projected two videos onto three surfaces with different settings. Here are the averages of the results for six setups.

Method	PSNR↑	SSIM↑	RMSE↓	$\Delta E \downarrow$
Uncompensated	15.1456	0.3395	0.3298	17.1034
FFCmpNeSt	18.6041	0.5959	0.2155	13.7359
CmpNeSt-8	23.0093	0.6489	0.1358	7.9959
CmpNeSt-500	23.3515	0.6693	0.1322	**7.6656**
ViComp (ours)	**23.8559**	**0.6833**	**0.1280**	7.9059

5.3 Comparison with different compensation methods

In this experiment, as there is no existing online video compensation method, we compare the proposed ViComp with CompenNeSt++ [21], a state-of-the-art offline full compensation method. To apply to an unknown environment, CompenNeSt++ should be trained/fine-tuned using additional natural image samples collected under the target configuration. Thus, under each setup, we first project 500 natural images onto the surface and collect image pairs in this static environment as training samples. During offline training, we initialize CompenNeSt++ using the pre-trained photometric compensation model provided by [21], which is trained on synthetic data from 100 setups, and then fine-tune the parameters of both $\mathcal{G}^{-1}$ and $\mathcal{P}^{\dagger}$ using 8 natural image samples (CmpNeSt-8) and 500 natural image samples (CmpNeSt-500) respectively. All hyper-parameters are set as [21]. During video playback, our online compensate frames using ViComp and CompenNeSt++ with the two fine-tuned models and then project them onto the surface. Subsequently, we evaluate the display quality of these methods. For a fair comparison, the photometric compensation model of each method uses low-resolution frames as input, and the high-resolution projector output is obtained through upsampling. Additionally, we also evaluate the method that estimates displacement field using FlowFormer and compensates frame color using the pre-trained photometric compensation model in CompenNeSt++. We name this combined method FFCmpNeSt.

From the results in Tab. 2, it can be observed that the proposed online video compensation system achieves significantly better display quality compared to CompenNeSt++ fine-tuned with 8 samples (CmpNeSt-8), while slightly surpassing the results obtained by that fine-tuned with 500 samples (CmpNeSt-500) on PSNR/SSIM/RMSE. In terms of the basic model structure, CompenNeSt++ uses grid sampling as the main method for estimating geometric deformations. As ViComp employs a transformer-based method, it performs better than CmpNeSt-8 and CmpNeSt-500 in handling surface deformations, especially in cases where the surface shape is complex. As FlowFormer can not address the issue caused by surface texture content, the end-to-end methods (CmpNeSt-8 and CmpNeSt-500) achieve better geometric correction effects if the surface texture is easily confused with the projected content. In terms of model fine-tuning, CmpNeSt-8 and CmpNeSt-500 need to acquire additional samples in advance so that they are too complex and time-consuming for non-expert users. Furthermore, they can only learn environmental information, as these samples are unrelated to the video content. Additionally, due to the use of fewer samples for fine-tuning, the display quality of CmpNeSt-8 is consistently lower than that of CmpNeSt-500. In contrast, ViComp directly uses synchronized frame pairs to update the model during video playback. It is user-friendly and can adapt to both the current scene environment and video content.

We also show an example of captured frames in Fig. 5. At the beginning of video playback, the display quality obtained by ViComp is slightly lower than that of CmpNeSt-8 and CmpNeSt-500. However, as the video progresses, our ViComp gradually adapts to the current configuration and learns knowledge of video content from preceding frames, so that it achieves better display quality in subsequent frames. CmpNeSt-8 learns limited information from eight training samples, resulting in the loss of high-frequency details. CmpNeSt-500 achieves good compensation effects in the entire sequence, while its performance is poorer than our method for parts close to pure white or black. Due to

Table 3: Running time of the proposed framework. We projected the video "Spring" ten times. Here is the average running time of the main components. Note that the time spent on displacement estimation is the time taken for flow inference.

Displacement estimation	Projection	Capturing & Recognition	Compensation	Short-term update	Long-term update
2948.8096	24.9066	6.2155	0.8110	5.4570	7.2575

Fig. 5: Qualitative comparison. Project the video "BBB" onto the surface "paint". From top to bottom: desired frame, uncompensated, captured frames with different methods (FFCmpNeSt, CmpNeSt-8, CmpNeSt-500, ViComp). From left to right: frame #480, 2201, 4274, 6087, 7857, 9487, 10098.

the pre-trained model not being fine-tuned with data from the current configuration, the captured frames obtained by FFCmpNeSt differ significantly from the desired effects.

Next, we give the average computational consumption of the main components of our system. As shown in Tab. 3, the offline displacement field estimation takes about 3 seconds, while projecting a frame in the primary thread costs approximately 25 milliseconds. The running time for compensating is under 1 millisecond per frame, and the time needed for updating the parameters of our compensation model is about 5.5 milliseconds per iteration. Therefore, benefiting from the cooperation of multi-threads, the proposed system can achieve real-time projection compensation and model update under our experimental conditions.

5.4 Comparison with different model update strategies

In this section, we show the effectiveness of our model update strategy. In our system, the parameters of the displacement field estimation network are frozen during the model update and only the parameters of the photometric transformation estimation network are adjusted. To ensure a fair comparison, we pre-estimate a displacement field in each setup and then online adjust parameters of $\mathcal{P}^{\dagger}$ by sequentially activating short and long-term memory model update threads with different

Table 4: Quantitative comparison of different models update strategies. We projected two videos onto three different surfaces. Here are the averages of the results for six setups. (S is the abbreviation of the short-term memory model, and L is the abbreviation of the long-term memory model. FineCoTTA: Fine-tuned (S) + CoTTA (L))

Method	PSNR↑	SSIM↑	RMSE↓	$\Delta E\downarrow$
Uncompensated	15.5524	0.3467	0.3093	15.5923
Pre-trained	19.8838	0.5886	0.1859	11.3167
Fine-tuned (S)	22.7853	0.6011	0.1518	8.5260
Fine-tuned (L)	23.2339	0.6423	0.1332	8.4541
CoTTA(L)	20.7329	0.6063	0.1698	10.6009
FineCoTTA	22.7181	0.6429	0.1390	8.5822
ViComp (L)	23.3521	0.6517	0.1303	8.1515
ViComp (S+L)	**24.0956**	**0.6689**	**0.1214**	**7.4182**

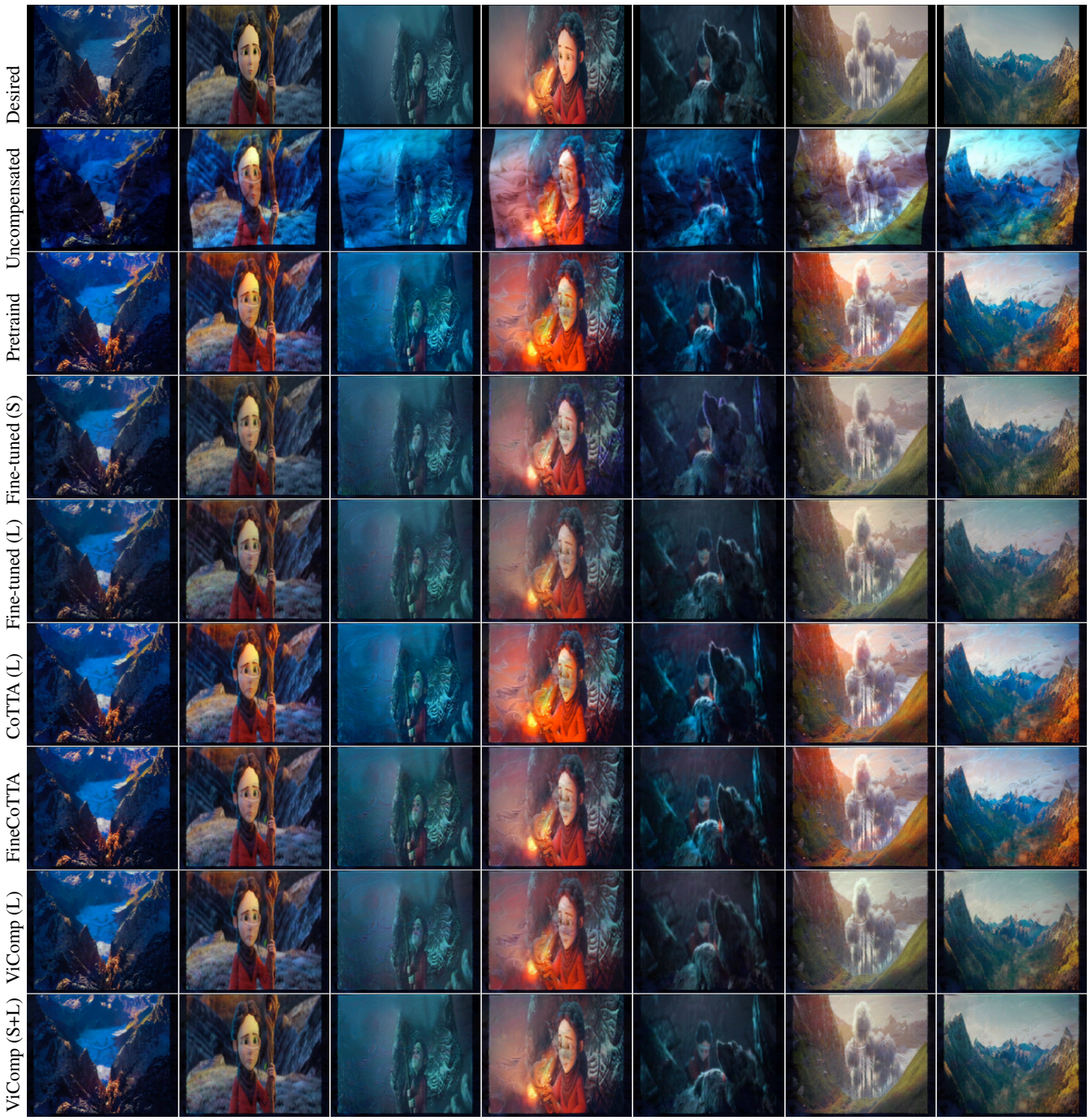


Fig. 6: Qualitative comparison. Projecting the video "Spring" to the surface "wool". From top to bottom: desired frame, uncompensated, captured frames with different model update strategies (Pre-trained, Fine-tuned (S), Fine-tuned (L), CoTTA (L), FineCoTTA, ViComp (L), ViComp (S+L)). From left to right: frame #487, 1501, 5053, 5503, 6374, 8436, 9095.

update strategies. First, we directly compare with the pre-trained model of $\mathcal{P}^{\dagger}$ provided by [21]. Subsequently, We enable the long-term and short-term threads respectively and utilize the naive fine-tuning method for parameter adjustment in the activated model update thread. These two model update approaches are referred to as Fine-tuned (S) and Fine-tuned (L). Following this, we evaluate the method that updates the compensation model by the state-of-the-art test time adaptation approach CoTTA [55] using the first synchronized frame pairs of each cut and refer to it as CoTTA (L). The hyper-parameters of CoTTA are set as specified in [55]. Additionally, we replace our long-term memory model update strategy with CoTTA and name this combined method FineCoTTA. Finally, we disable the short-term memory update thread of ViComp and denote it as ViComp (L). To mark the active state of each model update thread, we represent ViComp as ViComp (S+L). In this experiment, the high-resolution projector input of each method is still obtained through bicubic interpolation. The data acquisition method and model update frequency used during each thread are consistent with ViComp. The learning rates of short and long-term memory model update processes are set to 0.0001 and 0.001 respectively.

In Tab. 4, compared with other methods, benefiting from the cooperation of short-term and long-term memory model update, ViComp (S+L) achieves the best display quality in terms of PSNR, SSIM, RMSE, and ΔE. Three factors contribute to this notable performance. First, by employing every synchronous image pair to update the short-term model via the naive fine-tuning method, Fine-tuned (S) enables the parameters to rapidly adapt to the target environment and the played content. However, as the main loss function involves the distance between two compensated images, frequent training with frame pairs containing similar content may lead to error accumulation and overfitting so that the captured images lose some details. Consequently, Fine-tuned (S) achieves higher PSNR and lower SSIM in the entire video. Second, due to the limited number of training samples and iterations, updating the model with the first frame pair after each cut transition results in lower scores at the beginning of video playback. The performance of methods using only the long-term memory model update strategy (ViComp (L) and Fine-tuned (L)) improves as the number of updates increases. Third, in ViComp (L), we retain the averaging mechanism but omit the stochastic model parameter reset strategy used in CoTTA. Because unlike tasks commonly addressed by test-time adaptation methods, our system can obtain pairs of projection-captured frames during video playback in a static environment. These frame pairs can be used as training data with ground truth to fine-tune the model for better adaptation to the current configuration and video content. The scores of CoTTA (L) and FineCoTTA show that this parameter reset method is unsuitable for our task, while the fact that ViComp (L) obtains slightly higher scores than Fine-tuned (L) demonstrates the averaging mechanism can further reduce error accumulation during model update. In conclusion, benefiting from the use of long-term and short-term memory model update strategy, ViComp (S+L) achieves the best display quality.

We also show the qualitative results in Fig. 6. At the beginning of video playback, due to using all synchronized frame pairs for model updates, Fine-tuned (S) rapidly converges and obtains captured effects close to the desired frames. While, since only several frame pairs with different content are used to update the model, the display quality of Fine-tuned (L) is slightly lower than that achieved by Fine-tuned (S). As the video plays continuously, the naive fine-tuning method tends to forget the previous models and accumulate errors in the updating process, gradually losing image details. In contrast, as CoTTA restores the parameters of the source model stochastically, CoTTA(L) and FineCoTTA which adjust parameters using CoTTA achieve a slightly better display quality than the pre-trained model, even after multiple iterations. Our system employs an averaging mechanism to update the long-term memory model gradually and continually resets the short-term memory model after a cut transition. Benefiting from this model update strategy, it can converge to the optimal solution quickly and reduce the accumulation of errors to some extent. As a result, our ViComp (S+L) achieves the best performance across the entire video.

6 Discussion

In our system, estimating the displacement field using an optical flow-based method allows for quick adaptation to different environments. However, environmental disturbances, such as significant displacement differences between pixels, textures on the surface closely resembling those in the projected image, and highlights and shadows on the surface, significantly impact its accuracy. Although projecting an additional natural image with a rich texture can enhance the accuracy of estimating the displacement field to some extent, more stable geometric correction methods should be explored in future studies.

Additionally, as it is easy to implement without requiring extensive data collection, the proposed system is user-friendly and can be effectively applied in scenarios that require continuous video playback. However, since our photometric compensation method relies on pre-captured surface images, it becomes less effective when the configuration changes. Therefore, aiming to apply it to dynamic environments, a photometric compensation model with fewer constraints is expected.

7 Conclusion

This paper builds a user-friendly online video compensation system for projector applications in unknown environments. The system maintains five threads in parallel to ensure simultaneous video compensation, data collection, and model update. To enhance the efficiency of video compensation and model update, we present a base model that can generate compensation frames with high display quality and can be rapidly optimized with a small amount of data during testing. Furthermore, to improve the video compensation performance, by taking into account the strong correlation between adjacent frames within a single cut and the difference across different cuts, we propose a model update strategy that leverages a long-term memory model to acquire knowledge about the target environment and a short-term memory model to extract information about the video content. Benefiting from these techniques, our system can compensate video frames and adapt to the target environment without manual intervention.

Acknowledgments

Wang was supported in part by the Nature Science Foundation of China (62202135). Huang was supported in part by the Nature Science Foundation of China (62302401). Ling was not supported by any research fund for this work.

References

[1] R. Akiyama, T. Fukiage, and S. Nishida. Perceptually-based optimization for radiometric projector compensation. In *IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops*, pp. 750–751, 2022. 2, 3

[2] O. Bimber. Multi-projector techniques for real-time visualizations in everyday environments. In *IEEE Virtual Reality Conference*, pp. 320–320, 2006. 1

[3] O. Bimber, A. Emmerling, and T. Klemmer. Embedded entertainment with smart projectors. *Computer*, 38(1):48–55, 2005. 1

[4] O. Bimber, D. Iwai, G. Wetzstein, and A. Grundhöfer. The visual computing of projector-camera systems. *Comput. Graph. Forum*, 27:2219–2245, 2008. 2

[5] P.-A. Bokaris, M. Gouiffès, C. Jacquemin, and J.-M. Chomaz. Photometric compensation to dynamic surfaces in a projector-camera system. In *European Conference on Computer Vision Workshops*, pp. 283–296. Cham, 2015. 2

[6] P.-A. Bokaris, M. Gouiffès, C. Jacquemin, J.-M. Chomaz, and A. Trémeau. One-frame delay for dynamic photometric compensation in a projector-camera system. In *IEEE International Conference on Image Processing*, pp. 2675–2679, 2015. 2

[7] D. Brahma and P. Rai. A probabilistic framework for lifelong test-time adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3582–3591, 2023. 2

[8] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, pp. 611–625. Berlin, Heidelberg, 2012. 3

[9] B. O. Community. Blender - a 3D modelling and rendering package, 2018. 6

[10] M. Döbler, R. A. Marsden, and B. Yang. Robust mean teacher for continual and gradual test-time adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7704–7714, 2023. 2

[11] Y. Erel, D. Iwai, and A. H. Bermano. Neural projection mapping using reflectance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(11):4339–4349, 2023. 2, 3

[12] K. Fujii, M. Grossberg, and S. Nayar. A projector-camera system with real-time photometric adaptation for dynamic environments. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005. 1, 2, 3

[13] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014. 4

[14] J. Geng. Structured-light 3D surface imaging: a tutorial. *Advances in Optics and Photonics*, 3:128–160, 2011. 2, 3

[15] A. Grundhöfer and D. Iwai. Recent advances in projection mapping algorithms, hardware and applications. *Computer Graphics Forum*, 37:653–675, 2018. 2

[16] T. Hamasaki, Y. Itoh, Y. Hiroi, D. Iwai, and M. Sugimoto. Hysar: Hybrid material rendering by an optical see-through head-mounted display with spatial augmented reality projection. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1457–1466, 2018. 1

[17] N. Hashimoto, R. Koizumi, and D. Kobayashi. Dynamic projection mapping with a single IR camera. *International Journal of Computer Games Technology*, 2017:1–10, 2017. 2

[18] N. Hashimoto and K. Yoshimura. Radiometric compensation for non-rigid surfaces by continuously estimating inter-pixel correspondence. *The Visual Computer*, 37(1):175–187, 2021. 2

[19] B. Huang and H. Ling. End-to-end projector photometric compensation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6803–6812, 2019. 2

[20] B. Huang and H. Ling. DeProCams: Simultaneous relighting, compensation and shape reconstruction for projector-camera systems. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2725–2735, 2021. 2

[21] B. Huang, T. Sun, and H. Ling. End-to-end full projector compensation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):2953–2967, 2022. 1, 2, 3, 4, 6, 8

[22] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li. Flowformer: A transformer architecture for optical flow. In *European Conference on Computer Vision*, pp. 668–685, 2022. 3

[23] M. T. Ibrahim, M. Gopi, and A. Majumder. Projector-camera calibration on dynamic, deformable surfaces. In *IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops*, pp. 905–906, 2023. 2

[24] Y. Iwasawa and Y. Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. In *Advances in Neural Information Processing Systems*, vol. 34, pp. 2427–2440, 2021. 2

[25] S. Kagami and K. Hashimoto. Animated stickies: Fast video projection mapping onto a markerless plane through a direct closed-loop alignment. *IEEE Transactions on Visualization and Computer Graphics*, 25(11):3094–3104, 2019. 2, 3

[26] Y. Kageyama, D. Iwai, and K. Sato. Online projector deblurring using a convolutional neural network. *IEEE Transactions on Visualization and Computer Graphics*, 28(5):2223–2233, 2022. 2

[27] Y. Kemmoku and T. Komuro. AR tabletop interface using a head-mounted projector. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, pp. 288–291, 2016. 1

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 6

[29] P. Kurth, V. Lange, M. Stamminger, and F. Bauer. Real-time adaptive color correction in dynamic projection mapping. In *IEEE International Symposium on Mixed and Augmented Reality*, pp. 174–184, 2020. 2

[30] Y. Li, N. Bao, Q. Yuan, and D. Lu. Real-time continuous geometric calibration for projector-camera system under ambient illumination. In *International Conference on Virtual Reality and Visualization*, pp. 7–12, 2012. 2

[31] Y. Li, W. Yin, J. Li, and X. Xie. Physics-based efficient full projector compensation using only natural images. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–15, 2023. 1, 2, 3

[32] K.-L. Low, A. Ilie, G. Welch, and A. Lastra. Combining head-mounted and projector-based displays for surgical training. In *IEEE Virtual Reality*, pp. 110–117, 2003. 1

[33] A. Majumder, D.-Q. Lai, and M. A. Tehrani. A multi-projector display system of arbitrary shape, size and resolution. In *IEEE Virtual Reality*, pp. 339–340, 2015. 1

[34] L. Miyashita, Y. Watanabe, and M. Ishikawa. Midas projection: Markerless and modelless dynamic projection mapping for material representation. *ACM Transactions on Graphics*, 37(6), 2018. 1

[35] Y. Miyatake, T. Hiraki, D. Iwai, and K. Sato. Haptomapping: Visuo-haptic augmented reality by embedding user-imperceptible tactile display control signals in a projected image. *IEEE Transactions on Visualization and Computer Graphics*, 29(4):2005–2019, 2023. 1

[36] G. Narita, Y. Watanabe, and M. Ishikawa. Dynamic projection mapping onto deforming non-rigid surface using deformable dot cluster marker. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1235–1248, 2017. 1, 2

[37] A. T. Nguyen, T. Nguyen-Tang, S.-N. Lim, and P. H. Torr. TIPI: Test time adaptation with transformation invariance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24162–24171, 2023. 2

[38] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. 3

[39] K. Ozacar, T. Hagiwara, J. Huang, K. Takashima, and Y. Kitamura. Coupled-clay: Physical-virtual 3d collaborative interaction environment. In *IEEE Virtual Reality*, pp. 255–256, 2015. 1

[40] H. Park, M.-H. Lee, B.-K. Seo, J.-I. Park, M.-S. Jeong, T.-S. Park, Y. Lee, and S.-R. Kim. Simultaneous geometric and radiometric adaptation to dynamic surfaces with a mobile projector-camera system. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(1):110–115, 2008. 2

[41] R. Raskar. Immersive planar display using roughly aligned projectors. In *Proceedings IEEE Virtual Reality*, pp. 109–115, 2000. 1

[42] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Cham, 2015. 2

[43] T. Roosendaal(Producer). Big buck bunny. Blender Foundation, 2008. www.bigbuckbunny.org. 6

[44] T. Roosendaal(Producer). Spring. Blender Foundation, 2012. cloud.blender.org/spring. 6

[45] M. Shahpaski, L. R. Sapaico, G. Chevassus, and S. Süsstrunk. Simultaneous geometric and radiometric calibration of a projector-camera pair. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3596–3604, 2017. 2

[46] G. Sharma, W. Wu, and E. N. Dalal. The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research and Application*, 30:21–30, 2005. 6

[47] K.-T. Shih, J.-S. Liu, F. Shyu, and H. H. Chen. Enhancement and speedup of photometric compensation for projectors by reducing inter-pixel coupling and calibration patterns. *IEEE Transactions on Image Processing*, 30:418–430, 2021. 1, 2

[48] C. Siegl, M. Colaianni, L. Thies, J. Thies, M. Zollhöfer, S. Izadi, M. Stamminger, and F. Bauer. Real-time pixel luminance optimization for dynamic multi-projection mapping. *ACM Transactions on Graphics*, 34:1–11, 2015. 2

[49] T. Souček and J. Lokoč. Transnet v2: An effective deep network architecture for fast shot transition detection. *arXiv preprint arXiv: 2008.04838*, 2020. 6

[50] M. Sugimoto, D. Iwai, K. Ishida, P. Punpongsanon, and K. Sato. Directionally decomposing structured light for projector calibration. *IEEE Transactions on Visualization and Computer Graphics*, 27(11):4161–4170, 2021. 1, 2

[51] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning*, vol. 119, pp. 9229–9248, 2020. 2

[52] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, vol. 30, 2017. 2

[53] T. Ueda, D. Iwai, T. Hiraki, and K. Sato. Illuminated focus: Vision augmentation using spatial defocusing via focal sweep eyeglasses and high-speed projector. *IEEE Transactions on Visualization and Computer Graphics*, 26(5):2051–2061, 2020. 1

[54] D. Wang, E. Shelhamer, S. Liu, B. A. Olshausen, and T. Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. 2

[55] Q. Wang, O. Fink, L. Van Gool, and D. Dai. Continual test-time domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7191–7201, 2022. 2, 3, 5, 8

[56] Y. Wang, H. Ling, and B. Huang. Compenhr: Efficient full compensation for high-resolution projector. In *IEEE Conference Virtual Reality and 3D User Interfaces*, pp. 135–145, 2023. 1, 2

[57] Y. Wang, R. Wang, Z. Wang, and W. Gao. Asymmetric circular projection for dynamic virtual reality video stream switching. In *IEEE International Conference on Image Processing*, pp. 2726–2730, 2017. 1

[58] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 5, 6

[59] Y. Watanabe, T. Kato, and M. ishikawa. Extended dot cluster marker for high-speed 3d tracking in dynamic projection mapping. In *IEEE International Symposium on Mixed and Augmented Reality*, pp. 52–61, 2017. 2

[60] R. Zheng, X. Huangfu, Y. Sun, and L. Yu. Projection deformation based motion compensation for panoramic video. In *IEEE International Symposium on Circuits and Systems*, pp. 1–5, 2018. 1