4D Visualization of Dynamic Events from Unconstrained Multi-View Videos

Aayush Bansal Minh Vo Yaser Sheikh Deva Ramanan Srinivasa Narasimhan
Carnegie Mellon University
{aayushb,mpvo,yaser,deva,srinivas}@cs.cmu.edu
http://www.cs.cmu.edu/~aayushb/Open4D/

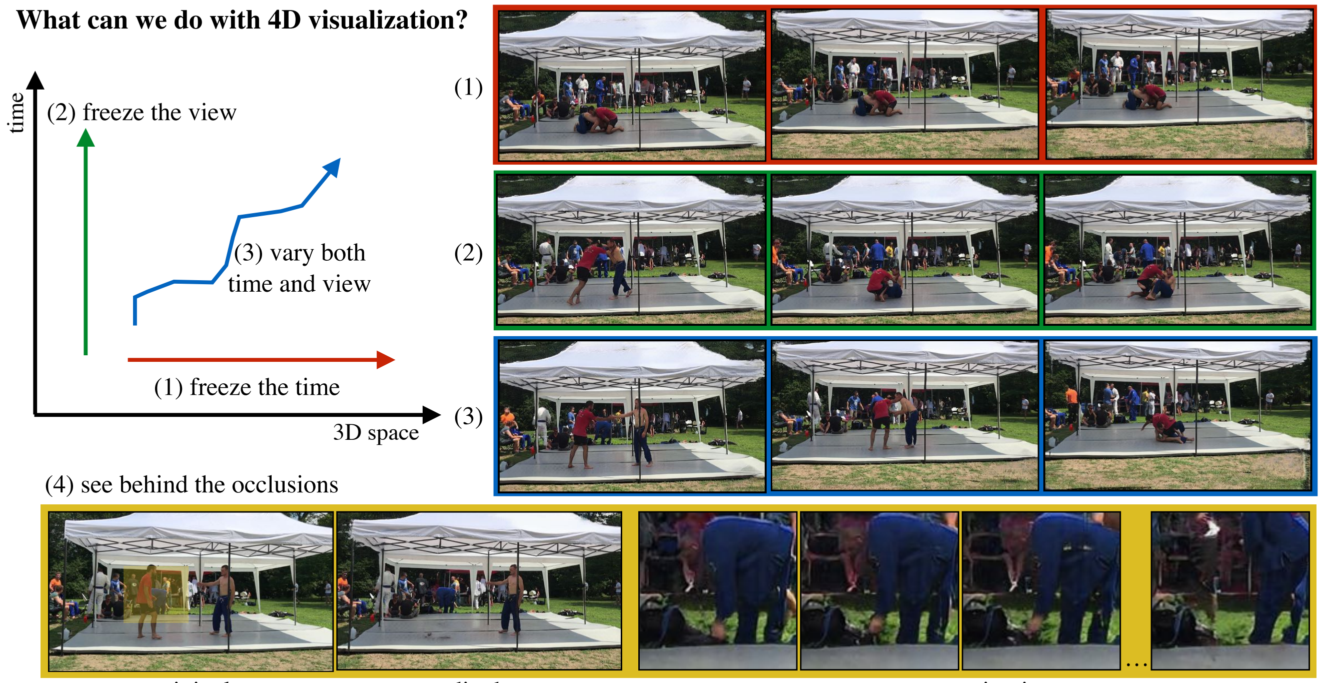
What can we do with 4D visualization?
time
(2) freeze the view
(3) vary both time and view
(1) freeze the time
3D space
(1)
(2)
(3)
(4) see behind the occlusions
original
edited
zoom-in view

Figure 1. We can create virtual cameras that facilitate: (1) freezing the time and exploring views (**red**); (2) freezing a view and moving through time (**green**); (3) varying both time and view (**blue**); and (4) seeing behind the occlusions (yellow).

Abstract

We present a data-driven approach for 4D space-time visualization of dynamic events from videos captured by hand-held multiple cameras. Key to our approach is the use of self-supervised neural networks specific to the scene to compose static and dynamic aspects of an event. Though captured from discrete viewpoints, this model enables us to move around the space-time of the event continuously. This model allows us to create virtual cameras that facilitate: (1) freezing the time and exploring views; (2) freezing a view and moving through time; and (3) simultaneously changing both time and view. We can also edit the videos and reveal occluded objects for a given view if it is visible in any of the other views. We validate our approach on challenging in-the-wild events captured using up to 15 mobile cameras.

1. Introduction

Imagine going back in time and revisiting crucial moments of your lives, such as your wedding ceremony, your graduation ceremony, or the first birthday of your child, immersively from any viewpoint. The prospect of building such a *virtual time machine* [38] has become increasingly realizable with the advent of affordable and high-quality smartphone cameras producing extensive collections of social video data. Unfortunately, people do not benefit from this broader set of captures of their social events. When looking back, we are likely to only look at one video or two when potentially hundreds might have been captured from different sources. We present a data-driven approach that leverages all perspectives to enable a more complete exploration of the event. With our approach, the benefits from each extra perspective that is captured leads to a more

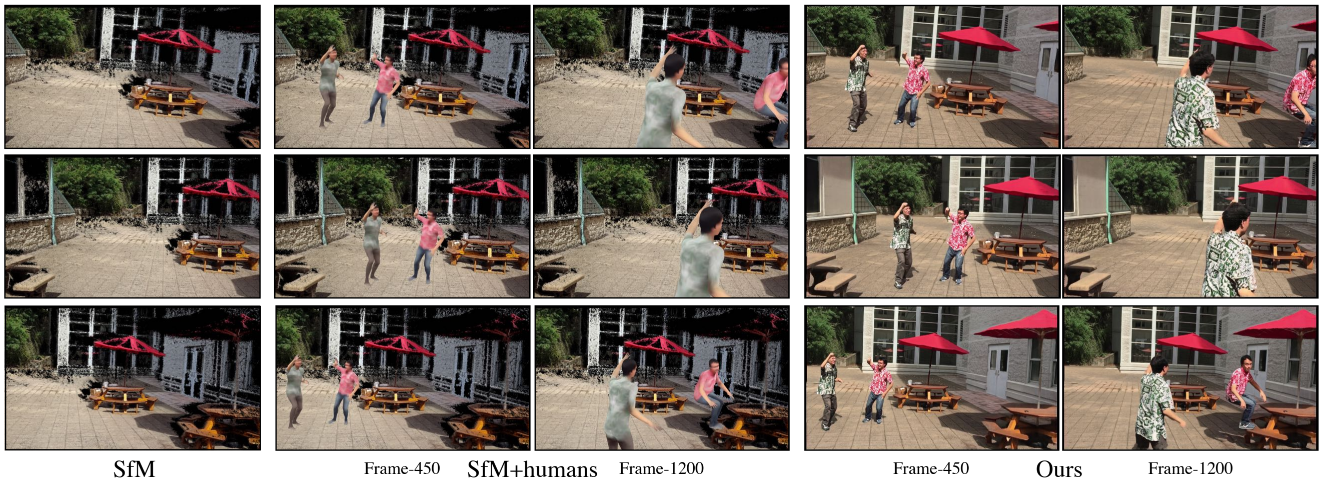

Figure 2. **Comparison to existing work:** Given a dynamic event captured using 10 phones, we **freeze time and explore views** for two time instances. We use a standard Structure-from-Motion (SfM) [40, 41] to reconstruct the camera trajectory. As shown in first-column, SfM treats dynamic information as outliers for rigid reconstruction. We use additional cues such as 2D keypoints [4], statistical human body model [34], and human association [49] along-with the outputs of SfM to generate dynamic information for these two time instances (Frame-450 and Frame-1200 in second and third columns respectively). We call this **SfM+humans**. These three outputs lack *realism*. Additionally, the reconstruction fails for non-Lambertian surfaces (see glass windows), non-textured regions (see umbrellas), and shadows (around humans). Our approach, on the other hand, can densely synthesize the various static and dynamic components, as shown in fourth and fifth columns for the same moments.

complete experience. We seek to automatically organize the disparate visual data into a comprehensive four-dimensional environment (3D space and time). The complete control of spatiotemporal aspects not only enables us to see a dynamic event from any perspective but also allows geometrically consistent content editing. This functionality unlocks many potential applications in the movie industry and consumer devices, especially as virtual reality headsets are becoming popular by the day. Figure 1 show examples of virtual camera views synthesized using our approach for an event captured from multi-view videos.

Prior work on virtualized reality [27, 29, 31] has primarily been restricted to studio setups with tens or even hundreds of synchronized cameras. Four hundred hours of video data is uploaded on YouTube every minute. This feat has become possible because of the commercial success of high quality hand-held cameras such the iPhones or GoPros. Many public events are easily captured from multiple perspectives by different people. Despite this new form of big visual data, reconstructing and rendering the dynamic aspects have mostly been limited to studios and not for in-the-wild captures with hand-held cameras. Currently, there exists no method for fusing the information from multiple cameras into a single comprehensive model that could facilitate content sharing. This gap is largely because the mathematics of dynamic 3D reconstruction [19] is not well-posed. The segmentation of objects [18] are far from being consistently recovered to do 3D reconstruction [54]. Large scale analytics of internet images exist for static scenes [23, 40, 41, 44] alone, and ignores the interesting dynamic events (as shown in Figure 2-first-column).

We pose the problem of 4D visualization from in-the-wild captures within an image-based rendering paradigm utilizing large capacity parametric models. The parametric models based on convolutional neural nets (CNNs) can circumvent the requirement of explicitly computing a comprehensive model [2, 5] for modeling and fusing static and dynamic scene components. Key to our approach is the use of self-supervised CNNs specific to the scene to compose static and dynamic parts of the event. This data-driven model enables us to extract the nuances and details in a dynamic event. We work with in-the-wild dynamic events captured from multiple (not a fixed number) mobile phone cameras. These multiple views have arbitrary baselines and unconstrained camera poses.

Despite impressive progress with CNN-based scene reconstruction [51, 25, 32, 50], noticeable holes and artifacts are often visible, especially for large texture-less regions or non-Lambertian surfaces. We accumulate spatiotemporal information available from multiple videos to capture content that is not visible at a particular time instant. This accumulation helps us to capture even the large non-textured regions (umbrellas in Figure 2) or non-Lambertian surfaces (glass windows in Figure 2). Finally, a complete control of static and dynamic components of a scene, and viewpoint and time enables user-driven content editing in the videos. In public events, one often encounters random movement obstructing the cameras to capture an event. Traditionally

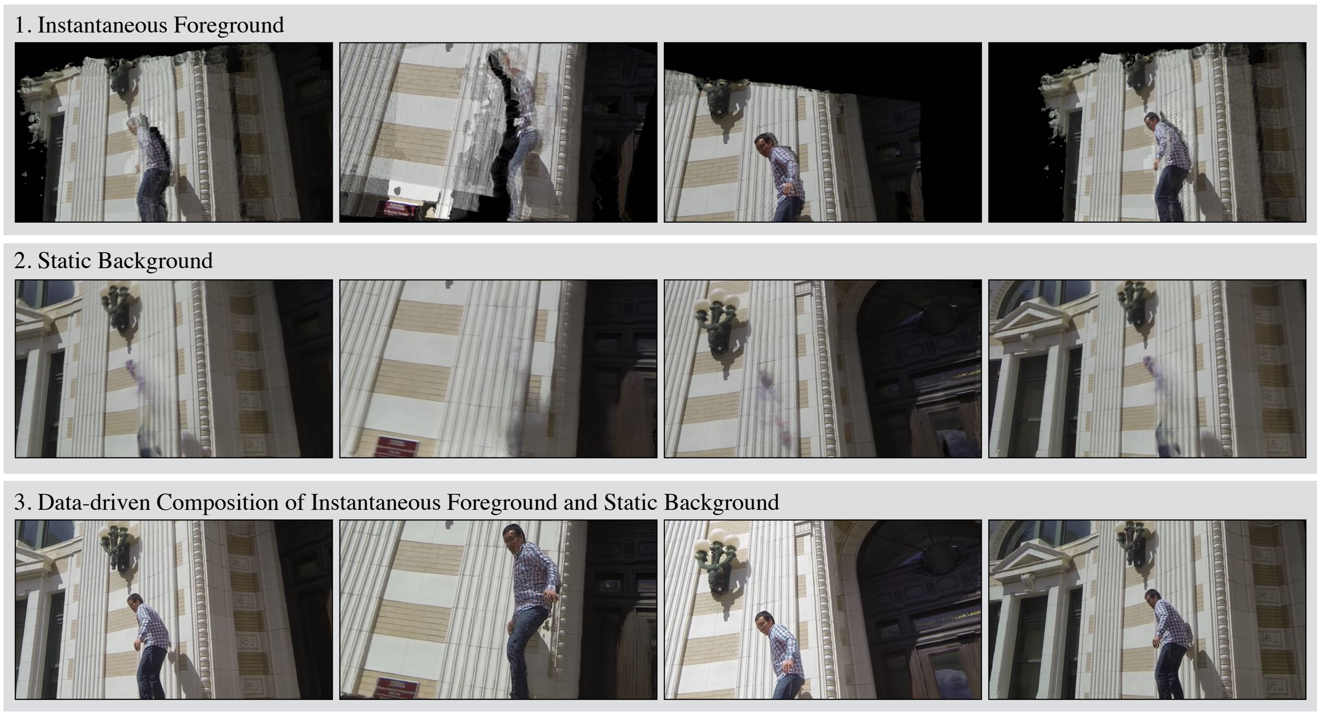
1. Instantaneous Foreground
2. Static Background
3. Data-driven Composition of Instantaneous Foreground and Static Background

Figure 3. **Overview:** We pose the problem of 4D visualization of dynamic events captured from multiple cameras as a data-driven composition of instantaneous foreground (**top**) and static background (**middle**) to generate the final output (**bottom**). The data-driven composition enables us to capture certain aspects that may otherwise be missing in the inputs, e.g., parts of the human body are missing in the first and second column, and parts of background are missing in first row.

nothing can be done about such spurious content in captured data. The complete 4D control in our system enables the user to remove unwanted occluders and obtain a clearer view of the actual event using multi-view information.

2. Related Work

There is a long history of 4D capture systems [29] to experience immersive virtualized reality [13], especially being able to see from any viewpoint that a viewer wants irrespective of the physical capture systems.

4D Capture in Studios: The ability to capture depth maps from a small baseline stereo pair via 3D geometry techniques [19] led to the development of video-rate stereo machines [31] mounting six cameras with small baselines. This ability to capture dense depth maps motivated a generation of researchers to develop close studios [27, 30, 37, 56] that can precisely capture the dynamic events happening within it. A crucial requirement in these studios is the use of synchronized video cameras [30]. This line of research is restricted to a few places in the world with access to proper studios and camera systems.

Beyond Studios: The onset of mobile phones have revolutionized the capture scenario. Each one of us possess high-definition smartphone cameras. Usually, there are more cameras at a place than there are people around. Many public events are captured by different people from various perspectives. This feat motivated researchers to use in-the-wild data for 3D reconstruction [22, 44] and 4D visualization [2, 5]. A hybrid of geometry [19] and image-based rendering [42] approaches have been used to reconstruct 3D scenes from pictures [7]. Photo tourism [44] and the works following it [1, 14, 15, 23, 43] use internet-scale images to reconstruct architectural sites. These approaches have led to the development of immersive 3D visualization of static scenes.

The work on 3D reconstruction treats dynamic information as outliers and reconstructs the static components alone. Additional cues such as visual hulls [12, 17, 35], or 3D body scans [5, 6], or combination of both [3, 47] are used to capture dynamic aspects (esp. human performances) from multi-view videos. Hasler et al. [20] use markerless method by combining pose estimation and segmentation. Vedula et al. [46] compute scene shape and scene flow for 4D modeling. Ballan et al. [2] model foreground subjects as video-sprites on billboards. However, these methods assume a single actor in multi-view videos. Recent approaches [9, 48] are not restricted by this assumption but does sparse reconstruction.

CNN-based Image Synthesis: Data-driven approaches [8, 16, 26, 52] using convolutional neural networks [33] have led to impressive results in image synthesis. These results inspired a large body of work [10, 11, 28, 36, 45, 55] on con-

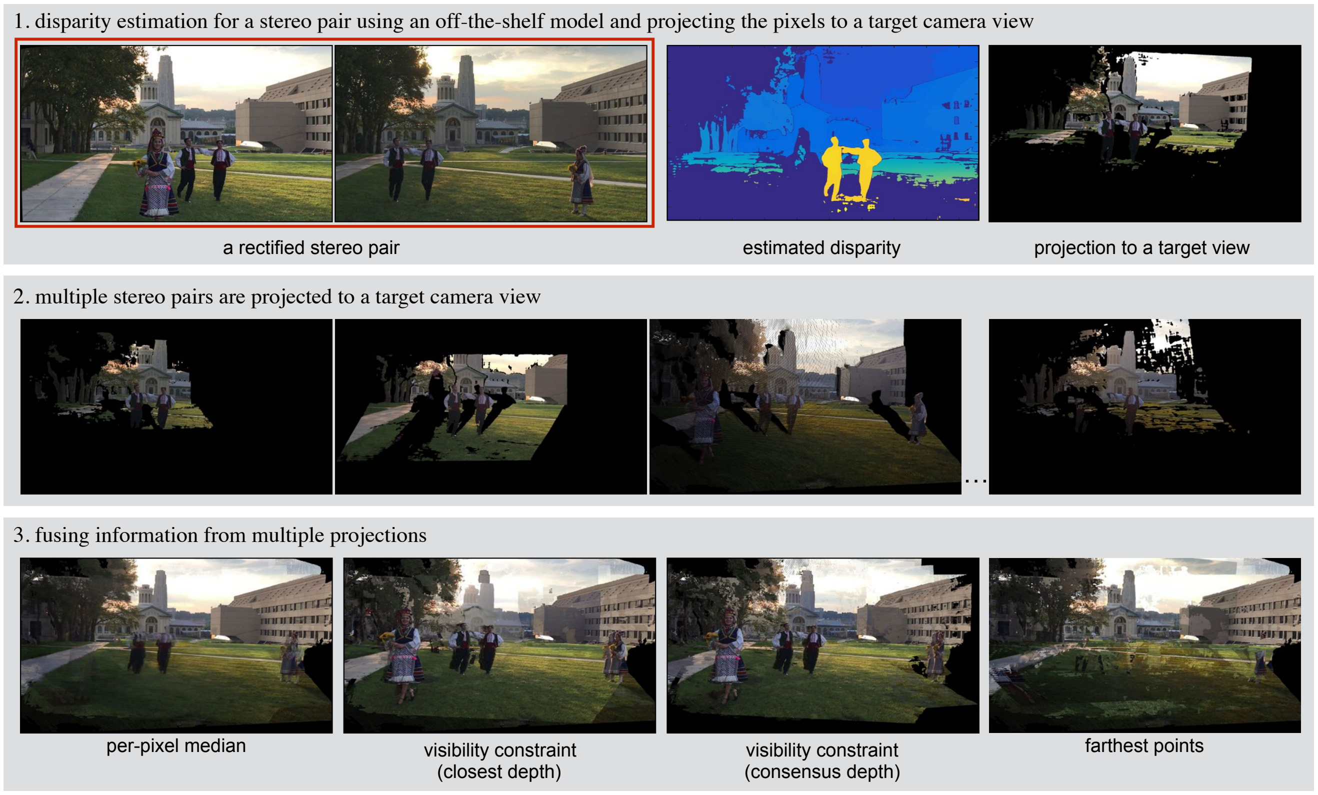
1. disparity estimation for a stereo pair using an off-the-shelf model and projecting the pixels to a target camera view
a rectified stereo pair
estimated disparity
projection to a target view
2. multiple stereo pairs are projected to a target camera view
3. fusing information from multiple projections
per-pixel median
visibility constraint (closest depth)
visibility constraint (consensus depth)
farthest points

Figure 4. **Instantaneous Foreground Estimation:** Given multiple stereo pairs and a target camera view, there are three steps for estimating instantaneous foreground. (1) We estimate disparity for a rectified stereo pair using an off-the-shelf disparity estimation approach [50] and project it to a target camera view using standard 3D geometry [19]. (2) We repeat Step-1 for the $\binom{N}{2}$ stereo pairs. (3) A crucial aspect is to fuse the information from multiple projections to generate a comprehensive instantaneous foreground. A simple per-pixel median over the multiple projections leads to a loss of dynamic information because of the poor 3D estimates (shown in first column). We propose a visibility constraint using painter's algorithm. The visibility constraint builds a per-pixel cost volume of depth. It is natural to have multiple depth values for a given 2D pixel location in the image. For each pixel location, we consider the 3D point corresponding to the *closest depth* to the target camera view. Due to noisy 3D estimates, it often leads to ghosting artifacts (shown in second column). However, we have multiple views and we ensure consistency in the depth values of the closest 3D point for projection to a 2D pixel location by analyzing their visibility. As shown in third column, this *consensus in depth* leads to improved results. Finally, we also show the projection of pixels corresponding to the *farthest points* (fourth column) for a given pixel location to illustrate cost volume of depth. Note that it removes dynamic information in the scene.

tinuous view synthesis for small baseline shifts. Hedman et al. [21] extended this line of work to free-viewpoint capture. However, these methods are currently applicable to static scenes only. We combine the insights from CNN-based image synthesis and earlier work on 4D visualization to build a data-driven 4D Browsing Engine that makes minimal assumption about the content of multi-view videos.

3. 4D Browsing Engine

We are given N camera views with extrinsic parameters $\{C_1, C_2, ..., C_N\}$, and intrinsic parameters $\{M_1, M_2, ..., M_N\}$. Our goal is to generate virtual camera view C that does not exist in any of these N cameras. We temporally align all cameras using spatiotemporal bundle adjustment [48], after which we assume the video streams are perfectly synchronized. Our method should be robust to possible alignment errors. Figure 3 shows an overview of our approach via a virtual camera that freezes time and explores views. We begin with estimation of instantaneous foreground in Section 3.1, static background in Section 3.2, and data-driven composition in Section 3.3. We finally discuss practical details and design decisions in Section 4.

3.1. Instantaneous Foreground Estimation

The N-camera setup provides us with $\binom{N}{2}$ stereo pairs. We build foreground estimates for a given camera pose and a time using multiple rectified stereo pairs. There are three essential steps for estimating instantaneous foreground (shown in Figure 4). (1) We begin with estimating disparity using an off-the-shelf disparity estimation module [50]. The knowledge of camera parameters for the stereo pair allows us to

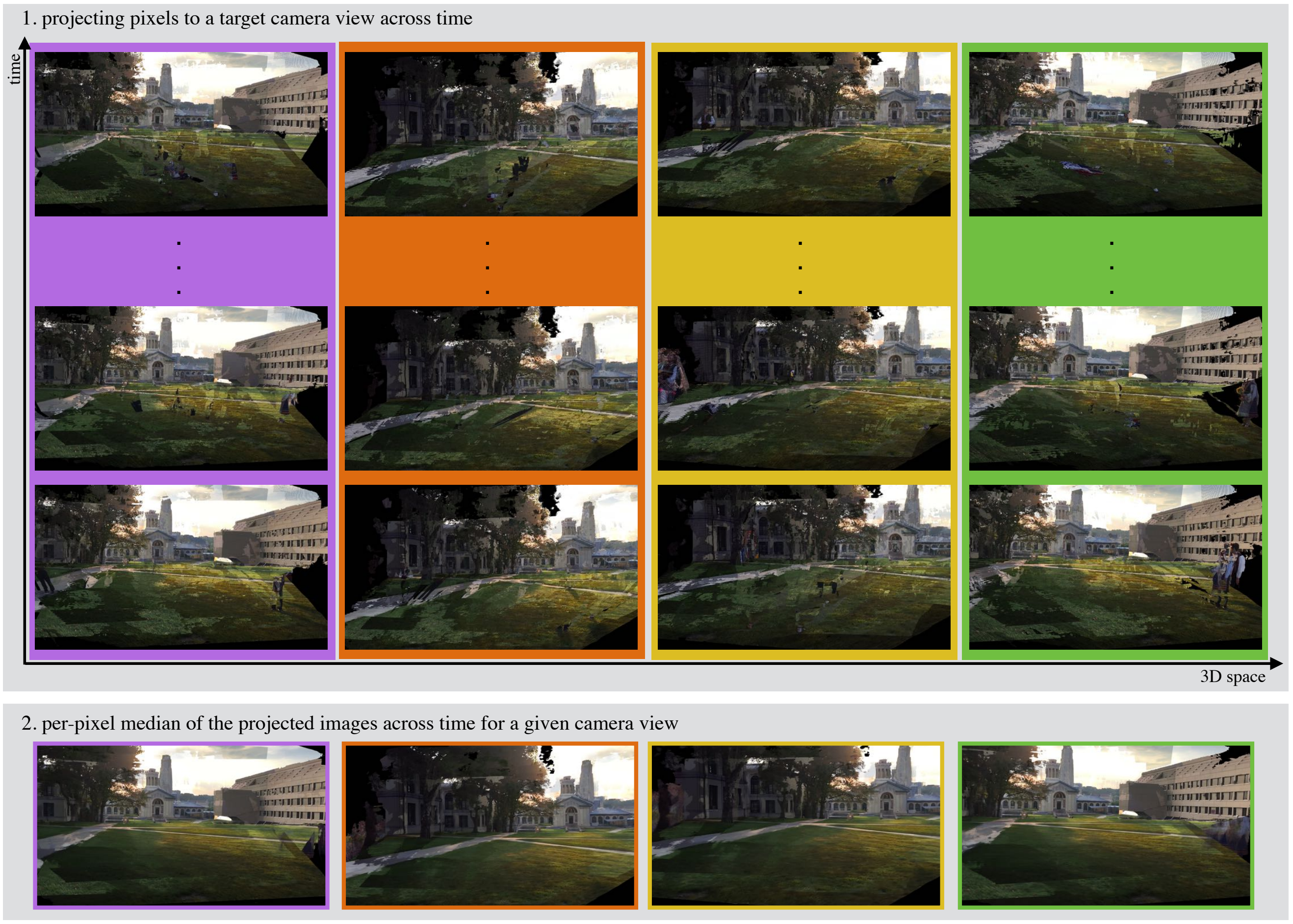
1. projecting pixels to a target camera view across time
time
3D space
2. per-pixel median of the projected images across time for a given camera view

Figure 5. **Static Background Estimation:** (1) We use the pixels corresponding to the farthest 3D points (from the per-pixel cost volume of depth) to generate an image for a target camera pose and time instant. This enables us to get the farther components of the scene that are often stationary. However, this information is noisy. We, therefore compute the images for a given camera pose across all time. (2) A per-pixel median of the images over a large temporal window for a target camera pose results in a smooth stationary background.

project the pixels to a target camera view using standard 3D geometry [19]. Figure 4-1 shows an example of projection for a rectified stereo pair. (2) We repeat Step-1 for multiple stereo pairs. This step gives us multiple projections to a target camera view (shown in Figure-4-2) and per-pixel depth estimates. The projections from various $\binom{N}{2}$ stereo pairs tends to be noisy. This is due to sparse cameras, large stereo baseline, bad stereo-pairs, or errors in camera-poses. We cannot naively fuse the multiple projections to synthesize a target view in all conditions for a target camera view. As an example, a simple per-pixel median on these multiple projection leads to the loss of dynamic information as shown Figure-4-3 (first column). (3) We enforce a visibility constraint to fuse information from multiple stereo pairs.

Visibility Constraint: We use a visibility constraint that is a close approximation of painter's algorithm. The visibility constraint builds a per-pixel cost volume of depth that allows us to retrace the ray of light to its origin for a given camera view. Since we have multiple views, it is natural to have multiple depth values for a given 2D pixel location in the image. For each pixel location, we consider the 3D point corresponding to the closest depth. However, the closest 3D point from the cost volume of depth may not be accurate in our setup. As shown in the second column of Figure-4-3, we observe ghosting artifacts due to noisy 3D estimates.

Fortunately, we can use multiple views to ensure consistency in the depth values of the closest 3D point that we should consider for projection at a given 2D pixel location. While a certain 3D location may not be viewed by all the cameras, it is highly likely that it would be seen in a significant subset of cameras. This means that we will estimate depth of same 3D point in our setup from many stereo pairs. We use this insight to build a consensus about the closest 3D point by ensuring that same depth value is observed in at least 3 stereo pairs. We start with the smallest depth value (> 0) at a pixel location in the cost volume and iterate till we find

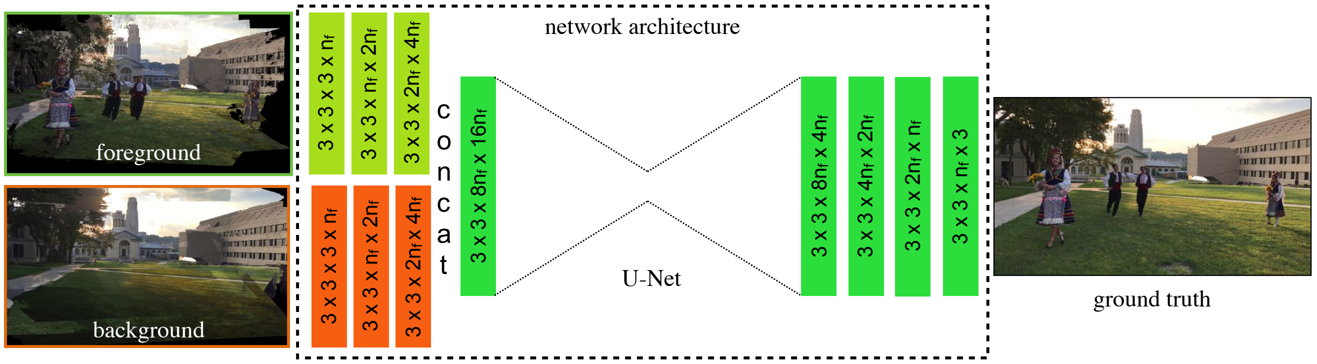

Figure 6. **Self-Supervised Composition of Foreground and Background:** Given an input-output paired data created for a held out camera, we train a neural network to learn the composition. We show our neural network architecture that is used to compose the instantaneous foreground ($f_{c,t}$) and static background (b_c). The first part of our network consists of three convolutional layers. We concatenate the outputs of both foreground and background streams. The concatenated output is then fed forward to the standard U-Net architecture [39]. Finally, the output of U-Net is then fed forward to three more convolutional layers that give the final image output.

a consensus. We ignore the pixel location (leave it as blank) if it is not observed from a sufficient number of cameras (less than 3). Ideally, the depth values of a 3D point (though computed from different stereo pairs) should be same from the target camera view. For the practical purposes, we define a threshold that can account for noise. As shown in third column of Figure-4-3, this leads to improved results. However, there is still missing information and ghosting artifacts in this output to qualify as a real image.

3.2. Static Background Estimation

The missing information in the output of Section 3.1 is due to the lack of visibility of points across multiple views for a given time instant. We accumulate long-term spatiotemporal information to compute static background for a target camera view. The intrinsic and extrinsic parameters from N physical cameras enable us to create the views over a large temporal window of $[0, t]$ for a target camera position. Figure 5-1 shows examples of virtual cameras for various poses and time instants. For a given camera pose and time instant, we once again use the dense per-pixel cost volume of depth. This time we use the farthest depth values to capture missing information (also refer to the last column of Figure 4-3).

The estimates corresponding to the farthest point for one time instant are noisy. We compute the images for a given camera pose across all time. A per-pixel median of different views (across time) for a given camera pose results in a smoother static background (albeit still noisy). Empirically, such median image computed over large temporal window for a given camera position also contain textureless and non-Lambertian stationary surfaces in a scene (observed consistently in Figure 5-2, Figure 2, and Figure 3). We now have a pair of complementary signals: (1) sparse and noisy foreground estimates; and (2) dense but static background estimates.

3.3. Self-Supervised Composition

We use a data-driven approach to learn the fusion of instantaneous foreground, F, and static background, B, to generate the required target view for given camera parameters. The instantaneous foreground for a camera pose c and time t is notated as $f_{c,t}$. The static background for a camera pose c is notated as b_c. However, there exists no ground truth or paired data to train such a model in a data-driven manner. We formulate the data-driven composition in a self-supervised manner by reconstructing a known held-out camera view, $o_{c,t}$, from the remaining $N-1$ views. This gives us a paired data $\{((f_{c,t}, b_c,), o_{c,t})\}$ for learning a mapping $G : (B, F) \rightarrow O$. We now have a pixel-to-pixel translation [26] and therefore, we can easily train a convolutional neural network (CNN) specific to a scene. We use three losses for optimization: (1) Reconstruction loss; (2) Adversarial loss; and (3) Frequency loss.

Reconstruction Loss: We use standard l_1 reconstruction loss to minimize reconstruction error on the content with paired data samples:

$$\min_G L_r = \sum_{(c,t)} ||o_{c,t} - G(f_{c,t}, b_c)||_1 \tag{1}$$

Adversarial Loss: Recent work [16] has shown that learned mapping can be improved by tuning it with a discriminator D that is adversarially trained to distinguish between real samples of $o_{c,t}$ from generated samples $G(f_{c,t}, b_c)$:

$$\min_G \max_D L_{adv}(G, D) = \sum_{(c,t)} \log D(o_{c,t}) + \sum_{(c,t)} \log(1 - D(G(f_{c,t}, b_c))) \tag{2}$$

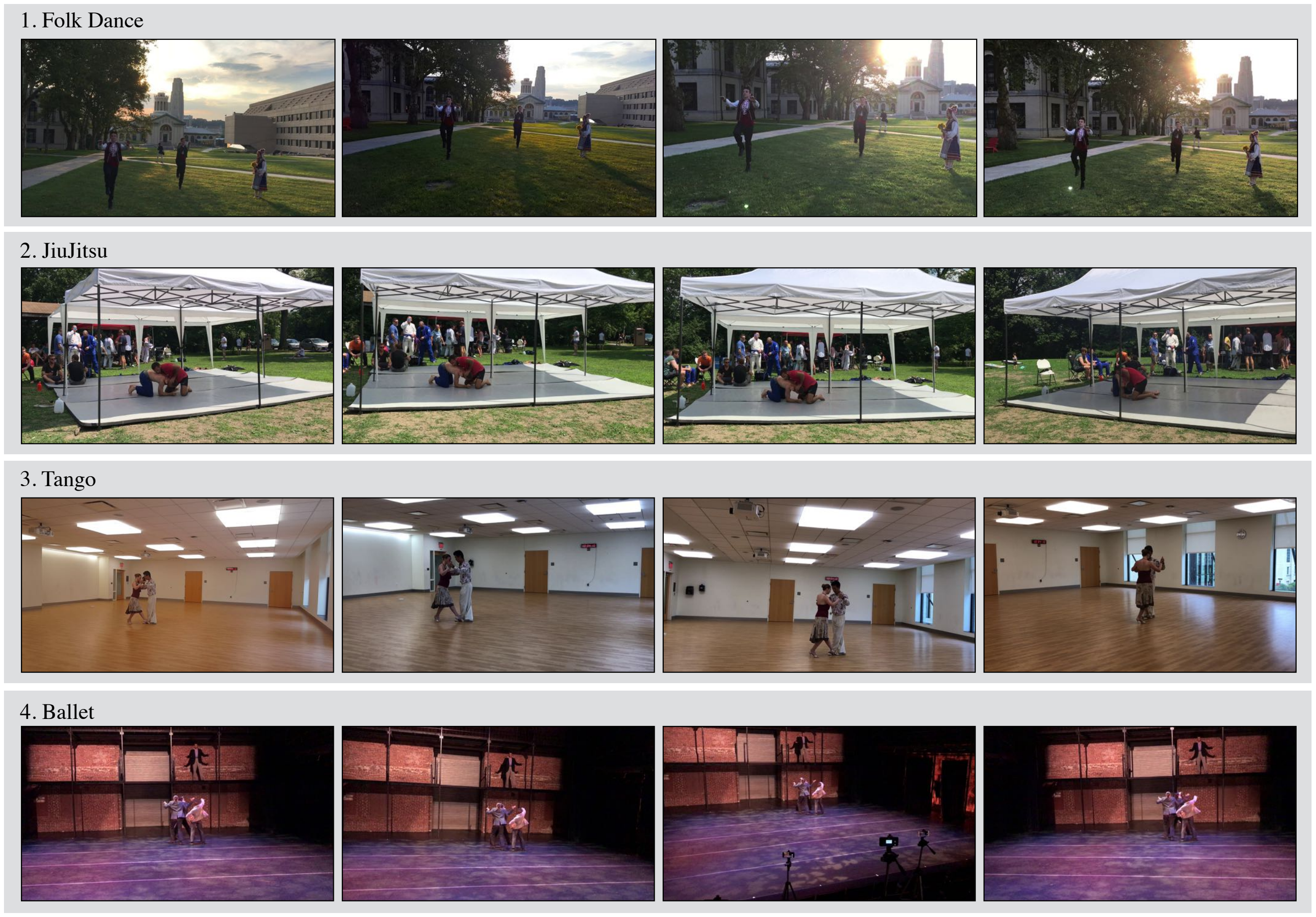

Figure 7. **Human Performances**: We captured a wide variety of human performances from multiple cameras. The human performances in these sequences have a wide variety of motion, clothing, human-human interaction, and human-object interaction . These sequences were captured in varying environmental and illumination condition. Shown here are examples from four such sequences to give a sense of extremely challenging setup dealing with a wide and arbitrary camera baseline.

Frequency Loss: We enforce a frequency-based loss function via Fast-Fourier Transform to learn appropriate frequency content and avoid generating spurious high-frequencies when ambiguities arise (inconsistent foreground and background inputs):

$$\min_G L_{fr} = \sum_{(c,t)} ||\mathscr{F}(o_{c,t}) - \mathscr{F}(G(f_{c,t}, b_c))||_1 \qquad (3)$$

where $\mathscr{F}$ is fast-Fourier transform. The overall optimization combines Eq. 1, Eq. 2, and Eq. 3:

$$L = \lambda_r L_r + \lambda_{adv} L_{adv} + \lambda_{fr} L_{fr}$$

where, $\lambda_r = \lambda_{fr} = 100$, and $\lambda_{adv} = 1$. Explicitly using background and foreground for target view makes the model independent of explicit camera parameters.

Network Architecture & Optimization: We use HD-image inputs (1080$\times$1920). The input images are zero-padded making them 1280$\times$2048 in dimensions. We use a modified U-Net architecture in our formulation as shown in Figure 6. There are three parts of our neural network architecture. (1) The first part of our network consists of three convolutional layers. We concatenate the outputs of this part for both foreground and background stream. (2) The concatenated output is fed forward to the second part that is a standard U-Net architecture [39]. (3) Finally, we feed the outputs of U-Net through three more convolutions that generates the final image. The batch-size is 1. The number of filter, n_f, in the first conv-layer of our network is 13 (see Figure 6 for reference). For data augmentation purposes, we randomly resize the input to upto 1.5$\times$ scale factor and randomly sample a crop from it. We use the Adam solver. A model is trained from scratch with a learning rate of 0.0002, and remains constant throughout the training. The model converges quickly in around 10 epochs for a sequence.

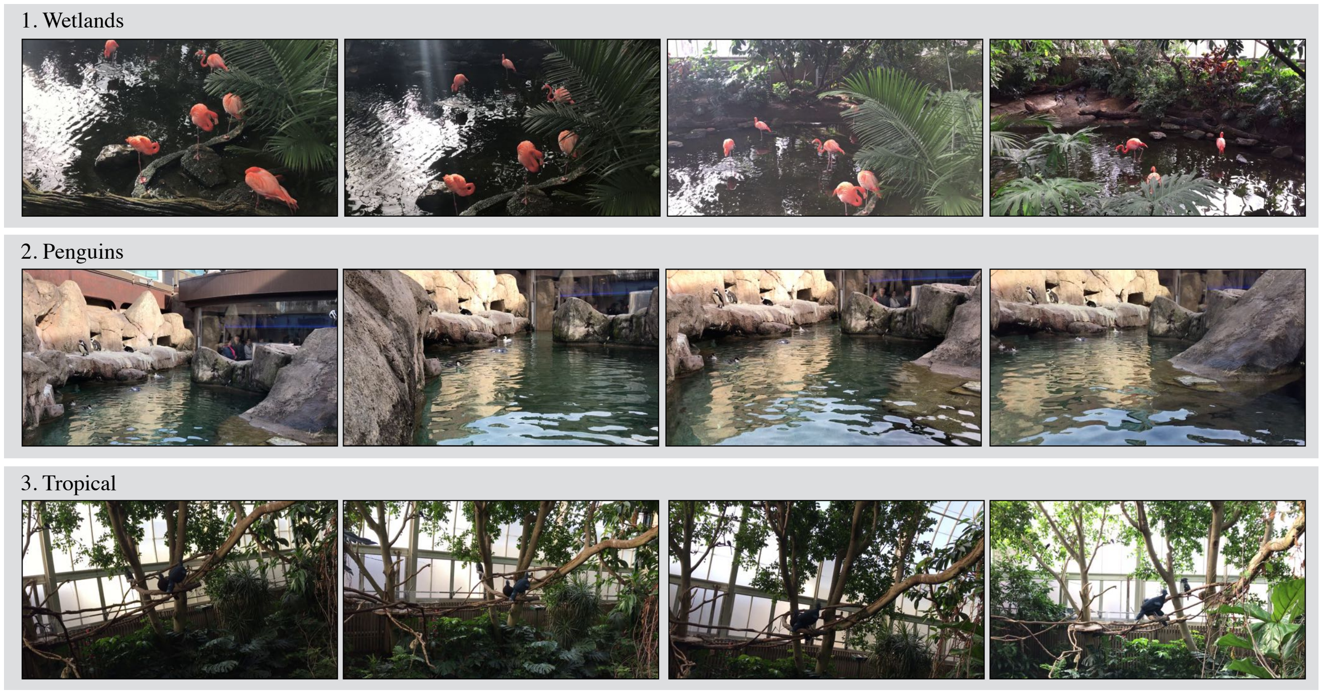

Figure 8. **Bird Sequences**: We captured a wide variety of birds at the National Aviary of Pittsburgh. We have no control on the motion of birds, their environment, lighting condition, and dynamism in the background due to human movement in the aviary. Shown here are examples from three such sequences to give a sense of our capture scenario.

4. Practical Limitations and Design Decisions

There are two practical challenges: (1) how to hallucinate missing information?; (2) how to avoid overfitting to specific input-output pair for a scene-specific CNN? In this section, we discuss the design decisions that we made to overcome these challenges.

4.1. Hallucinating Missing Information

We have so far assumed the scenario where sufficient foreground and background information are available that can be composed to generate a comprehensive new view. However, it is possible that certain regions are never captured (e.g. blank pixels in foreground and background estimation in Figure 4 and Figure 5 respectively). This is possible due to many reasons such as sparse cameras, large stereo baseline, bad stereo-pairs, or errors in camera-poses. This is also possible if we try to visualize something beyond the convex hull of physical cameras. While filling smaller holes is reasonable for a parametric model, filling larger holes lead to temporally inconsistent artifacts. One way to deal with it is to learn a higher capacity model that may learn to fill larger holes. Other than fear of overfitting, the larger model needs prohibitive memory. Training a neural network combining the background and foreground information with HD images already require 30 GB memory of a v100 GPU. In this work, we use a stacked multi-stage CNN to overcome this issue.

Multi-Stage CNN: We use a high capacity model for low-res image generation that learns overall structure, and improve the resolution with multiple stages. We train three models for three different resolutions, namely: (1) low-res (270$\times$480); (2) mid-res (540$\times$960); and (3) hi-res (1080$\times$1960). These models are trained independently and form multiple stages of our formulation. At test time, we use these models sequentially starting from low-res to mid-res to hi-res outputs. This sequential processing is done when there are large holes in the instantaneous foreground and static background inputs. The output of the low-res model is used to fill the holes to the input of next model in sequence. Here we leverage the fact that we can have a very high capacity model for a low-resolution image and that holes become smaller as we make the image smaller. The overall network architecture remains the same as the high-res model described in Section 3.3. We mention the differences for the low-res and mid-res models.

Low-Res Model: The low-res model inputs 270$\times$480 images. As such, it can have more parameters than a mid-res or hi-res model that inputs 4$\times$ and 16$\times$ higher resolution inputs respectively. We use a $n_f = 28$ for this model. The input images are zero-padded, and make them 320$\times$512 in dimensions.

Mid-Res Model: We use a $n_f = 23$ for this model. The input images are zero-padded, and make them 640$\times$1024 in dimensions.

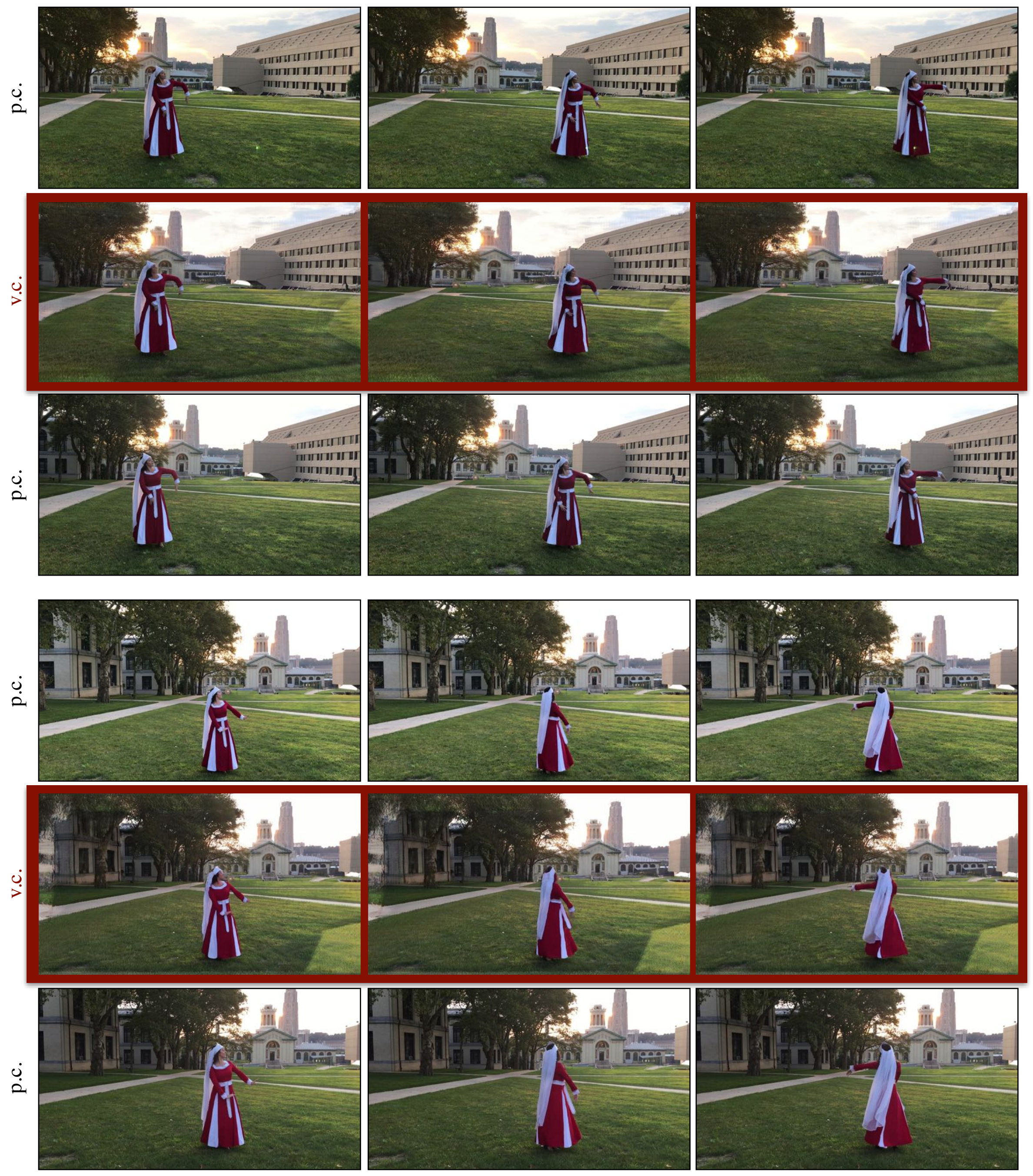

Figure 9. **Flowing Dress and Open Hair:** We show two examples of virtual cameras generated for the Western Folk Dance sequence in which the performer is wearing a flowing dress with open hairs. For each virtual camera (v.c.), we show the physical cameras (p.c.) on its sides. Zoom-in for the detailed human motion and the dress in these two examples.

4.2. Limited Data and Overfitting

In this work, we train a CNN specific to a scene/sequence. Owing to limited data for a sequence, the models are susceptible to overfit to specific inputs and fail to generalize for a wide range of camera views not available at training time. We could not naively use very high capacity models and train them using limited input-output pairs. Now, there are two popular ways to increase the capacity of CNNs: (1) adding more convolutions to make them deeper; and (2) increasing the number of filters in each convolutional layer. We take a hybrid of these methods. We add more convolutions, but in

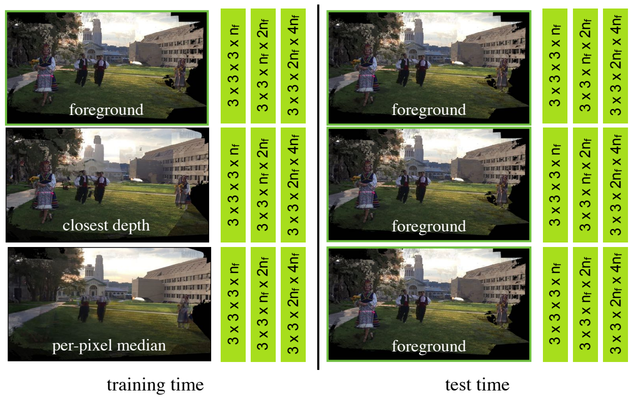

Figure 10. **Foreground Streams:** We use three parallel streams of convolution to encode foreground information. At training time, we use different foreground estimates (see Section 3.1) as input to each stream. At test time, we use the same foreground estimated using the consensus depth for all streams.

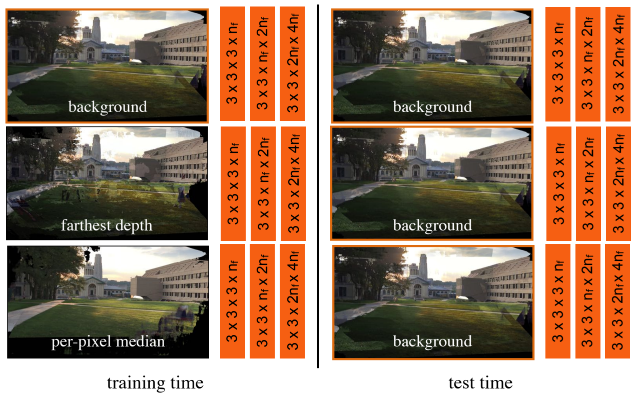

Figure 11. **Background Streams:** We use three parallel streams of convolution to encode background information. At training time, we use different background estimates (see Section 3.2) as input to each stream. At test time, we use the same background estimated using our long-term spatiotemporal accumulation method for all streams.

parallel and not sequential, which also increase the number of filters in the later parts of our network. Specifically, we add more foreground and background streams in our network. Figure 6 shows one stream each of foreground and background feeding to the U-Net. We use three streams each for foreground and background in our setup as shown in Figure 10 and Figure 11. We use different inputs for each stream at training time to provide variability in the data.

Foreground Streams: Figure 10 shows three parallel foreground streams used to encode foreground information. We use different foreground estimates as an input to each stream. We refer the reader to Section 3.1 for the details of each input shown in Figure 10. At test time, we only use our foreground estimated using the consensus depth for all streams.

Background Streams: Similarly, Figure 11 shows three parallel background streams used to encode background information. We use different background estimates as an input to each stream. We refer the reader to Section 3.1 and Section 3.2 for the details of each input shown in Figure 11[1]. At test time, we only use our static background estimated using our long-term spatiotemporal accumulation method for all streams.

The addition of multiple streams allow us to increase the capacity of the model and use of different inputs provide variability in the data. This enables us to train a model without overfitting to specific input-output pairs.

5. Unconstrained Multi-View Sequences

We collected a large number of highly diverse sequences of unrestricted dynamic events consisting of humans and birds. These sequences were captured in different environments and varying activities using up to 15 hand-held mobile phones. At times, we also mount the cameras on tripod stand as a proxy of hand-held capture. Note that an actual hand-held capture is better than one mounted on tripod for two reasons: (1) diversity of captured views for training our model; (2) the videographer bias enables a better capture of the event. We now describe a few sequences used in this work to get a better sense of the data.

5.1. Human Performances

We captured a wide variety of human motion, human-human interaction, human-object interaction, clothing, both indoor and outdoor, under varying environmental and illumination conditions.

Western Folk Dance: We captured sequences of western folk dance performances. This sequence is challenging due to *flowing* dresses worn by performers, open hair, self-occlusions, and illumination conditions. We believe that this sequence also paves the path for incorporating illumination condition in the future work. Figure 7-1 shows one of the two western folk dance sequences that we captured.

We show examples of virtual cameras created using our approach for these sequences in Figure 9 and Figure 13 .

Jiu-Jitsu Retreat: Jiu-Jitsu is a type of Brazilian Martial Art. We captured sequences of this sporting event during a summer retreat of the Pittsburgh Jiu-Jitsu group. This sequence is an extreme example of unchoreographed dynamic motion from more than 30 people who participated in it. Figure 7-2 shows the capture of a JiuJitsu event in the foreground with arbitrary human motion in a picnic in the background.

We show examples of virtual camera created for this sequence in Figure 12.

Tango: We captured sequences of Tango dance in an in-

[1]We have not described *per-pixel median* for the background in Section 3.2. It refers to the spatiotemporal accumulation done over median foreground images.

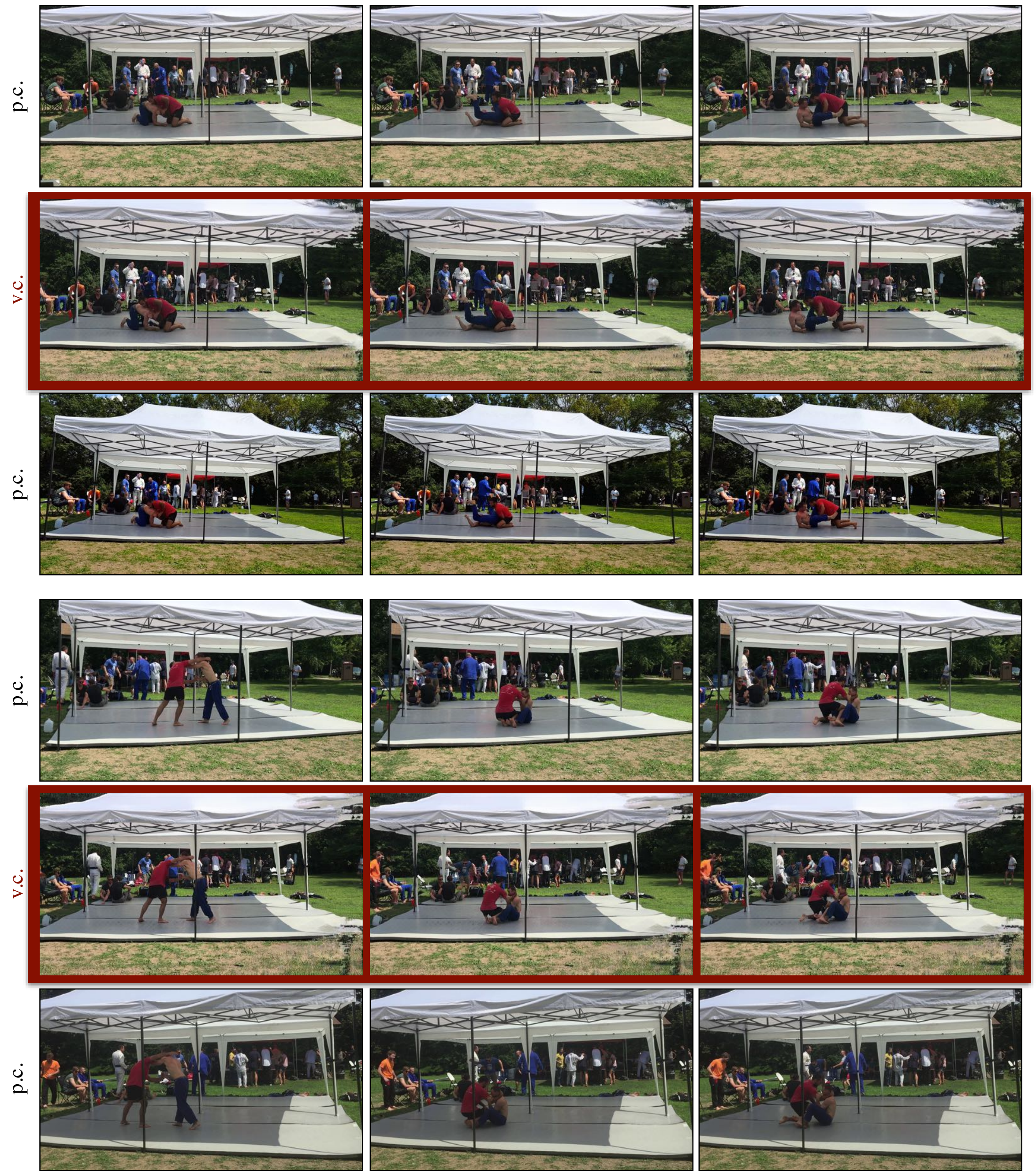

Figure 12. **Many People and Unchoreographed Sequence:** We show two examples of virtual cameras generated for Jiu-Jitsu Retreat sequence that is an example of capture with many people and unchoreographed event. For each virtual camera (v.c.), we show the physical cameras (p.c.) on its sides.

door environment. Both performers wore proper dress for Tango. Self-occlusion between the performers makes it challenging for 4D visualization. Shown in Figure 7-3 is an example of one of the four Tango dance sequences that we captured. Note the reflections on the semi-glossy ground and featureless surroundings.

Performance Dance: We captured many short performance dances including Ballet, and reenactments of plays. These sequences were collected inside an auditorium. The lighting condition, clothing, and motion change drastically in these sequences. Figure 7-4 shows an example of performance with an extremely wide baseline and challenging illumination.

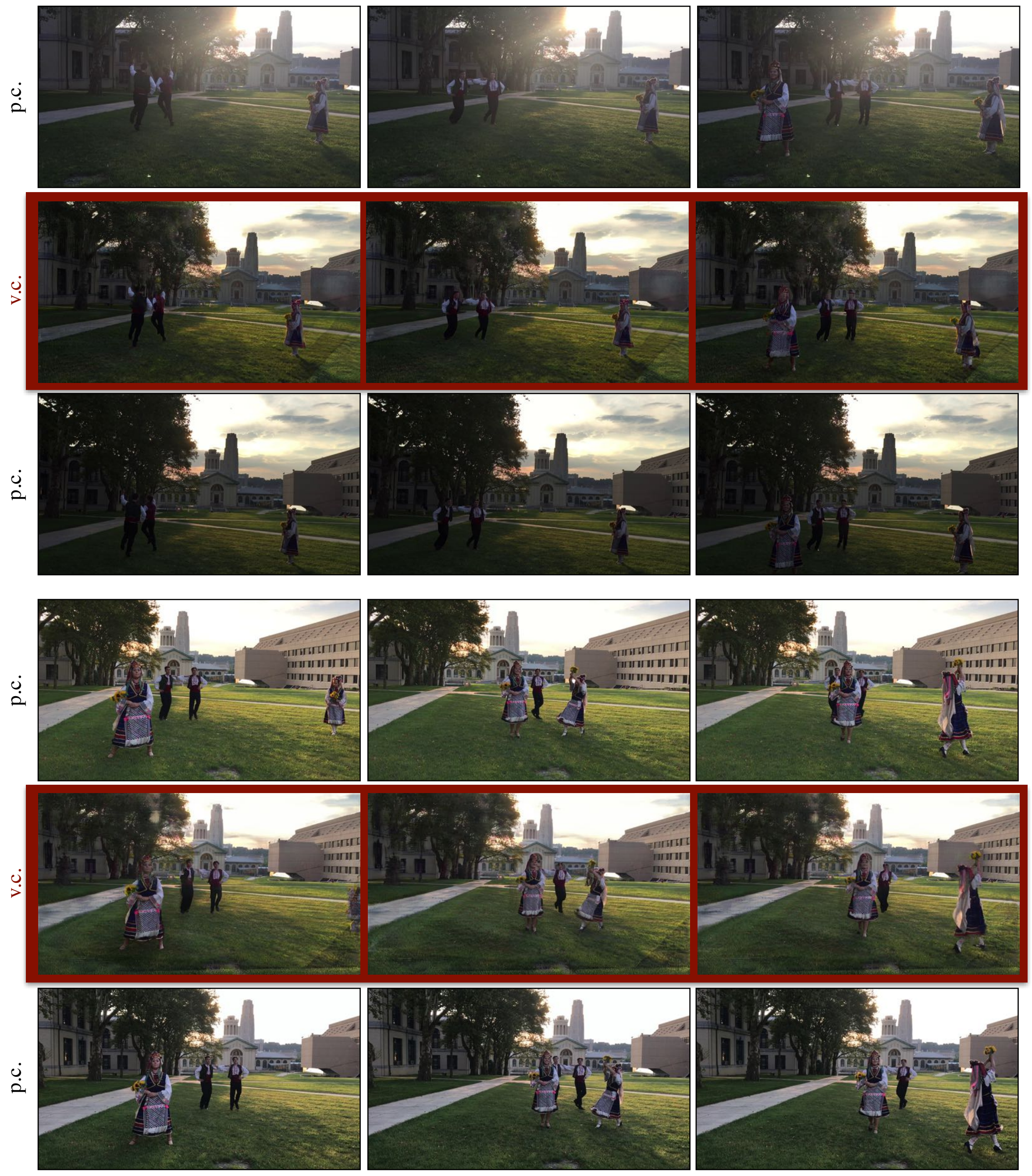

Figure 13. **Challenging Illumination and Multiple Dancers:** We show two examples of virtual cameras generated for another Western Folk Dance sequence with challenging illumination condition and self occlusion due to multiple dancers. For each virtual camera (v.c.), we show the physical cameras (p.c.) on its sides.

5.2. Bird Sequences

We captured a wide variety of birds at the National Aviary of Pittsburgh. We have no control on the motion of birds, their environment, lighting condition, and dynamism in the background due to human movement in the aviary.

Wetlands: The American Flamingos are the most popular wetlands bird at the National Aviary of Pittsburgh. In this sequence, we captured a free motion of many American Flamingos from multiple cameras. Slowly moving water with reflection of surroundings in which these birds live makes it challenging and appealing (Figure 8-1). There may be an occasional sight of Brown Pelicans and Roseate

Spoonbills in this sequence.
Penguins: There are around 20 African Penguins at the Penguin Point in National Aviary of Pittsburgh. Penguin Point consists of rocky terrain and a pool for penguins. The arbitrary motion of penguins and reflection in water makes this sequence totally uncontrolled. Figure 8-2 shows an example of a sequence captured at the Penguin Point. There is also a frequent movement of humans in this sequence making the background dynamic.
Tropical Rain-forests: There are a wide variety of tropical rain forest birds at the National Aviary of Pittsburgh. These include Bubba the Palm Cockatoo, a critically endangered parrot species, Gus and Mrs. Gus the Great Argus Pheasants, a flock of Victoria Crowned Pigeons, Southern Bald Ibis, Guam Rails, Laughing Thrushes, Hyacinth Macaws, and a two-toed Sloth. The dense trees in this section act as a natural occlusion while capturing the birds and makes it exciting for 4D capture. Figure 8-3 shows an example of a sequence that captured Victoria Crowned Pigeons.

6. Quantitative Analysis

We used sequences from Vo et al. [48] to properly compare our results with their 3D reconstruction (SfM+humans). Figure 2 and Figure 3 shows the results of freezing the time and exploring the views for these sequences.
Evaluation: We use a mean-squared error (MSE), PSNR, SSIM, and LPIPS [53] to study the quality of virtual camera views created using our approach. **MSE**: Lower is better. **PSNR**: Higher is better. **SSIM**: Higher is better. **LPIPS**: Lower is better. We use held-out cameras for proper evaluation. We also compute a **FID** score [24], lower the better, to study the quality of sequences where we do not have any ground truth (e.g., freezing the time and exploring views). This criterion contrast the distribution of virtual cameras against the physical cameras.

Baselines: To the best of our knowledge, there does not exist a work that has demonstrated dense 4D visualization for in-the-wild dynamic events captured from unconstrained multi-view videos. We, however, study the performance of our approach with: (1) a simple nearest neighbor baseline **N.N.**: We find nearest neighbors of generated sequences using conv-5 features of an ImageNet pre-trained AlexNet model. This feature space helps in finding the images closer in structure. Additionally, there are two kinds of nearest neighbors in our scenario. The first is the nearest camera view at the **same time** instant and the other is the nearest camera view across **all time** instants.; (2) **SfM+humans**: We use work from Vo et al [48, 49] for these results.; and finally (3) we contrast it with instantenous foreground image (**Inst**) that is defined in Section 3.1.

Table 1 contrasts our approach with various baselines on held-out cameras for different sequences. We observe significantly better outputs under all the criteria. We provide more qualitative analysis on our project page – http://www.cs.cmu.edu/~aayushb/Open4D/

Approach	M.S.E	PSNR	SSIM	LPIPS [53]	FID [24]
N.N (same-time)	5577.65 ±927.47	10.72 ±0.73	0.27 ±0.05	0.57 ±0.07	-
N.N (all-time)	4948.64 ±1032.76	11.29 ±1.00	0.31 ±0.04	0.50 ±0.11	-
SfM + Humans	3982.26 ±1050.28	12.25 ±1.02	0.33 ±0.08	0.45 ±0.022	190.84
Inst.	5956.25 ±3139.61	11.18 ±2.89	0.42 ±0.17	0.43 ±0.16	100.10
Ours	**714.95 ±364.99**	**20.14 ±2.21**	**0.79 ±0.07**	**0.13 ±0.06**	**21.98**

Table 1. **Comparison:** We contrast our approach with: (1). a simple nearest neighbor (**N.N.**) baseline. There are two kinds of nearest neighbors in our scenario. The first is the nearest camera view at the **same time** instant and the other is the nearest camera view across **all time** instants. (2). reconstructed outputs of **SfM+humans**; and finally (3). instantaneous foreground (**Inst**) defined in Section 3.1. We use various evaluation crietria to study our approach in comparisons with these three methods: (1). **M.S.E**: We compute a mean-squared error of the generated camera sequences using held-out camera sequences.; (2). **PSNR**: We compute a peak signal-to-noise ratio of the generated sequences against the held out sequences; (3). **SSIM**: We also compute a SSIM in similar manner.; (4). We also use LPIPS [53] to study structural similarity and to avoid any biases due to MSE, PSNR, and SSIM. Lower it is, better it is. Note that all the above four criteria are computed using held-out camera sequences; and finally (5) we compute a **FID**-score [24] to study the quality of generations when a ground-truth is not available for comparisons. Lower it is, better it is.

7. User-Controlled Editing

We have complete control of the 3D space and time information of the event. This 4D control allows us to browse the dynamic events. A user can see behind the occlusions if a certain information is visible in other views. A user can also edit, add, or remove objects. To accomplish this, a user marks the required portion in a video. Our approach automatically edits the content, i.e., update the background and foreground, via multi-view information — the modified inputs to stacked multi-stage composition results in desirable outputs. Importantly, marking on a single frame in the video is sufficient, as we can propagate the mask to the rest of the video (4D control of foreground). We show two examples of user-controlled editing in Figure 14. In the first example,

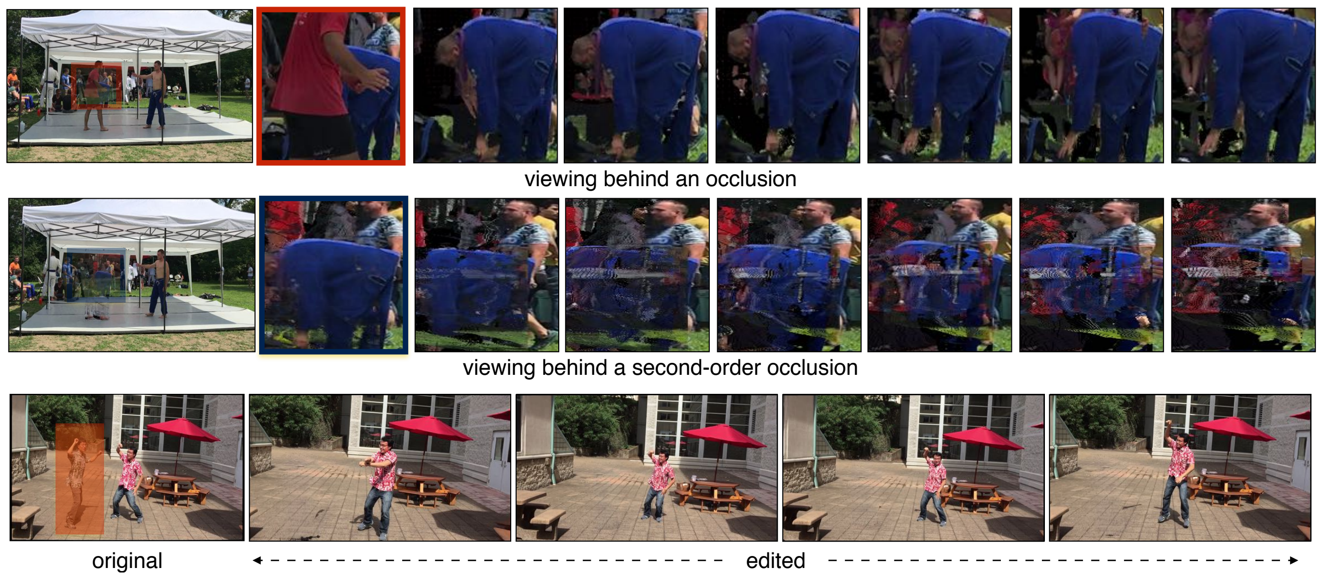
viewing behind an occlusion
viewing behind a second-order occlusion
original
edited

Figure 14. **User-Controlled Editing:** We show two examples of user-controlled editing in videos. In the **top**-row, a user selects a mask to see the occluded blue-shirt person (behind red shirt person). There is no way we can infer this information from a single-view. However, multi view information allows us to not only see the occluded human but also gives a sense of activity he is doing. We show frames from 2 seconds of video. In the **middle**-row, we want to see the part of scene behind the blue-shirt person who is *disoccluded* above. This is an example of a seeing behind a second-order occlusion. While not as sharp as first-order occlusion result, we can still see green grass and white bench in the background with a person moving. This particular scenario is not only challenging due to second-order occlusion but also because of larger distance from cameras. In the **bottom**-row, a user can remove the foreground person by marking on a single frame in video. Our system associates this mask to all the frames in video, and edit it to show background in place of human. We show frames of edited video (20 seconds long).

we enable a user to see occluded person without changing the view. Our system takes input of mask from the user, and *disocclude* the blue-shirt person (Figure 14-top-row). We also explore viewing behind a second-order occlusion. Figure 14-middle shows a very challenging example of viewing behind the blue-shirt person. Despite farther away from the camera, we see grass, white table, and a person moving in the output. Finally, we show an example of editing where a user can mark region in a frame of video (Figure 14-bottom-row). Our system generates full video sequence without the masked person.

8. Discussion & Future Work

The world is our studio. The ability to do 4D visualization of dynamic events captured from unconstrained multi-view videos opens up avenue for future research to capture events with a combination of drones, robots, and hand-held cameras. The use of self-supervised and scene-specific CNNs allows one to browse the 4D space-time of dynamic events captured from unconstrained multi-view videos. We extensively captured various in-the-wild events to study this problem. We show different qualitative and quantitative analysis in our study. A real-time user guided system that allows a user to upload videos and browse will enable a better understanding of 4D visualization systems. The proposed formulation and the captured sequences, however, open a number of opportunities for future research such as incorporating illumination and shadows in 4D spatiotemporal representation, and modeling low-level high frequency details. One drawback of our method is that the video streams are treated as perfectly synchronized. This introduces motion artifacts for fast actions [48]. Future work will incorporate sub-frame modeling between different video streams in depth estimation and view synthesis modules for more appealing 4D slow motion browsing.

Acknowledgements: We are extremely grateful to Bojan Vrcelj for helping us shape the project. We are also thankful to Gengshan Yang for his help with the disparity estimation code and many other friends for their patience in collecting the various sequences. We list them on our project page. This work is supported by the Qualcomm Innovation Fellowship, NSF CNS-1446601 and ONR N00014-14-1-0595 grant.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 3

[2] Luca Ballan, Gabriel J. Brostow, Jens Puwein, and Marc Pollefeys. Unstructured video-based rendering: Interactive

exploration of casually captured videos. *ACM Trans. Graph.*, 2010. 2, 3

[3] Luca Ballan and Guido Maria Cortelazzo. Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. 3DPVT, 2008. 3

[4] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018. 2

[5] Joel Carranza, Christian Theobalt, Marcus A. Magnor, and Hans-Peter Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 2003. 2, 3

[6] Edilson De Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. Performance capture from sparse multi-view video. In *ACM SIGGRAPH*. 2008. 3

[7] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *ACM Trans. Graph.* ACM, 1996. 3

[8] Emily L Denton, Soumith Chintala, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *NeurIPS*, 2015. 3

[9] N. Dinesh Reddy, Minh Vo, and Srinivasa G. Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In *CVPR*, 2018. 3

[10] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *CVPR*, 2019. 3

[11] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016. 3

[12] J-S Franco and Edmond Boyer. Fusion of multiview silhouette cues using a space occupancy grid. In *ICCV*, 2005. 3

[13] H. Fuchs, G. Bishop, K. Arthur, L. McMillan, R. Bajcsy, S. Lee, H. Farid, and Takeo Kanade. Virtual space teleconferencing using a sea of cameras. In *Proc. First International Conference on Medical Robotics and Computer Assisted Surgery*, 1994. 3

[14] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010. 3

[15] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 2009. 3

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 3, 6

[17] Jean-Yves Guillemaut, Joe Kilner, and Adrian Hilton. Robust graph-cut scene segmentation and reconstruction for free-viewpoint video of complex dynamic scenes. In *ICCV*, 2009. 3

[18] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 2

[19] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2, 3, 4, 5

[20] Nils Hasler, Bodo Rosenhahn, Thorsten Thormahlen, Michael Wand, Jürgen Gall, and Hans-Peter Seidel. Markerless motion capture with unsynchronized moving cameras. In *CVPR*, 2009. 3

[21] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 2018. 4

[22] Jared Heinly. *Toward Efficient and Robust Large-Scale Structure-from-Motion Systems*. PhD thesis, The University of North Carolina at Chapel Hill, 2015. 3

[23] Jared Heinly, Johannes Lutz Schönberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the World* in Six Days *(As Captured by the Yahoo 100 Million Image Dataset). In *CVPR*, 2015. 2, 3

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 13

[25] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. *CVPR*, 2018. 2

[26] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 3, 6

[27] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Scott Godisart, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social interaction capture. *IEEE TPAMI*, 2017. 2, 3

[28] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Trans. Graph.*, 2016. 3

[29] Takeo Kanade and PJ Narayanan. Historical perspectives on 4d virtualized reality. In *CVPR Workshops*, 2006. 2, 3

[30] Takeo Kanade, Peter Rander, and PJ Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE multimedia*, 1997. 3

[31] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *CVPR*, 1996. 2, 3

[32] Tejas Khot, Shubham Agrawal, Shubham Tulsiani, Christoph Mertz, Simon Lucey, and Martial Hebert. Learning unsupervised multi-view stereopsis via robust photometric consistency. *arXiv preprint arXiv:1905.02706*, 2019. 2

[33] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015. 3

[34] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Trans. Graph.*, 2015. 2

[35] Wojciech Matusik, Chris Buehler, Ramesh Raskar, Steven J Gortler, and Leonard McMillan. Image-based visual hulls. In *ACM Trans. Graph.*, 2000. 3

[36] Moustafa Meshry, Dan B. Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *CVPR*, 2019. 3

[37] Martin Oswald and Daniel Cremers. A convex relaxation approach to space time multi-view 3d reconstruction. In *ICCVW*, 2013. 3

[38] Raj Reddy. *Teleportation, Time Travel, and Immortality*. Springer New York, 1999. 1

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015. 6, 7

[40] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2

[41] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 2

[42] Harry Shum and Sing Bing Kang. Review of image-based rendering techniques. In *Visual Communications and Image Processing*, 2000. 3

[43] Sudipta N Sinha, Drew Steedly, Richard Szeliski, Maneesh Agrawala, and Marc Pollefeys. Interactive 3d architectural modeling from unordered photo collections. In *ACM Trans. Graph.* ACM, 2008. 3

[44] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 2006. 2, 3

[45] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *CVPR*, 2019. 3

[46] Sundar Vedula, Simon Baker, and Takeo Kanade. Image-based spatio-temporal modeling and view interpolation of dynamic events. *ACM Trans. Graph.*, 2005. 3

[47] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. In *ACM SIGGRAPH 2008*. 2008. 3

[48] Minh Vo, Srinivasa G. Narasimhan, and Yaser Sheikh. Spatiotemporal bundle adjustment for dynamic 3d reconstruction. In *CVPR*, 2016. 3, 4, 13, 14

[49] Minh Vo, Ersin Yumer, Kalyan Sunkavalli, Sunil Hadap, Yaser Sheikh, and Srinivasa Narasimhan. Automatic adaptation of person association for multiview tracking in group activities. *IEEE TPAMI*, 2020. 2, 13

[50] Gengshan Yang, Joshua Manela, Michael Happold, and Deva Ramanan. Hierarchical deep stereo matching on high-resolution images. In *CVPR*, 2019. 2, 4

[51] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *CVPR*, 2018. 2

[52] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017. 3

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 13

[54] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape-from-shading: a survey. *TPAMI*, 1999. 2

[55] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Trans. Graph.*, 2018. 3

[56] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 2004. 3