# EpipolarNVS: leveraging on Epipolar geometry for single-image Novel View Synthesis

Gaëtan Landreau[1,2]
gaetan.landreau@meero.com
Mohamed Tamaazousti [2]
mohamed.tamaazousti@cea.fr

[1] Meero
Paris, France

[2] Université Paris-Saclay,
CEA-LIST,
F-91120 Palaiseau, France

**Abstract**

Novel-view synthesis (NVS) can be tackled through different approaches, depending on the general setting: a single source image to a short video sequence, exact or noisy camera pose information, 3D-based information such as point clouds etc. The most challenging scenario, the one where we stand in this work, only considers a unique source image to generate a novel one from another viewpoint. However, in such a tricky situation, the latest learning-based solutions often struggle to integrate the camera viewpoint transformation. Indeed, the extrinsic information is often passed as-is, through a low-dimensional vector. It might even occur that such a camera pose, when parametrized as Euler angles, is quantized through a one-hot representation. This vanilla encoding choice prevents the learnt architecture from inferring novel views on a continuous basis (from a camera pose perspective). We claim it exists an elegant way to better encode relative camera pose, by leveraging 3D-related concepts such as the epipolar constraint. We, therefore, introduce an innovative method that encodes the viewpoint transformation as a 2D feature image. Such a camera encoding strategy gives meaningful insights to the network regarding how the camera has moved in space between the two views. By encoding the camera pose information as a finite number of coloured epipolar lines, we demonstrate through our experiments that our strategy outperforms vanilla encoding.

## 1 Introduction

Synthesise a novel and realistic image from another viewpoint based on single or multiple images and some camera pose information commonly referred to as novel view synthesis. It has tremendous applications, from a pure computer vision and graphics perspective (such as cinemagraph, video stabilisation or 3D-based virtual staging) to Augmented and Virtual Reality (AR/VR).

This problem can be addressed from different perspectives, depending on the available data: multiple source images, depth maps, 3D labels such as 3D scene point cloud, accurate or jittered camera poses etc. We considered in this work one of the most extreme cases for the novel-view synthesis issue. Our work constrains the prediction of a scene from a

novel viewpoint by solely leveraging a single source image and the corresponding camera viewpoint transformation.

Currently, efforts are oriented toward getting the most visually appealing results on novel view synthesis, mainly through NeRF-based methods [1, 2, 11, 19]. However, only a few works pursue another tricky challenge in the novel-view synthesis: investigating the most efficient way to condition an NVS architecture on camera pose information. From a general perspective, single-image novel-view methods often restrict the viewpoint transformation to the extrinsic matrices. Intrinsic is thus discarded since most methods around single-image novel-view synthesis ignore physical image formation properties and do not account for rendering, epipolar geometry or homography concepts. Such camera pose conditioning task remains too weakly addressed in the current literature, which motivated us to tackle such an issue elegantly.

While one of the most straightforward solutions to do so consists in encoding the relative camera viewpoint transformation as a feature vector, we claim such a method is sub-optimal, supported by one of the latest state-of-the-art works in monocular depth prediction [23]. We thus propose in this work an elegant solution to encode the camera relative transformation as a 2D feature RGB image, by leveraging epipolar constraints. The new and implicitly encoded camera viewpoint transformation has a similar spatial resolution as the source image, somehow filling the dimensional gap between pose matrices and the RGB space. The contribution we propose in this paper is thus three-fold:

- A strategy to encode the camera pose transformation into an implicit feature image builds upon epipolar geometry considerations.
- A neural network architecture which leverages such camera viewpoint transformation encoding.
- A spectral loss function that extensively accounts for the higher frequencies of an image to better retrieve tiny and complex details.

## 2 Related work

**Novel view synthesis in a large extent.** Since modalities involved in novel-view synthesis are broad (images, video sequence, 3D point clouds, depth and disparity maps, camera poses), tremendous approaches investigated ways to tackle such issues. One of the latest trends takes advantage of Neural Radiance Fields (NeRF [11]) and their incredible powerfulness to generate highly realistic scenes from unseen viewpoints [2, 12, 19]. However, such architectures are somehow over-fitted over a unique scene and do not have any generalisation abilities. Such a drawback is growingly overcome by recent works [10, 22] that tackle novel view synthesis through the prism of NeRF-based architectures. While MINE [10] is a Multiplane Images (MPI) based method that requires accurate ground truth disparity maps, PixelNeRF [22] works with several input views to refine the predicted novel view. Another important line of work in novel view synthesis for indoor navigation, with datasets such as RealEstate10K [25] or Matterport3D [3], produces extremely appealing results with complex architectures [13, 14, 21], often at the expense of complex information to get, such as ground truth depth maps or dense 3D point clouds.

**Camera pose encoding.** Camera pose is compactly represented through: 3 degrees of rotations around each world axis, 3 other ones for translation (both defining the extrinsic matrix) and a few additional ones when intrinsic must be considered (focal length, sensor size etc). One of the most straightforward solutions encodes the viewpoint transformation by computing camera poses difference [16]. Another pose encoding strategy embeds such a low-dimensional camera pose into a higher space, as in [9, 14]. These last two possibilities can be simplified if one wants to consider one-hot vectors for camera pose encoding. Finally, the closest work to ours regarding camera pose encoding is [23], which encodes the camera location (parametrized through a roll and pitch angles as well as a fixed height above a ground plane) as a 2D feature image for depth maps prediction purpose.

**Camera pose conditioning.** Extrinsic camera pose is thus often the unique 3D prior information that conditions the neural network for generating a novel view. Based on the camera pose difference $P_{diff} = P_{target} - P_{source} \in \mathbb{R}^v$, the authors from [16] tiled such vector across all the pixels of the source RGB image, feeding their CNN-based architecture with inputs that size $\mathbb{R}^{H \times W \times (3+v)}$. On the other hand, [9] adopts a different strategy and concatenates its camera pose feature vector with the one obtained from their CNN-encoder before feeding it to the decoder. Finally, [21] designed a single image novel-view synthesis method that extensively relies on 3D point cloud consideration. Camera viewpoint transformation is used within the network architecture to update the predicted point cloud before rendering.

**Single-image novel view synthesis.** Such a framework is the most challenging one since only minimal information is available during training and inference: a source image and a corresponding camera pose transformation that accounts for the target image we aim to generate. To the best of our knowledge, only a few recent works [9, 16, 22] deal with such a highly constrained setting. While [9, 16] both handle "discrete" (from ShapeNet [4], parametrized through a unique azimuthal angle, the elevation one being fixed) and continuous (as in Synthia [15] and KITTI [7]) camera transformations, the pose-feature vector needs to be updated accordingly from a size perspective. This is one of the main benefits of the method we designed. Both discrete and continuous camera information is encoded through a featured image that sizes the same as the source image and that truly leverages the real-world camera transformation that occurred between the source and the target view. Such convenient property allows to inferring (at least with discrete camera poses) viewpoints that were not represented within the training set.

# 3 Method

## 3.1 Camera viewpoint transformation encoding

### 3.1.1 Epipolar geometry overview

The general framing of our work might be considered one of the trickiest ones in novel-view synthesis since the image generation from a different camera viewpoint is only made prior to a single source image and a relative camera transformation.

We denote by $I_s \in \mathbb{R}^{H \times W \times 3}$ the RGB source image and $I_t$ the target one we aim to predict. The pinhole convenient camera model that we get consideration for is represented through an intrinsic matrix $K \in \mathbb{R}^{3 \times 3}$. The rigid motion that accounts for the relative transformation

between the source and the target view consists of a rotation $R \in SO(3)$ and translation $T \in \mathbb{R}^{3\times1}$, expressed through each camera's extrinsic:

$$\begin{cases} R = R_t R_s^T \\ T = t_t - R t_s \end{cases} \tag{1}$$

with $(R_s, t_s)$ and $(R_t, t_t)$ respectively accounting for the source and target camera extrinsic. Epipolar geometry [8] has consideration for the projective geometry that connects two camera viewpoints and has various applications such as Structure from Motion [17]. Epipolar geometry aims to describe the relationship that stands between 3D world location and 2D pixel coordinates, given a stereo pair of cameras and their corresponding poses. The fundamental matrix F is a $3 \times 3$ matrix that entirely describes such 3D/2D mapping and can be obtained through:

$$\mathbf{F} = K^{-T}[T]_X R K^{-1} \tag{2}$$

with $[.]_X$ the skew-symmetric matrix representation of any one-dimensional vector. Given a pixel location[1] $p_s \in I_s$, such fundamental matrix $\mathbf{F} \in \mathbb{R}^{3\times3}$ allows to define:

$$\mathcal{P}_{p_s} = \{p_t \in I_t | p_t^T \mathbf{F} p_s = 0\} \tag{3}$$

as the finite set of pixels from $I_t$ that live on the epipolar line defined by $l = \mathbf{F}p_s$. From a pure geometrical perspective, such line corresponds to the rendered (on $I_t$ camera plane) 3D ray that passed through both the camera center of $I_s$ and $p_s$.

The fundamental matrix $\mathbf{F}$ makes a pixel-to-line correspondence through a linear equation that involves both the source and the target original camera location. As soon as we aim to use the epipolar geometry to encode the viewpoint transformation, a sampling strategy needs to be set in order to determine which pixel location from the source image $I_s$ are going to be used to compute these epipolar lines. Instead of randomly sampling location over the $H \times W$ possibilities, and motivated by experiments that can be found in the Supplementary, pixels are sampled according to a regular grid $\mathbf{G}_r$ that spans the whole image, parameter $r$ controlling how coarse the grid is:

$$\mathbf{G}_r = \left\{ (p_x, p_y) \in \{1,..,H\} \times \{1,..,W\} \middle| \begin{array}{ll} p_x \equiv 0 & (\text{mod } H/r) \\ p_y \equiv 0 & (\text{mod } W/r) \end{array} \right\} \tag{4}$$

### 3.1.2 Encode the camera viewpoint transformation

Our module encodes the relative camera motion it exists between the source and target views by extensively leveraging epipolar geometry. Its output is fed to one of the branches of our NVS neural network as a feature image, that thus implicitly represents the transformation that occurred, and is referred as $E_{s\to t}$. An overview of the different stages involved to compute $E_{s\to t}$ is presented through the pseudo-code[2] below in Algorithm 1.

As seen in Algorithm 1, each epipolar line reported on $E_{s\to t}$ has a distinct colour, the one associated with the sampled pixel on $I_s$. Such implementation choice gives additional RGB prior information to the network regarding the colour that should be generated, even though

[1] Homogeneous coordinates are implicitly used here but omitted for clarity reason

[2] Some pixel values on $E_{s\to t}$ might be overwritten by another pixel $p$. We claim the sampling order over $\mathbf{G}$ has no impact on the encoding strategy.

lightning issues are not considered. One might notice that the last epipolar lines plotted on $E_{s\rightarrow t}$ would overwrite some previous ones, at least at some specific pixel location (on epipoles for instance). We claim, based on experimental observations, that pixel sampling order (and thus epipolar line overwriting) does not have an impact that is significant on the training and inference performances of the model.

The encoding method depicted here is thus able to encode any form of camera viewpoint transformation without any structural adaptation. Indeed, most concurrent works need to change the way viewpoint transformation is encoded since continuous poses are not processed the same way as discrete ones, where a one-hot encoding strategy might be used.

**Algorithm 1** Epipolar Encoding module

**procedure** INPUT: $(I_s, F, \mathbf{G}_r)$
  $E_{s\rightarrow t} = zeros(H,W,3)$
  **for** $p_G$ in $\mathbf{G}_r$ **do**
    $colRGB = I_s[p_G]$
    Build up $\mathcal{P}_{p_G}$
    $\forall p \in \mathcal{P}_{p_G}, \quad E_{s\rightarrow t}[p] = colRGB$
  Return $E_{s\rightarrow t}$

Only non-null pixel values are sampled from $\mathbf{G}_r$ in our encoding strategy for ShapeNet [2] since pixels located in the background do not bring any valuable information regarding the corresponding coloured epipolar lines. Figure 1 represents the kind of results one might expect with our viewpoint camera transformation encoding strategy on ShapeNet [2].

Figure 1: From left to right: Source image $I_s$, Target image $I_t$ and the corresponding $E_{s\rightarrow t}$. From the ShapeNet [2] *chair* class. Pixel locations sampled from $\mathbf{G}_{25}$ are highlighted through red circles.

### 3.1.3 Extended strategy for encoding

Performing novel view synthesis on Synthia [15] and KITTI [7] datasets is, from a relative camera transformation perspective, somehow redundant and tricky through our encoding framework. Indeed images contained in these datasets have been recorded through a car driving across city streets, and most of the camera transformations considered are translation motions. Such viewpoint changes between the source and target views are improperly handled by the epipolar theory since depth cues are lost. We therefore extended our initial encoding strategy for those datasets through a fourth channel that primarily accounts for such depth information.

Let's denote:

$$\Delta_t = |t_t| - |t_s| = [\Delta t_X, \Delta t_Y, \Delta t_Z]^T \in \mathbb{R}^3 \tag{5}$$

the difference between the two absolute translations $t_s$ and $t_t$. Absolute values are taken in Equation 5 since 2D planes coordinates are not standardised across scenes in KITTI [7] and Synthia [15]. The meaningful scalar value of interest is referred as $\delta_t$ and is computed through:

$$\delta_t = sign(t_M) \times |t_M| \tag{6}$$

with

$$t_M = \Delta_t [p_M] \quad ; \quad p_M = \arg\max |\Delta_t| \tag{7}$$

The expression of $\delta_t$ satisfies two resourceful constraints in our case:

- It accounts for the main car motion direction and gives some insight regarding the distance the car moved between the source and target view.
- The sign of $\delta_t$ indicates whether the source/target frames that were sampled correspond to a forward or backward motion.

Such properties allow the network to better apprehend the direction of the motion it should take into consideration. The value of $\delta_t$ is finally stored on a fourth channel of $E_{s\rightarrow t}$ only at locations where the epipolar lines had non-zero values in the first three RGB channels.

## 3.2 Network architecture and Training loss

The overall network architecture is presented in Figure 2. Such architecture takes inspiration through the one [9] introduced in their work, at least regarding the image-to-image U-Net based encoder/decoder structure with the hard-flow attention strategy. However, the way the transformation viewpoint information is provided to the network architecture drastically changes for our model. While [9] performed feature-vectors concatenation at the network's bottleneck stage, we claim such choice is sub-optimal, at least for discrete camera pose information as contained in ShapeNet [4]. We thus rather encode this camera pose transformation through $E_{s\rightarrow t}$ as an image that feeds a second CNN-based encoder. This last encoder produces a feature vector that will be concatenated to the one obtained by the U-Net based encoder before being consumed by the decoder.

The total loss function is a weighted sum of a Mean Average Error (MAE) , referred as $\mathcal{L}_1$ and used in [9]), and a second term, called $\mathcal{L}_{spectral}$, directly inspired form prior super-resolution work [6] and that extensively focuses on the preserving high frequencies. Indeed, considering a 2D Gaussian filter $w_{gauss}$, an image $I$ can be decomposed into a low and a high-frequency components, respectively noted as $I^{LF}$ and $I^{HF}$ :

$$\begin{cases} I^{LF} = I \circledast w_{gauss} \\ I^{HF} = I - I^{LF} = (\delta - w_{gauss}) \circledast I \end{cases} \tag{8}$$

where $\circledast$ represents a 2D convolution operation. The final loss function used during training is thus given by:

$$\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_{spectral} = |\hat{I}_t - I_t| + \lambda \left(\hat{I}_t^{HF} - I_t^{HF}\right)^2 \tag{9}$$

Additional details on $\mathcal{L}_{spectral}$ are available in the Supplementary.

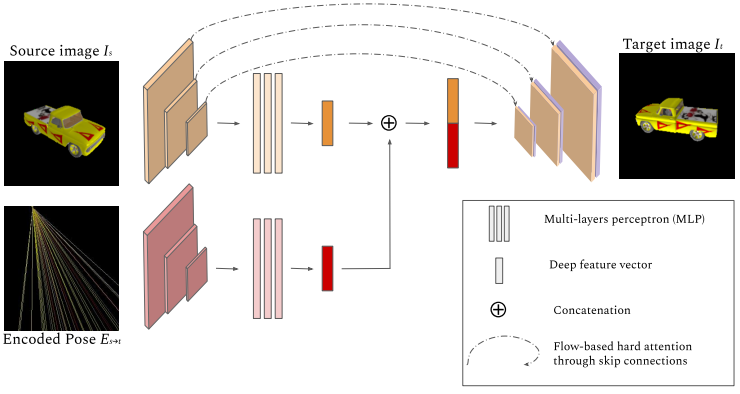


Figure 2: General overview of our architecture. Network takes as inputs both $I_s$ and $E_{s\to t}$ through two distinct encoders that produce feature vectors that are concatenated before being fed to the decoder. The network structure also leverage on the hard flow attention connections introduced in [9].

# 4 Experiments

All qualitative and quantitative results we obtained through our camera pose encoding strategy are presented in this section.

**Datasets.** We experimented our method on the same dataset as [9, 16]: on the *chair* and *car* class from ShapeNet [4] as well as on real scene images from KITTI [7] and Synthia [15]. Results we reported for [9] slightly differ from the ones published since we get consideration for a more challenging rendering scenario: jittered camera pose in ShapeNet [4], elements on image borders from Synthia [15] and KITTI [7] are not removed anymore through center-cropping , etc. Such changes are motivated and fully exposed in the Supplementary.

**Training - Testing.** We trained our model on 100,000 iterations, using the same training procedure as [9]. Since our camera pose encoding strategy is light enough, we perform "on the fly" batch creation during training. Inference scores were averaged over 100 runs with batch sizes of 16.

**Metrics.** We used the Mean Average Error (MAE), Structural Similarity Index Measure (SSIM) [20] and Peak Signal-to-Noise Ratio (PSNR) metrics to score our novel view synthesis architecture.

## 4.1 Qualitative Results

We present in Figure 3 an instance of all the datasets we get consideration for. Whereas our model manages to infer and reason about the object size according to the target view on the *Car* and *Chair* classes, novel-view produced by [9] fails to do so, producing a result that rather roughly matches the source object size. Our results on these two classes are also sharper and more realistic, both from colour and geometrical perspectives. The car's wheels

or the chair's complex back are thus better synthesise with our method.

Because [9] integrates the camera pose as discretized bins, the viewpoint transformation loses its physical inner 3D consistency structure. On the other hand, our method produces an encoding that fully accounts for the continuous pose transformation that occurred.

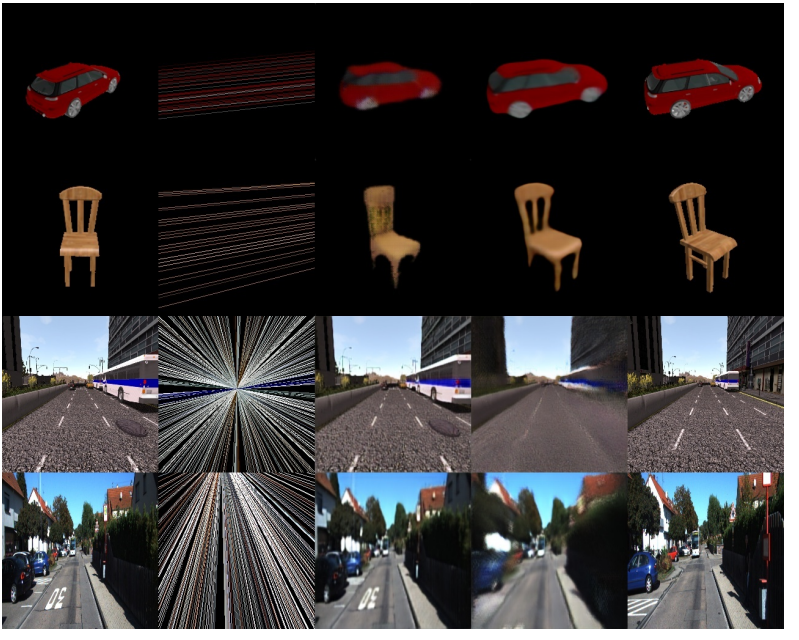

Figure 3: Inference results on all the four datasets considered. $E_{s\rightarrow t}$ encoded transformation have been obtained with $\mathbf{G}_{15}$. From left to right: the source image $I_s$, $E_{s\rightarrow t}$, [9] prediction, our prediction, the target image $I_t$. On overall, our method predicts more consistent results since complete pose transformation has been fed to the network contrary to the concurrent work from [9].

Results of our method synthesised on real-world datasets are depicted on third (Synthia [15]) and fourth (KITTI [7]) rows of the Figure 3. One might noticed how the car moved between the source and the target view in the Synthia [15] dataset. While our results remain blurry, our network architecture successfully hallucinated the forward motion that the bus had. On the other hand, our main concurrent work mostly predicts the source image, and fails to capture the relative displacement that occurred in this example. An almost similar scenario is inspected on KITTI [7]. While the sign "30" on the ground has disappeared between the two views, our method successfully captures such drastic change while [9] fails to the same extent as previously. We emphasise the fundamental role our camera encoding strategy has here, especially regarding the shape of the various objects where the perspective changes a lot between $I_s$ and $I_t$ (on the roof of the house, top left border).

While results presented on Figure 3 for real world datasets focus on the overall motion that occurred, we highlight on Figure 4 in which extent high-frequency details can be retrieved by our model. The global motion is quite properly handled by [9] too, but our method manages to better retrieved tiny details structure for the paintings on the ground.

The Spectral loss we used during training helped the network to handle these high frequencies. A complete ablation study regarding the impact the Spectral loss has can be found in the Supplementary, in addition to several other visual results on the four datasets we considered.

Figure 4: From left to right: the source image $I_s$, [9] prediction, our prediction, the target image $I_t$. Painting on the ground are better reconstructed in our method compared to [9].

## 4.2 Quantitative Results

We report some performance results over the four datasets we get consideration for. Tables 1 and 2 respectively summarise the scores for the synthetic ShapeNet [4] dataset and the real-scene ones (Synthia [15] and KITTI [7]). While our method outperforms concurrent works on MAE, we reach extremely competitive results with the SSIM metric too, since only [16] gets better scores.

However, we emphasise on a crucial aspect regarding the reported scores. Only our method and the one from [9] has been retrained with the extended and challenging datasets that we presented earlier. Results from remaining works [6, 16, 18, 24] are the ones that were originally published by authors.

| Modality | Method | Car | | | Chair | | |
|---|---|---|---|---|---|---|---|
| | | *MAE* (↓) | SSIM (↑) | PSNR (↑) | *MAE* (↓) | SSIM (↑)) | PSNR (↑) |
| *Multi-views* | [16] | 0.078 | 0.935 | - | 0.141 | 0.911 | - |
| *Single-view* | [18] | 0.139 | 0.875 | - | 0.223 | 0.882 | - |
| | [24] | 0.148 | 0.877 | - | 0.229 | 0.871 | - |
| | [6] | 0.119 | <u>0.913</u> | - | 0.202 | 0.889 | - |
| | [22] | - | 0.900 | <u>23.17</u> | - | **0.911** | **23.72** |
| | [9] | <u>0.026</u> | 0.892 | 21.18 | <u>0.045</u> | 0.865 | 17.89 |
| | Ours | **0.016** | **0.928** | **24.23** | **0.032** | <u>0.901</u> | <u>19.55</u> |

Table 1: Performance on ShapeNet [4]. Best scores for the single-view modality are highlighted in bold while second best ones are underlined.

Results on Synthia[15] and KITTI [7] are reported on Table 2 and are competitive with current state of the art methods. We emphasize that our method (as well as [9]) was trained on more complex scenarios from an image-content perspective: both Synthia and KITTI images were resized to $256 \times 256$ without any center-cropping operation (which allows to discard the most challenging elements from the scenes).

| Modality | Method | Synthia | | | KITTI | | |
|---|---|---|---|---|---|---|---|
| | | *MAE* (↓) | SSIM (↑) | PNSR (↑) | *MAE* (↓) | SSIM (↑) | PNSR (↑) |
| *Multi-views* | [16] | 0.118 | 0.737 | - | 0.163 | 0.691 | - |
| *Single-view* | [13] | 0.175 | 0.612 | - | 0.295 | 0.505 | - |
| | [24] | 0.221 | **0.636** | - | 0.418 | 0.504 | - |
| | [9] | <u>0.065</u> | <u>0.632</u> | **19.81** | <u>0.087</u> | <u>0.602</u> | <u>16.84</u> |
| | Ours | **0.065** | 0.631 | <u>19.44</u> | **0.082** | **0.609** | **17.11** |

Table 2: Performance on Synthia [15] and KITTI [7]. Best scores for the single-view modality are highlighted in bold while second best ones are underlined.

# 5 Limitations and further works

The pose encoding strategy introduced in this paper is an innovative and elegant solution to integrate the camera pose transformation in the novel view synthesis issue. However, our method suffers from some limitations that might be tackled to reach even better results, both from image quality and timely processing perspectives. Indeed, since we compute the encoded viewpoint transformation $E_{s\rightarrow t}$ "on the fly" during training, our method is slower than our main concurrent work [9]. Another further line of work concerns the camera data requirements that could be more flexible since one might argue that our method requires intrinsic parameters in addition to the extrinsic ones.

One might finally also think for future work about leveraging onto the encoded relative poses we designed to better constraints the network during training through a cyclic loss function. Indeed, it would be possible to define a reversed encoded relative pose through $E_{t\rightarrow s}$, by leveraging the predicted novel view.

# 6 Conclusion

In this paper, we proposed a new and innovative method to encode the camera transformation for the deep learning-based novel view synthesis task. For this, we leverage epipolar geometry in order to encode such viewpoint displacement as an image that we call the encoded relative pose $E_{s\rightarrow t}$ (made of several coloured epipolar lines). We argue this new camera transformation encoding is better suited for the single-image novel view synthesis issue than the standard way that only consists in considering the extrinsic values of the camera transformation. Indeed, the idea behind the vanilla approach is to feed the neural network with an RGB source image and a camera viewpoint transformation to generate a new image that can be viewed as the impact of such displacement on the input image. Our method rather proposes to provide both an RGB source image and the encoded image of the viewpoint transformation, which is already a strong insight into the impact of the displacement on this image. In other words, the core motivation of our proposed method is to help the network to better understand the correlation between the input image and the desired camera viewpoint transformation. The experimental results on the different datasets presented in this work confirm our claim. These results tend to prove that this new encoding strategy is more robust to complex displacements, namely large perspective changes and also generates images with sharper details.

# References


[1] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021.

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022.

[3] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, 2017.

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, and al. Shapenet: An information-rich 3d model repository. *arXiv*, 2015.

[5] Park Eunbyung, Yang Jimei, Yumer Ersin, and al. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017.

[6] Manuel Fritsche, Shuhang Gu, and Radu Timofte. Frequency separation for real-world super-resolution. In *ICCV*, 2019.

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[8] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003. ISBN 0521540518.

[9] Kim Juhyeon and Young Min Kim. Novel view synthesis with skip connections. In *ICIP*, 2020.

[10] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *ICCV*, 2021.

[11] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, and al. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[12] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022.

[13] Chris Rockwell, David F. Fouhey, and Justin Johnson. Pixelsynth: Generating a 3d-consistent experience from a single image. In *ICCV*, 2021.

[14] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis: Transformers and no 3d priors. In *ICCV*, 2021.

[15] German Ros, Laura Sellart, Joanna Materzynska, and & al. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[16] Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J Lim. Multi-view to novel view: Synthesizing novel views with self-learned confidence. In *ECCV*, 2018.

[17] Mohamed Tamaazousti, Vincent Gay-Bellile, Sylvie Naudet Collette, Steve Bourgeois, and Michel Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *CVPR*, 2011.

[18] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Single-view to multi-view: Reconstructing unseen views with a convolutional network. *ArXiv*, 2015.

[19] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multiview reconstruction. In *NeurIPS*, 2021.

[20] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004. doi: 10.1109/TIP.2003.819861.

[21] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *CVPR*, 2020.

[22] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021.

[23] Yunhan Zhao, Shu Kong, and Charless Fowlkes. Camera pose matters: Improving depth prediction by mitigating pose distribution bias. In *CVPR*, 2021.

[24] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.

[25] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018.