Deep Learning of Image Inpainting for Semiconductor Wafer Image Recognition

Michelle Meng Shi
Engineering Optics, Office of CTO, Applied Materials
Stanford Center for Professional Development
mshi101@stanford.edu

Abstract

As AR / VR field advances and moving towards achieving 'Metaverse' in mixed reality world, creating the next generation devices – AR glasses, is identified as key for people to access the new world. From semiconductor manufacturing to nano photonics manufacturing which enables the 'display' - the optical waveguide being designed and developed on transparent substrates. We are now facing new challenges in transparent wafer auto handling, which requires high throughput, high precision with high stability. Inspired by the recent advancement and blooming in machine learning and AI field, in which lots of new algorithms have been created and put in application and gaining wide popularities quickly, such as context encoder [1], Google magic eraser [2] etc. In this project, I am looking at applying deep learning algorithms of image inpainting to automated wafer handling and image recognition, especially for fiducial marks on patterned wafer and to find a way for solving the new challenges in developing new products. Since all confidential information is under NDA with partners, so no real data can be shared for training. I am exploring a Sim2Real solution as a first step and to better fit into future development plan. The encouraging results in the project will serve as proof of concept of a potential AI feature in our product development roadmap and lead us towards a new 'Metaverse' future.

1. Introduction

Wafer handling is a high value problem which involves a lot of advanced technics to achieve high volume, high precision with high stability. As an application engineer working at Applied Materials CTO office for AR / VR related projects, I am constantly facing new challenges in wafer process development. I often encounter questions on how to improve the efficiency of wafer handling, which includes image recognition for wafer automatic positioning. It is an interesting and high value question to improve the image recognition algorithm so all the wafers can be properly handled with precise automated positioning for process development such as patterning or inspection. The less manual handling it gets, the less chance of defects or process inconsistency will get to the wafer surface. Figure 1 shows concept Metaverse AR glasses, in which the eye glasses piece can be manufactured on a transparent wafer. Figure 2 shows demo optical waveguide wafer which includes a lot of singular eye pieces for batch processing.

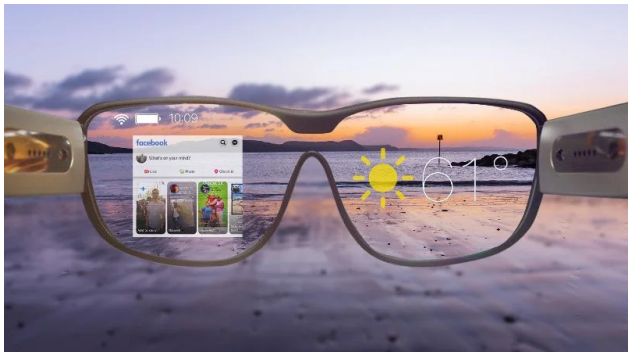


Fig 1. Metaverse AR glasses concept
Image credit: https://artlabs.ai/blog/the-best-smart-glasses-and-ar-specs-of-2021/

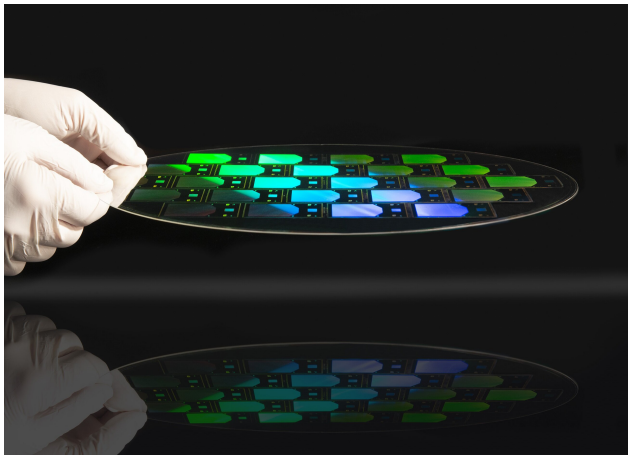


Fig 2. Next generation optical waveguide demo wafer.
Image credit: EVG public

A lot of automation tools are used in manufacturing processes for the AR glasses, including deposition, etching, lithography, defect inspection, optical metrology tools etc. All these tools have different image recognition algorithms. The tools can vary from different generations, which some of the software can seem to a black box. Most of the image recognition algorithms on the tools usually involves comparing a ground truth image to the camera acquired image, which often time is a fiducial mark, such as a cross pattern to the image captured by camera in the positioning search process. Due to the less maturity of the process, a lot of times, the fiducial mark has some type of defects, such as part of the mark can be missing. With such defects, image recognition algorithm will often fail due to the big differences between the ground truth image and the real wafer surface fiducial mark image, since the image recognition algorithm is simply comparing the two images see if they match within certain tolerance.

In this project, I propose to use deep learning of image inpainting method to solve the fiducial mark failure issue, and improve the image recognition algorithm success rate, to make it 'smarter' in recognizing the fiducial mark, which will increase the efficiency of process automation with less failure and less manual operational time.

Since the tools can all be different in some sense. Finding a 'universal' solution which can quickly resolve the fiducial mark problem is essential. A potential solution can be, adding a deep learning layer in camera ISP to generate fiducial images which can be clearly recognized by the image recognition algorithm. This 'middle layer' can then be applied for all the process tools without changing the existing image recognition and alignment algorithm, see figure 3. Input and output will be images, while input images are acquired from camera, the image than go through a convolutional neural network to output a predicted image which can be easily recognized by the existing image recognition and alignment software on the process tools.

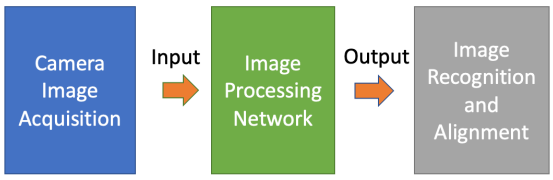


Fig.3 Illustration of camera image acquisition process, input image to CNN layer and output predicted image to existing image recognition algorithm.

2. Related Work

Image inpainting is a very popular topic in computer vision these days. Deepak's paper on context encoders has presented an unsupervised visual feature learning algorithm driven by context-based pixel prediction. In the paper, a convolutional neural network is trained to reproduce the missing image content according to the context of image. The features in the remaining images were learned and later be adapted into the generation of the content of the missing pixels. Just like human prediction based on the image context, such as the periodic pattern, different background scene, environments where certain things would have high possibilities to appear. The algorithm is advanced in incorporating such ideas into the design using CNN. The results were evaluated by comparing the generated inpainted image to the ground truth. The results turned out to be very impressive and promising which also encourages me to look more into this field.

Another interesting example comes from Google's new algorithm, the magic eraser in Google photos. Google pixel 6 phone has a new feature to remove the unwanted features in photos and inpainting with proper computer-generated content. This method is using convolutional neural network to train on a big dataset with simulated occlusion images compare to the ground truth images. A lot of times, people would like the pictures to be as 'clean' as possible, such as removing unwanted power lines, random people in the background etc. The app would predict the pixels which need to be filled into the area where certain unwanted features have been taken out. And the results showed very 'natural' content, where people who is viewing the processed image will not notice that part of the pictures were computer generated and may not reflect the real scene where the pictures were taken.

I also looked at some literatures related to semiconductor process technologies. Chen et al.'s paper described automatic wafer alignment algorithms for wafer bonding system. In semiconductor process, wafer bounding needs to have high accuracy and the throughput needs to be higher for production purposes. The alignment is usually done using alignment marks, which are the crisscross marks. The technology involves image recognition and processing technology, robotics movement and positioning of two wafers, so it will stack precisely on each other. Any positional error will cause problem in the process. Active feedback and calibration is the key to achieve. While this paper described a more traditional way of doing the wafer alignment. I can see with the deep learning technology in computer vision system, the process can be improved even more with machine vision and training. A lot can be done with the new technology in the wafer auto processing field,

since a lot of automation process are highly dependent on the image recognition and robotics positioning. [3]

Some other paper I am looking at includes fiducial mark design using neural networks. Over the years, there are different fiducial marks designed for machine vision and a good code / pattern is key. The markers illustrated in the paper are for use in augmented reality applications. It can be used for obtaining the position, rotation, relative size, and projective transformation. This is some application beyond semiconductor process, but extended to the AR / VR world, which I find is very interesting too. [4]

3. Methods

Image inpainting can be very useful technic to solve part of the fiducial mark missing problem. We have struggled from the failures due to partly missing fiducial marks which caused the entire image recognition process to fail, and system throws an error for alignment. Manual alignment usually induced more errors and cannot be reliably regarding as a feasible solution. After some research on deep learning and computer vision. I am thinking about using context encoder based on Deepak's paper to see if the algorithm can help to fill in the gaps between a partially missing image to a clear fiducial mark which can be recognized by the image recognition algorithm.

3.1 Context Encoder and Decoder

A context encoder and decoder pipeline are implemented in the model. The context encoder uses convolutional neural network to learn the surrounding feature of an image, in our dataset, simulated fiducial mark images will be pass through the encoder to study the feature based on the remaining part of the image. The encoder will than produce the feature representation of the image. A channel wise fully connected layer is in the middle between the context encoder and decoder. The context decoder studies the feature which passed through the network and making predictions of the missing image based on the features. Loss function used in the process will compare the predicted image and the ground truth missing image part and feed back to the training process.

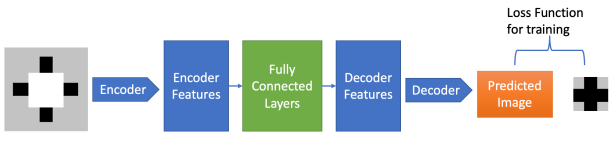


Fig 4. Context encoder network structure. Using simulated fiducial mark image to train the network with encoder, fully connected layers and decoder to generate predicted image which fills in the gap of missing image in center.

The encoder described in this project is derived from AlexNet architecture [5]. The input image size in the traning is 3 x 128 x 128. In the first encoder layer, 64 features were applied. Encoder architecture includes 5 layers of Conv2d, BatchNorm2d and LeakyReLu to generate features.

To properly pass information from the encoder to decoder, a channel-wise fully-connected layer is implemented in the architecture. It will only pass information within feature maps. Here a bottle neck of 4000 is used to restrict dimension of the bottleneck layer. This will be helpful in speeding up training time without over challenging the computing power.

The decoder part is constructed with five up convolutional layers to generate the missing piece of image. The up convolutional layer involves ConvTranspose2d layer, which can be seen as the gradient of Conv 2d layer with respect to its input, also known as fractionally-strided convolution. [6] Followed by BatchNorm2d and ReLu layers until the fully sized missing image is generated.

3.2 Loss Function

The context encoder was trained by the loss function comparing the ground truth which is the missing content in the image and the predict filling part of the image. A general question facing image inpainting is that there are many ways to fill in missing pixels in an image. So how to train the model with proper loss function and methods are essential to a successful development of network architecture.

The loss function contains two parts which are decoupled, and uses L2 loss for calculation.

Reconstruction Loss Function:

$$Loss\ rec = \| \hat{M} \cdot \left(x - F\left(\left(1 - \hat{M} \right) \cdot x \right) \right) \|_2^2$$

M is the mask where pixel values are 1 indicate for the dropped region and 0 for the remaining input pixels. When applying the mask, image can be generated during training.

Adversarial Loss Function:

$$Loss\ adv = \max D\ Ex \in x\ [\log\left(D(x) \right) + \log(\ 1 - D\ \left(F\ \left(\left(1 - \hat{M} \right) \cdot x \right) \right))]$$

The adversarial loss in the model is based on Generative Adversarial Networks (GAN).[7] Generative model G and adversarial discriminative model D are being used in the training. The learning procedure is a two-player game where model D tries to distinguish between the ground truth and the prediction of the generator G. G is trying to produce 'fake' images which is trained to be as 'real' as possible to confuse D. In the equation above, F is regarded as G. So we are looking at the logistic possibility that if an input is a real one or a predicted one.

Joint Loss Function:

$$Loss\ joint = \lambda_{rec} L_{rec} + \lambda_{adv} L_{adv}$$

The overall loss function is defined as a combination of reconstruction loss and adversarial loss, where two losses are decoupled from each other.

3.3 Mask Region

The mask being used in the project is a center square region for simplicity in training and analysis purposes. The regions where the pixel values are set to zero is regarded as 'drop out' regions. The mask has a certain size, for the simulated dataset, some feature might be blocked, and the network may not be able to learn from any 'hint' or features from the surroundings, which in turn can generate higher loss values. More discussions will be in the following sessions.

4. Dataset and Features

Since all the real wafer and fiducial data are under NDA with customers, no access can be granted for external sharing. We are facing a Sim2Real problem, which is how to generate simulated dataset which represents what is happening in real life. [8] With as real as possible datasets, one can expect a lot of problems in the algorithm design and model training can be found early on, and once the model deploys, it will be able to generate plausible results in real environment without too much model tuning. Figure 5 shows potential locations of fiducial marks on wafer die, which normally being designed to be found in four die corners for alignment purposes.

For the image dataset, I used a set of simulated fiducial mark images and consider multiple factors to make the fiducial marks look as 'real' as possible to the ones on wafer surfaces. First, I created a set of fiducial mark images with some size and rotational transform to simulate the fiducial positions when detector camera sees it. Secondly, consider different light conditions and the wafer surface is not a binary black and white surface. I chose a grey background to simulate the background color the camera may see the fiducial marks. Fiducial marks are usually crisscross pattern, the reasoning has two parts, first, a cross pattern compare to a round circle is good for finding the rotational degree differences for wafer alignment, since the wafer may be misaligned by some small degrees, but not a large one. Since wafer will usually go through pre-aligner, which aligns the wafer notch with a small degree of errors. So the simulated dataset is chose to have 20 degrees offset to simulate a larger than normal angular difference. Figure 6 shows part of the fiducial mark patterns I generated for this project. There are a total of 120 fiducial marks being generated, 90 are used in training dataset and 30 are used for testing. Another test dataset of 30 images with 'scratch' background to simulate the real wafer environment is also generated for testing purposes to see how the model performs.

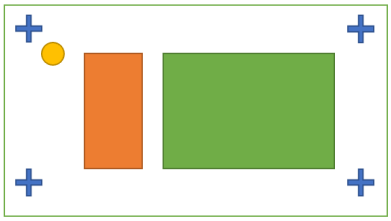


Fig 5. Wafer die illustration with potential fiducial marks positions for automatic positioning.

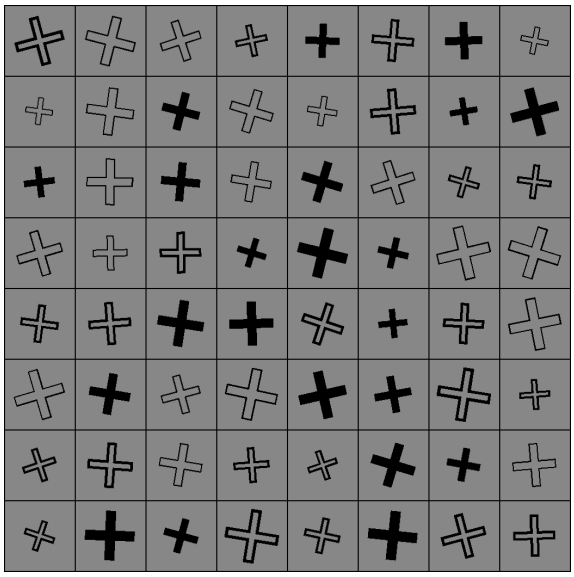


Fig 6. Simulated crisscross fiducial mark ground truth.

5. Experiments and Results Discussions

The code I start with is from Deepak et al.'s encoder paper, where the image inpainting model is well trained and has been put in use in a lot of applications. The codes can be found in *https://github.com/pathak22/context-encoder* webpage. Since the model is trained on natural image set, for the application in semiconductor process, the training image is changed to the simulated fiducial marks dataset. Based on the simulated to 'real' fiducial mark dataset, expect some reasonable predictions to be given by the inpainting neural network.

The training process was done with 200 epochs which means all the training data have been seen 200 times in the process. The model is then generated for testing and further investigation. The hyperparameters in the model is selected as learning rate of 0.0002 to keep the model from learning fast without overshooting and finding a balance. Due to the smaller sample sizes, the batch size is chosen to be 10. Stochastic gradient decent was used in the training. The optimizer used in training process is ADAM optimizer. [9] The training time was about 1hr depends on the hardware configurations. Compare to other more complicated networks and more difficult tasks to do. Training such a small but deep neural network is a fast and efficient way in solving relatively easier prediction problems. Which means, with proper implementation of the neural network design and proper datasets and clear targets, once can achieve desirable results in a short time frame for faster product and feature launch.

To evaluate the results, once can compare the success rate of image recognition on fiducial marks before and after implementing the image inpainting algorithm as a pre-process step. With the cropped images, there will be 0% of the chance for any image recognition algorithm to pass the test. Since the image recognition algorithms are looking for 'ground truth', which is significantly different compared to the cropped image. So the baseline in our specific application is actually zero. Any improvement by implementing image inpainting CNN to the process and pass the image recognition algorithm regarding to 0% baseline is a win. I would see after the training results, at least 50% of the image can pass the image recognition algorithm test for a higher success rate.

Here I used two sets of data to test the model, and to compare the output inpainted images qualitatively and quantitatively to the ground truth. The first test dataset is very similar to the training dataset with 'clean' background. We can see the PSNR is about 15.2995, which is acceptable, but the higher the better. And L2 loss is 0.1651, which is not too high after the training. An interesting phenomenon is that for the relatively smaller fiducial masks, which the center square can cover almost all the features and leave only the background image. Also looking at inpainted images, some has a not so clean inside color, which is a mixed pixel region of black and white. That can be explained as the network sees solid crisscross also some hollow ones, so the prediction will be a mixture internal region. Most of the images were predicted successfully by the network, which is very encouraging.

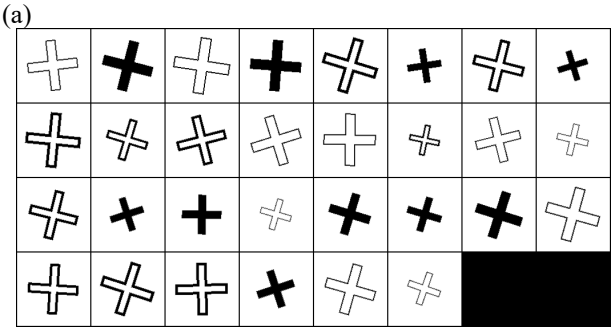
Thinking about the real process conditions, sometimes, the background is not 'clean' but with scratches. A second set of test dataset was generated for the purpose of simulating the real environment of the fiducial marks. The reason to use the existing model is to see how well the model will adapt to the real environment. It was interesting to find out, most fiducial mark patterns were 'recovered' during the process, which suggests this network can be used in some type of denoising process. Where we have some known pattern, we want to 'clean up', and by training the model with clean data, even in a 'noisy' background, the clean data can be 'recovered' using image inpainting.

Also, surprisingly, the L2 loss and PSNR value of the second dataset with noisy scratch background is higher than the clean background dataset. One hypothesis is that the results can have some dependency on random sample selection. A dataset with less 'small' fiducial marks tends to test better. Which is expected to be true, since the context encoder is learning the feature in the non-blocked region and making guesses, so if more information is provided, a higher success rate is expected. Another hypothesis is that the scratch background is 'helping' the neural network in the guessing process. Since similar type of dataset has never been seen during the training process. It is not sure how the background is contributing in the process, other than providing more content feature to learn. Once can also design some experiments to verify the hypothesis, such as adding random noisy background in the image processing pipeline, and the loss function will be calculated regarding ground truth. So such algorithm can potentially do two things at the same time. (1) image denoising of certain undesired background features. (2) image inpainting in the missing area.

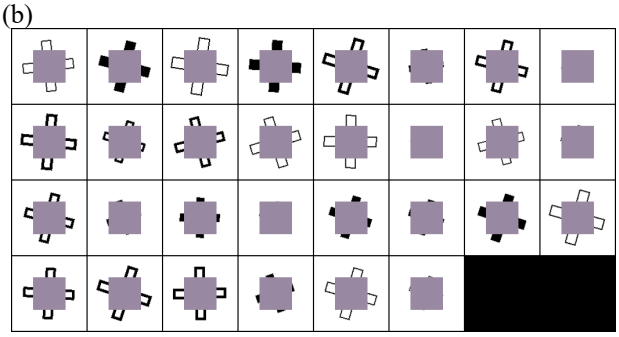
Dataset	L2 Loss	PSNR
Test Dataset with clean background (30 images)	0.1651	15.2995
Test Dataset with 'scratch pattern' background (30 images)	0.0828	20.0537

Table 1. Test datasets L2 losses PSNRs.

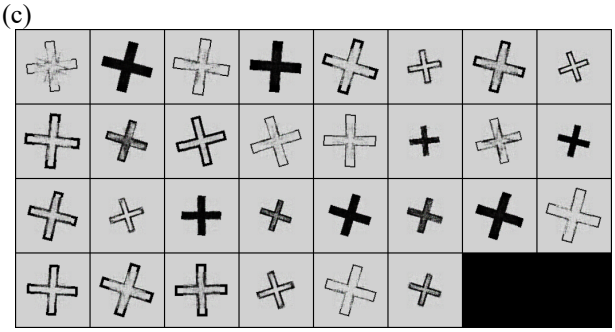
(a)



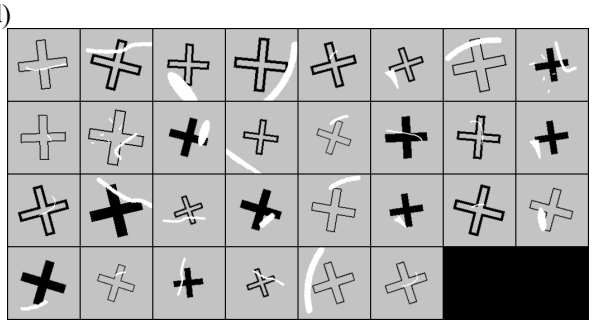
(b)



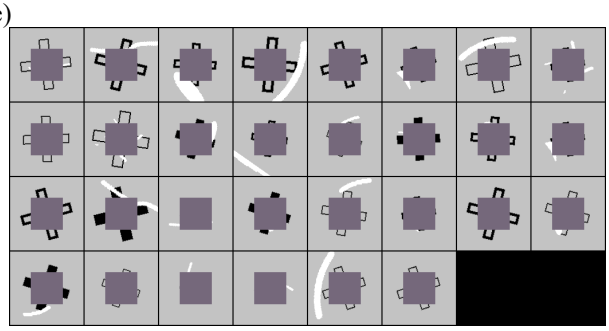
(c)



(d)



(e)



(f)

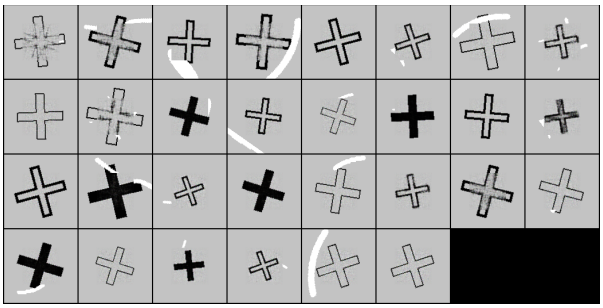


Fig 7. (a) Test fiducial mark dataset with clean background. (b) Cropped center images for testing with clean background. (c) Network generated inpainting fiducial mark image. (d) Simulated 'real life' fiducial mark with scratch in the background. (e) Cropped center test image with scratch in background. (f) Network generated fiducial marks with scratch in background.

6. Conclusion and Future Work

In this class project, I looked at using deep learning algorithms for the application in semiconductor wafer handling and image recognition by adapting a 'universal' intermediate layer for image inpainting and processing. The results were very encouraging, to see the potential success rate of image recognition algorithm to increase from 0% to over 50% with implementing the neural network algorithm. This will serve as proof of concept and first step in the product development for a more robust wafer automation system to help with process development with higher 'tolerance' in imperfections on the wafer surfaces.

For the future work, I can see there are multiple projects can be done in stages and be applied to AR / VR manufacturing:

(1) Using the real wafer and fiducial mark data for training and testing. Which I expect will be more difficult and prone to more problems. But I believe this is something we can do given the fact that fiducial marks, if we would like to train regarding certain wafer design will not have a huge variation. So the concern will be more to the background compare to the pattern itself.

(2) Creating a camera ISP with denoising and image inpainting algorithm for dealing with real environment. Since there can be multiple scratches, defects, dusts and other imperfections on wafer surfaces, so adding denoising step is necessary in real product development. Also from some of the test results, we can see that image inpainting algorithm can potentially 'denoise' in certain cases. One can try on different training dataset design to achieve the functionalities desired.

7. Acknowledgements

I would like to thank cs231n course team for putting everything together in these very interesting lectures. The class is so big with attendants all over the world, it has been a very challenging task to help and guide everyone, fulfill all the needs. I would also like to thank my TA Haochen Shi for using his weekend time to guide me through the project, and really appreciate the discussions and suggestions. Wish Haochen a bright future in robotics field!

Before attending this course, my expectation is to get more understandings on computer image processing related to my current work in AR / VR optical engineering field, also to understand if there is any way to improve our process, efficiency, and brining in new technologies to our lab and contributing in an innovative way. The course is delivering more than I ever imagined and I am thrilled to learn the latest deep learning technologies, which will for sure be revolutionary to lots of new fields.

The course will come to an end soon, though my learning in deep learning and computer vision just started. I would like to give special thanks to Thomas Wan, who always supports me and encourages me to keep learning and never give up.

Contributions

The project has referenced and used part of the starter codes from Github for context encoder study. Author would like to thank for the knowledge sharing on Github.

[1] https://github.com/pathak22/context-encoder
[2]https://github.com/BoyuanJiang/context_encoder_pytorch
[3]https://github.com/fbuchert/context-encoder-pytorch

References

[1] Pathak, Deepak et al. “Context Encoders: Feature Learning by Inpainting.” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016): 2536-2544.
[2] Senzer, Zachary et al. “Photobombs begone with Magic Eraser in Google Photo”, https://blog.google/products/photos/magic-eraser/
[3] M. Chen, Y. Ho and S. Wang, "A fast positioning method with pattern tracking for automatic wafer alignment," *2010 3rd International Congress on Image and Signal Processing*, 2010, pp. 1594-1598, doi: 10.1109/CISP.2010.5647710.
[4] Košťák, M.; Slabý, A. Designing a Simple Fiducial Marker for Localization in Spatial Scenes Using Neural Networks. *Sensors* 2021, 21, 5407.
[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton. *ImageNet classification with deep convolutional neural networks. In NIPS, 2012.*
[6] *https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html*
[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D.Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. *Generative adversarial nets*. In NIPS, 2014
[8] Nvidia sim2real platform. https://www.nvidia.com/en-us/on-demand/session/gtcspring21-s31824/
[9] D. Kingma and J. Ba. Adam*: A method for stochastic optimization. ICLR, 2015.*