LEARNED RESIDUAL GERCHBERG-SAXTON NETWORK FOR COMPUTER GENERATED HOLOGRAPHY

Anonymous authors
Paper under double-blind review

ABSTRACT

Computer generated holography (CGH) aims to generate phase plates that create an intensity pattern at a certain distance behind the holography plate when illuminated. Since only the intensity and not the phase of the wave is of interest, this is an ill-defined inverse problem. Usually these problems are tackled by iterative optimization algorithms which are part of the convex optimization framework. These algorithms essentially minimize a loss using a forward model. Even though many of the tackled inverse problems are non-convex, these algorithms reach acceptable solutions by finding a local minimum. The ability of Deep Neural Networks to estimate a large range of functions has made a different approach to these problems possible. Instead of an iterative optimization algorithm that converges to a (sub-)optimal solution, the inverse problem can be solved by training a neural network to directly estimate the inverse operator. However simple convolutional neural networks tend to overfit when learning the inverse operator and do not generalize well outside the training distribution. Therefore this paper introduces a hybrid approach that can be interpreted as an unrolled Gerchberg-Saxton algorithm, which we term Learned Residual Gerchberg-Saxton (LRGS) network. We train this network for the generation of multi-focus computer generated holograms, and beat state-of-the-art existing methods.

1 INTRODUCTION

Computer generated holography is the practice of calculating the phase distribution for a phase mask (the hologram), to generate an amplitude distribution behind the phase plate via free space propagation. The phase changes can be created by a change of the wave speed inside the phase plate medium. Because the phase-front of a wave is subject to various constrints, it is generally not possible to perfectly create a hologram plate that leads to the desired intensity in the image plane. In fact the problem of hologram creation was found to be a non-convex ill-posed inverse problem (Bauschke et al., 2002) and is strongly related to the problem of phase retrieval in physics (Fienup, 1982).

If one allows imperfect target amplitudes it is possible to create a single hologram plate $f(x)$ that leads to multiple images at different distances from the hologram. This is displayed in figure 1. Because the forward propagation is deterministic, one needs to find a balance between image quality and number of target amplitudes. The calculation of this hologram plate is harder than the calculation of the hologram plate for a single image, since in this case the information for all images has to be encoded in a single hologram plate. Applications for holograms and multi-focus holograms can amongst others be found in optical tweezers (Vizsnyiczai et al., 2014), 3D-displays (Slinger et al., 2005) and ultrasound imaging (Hertzberg & Navon, 2011).

Usually the hologram plates are generated by iterative optimization algorithms that try to solve a minimization problem. Because of the non-convexity of the problem, these methods do not find a global minimum. Here we present a deep neural network that is able to generate multi-focus hologram plates and compare the results with state-of-the-art methods for the generation of these multi-focus holograms.

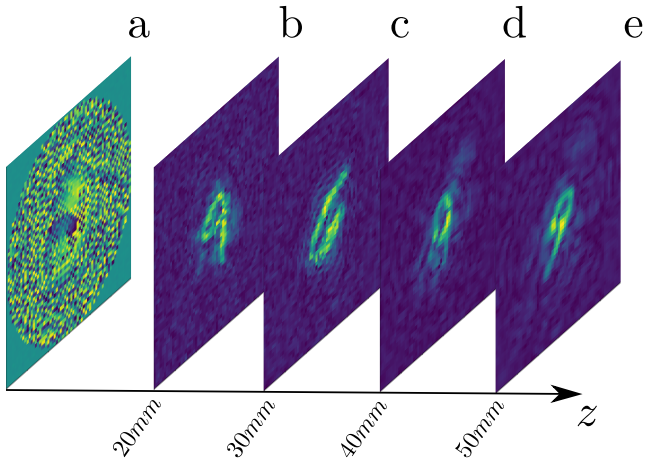


Figure 1: Schematic depiction of a hologram plate in front of a wave emitter (**a**) and four amplitude measurements (**b**, **c**, **d**, **e**). The hologram plate has been created to display four numbers out of the MNIST-dataset, each 10mm apart from each other.

2 THEORY

Let $f = f(x, y)$ denote the complex field in the wave emitter plane and $g_i(x, y)$ the complex field in the image plane at distance z_i from the emitter. The wave emitter plane is the plane where the hologram plate is put in front of a planar wave, also called hologram plane. Given a fixed amplitude of the wave emitter $u(x, y)$ and the desired amplitudes in the N image planes $g_{i,\text{target}}(x, y)$, $i = 1, ..., N$, the problem of computer generated holography is to find a complex field that satisfies the amplitude constraints in the hologram plane $|f| = u$ as well as the amplitude constraints in the image planes $|g_i| = g_{i,\text{target}}$ for $i = 1, ..., N$. The propagation from the source plane to the image planes can be described by a linear operator A,

$$g_i(x, y) = A_i f(x, y), \tag{1}$$

where A_i corresponds to the propagation of the field in the hologram plane f to the i-th image plane located at z_i. Directly enforcing the emitter amplitude constraint, we define the field in the emitter plane as $f(x, y) = u(x, y) \exp(i\phi(x, y))$, where ϕ denotes the phase to be encoded in the hologram plate. As there is generally no solution to this problem, one usually resorts to a relaxed version given as the unconstrained optimization problem

$$\min_{\phi} \sum_{i} \| |A_i(ue^{i\phi(x,y)})| - g_{i,\text{target}}(x, y) \|^2. \tag{2}$$

We implemented the propagation operator using the angular spectrum method Liu & Waag (1997). To calculate the propagated field at a distance z from a given $f(\vec{x}) = |f(\vec{x})| \, e^{i\Delta\phi(\vec{x})}$ with the phase $\Delta\phi(\vec{x})$, first the Fourier transform $\tilde{f}(\vec{k}) = \mathcal{F}f(\vec{x})$ is calculated. The propagated field is then calculated by multiplying with a forward propagator $K(\vec{k}, z)$ and reversing the Fourier transform as shown in equation 3.

$$A = \mathcal{F}^{-1} K(\vec{k}, z) \mathcal{F} \tag{3}$$

$$A^{-1} = \mathcal{F}^{-1} K(\vec{k}, -z) \mathcal{F} \tag{4}$$

The propagator $K(\vec{k}, z)$ is defined as $K(\vec{k}, z) = \exp^{ikz\sqrt{1-\lambda^2(k_x^2+k_y^2)}}$, where λ is the wavelength of the propagating wave and $\vec{k}$ is the wave vector, with $|\vec{k}| = \frac{2\pi}{\lambda}$. Ignoring evanescent waves, that arise when $(k_x^2 + k_y^2) > \frac{1}{\lambda^2}$, we can use $K(\vec{k}, -z)$ as the inverse operator of $K(\vec{k}, z)$.

Among the most popular methods for finding a solution for equation 2 is the traditional Gerchberg-Saxton algorithm (Gerchberg & Saxton, 1972). The algorithm finds an approximate solution of the feasibility problem by iterating the following steps:

1. Propagate the field in the wave emitter plane f to the image plane: $g = Af$
2. Substitute the amplitude of the field in the image plane g with the target amplitude g_{target}, while keeping the phase: $\tilde{g} = |g_{\text{target}}| \frac{g}{|g|}$
3. Propagate the field in the image plane $\tilde{g}$ back to the source plane $\tilde{f} = A^{-1}\tilde{g}$
4. Substitute the amplitude of the field in the source plane $\tilde{f}$ with the source amplitude u while keeping the phase: $f = |u| \frac{\tilde{f}}{|\tilde{f}|}$

The original Gerchberg-Saxton algorithm can be generalized to multiple focus planes. To do this in step 1 one propagates the source plane field into every image plane, set the amplitudes to the corresponding targets in step 2 and finally propagates all fields back to the source in step 3. Before setting the amplitude to the source amplitude in step 4, one sums over all backwards propagated contributions from the different image planes.

3 Previous Work

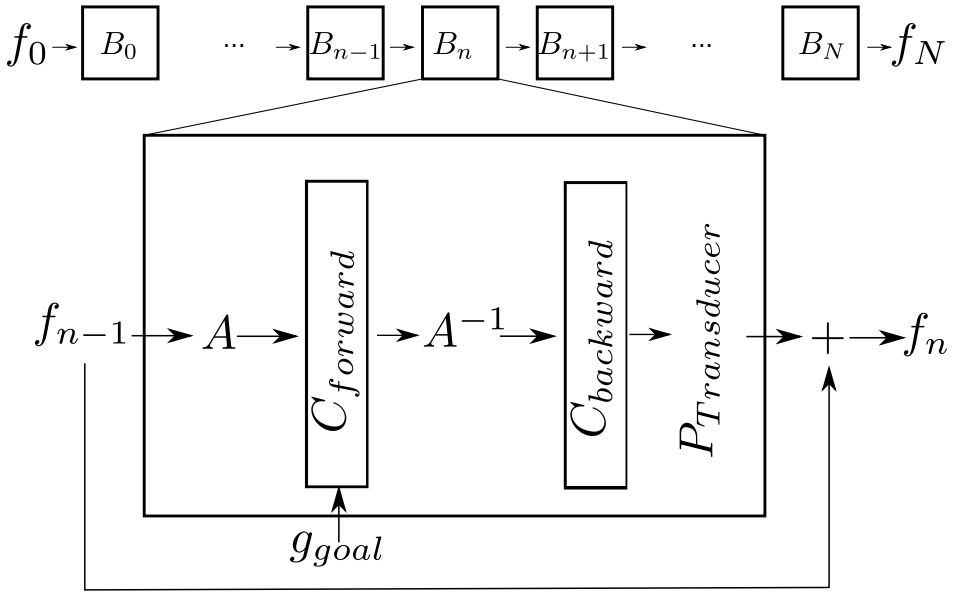
There have been some improvements on the GB algorithm since its establishment. It was shown, that the GB algorithm essentially is equivalent to a nonconvex variant of the projection onto convex sets (POCS) algorithm (Levi & Stark, 1984). Some of these projection methods work better for certain problems in phase retrieval. These methods use relaxed projections onto the set of images that are possible to create with a constraint in the hologram space and those images that fulfill the amplitude constraint in the image space. In the case of CGH the first constraint is the shape of the source and the second constraint is the target amplitude of the resulting waves.

Recently the problem of generatwthree-dimensional phase-only holograms for light waves has been tackled by directly minimizing a loss functional in the form of equation 2 Zhang et al. (2017). Instead of an alternating projections scheme,this functional is minimized with a quasi-Newton L-BFGS (Liu & Nocedal, 1989) solver. This so-called non-convex optimization for volumetric computer generated holography (NOVO-CGH) approach allows for the optimization of the hologram with respect to a customizable error metric defined by the corresponding loss functional. With the right choice of loss functional the method outperforms existing projection-based methods like the Gerchberg-Saxton algorithm.

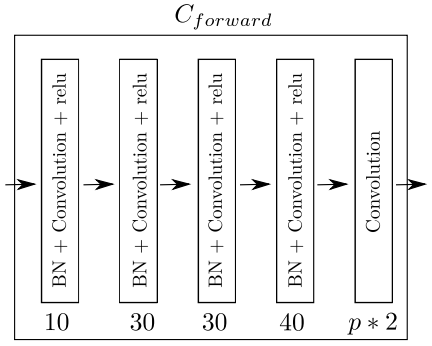
In this paper we use a neural network, that directly inverts the forward operator, by unrolling the Gerchberg-Saxton algorithm and adding learned layers. Neural networks have previously been explored for the creation of holograms with only one focus plane by Horisaki et al. (2018) with a network that does not use unrolling or knowledge about the forward operator but still manages to create results of reasonable quality. For inverse problems in general, several machine learning approaches have been proven to work, including methods similar to ours, that work by unrolling iterative algorithms and training parameters inside those algorithms, see Schuler et al. (2016), Adler & Öktem (2017), Hammernik et al. (2018), Adler & Öktem (2018). One of the main benefits of using neural networks to create hologram plates is the speed advantage one gains with a relatively shallow implementation once they are trained, when compared to iterative optimization schemes (Eybposh et al., 2020).

4 Methods

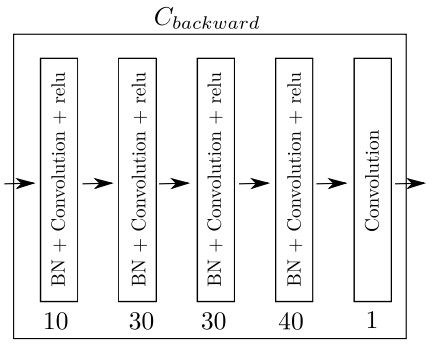
Our approach to solving the problem of multi-focused hologram creation is to train a neural network that is in structure similar to the Gerchberg-Saxton algorithm and other projection methods, but contains trainable convolutional layers inside of this iterative algorithm. The network is displayed in figure 2. The network consists of a sequence of blocks that each resemble one single Gerchberg-Saxton iteration, but with include learned convolutional layers. A block takes the result of the previous block, propagates it to the image space with the forward operator A and feeds the resulting images together with the target amplitude g_{goal} into a block consisting of convolutions. Since the physical wave can be described using a complex valued field, we represent the field using two channels, one corresponding to the real part and the other to the imaginary part of the field. The convolutional block $C_{forward}$ consists of 5 convolutions with rectified linear unit (relu) activation



(a) Network Structure



(b) Forward convolutional block



(c) Backward convolutional block

Figure 2: a) Network structure for the unrolled residual Gerchberg-Saxton network. The network starts with an arbitrary phase plate f_0 (usually zeros). This initial hologram is put through N blocks that are in structure similar to a Gerchberg-Saxton iteration. Each block propagates the field f_{n-1} to the image space with the forward operator A. The propagated wave fields is concatenated with the goal amplitudes g_{goal} and fed to a block consisting of trainable convolutions $C_{forward}$. The result is propagated to the hologram plane with the inverse operator A^{-1} and again fed to a block consisting of convolutions. Finally the result is projected onto the hologram constraint and summed with the result of the last layer. b) Structure of the block applied to the image space input. The block consists of 5 convolutions with 10, 30, 30 and 40 kernels. The last convolution has $2 * p$ output layers, where p is the number of different focus planes. The output of the convolution is double the number of focus planes, since we need a complex field for each focus plane. Each convolution is preceded by a batch normalization layer and followed by a relu activation function. The last layer does not use the relu activation function, since the output is a complex number which must be allowed to be negative c) The structure of the backward convolutional block is almost identically to that of the forward block. The only difference is, that the block has a single output. This output is the phase of the hologram plate.

functions and batch normalization layers in front of every block. The last block does not have a relu activation function, since the output of the block must be allowed to be negative. The output is converted to p complex images, where p is the number of focus planes. These complex images are propagated into hologram space again with the inverse operator A^{-1} and fed to another block

of convolutions, $C_{backward}$. This block is similar in structure to the first block, but the last layer has only a single kernel, since the output of this layer is used as a phase for the hologram plate. This phase is fitted to the source constraint by multiplying it with the amplitude of the source. The old hologram f_{n-1} is added to this new output to form the new hologram f_n which is fed to the next block. This is repeated N times. This structure resembles the Gerchberg-Saxton algorithm and other projection algorithms for phase retrieval, apart from the fact, that the network contains learned parameters and the algorithm stops exactly at N iterations of the iterative algorithm.

4.1 TRAINING

The network was trained with images from the MNIST dataset (LeCun et al., 1998) containing hand written digits, randomly shuffled and stacked in blocks of k images. For evaluation the fashion MNIST dataset (Xiao et al., 2017), containing various clothing items, was also used, but no network was trained on these images. For all experiments we consider acoustic waves propagating in water, which have recently become a promising application of CGH Melde et al. (2016). We assume a hologram and image size of 50mm $\times$ 50mm, with 60×60 trainable pixels in the hologram. The frequency of the wave was chosen to be $f = 1$Mhz with a speed of sound of $c_w = 1.484\frac{m}{s}$. The constraint in the source plane was set to a circular amplitude with a radius of 25mm. These values are consistent for ultrasonic waves generated by a circular transducer and can be recreated in the lab. We note however, that this method can also be applied to standard holograms generated with light waves by choosing different values for the wave speed and frequency.

4.2 LOSSES

Since the loss function heavily determines the results of the network multiple loss functions were tried. The first loss function was the standard mean squared error between the resulting amplitudes $|Af_N|$ and the target amplitudes g_{goal}.

$$L_{mse} = \frac{1}{N}\sum_{i}^{N}(|Af_N|_i - g_{goal,i})^2 \tag{5}$$

However, this creates the predicament that the network is forced to generate images of a certain amplitude, while not every amplitude is always realizable due to the deterministic wave propagation and the total energy input of the wave emitter. For a better measure of the hologram image quality the mean squared error was normalized by the maximum, seen in equation 6, and the relaxed standard deviation of the images, seen in equation 7.

$$L_{nmse} = \frac{1}{N}\sum_{i}^{N}\left(\frac{|Af_N|_i}{max(|Af_N|)} - \frac{g_{goal,i}}{max(g_{goal})}\right)^2 \tag{6}$$

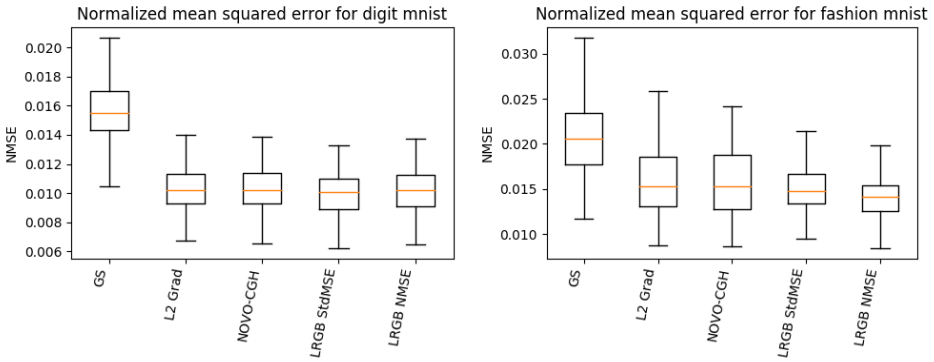
$$L_{stdmse} = \frac{1}{N}\sum_{i}^{N}\left(\frac{|Af_N|_i}{std(|Af_N|)} - \frac{g_{goal,i}}{std(g_{goal})}\right)^2 \tag{7}$$

These two losses where used to train two LRGS networks in an almost self supervised learning setup. The training works by giving the network target amplitudes, which in turn creates a phase plate, from which one of the above loss functions is calculated. This works, because the complete network, including the forward and backward wave propagation operator A and A^{-1}, is backpropagatable.

5 RESULTS

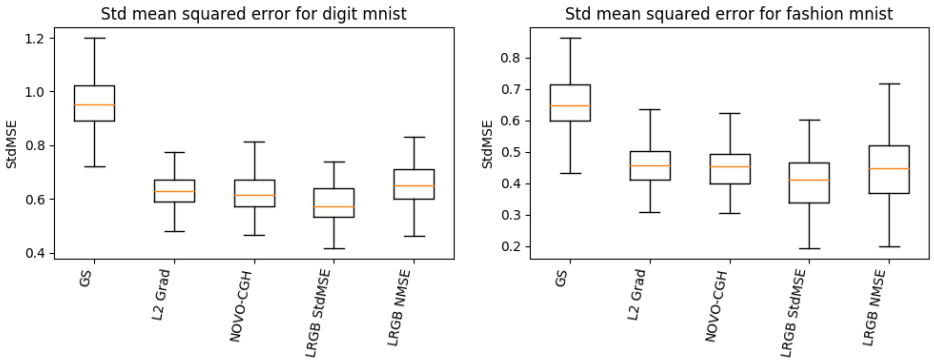
We compared the learned residual Gerchberg-Saxton network with the standard Gerchberg-Saxton algorithm for 3D hologram generation, the NOVO-CGH algorithm and a simple first order gradient descent minimization of the $l2$-loss functional, which we call L2-Grad in the following section. The iterative optimization schemes where run for 1000 iterations. The two LRGS networks each had a depth of $N = 10$ and where trained to generate four output amplitude images from the digit mnist dataset in a distance of 20, 30, 40 and 50 mm behind the hologram plate. The networks only trained

on this dataset, but manage to generalize to other datasets. The first network was trained using the normalized mean squared error loss function L_{nmse} given by equation 6 and the second network was trained using the mean squared error normalized by the standard deviation L_{stdmse}, given by equation 7. Both networks where trained using the Adam optimizer (Kingma & Ba, 2014).
To compare the image quality the structural similarity index measure (ssim) was also calculated and compared. The LRGS networks were able to outperform both the standard Gerchberg-Saxton algorithm as well as the NOVO-CGH and the L2-Grad algorithm. The results of 100 test images from the mnist digit dataset can be seen in figure 3 a) b) and c).



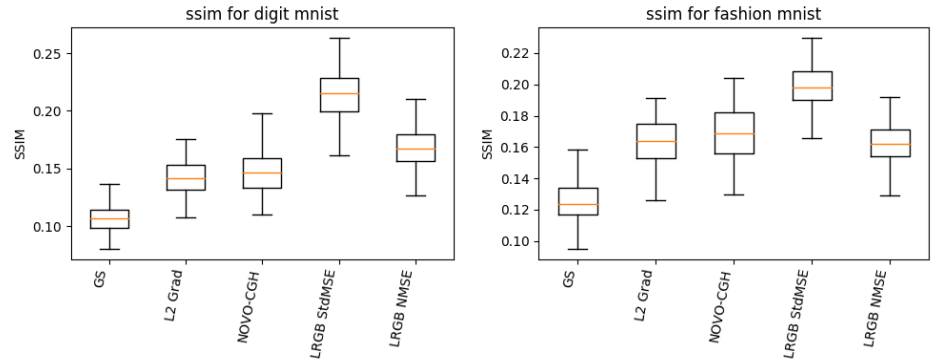
(a) L_{nmse} for mnist digits

(b) L_{nmse} for fashion images



(c) L_{stdmse} for mnist digits

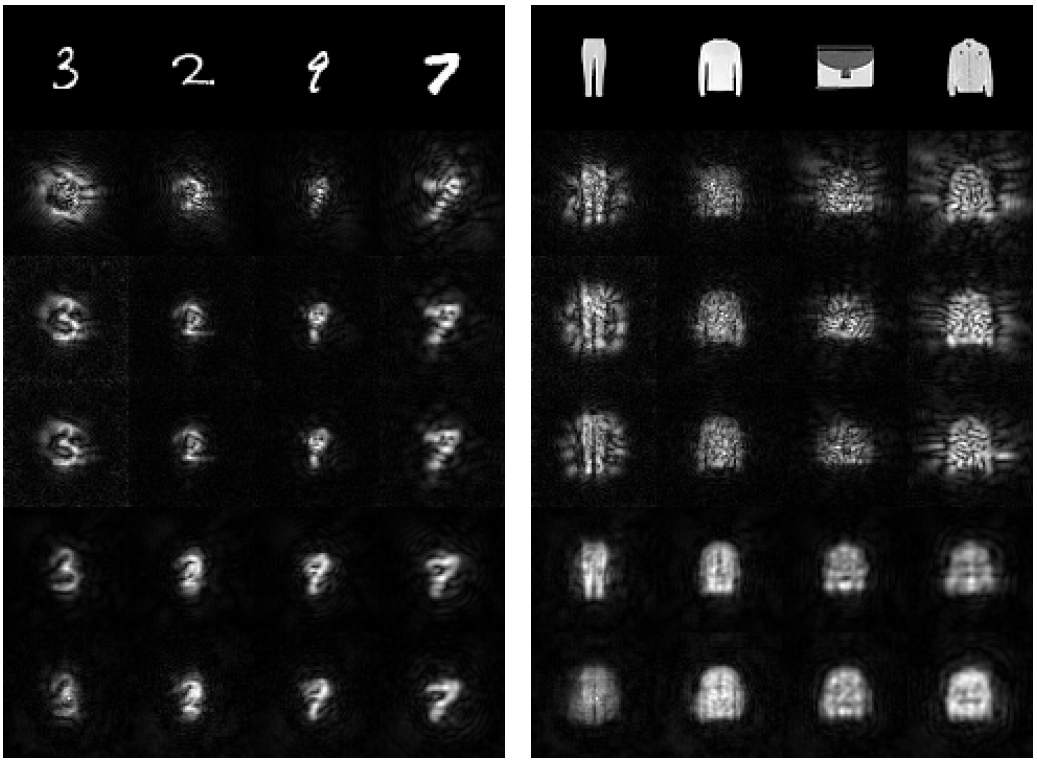
(d) L_{stdmse} for fashion images



(e) ssim for mnist digits

(f) ssim for fashion images

Figure 3: Different losses for 100 test images from the MNIST digit dataset and the fashion MNIST dataset, on which the network was not trained before. The top row contains the losses for the MNIST digit dataset and the bottom row the losses for the fashion MNIST images on which the network was not trained. For the L_{nmse} and the L_{stdmse} smaller values are better, while for the ssim higher values are better.



(a) Mnist digits (b) Fashion mnist images

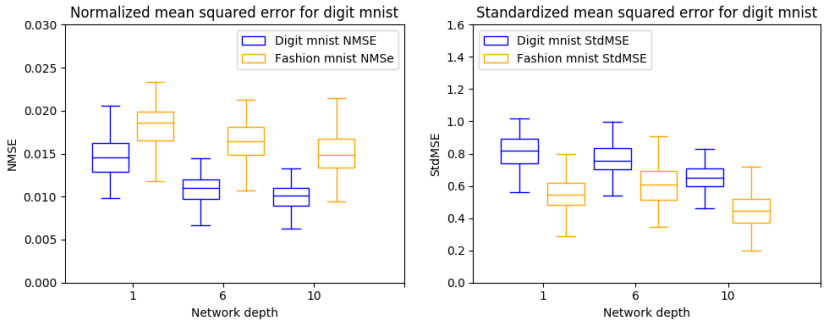
Figure 4: One example output from the five different algorithms for two test datasets. From top to bottom the rows contain: 1) The goal amplitude, 2) Images generated by the Gerchberg-Saxton algorithm, 3) Images generated by the L2-Grad algorithm, 4) Images generated by the NOVO-CGH algorithm, 5) Images generated by the L_{stdmse} LRGS network and 6) Images generated by the L_{nmse} LRGS network

Example amplitude images are displayed in figure 4a. The images are all distorted, illustrating the hard nature of the problem. In fact the NOVO-CGH algorithm failed to converge 4 times out of 100 test images. The outliers are not shown in the plots.

Even though the networks were only trained on MNIST digits they are able to generalize to images that come from a different input distribution. We tested the networks with images from the fashion MNIST dataset, that have the same dimensions as the digit MNIST images but show silhouettes of everyday clothing items. The resulting losses can be seen in figure 3 d), e) and f). The networks manage to outperform the other algorithms for the fashion MNIST images. Example outputs can be seen in figure 4b. It should be noted, that the images generated with the network had ten times higher maximum amplitudes than those generated with the NOVO-CGH algorithm and the L2-Grad algorithm, even when trained on standardized loss functions. It seems the gradient descent like algorithms generate hologram plates, that disperse a lot of energy, while Gerchberg-Saxton like methods focus more.

5.1 Dependence on network depth

To check, if the network depth makes a difference we trained 4 more networks with the same loss functions as the big networks above. Those networks had 1, 6 and 10 blocks. It can be seen in figure 5, that the loss decreases with increasing network depth. In general more learned Gerchberg-Saxton layers should only increase the performance of the network, because a bigger network can always generate the same results as a smaller network by relying on residual connections.



(a) Normalized mean squared error L_{nmse} for different network depths.

(b) Standard deviation mean squared error L_{stdmse} for different network depths.

Figure 5: Boxplots for the two different losses used in network training. Panel a) displays the normalized mean squared error L_{nmse} for the digit MNIST dataset and the fashion MNIST dataset for network depths of 1, 6 and10. Panel b) visualizes the standardized mean squared error L_{stdmse} for the same network depths. The networks where each trained on the specific loss that is given in the plots. It can be seen, that the loss decreases for higher network depths.

5.2 Execution Speed

One of the main advantages of neural networks is the time it takes for them to generate a hologram plate once they have been trained. While iterative algorithms need to iterate until they converge, a learned network only needs one pass to generate a hologram plate. To illustrate this major difference we measured the time the algorithms took to generate images. The results can be seen in table 1. All results where obtained on the same machine containing a Geforce GTX 1080 graphics card and a Intel(R) Core(TM) i7-5820K CPU with a main clock frequency of 3.3 Ghz. The iterative algorithms were run for a maximum of 500 iterations.

Table 1: Execution times for the different hologram creation algorithms for the creation of a single hologram plate.

LRGS-1	LRGS-6	LRGS-10	GS	NOVO-CGH	L2-Grad
0.028s	0.114s	0.181s	5.159s	32.067s	8.047s

With a faster implementation of the iterative schemes it might be possible to reduce the runtime of these algorithms, however the networks structure has the benefit of having a fixed execution time for every plate generated. Since the structure of the LRGS network is similar to the iterative GS algorithm, it should outperform the other iterative schemes almost all the time, unless they converge in less than N iterations.

6 Conclusion

We introduced a network structure that can be viewed as an unrolled Gerchberg-Saxton algorithm, which we termed the Learned Residual Gerchberg-Saxton (LRGS) network. This network was able to outperform state-of-the-art hologram generation methods in visual quality and speed. Furthermore, we demonstrated that the network is capable of producing images outside of the training distribution. We hypothesize that this is due to its structure and the implementation of the propagation operator A into the network structure, thus implementing physical prior knowledge into the architecture. We argue that this approach is superior to iterative optimization algorithms in computer generated holography. The LRGS network, once learned, also generates new holograms more quickly than iterative methods.

REFERENCES

J. Adler and O. Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, 2017.

J. Adler and O. Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

H. H. Bauschke, P. L. Combettes, and R. D. Luke. Phase retrieval, error reduction algorithm, and fienup variants: a view from convex optimization. *JOSA A*, 19(7):1334–1345, 2002.

M. H. Eybposh, N. W. Caira, M. Atisa, P. Chakravarthula, and N. C. Pégard. Deepcgh: 3d computer-generated holography using deep learning. *Opt. Express*, 28(18):26636–26650, Aug 2020.

J. R. Fienup. Phase retrieval algorithms: a comparison. *Appl. Opt.*, 21(15):2758–2769, Aug 1982.

R. W. Gerchberg and W. O. Saxton. A practical algorithm for the determination of phase from image by diffraction plane pictures. *Optik*, 35:227–246, 1972.

K. Hammernik, T. Klatzer, E. Kobler, M. P. Recht, D. K. Sodickson, T. Pock, and F. Knoll. Learning a variational network for reconstruction of accelerated mri data. *Magnetic resonance in medicine*, 79(6):3055–3071, 2018.

Y. Hertzberg and G. Navon. Bypassing absorbing objects in focused ultrasound using computer generated holographic technique. *Medical physics*, 38(12):6407–6415, 2011.

R. Horisaki, R. Takagi, and J. Tanida. Deep-learning-generated holography. *Applied optics*, 57(14): 3859–3863, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

A. Levi and H. Stark. Image restoration by the method of generalized projections with application to restoration from magnitude. *J. Opt. Soc. Am. A*, 1(9):932–943, Sep 1984.

D. Liu and R. C. Waag. Propagation and backpropagation for ultrasonic wavefront design. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 44(1):1–13, 1997.

D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

K. Melde, A. G. Mark, T. Qiu, and P. Fischer. Holograms for acoustics. *Nature*, 537(7621):518–522, 2016.

C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf. Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1439–1451, 2016.

C. Slinger, C. Cameron, and M. Stanley. Computer-generated holography as a generic display technology. *Computer*, 38(8):46–53, 2005.

G. Vizsnyiczai, L. Kelemen, and P. Ormos. Holographic multi-focus 3d two-photon polymerization with real-time calculated holograms. *Opt. Express*, 22(20):24217–24223, Oct 2014.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

J. Zhang, N. Pégard, J. Zhong, H. Adesnik, and L. Waller. 3d computer-generated holography by non-convex optimization. *Optica*, 4(10):1306–1313, Oct 2017.