CompenNet++: End-to-end Full Projector Compensation

Bingyao Huang
Temple University
bingyao.huang@temple.edu

Haibin Ling*
Stony Brook University
hling@cs.stonybrook.edu

Abstract

Full projector compensation aims to modify a projector input image such that it can compensate for both geometric and photometric disturbance of the projection surface. Traditional methods usually solve the two parts separately, although they are known to correlate with each other. In this paper, we propose the first end-to-end solution, named CompenNet++, to solve the two problems jointly. Our work non-trivially extends CompenNet [15], which was recently proposed for photometric compensation with promising performance. First, we propose a novel geometric correction subnet, which is designed with a cascaded coarse-to-fine structure to learn the sampling grid directly from photometric sampling images. Second, by concatenating the geometric correction subset with CompenNet, CompenNet++ accomplishes full projector compensation and is end-to-end trainable. Third, after training, we significantly simplify both geometric and photometric compensation parts, and hence largely improves the running time efficiency. Moreover, we construct the first setup-independent full compensation benchmark to facilitate the study on this topic. In our thorough experiments, our method shows clear advantages over previous arts with promising compensation quality and meanwhile being practically convenient.

1. Introduction

With the recent advance in projector technologies, projectors have been gaining increasing popularity with many applications [1, 4, 8, 9, 11, 16, 24, 29, 35, 36, 39]. Existing systems typically request the projection surface (screen) to be planar, white and textureless, under reasonable environment illumination. These requests often create bottlenecks for generalization of projector systems. Projector geometric correction [5, 24, 28, 29, 38] and photometric compensation [1, 3, 10, 15, 39], or full projector geometric correction and photometric compensation[1] [2, 4, 12, 30, 35, 36] aim to address this issue by modifying a projector input image to compensate for the projection setup geometry and associated photometric environment. An example from our solution is illustrated in Fig. 1, where the compensated projection result (e) is clearly more visually pleasant than the uncompensated one in (c).

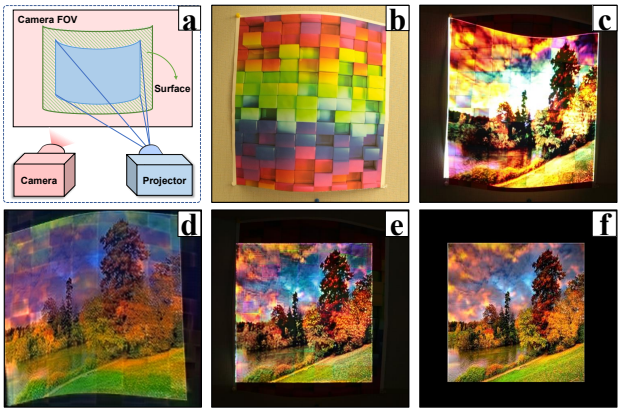

Figure 1: Full projector geometric correction and photometric compensation: **(a)** system setup with nonplanar and textured surface **(b)**, **(c)** projection result without compensation, **(d)** fully compensated image by our method, **(e)** camera-captured compensated projection result (*i.e.* (d) projected onto (b)), and **(f)** desired visual effect. Comparing (c) and (e) we see clearly improved geometry, color and details.

A typical full compensation system consists of a projector-camera (pro-cam) pair and a nonplanar textured projection surface placed at a fixed distance and orientation (Fig. 1(a)). Most existing methods work in two separate steps: (1) geometric surface modeling, *e.g.*, via a sequence of structured light (SL) patterns [8, 22], and (2) color compensation on top of the geometrically corrected projection. Despite relatively easy to implement, this two-step pipeline has two major issues. First, geometric mapping/correction is usually performed offline and assumed independent of photometric compensation. This step typically requests certain patterns (*e.g.* SL grid) that may be disturbed by surface appearance (*e.g.* reflection, see Fig. 6). Second, due to the extremely complex photometric process involved in pro-cam systems, it is hard for traditional photometric compensation solutions to faithfully accomplish their task.

*Corresponding author.
[1] In the rest of the text, we call it *full compensation* for conciseness.

Recently, an end-to-end photometric compensation algorithm named CompenNet [15] is introduced and shows great advantage of deep neural networks over traditional solutions. However, it leaves the geometric correction part untouched and hence is restricted on planar surfaces. Moreover, as will be shown in this paper, its running time efficiency still has room to improve.

To address the above mentioned issues, in this paper we propose the first end-to-end solution, named *CompenNet++*, for full projector compensation. CompenNet++ non-trivially extends CompenNet and jointly solves both geometric correction and photometric compensation in a unified convolutional neural network (CNN) pipeline. In particular, by taking into consideration of both geometric and photometric ingredients in the compensation formulation, we carefully design CompenNet++ as composed of two subnets. The first subnet is a novel cascaded coarse-to-fine sampling grid prediction subnet, named *WarpingNet* (Fig. 3), which performs geometric correction; while the second subnet is an improved version of the original CompenNet for photometric compensation. It is worth highlighting that the two subnets are concatenated directly, which makes CompenNet++ end-to-end trainable.

Moreover, following evaluation procedure in [15], we construct the first known setup-independent full compensation evaluation benchmark for nonplanar textured surfaces. The proposed CompenNet++ is evaluated on this benchmark that is carefully designed to cover various challenging factors. In the experiments, CompenNet++ demonstrates clear advantages compared with state-of-the-arts.

Our contributions can be summarized as follows:

1. The proposed CompenNet++ is the first end-to-end full compensation system.
2. Compared with two-step methods, CompenNet++ learns the geometric correction without extra sampling images and outperforms the compared counterparts.
3. Two task-specific weight initialization approaches are proposed to ensure the convergence and stability of CompenNet++.
4. Novel simplification techniques are developed to improve the running time efficiency of CompenNet++.

The source code, benchmark and experimental results are available at https://github.com/BingyaoHuang/CompenNet-plusplus.

2. Related Works

In this section, we review existing projector compensation methods in roughly two types: full compensation [4, 12, 30, 34–36] and partial ones [1, 3, 9, 10, 15, 20, 25, 33, 37].

2.1. Full compensation methods

Full compensation methods perform both geometric correction and photometric compensation. The pioneer work by Raskar *et al*. [30] creates projection mapping animations on nonplanar colored objects with two projectors. Despite compensating both geometry and photometry, manual registrations using known markers are required. Harville *et al*. [12] propose a full multi-projector compensation method applied to a white curved screen. The pro-cam pixel correspondences are obtained via 8-12 SL images. Despite being effective to blend multiple projector's color, this method assumes a textureless projection surface.

Recently, Siegl *et al*. [35, 36] perform full compensation on nonplanar Lambertian surfaces for dynamic real-time projection mapping. Similar to [12], they assume the target objects are white and textureless. Asayama *et al*. [2] attach visual markers to nonplanar textured surfaces for real-time object pose tracking. To remove the disturbance of the markers, photometric compensation is applied to hide the markers from the viewer, and extra IR cameras/emitters are required accordingly. Shahpaski *et al*. [34] embed color squares in the projected checkerboard pattern to calibrate both geometry and gamma function. Although only two shots are required, this method needs a pre-calibrated camera and another planar printed checkerboard target. Moreover, it only performs a uniform gamma compensation without compensating the surface, and thus may not work well on nonplanar textured surfaces.

2.2. Partial compensation methods

Compared to full compensation methods, partial compensation ones typically perform either geometric correction [5, 24, 28, 29, 38] or photometric compensation [1, 3, 10, 15, 39]. Due to the strong mutual-dependence between geometric correction and photometric compensation and to avoid propagated errors from the other part, these methods assume the other part is already performed as a prerequisite.

Geometric correction. Without using specialized hardware, such as a coaxial pro-cam pair [7], pro-cam image pairs' geometric mapping need to be estimated using methods such as SL [5, 28, 29, 38], markers [24] or homographies [15]. Raskar *et al*. [29] propose a conformal texture mapping method to geometrically register multiple projectors for nonplanar surface projections, using SL and a calibrated camera. Tardif *et al*. [38] achieve similar results without calibrating the pro-cam pair. The geometrically corrected image is generated by SL inverse mapping. Similarly, Boroomand *et al*. [5] propose a saliency-guided SL geometric correction method. Narita *et al*. [24] use IR ink printed fiducial markers and a high-frame-rate camera for dynamic non-rigid surface projection mapping, which requires extra devices as [2].

Photometric compensation. These methods assume the pro-cam image pairs are registered as a prerequisite and can be roughly categorized into two types: context-independent [9, 10, 25, 33] and context-aware ones [1, 3, 15, 20, 37],

where context-aware ones typically assume pro-cam pixels one-to-one mapping and context-aware ones also consider neighborhood/global information. A detailed review can be found in [11]. Previous compensation methods either assume the compensation is partially done as a prerequisite or perform two-step compensation separately. However, separating the two steps is known to subject to suboptimal solutions. To the best of our knowledge, there exists no previous method that performs simultaneous full pro-cam image geometric correction and projector photometric compensation.

Belonging to the full compensation regime, our CompenNet++ is the first to jointly learn geometric correction and photometric compensation in an end-to-end framework. Though some part of CompenNet++ is based on CompenNet, there are significant differences: (1) CompenNet++ is for full projector compensation; (2) the photometric part in CompenNet++ extends CompenNet by trimming the surface image branch, and hence improves runtime efficiency with no performance drop; and (3) the concatenation of the geometric and photometric parts in CompenNet++ allows both parts to be jointly trained end-to-end.

3. End-to-end Full Projector Compensation

3.1. Problem formulation

Our full projector compensation system consists of an uncalibrated pro-cam pair and a nonplanar textured projection surface placed at a fixed distance and orientation (Fig. 1(a)). Following the convention of [15] we extend the photometric compensation formulation to a full compensation one. Denote a projector input image by $\boldsymbol{x}$, the composite geometric projection and radiometric transfer function by π_p and projector geometric and photometric intrinsics and extrinsics by $\boldsymbol{p}$. Then, the projected radiance can be denoted by $\pi_p(\boldsymbol{x}, \boldsymbol{p})$. Let the composite surface reflectance, geometry and pose be $\boldsymbol{s}$, surface bidirectional reflectance distribution function (BRDF) be π_s, the global lighting irradiance distribution be $\boldsymbol{g}$, camera's composite capturing function be π_c, and its composite intrinsics and extrinsics be $\boldsymbol{c}$. Then the camera-captured image $\tilde{\boldsymbol{x}}$ is given by[2]:

$$\tilde{\boldsymbol{x}} = \pi_c\big(\pi_s\big(\pi_p(\boldsymbol{x}, \boldsymbol{p}), \boldsymbol{g}, \boldsymbol{s}\big), \boldsymbol{c}\big) \tag{1}$$

Note the composite geometric and radiometric process in Eq. 1 is very complex and obviously has no closed form solution. Instead, we find that $\boldsymbol{p}$ and $\boldsymbol{c}$ are constant once the setup is fixed, thus, we disentangle the geometric and radiometric transformations and absorb $\boldsymbol{p}$ and $\boldsymbol{c}$ in two functions: $\mathcal{T} : \mathbb{R}^{H_1 \times W_1 \times 3} \mapsto \mathbb{R}^{H_2 \times W_2 \times 3}$ that geometrically warps a projector input image to camera-captured image; and $\mathcal{F} : \mathbb{R}^{H_1 \times W_1 \times 3} \mapsto \mathbb{R}^{H_1 \times W_1 \times 3}$ that photometrically transforms a projector input image to an uncompensated camera capture image (aligned with projector's view). Thus, Eq. 1 can be reformulated as:

$$\tilde{\boldsymbol{x}} = \mathcal{T}(\mathcal{F}(\boldsymbol{x}; \boldsymbol{g}, \boldsymbol{s})) \tag{2}$$

Full projector compensation aims to find a projector input image $\boldsymbol{x}^*$, named *compensation image* of $\boldsymbol{x}$, such that the viewer perceived projection result is the same as the ideal desired viewer perceived image[3], *i.e.*,

$$\mathcal{T}(\mathcal{F}(\boldsymbol{x}^*; \boldsymbol{g}, \boldsymbol{s})) = \boldsymbol{x} \tag{3}$$

Thus the compensation image $\boldsymbol{x}^*$ in Eq. 3 is solved by:

$$\boldsymbol{x}^* = \mathcal{F}^\dagger(\mathcal{T}^{-1}(\boldsymbol{x}); \boldsymbol{g}, \boldsymbol{s}). \tag{4}$$

Following [15], we capture the spectral interactions between $\boldsymbol{g}$ and $\boldsymbol{s}$ using a camera-captured surface image $\tilde{\boldsymbol{s}}$ under the global lighting and the projector backlight:

$$\tilde{\boldsymbol{s}} = \mathcal{T}(\mathcal{F}(\boldsymbol{x}_0; \boldsymbol{g}, \boldsymbol{s})), \tag{5}$$

where $\boldsymbol{x}_0$ is set to a plain gray image to provide some illumination.

It is worth noting that other than the surface patches illuminated by the projector, the rest part of the surface outside the projector FOV does not provide useful information for compensation (Fig. 1(a) green part), thus $\tilde{\boldsymbol{s}}$ in Eq. 5 can be approximated by a subregion of camera-captured image $\mathcal{T}^{-1}(\tilde{\boldsymbol{s}})$ (Fig. 1(a) blue part). Substituting $\boldsymbol{g}$ and $\boldsymbol{s}$ in Eq. 4 with $\mathcal{T}^{-1}(\tilde{\boldsymbol{s}})$, we have the compensation problem as

$$\boldsymbol{x}^* = \mathcal{F}^\dagger\big(\mathcal{T}^{-1}(\boldsymbol{x}); \mathcal{T}^{-1}(\tilde{\boldsymbol{s}})\big), \tag{6}$$

where $\mathcal{F}^\dagger$ is the pseudo-inverse of $\mathcal{F}$ and $\mathcal{T}^{-1}$ is the inverse of the geometric transformation $\mathcal{T}$. Obviously, Eq. 6 has no closed form solution.

3.2. Learning-based formulation

Investigating the formulation in §3.1 we find that:

$$\tilde{\boldsymbol{x}} = \mathcal{T}\big(\mathcal{F}(\boldsymbol{x}; \boldsymbol{s})\big) \;\Rightarrow\; \boldsymbol{x} = \mathcal{F}^\dagger\big(\mathcal{T}^{-1}(\tilde{\boldsymbol{x}}); \mathcal{T}^{-1}(\tilde{\boldsymbol{s}})\big) \tag{7}$$

We model $\mathcal{F}^\dagger$ and $\mathcal{T}^{-1}$ jointly with a deep neural network named *CompenNet++* and denoted as $\pi^\dagger_{\boldsymbol{\theta}}$ (Fig. 2(b)):

$$\hat{\boldsymbol{x}} = \pi^\dagger_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}; \tilde{\boldsymbol{s}}), \tag{8}$$

where $\hat{\boldsymbol{x}}$ is the compensation of $\tilde{\boldsymbol{x}}$ (not $\boldsymbol{x}$) and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_\mathcal{F}, \boldsymbol{\theta}_\mathcal{T}\}$ contains the learnable network parameters. In the rest of the paper, we abuse the notation $\pi^\dagger_{\boldsymbol{\theta}}(\cdot, \cdot) \equiv \mathcal{F}^\dagger_{\boldsymbol{\theta}_\mathcal{F}}\big(\mathcal{T}^{-1}_{\boldsymbol{\theta}_\mathcal{T}}(\cdot); \mathcal{T}^{-1}_{\boldsymbol{\theta}_\mathcal{T}}(\cdot)\big)$ for conciseness. Note that $\mathcal{F}^\dagger$ rather than $\pi^\dagger$ here is the equivalent $\pi^\dagger$ in [15].

We train CompenNet++ over sampled image pairs like $(\tilde{\boldsymbol{x}}, \boldsymbol{x})$ and a surface image $\tilde{\boldsymbol{s}}$ (Fig. 2(a)). By using Eq. 8, we can generate a set of N training pairs, denoted as $\mathcal{X} = \{(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_i)\}_{i=1}^N$. Then, with a loss function $\mathcal{L}$, CompenNet++ can be learned by

[2] As in [15], we use 'tilde' ($\tilde{\boldsymbol{x}}$) to indicate a camera-captured image.

[3] In practice, it depends on the optimal displayable area (Fig. 4).

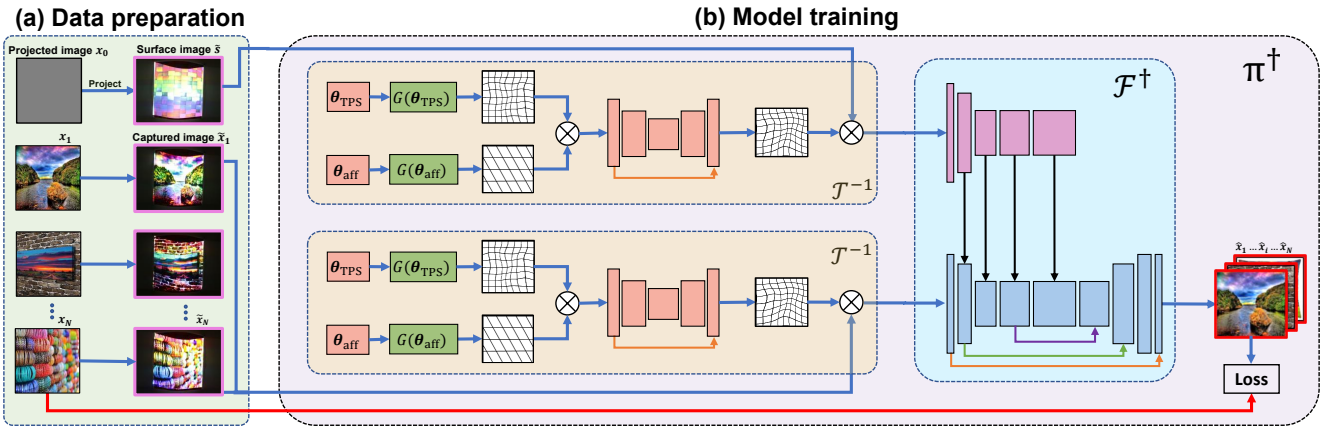

Figure 2: Training of CompenNet++ in two major steps. **(a)** Project and capture a surface image and a set of sampling images. **(b)** CompenNet++, *i.e.*, $\pi^{\dagger}_{\boldsymbol{\theta}}$, is trained using the data prepared in (a).

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}'} \sum_i \mathcal{L}\big(\hat{\boldsymbol{x}}_i = \pi^{\dagger}_{\boldsymbol{\theta}'}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{s}}),\ \boldsymbol{x}_i\big) \tag{9}$$

We use the loss function below to jointly optimize the color fidelity (pixel-wise ℓ_1) and structural similarity (SSIM):

$$\mathcal{L} = \mathcal{L}_{\ell_1} + \mathcal{L}_{\text{SSIM}} \tag{10}$$

The advantages of this loss function are shown in [15, 40].

3.3. Network design

Based on the above formulation, our CompenNet++ is designed with two subnets, a **WarpingNet** $\mathcal{T}^{-1}$ that corrects the geometric distortions and warps camera-captured uncompensated images to projector image space; and a **CompenNet** $\mathcal{F}^{\dagger}$ that photometrically compensates warped images. The network architecture is shown in Fig. 2. For compactness, we move the detailed parameters of CompenNet++ to the supplementary material.

WarpingNet. Note directly estimating nonparametric geometric correction is difficult and computationally expensive. Instead, we model the geometric correction as a cascaded coarse-to-fine process, as inspired by the work in [18, 31]. As shown in Fig. 3, WarpingNet consists of three learnable modules ($\boldsymbol{\theta}_{\text{aff}}$, $\boldsymbol{\theta}_{\text{TPS}}$ and $\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}$), a grid generation function G, a bilinear interpolation-based image sampler ϕ, and three generated sampling grids with increased granularity, ranked as $\boldsymbol{\Omega}_{\text{r}} = G(\boldsymbol{\theta}_{\text{r}}) > \boldsymbol{\Omega}_{\text{TPS}} = G(\boldsymbol{\theta}_{\text{TPS}}) > \boldsymbol{\Omega}_{\text{aff}} = G(\boldsymbol{\theta}_{\text{aff}})$.

Specifically, $\boldsymbol{\theta}_{\text{aff}}$ is a 2×3 learnable affine matrix and it warps the input image $\tilde{\boldsymbol{x}}$ to approximate projector's front view. Similarly, $\boldsymbol{\theta}_{\text{TPS}}$ contains $(6\times6+2)\times2$ =76 learnable thin plate spline (TPS) [6] parameters and it further nonlinearly warps the output of the affine transformed image $\phi(\tilde{\boldsymbol{x}}; \boldsymbol{\Omega}_{\text{aff}})$ to exact projector's view. Unlike [18, 31], $\boldsymbol{\theta}_{\text{aff}}$ and $\boldsymbol{\theta}_{\text{TPS}}$ are directly learned without using a regression network, which is more efficient and accurate in our case.

Although TPS can approximate nonlinear smooth geometric transformations, its accuracy depends on the number of control points and the spline assumptions. Thus, it may not precisely model image deformations involved in procam imaging process. To solve this issue, we design a grid refinement CNN, *i.e.*, $\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}$ to refine the TPS sampling grid. Basically, this net learns a fine displacement for each 2D coordinate in the TPS sampling grid with a residual connection [14], giving the refined sampling grid $\boldsymbol{\Omega}_{\text{r}}$ higher precision. The advantages of our CompenNet++ over a degraded CompenNet++ without grid refinement net (named CompenNet++ w/o refine) are evidenced in Tab. 1 and Fig. 6.

Besides the novel cascaded coarse-to-fine structure with a grid refinement network, we propose a novel sampling strategy that improves WarpingNet efficiency and accuracy. Intuitively, the cascaded coarse-to-fine sampling method should sequentially sample the input $\tilde{\boldsymbol{x}}$ as

$$\mathcal{T}^{-1}(\tilde{\boldsymbol{x}}) = \phi\big(\phi(\phi(\tilde{\boldsymbol{x}}; \boldsymbol{\Omega}_{\text{aff}}); \boldsymbol{\Omega}_{\text{TPS}}); \boldsymbol{\Omega}_{\text{r}}{=}\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}(\boldsymbol{\Omega}_{\text{TPS}})\big) \tag{11}$$

However, the three bilinear interpolations above are not only computationally inefficient but also produce a blurred image. Instead, we perform the sampling in 2D coordinate space, *i.e.*, let the finer TPS grid sample the coarser affine grid, then refine the grid using $\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}$, as shown in Fig. 3. Thus, the output image is given by:

$$\mathcal{T}^{-1}(\tilde{\boldsymbol{x}}) = \phi\big(\tilde{\boldsymbol{x}}; \mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}(\phi(\boldsymbol{\Omega}_{\text{aff}}; \boldsymbol{\Omega}_{\text{TPS}}))\big) \tag{12}$$

This strategy brings two benefits: (1) only two sampling operations are required and thus is more efficient; and (2) since the image sampling is only performed once on $\tilde{\boldsymbol{x}}$, the warped image is sharper compared with using Eq. 11.

Another novelty of WarpingNet is network simplification owing to the sampling strategy above. During testing, WarpingNet is simplified essentially to a single sampling grid $\boldsymbol{\Omega}_{\text{r}}$, and geometric correction becomes a single bilinear interpolation $\mathcal{T}^{-1}(\tilde{\boldsymbol{x}}) = \phi(\tilde{\boldsymbol{x}}; \boldsymbol{\Omega}_{\text{r}})$ bringing improved testing efficiency (see Fig. 5).

CompenNet. During training, $\mathcal{F}^{\dagger}$ takes two WarpingNet transformed images as inputs, *i.e.*, a surface image $\mathcal{T}^{-1}(\tilde{\boldsymbol{s}})$

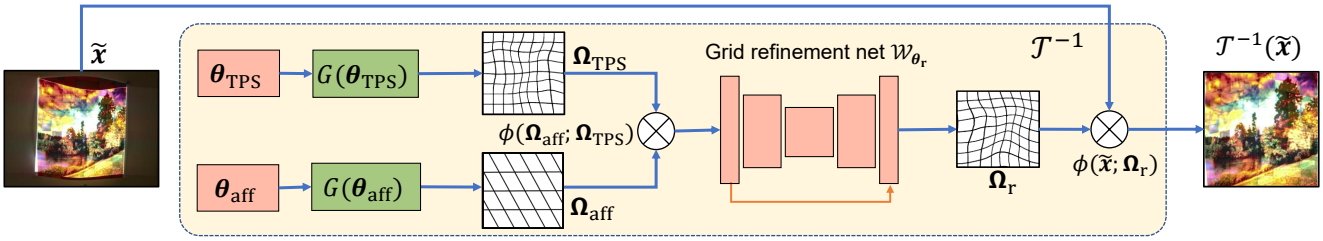

Figure 3: WarpingNet ($\mathcal{T}^{-1}$) architecture (activations layers [21, 23] omitted). It warps the input camera-captured image $\tilde{\boldsymbol{x}}$ to the projector's view using a cascaded coarse-to-fine structure. The red and green blocks are learnable parameters and grid generation functions, respectively. Operator $\otimes$ denotes bilinear interpolation, *i.e.*, $\phi(\cdot;\cdot)$. The grid refinement network $\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}$ consists of a UNet-like [32] structure, it generates a refined sampling grid that samples the input image directly.

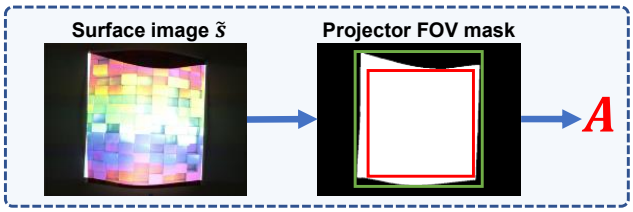

Figure 4: Projector FOV mask, bounding rectangle (green) and optimal displayable area (red). The optimal displayable area is defined as the maximum inscribed rectangle (keep aspect ratio) [29]. The affine matrix $\boldsymbol{A}$ is estimated given the displayable area and projector input image size.

and a camera-captured image $\mathcal{T}^{-1}(\tilde{\boldsymbol{x}})$. The architecture basically follows [15], but with two improvements below.

The CompenNet in [15] cannot be directly applied to our CompenNet++ with its original initialization technique, since the joint geometric and photometric process is too complex to learn. Tackling this issue, we propose some useful training techniques in §3.4.

Another improvement is that, for the testing phase, the surface feature autoencoder subset is trimmed by merging into the main backbone as biases (Fig. 5). This network simplification, together with the one on WarpingNet, largely improves the running time and memory efficiency of CompenNet++, without any sacrifice in performance quality.

3.4. Training details

Compared with CompenNet [15] training, simultaneously optimizing WarpingNet parameters $\boldsymbol{\theta}_{\mathcal{T}}$ and CompenNet parameters $\boldsymbol{\theta}_{\mathcal{F}}$ is hard without proper weights initialization and automatic data preprocessing.

Projector FOV mask. According to Eq. 6, full projector compensation's region of interest is the projector FOV, *i.e.* Fig. 1(a) blue part. Thus we can compute a projector FOV mask by automatically thresholding the camera-captured surface images with Otsu's method [26] followed by some morphological operations (Fig. 4). This mask brings threefold benefits: **(1)** masking out the pixels outside of FOV improves training stability and efficiency; **(2)** the projector FOV mask is the key to initialize WarpingNet affine weights below and **(3)** to find the optimal displayable area in §3.6.

WarpingNet weights initialization. We further improve the training efficiency by providing a task specific prior, *e.g.*, the coarse affine warping branch in WarpingNet aims to transform the input image $\tilde{\boldsymbol{x}}$ to projector's front view, as mentioned in §3.3. Thus, we initialize the affine parameters $\boldsymbol{\theta}_{\text{aff}}$ such that the projector FOV mask's bounding rectangle (Fig. 4 green rectangle) is stretched to fill the warped image. Then, $\boldsymbol{\theta}_{\text{TPS}}$ and grid refinement net $\mathcal{W}_{\boldsymbol{\theta}_{\text{r}}}$ are initialized with small random numbers at a scale of 10^{-4}, such that they generate identity mapping. These task specific initialization techniques provide a reasonably good starting point, allowing CompenNet++ to converge stably and efficiently.

CompenNet weights initialization. In [15], the CompenNet weights are randomly initialized with He's method [13] and it works well when input images are registered to projector's view offline. In our end-to-end full compensation pipeline, despite with the training techniques above, joint training WarpingNet and CompenNet may subject to suboptimal solutions, *e.g.*, the output images become plain gray. Similar to WarpingNet weights initialization, we introduce some photometric prior knowledge to improve CompenNet stability and efficiency. Inspired by traditional context-independent linear method [25], we initialize CompenNet to a simple linear channel-independent model such that:

$$\boldsymbol{\theta}_{\mathcal{F}} = \arg\min_{\boldsymbol{\theta}'_{\mathcal{F}}} \sum_i \mathcal{L}\big(\mathcal{F}^{\dagger}_{\boldsymbol{\theta}'_{\mathcal{F}}}(\boldsymbol{x}_i; \dot{\boldsymbol{s}}),\ \max(0, \boldsymbol{x}_i - \dot{\boldsymbol{s}})\big), \quad (13)$$

where $\boldsymbol{x}_i$ is a projector input image and $\dot{\boldsymbol{s}}$ is a colorful textured image that mimics the warped surface image $\mathcal{T}^{-1}(\tilde{\boldsymbol{s}})$. Compared with CompenNet's pre-train method [15], our approach creates a simple yet effective initialization without any actual projection/capture. Note this weight initialization is only performed once and independent of setups. For a new setup, $\boldsymbol{\theta}_{\mathcal{F}}$ is initialized by loading the saved weights.

3.5. Network Simplification

During testing, the structure of CompenNet++ shown in Fig. 5 is simplified from training structure (Fig. 2). **(a)** As

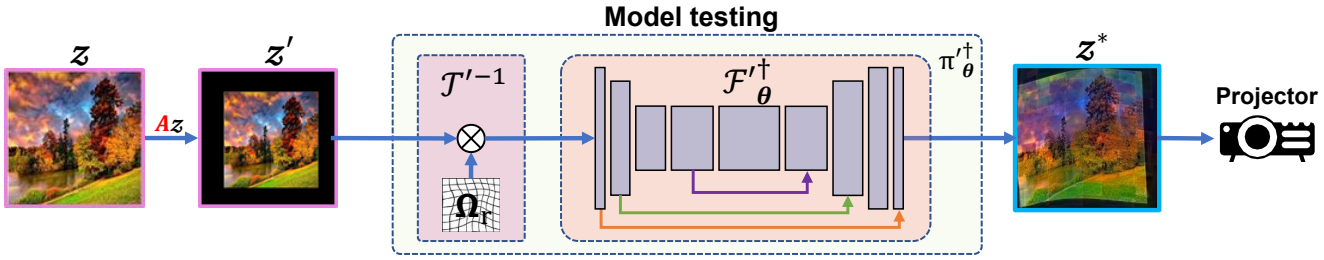

Figure 5: The testing phase of the proposed CompenNet++. Due to our novel WarpingNet structure and sampling strategy, the network is simplified to improve computational and memory efficiency. As we can see the compensation image $\boldsymbol{z}^*$ is both geometrically and photometrically compensated, such that after projection it cancels the geometric and photometric distortions and produce an image that is close to $\boldsymbol{z}'$, *i.e.* Fig. 1(e).

mentioned in §3.3, due to our novel cascaded coarse-to-fine network design and sampling strategy, WarpingNet can be substituted by a sampling grid and an image sampler shown as $\mathcal{T}'^{-1}$ in Fig. 5. **(b)** Similarly, CompenNet's surface feature extraction branch's (the top subnet of $\mathcal{F}^\dagger$) weights and input are both fixed during testing, thus, it is trimmed and replaced by biases to reduce computation and memory usage. The biases are then directly added to the CompenNet backbone, we denote this simplified CompenNet++ as $\pi'^\dagger_{\boldsymbol{\theta}}$. The two novel network simplification techniques make the proposed CompenNet++ both computationally and memory efficient with no performance drop.

3.6. Compensation pipeline

To summarize, our full projector compensation pipeline consists of three major steps (Fig. 2 and Fig. 5). **(1)** We start by projecting a plain gray image $\boldsymbol{x}_0$, and N sampling images $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ to the projection surface and capture them using the camera, and denote the captured images as $\tilde{\boldsymbol{s}}$ and $\tilde{\boldsymbol{x}}_i$, respectively. **(2)** We gather the N image pairs $(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_i)$ and $\tilde{\boldsymbol{s}}$ to train the compensation model $\pi^\dagger_{\boldsymbol{\theta}} = \{\mathcal{F}^\dagger_{\boldsymbol{\theta}}, \mathcal{T}_{\boldsymbol{\theta}}^{-1}\}$ end-to-end. **(3)** As shown in Fig. 5, we simplify the trained CompenNet++ to $\pi'^\dagger_{\boldsymbol{\theta}}$ using techniques in §3.5. Finally, for an ideal desired viewer perceived image $\boldsymbol{z}$, we generate its compensation image $\boldsymbol{z}^*$ and project $\boldsymbol{z}^*$ to the surface.

In practice, $\boldsymbol{z}$ is restricted to the surface displayable area. Similar to [29], we find an optimal desired image $\boldsymbol{z}' = \boldsymbol{A}\boldsymbol{z}$, where $\boldsymbol{A}$ is a 2D affine transformation that uniformly scales and translates the ideal perceived image $\boldsymbol{z}$ to optimally fit the projector FOV as shown in Fig. 4 and Fig. 5.

3.7. System configuration and implementation.

Our projector compensation system consists of a Canon 6D camera and a ViewSonic PJD7828HDL DLP projector with resolutions set to 640×480 and 800×600, respectively. In addition, an Elgato Cam Link 4K video capture card is connected to the camera to improve frame capturing efficiency (about 360ms per frame).

The distance between the camera and the projector is varied in the range of 500mm to 1,000mm and the projection surface is around 1,000mm in front of the pro-cam pair. The camera exposure, focus and white balance modes are set to manual, the global lighting is varied for each setup but fixed during each setup's data capturing and system testing.

CompenNet++ is implemented using PyTorch [27] and trained using Adam optimizer [19] with a penalty factor of 10^{-4}. The initial learning rate is set to 10^{-3} and decayed by a factor of 5 at the 1,000$^{\text{th}}$ iteration. The model weights are initialized using the techniques in §3.4. We train the model for 1,500 iterations on three Nvidia GeForce 1080Ti GPUs with a batch size of 48, and it takes about 15min to finish.

3.8. Dataset and evaluation protocol

Following [15], we prepare 700 colorful textured images and use $N = 500$ for each training set $\mathcal{X}_k$ and $M = 200$ for each validation set $\mathcal{Y}_k$. In total $K = 20$ different setups are prepared for training and evaluation, each setup has a nonplanar textured surface.

We collect the setup-independent validation set of M samples as $\mathcal{Y} = \{(\tilde{\boldsymbol{y}}_i, \boldsymbol{y}_i)\}_{i=1}^M$, under the same system setup as the training set $\mathcal{X}$. Then the algorithm performance is measured by averaging over similarities between each validation input image $\boldsymbol{y}_i$ and its algorithm output $\hat{\boldsymbol{y}}_i = \pi^\dagger_{\boldsymbol{\theta}}(\tilde{\boldsymbol{y}}_i; \tilde{\boldsymbol{s}})$ and reported in Tab. 1. Note we use the same evaluation metrics PSNR, RMSE and SSIM as in [15].

4. Experimental Evaluations

4.1. Comparison with state-of-the-arts

We compare the proposed full compensation method (*i.e.* CompenNet++) with four two-step baselines, a context-independent TPS[4] model [10], an improved TPS model (explained below), a Pix2pix [17] model and a CompenNet [15] model on the evaluation benchmark.

To fairly compare two-step methods, we use the same SL warping for geometric correction. We first project 42 SL patterns to establish pro-cam pixel-to-pixel mapping using the approach in [22], the mapping coordinates are then

[4]Not geometric correction [6], instead using TPS to model pixel-wise *photometric transfer function*.

Table 1: Quantitative comparison of compensation algorithms. Results are averaged over $K = 20$ different setups. The top-3 results of each column in each **#Train** section are highlighted as red, green and blue, respectively. Note the metrics for uncompensated images are PSNR=9.5973, RMSE=0.5765 and SSIM=0.0767. The metrics for the original TPS [10] w/ SL (#Train=125) are PSNR=16.7271, RMSE= 0.2549 and SSIM=0.5207.

Model	#Train=48			#Train=125			#Train=250			#Train=500		
	PSNR↑	RMSE↓	SSIM↑	PSNR↑	RMSE↓	SSIM↑	PSNR↑	RMSE↓	SSIM↑	PSNR↑	RMSE↓	SSIM↑
TPS [10] textured w/ SL	18.0297	0.2199	0.5390	18.0132	0.2205	0.5687	18.0080	0.2206	0.5787	17.9746	0.2215	0.5830
Pix2pix [17] w/ SL	17.7160	0.2271	0.5068	17.1141	0.2468	0.5592	16.5236	0.2669	0.5763	19.4160	0.1903	0.6196
CompenNet [15] w/ SL	20.2023	0.1722	0.6690	20.7684	0.1609	0.7022	20.8347	0.1596	0.7142	20.9552	0.1573	0.7117
CompenNet++ w/o refine	19.4139	0.1909	0.6252	20.6061	0.1635	0.6958	20.7307	0.1613	0.7106	20.9172	0.1577	0.7113
CompenNet++	19.8552	0.1781	0.6637	20.7947	0.1598	0.7116	20.8959	0.1581	0.7227	21.1127	0.1540	0.7269
CompenNet++ fast	19.9696	0.1760	0.6699	20.5171	0.1650	0.7001	20.5795	0.1638	0.7063	20.6711	0.1622	0.7081
CompenNet++ faster	19.2536	0.1912	0.6249	19.5309	0.1844	0.6546	19.7212	0.1806	0.6613	19.6989	0.1811	0.6574

Figure 6: Qualitative comparison of TPS [10] w/ SL, TPS textured w/ SL, Pix2pix [17] w/ SL, CompenNet [15] w/ SL, proposed CompenNet++ w/o refine and proposed CompenNet++ on two different surfaces. The 1st to 3rd columns are the camera-captured projection surface, desired viewer perceived image and camera-captured uncompensated projection, respectively. The rest columns are the compensation results of different methods. Each image is provided with two zoomed-in patches for detailed comparison. More comparisons are provided in supplementary materials.

bilinear-interpolated to fill missing correspondences. Afterwards, we capture 125 pairs of plain color sampling image as used in the original TPS method [10] for photometric compensation, we warp the sampling image to projector's view using SL and name this method **TPS w/ SL**. We also fit the TPS method using SL-warped diverse textured training set $\mathcal{X}_k$, and name this method **TPS textured w/ SL**.

The experiment results in Tab. 1 and Fig. 6 show clear improvement of TPS textured over the original TPS method. Our explanations are **(a)** compared with plain color images,

the textured training images and validation/testing images share a more similar distribution. **(b)** Although original TPS method uses 5^3 plain color images, each projector pixel's R/G/B channel only has five different intensity levels, training the TPS model using these samples may lead to a suboptimal solution. While our colorful textured samples evenly cover the RGB space at each projector pixel, resulting a more faithful sampling of the photometric transfer function.

To demonstrate the difficulty of full compensation problem, we compare with a deep learning-based image-to-image translation model Pix2pix[5] [17] trained on the same SL-warped $\mathcal{X}_k$ as TPS textured w/ SL, we name it **Pix2pix w/ SL**. We use the same adaptation as [15], except that Pix2pix is trained for 12,000 iterations to match the training time of our model. The results show that the proposed CompenNet++ outperforms Pix2pix w/ SL, demonstrating that the full compensation problem cannot be well solved by a general deep-learning based image-to-image translation model.

We then compare our method with the partial compensation model CompenNet [15], we train it with the same SL-warped training set $\mathcal{X}_k$ as TPS textured w/ SL and Pix2pix w/ SL, and name this two-step method **CompenNet w/ SL**. The quantitative and qualitative comparisons are shown in Tab. 1 and Fig. 6, respectively.

4.2. Effectiveness of the proposed CompenNet++

Tab. 1 clearly shows that CompenNet++ outperforms other two-step methods. This indicates that **(a)** even without building pixel-to-pixel mapping using SL, the geometry correction can be learned directly from the photometric sampling images. **(b)** Solving full compensation problem separately may lead to suboptimal solution and the two steps should be solved jointly, as proposed by CompenNet++. **(c)** Besides outperforming CompenNet w/ SL, we use 42 less images than two-step SL-based method.

We explain why two-step methods may find suboptimal solution in Fig. 6, where SL decoding errors affect the photometric compensation accuracy. As shown in the 1st row red zoomed-in patches, compared with end-to-end methods (last two columns), SL-based two-step methods (4th-7th columns) produce curved edges, due to inaccurate SL warping. Furthermore, in the 3rd and 4th rows, the non-planar surface is behind a glass with challenging specular reflection. Comparing the two groups, specifically the blue zoomed-in patches, we see unfaithful compensations by the SL-based two-step methods, whereas, end-to-end methods **CompenNet++ w/o refine** and **CompenNet++** show finer geometry, color and details. This is because SL suffers from decoding errors due to specular reflection and creates false mappings, then the mapping errors propagate to the photometric compensation stage. This issue is better addressed by the proposed CompenNet++, where global geometry and photometry information is considered in full compensation. In summary, CompenNet++ not only brings improved performance than two-step SL-based methods, but also waives 42 extra SL projections/captures, and meanwhile being insensitive to specular highlights.

[5] https://github.com/junyanz/pytorch-CycleGAN-and-Pix2pix

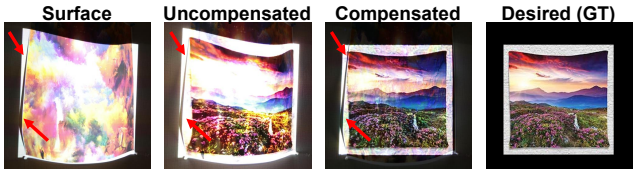

Figure 7: Failed example. CompenNet++ is unable to compensate self-occlusion regions as pointed by red arrows.

To demonstrate the practicability of CompenNet++ when efficiency is preferred over quality, *i.e.*, less data and shorter training time, we train CompenNet++ using only 48 images and reduce the training iterations to 1,000/500 and batch size to 24/16, we name the efficient methods **CompenNet++ fast/faster** and it takes only 5min/2.5min to finish training. As shown in Tab. 1, even when trained with only 48 images, **CompenNet++ fast/faster** still outperform TPS textured w/ SL and Pix2pix w/ SL trained with 500 images on SSIM.

4.3. Effectiveness of the grid refinement network

To demonstrate the effectiveness of the sampling grid refinement network $\mathcal{W}_{\theta_r}$ (Eq. 12 and Fig. 3), we create a degraded CompenNet++ by removing $\mathcal{W}_{\theta_r}$, and name it **CompenNet++ w/o refine**. As reported in Tab. 1, CompenNet++ clearly outperforms this degraded model, showing the effectiveness of the grid refinement network $\mathcal{W}_{\theta_r}$.

5. Conclusions and Limitations

In this paper, we extend the partial projector compensation model CompenNet to a full compensation pipeline named CompenNet++. With the novel cascaded coarse-to-fine WarpingNet, task specific training and efficient testing strategies, CompenNet++ provides the first end-to-end simultaneous projector geometric correction and photometric compensation. The effectiveness of our formulation and architecture is verified by comprehensive evaluations. The results show that our end-to-end full compensation outperforms state-of-the-art two-step methods both qualitatively and quantitatively.

Limitations. We assume each single patch of the projection surface can be illuminated by the projector. That said, CompenNet++ may not work well on complex surfaces with self-occlusion (Fig. 7). One potential solution is to use multiple projectors covering each other's blind spots. In fact, extending the end-to-end full compensation framework to multiple projectors is an interesting future direction.

References

[1] Daniel G Aliaga, Yu Hong Yeung, Alvin Law, Behzad Sajadi, and Aditi Majumder. Fast high-resolution appearance editing using superimposed projections. *ACM Tran. on Graphics*, 2012. 1, 2

[2] Hirotaka Asayama, Daisuke Iwai, and Kosuke Sato. Fabricating diminishable visual markers for geometric registration in projection mapping. *IEEE TVCG*, 2018. 1, 2

[3] Mark Ashdown, Takahiro Okabe, Imari Sato, and Yoichi Sato. Robust content-dependent photometric projector compensation. In *CVPRW PROCAMS*, 2006. 1, 2

[4] Oliver Bimber, Andreas Emmerling, and Thomas Klemmer. Embedded entertainment with smart projectors. *Computer*, 2005. 1, 2

[5] Ameneh Boroomand, Hicham Sekkati, Mark Lamm, David A Clausi, and Alexander Wong. Saliency-guided projection geometric correction using a projector-camera system. In *ICIP*, 2016. 1, 2

[6] Gianluca Donato and Serge Belongie. Approximate thin plate spline mappings. In *ECCV*, 2002. 4, 6

[7] Kensaku Fujii, Michael D Grossberg, and Shree K Nayar. A projector-camera system with real-time photometric adaptation for dynamic environments. In *CVPR*, 2005. 2

[8] Jason Geng. Structured-light 3D surface imaging: a tutorial. *Advances in Optics and Photonics*, 2011. 1

[9] Michael D Grossberg, Harish Peri, Shree K Nayar, and Peter N Belhumeur. Making one object look like another: controlling appearance using a projector-camera system. In *CVPR*, 2004. 1, 2

[10] Anselm Grundhöfer and Daisuke Iwai. Robust, error-tolerant photometric projector compensation. *IEEE TIP*, 2015. 1, 2, 6, 7

[11] Anselm Grundhöfer and Daisuke Iwai. Recent advances in projection mapping algorithms, hardware and applications. In *Computer Graphics Forum*. Wiley Online Library, 2018. 1, 3

[12] Michael Harville, Bruce Culbertson, Irwin Sobel, Dan Gelb, Andrew Fitzhugh, and Donald Tanguay. Practical methods for geometric and photometric correction of tiled projector. In *CVPRW*, 2006. 1, 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 5

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[15] Bingyao Huang and Haibin Ling. End-to-end projector photometric compensation. In *CVPR*, 2019. 1, 2, 3, 4, 5, 6, 7, 8

[16] Bingyao Huang, Samed Ozdemir, Ying Tang, Chunyuan Liao, and Haibin Ling. A single-shot-per-pose camera-projector calibration system for imperfect planar targets. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 15–20. IEEE, 2018. 1

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 6, 7, 8

[18] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015. 4

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[20] Yuqi Li, Aditi Majumder, Meenakshisundaram Gopi, Chong Wang, and Jieyu Zhao. Practical radiometric compensation for projection display on textured surfaces using a multidimensional model. In *Computer Graphics Forum*. Wiley Online Library, 2018. 2

[21] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 5

[22] Daniel Moreno and Gabriel Taubin. Simple, accurate, and robust projector-camera calibration. In *3DIMPVT*, 2012. 1, 6

[23] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 5

[24] Gaku Narita, Yoshihiro Watanabe, and Masatoshi Ishikawa. Dynamic projection mapping onto deforming non-rigid surface using deformable dot cluster marker. *IEEE TVCG*, 2017. 1, 2

[25] Shree K Nayar, Harish Peri, Michael D Grossberg, and Peter N Belhumeur. A projection system with radiometric compensation for screen imperfections. In *ICCVW PROCAMS*, volume 3, 2003. 2, 5

[26] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE T-SMC*, 1979. 5

[27] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 6

[28] Ramesh Raskar and Paul Beardsley. A self-correcting projector. In *CVPR*, 2001. 1, 2

[29] Ramesh Raskar, Jeroen Van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. ilamps: geometrically aware and self-configuring projectors. *ACM Tran. on Graphics*, 2003. 1, 2, 5, 6

[30] Ramesh Raskar, Greg Welch, Kok-Lim Low, and Deepak Bandyopadhyay. Shader lamps: Animating real objects with image-based illumination. In *Rendering Techniques*. Springer, 2001. 1, 2

[31] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. In *CVPR*, 2017. 4

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 2015. 5

[33] Behzad Sajadi, Maxim Lazarov, and Aditi Majumder. Adict: accurate direct and inverse color transformation. In *ECCV*. Springer, 2010. 2

[34] Marjan Shahpaski, Luis Ricardo Sapaico, Gaspard Chevassus, and Sabine Susstrunk. Simultaneous geometric and radiometric calibration of a projector-camera pair. In *CVPR*, 2017. 2

[35] Christian Siegl, Matteo Colaianni, Marc Stamminger, and Frank Bauer. Adaptive stray-light compensation in dynamic multi-projection mapping. *Computational Visual Media*, 2017. 1, 2

[36] Christian Siegl, Matteo Colaianni, Lucas Thies, Justus Thies, Michael Zollhöfer, Shahram Izadi, Marc Stamminger, and Frank Bauer. Real-time pixel luminance optimization for dynamic multi-projection mapping. *ACM Tran. on Graphics*, 2015. 1, 2

[37] Shoichi Takeda, Daisuke Iwai, and Kosuke Sato. Inter-reflection compensation of immersive projection display by spatio-temporal screen reflectance modulation. *IEEE TVCG*, 2016. 2

[38] Jean-Philippe Tardif, Sébastien Roy, and Martin Trudeau. Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction. In *3DIM*, 2003. 1, 2

[39] Takenobu Yoshida, Chinatsu Horii, and Kosuke Sato. A virtual color reconstruction system for real heritage with light projection. In *Proceedings of International Conference on Virtual Systems and Multimedia*, volume 3, 2003. 1, 2

[40] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE TCI*, 2017. 4