




# NIGHT - Non-Line-of-Sight Imaging from Indirect Time of Flight Data

Matteo Caligiuri<sup>1</sup>, Adriano Simonetto<sup>1</sup>, Gianluca Agresti<sup>2</sup>, and Pietro Zanuttigh<sup>1</sup>

<sup>1</sup> Università degli Studi di Padova - DEI, Via Gradenigo 6/b 35131, Padova, Italia  
{matteo.caligiuri,adriano.simonetto,zanuttigh}@dei.unipd.it

<sup>2</sup> Sony Europe B.V. - SL1, Hedelfinger Str. 61 70327, Stuttgart, Germany  
gianluca.agresti@sony.com

**Abstract.** The acquisition of objects outside the Line-of-Sight of cameras is a very intriguing but also extremely challenging research topic. Recent works showed the feasibility of this idea exploiting transient imaging data produced by custom direct Time of Flight sensors. In this paper, for the first time, we tackle this problem using only data from an off-the-shelf indirect Time of Flight sensor without any further hardware requirement. We introduced a Deep Learning model able to re-frame the surfaces where light bounces happen as a virtual mirror. This modeling makes the task easier to handle and also facilitates the construction of annotated training data. From the obtained data it is possible to retrieve the depth information of the hidden scene. We also provide a first-in-its-kind synthetic dataset for the task and demonstrate the feasibility of the proposed idea over it.

**Keywords:** NLoS Imaging · ToF · Depth Estimation · Deep Learning

## 1 Introduction

Many different strategies have been proposed for the acquisition of 3D data, but, regardless of the technology used, a key requirement to acquire 3D information from a real-world scene is to ensure that there is a direct path between the sensor and the acquired point. This setting is denoted as Line-of-Sight (LoS) acquisition. Recent research showed that active devices could also be used for the extremely challenging task of Non-Line-of-Sight (NLoS) imaging. This essentially concerns being able to capture a scene or an object located outside the direct Field of View (FoV) of the sensor. This category of imaging goes one step further w.r.t. its LoS counterpart, analyzing the light scattered from multiple surfaces along indirect paths to reveal the 3D shape of objects located outside the LoS [7]. NLoS imaging using a 3D sensor is currently feasible only in very few scenarios where there is a simple way for the emitted light to reach the hidden scene. In particular, the *looking behind a corner* setting depicted in Fig. 2a can be

considered the baseline for this task. The goal of our work is thus to use an indirect Time of Flight (iToF) sensor to perform NLoS 3D imaging in this setup. More precisely, the objective is to build a learned approach that, given the raw iToF measurements as input, can retrieve the depth map of an object located behind a corner, outside of the direct FoV of the camera. We have chosen to use an iToF instead of direct Time of Flight (dToF) cameras, used in other works tackling this setting, because it guarantees a greater lateral resolution [26]. Furthermore, this kind of device is able to retrieve a more refined depth w.r.t. the one measured by a dToF. Notice also that most current NLoS works require custom cameras with impressive specifications while we target the usage of off-the-shelf consumer camera. This makes it more suitable for integration into other devices such as cars, robots and helmets. As of the time of writing, there is a limited number of approaches tackling NLoS imaging with iToF data and, consequently, a dedicated dataset for this specific application does not currently exist. For this reason, we have decided to build it from scratch.

The proposed approach exploits the *Mirror Trick*, an innovative idea that we have introduced to reframe the problem as a LoS one by interpreting the front wall as a virtual mirror, thus simplifying the definition of the ground truth in the dataset and the task itself. Note that the reflecting wall in the dataset could be either diffuse or glossy and the *mirror trick* is used only for re-framing the task. The data are then used to train a deep learning model that predicts sensor data in the *mirrored* setting and thus extracts depth information for the original task from the iToF data. The main contributions of this work are:

- **mirror trick**: A new approach to reframe a NLoS setting as a LoS one;
- **deep learning model**: A Deep Neural Network (DNN) able to estimate the depth in this setting using combined depth and shape losses;
- **iToF dataset**: The first synthetic dataset with iToF data for NLoS imaging.

## 2 Related work

NLoS perception using depth sensors is a new field, but there are already some works that pursue this goal [6, 14, 17, 23, 24, 27]. In this section, we focus only on the solutions based on the use of Time of Flight (ToF) sensors since these are closer to the proposed one. In particular, if we consider iToF sensors, we realize that only a few examples exist [11]. Furthermore, all of them assume to know some a priori information on the scene or to be in a simplified scenario. To our knowledge, we are the first to perform NLoS imaging using an off-the-shelves indirect Time of Flight sensor that illuminates in a single shoot (*full field*) a scene on which we know anything else but the fact that it satisfies the *behind a corner* setup. Indeed, the vast majority of existing works exploit data from Single-photon avalanche diode (SPAD) sensors embedded into direct Time of Flight cameras [5, 7–10, 12, 15, 16, 19, 25, 28–30, 32, 34]. This makes the task easier w.r.t. our setting since the direct and indirect light components are much simpler to separate. Another common trend in this field is to illuminate the scene not in a single shoot with a *full field illumination* but, rather illuminate a grid of

points on the front wall one by one. This makes the reconstruction of the hidden scene much simpler since each single illumination step will not interfere with the others. On the other hand, the acquisition procedure becomes very long.

**Analytical methods** To the best of our knowledge, the only work exploiting iToF data is [11]. This paper considers a huge simplification in the *look behind a corner* setup by spatially splitting the ToF device into two separate entities: the light emitter, located in the NLoS and the actual sensor, located in the LoS.

A key work in the field is [30], which introduces a novel approach to transient-based NLoS imaging. The authors introduced the concept of *Fermat path*, which corresponds to a specific path followed by a photon between the LoS and NLoS scene. Finally, they build the *Fermat flow* algorithm to perform shape recovery.

Heide *et al.* [9] exploits the fact that the temporal structure of the light, between a diffuse wall and a mirror, is left intact to recover objects outside the LoS. They formulated the reconstruction task as a linear inverse problem.

A joint albedo-normal approach to NLoS surface reconstruction using the Directional Line-Cone transformation (D-LCT) is proposed in [32]. The work uses a SPAD that, spot by spot, illuminates the illumination grid and expresses NLoS surface normal recovery as a vector deconvolution problem in time-resolved measurements, that is solved using the Cholesky-Wiener solver.

Kirmani *et al.* [12] proposed a new framework that uses transient data analysis to extract the properties of the NLoS scene. The theory is then supported by experiments using a femtosecond laser and an ultra-fast photo-detector array.

A combination of ToF data and computational reconstruction algorithms is used in [28] to untangle the image information mixed by diffuse reflections.

A NLoS imaging system that uses a single SPAD pixel to collect ToF information is proposed in [5]. This work focuses attention on practical aspects like power requirements, form factor, cost, and reconstruction time.

**Deep Learning based methods** The work by Shen *et al.* [25] aims to fuse NLoS reconstruction with the Neural Radiance Field (NeRF) approach. More precisely, they developed a Multi-Layer Perceptron (MLP) to represent a spherical volume Neural Transient Field (NeTF). The approach requires the use of a SPAD sensor that illuminates spot by spot a sparse grid.

A tailored autoencoder architecture, trained end-to-end, that maps NLoS transient images directly to a depth map is proposed in [8]. The results are computed over a synthetic simulation of SPAD measurements.

Wang *et al.* [29] proposed a general learning-based pipeline for NLoS imaging using only a few acquisition points. A Neural Network (NN) is tailored to learn the operator that recovers the high spatial resolution signal.

A synthetic photon ToF histogram with picosecond temporal resolution is fed to a compressive imaging algorithm that exploits Deep Learning (DL) in [34].

### 3 Basic principles on Time of Flight imaging

Before introducing the proposed method, we recall some basic principles of ToF sensors. There are two main families of ToF sensors: *indirect Time of Flight (iToF)* and *direct Time of Flight (dToF)*. Independently from the ToF technology used, the measured data could be represented in different formats, in this work we have decided to use the *phasorial* one.

#### 3.1 Indirect Time of Flight

Indirect Time of Flight sensors do not measure the distance directly from the time of flight along the light ray but instead, calculate it indirectly from the phase shift  $\phi$  between the emitted modulated signal and the received one [33]. Ideally, the camera is assumed to be composed of two co-located elements: the emitter (light source) and the sensor (detector). Notice that in a real device, they cannot be exactly in the same location and it is necessary to perform some additional compensation steps. The source emits an amplitude-modulated light signal ( $o$ ) that travels toward the scene and is reflected back to the detector. The detector convolves the received signal ( $r$ ) with a square signal characterized by the same frequency ( $f_m$ ) of the emitted one. By sampling this convolution on 4 points, it is possible to extract the phase shift ( $\phi$ ) between the two signals and use it to compute the distance. ToF devices based on this principle are also known as Amplitude Modulated Continuous Wave (AMCW) ToF. From the convolution samples, it is then possible to compute the amplitude ( $A$ ) of the reflected signal and the phase shift ( $\phi$ ) [33]. In particular, using the phasor notation we get:

$$c = Ae^{j\phi} = A \cdot (\cos(\phi) + j \sin(\phi)) = \Re + j\Im, \quad (1)$$

the amplitude and phase correspond to:

$$A = \sqrt{\Im(c)^2 + \Re(c)^2}, \quad \phi = \text{atan2}(\Im(c), \Re(c)). \quad (2)$$

The  $\text{atan2}(\cdot)$  function is used to capture the full  $2\pi$  range of the phase. At this point, using the phase ( $\phi$ ), the modulation frequency ( $f_m$ ) and the speed of light ( $c$ ), we can simply compute the target point distance ( $d$ ) as:

$$d = \frac{c}{4\pi f_m} \phi. \quad (3)$$

For a better comprehension of the data generated by an iToF, it is essential to remember that the amplitude ( $A$ ) directly correlates with the intensity of the incident light beam striking an object. The phase shift ( $\phi$ ), conversely, is proportional to the depth of the object that reflected such a light ray.

#### 3.2 Direct Time of Flight Cameras

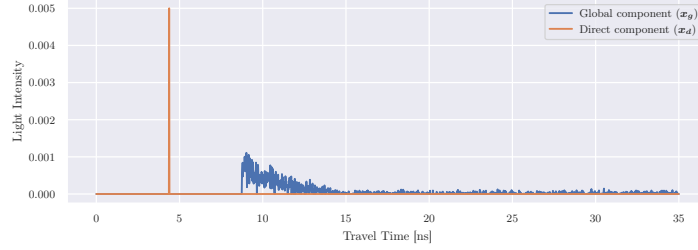
Direct Time of Flight cameras work by generating a Near-InfraRed (NIR) light pulse of known duration and then discretizing the front of the reflected light.



The discretization is performed before the return of the whole light pulse using a fast camera shutter. The resulting portion of the reflected light signal is the element that describes the observed object. Using this approach, the depth is directly linked to the time delay of the received signal [13]. Indeed, knowing the light pulse duration ( $t_{\text{pulse}}$ ) and the shutter time ( $\delta_s$ ), it can be computed as:

$$d = \frac{c}{2(t_{\text{pulse}} + \delta_s)}. \quad (4)$$

The output produced by this kind of sensor is sensibly different from the one of an iToF. A dToF can return for each sensor pixel a *transient vector* ( $\mathbf{x}$ ) that describes the light intensity received at each time instant. This, as shown in Fig. 1, can be easily separated into its *direct* ( $\mathbf{x}_d$ ) and *global* ( $\mathbf{x}_g$ ) components, a challenging task for iToF data. This difference with its indirect counterpart is the key aspect that makes performing NLoS imaging easier using a dToF sensor. Finally, note that converting a dToF output to an iToF one is a straightforward analytical approach (see [4] for the mathematical details) while the reverse is an ill-posed problem. Starting from the dToF transient vector ( $\mathbf{x}$ ) and the iToF measurement's model ( $\Phi$ ), the iToF data can be obtained as  $\mathbf{c}_{\text{mf}} = \Phi \mathbf{x}$ .

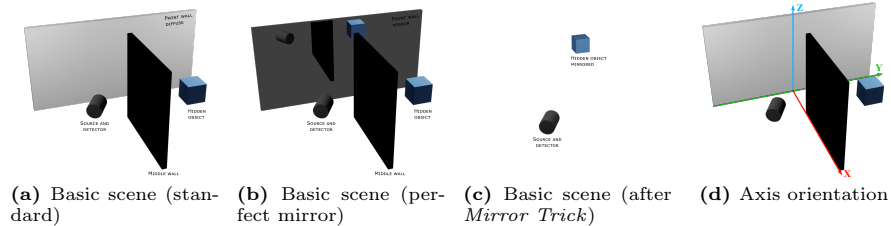


**Fig. 1:** Transient vector representation

### 3.3 NLoS perception using ToF cameras

Regarding the task of Non-Line-of-Sight imaging using a ToF sensor, the main idea is to analyze the output of a dToF and from that extract the *global component* ( $\mathbf{x}_g$ ) excluding the direct one, since an object located in NLoS, for sure, will not be the closest object to the sensor. It is important to note that even if we can precisely extract the global component, discerning which element corresponds to noise, or to other reflections in the scene, and which to the hidden object is an extremely difficult and ill-posed task. Currently, most of the state-of-the-art approaches perform the extraction of useful data from ( $\mathbf{x}_g$ ) using some kind of DL model. Furthermore as stated in [11] one of the biggest limitations that arise from the use of off-the-shelves ToF device for NLoS imaging is the fact

that the information coming from the hidden scene is carried by light rays that have performed at least three reflections. Each of them starts from the emitter, bounces on the front wall, then on the hidden object, then again on the front wall and finally, it reaches the sensor. Note that this represents the most ideal path since, in most cases, there will be more reflections before the light rays reach the sensor. This means that, even in the best scenario, such information necessarily travels a long path and so, the measured intensity greatly reduces, making the process of discerning between noise and useful information even harder. This problem is even more severe if we consider that the reflections coming directly from the front wall are extremely more intense and could saturate the sensor. As it is clear from what was said above, the standard approach is to extract meaningful information about the NLoS scene from the *global component*. Using an iToF instead of a dToF makes the task more complex. Indeed, as stated in Sec. 3.2, the iToF output doesn't explicitly distinguish between direct and global components, making it harder to extract information from the hidden scene. It can be inferred from this general introduction that NLoS applications using ToF technology are restricted to functioning in just a few specific scenarios, such as *look behind a corner*, due to the necessity that the light reflected from the hidden objects reach the sensor with a relatively simple path.



**Fig. 2:** Representation of the *Mirror trick*.

## 4 Proposed method

As anticipated in Sec. 1 the proposed approach for Non-Line-of-Sight imaging is based on a Deep Learning model working on iToF data. The most straightforward approach is to use a *supervised learning* model that takes in input the iToF data and tries to predict the geometry of the hidden object. More in detail, the network should directly output the depth map of the NLoS scene as if the point of view is located on the front wall looking in the direction of the object. Unfortunately, in this way, the relation between the input and output is very complex to model and the NN struggles in solving this task, thus leading to poor results. A possible strategy to simplify the problem is to train the network to estimate the iToF data corresponding to a LoS scene, relieving the model from learning also how to perform the conversion between ToF and depth domain.

Of course, this approach still has its challenges, indeed, it is extremely hard to accurately simulate the iToF data of an unknown scene. For these reasons, we introduce a novel re-formulation of the problem that makes it more tractable and easier to solve for the learning model and also, simplifies the construction of the ground truth for training data.

#### 4.1 The mirror trick

As previously introduced, we aim at simplifying the task that the network model has to perform, reducing as much as possible the implicit transformations. To this aim, we introduce the *Mirror trick*. If we consider the toy scenario in which the front wall is a mirror, the NLoS settings do not hold anymore and the retrieving of the hidden object becomes straightforward. Exploiting some fundamental principles of optics, we can effortlessly explain this behavior. When the front wall functions as an ideal mirror, the reflections of incoming rays are perfectly specular, leading to a substantial simplification in the geometric description of the scene. With these specular reflections, we can now treat the trajectory traversed by the ray emitted from the sensor and its subsequent reflection off the front wall as a solitary ray with a length equivalent to the sum of the individual ray lengths between the sensor and the front wall and between the front wall and the object. A similar analysis applies to the returning path (from the object to the front wall and then to the sensor). The described setup is depicted in Fig. 2b. Here, it is possible to see that the ToF sensor could directly see the object in the NLoS area as if it is in LoS. Furthermore, if the mirror used is ideal, given how a ToF camera works, the depth map produced by the sensor in a setup like the one of Fig. 2b, is the same as that produced from the setup of Fig. 2c. Essentially, if the front wall is an ideal mirror it is possible to assume that the NLoS object appears to the sensor as if it is flipped around the front wall plane. The *mirror trick* also allows our method to be completely agnostic to the material of which the front wall is composed. Indeed, if we found a way to teach a DL model how to convert a surface of unknown material to something as close as possible to an ideal mirror, we will be able to retrieve information about the NLoS scene with a small effort. To summarize, the *mirror trick* is a clever approach to handle *look behind a corner* setup without any knowledge about the front wall material. In particular, a learning model exploiting it should be able to take in input iToF measurements of a scene like the one of Fig. 2a and transform them in such a way that the obtained data still represents iToF measurements data but acquired in a setup like the one depicted by Fig. 2b. Once we've reached this goal, the obtained data is the same as the one coming from a scene like the one in Fig. 2c thus extracting the NLoS scene becomes a straightforward operation. We want to stress that the application of the *mirror trick* does not require the front wall to be a mirror, but is used to convert a general *look behind a corner* scene, with no further assumption on the used materials, to one that uses an ideal mirror as a reflective wall. Another benefit of using the *mirror trick* is that it simplifies the Ground Truth (GT) data generation process. This represents a key step of our work, since generating good quality data, that can be reliably

used as Ground Truth, is extremely challenging in this field. The idea behind the ground truth data generation is very similar to the one presented in the paragraph above: it can be directly generated by assuming that the front wall is an ideal mirror. Under this assumption to generate each GT sample, it is sufficient to take each scene in the standard setting (Fig. 2a), remove the front wall, and flip the hidden object across the wall plane (Fig. 2c). After applying the *mirror trick*, we simulate the ToF data for the obtained scene by feeding it into the ToF simulator (*Mitsuba 2 Renderer* [18]), as explained in Sec. 5.

## 4.2 Deep learning framework

In this section, we detail the input and output data, the employed Neural Network (NN) model, and the loss function used.

**Input data** The input is defined as raw iToF measurements of a *look behind a corner* scene like the one in Fig. 2a. We decided to represent the data in phasor format so that each measurement has a real ( $\Re(c)$ ) and an imaginary ( $\Im(c)$ ) component. In order to provide the network with enough data, and following the same rationale presented in [1, 26] we have decided to perform each measurement at three different acquisition frequencies: 20MHz, 50MHz and 60MHz. Doing that, as stated also in [4] it is possible to get different interfering characteristics between the visible wall and the hidden object. To improve stability to such large input values, a normalization step is also required: we divided each component of the iToF measurement by the corresponding amplitude at 20MHz, calculated using Eq. (2). By doing that we ensure that the value range of the input data is  $[0, 1]$ . A side effect of this operation is the removal of the information about the amplitude at 20MHz, which is useful for extracting information about the noise level of the measurements. To overcome this issue, we have decided to add this information (divided by a constant to bring it also in the  $[0, 1]$  range) as an additional channel to the normalized iToF data. After this small pre-processing, we feed the network with a matrix of dimension:  $b_s \times 7 \times w \times h$ . Where  $b_s$  is the batch size, while  $w$  and  $h$  are the image dimensions. The second dimension contains the 20MHz amplitude followed by the three normalized real components and the three normalized imaginary components (one for each frequency).

**Data augmentation** In order to maximize the amount of information that the DNN can gather from the input data, we performed an augmentation step of the training data. More precisely we applied a random rotation in  $[\pm 180]^\circ$ , a random translation in  $[\pm 0.2 \cdot (w, h)]$ px, flipped samples vertically and horizontally and added Gaussian noise with mean  $\mu = 0.0$  and standard deviation  $\sigma = 1.0$ .

**Output data** As for the output, we train the network to predict the iToF data, only at 20MHz, corresponding to the scene obtained by applying the *mirror trick* to the input (Fig. 2c). This means that the network produces an output iToF measurements of the input scene where the front wall is transformed into an ideal mirror. In this way, we reduce to the minimum the complexity of the task the network has to perform internally, and at the same time, we obtain an output

from which it is straightforward to compute the depth map of the hidden object. Indeed, from the network output, it is enough to use Eq. (2) together with Eq. (3) to obtain the desired depth map. For coherence with the input iToF samples, the output ones are expressed in phasor form ( $\Re(c)$  and  $\Im(c)$ ).

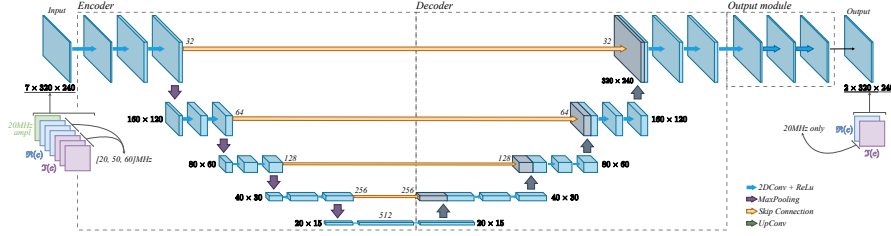


Fig. 3: Schematized representation of the proposed NN model

**NN architecture** We work with input data that can be seen as a multi-channel image (each layer of the input data is a 2D matrix where each pixel contains an intensity value) from which we want to retrieve another image-like representation, similarly to a denoising or image translation problem. For this reason, the use of a 2D convolutional model is a fairly straightforward choice. In particular, as it is possible to see from Fig. 3, we used an Encoder-Decoder architecture similar to a *U-Net* [20]. This model has been widely used for segmentation and denoising tasks and it has good convergence properties even with a limited amount of data. This is a key requirement in our case since there is no existing dataset and we could not build a very large one. Finally, we concatenated to the output of the Encoder-Decoder section a set of standard 2D convolutional layers (with ReLU activation function) to fine-tune the prediction and reduce the number of output channels to 2, *i.e.*, the two components of the output iToF data.

**Loss function** The loss function of the proposed model aims at jointly optimizing the shape of the hidden object and the depth value of its points. To this end, the loss function is designed as a combination of two terms:

$$L_{\text{MAE}} = w_l L_{\text{MAE}}^{\text{obj}} + L_{\text{MAE}}^{\text{bgr}}, \quad L_{\text{IoU}} = 1 - \text{IoU}(s_p, s_t); \quad L = \frac{L_{\text{MAE}} + L_{\text{IoU}}}{2}. \quad (5)$$

The first term,  $L_{\text{MAE}}$ , is a modified version of the standard Mean Absolute Error (MAE) loss function on the depth values. Since in our setting, there are many more pixels corresponding to the background than corresponding to hidden objects, we balanced the contribution of the object and background in the MAE loss in order to give a similar relevance to both regions. We separately calculated the MAE loss on the object region  $L_{\text{MAE}}^{\text{obj}}$  and the same loss on the background  $L_{\text{MAE}}^{\text{bgr}}$  and then computed the loss as a weighted sum of the two. In particular, the weighting parameter is computed as  $w_l = \alpha \cdot \frac{|B|}{|O|}$  where  $\alpha$  is empirically set to  $1/7$  while  $|B|$  and  $|O|$  are, respectively, the number of pixels

in the background and object regions. The second term  $L_{\text{IoU}}$ , is computed using the Intersection over Union (IoU) metric across the predicted and ground truth segmentation maps  $s_p$  and  $s_t$ , to help the NN model distinguish between the object and background regions (foreground/background separation) and ultimately helps the algorithm to better reconstruct the precise shape of the hidden object. The computation of the segmentation maps is done by thresholding the depth values, a non-differentiable operation. This represents a challenge for the training procedure since after that, it is no longer possible to backpropagate the gradient. To solve this issue, we implement this thresholding function together with a Straight-Through Estimator (STE) [31] in order to perform the non-differentiable operation only during the forward step, while in the backward step, we perform a simple linear, and so differentiable operation. Before implementing the IoU together with the STE, we also tested the Lovász-Softmax loss [2] as an alternative method for direct optimization of the IoU, but this loss led to lower performances.

**Training parameters** The training procedure was carried out for 695 epochs on the 2833 training samples obtained from the proposed dataset after applying data augmentation. It took  $\approx 4\text{h}$  and 40min using the Adam optimizer with a learning rate of  $lr = 1 \cdot 10^{-4}$ . The training was set to run for 1500 epochs, but we applied an early stopping criterion, i.e., the learning procedure was automatically stopped after 150 consecutive epochs with no validation error improvement. The model was trained on a *NVIDIA RTX 2080Ti* with 11GB of RAM.

## 5 Proposed dataset

It has been acknowledged in Secs. 1 and 2 that the utilization of iToF sensors for NLoS imaging is a relatively unexplored area within the research community. For this reason, we also needed to build an ad-hoc dataset from scratch. To be able to generate a dataset large enough for NNs training, due to the complex acquisition process, the only option was to construct a synthetic one. By employing this methodology, we are able to precisely control each acquisition parameter involved in the process. We acknowledge that real-world experiments are the key next step but since this is the first work to tackle this task, demonstrating its feasibility on synthetic data is a reasonable target. Anyway, to move to real data, a good strategy would be to pre-train the model on a large synthetic dataset and then adapt or fine-tune it using real data. This makes the proposed dataset a great starting point also for future works tackling real data. A generic dataset scene is shown in Figure 2a: this is a relatively simple setting but a reasonable starting point to explore this new task.

### 5.1 Rendering pipeline

To generate the dataset we needed a way to automate the construction of each 3D scene and a rendering engine able to render it as if it were observed by a Time of Flight sensor. We have built an automatic pipeline in *Blender*

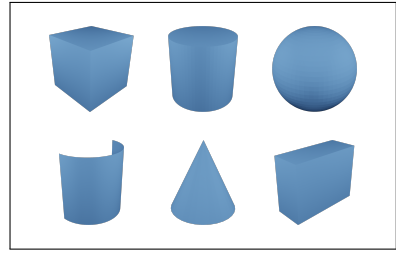
2.83 [3], which, starting from a basic scene, generates many variations of it. From these scenes, using the `Mitsuba Blender Add-on` [21], we were able to convert them into the format required by the rendering engine. We used the `Mitsuba2 renderer` [18] since it is an extremely realistic open-source renderer. More precisely, since we need ToF data in the output, we have used the modified version `Mitsuba2-transient-nlos` [22] that can render the scene as a *transient vector*. We have chosen this specific renderer since it is based on *ray tracing* technology and accurately simulates the Bidirectional Reflectance Distribution Function (BRDF) of all the surfaces in the scene. These two elements are fundamental to ensure that the rendered scene is as close as possible to a real-world acquisition. The last step was to convert the obtained data from dToF to iToF.

## 5.2 Structure of the dataset

We generated a dataset composed of 1344 scenes: each scene is acquired using  $10^6$  rendering samples, 2000 temporal bins of size 0.01s (that is equivalent to 0.01m), a sensor horizontal FoV of  $60^\circ$  and a resolution of  $320 \times 240$  pixels (QVGA). Another key aspect is the fact that the illumination is *full-field*, i.e., the whole scene is acquired and illuminated in a single step. This means that if we capture the same scene in the real-world the acquisition process requires a single shoot (allowing real-time). This last characteristic highly differentiates our model from those based on SPAD sensors, since they are limited to a sparse acquisition performed one dot at a time (much longer acquisition time). An essential aspect of a synthetic dataset is that it has to guarantee a high enough level of variability. This is fundamental since the data contained in the dataset must contain enough information to allow a NN model to learn the desired task. In order to ensure the aforementioned variability, we decided to consider different locations and orientations for the hidden objects and also to allow the front wall to assume different roughness values. We consider a set of primitive geometrical shapes for objects, i.e., the ones presented in Fig. 4 with their locations, orientation and dimensions. Possible positions and rotations are randomly sampled. In addition to those basic shapes,  $\frac{1}{7}$  of the scenes contains, as a hidden object, a random combination of two primitives sampled randomly. The distance between the two walls is fixed at 70cm. Finally, the roughness value of the front wall can take values in  $\alpha \in [0.3, 1]$  with a step 0.05. This corresponds to a variation in the glossiness of the wall, which implies that our dataset does not contain only diffuse surfaces. Code and dataset are available here<sup>3</sup>.

**Train-test split** A key aspect of a dataset is how has been split into train and test sets. In particular, we have used 1075 images for training and 269 for testing. It is important to notice that the object present in the former sets will never be present in the latter in neither the same location and orientation. Other than that, since the “composed objects”, built mixing two “basic shapes”, are randomly generated, the ones present in the training set will never be present in the test

<sup>3</sup> <https://github.com/matteocali/nlos-imaging-from-itof-data>



OBJECT	POSITION	ROTATION
<b>front wall</b> (8 × 4)m	X: 0m	0°
	Y: 0m	90°
	Z: 2m	0°
<b>middle wall</b> (3.30 × 4)m	X: 2.35m	90°
	Y: 0m	0°
	Z: 2m	0°
<b>source/detector</b>	X: 1m	90°
	Y: -1m	0°
	Z: 1.65m	50°
<b>hidden object</b> up to 0.5m <sup>3</sup>	X: [1, 1.3]m	[-90, 90]°
	Y: [0.5, 1.3]m	[-90, 90]°
	Z: [1.25, 1.95]m	[-90, 90]°

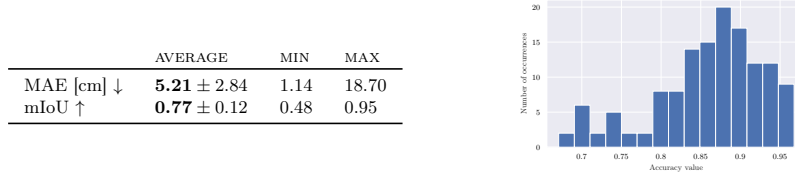
**Fig. 4:** Sample objects with their locations, orientations and dimensions.

set. Regarding the front wall’s roughness,  $\alpha$  spans over the same interval in both sets ensuring that the model learns to handle different materials.

## 6 Experimental Results

We evaluated our approach over the test set of the introduced dataset. In order to properly assess the results we will consider both the accuracy of the reconstructed object’s shapes and the depth estimation error. For the former, we computed the mean Intersection over Union (mIoU) between the predicted and Ground Truth object shapes, obtained by flattening the depth map to a 2D image and considering as object everything that is not background. For evaluating the accuracy of the actual depth, instead, we will use the Mean Absolute Error (MAE) between the estimated values and the GT ones. In Fig. 5 we show some visual results. More precisely, we include five examples of predictions (one for each row). The first two columns contain the real ( $\Re(c)$ ) and the imaginary component ( $\Im(c)$ ) of the predicted iToF (output of the network), while the last column contains the depth map computed from the predictions. Each column compares the ground truth (left) with the corresponding prediction (right). From this image, it is possible to qualitatively see that the model can recover the overall shape of the target reasonably well, indeed our approach obtains an average mIoU of 0.77. Performing a more in-depth analysis we could see that the model performs best over cylindrical and spherical shapes, while the most challenging shape is the concave plane due to its thin shape, especially if not oriented straight to the camera. Regardless of that, it is important to note that from our tests the model is able to maintain almost the same reconstruction accuracy over simple objects and composed ones (*i.e.* the scenes with two objects inside). For a more quantitative analysis, we can refer to the plot in Tab. 1: most of the sample predictions reach an overall accuracy, over the test set, between 80% and 95%. This represents a good result considering the very challenging scenario in which we are working. Indeed, considering that we illuminate the scene *full-field* reducing the acquisition to a single shoot, differently from the long acquisitions of most dToF-based approaches, the achieved performance is really promising. In Tab. 1 we also present the depth estimation error. It is possible to see that the network recovers the depth of the scene with an average error of 5.21cm, which, consider-



**Table 1:** Results over the test set. The plot shows the accuracy values distribution.

ing that the mean distance between the sensor and the object is around 3m is an encouraging result (the error is around 1 – 2% of the distance). It is important to note that in some cases the model can make an almost perfect prediction (in the best case, the error is only  $\approx 0.11\text{cm}$ ). To better visualize the depth reconstruction of the model, in Fig. 6 we also show the point cloud of each predicted sample of Fig. 5 together with the related ground truth. Analyzing Figs. 5 and 6 together we can confirm the evaluation done on the quantitative results. We can see that the model is able to correctly locate the hidden object and predict its dimensions. The most challenging aspect is the precise reconstruction of the shape. The first two samples have been recovered almost perfectly, while the last two shapes are not so accurate. Considering the third example, we notice that the network struggles a bit with the complex shape of the composed object, but is still able to predict a rough shape that, rather not being as accurate as the one of the cylinder or sphere, is still quite good.

### 6.1 Ablation study

Since there is no other work suitable for tackling the task for direct comparison, it is very interesting to show how the various design choices allow us to improve results w.r.t. baseline strategies. The findings of this analysis are in Tab. 2.

**Table 2:** Ablation scores for object shape (mIoU) and depth (MAE) over the test set.

SETTINGS	frequencies (MHz)			losses			our model MAE + IoU
	20	50	60	MAE	MSE	MSE + IoU	
MAE [cm] ↓	6.15	10.33	9.43	5.83	13.72	5.93	<b>5.21</b>
mIoU ↑	0.74	0.68	0.69	0.76	0.66	0.77	<b>0.77</b>

**Input Frequencies** Using multi-frequency data greatly helps the DNN to reach higher accuracy. Due to how an iToF works, higher frequencies ensure higher resolution but suffer from the *phase wrapping* [33] issue. On the other hand, using only the 20MHz frequency, we greatly reduce the influence of the *phase wrapping* but also affect the accuracy of the reconstruction. Taking into account this evaluation, it is clear that merging acquisitions at different frequencies leads to exploiting the strengths of each of them, minimizing the drawbacks.

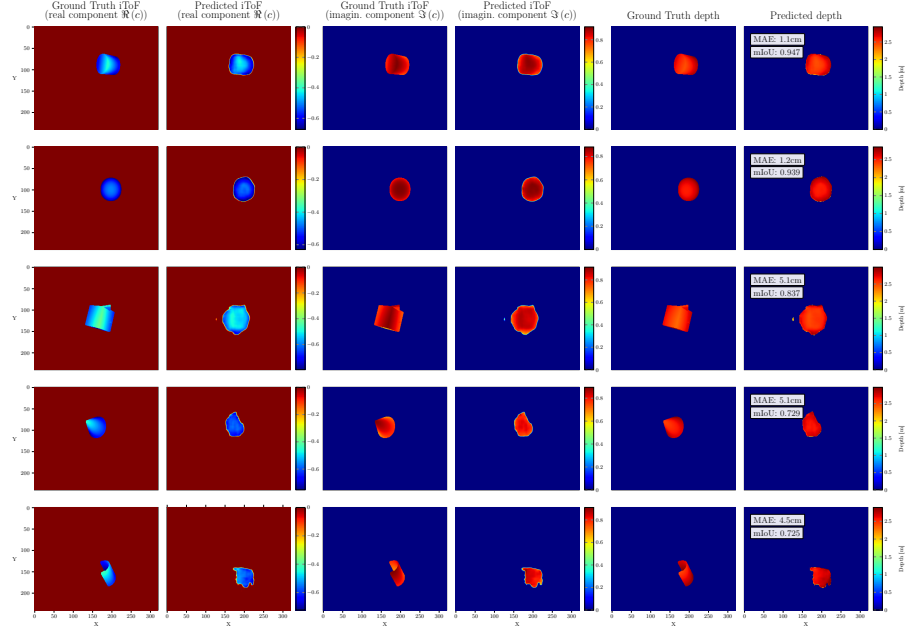
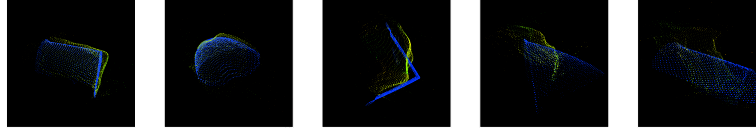


Fig. 5: Sample visual results

Fig. 6: Point cloud examples ( $[\bullet]=\text{GT}$ ,  $[\bullet \rightarrow \bullet]=[\text{closer} \rightarrow \text{further}]=\text{prediction}$ )

**Loss Functions** Training the model using only the MAE loss, the overall accuracy achieved by the network decreases, proving that the IoU metric helps to better identify the object. Furthermore, if we consider the qualitative results obtained by removing the IoU it is clear that the shape of the object is completely lost since the model tends to build a spherical object. If we use the Mean Square Error (MSE) loss instead of the MAE we can see that the error greatly increases, probably because more focus on smaller errors is needed. If we add the IoU to the MSE loss function, the accuracy of the model improves, but the combination of MAE and IoU is the best option.

## 7 Conclusions

We tackled the task of Non-Line-of-Sight imaging using an indirect Time of Flight sensor as the source of measurements: the proposed approach that combines the *Mirror trick* with an ad-hoc Deep Learning solution proved to be a

feasible solution for the task. The use of this specific type of ToF cameras, instead of the more common dToF, allows us to use off-the-shelf cameras, greatly reducing the cost and complexity of the system. Furthermore, by performing a *full field* illumination, the acquisition becomes extremely fast (multiple frames each second), allowing real-time applications. This, together with the good reconstruction results, makes the proposed approach an interesting alternative for Non-Line-of-Sight imaging. A key future step is to assess the approach with real-world data. The *Mirror trick* could be used to generate real-world Ground Truth by replacing the front wall with a mirror and reacquiring the scene.

### Acknowledgment

This work was supported in part by the European Union through the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, a partnership on “Telecommunications of the Future” (Program “RESTART”) under Grant PE0000001.

### References

1. Agresti, G., Schaefer, H., Sartor, P., Zanuttigh, P.: Unsupervised Domain Adaptation for ToF Data Denoising With Adversarial Learning. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5579–5586. IEEE, Long Beach, CA, USA (Jun 2019). <https://doi.org/10.1109/CVPR.2019.00573>, <https://ieeexplore.ieee.org/document/8954164/>
2. Berman, M., Triki, A.R., Blaschko, M.B.: The lovasz-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4413–4421 (2018)
3. Blender Online Community: Blender - a 3D modelling and rendering package. Blender Foundation, Blender Institute, Amsterdam (2020), <https://www.blender.org/download/releases/2-83/>
4. Buratto, E., Simonetto, A., Agresti, G., Schäfer, H., Zanuttigh, P.: Deep Learning for Transient Image Reconstruction from ToF Data. *Sensors* **21**(6), 1962 (Jan 2021). <https://doi.org/10.3390/s21061962>
5. Buttafava, M., Zeman, J., Tosi, A., Eliceiri, K., Velten, A.: Non-line-of-sight imaging using a time-gated single photon avalanche diode. *Optics express* **23**(16), 20997–21011 (2015)
6. Chen, W., Daneau, S., Brosseau, C., Heide, F.: Steady-State Non-Line-Of-Sight Imaging. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6783–6792. IEEE, Long Beach, CA, USA (Jun 2019). <https://doi.org/10.1109/CVPR.2019.00695>, <https://ieeexplore.ieee.org/document/8953404/>
7. Faccio, D., Velten, A., Wetzstein, G.: Non-line-of-sight imaging. *Nature Reviews Physics* **2**(6), 318–327 (2020)
8. Grau Chopite, J., Hullin, M.B., Wand, M., Iseringhausen, J.: Deep non-line-of-sight reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 960–969 (2020)

9. Heide, F., Xiao, L., Heidrich, W., Hullin, M.B.: Diffuse mirrors: 3d reconstruction from diffuse indirect illumination using inexpensive time-of-flight sensors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3222–3229 (2014)
10. Heide, F., Xiao, L., Heidrich, W., Hullin, M.B.: Diffuse Mirrors: 3D Reconstruction from Diffuse Indirect Illumination Using Inexpensive Time-of-Flight Sensors. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3222–3229. IEEE, Columbus, OH, USA (Jun 2014). <https://doi.org/10.1109/CVPR.2014.418>
11. Kadambi, A., Zhao, H., Shi, B., Raskar, R.: Occluded Imaging with Time-of-Flight Sensors. *ACM Transactions on Graphics* **35**(2), 1–12 (May 2016). <https://doi.org/10.1145/2836164>
12. Kirmani, A., Hutchison, T., Davis, J., Raskar, R.: Looking around the corner using transient imaging. In: 2009 IEEE 12th International Conference on Computer Vision. pp. 159–166. IEEE (2009)
13. Lefloch, D., Nair, R., Lenzen, F., Schäfer, H., Streeter, L., Cree, M.J., Koch, R., Kolb, A.: Technical foundation and calibration methods for time-of-flight cameras. In: Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications, pp. 3–24. Springer (2013)
14. Lindell, D.B., Wetzstein, G., Koltun, V.: Acoustic Non-Line-Of-Sight Imaging. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6773–6782. IEEE, Long Beach, CA, USA (Jun 2019). <https://doi.org/10.1109/CVPR.2019.00694>, <https://ieeexplore.ieee.org/document/8953208/>
15. Lindell, D.B., Wetzstein, G., O’Toole, M.: Wave-based non-line-of-sight imaging using fast  $f$ - $k$  migration. *ACM Transactions on Graphics* **38**(4), 1–13 (Aug 2019). <https://doi.org/10.1145/3306346.3322937>
16. Liu, X., Guillén, I., La Manna, M., Nam, J.H., Reza, S.A., Huu Le, T., Jarabo, A., Gutierrez, D., Velten, A.: Non-line-of-sight imaging using phasor-field virtual wave optics. *Nature* **572**(7771), 620–623 (Aug 2019). <https://doi.org/10.1038/s41586-019-1461-3>
17. Maeda, T., Wang, Y., Raskar, R., Kadambi, A.: Thermal Non-Line-of-Sight Imaging. In: 2019 IEEE International Conference on Computational Photography (ICCP). pp. 1–11 (May 2019). <https://doi.org/10.1109/ICCPHOT.2019.8747343>, [https://ieeexplore.ieee.org/abstract/document/8747343?casa\\_token=YaoXPjsjg18AAAAA:s6-\\_iy6mciKmtESj-xr7t61TGcMkEZoaxFHoPhtQiGdd4f7ZijNzQ53Jo2sKApIeMZxw4SltbQ](https://ieeexplore.ieee.org/abstract/document/8747343?casa_token=YaoXPjsjg18AAAAA:s6-_iy6mciKmtESj-xr7t61TGcMkEZoaxFHoPhtQiGdd4f7ZijNzQ53Jo2sKApIeMZxw4SltbQ)
18. Nimier-David, M., Vicini, D., Zeltner, T., Jakob, W.: Mitsuba 2: A retargetable forward and inverse renderer. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* **38**(6) (Dec 2019). <https://doi.org/10.1145/3355089.3356498>
19. O’Toole, M., Lindell, D.B., Wetzstein, G.: Confocal non-line-of-sight imaging based on the light-cone transform. *Nature* **555**(7696), 338–341 (Mar 2018). <https://doi.org/10.1038/nature25489>
20. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
21. Ros, D., Nicolet, B., Vicini, D., Vasilkovsky, M., Nimier-David, A.M.: Mitsuba blender add-on. <https://github.com/mitsuba-renderer/mitsuba-blender> (2020)

22. Royo, D., García, J., Muñoz, A., Jarabo, A.: Non-line-of-sight transient rendering. *Computers & Graphics* (2022). <https://doi.org/https://doi.org/10.1016/j.cag.2022.07.003>, <https://www.sciencedirect.com/science/article/pii/S0097849322001200>
23. Saunders, C., Murray-Bruce, J., Goyal, V.K.: Computational periscopy with an ordinary digital camera. *Nature* **565**(7740), 472–475 (Jan 2019). <https://doi.org/10.1038/s41586-018-0868-6>, <https://www.nature.com/articles/s41586-018-0868-6>
24. Scheiner, N., Kraus, F., Wei, F., Phan, B., Mannan, F., Appenrodt, N., Ritter, W., Dickmann, J., Dietmayer, K., Sick, B., Heide, F.: Seeing Around Street Corners: Non-Line-of-Sight Detection and Tracking In-the-Wild Using Doppler Radar. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2065–2074. IEEE, Seattle, WA, USA (Jun 2020). <https://doi.org/10.1109/CVPR42600.2020.00214>, <https://ieeexplore.ieee.org/document/9157505/>
25. Shen, S., Wang, Z., Liu, P., Pan, Z., Li, R., Gao, T., Li, S., Yu, J.: Non-line-of-sight imaging via neural transient fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **43**(7), 2257–2268 (2021)
26. Simonetto, A., Agresti, G., Zanuttigh, P., Schäfer, H.: Lightweight deep learning architecture for MPI correction and transient reconstruction. *IEEE Transactions on Computational Imaging* **8**, 721–732 (2022). <https://doi.org/10.1109/TCI.2022.3197928>
27. Tanaka, K., Mukaigawa, Y., Kadambi, A.: Polarized Non-Line-of-Sight Imaging. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2133–2142. IEEE, Seattle, WA, USA (Jun 2020). <https://doi.org/10.1109/CVPR42600.2020.00221>, <https://ieeexplore.ieee.org/document/9156511/>
28. Velten, A., Willwacher, T., Gupta, O., Veeraraghavan, A., Bawendi, M.G., Raskar, R.: Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. *Nature communications* **3**(1), 745 (2012)
29. Wang, J., Liu, X., Xiao, L., Shi, Z., Qiu, L., Fu, X.: Non-line-of-sight imaging with signal superresolution network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17420–17429 (2023)
30. Xin, S., Nousias, S., Kutulakos, K.N., Sankaranarayanan, A.C., Narasimhan, S.G., Gkioulekas, I.: A theory of fermat paths for non-line-of-sight shape reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6800–6809 (2019)
31. Yin, P., Lyu, J., Zhang, S., Osher, S., Qi, Y., Xin, J.: Understanding straight-through estimator in training activation quantized neural nets. *arXiv preprint arXiv:1903.05662* (2019)
32. Young, S.I., Lindell, D.B., Girod, B., Taubman, D., Wetzstein, G.: Non-line-of-sight surface reconstruction using the directional light-cone transform. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1407–1416 (2020)
33. Zanuttigh, P., Marin, G., Dal Mutto, C., Dominio, F., Minto, L., Cortelazzo, G.M.: Time-of-flight and structured light depth cameras. Springer (2016)
34. Zhu, S., Sua, Y.M., Bu, T., Huang, Y.P.: Compressive non-line-of-sight imaging with deep learning. *Physical Review Applied* **19**(3), 034090 (2023)