Gaining insights into the information distribution of Light Fields and enabling adaptive Light Field processing

Robin Kremer [1]

Abstract: Thanks to smartphones with several cameras, capturing a scene from multiple view points has become increasingly more available. Together with the evolving computing capabilities of modern hardware, light field processing has gained a lot of attention in the last years [Br20; Fl19; Mi20]. These techniques rely on neural networks to generate representations of the light field data. Other work assumes certain scene properties to enable light field processing (like lambertian radiation). The work shown here uses depth maps to transform the light field into a froxel (frustum + voxel)[Ev15] centered representation enabling unique post processing steps and analysis of the ray distribution in a scene. But more importantly it paves the way to quantify the information distribution within a scene. Based on this information appropriate adaptive filtering techniques can be applied. The transformation into the froxel centric representation is compatible with techniques like NERF.

Keywords: Lightfields; Froxels; Light Fields; Frustum; Voxel; Neural Radiance Field; Ray Classification

1 Introduction

Thanks to the increase in processing power in the last years, concepts like neural networks have gained a lot of popularity even though their theory has been around for much longer. A similar development is occurring on the topic of plenoptic capture and light field processing. Recent approaches even combined the two fields [Fl19] [Br20] [Mi20]. Other approaches applied more traditional processing techniques to achieve results like super resolution [LS20] or denoising [AS17]. The later techniques assume that a scene consists out of lambertian radiators, that is objects don't change color when viewed from different angles. This assumption may be false and can lead to visual artifacts.
With the proposed transformation to a froxel centered representation, the information content of regions in the scene can be identified. Furthermore, objects can be classified as lambertian or non-lambertian allowing finer control of succeeding processing steps. Traditionally the most common way to represent light field images is the two-plane parameterization $P = P(s, t, u, v)$ [DB03] [2]. Which itself is a simplification of the 7 dimensional plenoptic function $P = P(V_x, V_y, V_z, \Phi, \Theta, \lambda, t)$ [BA91]. This reduction is done by converting the angular coordinates Φ and Θ to Cartesian coordinates u and v. One spatial dependency is removed because light fields are only captured at a certain plane (by a camera gantry or

[1] Saarland Informatics Campus, Telecommunications Lab, Saarland University, 66123 Saarbrücken, Germany
[2] https://github.com/doda42/LFToolbox

plenoptic camera) and it is assumed that the rays don't significantly change while travelling from a scene object to the camera. The two remaining spatial dependencies are renamed to s and t. The time dependency is removed because most of the work is performed on still images while the wave length dependency is removed by creating a light field for each captured color channel.
A recent alternative to the two-plane parameterization are Multi-plane images (MPI) [Fl19] or Multi-sphere images (SPI) [Br20]. In these techniques the information of the light field is stored in layered meshes. The shape and color of these meshes is generated by a neural network which tries to minimize the difference between the original light field images and the reconstructed ones. While these approaches as a whole can process certain non-lambertian surfaces the data stored in each mesh layer is not view point dependent. As noted by the authors this leads to problems in displaying some visual phenomenons like curved reflective surfaces.
In Contrast NeRFs[Mi20] store the information of the light field in the weights of a trained network. Rendering a scene works by sampling this network for points along camera rays. An advantage over layered mesh approaches is that NeRFs store view point dependent colors for a scene point. Which greatly improves the visual quality of non-lambertian radiators and can handle phenomenons like curved reflective surfaces. Furthermore NeRFs don't store the color information at one specific scene point but work with a density based approach. This allows rays to change color while travelling from an object to the observer and approximating the underlying plenoptic function more closely.
As mentioned earlier the ongoing advances in computational power enabled the processing of light field images. With modern hardware even the processing of light field videos becomes feasible and thereby reintroducing the time dependency t. Taking light fields to 5 Dimensions. While the resulting amount of data increases significantly so do the available processing techniques. As an example the cameras of a light field video capture array don't have to be triggered at the same time. It is possible to offset different groups of cameras and thereby increasing the temporal resolution of the whole array. This idea is introduced as the concept of sub-framing.

2 Theory of Froxel representation

The distribution of information in a light field heavily depends on the scene geometry and the surface properties of the objects present. In the typical two-plane representation, information about both of these aspects is hard to extract. While depth maps give exceptional insights into the geometries in a scene, surface properties are still a problem. Combining both the two-plane parameterization and depth maps into a froxel centered representation gives direct access to both scene geometry and surface properties. This enables filtering techniques which are suited to the information content available and helps to identify parts of a scene where lots of redundancy is present (lambertian radiators).
The first step of the transformation is the creation of a discretization raster for the scene.

This consists out of froxels whose sizes are chosen in such a way as to perfectly match the capabilities of the capturing system. The width w_{froxel} and height h_{froxel} of a froxel are proportional to the pixel size p_{pixel}, the distance of the froxel from the camera D_{plane} and inversely proportional to the focus distance f_d of the capturing system (for square pixels $w_{froxel} = h_{froxel}$).

$$w_{froxel} = h_{froxel} = \frac{p_{pixel} D_{plane}}{f_d} \tag{1}$$

The depth of a froxel is based on disparity achieved by the capture system.

$$d_{froxel} = D_{plane}^2 / \left(\frac{f_d \cdot \max(a_{max} \cdot a_{spacing}, b_{max} \cdot b_{spacing})}{p_{pixel}} - D_{plane} \right) \tag{2}$$

Equation 2 is used to calculate the depth of a froxel at a distance D_{plane} from the capture system. $\max(a_{max} \cdot a_{spacing}, b_{max} \cdot b_{spacing})$ is the largest distance between two capture points of the system. Figure 1 illustrates that the largest baseline distance is the defining baseline for the froxel depth. This approach currently only works on planar capture systems. The resolution of the resulting discretization raster will perfectly match the capabilities of the capture array, such that no two rays captured by one camera will be assigned to the same froxel. The allocating of rays to froxels is the transformation from the two-plane parameterization to the froxel centered representation. In order to assign the rays to the froxels the s, t, u, v parameters and a depth map of the scene are used. With this the origin of a ray in the scene can be calculated. The previously generated discretization raster is then applied to these positions to assign all rays to their corresponding froxel. The result is a new representation in the way of a froxel centric data structure where a ray is not identified by a camera and pixel position (like in the two-plane parameterization) but identified by its origin in the scene. Note that for reconstruction purposes the camera number which captured the ray is stored alongside the color information. A scene point sampled by different cameras will be assigned to the same froxel after the transformation. This allows direct insight into which parts of a scene are only sampled by a few cameras and therefore unsuited for techniques like super resolution or denoising, while scene points for which lots of samples exist can be further analysed. For example the color distribution of all rays from a single froxel gives insight into the lambertian properties of the corresponding scene point. Transforming the light field from the froxel representation back to the two-plane parameterization or generating view points of the light field is done by selecting all froxels in a given view frustum. The rays assigned to these froxels are then projected onto a virtual camera sensor placed at the wanted view point. Note that this view point doesn't have to be at a position originally sampled by the capture system but can also be a novel one.

3 Implementation

A first implementation was done in Matlab® where concepts like the transformation to the froxel representation and further analysis were explored. Due to the chosen underlying data

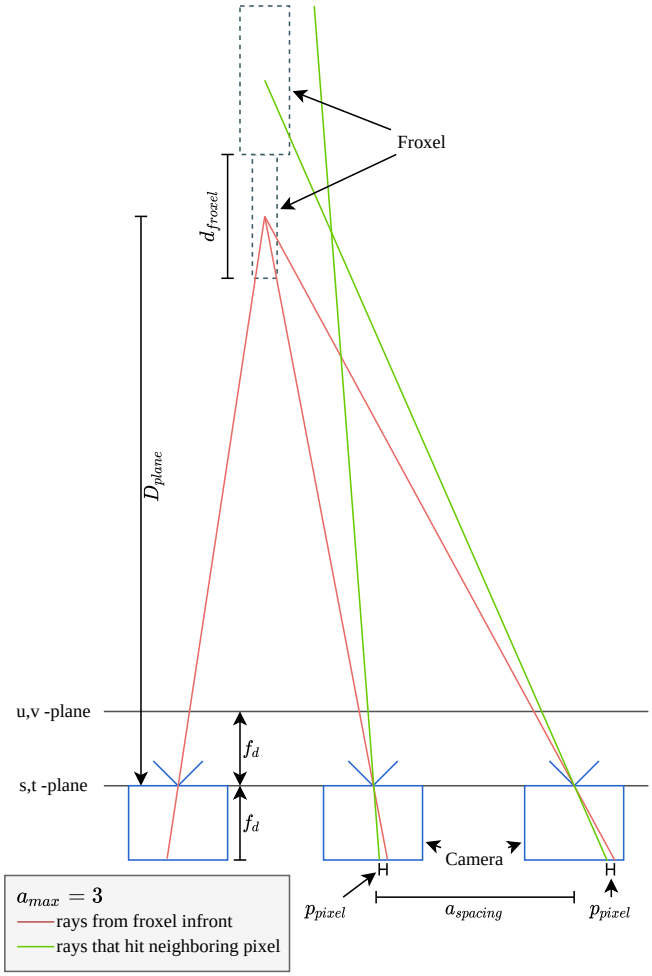
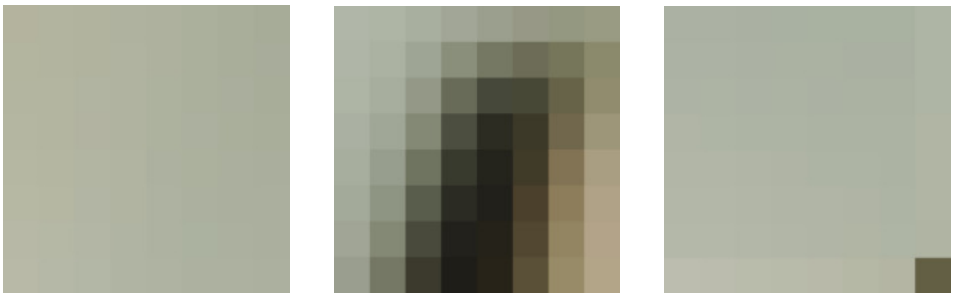


Fig. 1: Top view onto a multi-camera array with two froxels shown

structure and limitations of Matlab® this implementation turned to be prohibitively slow for further development.

After a small experiment to compare different programming languages, python together with the just-in-time compiler numba was used for a reworked implementation. Thanks to the performance improvements this implementation is also capable of working with 5D light fields (light field videos) were the dependency on time t is reintroduced. As the underlying data structure a dictionary/map is used, for which the keys are generated based on the spatial and temporal position of a froxel. This allows for efficient storage of the sparsely sampled scene space. After the transformation, the found froxels can be categorised based on the

color distribution of the contained ray-set. This is currently done with a standard deviation and z-score threshold where the first one classifies non-lambertian froxels and the second one finds outliers in non-lambertian froxels. In the 5D implementation this analysis can also be done over time. After this classification different post processing techniques can be applied. As an example a ray reduction filter is implemented which reduces the number of rays saved per froxel by applying a median filter to all rays classified as lambertian. While rays classified as non-lambertian or outliers are not reduced. The rendering of light field images can than be performed on this reduced set of total rays without a significant loss in visual quality.



(a) lambertian (b) non-lambertian (c) outlier

Fig. 2: Example data stored in a froxel from a 4D light field captured on a 8×8 camera array.

The quality of the depth maps is important so that one scene point sampled by different cameras is assigned to the same froxel. Due to this the development has been done with scenes created in blender where near perfect depth maps are available.

4 Results

The implementation (compare figure 3) was developed and tested on blender demo scenes [3]. A blender add-on [Ho16] was used to render the color data and depth maps for a 9×9 camera array with a baseline between neighbouring cameras of 70 mm.

After converting the light field from the two-plane parameterization into the froxel representation a fristogram (froxel + histogram) [He21] can be generated. Fristograms give first insights into how data is distributed in a light field. Figure 5 shows a CDF fristogram of the classroom scene. One observation is that over half of the froxels are seen by all 16 cameras of the 4×4 camera array. Furthermore distinct steps each four additional cameras are visible. This occurs because on a regular camera array vertical or horizontal occlusions often effect a whole column or row.

[3] `https://www.blender.org/download/demo-files/`

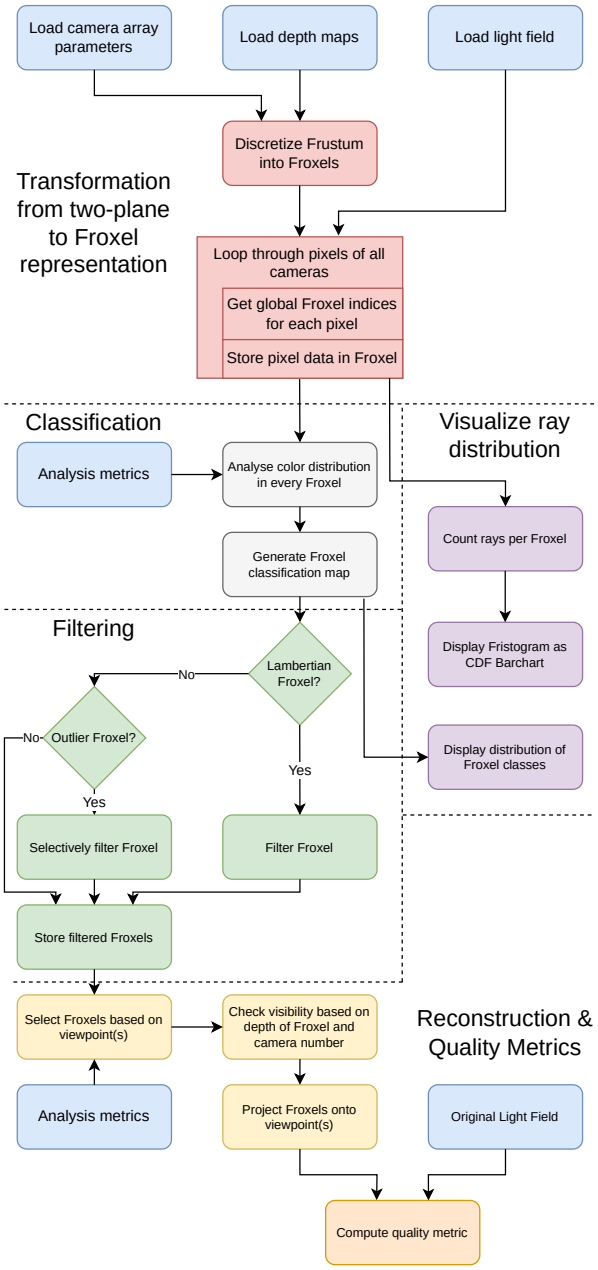


Fig. 3: Flowchart of the froxel transformation

Fig. 4: Classroom scene (left) and BMW scene (right) from blender which were used to develop the pipeline

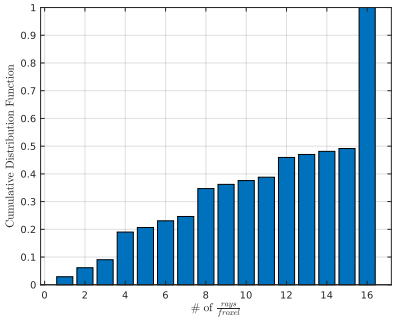


Fig. 5: Fristogram of the classroom scene captured by a 4×4 camera array. Empty froxels are omitted

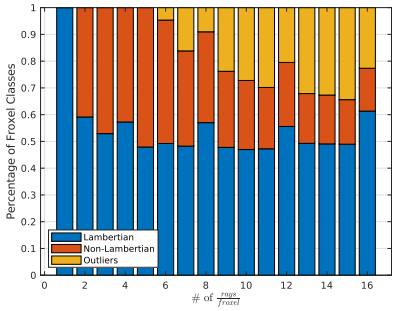


Fig. 6: Distribution of the ray classes over number of rays per froxel (classroom on 4×4 camera array)

In theory without further processing steps like ray reduction a transformation back to the two-plane parameterization should be lossless. As shown in figure 7b the SSIM of the reconstruction shows a slight degradation. This is most likely due to the rounding that happens during discretization and view reconstruction.

Since over half of the froxel are seen by all cameras, there is redundancy present in the light field data. By applying a median or mean filter to all rays present in the froxels this redundancy can be reduced. The total number of rays for the classroom scene captured on a 4×4 camera array is reduced from $4 \times 4 \times 1920 \times 1200 = 36{,}864{,}000$ to 3,185,991 (8.64%). Figure 7c shows the result of a mean filter. This technique assumes that one ray is sufficient to describe the color of a froxel, this is true for lambertian scene points (e.g. the wall). In contrast, non-lambertian froxels need more rays to be displayed accurately which is why this reduction leads to visible artifacts.
Because froxels are directly linked to scene points the ray color distribution stored inside each froxel gives insight into the surface properties of that scene point. A low standard deviation among all ray colors of a froxel is an indication that the underlying scene point acts

(a) Original, SSIM: 1.0

(b) No ray reduction, SSIM: 0.9987

(c) Reduction to one ray per Froxel, SSIM: 0.9641

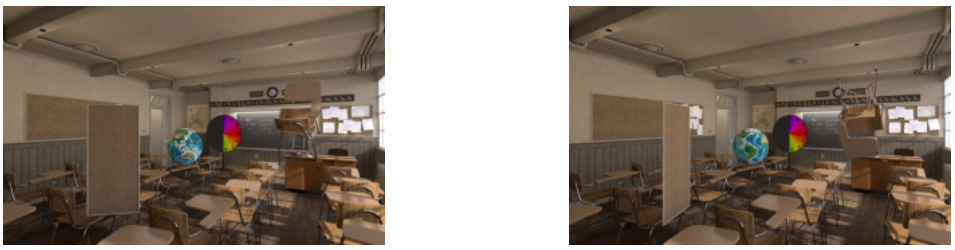
(d) Ray class aware reduction, SSIM: 0.9838

Fig. 7: Image of Classroom scene. The reduction to one ray per Froxel (Figure 7c shows artifacts on non lambertian surfaces. Detail is lost in the reflections of the table and the metal bars.

like a lambertian surface. While a large deviation most likely comes from non-lambertian properties of the surface like reflections or transparency. Applying a threshold to this deviation therefore makes it possible to classify scene points. During the development an outlier class was introduced next to lambertian and non-lambertian for froxels where only very few rays deviate from the mean (compare figure 2c). This classification is based on the z-score of all rays. With this information a surface property aware ray reduction is possible. By choosing a standard deviation threshold of 2.0 and a z-score threshold of 4.0 the total number of rays left after the reduction increases to 10,483,959 for the classroom scene. Which still is only 28.4% of the original ray count. This increase yields an SSIM improvement from 0.96419 to 0.98384 (figure 7d). The distribution of the ray classes over the number of rays per froxel can be seen in figure 6.

For 5D light fields the froxel analysis can also be extended over the time domain. This enables finer classification of scene points. As a demonstration a few moving objects were added to the classroom scene and a short animation was rendered (compare figure 8). This leads to a new class of objects that change their appearance not only when viewed from a different angle but also change it over time (compare figure 9). This information can be utilized by further post processing steps like a 5D ray reduction. The froxel representation also provides direct insight into how sub-framing a camera

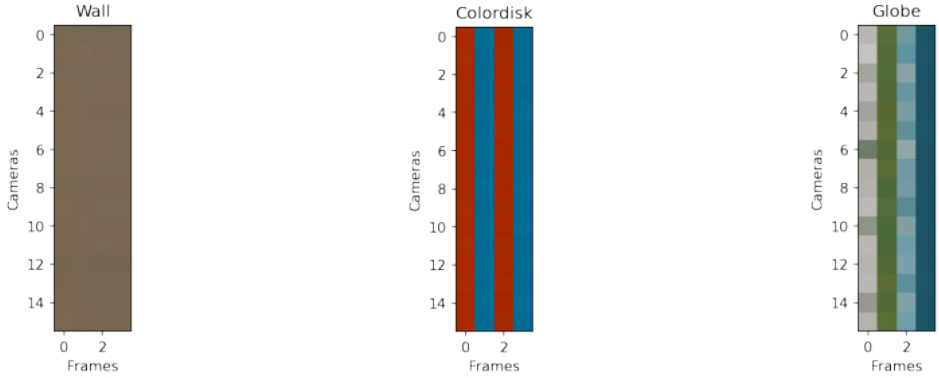
array trades spatial resolution for temporal resolution. As a demonstration the same animation as before was rendered but with a sub-frame pattern applied to the camera array. The chosen sub-frame pattern consists out of four groups (each array column is one group). The trigger time points were then uniformly distributed in one frame interval. This practically increases the temporal resolution by a factor of four while each camera still captures at the original frame rate. The resulting froxel data is shown in figure 10.



(a) First picture of the animation

(b) Last picture of the animation

Fig. 8: As a demonstration for the 5D capabilities a few moving objects were added to the classroom scene. Note that color wheel looks static because it's rotational speed is synchronized to the frame rate.



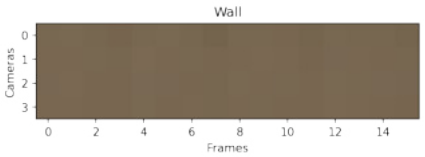
(a) Wall - lambertian and not changing over time

(b) Color disk - lambertian and changing over time

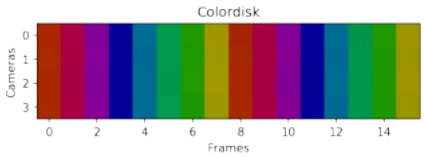
(c) Globe - non-lambertian and changing over time

Fig. 9: All 16 cameras of the 4×4 camera array are triggered at the same time and capture the scene with a constant frame rate (only 4 frames shown). Along the y-axis the ray color captured by each camera is display while the x-axis shows the change over time.

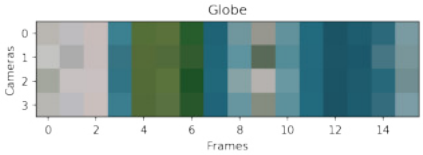
In conclusion the proposed transformation to a froxel centric light field representation allows for deeper insights into how data is distribution in a light field. It not only allows to identify regions of a scene which are suitable for certain post processing techniques like super resolution but also is able to classify the surface properties of these scene points. As a demonstration of the capabilities the total number of rays of a light field were reduced



(a) Wall - lambertian and not changing over time



(b) Color disk - lambertian and changing over time



(c) Globe - non-lambertian and changing over time

Fig. 10: The camera array columns are staggered against each other increasing the temporal resolution by a factor of four but decreasing the spatial resolution. Note that the overall captured time is the same for both cases.

significantly while retaining a very high visual quality. Furthermore the proposed technique can also be applied to light field videos which increases the possible post processing options even more.

5 Future Work

A big advantage of the proposed froxel based representation is the compatibility with other formats. As mentioned the implementation is currently only tested on virtual scenes because of the high quality depth maps. For real scenes it is proposed to use depth maps created by a NeRF network. While with the introduction of NeRFs to the overall pipeline the froxel based representation may seem unnecessary it should be noted that rendering views from NeRFs is very computationally expensive. This is because a NeRF MLP has to be queried more than 100 times for a single pixel resulting in more than 100 million querries for high quality images ”on an NVIDIA V100, this takes approximately 30 seconds per frame” [Mi20]. In contrast rendering a view point from a froxel based representation in a comparable quality takes 200 ms on a single core of an Intel i7-7700 CPU @ 3.60GHz.
Another feature of NeRFs is the use of a volume rendering approach. While the current view rendering of the froxel based representation works by only rendering the froxels closest to the camera. NeRFs therefore stay more true to the underlying plenoptic function where rays can change while travelling from an object to the observer. A volume density aware renderer is also compatible with the idea of the froxel based representation. Making it possible to further combine both ideas in order to achieve good visual quality and low computational complexity.
The current implementation together with the used data sets will be made available as open source.

References

[AS17] Alain, M.; Smolic, A.: Light field denoising by sparse 5D transform domain collaborative filtering. In: 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP). IEEE, pp. 1–6, 2017.

[BA91] Bergen, J. R.; Adelson, E. H.: The plenoptic function and the elements of early vision. Computational models of visual processing 1/, p. 8, 1991.

[Br20] Broxton, M.; Flynn, J.; Overbeck, R.; Erickson, D.; Hedman, P.; Duvall, M.; Dourgarian, J.; Busch, J.; Whalen, M.; Debevec, P.: Immersive light field video with a layered mesh representation. ACM Transactions on Graphics (TOG) 39/4, pp. 86–1, 2020.

[DB03] Dansereau, D.; Bruton, L.: A 4D frequency-planar IIR filter and its application to light field processing. In: Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS'03. Vol. 4, IEEE, pp. IV–IV, 2003.

[Ev15] Evans, A.: Introduction of the term Froxel, `https://www.realtimerendering.com/blog/tag/siggraph-2015/`, Accessed: 2022-05-02, 2015.

[Fl19] Flynn, J.; Broxton, M.; Debevec, P.; DuVall, M.; Fyffe, G.; Overbeck, R.; Snavely, N.; Tucker, R.: Deepview: View synthesis with learned gradient descent. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Pp. 2367–2376, 2019.

[He21] Herfet, T.; Chelli, K.; Lange, T.; Kremer, R.: Fristograms: Revealing and Exploiting Light Field Internals. arXiv preprint arXiv:2107.10563/, 2021.

[Ho16] Honauer, K.; Johannsen, O.; Kondermann, D.; Goldluecke, B.: A dataset and evaluation methodology for depth estimation on 4D light fields. In: Asian Conference on Computer Vision. Springer, 2016.

[LS20] Le Pendu, M.; Smolic, A.: High resolution light field recovery with fourier disparity layer completion, demosaicing, and super-resolution. In: 2020 IEEE International Conference on Computational Photography (ICCP). IEEE, pp. 1–12, 2020.

[Mi20] Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision. Springer, pp. 405–421, 2020.