On the Possibility of NLOS Reconstruction Using Spatial Decomposition

ECE901 Project

Eric Brandt
University of Wisconsin, Madison
Email: elbrandt@wisc.edu

***Abstract*—Hidden scene reconstruction by non-line of sight imaging is a popular and growing field of computational optics. Capabilities for fast capture of large hidden scenes are improving, placing increasing pressure on efficient computational methods to reconstruct those scenes. Using current techniques, this reconstruction process is quite computationally expensive if performed in the time domain, having a $O(N^5)$ complexity in the straightforward implementation. This is a significant impediment to considering reconstruction of hidden scenes larger than a small room. We investigate the opportunities for reducing this computational cost by combining techniques of variable resolution reconstruction and spatial decomposition. Using existing data sets, we show that by using efficient data structures, an empirical computational lower bound on time domain reconstruction can be as low as 0.5% of the cost of the straightforward implementation.**

I. Background

Hidden scene reconstruction by use of non-line of sight imaging techniques is a popular sub-genre of computational optics research. A typical experimental setup involves a laser capable of emitting short pulses and a camera capable of measuring arrival times of single photons with very high accuracy. A very short pulse of laser light is directed to a relay surface, typically a wall. This coherent light is scattered in all directions when reflected off the relay surface into a scene that is not directly visible to the camera. Some of the scattered laser light bounces off of objects in the scene, some of which bounces back to the relay wall, and finally some of which is detected by the camera, as depicted in figure 1. The difference between the emitted number of photons and the detected number of photons can be on the order of 10^{-15}. Sequentially, the laser is aimed at many positions on the relay surface, usually in a grid-like pattern, at each of which a pulse is emitted and the camera detects and time-stamps the photons that have completed the three-bounce voyage from wall, to hidden scene, to relay wall, to camera. From this, a '3D-Impulse Response' cube of data can be collected. The first two dimensions of this response cube are the X- and Y- location of the originating pulse on the laser wall. The third dimension is the arrival time of each photon captured by the camera from the pulse at the particular X,Y location.

Once the data is collected, any of several computational methods [1]–[3] can be used to reconstruct the hidden scene from the 3D-Impulse Response cube. Perhaps the most straightforward method is to perform backprojection in the time domain, described now: Consider a single relay wall location, and the 1D time-series of photon arrival times collected by the camera from the laser pulse directed at that relay wall location. We can mathematically construct a 'virtual phasor pulse' as shown in figure 2 and convolve this pulse with the 1D time series. This has the effect of simulating the sending of this virtual pulse through the scene. This convolution can be applied to the time-series collected for all X- and Y-locations on the relay wall, forming a data set representing how a laser pulse would travel through the scene. This is commonly known as the Virtual Phasor Field in literature [2].

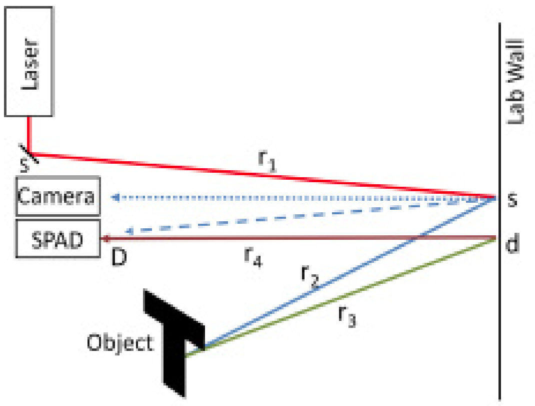

Fig. 1. Diagram of photon path in typical NLOS experimental setup

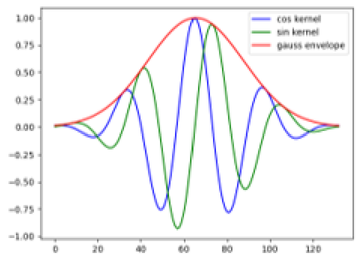

Fig. 2. Phasor pulse to be convolved with time impulse response

Having this Virtual Phasor Field, we can reconstruct the

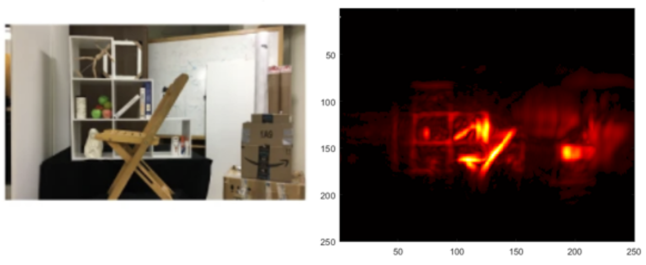

Fig. 3. Left: Actual scene. Right: NLOS Reconstruction

hidden scene as follows: The hidden scene is spatially divided into voxels of a uniform dimension that is a function of the wavelength used in the virtual phasor pulse (see Proposed Method section for details). The reconstruction algorithm considers each voxel, which has a known 3D coordinate in the scene, and calculates the flux of photons that were captured by the camera as a result of reflection off of surfaces in that voxel. To do this, for a given voxel, the algorithm considers each relay wall location of laser pulses, and computes the distances r_1, r_2, r_3 , r_4 in figure 1, where the distances r_2 and r_3 are determined by the location of the voxel and the laser position currently under consideration. The time-of-flight is calculated from the sum of these distances. The virtual Phasor Field is used as a sort of 'lookup table' to find out if any, and how many, photons were recorded that would have an arrival time coinciding with being reflected by this voxel. Summing this lookup over all laser positions for a given voxel gives the intensity of reflection for that voxel. Repeating this for all voxels produces a 3D spatial cube of signal response which represents the reconstructed scene. For a 2D view of the reconstructed scene, the maximum intensity for each X,Y coordinate is selected along the corresponding Z depth. The resulting 2D image is then normalized and color-mapped for presentation. A typical reconstruction can be seen in figure 3 (source: [2]).

As can be seen in the algorithmic description above, the computation time required to perform the reconstruction is a product of the relay wall dimensions (e.g. $U \times V$) and the voxel scene dimensions (e.g. $X \times Y \times Z$), resulting in a computational complexity of $O(UVXYZ)$ or more compactly $O(N^5)$.

II. Related Work

Early work on non-line of sight imaging was pioneered by [1] who began the work of guiding an ultra-fast pulsed laser to multiple relay wall locations using a galvo. In their work, a streak camera was used as the detector. This work introduced the intersecting ellipsoid method of scene reconstruction. As Single Photon Avalanche Diodes (SPADs) became available, the streak camera in the early setup was replaced for these simpler devices. In a different experimental setup, [4] used a confocal laser and detector setup to record hidden scenes with good accuracy and relatively simple reconstruction, but very long capture times. [5] showed that NLOS reconstruction was with much simpler hardware (e.g. commercial cell phone camera) over long distances, using novel methods and techniques. [2] used the non-confocal setup from [1], but with SPAD detectors as the camera, and introduced a so-called 'Fast RSD' method that computed the scene reconstruction by solving the Rayleigh-Sommerfeld diffraction integral in the frequency domain. Recently, [3] extended that work to use multiple SPAD detectors (e.g. multiple cameras) to capture and reconstruct a scene at 5 frames per second, proving that 'real-time' or 'live' NLOS reconstruction is a possibility, under the assumption of reasonable constraints on scene volume and relay wall size. The body of knowledge built by these successive advances in NLOS scene reconstructions provide a starting point and good source of existing data sets for this investigation in primal (time) domain scene reconstruction by multi-resolution space decomposition.

III. Proposed Method

When considering the computational complexity of the time domain reconstruction, the immediate observation of the algorithm described in the Background section is that while we dutifully compute the signal for every voxel in the scene, most of those voxels are 'empty' in that they have a negligible total signal response. In fact, when considering light propagation as ray geometry, one can easily argue that for each X,Y coordinate in the scene, only one voxel along the Z depth should have a non-zero value. That first non-zero voxel would be the location of the first surface encountered by the traveling photon that produced the reflection. (While this is true in theory, in practice it is not that ideal due to scene noise, higher order bounces, non-cartesian projection anomalies, and other experimental artifacts.) This exact principle commonly is used existing literature when projecting the 3D reconstruction to a 2D imagine using the 'max' filter along the Z dimension of each 2D pixel. For this reason, although imperfect, the 'one-hot-voxel' representation remains a useful thought experiment that we will use to pursue our investigation. For the investigation and results that follow, we will presume that we know *a priori* that we have a non-zero response from a single depth voxel for each X,Y coordinate of the scene.

The high level concept of our method is to find the large, empty areas of the reconstruction volume quickly. Once those large regions have been identified, we can mark that sub-region as empty and skip the effort of reconstructing the finest resolution voxels in that region, saving computation costs that grow as $O(N^3)$ with respect to the dimensions of the empty region.

To achieve computational gains, we need to develop a toolset admitting two capabilities: First, we must be able to reconstruct the scene using an arbitrary voxel dimension. Second, we need a convenient and efficient data structure to decompose 3D space into nested sub-regions. Each of these will now be considered.

A. *Variable Resolution Reconstruction*

As noted above, the resolution at which a reconstruction is performed is a function of the wavelength of the virtual phasor field that is convolved with the 3D impulse response and the spacing of the sampling grid on the relay wall. Prior work and reference implementations [2] have shown that the minimum voxel dimension for reconstruction is half of sampling grid spacing, due to the spatial Nyquist limit. The virtual wavelength for this finest resolution reconstruction is then chosen to be double the sampling grid spacing. Given a sampling grid spacing on the relay wall and we arrive at a simple relationship for the virtual phasor pulse wavelength (λ) and the voxel dimension:

$$\lambda = 2 \times n \times (\text{sampling grid spacing})$$

$$\text{voxel dimension} = \frac{\lambda}{4}$$

where n is a multiplier that we can apply to vary the voxel dimension at which we wish to reconstruct our scene. Choosing $n = 1$ will reconstruct the scene at the finest resolution that will not exhibit sampling artifacts, and choosing $n > 1$ will provide a wavelength that results in a scene reconstructed using fewer, larger voxels.

Once the desired voxel size is chosen and the necessary virtual phasor pulse wavelength is calculated, the 3D impulse response is convolved with that particular virtual phasor pulse as described in the Background section, producing a virtual phasor field for the desired reconstruction resolution. As an example, virtual phasor pulses for two different resolutions of reconstruction are shown in figure 4.

Additionally, when reconstructing at a lower resolution, the above equations guiding the virtual wavelength selection can be applied equally to the grid spacing of the relay surface. With a coarser reconstruction resolution, we could also have sampled the relay wall at a lower resolution. This can be accomplished by subsampling the relay wall impulse response before convolution, either by averaging or max-pooling. This will reduce the number of contributions that have to be summed for each coarse voxel reconstruction.

The required virtual phasor fields may be computed in the time domain using many straightforward 1-D convolutions (as was performed in the accompanying code of this paper for simplicity), or in the frequency domain using pre-computed RSD kernels of the appropriate wavelength, as detailed in [2] for higher performance. One virtual phasor field for each desired reconstruction resolution must be pre-calculated. While the resulting virtual phasor fields are relatively large in memory, they are also quite sparse, and considerable memory savings are possible using sparse representations (e.g. CSR, CSC).

B. *Spatial Decomposition*

The second tool that must be developed for our proposed method is a computation and memory efficient method to decompose 3D space. For this, we draw upon the well known data structure, the *octree*, first proposed by [6]. This is a data structure that represents an axis-aligned 3D volume as a tree. The root node of the tree represents the entire volume. Each node of the tree starting with the root node has eight children. Each child represents one octant of the space contained by its parent. Therefore, each of the eight children represent $\frac{1}{8}$ the volume of its parent. As the tree descends in depth, the volume of a voxel at the nth level of the tree (with the root being $n = 0$) represents a volume of $(\frac{1}{8})^n$ of the entire volume. A diagram of the tree structure, taken from [7] is shown in figure 5.

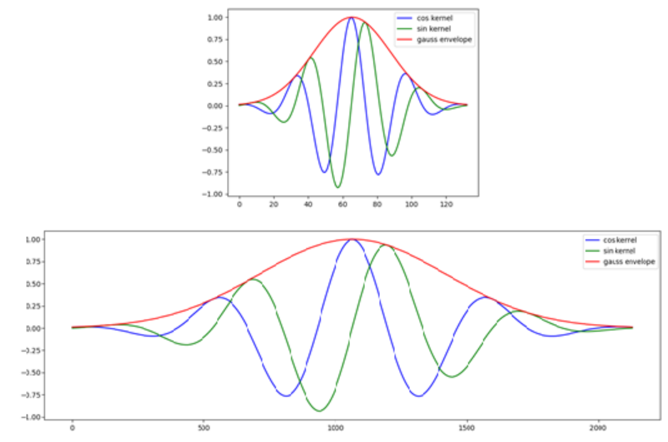

Fig. 4. Top: Short phasor pulse, resulting in a high resolution reconstruction. Bottom: Long phasor pulse, resulting in a low resolution reconstruction.

Implementation of our proposed method proceeds in the following manner. First, reconstruct the scene at a very coarse resolution. (In the limiting case, this would be reconstructing the scene as a single voxel). If there is sufficient signal present in this voxel, expand the octree by one level, dividing the current voxel into eight octants. For each of the eight octant voxels, reconstruct that voxel resolution and check if a sufficient signal is present. If a sufficient signal is not present, declare that voxel 'empty' and omit it from further processing. If the voxel has sufficient signal, repeat the process of subdividing and increasing the tree depth *for that subtree* by one. Proceed until either all remaining voxels have been marked 'empty,' or the tree depth produces leaf nodes whose voxels have dimension equal to the smallest possible reconstruction resolution of the source dataset. With this method, we identify large 'empty' areas early and do not spend memory computational resources on reconstructing those areas at detailed resolution.

IV. Results

The experimental portion of this work consisted of working with datasets collected by [2]. Python code was developed to perform the virtual phasor field and backprojection as described in the Variable Resolution Reconstruction section above. An example of the construction at various resolutions can be seen in figure 6. This figure contains the 2D projection of the 3D reconstructions using the method described in the Background section. The convolution and backprojection

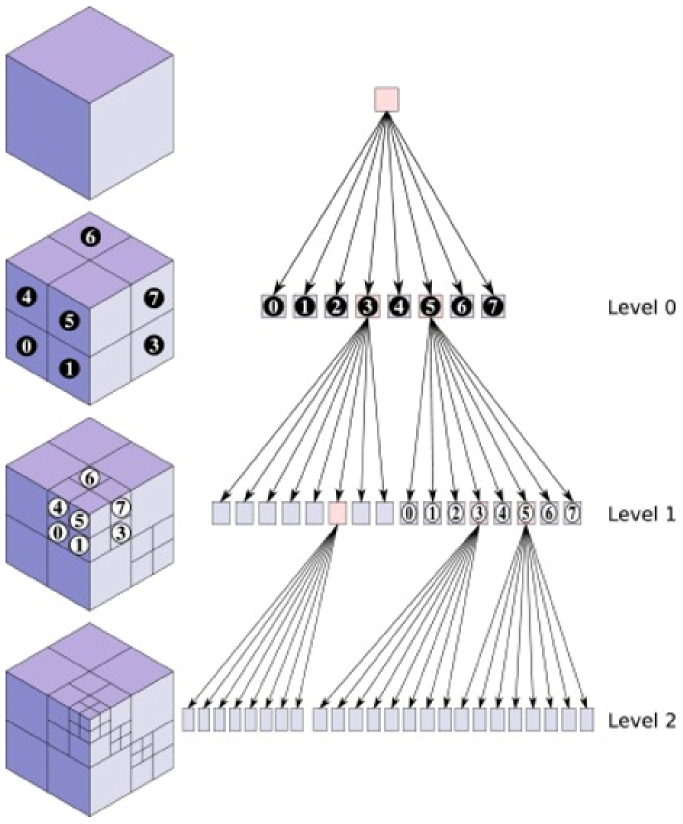

Fig. 5. Spatial decomposition by an octree data structure

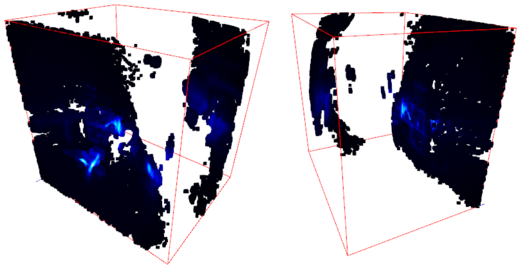

Fig. 7. Left: Ideal reconstruction of office scene at 1cm resolution, front view. Right: Same scene, rear view.

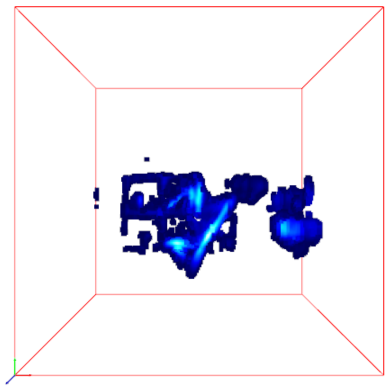

Fig. 8. Left: Ideal reconstruction of office scene at 1cm resolution, excluding low-signal return pixels

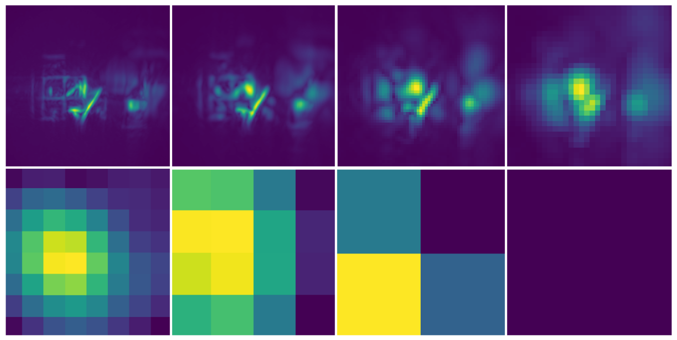

Fig. 6. Left: 2D Reconstruction of office scene at different resolutions. The top row from left to right is 1cm, 2cm, 4cm, and 8cm. The bottom row is 32cm, 64cm, 128cm, and 256cm. (16cm is not shown). The actual scene is 2.5m in each dimension, meaning the lowest resolution reconstruction (bottom right) is a single pixel (voxel).

algorithms were implemented using the `scipy` and `numpy` libraries, using vector operations everywhere possible, but running in a single thread. This implementation achieves performance results that exceed the reference Matlab/C++ implementation in [2] when the C++ portion is run in single threaded mode. Significant opportunities for further optimization occur in performing the RSD convolution in frequency space, in performing the backprojection in parallel, and in general using GPU resources.

For the multi-resolution spatial decomposition portion of the investigation, we first assumed an ideal reconstruction at the finest possible resolution, where a single voxel in each Z-column is marked as having 'sufficient signal' and the remainder of voxels in that column are marked empty. This represents the ideal case, and a lower bound on the highest sparsity we could hope for in a reconstruction without noise or other artifacts. In our specific office scene dataset, this starting point was a 3D volume of size 2.5m x 2.5m x 2.3m, reconstructed at 1cm resolution, yielding a total of 14,375,000 voxels, of which 62,500 contain signal. A 3D view of the reconstructed scene prepared in this manner is shown in figure 7.

As can be seen in this figure, there are significant areas of the scene where even the highest intensity Z value for a given X,Y does not represent a meaningful object in the scene, for example the black pixels around the perimeter. These low-signal return voxels present an opportunity to exclude voxels from the scene that don't contribute to the semantic content of the image by means of a low-pass filter. Choosing a value empirically for the candidate scene results in a much more useful scene representation, shown in figure 8.

This ideal scene reconstruction was loaded into an octree data structure with visualization capabilities as provided by the Python `open3d` [8] library. With this representation, the scene was regenerated using octrees with various maximal depths. Several of these depths are shown in figure 9.

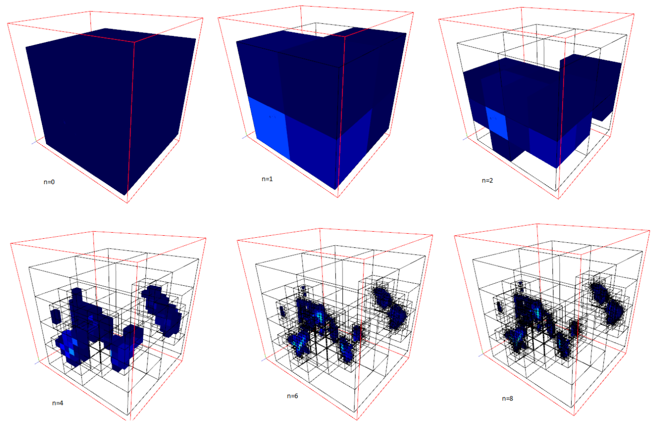

Fig. 9. Left: Octree representation of the scene with various maximum depths

	With Threshold	Without Threshold
Maximum Depth	8	8
# of Leaf Nodes	8,969	62,500
# of Internal Nodes	6,953	52,865
# of Empty Nodes	39,703	307,556
# of Reconstructions	55,625	422,921
% Reconstructed	0.45%	3.38%

TABLE I
OCTREE STATISTICS FOR OFFICE SCENE RECONSTRUCTION

Our example scene has a dimension of 2.5m and a minimum reconstruction resolution of 1cm, meaning that when the octree reaches a depth of 8 (e.g. 2^8 reductions in resolution), all original voxels that we want to represent will be located uniquely in leaf nodes of the tree. Statistics are calculated on the representation at the full spatial decomposition (depth = 8), and shown in table I for both for with-pre-threshold and without-pre-threshold octree representations. As can be seen, with a reasonable pre-filter threshold value, it is possible that a complete reconstruction of the office scene using spatial decomposition would require only 55,625 voxel reconstructions instead of the full 14,375,000 reconstructions required if all 1cm voxels were reconstructed. This represents a substantial computational cost savings during the backprojection phase of the reconstruction.

V. Limitations & Future Work

The work performed here and the proposed method is a proof-of-concept to demonstrate the potential gains of a complete implementation of the non-line of sight backrpojection reconstruction by means of spatial decomposition. There are multiple opportunities for improvement of method, and several unanswered practical questions before a complete implementation is feasible.

The largest open question is determining the appropriate threshold at which the decision should be made to ‘declare empty’ or ‘descend further.’ In our simulation, this question was easily answered because the desired voxels to represent in the final tree were pre-computed during a full-resolution reconstruction. In practice, at each level of decomposition, the algorithm will need to determine whether the intensity value for that voxel represents a sufficiently large value that the interior should be subdivided and reconstructed. Finding this value may be a combination of theoretical and empirical work. The theoretical work may involve calculating the full expected intensity of the entire scene, and devising a bookkeeping method to determine how much has been accounted for already at other levels of the octree under construction. The threshold for descend/not-descend decision may even depend on z-depth of a voxel under consideration, as the signal fall-off with depth has recently been investigated by [3]. This theoretical work could best be studied on simulated data, free of noise and higher order bounce artifacts. On the empirical side, it may be necessary to perform numerous experiments on existing data sets to understand and characterize the noise contributions to the scene.

Theoretical questions exist about the most efficient way to represent the virtual phasor field with multiple phasor field wavelengths. In the current work, each are stored separately in dense format. Choosing a sparse representation that is compact and easy to access will be important to reducing memory footprint of the method.

Implementation details also must be addressed. Modifications and additions will need to be made to the `open3d` library that supports the octree format. The current implementation chooses its root level dimensions based on the extents of the interior data supplied to it, rather than on an arbitrary work volume. Efficient use of GPU or other forms of parallel computing will still be necessary to fully utilize the processor capabilities of the host system. The current exploratory implementation is single threaded and CPU-only. Recursive descent through a tree structure should lend itself well to course-grained parallelism of multi-core CPUs, with individual voxel reconstruction being ideally suited for fine-grained parallelism of a GPU.

Currently, the scene reconstruction is done on a cartesian grid of voxels. It would be more appropriate to do the reconstruction in spherical coordinates with frustrum-shaped voxels that increase in size with increasing z-depth. The octree data structure would handle this sufficiently well, but the current implementation being used would require modifications.

An additional future possibility of the advantages of octree representation of a NLOS scene is to reconstruct different areas of the scene at different resolutions. This turns the ‘descend vs. not-descend’ decision from a binary decision of ‘empty vs signal-present’ to a multi-valued (or continuous) function of how important a particular area of a scene is deemed to be. Octree representation may be of significant utility if video (e.g. multi-frame) reconstruction with motion flow estimation is considered.

Ultimately, with the satisfactory resolution of the theoretical and practical questions raised above, the goal of this work should be to efficiently reconstruct scenes orders of magnitude larger than what is possible today. The encouraging results of this initial exploration suggest that this should indeed be possible.

REFERENCES

[1] A. Velten, T. Willwacher, O. Gupta, A. Veeraraghavan, M. Bawendi, and R. Raskar, “Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging,” *Nature communications*, vol. 3, p. 745, 03 2012.

[2] X. Liu, I. Guillén, M. Manna, J. Nam, A. Reza, T. Le, A. Jarabo, D. Gutiérrez, and A. Velten, “Non-line-of-sight imaging using phasor-field virtual wave optics,” *Nature*, vol. 572, 08 2019.

[3] J. Nam, E. Brandt, S. Bauer, X. Liu, E. Sifakis, and A. Velten, “Real-time non-line-of-sight imaging of dynamic scenes,” 10 2020.

[4] M. O’Toole, D. B. Lindell, and G. Wetzstein, “Confocal non-line-of-sight imaging,” in *ACM SIGGRAPH 2018 Talks*, ser. SIGGRAPH ’18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3214745.3214795

[5] W.-C. Wang, C.-W. Chow, L.-Y. Wei, Y. Liu, and C.-H. Yeh, “Long distance non-line-of-sight (nlos) visible light signal detection based on rolling-shutter-patterning of mobile-phone camera,” *Opt. Express*, vol. 25, no. 9, pp. 10 103–10 108, May 2017. [Online]. Available: http://www.opticsexpress.org/abstract.cfm?URI=oe-25-9-10103

[6] D. Meagher, “Octree encoding: A new technique for the representation, manipulation and display of arbitrary 3-d objects by computer,” *Rensselaer Polytechnic Institute*, 10 1980.

[7] V. Laurmaa, M. Picasso, and G. Steiner, “An octree-based adaptive semi-lagrangian vof approach for simulating the displacement of free surfaces,” *Computers Fluids*, vol. 131, pp. 190–204, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045793016300500

[8] Q.-Y. Zhou, J. Park, and V. Koltun, “Open3D: A modern library for 3D data processing,” *arXiv:1801.09847*, 2018.