Time-of-Flight Coding Function Optimization

Jonas Messner
Stanford University
messnerj@stanford.edu

Nicholas Gaudio
Stanford University
nsgaudio@stanford.edu

Gordon Wetzstein*
Stanford University

Boris Murmann*
Stanford University

Abstract

In this work Continuous-Wave Time-of-Flight modulation and demodulation functions are optimized in order to improve the depth precision of Time-of-Flight systems. The depth precision dependency on modulation and demodulation functions is studied and two optimization pipelines are proposed. First, a non-neural network optimization approach is shown, resulting in an improvement of the state-of-the-art coding scheme by 6% in terms of mean squared error. Secondly, it is studied how a convolutional neural network can be employed in order to obtain depth maps based on only one or two measurements instead of the usually required three measurements in a conventional Time-of-Flight pipeline. Mean squared errors of down to $550\,mm^2$ *are achieved which demonstrates the ability of the CNN to recover depth maps based on an underdetermined system.*

1. Introduction

3D imaging has become an important area of research for various applications including autonomous driving, robotics, and gaming. There are various techniques to record a 3D image, including interferometry, triangulation and Time-of-Flight (ToF).

One practical ToF technique, usually implemented in commercial products, is Continuous-Wave Time-of-Flight (CW-ToF) imaging. Figure 1 depicts an overview of a CW-ToF system. A light source illuminates an object by emitting a continuous wave of light (*modulation function*), e.g. a sinusoidal or square wave. The light wave hits the object, gets reflected, and travels to the sensor. In CW-ToF the phase difference of emitted and received light is measured. This is accomplished by altering the sensor exposure according to the *demodulation function*, e.g. a sinusoidal or square function. The demodulation function can be thought of as the amount of light the sensor admits, where 0 is a shut sensor (zero exposure) and 1 is an open sensor (maximum exposure). As is derived in [7] for sinusoidal modulation and demodulation functions, the calculation of the phase difference is obtained by calculating the cross-correlation $C(\varphi)$ of the received light $R(t)$ and the sensor demodulation function $D(t)$.

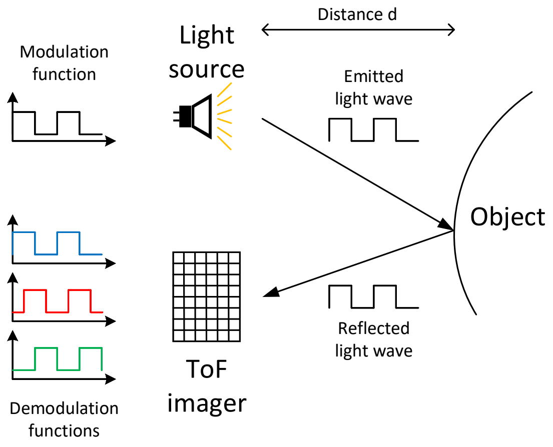


Figure 1: Overview of a Continuous-Wave Time-of-Flight system.

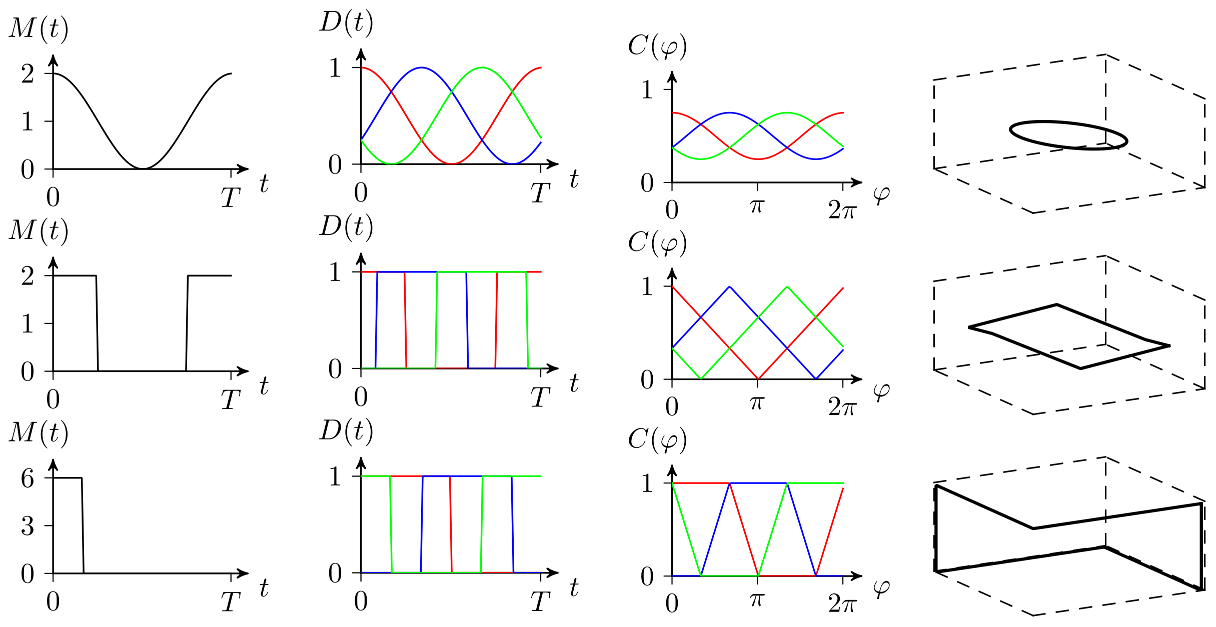
$$R(t) = 1 + \beta \cdot \cos(\omega t - \varphi) \tag{1}$$

$$D(t) = \cos(\omega t) \tag{2}$$

$$C(\varphi) = \frac{\beta}{2}\cos(\varphi + \omega\tau) + L_a \tag{3}$$

We notice that in order to recover the phase φ in Eq. 3 and thus the depth, we need to record at least three measurements to calculate the three unknowns β, L_a, and φ, where β represents the albedo (reflectivity) of the scene and L_a is the ambient light. Therefore, three or more demodulation functions are applied to the sensor. This is done by using

*Professor, Department of Electrical Engineering



(a) Modulation function. (b) Demodulation functions. (c) Correlation functions. (d) Coding curve.

Figure 2: Modulation, demodulation, correlation and coding curves of different coding schemes. Adapted from [3]. Row 1: Sinusoidal coding. Row 2: Square coding. Row 3: Hamiltonian coding.

so called K-tap lock-in pixels as described in [7]. Thus, K brightness measurements are obtained at each pixel. Larger values of K improve the resolution of the system at the cost of increased hardware complexity. Typically, a 4-tap lock-in pixel ($K = 4$) is applied in practical settings.

The modulation (source) and demodulation (sensor) functions are usually sinusoidal or square waves. However, in [3] it is shown that the correlation function $C(\varphi)$ and thus the modulation and demodulation functions inherently influence the depth resolution of a CW-ToF system. Specifically, it is shown that depth precision $\overline{\chi_C}$ is

$$\overline{\chi_C} = \frac{\beta_{mean}\, l_{curve}}{\Omega\, \varphi_{range}} \tag{4}$$

where β_{mean} is the mean albedo factor, Ω is the standard deviation of the brightness measurements due to noise, φ_{range} is the full range of the depth measurement, and l_{curve} is the length of the correlation function coding curve which is defined as

$$l_{curve} = \int_{\varphi_{min}}^{\varphi_{max}} \sqrt{\sum_{i=1}^{K} C_i(\varphi)^2 d\varphi} \tag{5}$$

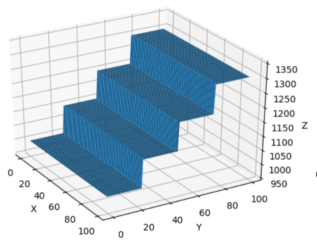
In order to illustrate the meaning of l_{curve} Fig. 2 shows the modulation (a), demodulation (b), correlation function (c), and the coding curve (d) of sinusoidal (row 1), square (row 2) and Hamiltonian (row 3) coding schemes. We denote a set of modulation, demodulation and correlation functions as coding schemes or coding functions. The coding curves in Fig. 2(d) are obtained by plotting the three correlation functions of one coding scheme over φ in a 3D-space. The parameter l_{curve} is the length of this coding curve.

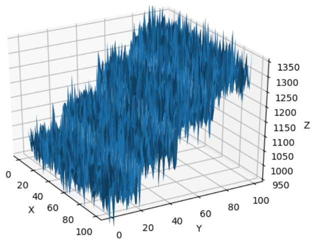
According to (4) it is desirable to maximize the length of the coding curve l_{curve}. [3] introduces Hamiltonian coding functions (row 3 of Fig. 2), which improves the depth resolution of the system compared to sinusoidal and square wave coding functions, but are not provably optimal.

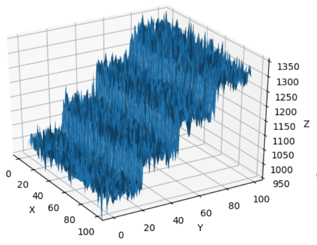
In order to provide intuition on how well depth maps can be recovered with the three different coding schemes, a 100x100 pixel 3D staircase model is simulated using the simulator of [3]. Figure 3 shows the results of the simulation. The ground truth depth map is shown in (a) and the recovered depth maps with sinusoidal, square, and Hamiltonian coding functions are shown in (b), (c), and (d), respectively. Hamiltonian coding outperforms both sinusoidal and square coding significantly which can not only be seen from the recovered shape but also from the mean squared error (MSE) that is used as a comparison metric in this work.

This work intends to employ two learning approaches to improving the state-of-the-art coding schemes:

1. Use the common CW-ToF imaging pipeline as depicted in Fig. 4(a) and backpropagate the MSE loss throughout the whole pipeline in order to optimize the coding functions (non-neural network approach).


(a) Ground truth.


(b) Sinusoidal coding. $MSE = 1643.21$.


(c) Square coding. $MSE = 624.66$.

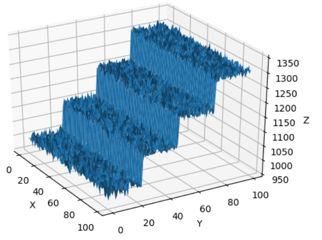

(d) Hamiltonian coding. $MSE = 113.20$.

Figure 3: Depth maps recovered using different coding functions.

The goal is to **outperform the Hamiltonian coding scheme** introduced in [3].

2. Replace the raw $\rightarrow$ depth decoding block with a fully Convolutional Neural Network (CNN) as shown in Fig. 4(b) and learn the depth map by both optimizing the weights of the neural network and the coding functions (data-driven neural network approach). As indicated by Eq. (3) at least three measurements are needed to calculate the depth in a conventional CW-ToF pipeline. With a CNN however, it is possible to **recover depth maps using only one or two measurements**.

2. Related Work

Coding Functions. Usually, sinusoidal or square waves are used in CW-ToF systems. Specialized coding functions are studied by M. Gupta *et al.* in [3]. They propose Hamiltonian coding which is compared to sinusoidal and square coding. A CW-ToF simulator was developed which can be used to compare different coding schemes. This CW-ToF simulator was adapted to be used in a Pytorch [10] learning environment in this work. In [4] it is investigated how Hamiltonian-like coding schemes can be practically implemented in actual imaging hardware. Further works with coding functions other than sinusoidal or square waves are [1], [2], [6], and [11].

Parametrized Functions. In order to ease learning it is sometimes helpful to reduce the degrees of freedom which can simplify learning. In [12] parametrized Sinc filters for speech recognition are learned instead of learning all points of a filter. This leads to faster convergence, fewer parameters, and computational efficiency. This scheme inspired the parametrized coding functions of this work which will be described in the next chapter.

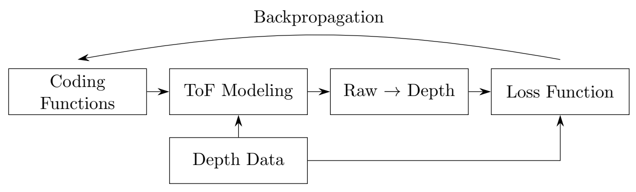
Reconstruction Architectures. Image segmentation/reconstruction performance is highly dependent on a network's ability to preserve (1) localization and (2) context. [5] suggests a "hypercolumn" localization solution that maintains fine-grained localization by leveraging the localization of earlier layers in addition to the last layer's feature representation. This localization retention is due to the fact that a hypercolumn at a particular pixel is the vector of all prior layer activations for that pixel (comparable to the idea of skip connections).

[14] focuses on preserving contextual information by proposing a multi-resolution contextual framework. Each level of this hierarchical framework is trained using downsampled versions of the input image and the outputs of the previous layers. The resulting multi-resolution contextual information is then incorporated into the output classification or reconstruction. This idea of downsampling is loosely related to the idea of spatial encoding. [15] proposes a fully convolutional neural network that encodes the input image to make dense predictions for per-pixel tasks like segmentation. The fully convolutional architecture permits arbitrary sized inputs and efficient training and inference compared to fully connected layers.

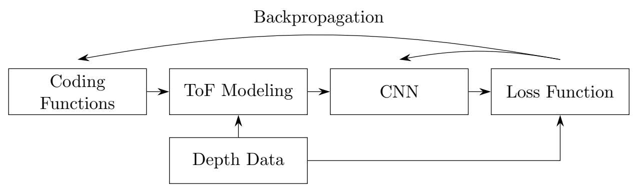
[13] proposes a fully convolutional encoding-decoding network that incorporates elements of [5], [14], and [15] preserving localization *and* context. This encoding-decoding network termed "U-Net" first contracts (encodes) the spatial dimensions, expanding the receptive field size to capture context and then expands (decodes) the spatial dimensions symmetrically with corresponding encoding to decoding skip connections preserving precise localization, which is important for image reconstruction tasks.

3. Methods

Although this work incorporates two optimization pipelines that differ from each other there are a few general considerations that hold for both tasks. As can be seen in Fig. 4, both pipelines model CW-ToF brightness measurements based on a given depth map and coding functions. This modeling step takes into account the cross-correlation of received light and demodulation functions as described in the introduction and also models photon noise and other imaging phenomena. The brightness measurements are then decoded to obtain a depth map. In the standard CW-ToF pipeline the decoding step is based on a closed-form solution based on three or more brightness measurements per pixel. In the CNN pipeline the decoding step is performed



(a) Non-neural network approach.



(b) Convolutional neural network as decoder.

Figure 4: Coding function optimization block diagrams.

by the neural network with parameters learned from data.

In both approaches we minimize the pixelwise MSE of the difference of the ground truth and predicted depth maps

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y_i})^2 \tag{6}$$

where n is the number of pixels, Y_i is the ground truth depth value, and $\hat{Y_i}$ is the predicted depth value.

Furthermore, it is common to both approaches that the CW-ToF coding functions, i.e. the modulation function and the demodulation functions, are optimized. We setup the coding functions in two different ways:

– Pointwise coding functions: Each function is setup as a vector with 10,000 points and the optimization is conducted in a pointwise fashion.

– Parametrized coding functions: Each function is setup as a Fourier Series with order N according to the form

$$y(t) = A_0 + \sum_{i=1}^{N} A_i \cdot \cos\left(i \cdot \omega t + \varphi_i\right) \tag{7}$$

where A_i and φ_i are the optimization variables. This approach is inspired by [12] and provides inherent regularization.

3.1. Optimization of the Standard ToF Pipeline

Figure 4(a) shows the non-neural network optimization block diagram. The coding functions are the input to the fully differentiable ToF modeling environment which is an adapted version of the simulator from [3] ported from Numpy to PyTorch. The modeling block takes the ground truth depth map as an additional input and outputs pixelwise ToF brightness measurements based on the depth map. These brightness measurements run through a raw-to-depth decoding step and are then compared to the ground truth depth map. The MSE between the decoded depths and ground truth depths is calculated and backpropagated through the network to update the coding functions and minimize the MSE loss.

This approach does not necessarily need a dataset as the coding functions can simply be optimized for all depths in a specified range (e.g. 0-10 meters). Every pixel is optimized by itself and there are no spatial connections (a receptive field of one is maintained throughout the pipeline). Theoretically, it is possible to choose one specific scene (e.g. 3D faces for face recognition) and generate scene-specific coding functions by inputting pictures of that scene. By doing this, the optimization would weight depths that occur frequently more heavily. However, the investigation of scene-specialized coding functions is beyond the scope of this project.

3.2. Convolutional Neural Network for Depth Estimation

If less than three demodulation functions are applied to the sensor ($K < 3$), the system of equations to resolve the phase φ and thus depth becomes underdetermined. In such cases the standard CW-ToF pipeline cannot be used. Instead, a CNN can serve as the decoder, learning a mapping of one or two brightness values per pixel ($K = 1, 2$) to a depth map. Although the CNN pipeline could also be used for systems with $K \geq 3$, we focus on using it only when $K < 3$. Figure 4(b) shows the block diagram of the parameter optimization using a CNN mapping from brightness values to depths where the loss is backpropagated through the pipeline to update both the network parameters *and* the coding functions to minimize the MSE loss.

Since the CNN is to construct a depth map from brightness values, the CNN performs a reconstruction task. Therefore, a fully convolutional encoder-decoder architecture analogous to the U-Net architecture [13] was initially tested. Encoding layer blocks comprised of two dimensional convolutions (kernel size=4, stride=2, padding=1), batch normalization, and ReLU activation components, resulting in activation maps of half the input spatial dimension and double the channel depth. Conversely, decoding layer blocks comprised of batch normalization, two dimensional transpose convolutions (kernel size=4, stride=2, padding=1), and ReLU activation components, resulting in activation maps of double the input spatial dimension and half the channel depth.

The encoding layers aid in preserving context as the encoded representation results in a large receptive field when

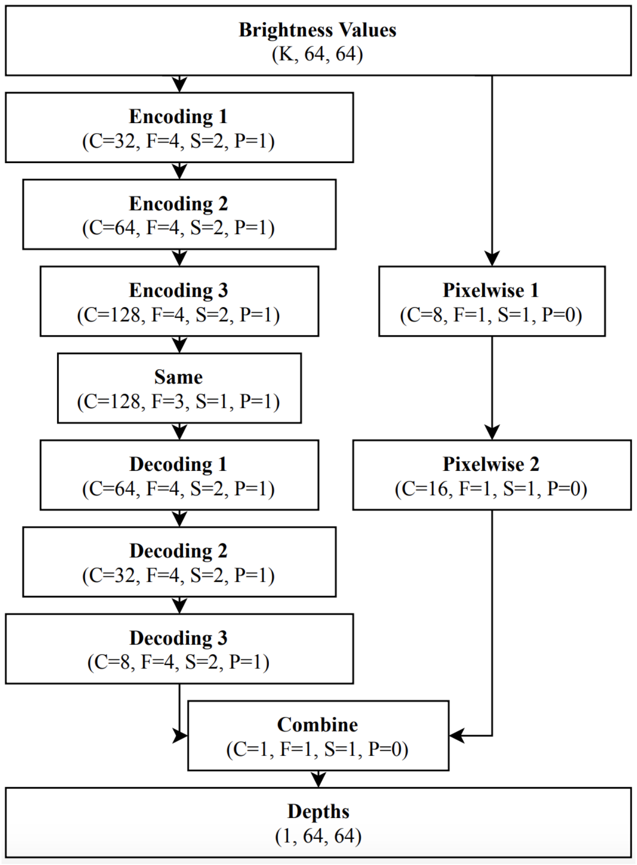


Figure 5: The convolutional neural network architecture.

combined sequentially. Another benefit of employing multiple, sequential encoding layers followed by their corresponding decoding layers is that the architecture allows for feature map spatial dimension symmetry between pairs of encoding and decoding layers. This allows for skip forward connections that can be added in attempt to preserve localized information in the deep network. However, it was empirically determined that the skip connections between all encoding, decoding layer pairs resulted in high MSE of the predicted depth map compared to the ground truth depth map and thus were all removed. We conjecture that skip connections do not make sense in our task since the input brightness values and the output depth map stem from very different distributions.

Upon visual inspection of the difference map between the ground truth and predicted depth maps, the encoder-decoder network performed poorly in high frequency areas of scenes such as edges where there is an abrupt depth change. This behavior is indicative of poor localization. Intuitively, this makes sense as oftentimes convolutional filters have slight low-pass filter effects on images due to their translational nature, resulting in smoothed images in reconstruction tasks. To mitigate this issue, while knowing that hidden layer skip connections (which are often employed to preserve localization) were ruled out, the idea of sending the input brightness values to be combined with the output of the encoder-decoder was developed.

The proposed fully convolutional network, as seen in Figure 5, contains an encoding-decoding branch (left) and a pixelwise branch (right). Knowing that the pixelwise branch has to map brightness values to depths, which is a nonlinear operation, two hidden layers were used to result in a large enough function space to model this transformation. Also of particular note is that the layers in the pixelwise branch contain 1x1 convolutions (kernel size=1, stride=1, padding=0), batch normalization, and ReLU activation components. The underlying notion is that to preserve the most localized information, using a pixelwise kernel would retain the most high frequency detail from the input brightness values all the way to the output depths. This is because a set of K brightness values at one pixel should correspond to one unique depth value at the analogous pixel location in the depth map. Combining the encoding-decoding branch information with the pixelwise branch information with a final two dimensional convolution (size=1, stride=1, padding=0) resulted in predicted depth maps with the lowest MSE when $K = 1$ or $K = 2$.

4. Dataset

The non-neural network approach described in Section 3.1 can be trained on random depth points and thus does not need a dataset.

In order to learn and evaluate our CNN architecture we use the NYU V2 depth dataset [9] which contains a broad range of indoor depth scenes (classrooms, living rooms, bookstores etc.). A sample RGB image and the respective depth map can be seen in Fig. 9 in Section 5.2.

The NYU V2 dataset contains 1,449 labeled 3D depth maps of VGA resolution (480×640 pixels). Deep image reconstruction tasks, such as in our application, consume an immense amount of memory. Patching is a method proposed by [8] as a solution to computationally running out of memory due too large input dimensions. We will resultantly employ this patching technique to get around our memory constraints. Each 480×640 depth map is patched into 64×64 patches. Thus, each depth map results in 63 patches[1], effectively expanding our dataset to 91,287 examples.

We split the dataset into training, validation, and test sets using a 90/5/5% split (82,159/4,564/4,564 images). Initially, we intended not to patch validation and test examples. However, in several tests we observed that the obtained MSE was far better when validating and testing on

[1]Theoretically, up to 70 patches can be obtained per image. Bad edge effects were observed in the dataset so we cut off 16/32 pixels of the 480×640 images at each edge.

patched images instead of full images. In order to obtain full 480×640 images at test time, the patched test images were stitched back together after inference. A comparison of training results with patched validation and test images vs. full validation and test images is shown in Table 2 in Section 5.2.

The ground truth depth map which is used in the ToF modeling block to model the ToF image (compare Fig. 4(b)) is not normalized since the modeling blocks works with raw depths given in millimeters. The output of the CNN is a depth map which is compared to the ground truth depth map (compare Fig. 4(b)). In this case the depth map is normalized to zero mean and unit variance. Consequently, normalized depth maps are learned.

5. Results

Subsequently, the training results of both optimization approaches are presented and discussed.

5.1. Optimization of Standard CW-ToF Pipeline

The MSE of standard coding schemes (sinusoidal, square, and Hamiltonian) serves as the baseline for the non-neural network optimization. The optimization is conducted in six different ways: The coding functions are initialized with three different coding schemes (sinusoidal, square, and Hamiltonian coding) and are trained with both the pointwise and parametrized coding setup. Backpropagation with an empirically determined learning rate of 0.01 was used to update the coding functions.

Table 1 shows the MSE of the three baseline coding schemes and the six trained configurations. All coding schemes were tested by generating 10,000 random depth points uniformly distributed in the allowable depth range and performing depth prediction on each pixel. A random seed is applied in order to have the same depth and noise settings for all schemes.

The results show that all learned coding schemes achieve better results than sinusoidal and square coding. The best MSE was achieved with a pointwise setup and initialization at Hamiltonian which even outperforms the Hamiltonian coding. The corresponding functions are shown in Fig. 6.

It is visible that the learned coding scheme is very similar to Hamiltonian coding (compare Fig. 2). The main difference is that the corners are smoothened compared to Hamiltonian coding. Intuitively, the better performance of the smooth corners makes sense as it helps to disambiguate the phase in the presence of noise at phases of multiples of $\pi/3$.

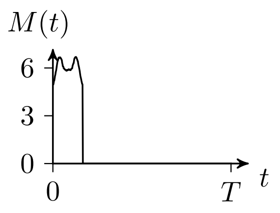
5.2. Convolutional Neural Network for Depth Estimation

Following the architectural progression described in Section 3.2 the encoding-decoding branch hyperparameter search was confined to determining the optimal learning rate (using the Adam optimizer) and the number of encoding / decoding layers. It was determined that the a learning rate of 3×10^{-4} resulted in the lowest MSE values. It was also determined that three encoding and three decoding layers is the optimal number. Adding more encoding and decoding layers resulted in decreased performance due to decreased localization which was seen in the difference maps.

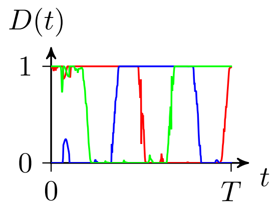
Our hyperparameter search was largely focused on the more unique, application-specific pixelwise branch and its combination with the encoding-decoding branch to predict depth maps as this idea is largely unexplored. Knowing

Coding scheme	MSE (in mm^2)
Standard coding	
Sinusoidal	1493.4
Square	583.7
Hamiltonian	180.1
Learned coding	
Pointwise init. @ Sinus	193.5
Pointwise init. @ Square	197.4
Pointwise init. @ Hamiltonian	**168.8**
Parametrized init. @ Sinus	297.2
Parametrized init. @ Square	354.8
Parametrized init. @ Hamiltonian	260.5

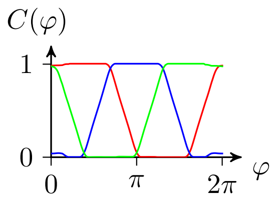
Table 1: MSE of depth maps generated with learned coding schemes.



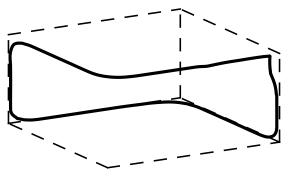
(a) Modulation function.



(b) Demodulation functions.



(c) Correlation functions.



(d) Coding curve.

Figure 6: Modulation, demodulation, correlation and coding curve of learned coding schemes with pointwise setup and initialization at Hamiltonian.

	MSE (in mm^2)
Validation & test on patched images (64×64)	
Parametrized coding ($K = 2$)	**548.7**
Pointwise coding ($K = 2$)	645.9
Parametrized coding ($K = 1$)	31156.5
Pointwise coding ($K = 1$)	**2571.4**
Validation & test on full images	
Parametrized coding ($K = 2$)	2518.3
Pointwise coding ($K = 2$)	2902.2
Parametrized coding ($K = 1$)	37971.1
Pointwise coding ($K = 1$)	17771.5

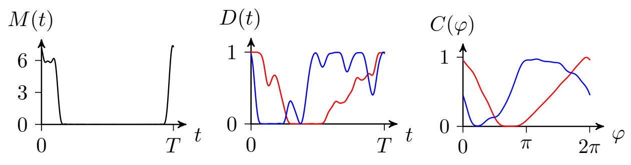
Table 2: MSE of depth maps generated with learned coding schemes and CNN.

that the kernel size has to be 1x1 to ensure that each set of brightness values (input) is mapped to a depth value (output), we tuned the number of pixelwise layers, number of channels per layer, and number of output channels in the encoding-decoding branch. It was determined that two pixelwise layers are to be used with filter numbers of 8 and 16 respectively. An encoding-decoding branch output channel number of 8 proved to be optimal combined with the 16 channels of the pixelwise branch.

Table 2 shows the test results of the depth estimation CNN based on an underdetermined system ($K = 2, 1$). As pointed out earlier, the table shows that validation and test on patched images leads to better scores than validation and test on full images. We conjecture that this is due to the different image distributions. Since we train on patched images, validating and testing on patches is easier than on full images.

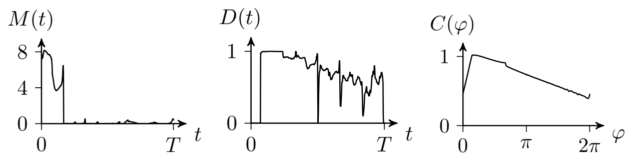
The best configuration with $K = 2$ (parametrized coding) achieves a MSE of $548.7\,\mathrm{mm}^2$. This is pretty close to the values achieved with $K = 3$ in the conventional CW-ToF pipeline (comp. results in Table 1). Figure 7 shows the learned coding functions for $K = 2$ with parametrized coding. Examining the correlation function in Fig. 7(c) it becomes clear that one distinct set of correlation function values is present for each phase φ. Intuitively, this is necessary to disambiguate different depths. This property was not enforced but learned by the neural network completely independently. In fact, the two demodulation functions were initialized at Hamiltonian which with $K = 2$ violates the property of having a distinct set of correlation function values for each phase φ. Figure 9 shows the depth map recovery using the CNN on one example of the test set. (a) and (b) show the ground truth RGB image and depth map, (c) shows the predicted depth map with $K = 2$, and (e) shows the difference between the ground truth and the prediction.

For $K = 1$ the pointwise coding function setup performs



(a) Modulation function. (b) Demodulation functions. (c) Correlation functions.

Figure 7: Modulation, demodulation, and correlation function of learned coding schemes with $K = 2$.



(a) Modulation function. (b) Demodulation function. (c) Correlation function.

Figure 8: Modulation, demodulation, and correlation function of learned coding schemes with $K = 1$.

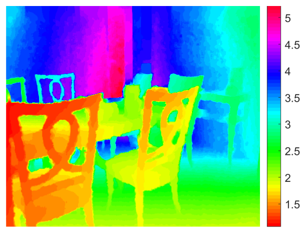
best and achieves a MSE of $2571.4\,\mathrm{mm}^2$. The learned coding function is shown in Fig. 8. The coding function was initialized with one of the Hamiltonian functions which clearly does not have unambiguous correlation function values for each depth. For $K = 1$ the correlation function can intuitively only be a monotonically increasing or decreasing curve which is in fact what the learned correlation function reflects. Figure 9 shows the predicted depth map with $K = 1$ for one test image. Again, (a) and (b) show the ground truth RGB image and depth map, (d) shows the predicted depth map, and (f) shows the difference between the ground truth and the prediction. Even though the MSE is significantly higher than with $K = 2$ the predicted depth map (d) shows a close match to the ground truth depth map.

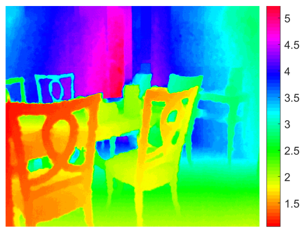
The difference images in Fig. 9 show slight grid lines induced by testing on patches and then stitching them back together to generate the full image. One solution to remove the noticeable grid lines is to leverage an overlap-patch strategy where 64×64 test patches are sampled every $64 - F$ pixels from the pre-patched brightness values and forward propagated to obtain depths, where F is the kernel size. Then, the outer $F/2$ pixels of each 64×64 patch are cropped and stitched. This algorithm would remove the grid lines that arise due to patching during test time, but is beyond the scope of the project.

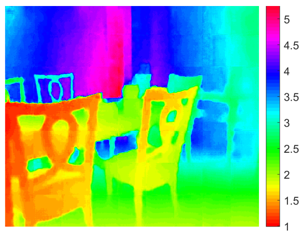
6. Conclusion & Future Work

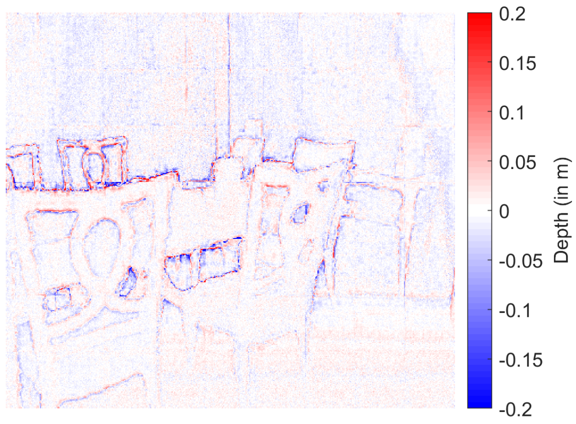
In this work we have developed two pipelines to optimize the modulation and demodulation functions for CW-

(a) Ground truth RGB.

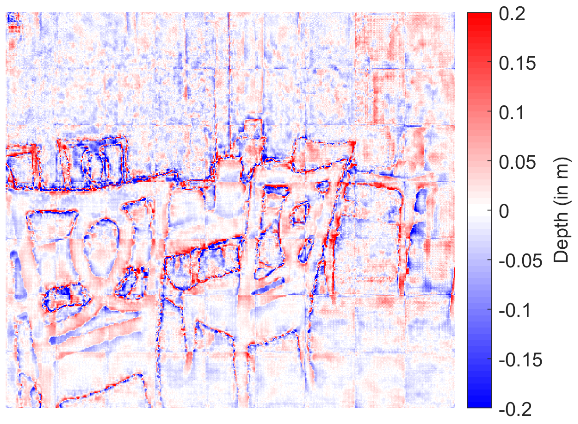

(b) Ground truth depth.


(c) Predicted depth $K = 2$, parametrized coding.


(d) Predicted depth $K = 1$, pointwise coding.



(e) Difference of ground truth and prediction for $K = 2$, parametrized coding.



(f) Difference of ground truth and prediction for $K = 1$, pointwise coding.

Figure 9: Results with $K = 1$. All depths given in meters.

ToF imaging in order to improve the depth precision of CW-ToF systems.

Using backpropagation in the conventional CW-ToF pipeline to optimize the coding functions we achieved a depth precision improvement of 6 % compared to the state-of-the-art Hamiltonian coding. Our developed coding scheme outperforms sinusoidal and square coding by a factor of 8.8 and 3.5, respectively.

A conventional CW-ToF system needs at least three measurements to calculate depth as there are three unknowns: Background light, albedo (reflectivity) and depth. We proposed a CNN that is able to resolve depth based on only one or two measurements. The CNN is split up into two branches. One branch acts as an encoding-decoding fully convolutional neural network to learn spatial features and the other branch acts on single pixels by applying 1×1 kernels to calculate the depth of each pixel individually. With two measurements, MSE values of down to $548.7\,\mathrm{mm}^2$ are achieved. When only having one brightness measurement MSE values of down to $2571.4\,\mathrm{mm}^2$ are obtained.

Among future paths are the exploration of scene-specialized coding functions, including further CW-ToF imaging artifacts such as multipath interference, regularizing modulation and demodulation function such that they become feasible in actual CW-ToF imaging hardware and exploring further CNN architectures in order to reduce the MSE.

Author Contributions

Both Jonas Messner and Nicholas Gaudio designed the CNN and wrote the final report.

Jonas ported the CW-ToF simulator code to Pytorch to build the learning pipeline. He conducted the non-neural network optimization and trained the neural networks. He created the report figures.

Nicholas wrote that scripts that prepare, patch, and stitch the dataset. He built training loop, validation/test code, and setup the hyperparameter tuning scripts. He made the poster.

Gordon Wetzstein gave advice on how to setup the optimization pipelines.

Boris Murmann provided advice in regular research meetings with Jonas. He also provided access to lab GPUs that were used to train the CNN.

The program code developed in this work can be

accessed by CS231N staff on:
`https://github.com/nsgaudio/ToF-Coding-Function-Optimization`.

Acknowledgements

We thank M. Gupta and F. Gutierrez Barragan for providing the CW-ToF simulator code. Their code can be found on: `https://github.com/felipegb94/ToFSim`.

References

[1] R. Ferriere, J. Cussey, and J. Dudley. Time-of-flight range detection using low-frequency intensity modulation of a cw laser diode: Application to fiber length measurement. *Optical Engineering - OPT ENG*, 47, 09 2008.

[2] R. Grootjans, W. van der Tempel, D. Van, C. De Tandt, and M. Kuijk. Improved modulation techniques for time-of-flight ranging cameras using pseudo random binary sequences. *Proc. IEEE LEOS Benelux Chapter*, 2006.

[3] M. Gupta, A. Velten, S. K. Nayar, and E. Breitbach. What are optimal coding functions for time-of-flight imaging? *ACM Trans. Graph.*, 37(2):13:1–13:18, Feb. 2018.

[4] F. Gutierrez-Barragan, S. A. Reza, A. Velten, and M. Gupta. Practical coding function design for time-of-flight imaging. *To appear in CVPR 2019*, 2019.

[5] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. *CoRR*, abs/1411.5752, 2014.

[6] A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight cameras in computer graphics. *Computer Graphics Forum*, 29(1):141–159, 2010.

[7] R. Lange. *3D Time-of-Flight Distance Measurement with Custom Solid-State Image Sensors in CMOS/CCD-Technology*. PhD thesis, University Siegen, 2000.

[8] P. Liu and E. Y. Lam. Image reconstruction using deep learning. *CoRR*, abs/1809.10410, 2018.

[9] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[10] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[11] A. Payne, A. A Dorrington, and M. Cree. Illumination waveform optimization for time-of-flight range imaging cameras. *Proceedings of SPIE - The International Society for Optical Engineering*, 8085, 06 2011.

[12] M. Ravanelli and Y. Bengio. Speaker recognition from raw waveform with sincnet. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*, pages 1021–1028, 2018.

[13] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[14] M. Seyedhosseini, M. Sajjadi, and T. Tasdizen. Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In *2013 IEEE International Conference on Computer Vision*, pages 2168–2175, Dec 2013.

[15] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1605.06211, 2016.