Journal of Imaging Science and Technology® 66(6): 060401-1–060401-14, 2022.
© Society for Imaging Science and Technology 2022

Array Camera Image Fusion using Physics-Aware Transformers

Qian Huang, Minghao Hu, and David J. Brady[▲]
Wyant College of Optical Sciences, University of Arizona, Tucson, AZ 85721
E-mail: djbrady@arizona.edu

Abstract. *We demonstrate a physics-aware transformer for feature-based data fusion from cameras with diverse resolution, color spaces, focal planes, focal lengths, and exposure. We also demonstrate a scalable solution for synthetic training data generation for the transformer using open-source computer graphics software. We demonstrate image synthesis on arrays with diverse spectral responses, instantaneous field of view and frame rate.* © *2022 Society for Imaging Science and Technology.*
[DOI: 10.2352/J.ImagingSci.Technol.2022.66.6.060401]

1. INTRODUCTION

In contrast with systems that use physical optics to form images, computational imaging uses physical processing to code measurements but relies on electronic processing for image formation [1]. In optical systems, computational imaging enables “light field cameras [2]” that capture high dimensional spatio-spectral-temporal data cubes. The primary challenge of light field camera design is that, while the light field is 3, 4 or 5 dimensional, measurements still rely on 2D photodetector arrays. High dimensional light field capture on 2D detectors can be achieved using three sampling approaches: interleaved coding, temporal coding and multiaperture coding. Interleaved coding, as is famously done with color filter arrays [3], consists of enabling adjacent pixels on the 2D plane to access different parts of the light field. Mathematically similar sampling strategies for depth of field are implemented in integral imaging [4] and plenoptic cameras [5]. This interleaved approach generalizes to arbitrary high dimensional data cubes in the context of snapshot compressive imaging [6]. Temporal coding consists of scanning the spectral [7] or focal [8] response of the camera during recording.

Array cameras [9] offer many potential advantages over interleaved and temporal coding. The advantage over temporal scanning is obvious, a camera array can capture snapshot light fields and does not therefore sacrifice temporal resolution. Additionally, development of cameras with dynamic spectral, spatial and focal sampling is more challenging than development of array components that sample slices of the data cube. The advantage of multiaperture cameras relative to interleaved sampling is more subtle, although implementation of interleaved sampling is also physically challenging. On a deeper level, however, interleaved sampling makes the physically implausible assumption that temporal sampling rates and exposure should be the same for different regions of the light field. In practice, photon flux in the blue is often very different from in the red and setting these channels to a common exposure level is injudicious. The design of lenses and sensors optimized for specific spectral and focal ranges leads to higher quality data.

With these advantages in mind, many studies have previously considered array cameras for computational imaging [10–12]. More recently, artificial neural networks have found extensive application in array camera control and image processing [13, 14]. Of course, biological imaging systems rely heavily on array imaging solutions. While conventional array cameras originally relied on image-based registration [15] for “stitching”, biological systems integrate multiaperture data deep in the visual cortex. In analogy with the biological system approach, here we demonstrate that array camera image fusion from deep layer features, rather than pixel maps, is effective in data fusion from diverse camera arrays. Our approach is based on transformer [16, 17] networks, which excel at establishing long-range connections and integrating related features.

Since transformer networks are more densely connected than convolutional networks, high computational costs have inhibited their use in common computer vision tasks. Nonlocal neural connections drastically increase the receptive field for each feature element. As shown here, however, when the transformer is integrated with the physics of the system, the connections outside of the physical receptive fields can be trimmed to the extent that the complexity of transformers is comparable to convolutional networks.

There are three main branches of combining physical models with neural algorithms. First, plugging a learned model as a prior into a physical model, which is also known as “plug and play”. RED [18] is an example that applied a denoiser as its prior. The second way initiated by deep image prior [19] is using a network architecture as a prior, thus removing the requirement of pretraining. The methods above, however, are optimized for a scene in a loop, restricting real-time applications. The third method is integrating the physics of the system with the neural algorithm. The physics of the system can take the form of, for example,

[▲] IS&T Member.

Received June 25, 2022; accepted for publication Oct. 5, 2022; published online Dec. 1, 2022. Associate Editor: Zeev Zalevsky.

1062-3701/2022/66(6)/060401/14/$25.00

a parameterized input to the algorithm (e.g., the noise level in a denoising system [20]) or a sub-module in the algorithm architecture (e.g., the spatial transformer [21]). In a camera array, the intrinsics and extrinsics are usually exploited. Using them to characterize an array has several advantages: (1) they are not likely to change once the cameras are encapsulated; (2) their derivatives like the epipolar geometry naturally build connections of sensor pixels; (3) they can be achieved by mature calibration techniques like [22] with efficiency. The epipolar transformer [23] is an example that leverages the epipolar geometry to estimate the human pose. Within the field of image fusion, the parallax-aware attention model [24] was introduced to derive a high-resolution image from two rectified low-resolution images. The model has been extended to process unrectified pairs [25] and to solve general image restoration tasks from homogeneous views [26]. Our algorithm is also inspired by this architecture but focuses on general fusion tasks in the camera array.

As with many physical image capture and processing tasks, the forward model for array camera imaging is easily simulated but the inverse problem is difficult to describe analytically. This class of problems can be addressed by training neural processors on synthetic data. Synthetic data is highly effective in training imaging systems with well-characterized forward models. Datasets that include synthetic data can be semi-virtual, containing synthetic labels from real media of high quality such as the DND denoising benchmark [27], the Flickr1024 stereo super-resolution dataset [28], and the KITTI2012 multitask vision benchmark [29]. On the other hand, datasets can be completely virtual from source to sensors. Among those, the MPI-Sintel Dataset [30] is one of the milestones that use CG software to generate data. It contains rich scenes and incredible labeling accuracy of the optical flow, depth, and segmentation, which are either not achievable or expensive to generate in real scenes. Its successors include FlyingChairs [31] and Scene Flow Datasets [32]. As CG software keeps evolving, we see more fancy features being developed and integrated into handy packages like BlenderProc [33]. In addition, recent achievements in render engines like GPU-aided ray tracing allow us to realistically, accurately and efficiently model the world and render the modeled world to sensors of ideal virtual cameras. The rendered frames can be regarded as the ground truth. Synthetic sensor data that is degraded can be generated from the ground truth via the forward model of the camera array.

Surprisingly, it is unnecessary to build photorealistic real-world scenes for some computer vision problems. This finding was implied by [31, 32] where unnatural synthetic data yielded advanced optical flow estimation results. Image fusion problems likewise focus on low-level features like color and texture, while are less concerned with high-level features modeling physical interactions or semantic information that contribute to naturality. Also, photorealistic rendering intentionally introduces aberrations, distortion, blur and other defects to resemble the performance of existent optical components and detectors. This add-on feature increases the computational cost but is unwanted for neural fusion algorithms that require ground truth labels of high quality and sometimes beyond the physical limits. Hence scenes with abstract objects native to computer graphics software with diverse colors and textures can span the problem domain of image fusion. Along with the programming interface provided by Blender (http://www.blender.org), this data synthesis pipeline can be easily deployed and automatically scaled up to fit diverse data demands. With the assistance of better synthetic data, we can expect networks of better performance to be easily deployed.

Here, we use this approach to build a physics-aware transformer (PAT) network that can fuse data from the array cameras of the diverse resolution, color spaces, focal planes, focal lengths, and exposure. The purpose of this system is to combine data from array cameras to return a computed image superior to the image available to any single camera. Array cameras are designed in these systems to exploit differences in the spatial and temporal resolution needed to capture color, texture and motion.

We demonstrate four example systems. The first system combines data from wide field color cameras with narrow field monochrome cameras, the second combines color information from visible cameras with the textural information from near infrared cameras, the third combines short exposure monochrome imagery with long exposure narrow spectral band data and the fourth combines high frame rate monochrome data with low frame rate color images. As a group, these designs combine with PAT processing to show that array cameras optimized for effective data capture can create virtual cameras with radically improved dynamic range, color fidelity and spatial/temporal resolution.

2. PROPOSED METHOD

The goal of PAT is to fuse data from cameras in an array. The fusion result reflects the viewpoint of one selected camera. The selected viewpoint is the alpha viewpoint α while others are alternative beta viewpoints $\beta_1 \sim \beta_m$, where m is the number of alternative viewpoints. To represent images, features, or parameters from a certain viewpoint, the viewpoint symbol is marked as the left superscript.

The architecture of PAT is illustrated in Figure 1. The workflow of PAT is (1) each sensor image goes through Fig. 1(a) to generate its corresponding feature; (2) features are fed into Fig. 1(b) to generate the final output. As images may differ in resolution, color spaces, focal planes, etc., proper representations are learned to facilitate correspondence matching. We adopt the residual atrous-spatial pyramid pooling (ASPP) module [24] to fulfill this task, which demonstrates effectiveness to generate multiscale features. For each camera, the feature produced by the image representation module shares the spatial dimensions with its sensor image, thus the position of each pixel is inherently encoded to the indices of voxels. Here a “voxel” denotes a D-length vector along the feature dimension, as illustrated in Figure 2. In this sense the epipolar

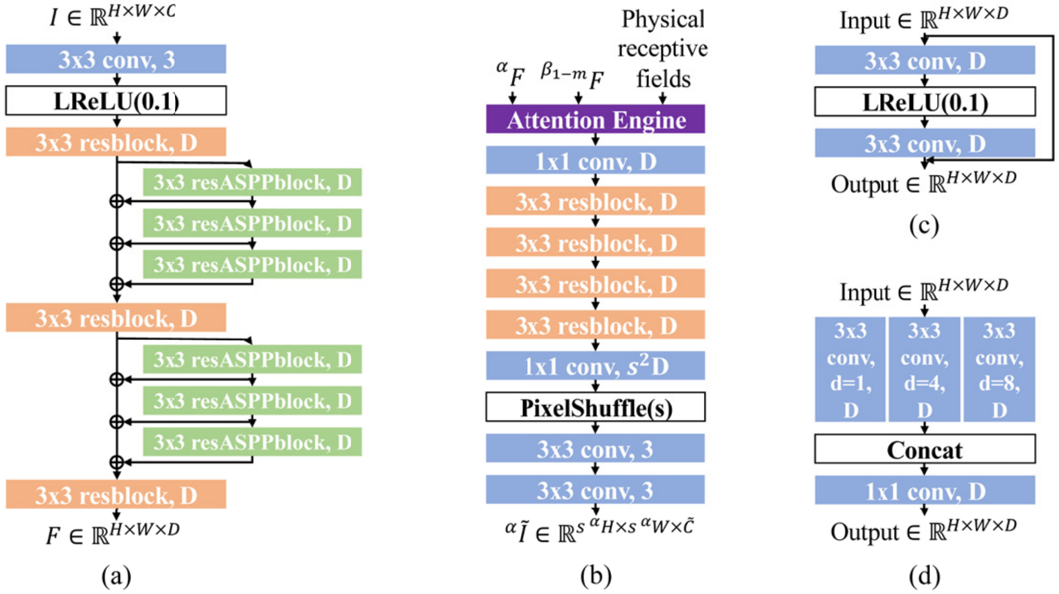

Figure 1. The architecture of physics-aware transformer (PAT). "$a \times a$ conv" are convolutional layers with kernels of $a \times a$ size. The stride of convolutional layers is 1 and the padding is $(a-1)/2$. The dilation of convolutional layers d is 1, unless specified. The depth dimension of convolutional layers and residual blocks are notated after commas. The parameter each hollow block may take is given in parenthesis. (a) Image representation module. It takes sensor image I and produces the associate feature F. $\oplus$ denotes addition. (b) The attention engine and post-fusion module. ${}^{\alpha}F$ is the feature of ${}^{\alpha}I$ from alpha viewpoint and ${}^{\beta}F$s are features of ${}^{\beta}I$s from alternative viewpoints. s is the upscaling factor. The output is the fusion result ${}^{\alpha}\tilde{I}$. (c) The architecture of "3×3 resblock". (d) The architecture of "3×3 resASPPblock".

geometry and other physical priori in the pixel domain are expected to work in the feature domain.

It is to be noted that we do not require the resolution of all the cameras in the array to be the same, thus features may have diverse spatial dimensions but share the third dimension D. The representation modules of different input frames share the weights. The attention engine, elaborated shortly, is where we process the image representations given the input physical information. The processed feature U goes through several convolutional layers and residual blocks to produce the image output, which reflects the alpha viewpoint but with information from the beta view.

2.1 *Attention Engine*

The attention engine densely aligns the features with regard to the input physical receptive fields. The attention engine starts from image representations (features) of sensor images. We apply dot-product attention to compare and transfer the alternative features. We perform $\mathbf{C}^3$ operations: **Collect**, **Correlate**, and **Combine** to generate the feature output. For simplicity, we use a system with two viewpoints to illustrate $\mathbf{C}^3$ operations with the receptive fields following the epipolar geometry, as shown in Figure 3. From the feature ${}^{\alpha}F \in \mathbb{R}^{{}^{\alpha}H\times{}^{\alpha}W\times D}$ of the alpha camera, we produce a query feature $Q \in \mathbb{R}^{{}^{\alpha}H\times{}^{\alpha}W\times D}$ through a residual block and a convolutional layer. Similarly, each alternative feature ${}^{\beta}F \in \mathbb{R}^{{}^{\beta}H\times{}^{\beta}W\times D}$ produces a key feature $K \in \mathbb{R}^{{}^{\beta}H\times{}^{\beta}W\times D}$ and a value feature $V \in \mathbb{R}^{{}^{\beta}H\times{}^{\beta}W\times D}$.

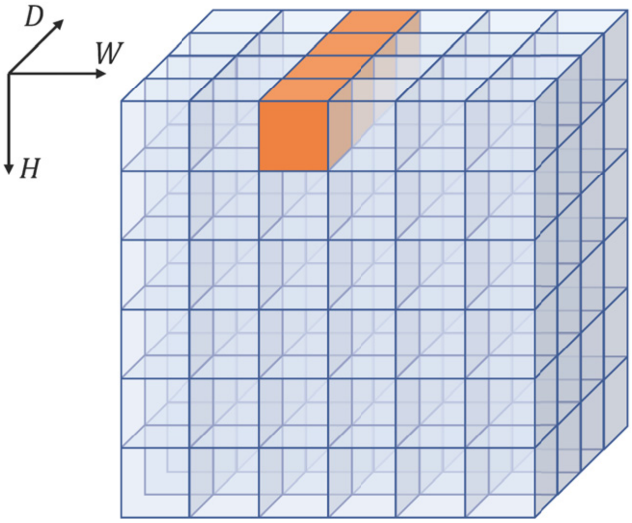

Figure 2. A feature (translucent cube) and one of its voxels (solid stick).

Collect: $q_j \in \mathbb{R}^D$ is jth voxel in Q. The range of j is from 1 to ${}^{\alpha}H \times {}^{\alpha}W$. Voxels $\{k_{j_1}, k_{j_2}, k_{j_3}, \dots, k_{j_n}\}$, $k \in \mathbb{R}^D$ in K and voxels $\{v_{j_1}, v_{j_2}, v_{j_3}, \dots, v_{j_n}\}$, $v \in \mathbb{R}^D$ are selected along the epipolar line of q_j. In other words, $j_1, j_2, j_3, \dots, j_n$ are top-n closest locations to the epipolar line of location j. n is predefined in practice depending on the spatial dimensions of the beta view.

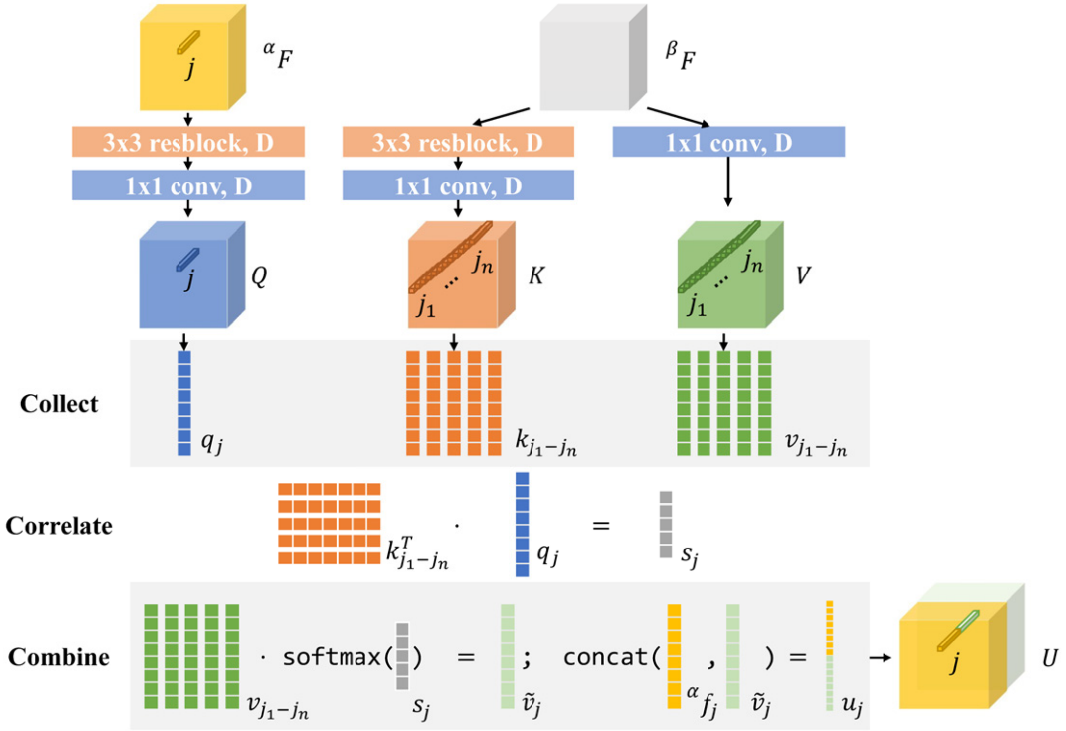

Figure 3. Workflow of the attention engine and C^3 operations in a dual-camera array with the awareness of the epipolar geometry. We use cubes to represent features and bars to represent voxels. "·" is the symbol of matrix multiplication.

Correlate: We calculate the score s_j between $^{\alpha}q_j$ and extracted ks to find the correspondence. s_j is equal to the dot product of $\left[k^T_{j_1}; k^T_{j_2}; k^T_{j_3}; \ldots; k^T_{j_n}\right]$ and $^{\alpha}q_j$.

Combine: We combine voxels $\{v_{j_1}, v_{j_2}, v_{j_3}, \ldots, v_{j_n}\}$ with regard to s_j by calculating the dot product of $\left[v_{j_1}, v_{j_2}, v_{j_3}, \ldots, v_{j_n}\right]$ and $\texttt{softmax}(s_j)$, where $\texttt{softmax}(\cdot)$ is the softmax function. We denote the combined voxel as $\tilde{v}_j$. The concatenation of $^{\alpha}f_j$ and $\tilde{v}_j$ is the jth voxel u_j of the output feature $U \in \mathbb{R}^{^{\alpha}H \times ^{\alpha}W \times 2D}$.

For more than one alternative viewpoints, the jth output vector u_j is equal to $\texttt{concat}$ $(^{\alpha}f_j, {}^{\beta_1}\tilde{v}_j, {}^{\beta_2}\tilde{v}_j, \ldots, {}^{\beta_m}\tilde{v}_j)$, for $\texttt{concat}(\cdot)$ is the concatenate function. **C**^{3} operations are fully vectorized, thus deployment of the attention engine on trending deep learning platforms is for convenience.

One may notice the attention engine reduces to PAM [24] when $m = 1, {}^{\alpha}H = {}^{\beta}H, {}^{\alpha}W = {}^{\beta}W, C = 3, D = 64$, and each receptive field follows the epipolar geometry in a rectified stereo pair, except that the intercorrelated validation mask is not incorporated into U. In comparison, PAT can process images of different characteristics from three or more cameras, where rectification cannot be performed. Moreover, PAT can integrate other physical clues, like maximum disparity or homography-based approximation, to optimize computation. In **Collect** for example, if we roughly estimate the correspondence of q_j using geometry transformation, the epipolar line can be truncated with regard to the maximum displacement, which is based on the depth distribution of the scene. The diagram of this process is shown in Figure 4.

2.2 *Complexity Analysis*

It is essential to ensure that the above operation are achievable and efficient with regard to time complexity. For j is the voxel index of the output feature U, let us assume $L = \max_j |\{k_{j_1}, k_{j_2}, \ldots, k_{j_n}\}|$, where $| \cdot |$ returns the size of the set. The complexity of **Collect** is $O(L)$, of **Correlate** is $O(D \times L)$, and of **Combine** is $O(D \times L)$. Hence overall to compute the entire output feature for m alternative viewpoints, we have the complexity $O(m \times H \times W \times D \times L)$.

For example, we assume we know the intrinsics and extrinsics in a dual-camera array, where $m = 1$ and the resolution of the cameras is $H \times W$. We can specify the physical receptive field in the attention engine to be the indices of beta feature voxels along the epipolar line of each alpha feature voxel, assuming the feature representations of images also follow the epipolar geometry in the spatial dimensions. In this case, the total time complexity is $O(H \times W \times D \times L)$, where L is linear to $H + W$. In comparison, the time complexity of a single convolutional layer is $O(H \times W \times D \times N_{conv})$, for N_{conv} is the number of elements in the convolutional kernel. We can see two complexities are basically the same up to a scale under big O notation. Furthermore, we can incorporate homography-based approximation, where we roughly locate associate voxels of each alpha voxel via perspective transformation.

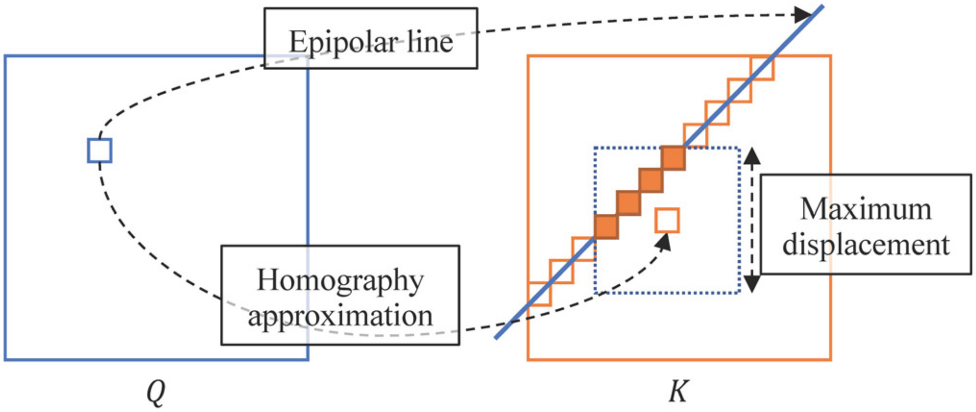

Figure 4. Incorporation with other physical clues in **Collect**. For conciseness we show Q and K as big squares with regard to the spatial dimensions and associate voxels as small squares. When all the clues are considered, only the voxels indicated by solid small squares that reside along the epipolar line and inside the dotted window are selected from K for the next **Correlate**.

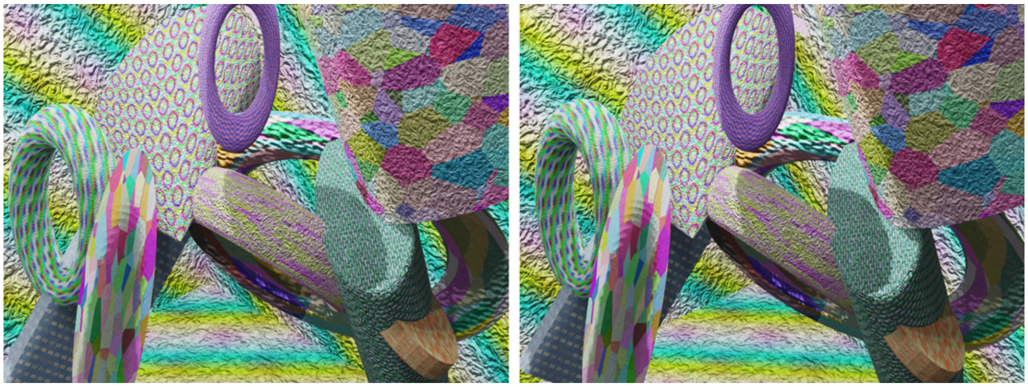

Figure 5. An example of the views of a dual camera system. The resolution of the images is 2048 × 1536.

Knowing the depth range of interest in a scene, we can set the maximum pixel displacement l between the rough estimates and exact correspondences to truncate the epipolar lines. Thus the complexity can be further reduced to $O(H \times W \times D \times l)$, as typically $l \ll W$.

Here, we can see another merit the attention engine carries in terms of time complexity. We can chop the alpha feature into patches with spatial resolution $H_p \times W_p$. The attention engine infers on those patches in parallel, bringing the time complexity down to $O(m \times H_p \times W_p \times D \times L)$.

2.3 *Data Synthesis*

As mentioned earlier, the pipeline of data simulation is fully automatic. We use the Python API of Blender to scale up the generation of scenes. Blender provides a variety of meshes, from which we select several representative meshes including “plane”, “cube”, and “uv-sphere”, and enrich the database by perturbing the surface to create diverse ridges and valleys. We can specify the dimensions, locations, and rotations of the meshes to diversify their distribution in a scene. Upon the creation of each shape, we can attach the “material” attribute to customize its interaction with the light source. There are dozens of knobs to adjust the base color, diffusion, or specularity; apart from those, we can apply vectorized textures, e.g., brick texture and checkerboard texture, to add varieties to color distribution on the mesh. Occlusion and shadows are naturally introduced while stacking up the meshes.

Blender provides camera objects to render the scene. Just as in real cameras, parameters like the focal length, sensor size, pixel pitch, and resolution can be easily set. If the “Depth of Field” feature is on, parameters like the focal plane and F-stop allow realistic modeling of the defocus blur. Blender allows common picture formats as outputs, including lossy JPEG, lossless PNG, or even RAW with full float accuracy. The color space of the output can be BW, RGB, or RGBA. Figure 5 shows an example of rendered views of a dual-camera system. We can see rich features, colors, and interactions of the objects in the frames, and also parallax between two frames.

We also implement other functions (The simulation functions, training scripts, and evaluation notebooks for the following experiments are available at https://github.com/arizonaCameraLab/physicsAwareTransformer), among which we would like to emphasize the function of animation generation. We can assign random trajectories and transformations to mashes, and stream the data with regard to a given

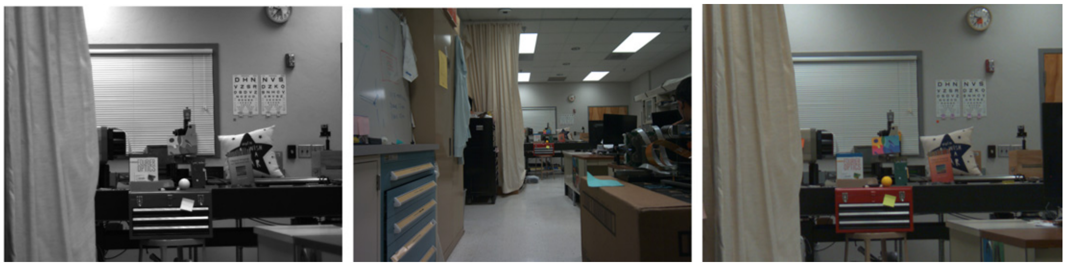

Figure 6. The views of the wide field - narrow field array. The pixel count of (a) is around 10× the pixel count of (c).

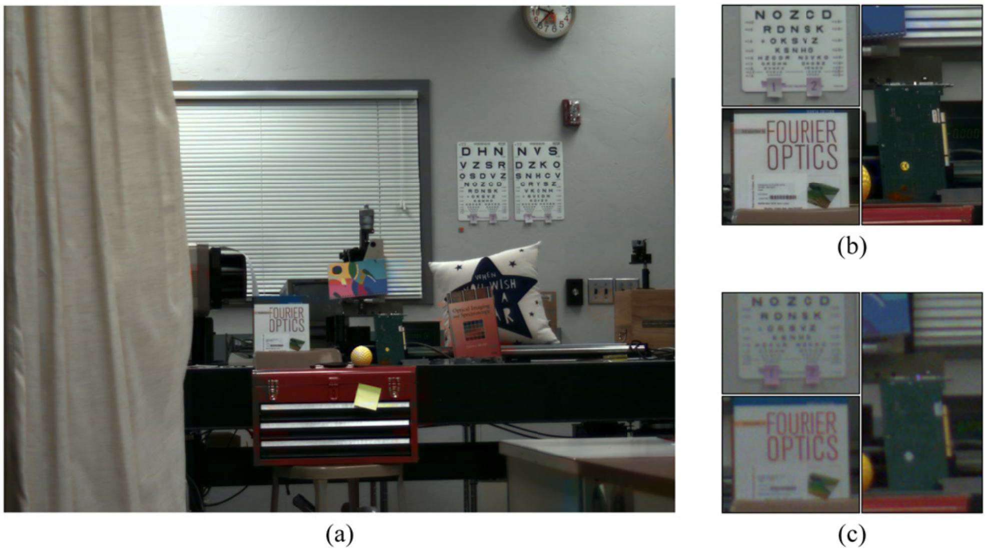

Figure 7. (a) is the fused frame on the wide field - narrow field array. (b) and (c) are associate details of the fused frame and the view of the wide field camera, respectively.

frame rate. It can benefit array camera research on temporal connections.

2.4 *Implementation Details*

The following details are shared by PATs for the experiments. The unique settings for each application are specified in the next section.

2.4.1 *Dataset*

We rendered 900 scenes of the resolution 1536 × 2048 to two cameras using the EEVEE render engine in Blender 2.92. 800 scenes were for the training and 100 scenes for the validation. One of the virtual cameras was selected to have the alpha viewpoint and the other had the beta viewpoint. The objects in the scene distribute within a 20-meter range. Rendered frames were in RGB color. We selected 49 patches of the resolution 128 × 384 across each scene, and then cubically downsampled the patches to 32 × 96. PATs were trained on these patches with ${}^{\alpha}H = {}^{\beta}H = 32$, ${}^{\alpha}W = {}^{\beta}W = 96$, and j ranging from 1 to 32 × 96. Each sample in the dataset has a pair of patches, where the patch from the alpha view is regarded as the ground truth. The degraded inputs, instead, were generated from the patch from the alpha view (alpha patch) and beta view (beta patch) while training via the forward model with regard to the array setting. We exported the extrinsics and intrinsics of two cameras and constructed the receptive fields according to the epipolar geometry, i.e., a dense map from each voxel index j in the alpha view to the associate voxel indices $j_1 \sim j_n$ in the beta views. n for all j was set to 96 in our dataset. The physical receptive fields for each sample were stored as arrays along with two patches.

2.4.2 *Training*

PATs were trained on the NVIDIA Tesla V100S GPU. Hyperparameters below were shared by the experimental systems:

$\boldsymbol{D}$	64	Epoch	80
$\boldsymbol{s}$	1	Criterion	Mean Square Error
$\tilde{\boldsymbol{C}}$	3	Optimizer	Adam [34]
Learning Rate	0.0002, decays by half per 30 epochs		

where $\mathbf{D}$, $\mathbf{s}$ and $\tilde{\mathbf{C}}$ are consistent with the notations in Fig. 1. The parameters that were specific to the application are clarified in the following subsections. The model with the best peak signal-to-noise-ratio (PSNR) performance on the validation set was selected for inference.

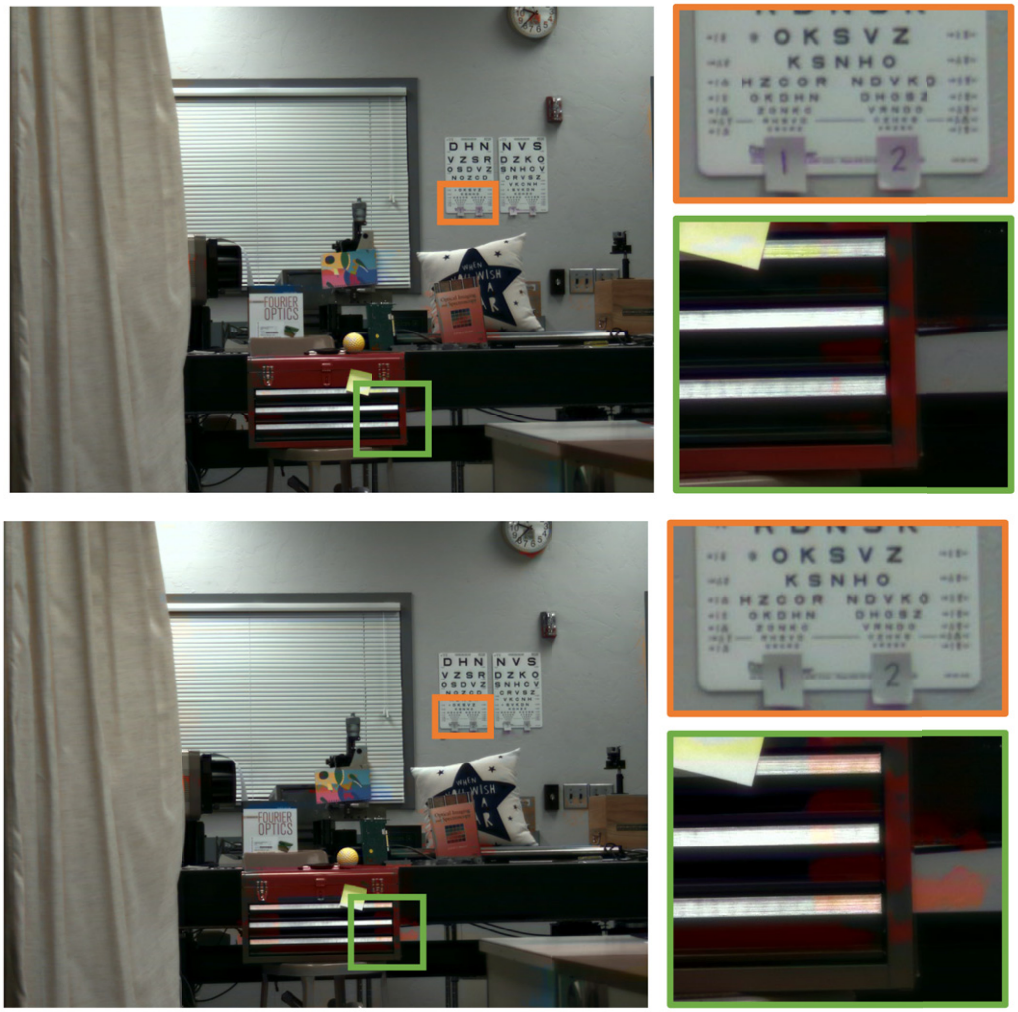

Figure 8. The fused frame with different receptive fields. (a) is generated with the calibrated receptive fields that accurately reflect the physics of the system. (b) is generated with the receptive fields assuming the inputs are rectified.

2.4.3 *Inference*

The intrinsics and extrinsics of the camera array were calibrated through MATLAB Stereo Vision Toolbox. We combined epipolar geometry and homography-based approximation to construct the physical receptive fields. The max displacement l was set to 80 unless specified. H_p and W_p were dependent on the resolution of input images and RAMs of computational devices.

3. EXPERIMENTS

Here we demonstrate four experimental systems with diverse sampling designs and PAT processing for image fusion, following the order mentioned in the introduction.

3.1 *Wide Field - Narrow Field System*

It is observed that chroma can be substantially compressed compared to luminance before the decompression error is perceived by humans. Inspired by that, we demonstrate a wide field color - narrow field monochrome system that compressed the color of the narrow field of view (FoV) by up to 40×. The configurations of the array were:

Narrow field camera

Body	Allied Vision Alvium 1800 U-1240m
Sensor	CMOS Monochrome
Lens	25 mm TECHSPEC HR Series
Resolution	4024×3036

Wide field camera

Body	iDS UI-3590LE-C-HQ
Sensor	CMOS Color
Lens	5 mm Kowa LM5JCM
Resolution	4912×3684

The focal plane of the wide field camera was set to its hyperfocal distance. The narrow field camera focused on the black optical table around 7 m away.

Figure 6 shows the camera views in an example scene. Considering the color filters on the wide field camera and 10× resolution gap in the narrow field, the red and blue raw signals were subsampled by $10 \times 4 = 40\times$ and the green raw

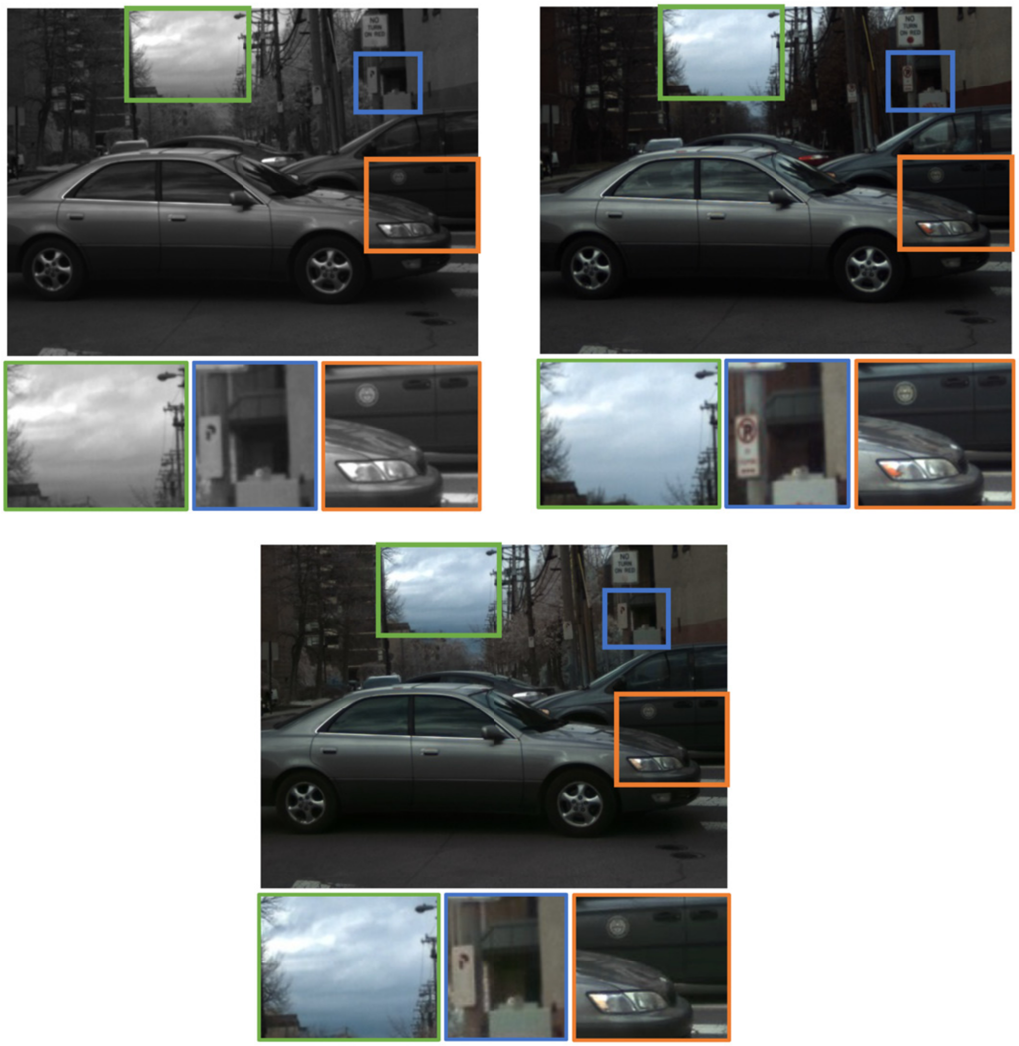

Figure 9. Data from PittsStereo-RGBNIR dataset [36] and the fused result. The orange, blue, and green windows contain the details in the scene from near to far. The brightness of details is adjusted to enhance contrast.

signals were subsampled by $10 \times 2 = 20\times$ compared to the luminance.

PAT acted as a color decompressor on this system that upsampled the colors in the narrow field. PAT was trained with two inputs. We converted the alpha patch to grayscale as one input and had the beta patch unchanged as the other input. To model possible resolution gaps and blur, we augmented the training data by (1) adding box blur to the alpha input; (2) adding box blur to the beta input; (3) 2× bicubically downsampling the beta input; (4) combining (2) and (3). These augmentation techniques were selected at random with equivalent probabilities during training and validation. The batch size of training was set to 32 and C was set to 3. In the training and inference phases, the alpha input was repeated along the feature dimension three times and the beta input was bicubically upsampled to its original dimensions if it had been downsampled.

Before implementing the trained PAT on the system, we evaluated the algorithm on Flickr1024 [28] and KITTI2012 [29] (20 frames) test sets. For each testing sample, we used whole frames instead of patches to generate inputs. The alpha frame was converted to grayscale as the alpha input and the beta frame was 2× or 4× bicubically downsampled as the beta input. Based on the characteristics of the test sets, the physical receptive fields indicated truncated horizontal epipolar lines of the length 120 divided by the downsampling rate. In Table I, we listed average PSNR and SSIM [35] scores between (1) the ground truth alpha frames and the grayscale alpha inputs in the “Alpha Input” column; (2) the beta frames and the beta inputs bicubically-upsampled to the original size in the “Beta Input” column; (3) the ground truth alpha frames and the fused results of PAT in the “Fusion” column. It can be observed that PAT improved the test system by maintaining the structures of the alpha input and improving the color upsampling results compared to the beta input solely.

We assigned the alpha viewpoint to the narrow field camera while inferencing. The result is shown in Figure 7. In comparison, colors were upsampled by up to 40× to

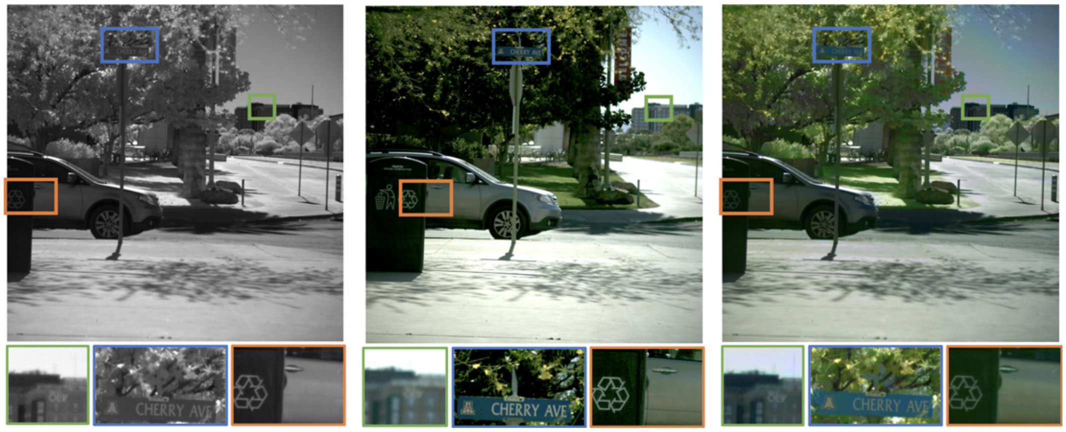

Figure 10. Data from our visible-NIR camera array and the fused result. The orange, blue and green windows contain the details in the scene from near to far. The brightness of details is adjusted to enhance contrast.

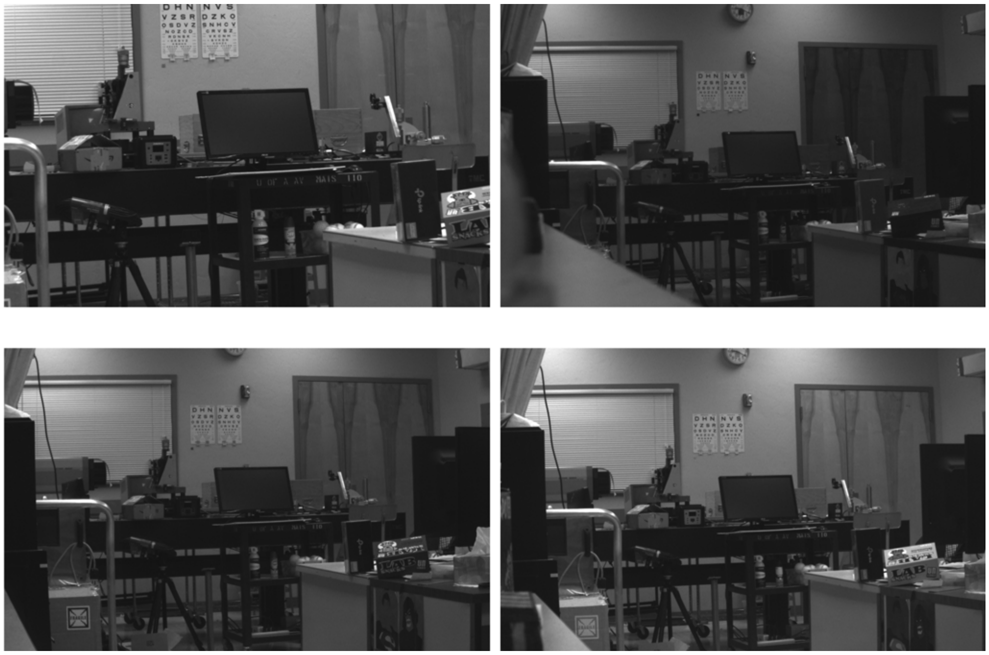

Figure 11. The views of the short exposure - long exposure array under diverse exposures in relative scales. 1 unit is approximately 12 microseconds.

Table I. Comparison between inputs and fused results of PAT (monochrome - color inputs).

Dataset	Scale	Alpha Input	Beta Input	Fusion
Flickr2014	×2	21.42/0.8800	24.95/0.8161	27.26/0.8992
	×4		21.84/0.6265	25.85/0.8840
KITTI2012	×2	26.40/0.9178	28.48/0.8845	29.55/0.9097
	×4		24.56/0.7376	28.40/0.8957

the narrow view without scarifying the sampling rate of the luminance. Although the color bleeding artifacts caused by a large upsampling rate can be observed in certain regions, we reduced the artifacts to the minimum by providing accurate physical information to the system. As illustrated in Figure 8, the correct receptive field yielded the result in Fig. 8(a) with correct colors (pink stickers in the orange window) and less artifacts (storage box in the green window).

3.2 *Visible - Near Infrared Systems*

As a result of reduced atmospheric scatter and absorption, near infrared (NIR) cameras achieve higher contrast in landscape photography. However, infrared (IR) signals are typically recorded as monochromatic data, thus are not visual friendly. Here we show PAT acted as a visualization tool to fuse color and NIR views while retaining the texture of remote objects on visible - NIR camera arrays. We used the data from two visible-NIR arrays; one was from a public

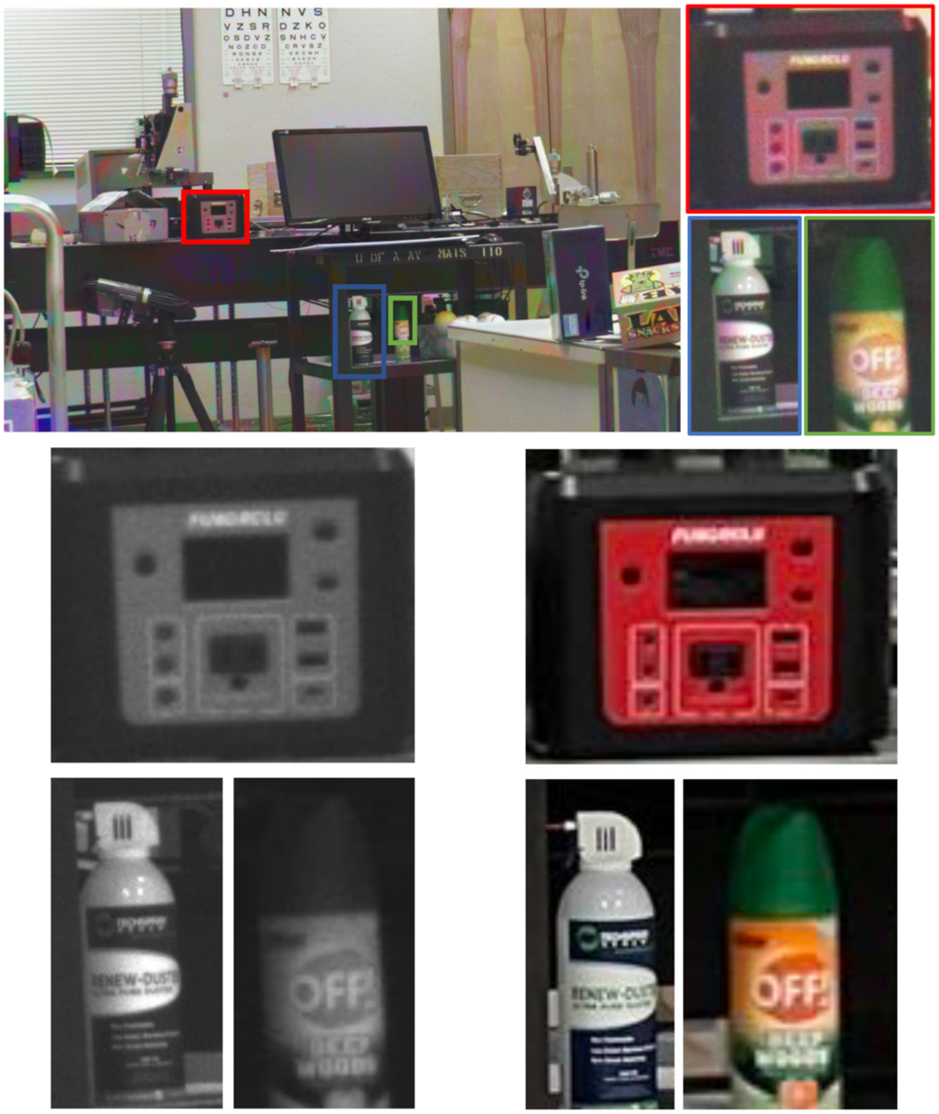

Figure 12. (a) is the fused frame on the short exposure - long exposure array. Zoomed-in views on the side highlight objects of different spectral responses. (b) are the associate details of the original frame from the short exposure camera. Details in (c) are captured by a cellphone camera to provide the readers with color references. The color balance of the fused frame and the brightness of details are adjusted for display.

database PittsStereo-RGBNIR [36] and the other was built by us. The configurations of the camera array from the online dataset are available in Reference [36]. We used rectified images of the resolution 582 × 429 from the database. Our visible-NIR system was composed of two 35 mm EO-4010 cameras, one with a color filter and the other with a NIR filter. The resolution of both cameras was 2048 × 2048.

We applied the pretrained PAT from the wide field - narrow field system to this fusion task to highlight the ability of domain adaptation of our algorithm. The attention engine of PAT operates on the features, thus is robust to the data that differs in appearance, brightness, etc.

The alpha viewpoint was assigned to the NIR camera while inferencing. Figures 9 and 10 demonstrate the fusion results with zoomed-in details on the given data. The color was well transferred to the fusion results in the presence of complicated occlusion and parallax. Moreover, different appearances of distant objects in the visible and NIR frames were fused nicely, as demonstrated in the green boxes.

3.3 *Short Exposure - Long Exposure System*

For visible color imaging, multiaperture sampling allows independent exposure and focus control for each band. We demonstrate this capability using a 2 × 2 camera array based on the Arducam 1MP×4 Quadrascopic OV9281. The cameras were monochromatic and had 1280 × 800 resolution. One camera with a 12 mm lens had no filter, while the others with 8 mm lenses were equipped with three filters. The central wavelengths of the filters were 450 nm, 550 nm, and 600 nm respectively. The filters shared 80 nm full width at half maximum. The exposure time of each camera was controlled independently to optimize the dynamic range of the signal. Figure 11 shows the views of four cameras.

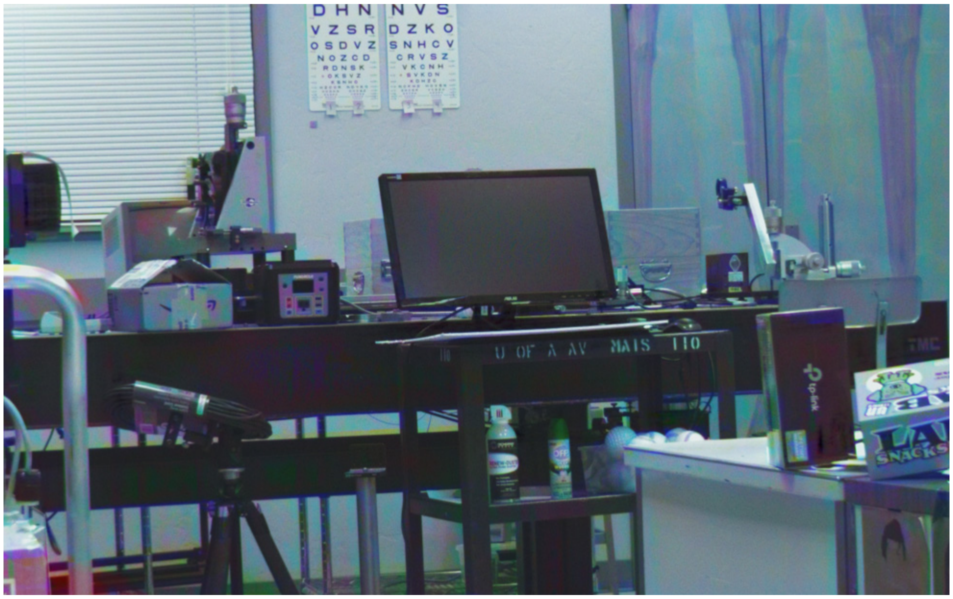

Figure 13. The fused results of the permutated input sequence, where the 450 nm view and 650 nm view were switched. The color balance is adjusted for display.

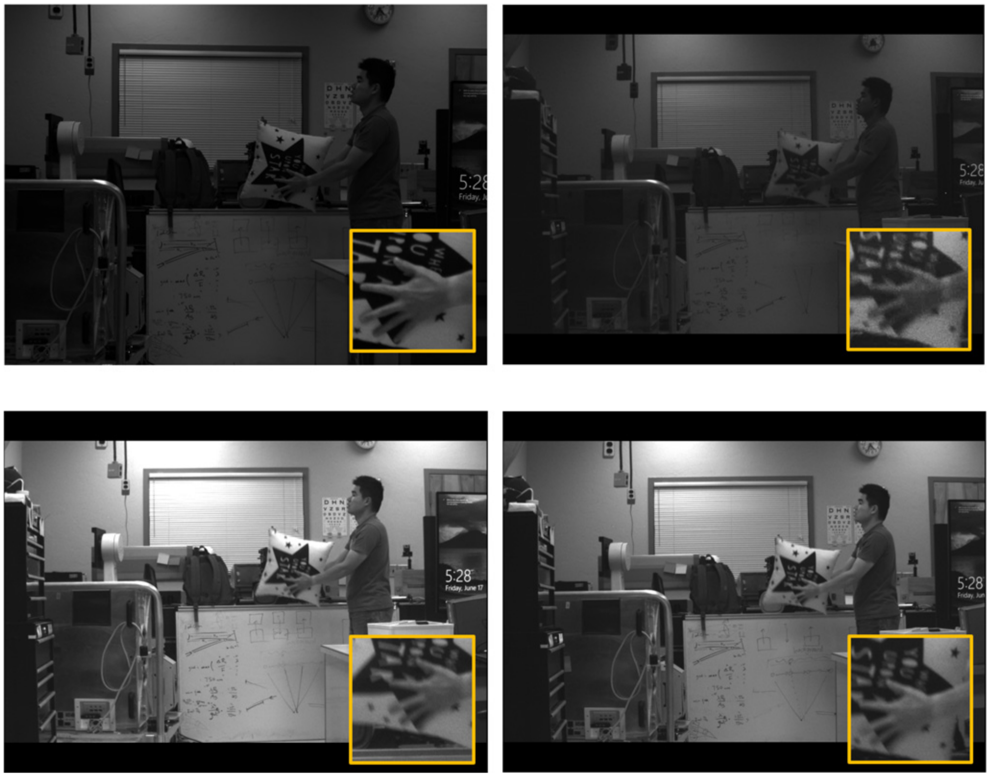

Figure 14. The views of the HFR - LFR array. The exposure time of (a) was 4.3 ms. The exposure times of (b)–(d) were the same, around 12 ms. Since the HFR camera was not synchronized with LFR cameras, (a) was captured ±2.15 ms away from the moment that (b)–(d) were captured. The orange windows highlight the moving pillow. The brightness and contrast of the patches in the orange windows were adjusted for display.

Compared to uni-exposure systems, such as cameras with the Bayer filter, our system allowed up to 5× differences in exposure, thus having higher overall throughput of spectral data.

PAT was trained with 4 inputs. The alpha patch was converted to be grayscale as the alpha input. The red, green, and blue channels were unpacked from the beta patch as three alternative inputs. Note the color filters of our synthetic

Table II. Comparison between inputs and fused results of PAT (monochrome - spectral inputs).

Dataset	Scale	Alpha Input	Beta Inputs	PAT
Flickr2014	×2	21.42/0.8800	24.95/0.8161	25.55/0.8714
	×4		21.84/0.6265	24.00/0.8493
KITTI2012	×2	26.40/0.9178	28.48/0.8845	28.77/0.8955
	×4		24.56/0.7376	28.13/0.8903

training data did not exactly resemble the filters we used with regard to the spectral curves. The batch size of training was set to 16 and C was set to 1 as inputs were monochromatic.

Most of the test settings agreed with those in the wide field - narrow field system, except that the beta frame was unpacked into three frames of a single color channel and downsampled to generate three beta inputs. In Table II, PSNR and SSIM scores in the “Beta Inputs” column were first averaged between three spectral bands of the beta frame and corresponding beta inputs, and then averaged across all beta frames. The meanings of other columns are the same as in Table I. Similarly we can also see PAT improved overall system performance.

While inferencing, the alpha viewpoint was assigned to the camera without the filter. Figure 12 shows the fused result. The result preserved the geometry of the alpha camera view and displayed the correct color, indicating that the algorithm effectively adapted to data with different filter functions. We can expect the result generated with the optimized spectral throughput to have a higher dynamic range. Note that PAT is physical-based rather than perception-based, therefore the network does not “guess” the color beyond physical clues. As shown in Figure 13, the color channels of the fused frame were permutated with regard to the way that the inputs were permutated.

3.4 *High Frame Rate - Low Frame Rate System*

Sensors with color filters sacrifice quantum efficiency compared to monochrome sensors, thus requiring a longer exposure time to achieve a comparable signal-to-noise ratio (SNR). This prevents standalone spectral cameras from achieving a higher frame rate. Here we demonstrate an imaging system that combines one high frame rate (HFR) monochrome camera with three low frame rate (LFR) spectral cameras as a better solution to sample the light field temporally. This system enables PAT to reconstruct the light field at a high frame rate.

We applied one Basler acA1440-220um camera with a 12 mm lens as the HFR camera, which can reach 227 frames per second (fps) at the 1456 × 1088 resolution. Three Arducam cameras with 8 mm lenses and spectral filters in the short exposure - long exposure system were applied as the LFR cameras. The LFR cameras are synchronized, operating at 30 fps. Figure 14 shows the views of four cameras in a scene where the moderate motion of the pillow occurred. We can see the LFR frames deteriorated in the region that has motions, while the associate region in the HFR frame remained sharp.

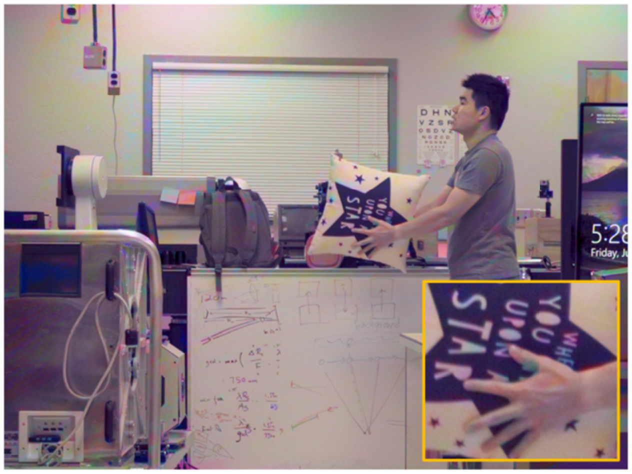

Figure 15. Fused results on the HFR - LFR array. The color balance is adjusted for display.

The alpha and beta viewpoints were assigned to the HFR camera and LFR cameras, respectively. For one LFR frame captured at a certain moment, the HFR frames captured ±15 ms from that moment correspond to that LFR frame. We assumed the epipolar constraint was valid in general between the LFR frame and associate HFR frames and built physical receptive fields accordingly. We applied the pretrained PAT from the short exposure-long exposure system while inferencing. Figure 15 shows the fused result, which fused the HFR camera view with colors from three spectral cameras. Because the epipolar geometry does not strictly hold for unsynchronized frames, slight color jittering of letters in the pillow was observed. However, the majority of colors of the pillow were effectively fused and the motion boundary was well preserved. We expect the physical information that characterizes the lags between frames and the motion in the scene to refine the receptive fields and yield an improved result.

Given two sets of LFR frames with three filters (6 frames in total) captured at 0 and 26 ms, PAT fused seven HFR frames with color in between. Figure 16 shows the patches of the moving pillow in the fused results. The pillow in the fused patches was in color with sharp motion boundaries, compared to LFR patches.

4. DISCUSSION

In this paper, we discussed the merits of sampling using array cameras and proposed a physics-aware transformer (PAT) for image fusion on array cameras.

We concluded that *heterogeneity* is a good criterion to evaluate the array design. Specifically, cameras in the array should be complementary to maximize the information throughput, sampling diverse perspectives of the light field, such as FoV, resolution, focal plane, focal length, color space, and exposure. Dynamic control and interleaved coding are also expected to incorporate multiaperture sampling to

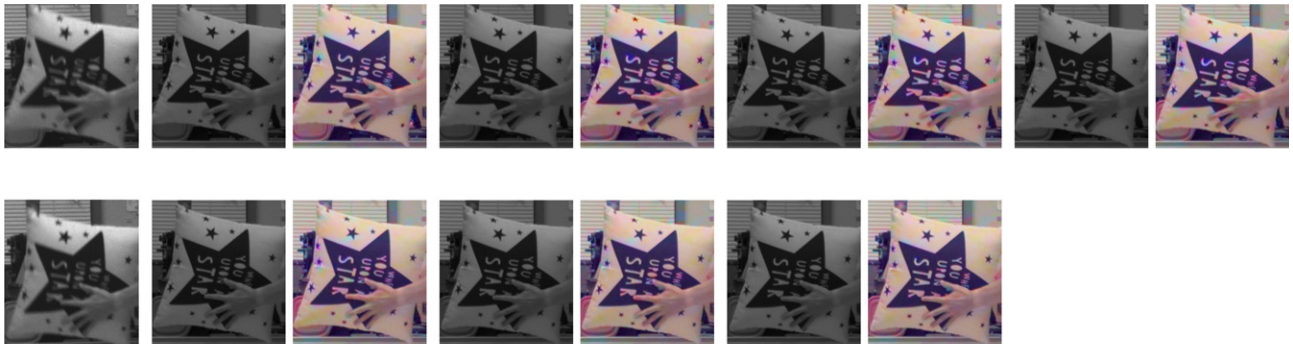

Figure 16. The patches of the moving pillow. (a) and (f) are two consecutive frames from a 600 nm LFR camera. The left images in (b)–(e) and (g)–(i) are the patches from the HFR camera while the right images are from the fused results. The labels are the estimated time elapsed from the moment that (a) was captured. Three LFR frames captured at 0 ms were used to generate the results in (b)–(e), while three LFR frames captured at 26 ms were used to generate the results in (g)–(h). The color balance is adjusted for display.

boost diversity. All these together pose novel challenges to camera designers. The main point of design shifts from optimizing a single lens to optimizing a multicamera system to achieve the target performance within budget. For instance, the multiscale spectral sampling or foveated spatial sampling [13] are more favorable. With that in mind, we demonstrated four experimental systems with diverse sampling strategies and anticipated the inner thoughts to inspire future designs of camera systems.

We showcased the versatility of PAT on four different camera arrays. In contrast to its predecessors, this network architecture can incorporate tailored receptive fields to reflect the physics of the imaging system like epipolar geometry and homography, thus being applicable to general arrays of multiple cameras, nonstandard layouts and heterogeneous specifications with comparable efficiency. The proposed pipeline of data synthesis effectively provides training data for transformers and has the potential to benefit other learning algorithms. We envision PAT being a standard processing tool for array cameras of the next generation, and inspiring designs, combinations and applications of array cameras for better light field sampling.

ACKNOWLEDGMENT

We would like to thank Emily Chau in Infrared Imaging Group at the University of Arizona for the visible-NIR array setup and data collection.

Qian Huang and Minghao Hu are students in the Department of Electrical and Computer Engineering at Duke University, Durham, NC 27708. This work was finished when they were interning at the University of Arizona.

REFERENCES

[1] J. N. Mait, G. W. Euliss, and R. A. Athale, “Computational imaging,” Adv. Opt. Photonics **10**, 409–483 (2018).

[2] I. Ihrke, J. Restrepo, and L. Mignard-Debise, “Principles of light field imaging: Briefly revisiting 25 years of research,” IEEE Signal Process. Mag. **33**, 59–69 (2016).

[3] R. Lukac and K. N. Plataniotis, “Color filter arrays: Design and performance analysis,” IEEE Trans. Consum. Electron. **51**, 1260–1267 (2005).

[4] X. Xiao, B. Javidi, M. Martinez-Corral, and A. Stern, “Advances in three-dimensional integral imaging: sensing, display, and applications,” Appl. Opt. **52**, 546–560 (2013).

[5] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, “*Light field photography with a hand-held plenoptic camera*,” Ph.D. thesis (Stanford University, 2005).

[6] X. Yuan, D. J. Brady, and A. K. Katsaggelos, “Snapshot compressive imaging: Theory, algorithms, and applications,” IEEE Signal Process. Mag. **38**, 65–88 (2021).

[7] X. Hu, X. Lin, T. Yue, and Q. Dai, “Multispectral video acquisition using spectral sweep camera,” Opt. Express **27**, 27088–27102 (2019).

[8] C. Wang, Q. Huang, M. Cheng, Z. Ma, and D. J. Brady, “Deep learning for camera autofocus,” IEEE Trans. Comput. Imaging **7**, 258–271 (2021).

[9] D. J. Brady, W. Pang, H. Li, Z. Ma, Y. Tao, and X. Cao, “Parallel cameras,” Optica **5**, 127–137 (2018).

[10] J. Tanida, “Multi-aperture optics as a universal platform for computational imaging,” Opt. Rev. **23**, 859–864 (2016).

[11] R. Plemmons, S. Prasad, S. Mathews, M. Mirotznik, R. Barnard, B. Gray, P. Pauca, T. Torgersen, J. Van Der Gracht, and G. Behrmann, “Periodic: integrated computational array imaging technology,” *Computational Optical Sensing and Imaging* (Optica Publishing Group, Washington, DC, 2007).

[12] P. M. Shankar, W. C. Hasenplaugh, R. L. Morrison, R. A. Stack, and M. A. Neifeld, “Multiaperture imaging,” Appl. Opt. **45**, 2871–2883 (2006).

[13] D. J. Brady, L. Fang, and Z. Ma, “Deep learning for camera data acquisition, control, and image estimation,” Adv. Opt. Photonics **12**, 787–846 (2020).

[14] X. Yuan, M. Ji, J. Wu, D. J. Brady, Q. Dai, and L. Fang, “A modular hierarchical array camera,” Light. Sci. Appl. **10**, 1–9 (2021).

[15] L. Juan and G. Oubong, “Surf applied in panorama image stitching,” *2010 2nd Int’l. Conf. on Image Processing Theory, Tools and Applications* (IEEE, Piscataway, NJ, 2010), pp. 495–499.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” Adv. Neural Inf. Process. Syst. **30** (2017).

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, and J. Uszkoreit, “An image is worth 16 × 16 words: Transformers for image recognition at scale”. arXiv *Preprint* arXiv:2010.11929 (2020).

[18] Y. Romano, M. Elad, and P. Milanfar, “The little engine that could: Regularization by denoising (red),” SIAM J. Imaging Sci. **10**, 1804–1844 (2017).

[19] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Deep image prior,” *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2018), pp. 9446–9454.

[20] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, “Deep joint demosaicking and denoising,” ACM Trans. Graph. **35**, 1–12 (2016).

[21] M. Jaderberg, K. Simonyan, and A. Zisserman, “Spatial transformer networks,” Adv. Neural Inf. Process. Syst. **28** (2015).

[22] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.* (IEEE, Piscataway, NJ, 2000), Vol. 22, pp. 1330–1334.

[23] Y. He, R. Yan, K. Fragkiadaki, and S.-I. Yu, "Epipolar transformers," *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2020), pp. 7779–7788.

[24] L. Wang, Y. Wang, Z. Liang, Z. Lin, J. Yang, W. An, and Y. Guo, "Learning parallax attention for stereo image super-resolution," *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2019), pp. 12250–12259.

[25] C. Chen, C. Qing, X. Xu, and P. Dickinson, "Cross parallax attention network for stereo image super-resolution," *IEEE Trans. Multimedia* (IEEE, Piscataway, NJ, 2021).

[26] B. Yan, C. Ma, B. Bare, W. Tan, and S. C. H. Hoi, "Disparity-aware domain adaptation in stereo image restoration," *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2020), pp. 13179–13187.

[27] T. Plotz and S. Roth, "Benchmarking denoising algorithms with real photographs," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2017), pp. 1586–1595.

[28] Y. Wang, L. Wang, J. Yang, W. An, and Y. Guo, "Flickr1024: A large-scale dataset for stereo image super-resolution," *Int'l. Conf. on Computer Vision Workshops* (IEEE, Piscataway, NJ, 2019), pp. 3852–3857.

[29] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," *Conf. on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2012).

[30] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," *European Conf. on Computer Vision (ECCV)* (Springer, Berlin, Heidelberg, 2012), pp. 611–625.

[31] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *Proc. IEEE Int'l. Conf. on Computer Vision* (IEEE, Piscataway, NJ, 2015), pp. 2758–2766.

[32] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2016), pp. 4040–4048.

[33] M. Denninger, M. Sundermeyer, D. Winkelbauer, D. Olefir, T. Hodan, Y. Zidan, M. Elbadrawy, M. Knauer, H. Katam, and A. Lodhi, "Blenderproc: Reducing the reality gap with photorealistic rendering," *Int'l. Conf. on Robotics: Sciene and Systems, RSS 2020* (Dagstuhl, Wadern, 2020).

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," IEEE Trans. Image Processing **13**, 600–612 (2004).

[36] T. Zhi, B. R. Pires, M. Hebert, and S. G. Narasimhan, "Deep material-aware cross-spectral stereo matching," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2018), pp. 1916–1925.