JAWS: Just A Wild Shot for Cinematic Transfer in Neural Radiance Fields

Xi Wang[1*] Robin Courant[1,2*] Jinglei Shi[3] Eric Marchand[1] Marc Christie[1]

[1]Inria, IRISA, CNRS, Univ. Rennes, [2]LIX, Ecole Polytechnique, IP Paris, [3]VCIP, CS, Nankai Univ.

xi.wang@inria.fr, robin.courant@polytechnique.edu, jinglei.shi@nankai.edu.cn

{eric.marchand, marc.christie}@irisa.fr

Abstract

This paper presents JAWS, an optimization-driven approach that achieves the robust transfer of visual cinematic features from a reference in-the-wild video clip to a newly generated clip. To this end, we rely on an implicit-neural-representation (INR) in a way to compute a clip that shares the same cinematic features as the reference clip. We propose a general formulation of a camera optimization problem in an INR that computes extrinsic and intrinsic camera parameters as well as timing. By leveraging the differentiability of neural representations, we can back-propagate our designed cinematic losses measured on proxy estimators through a NeRF network to the proposed cinematic parameters directly. We also introduce specific enhancements such as guidance maps to improve the overall quality and efficiency. Results display the capacity of our system to replicate well known camera sequences from movies, adapting the framing, camera parameters and timing of the generated video clip to maximize the similarity with the reference clip.

1. Introduction

Almost all film directors and visual artists follow the paradigm of *watch-and-learn* by drawing inspiration from others visual works. Imitating masterpieces (also known as visual homage) in their visual composition, camera motion or character action is a popular way to pay tribute to and honor the source work which influenced them. This subtly draws a faint, yet distinctive clue through the development of the whole film history. Examples are commonplace: across the dizzy effect of dolly-zoom in *Vertigo* to the scary scene in *Jaws* (see Fig. 1); or from the old school Bruce Lee's kung-fu films to blockbusters such as *Kill Bill* and *Matrix* series. The cinematic knowledge encompassed in reference sequences is therefore carried out and inherited through these visual homages, and have even been adapted
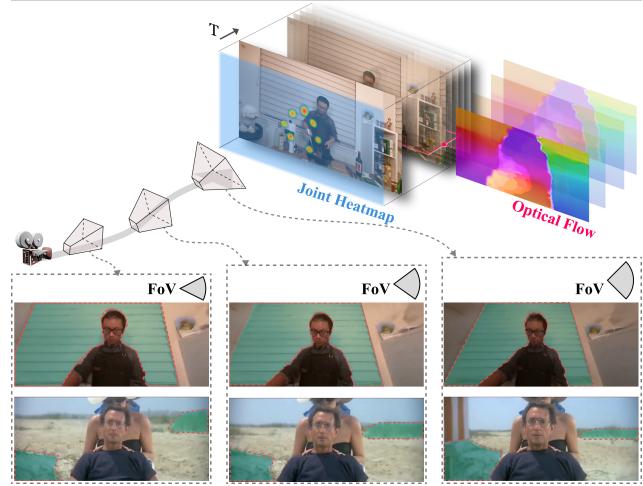

Figure 1. This illustration displays the capacity of our proposed method to transfer the cinematic motions from the in-the-wild famous film clip *Jaws* (bottom) to another context, replicating and adapting the dolly-zoom effect (a combined translation plus field of view change in opposite directions, causing distinctive motions on background and foreground contents).

to more modern visual media such as digital animation or video games. Audiences obviously acknowledge the strong references between the epic Western *The good, the bad and the ugly* of 1966 and the 2010 *Red Dead Redemption*.

The creation of such *visual homages* in real or virtual environments yet remains a challenging endeavor that requires more than just replicating a camera angle, motion or visual compositions. Such a *cinematic transfer* task is successful only if it can achieve a similar visual perception (*look-and-feel*) between the reference and homage clip, for example in terms of visual composition (*i.e.* how visual elements are framed on screen), perceived motion (*i.e.* how elements and scene move through the whole sequence), but also, camera focus, image depth, actions occurring or scene lighting.

Inspired by the aforementioned *watch-and-learn* paradigm, we propose to address the cinematic transfer problem by focusing on motion characteristics, *i.e.* solving a *cinematic motion transfer* problem.

---

*Equal contribution. Corresponding to jinglei.shi@nankai.edu.cn.

While different techniques are available to extract camera poses and trajectories from reference videos (*e.g.* sparse or dense localization and mapping techniques) and potentially transfer them, the naive replication of such trajectories to new 3D environments (mesh-based or implicit NeRF-based) generally fail to reproduce the perceived motion due to scaling issues, different screen compositions, lack of visual anchors, or scene dissimilarities. Furthermore, these extractiontechniques are very sensitive to widespread camera effects such as shallow depths of field or motion blur.

In this paper, we propose to address this cinematic motion transfer problem following a different path. Rather than extracting visual features from the reference clip (as geometric properties or encoded in a latent representation) and designing a technique to then recompute a new video clip using these visual features, we rely on the differentiable nature of NeRF representations. We propose the design of a fully differentiable pipeline which takes as input a reference clip, an existing NeRF representation of a scene, and optimizes a sequence of camera parameters (pose and focal length) in both space and time inside the NeRF so as to minimize differences between the motion features of the references views and the clip created from the optimized parameters. By exploiting an end-to-end differentiable pipeline, our process directly backpropagates the changes to spatial and temporal cinematic parameters. The key to successful cinematic motion transfer is then found in the design of relevant motion features and means to improve guidance in the optimization. Our work relies on the combination of an optical flow estimator, to ensure the transfer of camera directions of motions, and a character pose estimator to ensure the anchoring of motions around a target. A dedicated guidance map is then created to draw the attention of the framework on key aspects of the extracted features.

The contributions of our work are:

The first feature-driven cinematic motion transfer technique that can reapply motion characteristics of an in-the-wild reference clip to a NeRF representation.

The design of an end-to-end differentiable pipeline to directly optimize spatial and temporal cinematic parameters from a reference clip, by exploiting differentiability of neural rendering and proxy networks.

The proposal of robust cinematic losses combined with guidance maps that ensure the effective transfer of both on-screen motions and character framing to keep the cinematic visual similarity of the reference and generated clip.

2. Related work

Neural scene representation. Representing a scene efficiently and properly according to different applications, has long remained an open question in the fields of computer vision and graphics. Fueled by the development of deep learning in the past decade, typical scene representation methods such as mesh [1,2], voxel [3,4], point cloud [5] or light field [6,7] showed great progress in rendering photorealistic contents. More recently, the NeRF method [8] has drawn a huge attention from both industry and academia fields. NeRF proposes to learn attributes of light rays from multiple images, thereby encoding scene information in an implicit way using a deep neural network. Subsequent works [9–12] extended the NeRF-based methods to consider dynamic scenarios, where both temporal and spatial information are encoded by the network, enabling to synthesize images from any viewpoint and at any time.

In addition to the extension to dynamic scenes, many efforts have been made to improve network performance by either accelerating its training & rendering processes [13–16], deterring aliasing [17], extending bounds [18], reducing required samples [19,20]. All contribute to make NeRF-based methods a powerful scene representation approach for cinematic applications. It is noteworthy that NeRF scene representations can also serve as a differential environment to inversely optimize camera poses, as with iNeRF [21], where authors retrieve the reference images $SE(3)$ pose in a given NeRF scene description. Others [22,23] propose to extend the iNeRF model by addressing the localization and mapping problems. By then adding a functional network as a proxy, the whole workflow is capable of tackling more complex computer vision tasks [7,24,25]. NeRF networks can likewise work as a functional scene descriptor [26,27], followed by a predefined proxy network, to produce desired intermediate features for the final task.

Camera control and virtual cinematography. *Camera control* is a well established task in robotics that consists in exploiting sensor-acquired information to guide camera motion through an environment under some desired constraints, with an optimization framework. This topic covers a wide range of problems,*e.g.* visual servoing [28], viewpoint computation [29], or target tracking [30].

These seminal contributions have been largely extended to account for different visual quality metrics and constraints. Problems have also been transposed from real-environments (robotics) to virtual ones (computer graphics) by addressing virtual camera control problems [31].

While most contributions encoded motion properties (speed, jerk, optical flow) as constraints on the degrees of freedom of the camera (or of the camera trajectory path [32]), an increasing number of techniques have been exploiting real data to constrain/guide the camera motion. In [33], authors propose to perform camera motion style transfer by first extracting a camera trajectory from a given film clip using structure-from-motion techniques, performing a multi-frequential analysis of the motion, and regenerating the motion from its frequential parameters in a new 3D environment. This however only *replayed* the motion without adapting it the the target scene contents.

Later, through advances in deep learning techniques, contributions have started to train camera motion strategies from datasets of camera motions. Drone cinematography is a good example. Techniques such as Imitation Learning (IL) were used, combining the idea of Reinforcement Learning but training a model to learn from expert examples. In [34], the authors designed an IL system with simulated sensor noise to train drones to fly across in-the-wild agnostic scenarios.

Huang *et al*. [35] exploit optical flow and human poses to guide drone cinematography controls via an IL framework. Targeting similar drone cinematic objectives, [36] designed a DQN to control four directional actions for achieving tasks such as: obstacles avoidance, target tracking and shooting style application. Recently, Jiang *et al*. [37] trained a Toric-based feature extractor from human poses in synthetic and real film data for achieving cinematic style control via a pre-trained latent space when the 3D animation is known. Following the similar topic [38] tackled the keyframing problem which allows users to constrain the generated trajectory by a dedicated LSTM framework.

While these approaches do achieve some form of transfer of camera and motion characteristics, these are either dedicated to specific sub-fields (drone cinematography [35,36]), or focus on image composition [37,38] without considering the image motion characteristics.

3. Preliminary knowledge

Neural rendering. Our work strongly relies on NeRF representations of 3D scenes [8] which describe a scene through a Multiple Layer Perceptron (MLP) network by inputting different 3D locations and view directions, and inferring color and volume density. Volumetric rendering techniques then enable the reconstruction of an image from any viewpoint by retrieving the color and density of each pixel and integrating on the line direction of its pixel-correspondent rays in the MLP. A NeRF model $\mathcal{N}_{\Theta}$ can therefore be viewed as a mapping between a camera pose $\mathbf{T}$ and a reconstructed image $\hat{\mathbf{I}}$, where $\hat{\mathbf{I}} = \mathcal{N}(\mathbf{T})$ by abusing the term and regrouping all the rays into image pixels through a camera projective model: $\mathcal{N}_{\Theta} : SE(3) \rightarrow \mathbb{R}^{H \times W \times 3}; \mathbf{T} \mapsto \mathcal{N}(\mathbf{T})$.

In the training, parameters Θ are updated by minimizing an on-screen image loss (eg. photometric loss) on each ray, which compares pixel-level information of the reconstructed image $\mathcal{N}(\mathbf{T})$ and its associated reference view $\mathbf{I}^*$:

$$\hat{\Theta} = \arg\min_{\Theta} \mathcal{L}(\Theta \mid \mathcal{N}(\mathbf{T}), \mathbf{I}^*) \qquad (1)$$

Training a NeRF network can therefore be seen as optimizing MLP parameters Θ from a set of known camera poses and corresponding images.

Camera pose inference. The NeRF training process can also be applied in an inverse manner (see iNeRF [21]), to estimate a camera pose $\hat{\mathbf{T}} \in SE(3)$ in an already trained NeRF model, against a reference view $\mathbf{I}^*$. To ensure the estimated pose still lies in the $SE(3)$ manifold, the optimization process is performed in the canonical exponential coordinate system [39] based on an initial camera pose $\mathbf{T}(0)$:

$$\hat{\mathbf{T}}(\theta) = e^{[\xi]\theta} \mathbf{T}(0) \quad \text{where} \quad e^{[\xi]\theta} = \begin{bmatrix} e^{[\omega]\theta} & K(\theta, \omega, v) \\ 0 & 1 \end{bmatrix}$$

$$\text{with} \quad K(\theta, \omega, v) = \begin{cases} v\theta & \text{if} \quad \omega = 0 \\ \frac{(I - e^{[\omega]\theta})[\omega]v + \omega\omega^T v\theta}{\|\omega\|} & \text{otherwise} \end{cases} \qquad (2)$$

where $[\xi]$ is a twist matrix, (ω, v) are twist coordinates, and θ is a magnitude. The optimization problem is then changed to seek for the optimal camera parameters $(\hat{\theta}_k, \hat{\omega}_k, \hat{v}_k)$.

4. Method

4.1. Cinematic transfer

Inspired by the photorealistic recording capability of neural rendering methods (*e.g.* NeRF) and the related camera pose estimation methods such as iNeRF, in this paper we explore the possibility of proposing a NeRF-based *Cinematic Motion Transfer* from in-the-wild references, following the *watch-and-learn* paradigm explained in Sec. 1. Such a problem consists in extracting cinematic information from a reference clip and reapplying it in another scene s.t. the rendered visual content shares high cinematic similarity and effects. We propose our method, JAWS, displayed in Fig. 2, to address the *Cinematic Transfer* goal in a reference agnostic NeRF environment. The main idea is to optimize multiple cameras' cinematic parameters through the design of robust cinematic losses in a differentiable framework. To improve the performance of this optimization process, we extended our work with multiple techniques.

Formulation of the problem. Similar to the formulation of Eq. 1, we describe our problem as an inverted optimization. However, contrasting with iNeRF [21] which only optimizes a single camera pose and photometric loss, we target multiple cameras' cinematic parameters using two different, yet complementary, losses to ensure the transfer of key cinematic characteristics to synthesized clip. These robust cinematic losses are: (i) an *on-screen* loss $\mathcal{L}_s$, which guides the framing and position of significant elements; and (ii) an *inter-frame* loss $\mathcal{L}_i$, guiding the correct overall motion along the sequence.

We therefore re-formulate the problem as an inverted optimization of N cameras $\mathcal{T} = \{\mathbf{T_i}\}_N$ w.r.t. robust cinematic losses $\mathcal{L} = \mathcal{L}_s + \mathcal{L}_i$ between a set of synthesized views $\mathcal{N}(\mathcal{T})$ and reference film clip images $\mathcal{I}_r$:

$$\hat{\mathcal{T}} = \arg\min_{\mathcal{T}} \mathcal{L}(\mathcal{N}(\mathcal{T}) \mid \mathcal{I}_r, \Theta) \qquad (3)$$
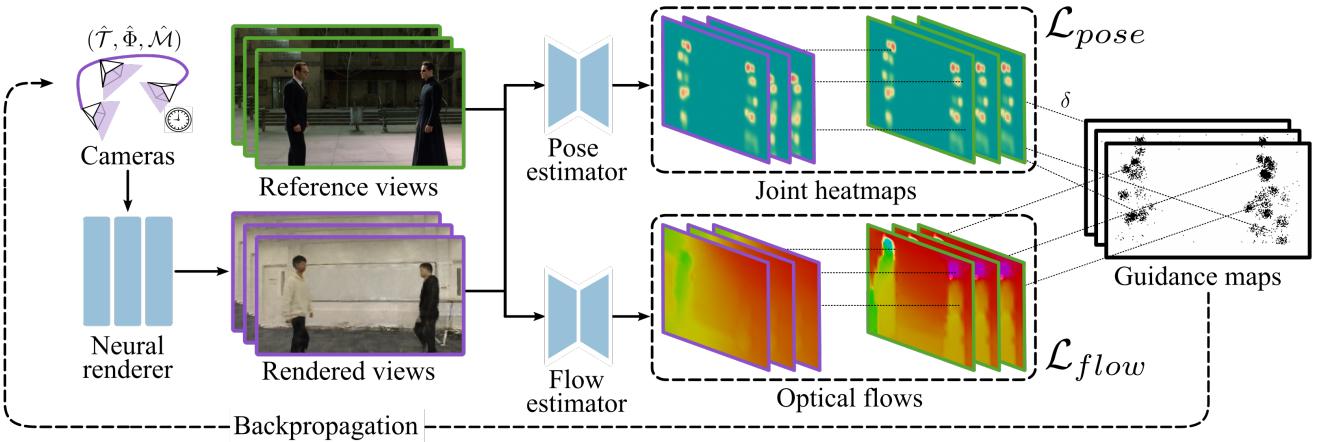
Figure 2. Overview of JAWS pipeline. Given the cinematic parameters (camera motion, focal length and timing parameters) to optimize $(\hat{\mathcal{T}}, \hat{\Phi}, \hat{\mathcal{M}})$, we first synthesize views through a fixed NeRF, then compute joint heatmaps and optical flows from the rendered (*purple*) and reference (*green*) views, via pose and flow estimators respectively. A guidance map (*black*) is calculated and helps the sampling of pixel with gradient s.t. cinematic parameters can be updated through a backpropagation tracing back to all proxy and NeRF networks.

The definition of cinematic camera parameters will be discussed in Sec. 4.1, and details of our proposed cinematic losses will be given in Sec. 4.2. We also propose specific enhancement techniques to further improve the efficiency of our system in Sec. 4.3.

Cinematic parameters. Film shooting cameras are usually more complex than relying on a $SE(3)$ camera model: variable focal length is often employed to realize many creative film effects, such as *whip zoom* (rapid zooming to close up on actors facial area to express surprise and draw sudden focus) or *dolly-zoom* (translating the camera in compensated direction to suggest strong emotional impact).

In addition to camera parameters, another factor we need to take into consideration is that film shooting always comprises the dynamic nature of actors and scene props, which are rarely discussed in most NeRF-based pose estimation works [21, 22, 40]. They only adopt static environmental hypothesis. It therefore appears mandatory to include extra timing parameter m representing the temporal condition in dynamic scenes, and focal length ϕ into our cinematic parameters with 6DoF camera pose, in order to address properly the cinematic motion transfer problem. Fortunately, thanks to the recent progress in extending static NeRF to the dynamic scene with the help of temporal deformation representation [9–11], dynamic NeRF methods can render unseen views at a specific instant (*i.e.* timing parameter m) for animated scenes or objects. One of our main intuitions behind trying to handle dynamic scenes is to rely on another MLP to encode warp information of rays such that the displacement of pixels can be retrieved from a canonical space (*i.e.* classic static NeRF methods). With this differentiable temporal MLP, we can follow the same optimization idea to further manipulate the timing of characters dynamic motion

or scenes changing in the neural rendering system.

We therefore include intrinsic parameters, especially the focal length for each camera $\Phi = \{\phi_i\}_N$, and the timing moments, *i.e.* temporal parameters $\mathcal{M} = \{m_i\}_N$ for dynamic scenes. We then transform the Eq. 3 into the following by relying our cinematic camera parameters:

$$\hat{\mathcal{T}}, \hat{\Phi}, \hat{\mathcal{M}} = \arg \min_{\mathcal{T}, \Phi, \mathcal{M}} \mathcal{L}(\mathcal{N}(\mathcal{T}, \Phi, \mathcal{M}) \mid \mathcal{I}_r, \mathbf{\Theta}) \quad (4)$$

4.2. Cinematic losses

Two challenges hinder the understanding and transferring of cinematic motion from reference clips to a different scene: (i) the difference of content in terms of the relative scale, mismatched appearance and even dissimilar dynamic motion of characters; (ii) the necessity to describe and compare the motions of the cameras between each other s.t. generated sequence shares high visual dynamic similarity.

As described in Sec. 4.1, we propose a combination of two different, yet complementary, optimization criteria, which we term as *on-screen* and *inter-frame* cinematic losses $\mathcal{L} = \mathcal{L}_s + \mathcal{L}_i$. Intuitively, they cover *framing* and *camera motion* aspects respectively.

On-screen loss. Our on-screen loss collects and transfers the on-screen information to ensure framing consistency. To this end, we decide to anchor the framing to the one of most significant entities on screen, human characters, like many previous works [35–37]. Exploiting character on-screen poses benefits from: (i) its sparse and robust information in comparison to pixel-wise criterion, and (ii) being descriptive enough to match corresponding key-regions between two views from different scenes. By contrast with similar tasks [35, 37], which only extract the final human pose keypoints, in our work we exploit the detection confidence
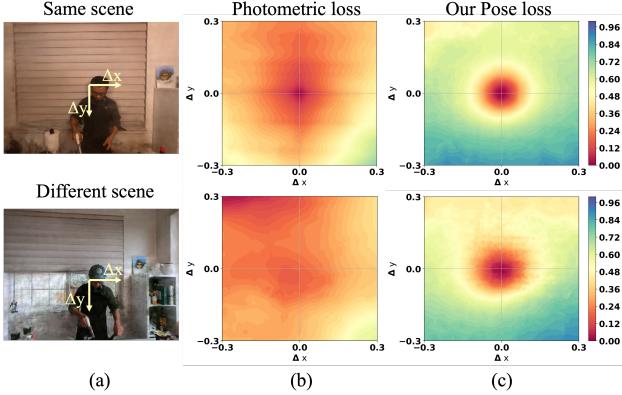
4

Figure 3. Normalized photometric loss *vs.* our pose loss under different camera position perturbations in the horizontal and vertical directions (x and y axis in col (b,c)) in the NeRF scene. First row shows loss convergence when applying the reference image to the same scene; and the second row shows results when testing the same reference under a different scene.

heatmaps predicted by a deep neural network. On the contrary to singular keypoints resulting from non-differentiable operations, those heatmaps are differentiable and contain richer information.

To compute a differentiable distance between two heatmaps, we use Wasserstein distance [41] (w-distance for short). The choice of w-distance aims to highlight the on-screen spatial relation (similar to on-image Euclidean joints distance) rather than MSE-like measures emphasizing on intensity difference (see more in [42]). By minimizing the w-distance between two character pose heatmaps, we are able to transfer the framing information from a target reference to a differentiable rendering space.

Lets consider a confidence heatmap $\mathbf{H} \in (\mathbb{R}^+)^{H \times W \times J}$ with J being joint number on human skeleton (see Fig. 2) generated from a pose estimation network, where each channel represents a detection probability of a given joint in the image domain. To calculate the character pose loss $\mathcal{L}_{pose}$, we compute w-distance d_w between confidence maps of a reference $\mathbf{H}^*$ and a synthetic view $\hat{\mathbf{H}}$ on each channel.

$$
\mathcal{L}_{pose} = \sum_{i=1}^{J} d_w(\mathbf{H}_i^*, \hat{\mathbf{H}}_i) + ||S(\mathbf{H}^*) - S(\hat{\mathbf{H}})||
$$
$$
\text{where} \quad S(\mathbf{H}) = \sum_{i=1}^{J} \sum_{j=1}^{J} d_w(\mathbf{H}_i, \mathbf{H}_j) \tag{5}
$$

The regularization S is defined as an inter-joint matrix measured in w-distance of each joint $\mathbf{H_j}$ to all the others from the *same* heatmap $\mathbf{H}$. This term assures the inter-joint shape similarity between heatmaps while optimizing the framing.

Fig. 3 illustrates the loss distribution when perturbing the camera pose around a reference position on x and y translational directions ($\Delta X, \Delta Y$), within same or different scene against the reference. We compare our proposed charac-

ter pose loss to the photometric loss, which is commonly used in inverted NeRF methods [21, 22]. Results demonstrate that the convergence cone of our loss preserves invariant while measuring the reference to a different content and provides correct human-centered framing information.

Inter-frame loss. On the other side, we present another criterion that depicts the relative camera motion performance along the sequence dimension of synthesized views.

In order to incorporate camera motion, we propose to extract the optical flow between two consecutive frames for estimating the similarity between reference and synthesized views. It takes advantages of (i) describing the motion within frames: enabling to infer indirectly the camera trajectory; and (ii) being agnostic towards frame content: allowing to compare views from different scenes.

In this work, we choose a differentiable deep neural network generated optical flow (see Fig. 2) for the purpose of back-propagating to the designed cinematic parameters through the neural rendering framework. To calculate the flow loss $\mathcal{L}_{flow}$, we compute the endpoint distance [43] between the reference $\mathbf{O}^*$ and synthetic $\hat{\mathbf{O}}$ sequences of flows:

$$
\mathcal{L}_{flow} = ||\mathbf{O}^* - \hat{\mathbf{O}}||_2 \tag{6}
$$

Optimization loss. The final loss for the optimization problem $\mathcal{L}_{total}$ is a linear combination of the pose and flow loss, corresponding to the *on-screen* and *inter-frame* loss termed in Eq. 4 respectively: $\mathcal{L}_{total} = \alpha \mathcal{L}_{pose} + \beta \mathcal{L}_{flow}$.

We adjust α and β during the optimization to achieve robust and dynamic emphasis on the two aspects separately.

4.3. Framework enhancements

Sampling strategy. Using pre-trained proxy networks to achieve high-level information is popular when solving complex computer vision tasks. The differentiable nature of neural networks allows the loss to be backpropagated to the upperstream networks such that high-level constraints can influence the whole system. Similarly, we design our on-screen and inter-frame losses, s.t. they are proxied by human pose [44] and optical flow [45] estimation networks.

However, two main factors hinder the direct connection of proxy networks to the NeRF-based methods: (i) backfire happens on the memory usage and computational complexity since the images from NeRF methods are composed in ray-pixel elements, each pixel links to all MLP parameters when back-propagating, yielding amplified memory according to pixel number; (ii) similar to avoiding textureless regions in computer vision tasks, our proposed loss is not uniformly distributed across the whole image field (especially the character pose loss). The divergence can be triggered by non-informative image regions.

Many NeRF-based camera estimation papers [21, 22] rely on sampling skills to accelerate the computation,
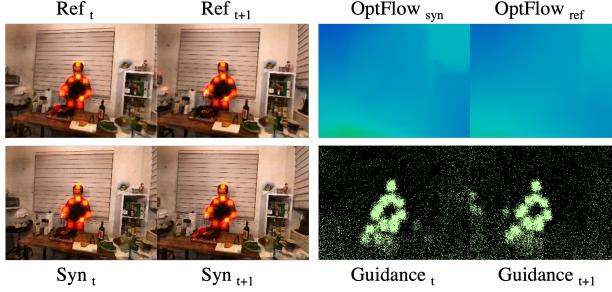
Figure 4. An example of computation of guidance maps (*bottom-right*) for two cameras respectively, via uniting the difference of heatmaps (*left*) and optical flows (*top-right*). High intensities on the guidance maps represent the sampled pixels *with* gradient.

lighten the memory and focus on relevant regions. Unfortunately, similar experiences can not be directly transplanted to our problem, since proxy networks expect *entire* input image rather than sparsely sampled pixels.

We therefore come up with our guidance map skill to address this problem: instead of keeping only sampled pixels, we detached the gradient w.r.t. a probabilistic guidance map to achieve: (i) lighter memory usage and fast computation; (ii) attention-like mechanism to help convergence focus on informative regions. More specifically, objective parameters are updated by referring gradients backpropagated from only a Gaussian sampled small portion of pixels according to the intensity from guidance map during the optimization, whereas *all* pixels contribute during the inference of proxy networks. In contrast to iNeRF [21] using keypoint-driven regions on color image, our guidance map **G** (Fig. 4) is computed as united differential map between the generated human pose heatmaps and optical flows in order to highlight the key-regions contributing to the gradient:

$$\mathbf{G} = f_n(|\mathbf{O}^* - \hat{\mathbf{O}}|) + f_n(|\mathbf{H}^* - \hat{\mathbf{H}}|) \qquad (7)$$

where f_n is min-max normalization, and $|.|$ the absolute value.

5. Experiments

Implementation details. Our implementation is based on 'torch-npg' [46] and Pytorch [47], with Adam optimizer [48] and 'GradNorm' algorithm [49] to adjust loss weights during the optimization. We used InstantNGP [16] and D-NeRF [9] for neural rendering, RAFT [45] for the optical flow estimation and LitePose [44] to infer the joint heatmaps. Cameras are optimized two by two, and linked with a sliding window. For more details see Suppl. material.
Datasets. We test on multiple heterogeneous datasets to show the effectiveness and generality of our proposed method: including Blender-made dynamic dataset from D-NeRF [9], multi-view light-field-like video dataset

from [50, 51], mobile phone recorded video from nerfstudio [52], and our Unity synthesized animation renderings.

5.1. Qualitative results

Movies to NeRF. Fig. 5 shows examples of cinematic transfers from a classic shots to synthesized clips. Reference keyframes are sampled for optimizing the proposed cinematic parameters, interpolation is then applied on calculated parameters for high fps of final rendering.
In Fig. 5: (a) highlights the ability of our pipeline to search for the optimal focal length in addition to the camera pose: making it possible to reproduce the well-known *Vertigo* effect, i.e. pushing forward and zooming out. With the parameters evolution (below *frames*), we show that the focal length (*blue*) and the z-translation (*green*) tend to compensate each other, realizing the visual effect of "retreating background and forwarding foreground"; (b) shows that JAWS can also reproduce complex arc motion, composed of yaw (z-rotation, *grey*) and y-translation (*purple*) at the same time; (c) demonstrates the aspect to mimic subtle camera motions, such as handheld movements. Evolution of x- and z-translation (*brown* and *green*) present the shaky effect with irregular progressions; (d) illustrates that our pipeline can align the temporal axis to fit synthesized content with the reference. The timing parameter (*red*) increases with the z-translation (*green*), s.t. both characters approach to each other while the camera is forwarding, to present similar framing to the reference. In addition to the shown examples, more demos are available in the Suppl. material.
Movies to 3D engine. We propose a possible application of our method on animation workflow: using JAWS to generate referenced camera trajectories to guide the animation shooting in graphics engine *e.g.* Unity3D. It consist of: (i) training a NeRF by multi-view images rendered by Unity3D; (ii) generate desired cinematic transfer trajectory from JAWS according to a reference clip; (iii) re-apply the generated trajectory into Unity3D for rendering high quality animation. The advantage of combining two systems is to bridge the indifferentiability in graphics engine and lower animation quality in dynamic NeRF. See Fig. 6 for the rendered results presenting high visual similarity with the original reference, and the exported trajectory can be easily used by artists as a reasonable starting point of their workflow.

5.2. Ablation study

Loss ablation. In order to demonstrate the effectiveness of each component and design of our system, we mainly test three aspects of our proposed method:
(i) a video copying task is first undertaken, consists in transferring a NeRF rendered reference video (20 keyframes) which the trajectory is known, under the same and different scenes (see Fig. 7), with same/close and different camera initialization positions. The objective is to demonstrate

Figure 5. Examples of movies to NeRF transfers (*frames*), and below the evolution along clips of: time (*red*), focal length (*blue*) and x-translation (*brown*), y-translation (*purple*), z-translation (*green*) and yaw (*grey*). (a) a dolly-zoom in *Jaws*; (b) an arc in *House of Cards*; (c) handheld pull-out in *Inception*; and (d) a push-in in *The Matrix Reloaded*. x is along the width, y along the height, and z along depth.

	Same scene						Different scene 1				Different scene 2				Different scene 3			
	Same init			Different init			Close init		Different init		Close init		Different init		Close init		Different init	
Loss	$ATE\downarrow$	$JE\downarrow$	$PE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$PE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$ATE\downarrow$	$JE\downarrow$	$ATE\downarrow$	$JE\downarrow$
inerf [21]	1.70	2.57	0.17	2.63	2.58	0.23	/	/	/	/	24.97	19.37	32.58	145.27	32.37	24.00	32.04	47.34
pixel	3.00	2.82	0.19	7.85	2.33	0.31	/	/	/	/	35.98	121.99	36.60	19.38	/	/	30.81	29.23
pose	32.55	3.66	2.31	39.94	5.47	2.36	28.82	17.88	25.30	7.02	31.67	16.06	32.55	19.86	36.30	21.93	31.83	20.12
flow	14.83	3.23	1.10	33.54	17.22	4.38	19.23	18.10	21.37	28.47	7.47	12.79	19.44	19.44	15.79	29.02	19.80	33.82
flow+pose	9.14	2.86	1.14	14.90	3.59	2.05	14.78	6.27	14.02	7.00	13.78	8.45	19.39	9.91	15.34	7.62	14.93	10.93

Table 1. Ablation of losses. We report absolute pose trajectory (*ATE*) in cm (alignment [53] is applied for the trajectory from different scenes), average joints error (*JE*) in pixel and image pixel-wised error (*PE* (10^{-2})). **/** means experiment fails to finish the trajectory.

Figure 6. we show that the neural rendered results share high *look-and-feel* quality to the original reference (see (a)), and the exported trajectory (b) can be easily manipulated by artists.

Figure 7. Scenes tested during the ablation study.

the robustness and characteristics of our two complementary cinematic losses across different scenes and influenced by different level of perturbations (*i.e.* initial position).

(ii) we carry out an experiment for retrieving correct timing and focal length information from images rendered with known parameters respectively. This experiment is to prove the ability of recovering these parameters by inverted optimization method on dynamic NeRF networks.

(iii) we investigate the influence of the guidance map by collecting the motion performance and memory usage of different sampling number (*i.e.* number of pixels with gradient) to demonstrate that the guidance can help the convergence and mitigate the memory usage simultaneously.

Tab. 1 reports the ablation of losses for different setups: (i) same/different target scenes to the reference to emphasize the cross-domain transfer ability; (ii) same/different initial positions to highlight the robustness. Three metrics are computed: Absolute Trajectory Error (RMSE-ATE) reflects the quality of retrieved motion on $SE(3)$ similar to

many tracking works [22, 23]. To depict the on-screen composition similarity, we measure Pixel Error (PE) (for the same scene) and average Joint Error (JE) in pixel computed by Litepose (with a more accurate model).

We study several losses combinations: iNeRF [21], *i.e.* pixel loss with keypoint-driven sampling (w/o guidance map); pixel loss [1], pose and flow loss followed by combination of flow and pose losses (*flow+pose* in Tab. 1).

According to Tab. 1, we can observe: (i) iNeRF and pixel losses behave similarly: they both show good performance and robustness against perturbation for the same scene on motion (ATE) and composition qualities (PE, JE). However pixel-based methods fail frequently (*i.e.* camera moving to non-defined area of the NeRF and yielding numerical error) under different scenes. For the barely succeeded experiments, the performances are low due to the misled pixel information by mismatched appearance from different scenes. (ii) Comparing to pixel-based methods, the others (*pose*, *flow*, *flow+pose*) show invariance against appearance changing. Nevertheless, they all act differently: (a) flow loss tends to drift heavily if the initialization position is far to the correct one (JE and PE). The phenomenon is due to the fact that the flow focus on the inter-frame information and extracts no hint on the compositing; (b) in contrast, pose loss completely ignores the inter-frame motion and causes lower performance on ATE yet relatively better results in PE and JE on the same scene, suggesting possible ambiguities on similar human pose and different camera parameters. Heatmap pose feature also shows robustness against initial perturbation, with a small difference of ATE between the close and the different initializations; (c) by combining the two complementary losses: pose and flow, we achieve overall better performances than pose and flow separately, especially under different scene condition. Yet under the same scene, the performance is lower than pixel level tracking (iNeRF and pixel), reflecting on PE and ATE this is due to less sharpen convergence cone (see Fig. 3) limited resolution on extracted heatmap, and possible ambiguities.

Parameter ablation. We also undertake ablation for parameters: time m and focal length ϕ. The experiment is done by applying our method on 6-frames clips, where only time or focal length varies. Each experiment is run 16 times with a random initial value for the studied parameters (m or ϕ). For each studied parameter, we show in Fig. 8 the distribution of the error between the prediction and the ground-truth (*red*). We compare our results with the controlled case where the studied parameter remains constant, *i.e.* equals to the random initial value, all along the 6-frames clips (*blue*). Fig. 8 (i) shows the result for the time ablation, where only the arms of the mutant waving and Fig. 8 (ii) shows the re-
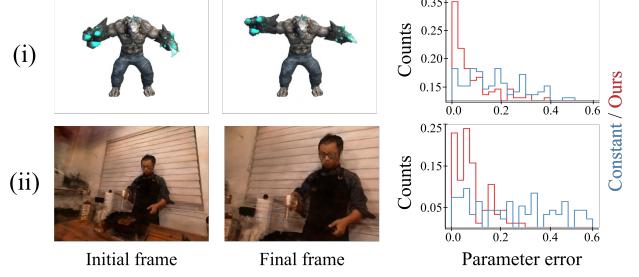
Figure 8. Visualization of (i) temporal ablation, and (ii) focal length ablation. We show the initial and final frames of the six-frames clips used for ablation. The right column shows the error distribution on the ablated parameter (time and focal length) using our method (red) and a constant parameter setup (green).

No. Guidance	500	1000	2000	4000	6000	8000	10000	32000	66752
Avg APE (cm)	35.0	27.6	25.3	21.0	23.4	29.8	32.6	23.0	14.3
Std APE (cm)	0.70	0.88	0.80	0.77	0.42	1.61	1.99	0.55	0.60
Mem usage (MB)	7273	7439	7457	7849	8339	8871	9121	14071	21151

Table 2. We show the influence of guidance map sampling number to the performance and memory usage.

sult for the focal length ablation, camera pose is fixed but the focal length varies. For both examples, our prediction error (*red*) is smaller than the controlled one (*blue*). Some large error values for our method are due to extreme initial points, making the convergence harder. Since we are optimizing *all* cinematic parameters, some effects may have ambiguities with other parameters than the studied one.

Guidance ablation. Tab. 2 shows the influence of the guidance map on the performance and memory usage. The experiment is done in similar methodology to the Tab. 1 under the same scene with different initial position (see more in Suppl. Material). Low ATEs are reported when the sampling number is insufficient. An experimental optimum is around 4000 as we used for our method. We notice higher std and memory usage when the sampling number keeps increasing. The performance re-improves when including the gradient of *all* pixel in the image (66752). The low-yield behavior of higher sampling numbers could be due to the confused the gradient direction by non-informative pixels.

6. Discussion

Despite good performance and robustness across different scenes, our method still shows limitations from: (i) highly mismatched character poses may compromise inter-frame motion and lead to undesired results; (ii) the temporal resolution and duration of D-NeRFs are still insufficient compared to graphics engine. Generating complex and long trajectory wrt active character motion can be laborious.

References

[1] C. Zhang, B. Miller, K. Yan, I. Gkioulekas, and S. Zhao. Path-space differentiable rendering. *ACM Trans. Graph.*, 2020. 2

[2] M. Nimier-David, S. Speierer, B. Ruiz, and W. Jakob. Radiative backpropagation: an adjoint method for lightning-fast differentiable rendering. *ACM Trans. Graph.*, 2020. 2

[3] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 2019. 2

[4] Y. Jiang, D. Ji, Z. Han, and M. Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[5] L. Li, S. Zhu, H. Fu, P. Tan, and C. Tai. End-to-end learning local multi-view descriptors for 3d point clouds. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[6] J. Shi, X. Jiang, and C. Guillemot. Learning fused pixel and feature-based view reconstructions for light fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[7] B. Mildenhall, P. Srinivasan, R. Ortiz-Cayon, N. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graph.*, 2019. 2

[8] B. Mildenhall, P. Srinivasan, M. Tancik, J. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 2, 3

[9] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2, 4, 6

[10] W. Xian, J. Huang, J. Kopf, and C. Kim. Space-time neural irradiance fields for free-viewpoint video. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2, 4

[11] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2, 4

[12] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[13] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Int. Conf. Comput. Vis.*, 2021. 2

[14] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[15] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[16] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 2022. 2, 6

[17] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Int. Conf. Comput. Vis.*, 2021. 2

[18] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2

[19] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Int. Conf. Comput. Vis.*, 2021. 2

[20] B. Roessle, J. T. Barron, B. Mildenhall, P. P. Srinivasan, and M. Nießner. Dense depth priors for neural radiance fields from sparse input views. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2

[21] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T. Lin. inerf: Inverting neural radiance fields for pose estimation. In *IEEE Int. Conf. on Intelligent Robots and Systems*, 2021. 2, 3, 4, 5, 6, 7, 8

[22] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 2, 4, 5, 8

[23] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison. iMAP: Implicit mapping and positioning in real-time. In *Int. Conf. Comput. Vis.*, 2021. 2, 8

[24] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Eur. Conf. Comput. Vis.*, 2018. 2

[25] S. Gao, F. Huang, W. Cai, and H. Huang. Network pruning via performance maximization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[26] M. Kirill, S. Edgar, and D. Andrew. Feature-realistic neural fusion for real-time, open set scene understanding. In *arXiv*, 2022. 2

[27] F. Li, H. Yu, I. Shugurov, B. Busam, S. Yang, and S. Ilic. Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation. *arXiv preprint arXiv:2203.04802*, 2022. 2

[28] E. Marchand and N. Courty. Controlling a camera in a virtual environment. *The Visual Computer*, 2002. 2

[29] T. Bill and L. Christian. Automatic camera placement for robot vision tasks. *IEEE Int. Conf. on Robotics and Automation*, 1995. 2

[30] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. *IEEE Trans. on Robotics and Automation*, 1993. 2

[31] Marc Christie, Patrick Olivier, and Jean-Marie Normand. Camera control in computer graphics. In *Computer Graphics Forum*. Wiley Online Library, 2008. 2

[32] Hui Huang, Dani Lischinski, Zhuming Hao, Minglun Gong, Marc Christie, and Daniel Cohen-Or. Trip synopsis: 60km in 60sec. In *Computer Graphics Forum*. Wiley Online Library, 2016. 2

[33] C. Kurz, T. Ritschel, E. Eisemann, T. Thormählen, and H-P. Seidel. Camera motion style transfer. In *IEEE Conf. on Visual Media Production*, 2010. 2

[34] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza. Learning high-speed flight in the wild. *Science Robotics*, 2021. 3

[35] C. Huang, C. Lin, Z. Yang, Y. Kong, P. Chen, X. Yang, and K. Cheng. Learning to film from professional human motion videos. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 3, 4

[36] R. Bonatti, W. Wang, C. Ho, A. Ahuja, M. Gschwindt, E. Camci, E. Kayacan, S. Choudhury, and S. Scherer. Autonomous aerial cinematography in unstructured environments with learned artistic decision-making. *Journal of Field Robotics*, 2020. 3, 4

[37] H. Jiang, B. Wang, X. Wang, M. Christie, and B. Chen. Example-driven virtual cinematography by learning camera behaviors. *ACM Trans. Graph.*, 2020. 3, 4

[38] H. Jiang, M. Christie, X. Wang, L. Liu, B. Wang, and B. Chen. Camera keyframing with style and control. *ACM Trans. Graph.*, 2021. 3

[39] Y. Ma, S. Soatto, J. Košecká, and S. Sastry. *An invitation to 3-d vision: from images to geometric models*. Springer, 2004. 3

[40] G. Avraham, J. Straub, T. Shen, T. Yang, H. Germain, C. Sweeney, V. Balntas, D. Novotny, D. DeTone, and R. Newcombe. Nerfels: Renderable neural codes for improved camera pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 4

[41] L. V. Kantorovich. Mathematical methods of organizing and planning production. *Management science*, 1960. 5

[42] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Int. conf. on Machine Learning*, 2017. 5

[43] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *Int. J. Comput. Vis.*, 2011. 5

[44] Y. Wang, M. Li, H. Cai, W. Chen, and S. Han. Lite pose: Efficient architecture design for 2d human pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 5, 6

[45] Z. Teed and J. Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Eur. Conf. Comput. Vis.*, 2020. 5, 6

[46] J. Tang. Torch-ngp: a pytorch implementation of instant-ngp, 2022. https://github.com/ashawkey/torch-ngp. 6

[47] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inform. Process. Syst.*, 2019. 6

[48] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[49] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Int. Conf. on Machine Learning*, 2018. 6

[50] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, R. Newcombe, et al. Neural 3d video synthesis from multi-view video. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 6

[51] J. Zhang, X. Liu, X. Ye, F. Zhao, Y. Zhang, M. Wu, Y. Zhang, L. Xu, and J. Yu. Editable free-viewpoint video using a layered neural representation. In *ACM Trans. Graph.*, 2021. 6

[52] M. Tancik*, E. Weber*, E. Ng*, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa. Nerfstudio: A framework for neural radiance field development, 2022. 6

[53] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1991. 7